Enhancing Communication Privacy Using Trustworthy Remote Entities

Andrew James Paverd
Balliol College
University of Oxford

Thesis submitted for the degree of Doctor of Philosophy
Trinity Term 2015

Enhancing Communication Privacy Using Trustworthy Remote Entities

D.Phil. Thesis

Andrew James Paverd
Balliol College

Trinity Term 2015

Abstract

Communication privacy is the property of a communication system that enables two or more distrusting participants to exchange information without compromising their privacy, with respect to internal and external adversaries. It encompasses aspects of anonymous communication as well as data privacy. A real-world example of the need for communication privacy is the smart energy grid, in which networked smart meters frequently measure energy consumption and communicate with grid operators. Privacy concerns arise from the possible inference of sensitive information from these measurements.

Using smart grid communication privacy as a case study, this thesis introduces the concept of the *Trustworthy Remote Entity (TRE)*. The TRE is an intermediary between distrusting participants that performs privacy-enhancing computations on the exchanged information. Unlike cryptographic secure multiparty computation protocols, this approach does not increase participants' computational or communication complexity. In contrast to a *trusted* third party, this *trustworthy* entity uses trusted computing and remote attestation to establish attestation-based trust relationships. As a single-function system, the TRE requires only a minimal software Trusted Computing Base, thus minimizing its attack surface and making it an ideal candidate for security audits. Two research hypotheses are investigated: firstly that the TRE can be realized and used to enhance consumers' privacy in the smart grid, and secondly that the TRE concept can be formalized and used in other application domains.

This thesis confirms both hypotheses and, in doing so, presents five main contributions. Firstly, it proposes a new methodology for modelling and analysing communication privacy terms of unlinkability and undetectability, which is implemented in the CSP process algebra and used to enhance the Casper/FDR analysis tool. Secondly, it presents and analyses a new TRE-based smart grid communication architecture. Thirdly, it compares different TRE system architectures and evaluates a fully functional TRE prototype. Fourthly, it defines a new highly-scalable remote attestation protocol for establishing the TRE's trustworthiness. Finally, it formalizes the fundamental characteristics of the TRE concept and demonstrates how the TRE can be used to enhance communication privacy in location-based services and wireless network roaming.

ad maiorem Dei gloriam

Acknowledgements

Firstly, I thank my supervisors, Andrew Martin and Ian Brown, for their insight and guidance throughout my DPhil. I am very grateful for all our collaborative research efforts and for the many opportunities they have provided.

I also appreciate the various interactions I have had with Cas Cremers, Ivan Flechais, Ivan Martinovic, Kasper Rasmussen, Andrew Simpson, and David Wallom during my time in the Department of Computer Science.

I thank my colleagues and office mates, especially John Lyle, Shamal Faily, Cornelius Namiluko, Anbang Ruan, Tulio de Souza, Justin King-Lacroix, Thomas Gibson-Robinson, Joss Wright, and all the members of the Security Reading Group, for the many stimulating discussions we have had.

I thank my former supervisors, professors and teachers at the University of Cape Town, the University of the Witwatersrand and De La Salle Holy Cross College for laying the foundations for this research endeavour.

I am grateful to those who have supported my DPhil financially: the UK Foreign and Commonwealth Office through a Chevening Scholarship; BT through a research studentship under the *Future Home Networks and Services* project at the Oxford Internet Institute; and Balliol College. I also appreciate the offer of support from Laurie Dippenaar through the FirstRand Laurie Dippenaar Scholarship.

I am grateful to Balliol College for adding immeasurably to my Oxford experience, and especially to Tom Melham, Douglas Dupree, Nicola Trott, and Bruce Kinsey for their support and friendship. In particular, I thank the Praefectus of Holywell Manor for his invaluable guidance.

I have had the privilege of meeting many amazing people during my time in Oxford and so I thank all my friends from the Department of Computer Science, the Balliol College MCR, the Balliol College Chapel Choir, and the Oxford University Catholic Chaplaincy for all the memorable occasions we have shared. Thank you especially for the cheerful breakfasts, long cycle rides, gatherings in the Megaron and, above all, the beautiful music.

Finally, I thank my family, especially my sister and my parents for their unfailing encouragement, inspiration, and wise advice.

Andrew James Paverd
28 September 2015

Details of Publications

The following publications and other dissemination activities have resulted from this research. All conference and workshop papers are full papers in published proceedings whilst the other research outputs do not appear in formal proceedings. Where the research was undertaken by multiple authors, my specific contribution is described. Professor Andrew Martin and Professor Ian Brown are listed as co-authors on various publications in recognition of their supervisory roles in this research. I am the sole author of any text from these publications that has been used in this thesis.

Conference and Workshop Papers

- A. J. Paverd, A. P. Martin, & I. Brown, "Privacy-Enhanced Bi-Directional Communication in the Smart Grid using Trusted Computing," in *Proceedings of the Fifth IEEE International Conference on Smart Grid Communications - SmartGridComm'14*, 2014.

- A. J. Paverd, F. El-Moussa, & I. Brown, "Characteristic-Based Security Analysis for the Personal Network," in *Proceedings of the HomeSys Workshop (co-located with ACM UbiComp'14)*, 2014.

- A. J. Paverd, A. P. Martin, & I. Brown, "Security and Privacy in Smart Grid Demand Response Systems," in *Proceedings of the Second Open EIT ICT Labs Workshop on Smart Grid Security - SmartGridSec'14*, 2014.

- C. Namiluko, A. J. Paverd, & T. De Souza, "Towards Enhancing Web Application Security Using Trusted Execution," in *Proceedings of the Workshop on Web Applications and Secure Hardware - WASH'13*, 2013.
  - I contributed the experimental results for performing cryptographic operations in the TrustZone secure world and the text of Sections 3, 4.2 and 5.1.

- A. J. Paverd, & A. P. Martin, "Hardware Security for Device Authentication in the Smart Grid," in *Proceedings of the First Open EIT ICT Labs Workshop on Smart Grid Security - SmartGridSec'12*, Vol. 7823, pp. 72-84, 2012.

- J. Lyle, A. J. Paverd, J. King-Lacroix, A. Atzeni, H. Virji, I. Flechais, & S. Faily, "Personal PKI for the smart device era," in *Proceedings of the 9th European PKI Workshop (co-located with ESORICS'12)*, 2012.
  - I investigated the related work on certificate and key infrastructures for personal networks and contributed the text of Section 2.

Poster Presentations

- A. J. Paverd, "Poster: Applications of a Trustworthy Remote Entity," Presented at the *University of Oxford Department of Computer Science Student Conference*, 2015.

- A. J. Paverd, "Poster: Enhancing Privacy in Location-Based Services using Trustworthy Remote Entities," Presented at the *7th ACM Conference on Security and Privacy in Wireless and Mobile Networks - WiSec'14*, 2014.

- A. Atamli, A. J. Paverd, & A. P. Martin, "Trustworthy and Secure Service-Oriented Architecture for the Internet of Things," Presented at the *4th International Conference on the Internet of Things - IoT'14*, 2014.
  - I contributed the example use cases and the text of Section 4.

Other Presentations

- A. J. Paverd, "Applications of a Trustworthy Remote Entity (Abstract)," *University of Oxford Department of Computer Science Student Conference*, 2015,
  Best Presentation Award.

- A. J. Paverd, "Security and Privacy in the Smart Energy Grid," *GCHQ Academic Centres of Excellent in Cyber Security Research (ACE-CSR) Conference*, 2013.

- A. Wilde, A. J. Paverd, O. Gasser, M. McGregor, "Perception and Realization of Information Privacy Using Measurements and Ethnography (EINS-PRIME)," *Towards Meaningful Perspectives on Online Consent (workshop at DE2013)*, 2013.
  - I contributed information on technical mechanisms for measuring users' privacy.

- A. J. Paverd, "Student Research Abstract: Trustworthy Remote Entities in the Smart Grid," *ACM Symposium on Applied Computing (SAC'13)*, 2013,
  ACM Student Research Competition Finalist.

- A. J. Paverd, & A. P. Martin, "Hardware Security for Device Authentication in the Smart Grid (Extended Abstract)," *University of Oxford Department of Computer Science Student Conference*, 2012, **Best Abstract Award**.

Contents

Abstract **iii**

Publications **ix**

Contents **xi**

Abbreviations **xviii**

1 Introduction **1**
- 1.1 Communication Privacy in the Smart Energy Grid . . . 3
- 1.2 Trustworthy Remote Entities . . . 4
- 1.3 Research Hypotheses . . . 6
- 1.4 Research Perspectives . . . 6
- 1.5 Contributions and Thesis Structure . . . 7

2 Background: Privacy, Trust and Trusted Computing **11**
- 2.1 Privacy . . . 12
- 2.2 Trust and Trustworthiness . . . 20
- 2.3 Trusted Computing . . . 21
- 2.4 Trusted Execution Environments . . . 25
- 2.5 Trusted Third Parties . . . 27

3 Security and Privacy in the Smart Grid **31**
- 3.1 Smart Grid Architecture . . . 32
- 3.2 Consumers' Perspective of the Smart Grid . . . 34
- 3.3 Information Flows . . . 35
- 3.4 Security and Privacy Threats . . . 39
- 3.5 Existing Solutions . . . 40
- 3.6 Gap Analysis and Summary . . . 51

4 Modelling and Analysing Privacy Properties **53**
- 4.1 Background and Related Work . . . 56
- 4.2 Definitions . . . 57
- 4.3 Adversary Model . . . 62
- 4.4 Integration with Existing Methods . . . 70
- 4.5 Evaluation . . . 75
- 4.6 Summary . . . 84

5 Enhanced Smart Grid Architecture **87**
- 5.1 Baseline System Model and Requirements . . . 88
- 5.2 Privacy-Enhancing Communication Architecture . . . 93

5.3 Implementation Considerations . . . 102
5.4 Evaluation of Architecture . . . 108
5.5 Summary . . . 114

6 TRE Design and Implementation **117**
6.1 Establishing the Trustworthiness of the TRE . . . 119
6.2 Adversary Model and Security Requirements . . . 120
6.3 Abstract Reference TRE Architecture . . . 123
6.4 x86-TPM TRE Architecture . . . 126
6.5 Alternative TRE Architectures . . . 139
6.6 Benchmarking and Evaluation . . . 144
6.7 Using the TRE in the Smart Grid . . . 152
6.8 Summary . . . 154

7 TRE Remote Attestation **157**
7.1 Adversary Model and Security Requirements . . . 159
7.2 Performance Challenges and Requirements . . . 160
7.3 Current Remote Attestation Protocols . . . 161
7.4 Final State Attestation Protocol . . . 170
7.5 Evaluation . . . 177
7.6 Verifying the Attestation . . . 184
7.7 Summary . . . 187

8 Formalization and Application to Other Domains **189**
8.1 Framework for Enhancing Communication Privacy using the TRE . . . 190
8.2 Location-Based Services . . . 195
8.3 Wireless Network Roaming . . . 200
8.4 Secure Multiparty Computation using the TRE . . . 204
8.5 Summary . . . 208

9 Conclusion and Future Work **211**
9.1 Research Context and Hypotheses . . . 212
9.2 Main Contributions . . . 214
9.3 Future Work . . . 217
9.4 Summary . . . 219

Bibliography **221**

Appendix A Casper-Privacy Protocol Models **245**
A.1 TAS3 Attribute Aggregation Protocol . . . 246
A.2 Unlinkability of RFID e-Passports . . . 248
A.3 Unlinkability of RFID e-Passports - Error Condition . . . 250
A.4 Protecting location privacy using k-anonymity . . . 252

A.5 Smart Meter Anonymization through Pseudonyms 254
A.6 Pseudonymous Smart Metering without a TTP 256
A.7 Smart Meter Anonymization through Group Identifiers 258
A.8 OpenADR Standard 260
A.9 Privacy-Enhancing Network Monitoring Protocol 262
A.10 Privacy-Enhancing Billing Protocol 264
A.11 Privacy-Enhancing Demand Bidding Protocol 266

Appendix B TCSP# System Models **269**
B.1 Attestation Scenario 270
B.2 Nonce-Challenge Attestation Protocol 272
B.3 Global Timestamp Attestation Protocol 273
B.4 Final State Attestation Protocol 274

List of Figures

1.1 Structure of the thesis and relationships between chapters. 8

3.1 Widely cited NIST overview of smart grid entities [198]. 32
3.2 Planned UK smart grid model (adapted from [270]). 33
3.3 Abstract model of a pseudonymization scheme 41
3.4 Abstract model of a temporal aggregation scheme 43
3.5 Abstract model of a spatial aggregation scheme 45

4.1 CSP implementation of the deductive system 74

5.1 Overview of the privacy-enhancing communication architecture 94
5.2 Spatial aggregation of consumption measurements 95
5.3 Temporal aggregation of billing information 98
5.4 Anonymization and temporal aggregation for demand bidding 100

6.1 Relationships between requirements, architectures and prototypes 118
6.2 Abstract reference architecture of the TRE 124
6.3 Overview of the x86-TPM TRE architecture 128
6.4 Overview of the x86-SGX TRE architecture 143
6.5 Relative sizes of TRE subsystem by lines of code 146
6.6 Computational performance of signature and verification operations 150
6.7 Overall performance of DLMS-COSEM operations 151

7.1 Nonce-challenge attestation protocol 162
7.2 Masquerading attack against a nonce-challenge attestation protocol 163
7.3 TLS handshake augmented with the Final State Attestation (FSA) protocol 172
7.4 Final State Attestation Certificate $cert_{FSA}$ 173
7.5 Common scenario for remote attestation protocol analysis 179
7.6 TEE-based mechanism for protecting an authentication key 186

8.1 Types of entities in the TRE formalization 191
8.2 Generalized model of a location-based service (LBS) 195
8.3 Communication architecture using the TRE to enhance LBS privacy 199
8.4 Privacy-enhancing wireless network roaming architecture using the TRE . . 203

8.5 Idealized setting of a cryptographic SMC protocol 205
8.6 Real setting of a cryptographic SMC protocol 206
8.7 Secure multiparty computation using the TRE 207

List of Tables

4.1 Components and notation used in the model 63
4.2 New Casper-Privacy Specifications 71
4.3 Enhanced Casper-Privacy Specifications 72

5.1 Privacy attacks and mitigation strategies in the bidding protocol 103
5.2 Examples of DLMS-COSEM commands for the new protocols 105

6.1 TRE memory layout for a platform with 4 GB physical memory 136
6.2 Lines of code in the TCB of the x86-TPM TRE prototype 146

7.1 Structure of the FSA TLS extension 174

Abbreviations

ACM	Authenticated Code Module
AIK	Attestation Identity Key (TPM)
AMI	Advanced Metering Infrastructure
AR	Attestation Requirement
CSP	Communicating Sequential Processes
DAA	Direct Anonymous Attestation
DCC	Data Communications Company (GB)
DECC	Department of Energy and Climate Change (GB)
DER	Distributed Energy Resource
DMA	Direct Memory Access
DNO	Distribution Network Operator
DoE	Department of Energy (US)
DR	Demand Response
DRBG	Deterministic Random Byte Generator
DRTM	Dynamic Root of Trust for Measurement
DSM	Demand Side Manager
EPID	Enhanced Privacy ID (Intel)
FSA	Final State Attestation
GPL	GNU General Public License
GPRS	General Packet Radio Service
HBC	Honest-But-Curious (adversary)
HEMS	Home Energy Management System
HN	Home Network
IC	Integrated Circuit
IHD	In-Home Device
IMA	Integrity Measurement Architecture
LBS	Location-Based Service
LPC	Low Pin Count (Bus)
MBR	Master Boot Record
MLE	Measured Launch Environment
MNO	Mobile Network Operator
NILM	Non-Invasive Load Monitoring

PCR	Platform Configuration Register (TPM)
PEV	Plug-in Electric Vehicle
PFS	Perfect Forward Secrecy
PRNG	Pseudo-Random Number Generator
RNG	Random Number Generator
RoT	Root of Trust
RTM	Root of Trust for Measurement
RTR	Root of Trust for Reporting
RTS	Root of Trust for Storage
SGX	Software Guard Extensions (Intel)
SKAE	Subject Key Attestation Evidence
SMC	Secure Multiparty Computation
SML	Secure Measurement Log
SSH	Secure Shell
SVM	Secure Virtual Machine (AMD)
SRK	Storage Root Key (TPM)
TC	Trusted Computing
TCB	Trusted Computing Base
TCG	Trusted Computing Group
TCSP#	Trusted CSP#
TLS	Transport Layer Security
TPM	Trusted Platform Module
TRE	Trustworthy Remote Entity
TSR	TRE Security Requirement
TTP	Trusted Third Party
TXT	Trusted Execution Technology (Intel)
UEFI	Unified Extensible Firmware Interface
VM	Virtual Machine
VMM	Virtual Machine Monitor
VN	Visited Network

Chapter 1

Introduction

1.1 Communication Privacy in the Smart Energy Grid 3
1.2 Trustworthy Remote Entities 4
1.3 Research Hypotheses 6
1.4 Research Perspectives 6
1.5 Contributions and Thesis Structure 7

Communication is an essential aspect of our modern information society. Digital communication technologies and the Internet have led to a significant increase in communication between computer systems. At a fundamental level, *communication* refers to any sharing or exchange of information between two or more communicating entities. From the perspective of each communicating entity, this exchange can be either uni-directional (i.e. either sending or receiving information) or bi-directional. Communication takes place in order to achieve some objective and is often necessary as part of a larger system. Intrinsically linked to communication is the concept of *communication privacy*. The ITU-T defines *privacy* as: *The right of individuals to control or influence what personal information related to them may be collected, managed, retained, accessed, and used or distributed* [136].

Communication privacy therefore refers to individuals' ability to control what personal information related to them can be collected or inferred as a result of their communication. Communication privacy is threatened by two types of adversaries:

The first is an external adversary who is not a legitimate participant in the communication exchange. The classical security property of *secrecy* or *confidentiality* aims to prevent external adversaries from learning the content of exchanged messages. Privacy properties such as *sender anonymity* and *recipient anonymity* aim to prevent a message being linked to an identifiable sender or recipient. For example, users communicating over the Internet aim to keep their communication secret from external adversaries. In some cases, users may wish to prevent external adversaries from linking messages to the real sender or recipient to prevent the adversaries from learning or inferring private information.

The second is an internal adversary who is a legitimate participant in the communication exchange. In some cases there is a requirement to communicate with an entity who is not fully trusted, or even for mutually distrusting entities to communicate. In these circumstances, each entity aims to achieve its communication objective without revealing private information to any untrusted entities. For example, an idealized voting system can be represented as a set of communication exchanges between voters and the adjudicator. The adjudicator must be convinced that each vote comes from an authorized voter, and that each voter only casts a single vote. Ideally, the voter's choice is private information, and thus it should be impossible for even an untrusted adjudicator to link this to the voter's identity. Although real-world voting systems do not achieve absolute privacy, they still aim to protect voters' privacy as far as possible. In some real-world voting systems, such as that used for elections in the UK, there is the capability to link votes to voter numbers, and thus to individual voters, in order to investigate allegations of fraud. However, this capability is carefully controlled, both procedurally and practically, such that it can only be used by authorized entities under appropriate circumstances. Other real-world voting systems, such as that used for elections in South Africa, deliberately omit this capability [68], making it significantly more difficult to link votes to individual voters.

There have been various efforts to formalize these notions of privacy, as explained in Chapter 2 and Chapter 4. Currently one of the best real-world examples of the need for

communication privacy is the smart energy grid, as described in the next section.

1.1 Communication Privacy in the Smart Energy Grid

The *smart energy grid*, or *smart grid*, is widely acknowledged to be the future of public energy infrastructure. Several national governments have mandated the design and implementation of smart grids in the electricity sector and are likely to undertake widespread implementation within the next five years. At a fundamental level, the smart grid involves the use of modern computing and communication technology to enhance various aspects of the energy distribution infrastructure. The overall objectives are to optimize the use of this infrastructure, improve energy efficiency and facilitate the inclusion of renewable energy sources. From a technical perspective, the term *smart grid* is an umbrella term that encompasses a number of different technologies and systems. Various technical bodies such as the National Institute of Standards and Technology (NIST) [198] and IEEE [130] are working towards standardization to ensure interoperability between these systems. Although the present view of the smart grid is largely based on existing technologies, there are numerous ongoing research activities that are continually influencing this field.

From the consumers' perspective, the most visible aspect of the smart grid is the upgrading of existing gas and electricity meters to smart energy meters. Smart meters are energy measurement devices with communication capabilities that enable them to automatically communicate energy measurements to remote entities such as the energy supplier or Distribution Network Operator (DNO). The combination of smart meters and the associated systems for receiving and processing these measurements is referred to as the Advanced Metering Infrastructure (AMI). In comparison to previous generations of energy meters that had to be read manually at the end of each billing period (e.g. monthly or quarterly, depending on the country), smart meters can record measurements over significantly shorter time intervals. In current deployment plans, each smart meter will have the capability to record a measurement every 15 or 30 minutes. Many smart meters are capable of recording measurements over even shorter time intervals, so it is possible that this measurement interval might decrease further to achieve higher temporal resolution. The measurements are communicated from the smart meter to the energy supplier or DNO according to a defined schedule. For example, in the UK the default option is for the energy supplier to receive daily readings from every consumer, but consumers can opt-in to provide measurements at 30 minute intervals [92].

The information provided by smart meters enables new approaches for optimizing the energy distribution infrastructure. The frequent measurements can be used by DNOs to monitor the status of different sectors and optimize the flow of energy in the distribution network. Similarly, energy suppliers can use this information to forecast supply and demand and to manage these using dynamic pricing or other Demand Response (DR) strategies. This functionality will become increasingly important for DNOs and energy suppliers as more time-variable renewable energy sources are integrated into the grid. It

will also be particularly important as consumers gain the ability to store or generate energy themselves and feed this back into the grid.

However, this exchange of information in the smart grid also leads to significant privacy concerns. These concerns arise primarily from the realization that energy consumption measurements with a sufficiently high temporal granularity can be used to infer additional information about the consumers which may be considered private. For example, the frequent measurements from smart meters can be used to determine whether a particular house is unoccupied at specific times or to observe the behavioural patterns of the occupants. Techniques such as Non-Invasive Load Monitoring (NILM) [125, 124, 86, 285] enable the identification of individual energy-consuming appliances in a home based on frequent measurements of the home's total energy usage. In some cases, these privacy concerns have delayed or even halted the deployment of smart meters and hence affected the realization of the overall smart grid. One of the most widely cited examples of this is the 2009 decision in the Netherlands to halt the mandatory deployment of smart meters based on a report from the University of Tilburg explaining these privacy concerns [72].

In addition to the concerns relating to smart meters, privacy concerns also arise from other aspects of the smart grid such as bi-directional Demand Response (DR) protocols. For example, in certain DR protocols, consumers communicate with external entities such as the Demand Side Manager (DSM) to coordinate reductions in consumption during peak periods. The information exchanged in these DR protocols could also be used to infer private information as discussed in Chapter 3.

Various approaches and solutions have been proposed to mitigate these privacy concerns. The majority of these are based on the observation that the information exchanged in smart grid communication can be modified in various ways to enhance consumers' privacy without diminishing the intended functionality of the system. These existing approaches are discussed in Chapter 3. However, it is often the case that previous approaches address only one aspect of the problem and cannot be used as an overall solution. In particular, previous approaches have not addressed the problem of privacy in bi-directional communication between consumers and other smart grid entities. Since bi-directional communication is only required for more advanced smart grid functionality, this is not the most immediate concern for the initial phases of the smart grid. However, this capability will be required to support the full potential of the smart grid and given the privacy concerns that have already been raised, it is critical to consider consumers' privacy in this type of communication. Using the smart grid as the primary case study, this thesis presents and investigates a new approach for enhancing communication privacy.

1.2 Trustworthy Remote Entities

This thesis introduces the concept of a *Trustworthy Remote Entity (TRE)* as an architectural component for enhancing privacy in communication exchanges. The TRE is a computational and communication system that is situated as an intermediary between

two or more communicating participants but is not controlled by any of the participants. The role of the TRE is to perform privacy-enhancing operations on the communicated information. The TRE exhibits two fundamental characteristics: Firstly, the TRE is trusted by all other participants in the communication architecture. Even though the participants may not trust one another, or may even be mutually distrusting, they each individually trust the TRE. Secondly, the TRE provides strong technical guarantees of its trustworthiness. This means that the communicating participants do not blindly trust the TRE as they would a Trusted Third Party (TTP) but instead use technical mechanisms to verify the state of the TRE and thus establish its trustworthiness.

In the smart grid, TREs can be used to enhance communication privacy by providing a layer of indirection in the communication between consumers and grid operators. For example, a TRE can be used to aggregate the frequent energy consumption measurements from multiple smart meters and provide the results to the DNO in a privacy-preserving manner. The consumers and the DNO independently establish trust relationships with the TRE: the consumers trust the TRE to protect their privacy whilst the DNO trusts the TRE to authenticate the consumers and perform the aggregation correctly. Even though the consumers and the DNO may not trust one another, they can still communicate in a privacy-preserving manner by using a mutually trusted TRE.

In order to provide guarantees of its trustworthiness, the TRE uses technologies and approaches from the field of Trusted Computing (TC). Further background information about TC is provided in Chapter 2. In particular, *remote attestation* has been shown to be a promising approach for establishing the trustworthiness of remote systems. However, this technique is seldom used in real-world systems due to scalability challenges. Although remote attestation protocols can provide an integrity-protected log of all software that has been executed on a platform, the verifier is still required to make a trust decision taking into account each individual software component. In order to use attestation in modern general-purpose systems such as PCs, the verifier is therefore required to maintain an extensive and frequently changing whitelist of trusted software components. Previous research has investigated the use of attestation in more specialized systems such as web servers [174]. Whilst the effort required for this is significantly lower than in the general-purpose case, it is still not practical to fully establish the trustworthiness of the system using only this method [174]. In contrast, the TRE is a single-function system that performs relatively simple information processing operations and can thus be realized using only a minimal software Trusted Computing Base (TCB). Minimizing the TCB reduces the likelihood of security vulnerabilities, reduces the TRE's attack surface, and makes the TRE more amenable to security audits and possibly even formal verification. This therefore is reason to believe that remote attestation can be used as a mechanism for establishing the trustworthiness of the TRE. By analogy, the TRE can be envisioned as a mechanism operating inside a glass case; all parties can observe and verify the correct operation of the mechanism in great detail but none can interfere with its operation.

1.3 Research Hypotheses

Having identified the main privacy challenges in the smart grid, this thesis proposes a solution based on the TRE concept and describes the investigation of two primary research hypotheses:

1. *In the context of the smart grid, the concept of a Trustworthy Remote Entity (TRE) can be realized and used to enhance the privacy of consumers whilst maintaining the primary functionality of the system.*

2. *The concept of a TRE can be formalized and used to enhance communication privacy in other application domains.*

The investigation of these hypotheses need not only result in a binary outcome. In addition to verifying or falsifying these hypotheses, this thesis aims to determine the *extent* to which each hypothesis is true as well as to identify any constraints or necessary assumptions that exist. The above hypotheses can therefore be restated as research questions such as:

a In the context of the smart grid, to what extent can the TRE be realized using current technologies?

b Which technologies are the most suitable for the realization of the TRE?

c To what extent does the TRE enhance privacy in the smart grid?

d Are there any trade-offs between privacy and functionality in this solution?

e In which types of application domains can the TRE be used to enhance privacy?

These research questions arise from considering both a top-down and a bottom-up perspective as explained in the next section.

1.4 Research Perspectives

This thesis considers both the top-down and the bottom-up perspectives of this research endeavour. The top-down perspective begins with a specific problem in the application domain, namely enhancing communication privacy in the smart grid. This leads to a set of functional requirements for the solution, such as the requirements of supporting bi-directional communication and providing guarantees of trustworthiness. The research focus is then the extent to which these requirements can be met using specific technical solutions and the selection of the underlying technologies. The first two research questions (*a* and *b* above) illustrate the top-down perspective.

Conversely, the bottom-up perspective begins with a specific technological capability, such as remote attestation. This choice leads to a set of constraints on the system, such as the constraint that the TRE can only perform relatively simple processing since it must

have a minimal TCB in order to facilitate remote attestation. The research focus is then the extent to which a system satisfying these constraints can provide useful functionality and the extent to which this functionality solves a problem in a specific application domain. The final two research questions (d and e above) arise from the bottom-up perspective.

Ultimately, the final solution can be considered to be the superposition of these two perspectives. Each perspective informs and contributes to the other. The top-down perspective can only be successful when it is supported by an appropriate technical solution and set of underlying technologies whilst the bottom up perspective can only be useful given an appropriate problem from a relevant application domain. The third research question (c above) is an example of this in that it investigates the extent to which the specific technology can be used to solve a particular problem.

The two primary research hypotheses presented in the previous section each span both perspectives. However the first hypothesis concerning the smart grid is more closely related to the top-down perspective whilst the second hypothesis concerning other application domains tends towards the bottom-up perspective. Through the investigation of these research hypotheses and questions, this thesis presents a number of novel contributions.

1.5 Contributions and Thesis Structure

A graphical representation of the structure of this thesis is shown in Figure 1.1.

Chapter 2 presents background information on the two major aspects of this thesis: privacy and trusted computing. It describes the current state-of-the-art formalizations, approaches and technologies in each field.

Chapter 3 builds on this foundation by applying these concepts to a real-world case study of the smart grid. This chapter also introduces the main privacy concerns in the smart grid. Some of these concerns have been identified in the literature whilst some are new contributions from this thesis. This chapter concludes with a review of existing solutions and a gap analysis that identifies the gap addressed by this research.

The primary contribution of this thesis is the development and investigation of the concept of the Trustworthy Remote Entity, and its use in enhancing communication privacy in real-world application domains such as the smart grid. Five distinct aspects of this contribution are presented in the subsequent chapters, which form the core of this thesis.

Firstly, a new methodology to systematically model and analyse the relevant privacy properties in communication exchanges is presented in Chapter 4. The properties of unlinkability and undetectability are modelled in the process algebra of Communicating Sequential Processes (CSP) and analysed using the FDR2 model checker. This methodology has been added to the *Casper/FDR* tool in order to allow concurrent analysis of security properties and privacy properties. Various protocols have been analysed in order to test this methodology and the associated software tools.

Secondly, a communication architecture and a set of communication protocols for enhancing privacy in the smart grid is described in Chapter 5. The key element of this

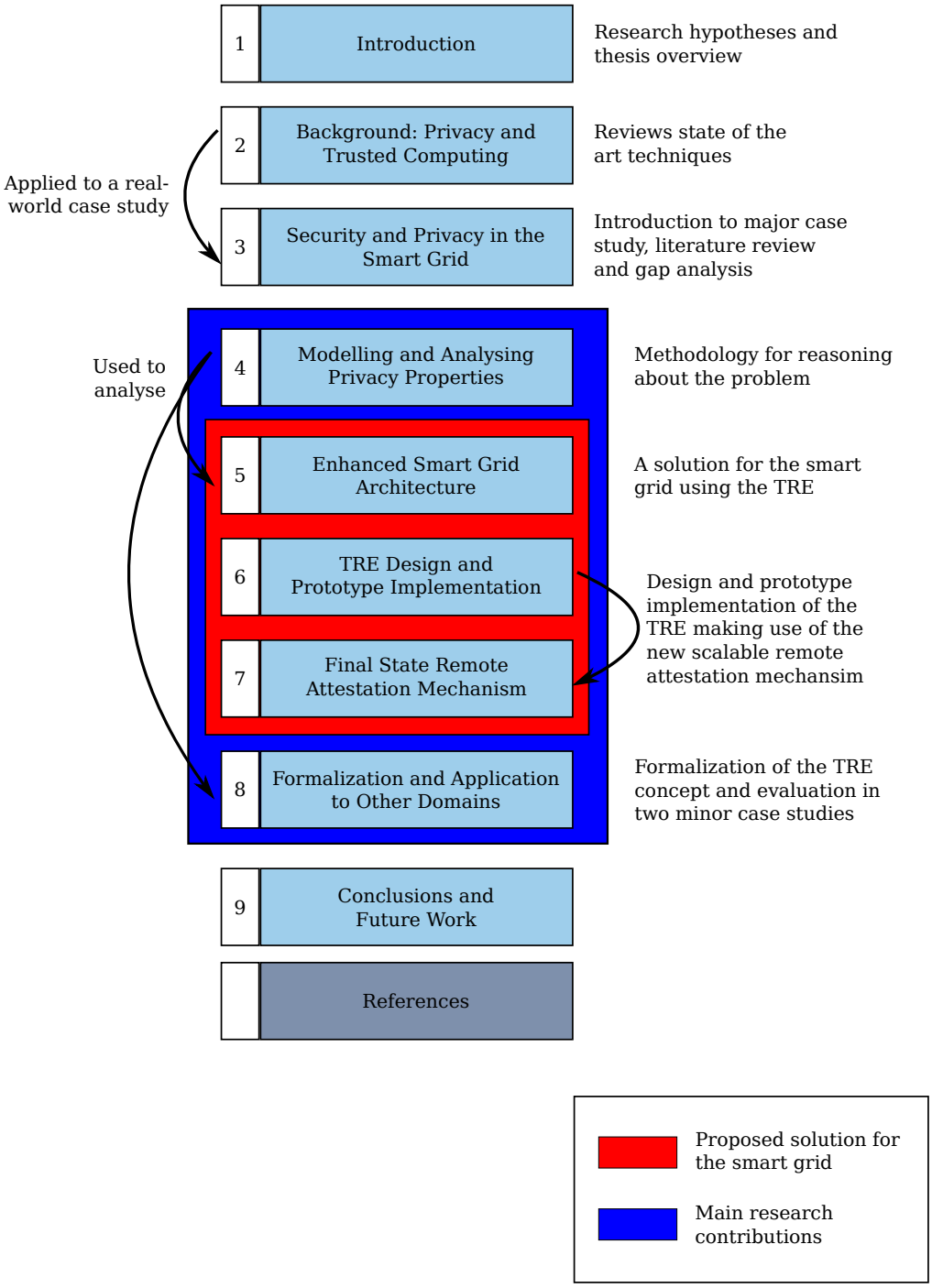

Figure 1.1: Structure of the thesis and relationships between chapters.

architecture is the TRE. Unlike existing approaches, this architecture aims to mitigate the privacy concerns in all three primary information flows in the smart grid, including the bi-directional demand response information flow. The communication protocols that make up this architecture are analysed in terms of their privacy and security properties using the methodology described above.

The third contribution is the design, implementation and evaluation of a prototype TRE system for use in the smart energy grid as described in Chapter 6. This chapter defines the overall design requirements and evaluation metrics for the TRE and presents an abstract reference architecture for its implementation. Based on this, a concrete TRE architecture is proposed and implemented as a fully-functional prototype on an x86 PC platform. This prototype implementation is then evaluated in terms of the defined requirements and, in particular, the size of the prototype's TCB is compared to that of other architectures and systems.

One of the primary requirements of the TRE is its ability to provide guarantees of its trustworthiness to relying parties through remote attestation. However, most current attestation mechanisms are designed to attest the state of client systems and are thus used relatively infrequently and with relatively few relying parties. These existing mechanisms are therefore not sufficiently scalable for use in the TRE, which must provide frequent attestation guarantees to a significantly larger set of relying parties. To overcome this challenge, the fourth contribution of this thesis is a novel highly-scalable remote attestation mechanism for use in systems such as the TRE. This mechanism is incorporated into the prototype implementation described in Chapter 6 but since it can also be used in other types of systems, its design, implementation and evaluation are presented as a stand-alone contribution in Chapter 7.

As the fifth contribution, Chapter 8 describes the formalization of the TRE concept. This chapter presents a framework that enables the use of the TRE in other application domains. This framework is evaluated using two smaller case studies from different application domains. The first investigates the use of the TRE in enhancing privacy in Location-Based Services (LBS) and the second focuses on the same challenge in the context of wireless network roaming.

The main points of each chapter are summarized at the end of the respective chapter and the overall conclusions of the thesis are presented in Chapter 9. Building on the contributions of this thesis, this chapter also presents recommendations for future work in enhancing communication privacy using Trustworthy Remote Entities.

Chapter 2

Background: Privacy, Trust and Trusted Computing

2.1 Privacy 12
2.2 Trust and Trustworthiness 20
2.3 Trusted Computing 21
2.4 Trusted Execution Environments 25
2.5 Trusted Third Parties 27

This chapter provides background information about the major themes that are relevant to this thesis. Section 2.1 gives an overview of recent privacy concepts: k-anonymity, ℓ-diversity, t-closeness, differential privacy, and Secure Multiparty Computation (SMC). Section 2.2 then provides background on the notions of trust and trustworthiness and is followed by explanations of current mechanisms for establishing trust in computer systems, including Trusted Computing (TC) and Trusted Execution Environments. The chapter concludes by discussing the concept of a Trusted Third Party (TTP). The contents of this chapter are based on existing literature that has been selected and summarized to provide background to the research presented in this thesis. Although this chapter contains the majority of the background information included in this thesis, the subsequent chapters also include discussions of related work that is of specific relevance to each chapter.

2.1 Privacy

Although the term *privacy* has many different meanings, this thesis focusses on the technical aspects of privacy that are relevant to communication systems. When considering privacy in communication, the focus is often placed on anonymous communication channels, including anonymity networks such as TOR [84] and anonymous remailers like Mixminion [76]. Anonymous communication channels aim to solve the problem of *how* information can be transferred without revealing the identities of the communicating parties. However, these systems usually focus only on transporting the data anonymously and do not deal with the equally important question of *what* information can be communicated without compromising privacy. For example, from the perspective of an adversarial recipient, the anonymity properties provided by TOR are trivially undermined by the user sending some type of identifying information. In addition to privacy requirements, some systems also have specific security requirements that cannot be met using only anonymous communication channels. For example, it would not be possible to protect privacy in the smart grid by simply having smart meters send their measurements via an anonymity network because the recipients could not verify that these measurements were sent by legitimate smart meters. This type of false data injection attack and other threats to security and privacy in the smart grid are discussed in the next chapter. Communication privacy must therefore consider both *how* the communication takes place as well as *what* information is communicated. To achieve this, communication privacy brings together aspects of anonymous communication as well as data privacy. The following subsections provide background on important concepts from the field of data privacy that are of specific relevance to communication privacy.

2.1.1 k-Anonymity

Given a dataset containing information about individuals (i.e. person-specific data), a frequent question is how to share this data without compromising the privacy of the included individuals. To address this challenge, Sweeney [249] proposed a formal protection model, named k-anonymity, that aims to provide a degree of privacy in this scenario.

Definition 1 (k-anonymity [249])**.** A released dataset provides k-anonymity if the information about each individual in the dataset is indistinguishable from that of at least $k-1$ other individuals also included in the dataset, with respect to their identifying information.

The value of k is a tunable privacy parameter that defines the size of the *anonymity set* and should therefore be defined based on the context of the dataset. Stated another way, k-anonymity aims to ensure that no piece of information from the dataset can be linked to fewer than k individuals.

If a person-specific dataset is represented as a table, each row corresponds to an individual and no two rows correspond to the same individual. The columns represent attributes of the individuals. A naive approach to anonymizing such a dataset would be to simply remove the unique identifying attributes for each row (e.g. names and identity numbers). However, this ignores the possibility of de-anonymizing the dataset by linking it to other datasets based on so-called *quasi-identifiers* [73]. Quasi-identifiers are sets of attributes that, on their own would not identify individuals, but when considered together can uniquely identify the individuals in the dataset. For example, Sweeney showed that 87% of the population of the United States could likely be identified based on the quasi-identifier of 5-digit ZIP code, gender and date of birth, based on data from the 1990 census [250]. Using data from the 2000 census, Golle confirmed this overall result, although at the lower percentage of 63% of the US population [120]. It has been shown that quasi-identifiers can also arise from other attributes such as frequently-visited locations [284, 193] and credit-card transaction history [192]. To reduce the possibility of datasets being de-anonymized in this way, k-anonymity requires that, for each quasi-identifier value, there are at least $k-1$ other rows in the table containing the same value.

Common mechanisms for achieving k-anonymity involve suppressing certain attributes (e.g. unique identifiers) and generalizing other attributes (e.g. specifying ranges rather than exact values in order to meet the k-anonymity criterion). However, it has been proved that optimal k-anonymization is an *NP*-hard problem [186].

2.1.2 ℓ-Diversity

However, k-anonymity is not always sufficient to protect privacy. Machanavajjhala et al. [175] presented two attacks against k-anonymity, namely a *homogeneity attack* and a *background knowledge attack*.

In the homogeneity attack [175], it is assumed that the adversary has access to a k-anonymized dataset and knows that a target individual is a member of a specific anonymity

set (i.e. the individual has been narrowed down to one of k possible rows). However, if all members of that particular anonymity set have the same value for a sensitive non-identifying attribute, the adversary learns the value of that attribute for the target individual. For example, in an employee salary dataset, if the adversary knows the quasi-identifier of the target individual, and all individuals in the table who share that particular quasi-identifier have the same salary, then the adversary learns this piece of information about the target, even though the individual has not been de-anonymized. Furthermore, even if the sensitive attributes are not completely homogeneous, there might still be a high percentage of the rows that share the same value, which might allow the adversary to infer this value for the target individual with high probability.

In the background knowledge attack [175], the adversary again has access to a correctly k-anonymized dataset, but also has some additional background knowledge relevant to the individuals in the dataset. Ordinarily, this background knowledge would not be sufficient to de-anonymize individuals, but if the sensitive attributes are not sufficiently diverse within each anonymity set, it could allow the attacker to infer sensitive information about the target individual. Continuing the previous employee salary dataset example, it could be the case that the target individual's anonymity set contains two different salary levels, one high and one low. The adversary might have background information, such as the average salary for employees in a particular role, that could be used to eliminate one of the two salary options and thus reveal the true value.

These two attacks arise from a lack of diversity of the sensitive attributes within a specific anonymity set. To overcome this challenge, Machanavajjhala et al. [175] proposed the notion of ℓ-diversity as follows:

Definition 2 (The ℓ-diversity Principle [175])**.** *An anonymity set is ℓ-diverse if it contains at least ℓ "well-represented" values for the sensitive attribute. A table is ℓ-diverse if every anonymity set is ℓ-diverse.*

Machanavajjhala et al. [175] have presented three different interpretations of the term "well-represented": the simplest approach, called distinct ℓ-diversity requires that there are at least $\ell \geq 2$ distinct values for the sensitive attribute in each anonymity set. However, this could still allow the adversary to infer information and thus the notions of Entropy ℓ-diversity and Recursive (c, ℓ)-diversity were introduced to mitigate this by bounding the frequency of sensitive values [175].

An ℓ-diverse table mitigates against the homogeneity attack because the adversary cannot determine or infer (with sufficiently high probability) the value of the sensitive attribute, even if the target individual's anonymity set is known. It is not possible to completely mitigate the background knowledge attack because the data publisher cannot know what background information the adversary possesses. However, for this attack to succeed, the adversary must be able to eliminate $\ell - 1$ values of the sensitive attribute for the target individual. Therefore the privacy parameter ℓ can be tuned to minimize the likelihood of this attack based on the context of the dataset.

2.1.3 t-Closeness

Shortly after the proposal of ℓ-diversity, Li et al. [163] presented two attacks against this improved technique: a *skewness attack* and a *similarity attack*.

The skewness attack arises if the overall distribution of the sensitive value in the table is skewed. If the target individual's anonymity set is known, the adversary could infer information about this individual if the distribution of the sensitive value in that set is significantly different from the overall distribution. For example, in the employee salary dataset, if a particular salary value is uncommon overall but occurs very frequently in a the target's anonymity set, it can be inferred that the probability of the target having this salary is significantly higher than the average for the dataset.

The similarity attack relies on the same principle as the homogeneity attack in the previous section but takes advantage of any semantic similarity between the values of the sensitive attribute, even if these are distinct and ℓ-diverse. For example, in the employee salary dataset, there might be a sufficient diversity of salary values for the target individual's quasi-identifier (i.e. correct ℓ-diversity). However, if all of these values are in a similar range, the target individual's salary will be within this range, which could differ significantly from the overall range of values in the table. Both of these attacks are possible even if the dataset correctly exhibits both k-anonymity and ℓ-diversity.

To mitigate these new attacks, Li et al. [163] proposed the new notion of t-closeness, which they define as follows:

Definition 3 (t-closeness [163])**.** *An equivalence class is said to have t-closeness if the distance between the distribution of a sensitive attribute in this class and the distribution of the attribute in the whole table is no more than a threshold t. A table is said to have t-closeness if all equivalence classes have t-closeness.*

Li et al. [163] describe the principle of t-closeness in terms of the amount of information about an individual that can be learnt from a dataset in which the individual is included. In the most extreme case of k-anonymity, all quasi-identifiers might be suppressed or generalized to the same value, making the entire table a single anonymity set. In this case, the only information that can be learnt from the table is the statistical distribution of the values of the sensitive attributes. This might reveal some statistical information about the target individual, but this information would still have been revealed irrespective of whether or not the target individual was included in the dataset. Since this corresponds to the usefulness of the dataset, the objective is to maximize this increase in information. However, if the table contains multiple anonymity sets (even if each contains at least k individuals and is ℓ-diverse), the distributions of each anonymity set might differ from that of the overall table, allowing some inference about the individuals in different anonymity sets. This represents a compromise of privacy, which t-closeness aims to minimize by requiring the distributions of the sensitive values in each anonymity set to be relatively close to the overall distribution in the table.

2.1.4 Differential Privacy

The preceding sections demonstrate the difficulty of protecting privacy in released datasets. One of the main challenges in this context will always be the possibility that the adversary has background information or auxiliary knowledge that can be used to de-anonymize the dataset or infer sensitive attributes of specific participants. Furthermore, since the dataset is released, the adversary can store the released version and make use of future knowledge, either from subsequent data releases or from completely different sources, to extract more information from the stored dataset.

In contrast to the three approaches discussed above, the concept of *differential privacy* presented by Dwork [87] and others [88] adopts a fundamentally different approach to protecting privacy in datasets. Instead of releasing an anonymized or modified version of the dataset, differential privacy considers a dataset that is held by a trusted entity who will answer statistical queries about the dataset according to a privacy-preserving algorithm. The main difference between this interactive setting, compared to the non-interactive setting of the released datasets, is that the trusted entity can decide which queries to answer and how they should be answered.

In the context of protecting privacy in statistical databases, Dalenius [74] proposed that if statistical information S makes it possible to determine a sensitive value more accurately than is possible without S, then a disclosure has taken place. In the ideal case, having access to statistical results from a dataset should not make it possible to learn any private information about individuals in the dataset that could not be learned without access to the results. However, Dwork [87] presented an impossibility result, showing that it is not possible to fully achieve this ideal situation. Instead, the overall objective of differential privacy is to quantify and control the degree to which an individual's privacy is diminished by participating in a dataset. In order to represent this, the formalization of differential privacy is based on the difference between whether a single individual either participates or does not participate in a dataset. Stated another way, it considers the difference between two datasets that differ by exactly one individual.

The applicability of this formulation can be seen directly through attacks such as the set-difference attack on wireless sensor networks as described by de Souza et al. [241]. In a wireless sensor network, it might be desirable to protect the measurements generated by individual sensor nodes, for example, if nodes correspond to individual users. It might appear that this could be achieved by aggregating the data from a sufficiently large group of nodes and only publishing the result. However, if the network is shared between multiple applications, two applications could request aggregate values for groups that differ by exactly one node. The difference between these two results is the value contributed by the target node [241]. This type of set-difference attack is also applicable in other application domains, such as the smart grid. As discussed in the next chapter, there are proposals to protect consumers' privacy in the smart grid by aggregating measurements over multiple consumers. However, by manipulating these aggregation groups as above,

an adversary could still obtain measurement data from a single smart meter. Dwork [87] defines differential privacy as follows:

Definition 4 (ϵ-differential privacy [87])**.** *A randomized function $\mathcal{K}$ gives ϵ-differential privacy if for all data sets D_1 and D_2 differing on at most one element, and all $S \subseteq Range(\mathcal{K})$:*

$$\Pr[\mathcal{K}(D_1) \in \mathcal{S}] \leq \exp(\epsilon) \times \Pr[\mathcal{K}(D_2) \in \mathcal{S}]$$

In any ϵ-differentially private mechanism, the parameter ϵ bounds the amount by which a single element (e.g. a single individual) can influence the result of a particular query. As the value of ϵ decreases, the contribution of each individual also decreases, which improves privacy but may have a negative impact on the usefulness of the dataset. One approach for constructing such a mechanism is to add appropriately chosen random noise to the actual answer of each query. In order to correctly calibrate this noise for a particular query, the maximum difference that a single element could make to the result of that query must be known. For a query function f, this quantity is called the *sensitivity* Δf of the query function and is calculated as:

Definition 5 (Sensitivity [87])**.** *For all D_1 and D_2 differing on at most one element:*

$$\Delta f = \max_{D_1, D_2} ||f(D_1) - f(D_2)||$$

It should be noted that the sensitivity is a property of the query function alone and is independent of the dataset. Dwork [87] shows how exponentially distributed (i.e. according to a Laplace distribution) random noise is suitable for this mechanism. The privacy mechanism, denoted as $\mathcal{K}_f$ for a query function f, computes the exact answer $f(X)$ and then adds random noise with a scaled symmetric exponential distribution with variance σ, according to the density function:

$$\Pr[\mathcal{K}_f(X) = a] \propto \exp(-||f(X) - a||/\sigma)$$

Dwork [87] then shows that this results in the following relationship for all values of r:

$$\Pr[\mathcal{K}_f(D_1) = r] \leq \Pr[\mathcal{K}_f(D_2) = r] \times \exp(-||f(D_1) - f(D_2)||/\sigma)$$

By the definition of the function's sensitivity, the exponential term in the product is bounded by $\exp(\Delta f/\sigma)$. This means that by choosing $\sigma \geq \epsilon/\Delta f$, this mechanism meets the definition of ϵ-differential privacy.

2.1.5 Secure Multiparty Computation

Techniques such as k-anonymity and differential privacy describe how a dataset should be anonymized or how statistical queries about a dataset should be answered to preserve privacy. Conceptually, these techniques assume that all data is held by a single trusted

entity which performs the relevant privacy-preserving computations. However, they do not deal with the question of how the dataset itself is constructed, or how the processing is performed if the data is not all held by a single entity.

Secure Multiparty Computation (SMC) is a cryptographic concept that aims to protect privacy in a scenario in which multiple participants, who each hold private inputs, wish to jointly compute some function on their private inputs. The objective of an SMC scheme is that no participant should learn anything other than the correct output of the function. Depending on the nature of the function, this output itself might reveal some information about the inputs. SMC is therefore complementary to techniques such as differential privacy because, in certain settings, the output of the function may reveal private information. For example, if a group of participants $p_1...p_n$ compute the sum of their private inputs and then a subset of these participants $p_1...p_{n-1}$ evaluate the same function, the private input of p_n can be determined by comparing the two outputs. This can be avoided if the function being computed is differentially private.

In SMC, the adversary is assumed to control some subset of the participants, which are referred to as *corrupted* participants. As explained by Lindell and Pinkas [164], there are five main security requirements in SMC:

- **Privacy:** No information about the other participants' inputs should be revealed other than what can be inferred from the output of the function.
- **Correctness:** Every participant must receive the correct output.
- **Independence of inputs:** Corrupted participants' inputs may not be based on or derived from honest participants' inputs.
- **Guaranteed output delivery:** Corrupted participants should not be able to prevent honest participants from receiving the output (i.e. denial of service).
- **Fairness:** Corrupted participants should only receive their outputs if all honest participants also receive their outputs.

The security of an SMC scheme is evaluated using an *ideal/real simulation* in which the behaviour of the real system is compared to that of an ideal system [56]. In this context, the ideal protocol involves an external trusted entity which is assumed to be incorruptible and is willing to assist the participants. All participants send their private inputs directly to this trusted entity. The trusted entity evaluates the function and returns the relevant outputs to the participants. This ideal protocol fulfils all of the above requirements: the inputs are private and independent because they are sent directly to the trusted entity, the output is correct because it is calculated directly by the trusted entity, and the scheme is assumed to be fair and provide guaranteed output delivery because of the incorruptible nature of the trusted entity. In the real world setting, it is usually assumed that this ideal trusted entity does not exist and that the participants must therefore perform some type of cryptographic protocol among themselves, which aims to emulate the behaviour

of the ideal system [164]. The real world protocol is secure if no adversary can do more harm in the real world setting than in the ideal setting [164]. This means that for every possible attack on the real world protocol, the same attack must be possible in the ideal setting, but since the ideal setting fulfils all the requirements, the real setting must be secure. Most real world SMC protocols from the literature are implicitly designed for the *zero-trust* paradigm in which participants do not trust any other entities, irrespective of whether or not these entities are trustworthy. This is a stringent requirement, but if it can be achieved by an efficient SMC protocol, the protocol will exhibit the security guarantees described above. If a protocol were to rely on a some type of real world trusted entity, the security of the protocol would also depend on the trustworthiness of this entity. In real world protocols, two commonly used building blocks of cryptographic SMC protocols are oblivious transfer and homomorphic encryption.

Oblivious Transfer

Oblivious transfer, as introduced by Rabin [219], is a building block of many SMC protocols [164]. It has been shown that any SMC protocol can be constructed using only oblivious transfer [150]. Oblivious transfer is a two-party protocol that allows a recipient to request information from a sender without the sender learning what information was requested. Specifically, in 1-out-of-2 oblivious transfer, the sender has two values m_0 and m_1 and the recipient has a single selection bit $\sigma = \{0, 1\}$. By the end of the protocol, the recipient will have received only m_σ and the sender will not have gained any further information about σ. Various oblivious transfer protocols have been designed based on similar assumptions to those used in public key protocols (e.g. trapdoor permutations) [164]. Oblivious transfer has also been generalized to 1-out-of-n and k-out-of-n protocols.

Homomorphic Encryption

Homomorphic encryption schemes allow certain operations to be performed on an encrypted plaintext by performing specific operations directly on the ciphertext [164]. This means that an untrusted participant can perform these operations on encrypted data without knowing the decryption key. When the result is decrypted, it is as if the corresponding operations had been performed directly on the plaintext. Partially homomorphic encryption schemes only allow certain operations to be performed on the encrypted plaintext. For example, an additively homomorphic scheme allows addition of two encrypted plaintexts and multiplication of an encrypted plaintext by a constant. An efficient additively homomorphic encryption scheme that is also semantically secure has been proposed by Paillier [205]. Similarly, unpadded RSA and ElGamal encryption are examples of multiplicative homomorphic encryption schemes that allow multiplication of two encrypted plaintexts. In contrast, fully homomorphic encryption (FHE) enables arbitrary computation on encrypted data. Gentry [113] proposed the first plausible construction of an FHE scheme, which is based on ideal lattices.

SMC Construction

In general, a cryptographic SMC protocol can be constructed by representing the function to be evaluated as a *garbled circuit*. This is a combinatorial circuit consisting of logic gates (e.g. AND, OR, XOR etc.) and wires that interconnect these gates. Each possible value (i.e. 0 or 1) for each wire is encoded as an encryption key. Each possible output of each gate is also encoded as an encryption key that is encrypted using the respective input values as keys. For example, in a two-input gate, the participant evaluating the circuit has two encryption keys corresponding to the inputs and can use these to decrypt the key corresponding to the output. However, this participant does not learn the actual values of the inputs and output because these have been substituted by random numbers. In the simplest case of secure two-party computation (as described by Yao [282, 283]), there are two participants, p_0 and p_1, with respective inputs x_0 and x_1 who wish to compute some function $f(x_0, x_1)$. One participant, p_0, constructs a garbled circuit that already includes p_0's input $f(x_0, \cdot)$ and sends this to p_1. In order to evaluate this circuit, p_1 must obtain the encoding for x_1. To protect p_1's privacy, the participants use an oblivious transfer protocol so that p_1 can obtain the encoding of x_1 without revealing its value to p_0. Depending on the length of the input and the type of oblivious transfer protocol, this step may have to be repeated until the full encoding of x_1 has been communicated. The circuit can then be evaluated and the results shared with p_0 if required. For a two-party protocol, the garbled circuit can be constructed using symmetric cryptography, which is relatively fast to evaluate, and thus the largest overhead is the oblivious transfer protocol since this usually involves more expensive cryptographic operations such as modular exponentiation. There are also multiparty protocols that use a similar approach but sometimes incur additional performance limitations, such as requiring asymmetric operations in the circuit. Multiparty protocols also require communication between every pair of participants and some require a broadcast channel [164]. It has been shown that for a computation with n participants, there are cryptographic SMC protocols that require $O(n^2)$ operations per secure multiplication [35]. A more recent protocol allows certain SMC operations to be performed in $O(n)$ online operations per multiplication, provided that $O(n^2)$ operations of pre-computation have already taken place [75]. Orlandi [204] explains how the speed of cryptographic SMC protocols has increased significantly in recent years, but notes that these are still computationally expensive in comparison to the use of a trusted entity.

2.2 Trust and Trustworthiness

The terms *trust* and *trustworthiness* often take on slightly different meanings depending on the context in which they are used. Since these concepts are central to this research endeavour, this section provides background and defines these concepts in this context.

In broad terms, *trust* generally refers to some form of belief about another entity. To be complete, a statement about trust must identify three things: the entity that holds the

belief, the entity about which the belief is held, and the nature of the trust relationship. Complete statements about trust can be reduced to the canonical forms: *"A trusts B to do X"* or *"C trusts D to be Y"*.

When applied to computational systems, it is usually manifest as a belief about the behaviour of a system or a set of characteristics exhibited by the system. For example, Garfinkel et al. [108] use the term trust to describe the *"level of confidence that a computer system will behave as expected"*. The Internet Security Glossary (RFC 4949) [131] defines trust as: *"A feeling of certainty (sometimes based on inconclusive evidence) either that the system will not fail or that the system meets its specifications (i.e., the system does what it claims to do and does not perform unwanted functions)"*. The definition of trust inherently implies that the system could potentially behave in a manner contrary to what is expected.

RFC 4949 [131] also explains that an entity is said to trust another *"when the first entity makes the assumption that the second entity will behave exactly as the first entity expects"*. In this explanation, the second entity is said to be a *trusted system*. In contrast to this, a *trustworthy system* is defined as: *"A system that not only is trusted, but also warrants that trust because the system's behaviour can be validated in some convincing way"*. Avizienis et al [22] define *trustworthiness* as: *"Assurance that a system will perform as expected"*. Proudler [217] identifies the following three necessary conditions for establishing trust in a computational entity:

1. The entity can be unambiguously identified by the relying party.
2. The entity must be known to operate unhindered.
3. The relying party must have (or trust someone who has had) experience of consistent, good behaviour by the entity.

There are various methods for achieving these conditions including technological and non-technological means. This thesis focusses primarily on technological means for establishing trust, such as Trusted Computing, as described in the following sections.

2.3 Trusted Computing

Trusted Computing (TC) is a broad term that encompasses various technologies and approaches that make use of hardware-based functionality to enhance the security of computer systems. The Trusted Computing Group[1] (TCG) is an industry consortium that has developed and standardized many TC technologies, including the Trusted Platform Module (TPM) [255, 256, 257]. Certain hardware-based security technologies developed by companies such as Intel and ARM can also be considered TC technologies. This section provides a brief overview of the aspects of TC relevant to this thesis. A full introduction to TC and, in particular, TCG technologies is provided by Martin [178].

[1] `http://www.trustedcomputinggroup.org/trusted_computing`

2.3.1 Trusted Platform Module

One of the main aspects of TC is the use of a hardware root of trust, such as a Trusted Platform Module (TPM) [255, 256, 257]. The TPM is a standards-based cryptographic co-processor that provides tamper-resistant security functionality including secure storage, random number generation and various cryptographic operations. Unlike a smart card, a TPM is permanently bound to a specific platform. The current generation TPM 1.2 is implemented as a discrete hardware module that is currently integrated into the majority of business-focused PCs and servers. Although the core standards for the next generation TPM 2.0 have recently been published [258, 259, 260, 261], widespread deployment of TPM 2.0 systems has not yet commenced. The TPM 2.0 provides numerous improvements, including a wider choice of cryptographic algorithms (algorithm agility) and enhanced authorization mechanisms. Importantly, the TPM 2.0 functionality does not need to be implemented as a discrete hardware module and could, for example, be implemented as firmware on the platform. This flexibility could lead to significant increases in TPM performance on feature-rich platforms and could enable TPMs to be used in a much broader range of devices, including mobile devices and embedded systems. However, this thesis focuses primarily on the core TPM functionality, as described below, which is provided by both TPM versions. Unless otherwise specified, this research is applicable to both TPM 1.2 and TPM 2.0 platforms.

2.3.2 Platform Configuration Registers

One of the core features of the TPM is a set of special-purpose Platform Configuration Registers (PCRs) that are used to record the software state of the platform. Each PCR stores a single cryptographic hash value (a 20 byte SHA-1 hash in TPM 1.2) which is set to zero when the platform is reset. The contents of the PCRs can be read by system software but cannot be directly modified. Instead, a new value can be *extended* into a PCR by the TPM. The system software issues the `TPM_Extend(`PCR_k,x_{new}`)` command, indicating that the new value x_{new} should be extended into PCR_k. The TPM concatenates x_{new} with the current value of PCR_k (i.e. x_{old}), computes the hash of the concatenation, and stores the result in PCR_k (i.e. $PCR_k = hash(x_{old}||x_{new})$). The PCRs form the basis for various other TPM capabilities, including *sealed storage*, *measured boot*, and *remote attestation* as described below.

2.3.3 Secure Storage

The TPM provides secure storage using a unique asymmetric Storage Root Key (SRK) of which the private key never leaves the TPM. The SRK can encrypt any number of symmetric keys which can in turn encrypt data. Since all keys other than the SRK are stored outside the TPM (in encrypted form), the number of keys that can be protected by the TPM is only limited by the platform's storage capacity. By encrypting data using a TPM-protected key, it is possible to *bind* this data to a specific platform by specifying

that the key may not be migrated to another TPM. It is also possible to *seal* data to a specific software state by instructing the TPM not to decrypt the data unless the PCRs match some predefined value.

However, the current TCG approach to sealed storage has various shortcomings [155]. These mainly arise because the PCR state is dependent on the measurements of all software binaries on the platform. This means that any change to these binaries, such as the installation of a patch or update, results in a different set of PCR values, which would prevent the TPM from unsealing data. Efforts to address this challenge include the Dynamic Root of Trust for Measurement (DRTM), as described below, and the TPM 2.0's enhanced authorization mechanisms [258].

2.3.4 Measured Boot and Secure Boot

A *measured boot*[2] refers to a start-up sequence of a platform in which all executed software components are recorded in the Secure Measurement Log (SML) [178]. This is achieved by creating a *chain of trust* in software and using the TPM as a *root of trust for measurement* (RTM) to protect the integrity of the SML. On start-up, the platform measures the first piece of software to be executed and records this measurement in the SML and extends it into a pre-defined PCR. Before each subsequent piece of software is executed, it is first measured by the preceding software. These measurements are all recorded in the SML and extended into the PCRs. In this context, software is *measured* by computing the cryptographic hash of the executable binary using a system such as the Linux Integrity Measurement Architecture (IMA) [232, 129]. The SML therefore contains a record of all software which has been executed on the system and the integrity of this log can be verified by computing the hash of all entries and comparing this to the PCR values. Even though a malicious piece of software might be able to modify its entry in the SML, it will not be able to change the hash that has been extended into the PCRs and so it will be detected. If the SML and PCRs are checked by some entity after the boot process has been completed, the system is said to have completed an *authenticated boot*. However, since this process relies on a chain of trust, a single untrusted piece of software would break this chain. Although this could be detected from the SML, it would still invalidate any guarantees about subsequent software or the overall platform state. A *secure boot*, proceeds in a similar manner but adds a verification step before each piece of software is executed. If the measurement of the software does not match any permitted value from a secure whitelist, the software is prevented from executing.

2.3.5 Remote Attestation

If a system has performed a measured or secure boot, it can communicate this information to a remote party through the process of remote attestation. The objective of this process is for the attesting entity (the *prover*) to provide the remote entity (the *verifier*)

[2]This process is also sometimes referred to as a *trusted boot* [58].

with an integrity-protected record of all software that has been executed on the prover's system. With this information, the verifier should be able to make a decision about the trustworthiness of the prover's system. By providing this information, the prover's system is said to be *trustable*. In the remote attestation process, the verifier is provided with the SML as well as a signed representation of the current PCR values.

However, since the SML is based on the measurements of the executable binaries, there is a high degree of variability in this log. This leads to a similar problem as encountered with secure storage, that a change to any software binary results in a different measurement value. Additionally, since the measurements are based on a chain of trust, the verifier must check that every entry in the log is trusted. In 2009, Lyle performed an experiment to quantify the number of SML measurements generated by a web server and a standard client PC [172]. The web server's SML contained 277 initial entries, whilst the client system exhibited at least 1800 entries [172]. The rate of change of these entries was also monitored for the web server, showing that there were 1137 changes to these measurements over a 32 month period [172]. This makes it challenging for the verifier to make an informed decision about the trustworthiness of the platform and is sometimes referred to as the binary attestation problem. Furthermore, current remote attestation protocols were not designed to handle frequent attestations to different verifiers and so are not sufficiently scalable for use in systems such as the TRE. These and other challenges of remote attestation are discussed in depth in Chapter 7, in which a new scalable remote attestation mechanism is proposed.

2.3.6 Dynamic Root of Trust for Measurement

As described above, one of the challenges in using the TPM is that all system software that is measured into the PCRs, including the BIOS, master boot record (MBR) and boot loader, forms part of the software Trusted Computing Base (TCB). This increases both the size and the rate of change of the TCB, thus increasing the effort required by the verifier to establish the trustworthiness of the system. To address this challenge, the concept of a *late-launch* was introduced to allow the system to start-up in an unmeasured state and then transition into a measured state. In practice, a late-launch is initiated by loading an Authenticated Code Module (ACM), which is supplied and signed by the CPU manufacturer, rendezvousing the CPU cores and executing a special CPU instruction. The late-launch resets a subset of the PCRs, known as the Dynamic PCRs, disables Direct Memory Access (DMA) and partially resets the CPU. The ACM is then measured and extended into the dynamic PCRs before it is executed. The ACM in turn measures, extends and executes whatever software has been specified by the user, known as the Measured Launch Environment (MLE). This is said to provide a *Dynamic Root of Trust for Measurement (DRTM)* which can be used as a trust anchor for subsequent operations. The chain of trust begins from the late-launch and thus excludes from the TCB all software that had previously been executed on the platform. A DRTM late-launch is therefore a

collaborative operation between the platform and the TPM and requires special CPU extensions. This functionality is available on current CPUs that support Intel's Trusted Execution Technology (TXT) [132] or AMD's Secure Virtual Machine (SVM) extensions. The *tboot* project[3] uses Intel's TXT functionality to perform a DRTM late-launch of an operating system (OS) kernel or a virtual machine monitor (VMM). By using tboot, the BIOS, MBR and boot loader are all removed from the TCB. Toegl et al. [253] use the DRTM late-launch in their *acTvSM* software platform to provide integrity guarantees to applications and services running on the platform. The late-launch measures and executes a Linux kernel and the acTvSM base system, which takes the role of a hypervisor by managing the platform hardware and providing virtualized services to applications. The platform is designed to ensure that the PCR values consistently reach the correct state and can thus be used to seal data to this state [253].

2.4 Trusted Execution Environments

Even if a system makes use of a DRTM late-launch, as described above, the TCB will still contain a significant amount of software. For example, the quantitative measurements obtained by Lyle [172], as described above, would not be significantly reduced if the OS is still included in the TCB. All software in the TCB must be trusted both by applications running on the platform as well as by remote parties wishing to verify the platform through remote attestation. It could be argued that most of the software running in the TCB is not actually security-sensitive and is only included due to the constraints of the system architecture. This has led to various efforts to split up or partition this software into sensitive and non-sensitive sections, in order to reduce the TCB of the sensitive section. For example, Lyle and Martin [173] proposed a *split service architecture* for web services in which the software is partitioned into an untrusted front-end and a trustworthy back-end with a minimized TCB. The idea of partitioning software based on security requirements has given rise to the concept of a Trusted Execution Environment (TEE). In general, a TEE is a mechanism for protecting sensitive code and data on an untrusted platform. The TEE provides an isolated execution environment and various memory protection mechanisms that enable certain pieces of code to execute without interference from untrusted software on the platform. In order to achieve this, the TEE generally requires some type of hardware security mechanism on the platform, and thus the TEE architecture is often defined by the underlying technology. This section provides background on some of the main TEE architectures and technologies.

2.4.1 Flicker

One of the first examples of a hardware-backed TEE on the PC platform was the *Flicker* research project [181], which uses the DRTM late-launch to provide an isolated and pro-

[3] `http://sourceforge.net/projects/tboot/`

tected execution environment. When invoked, Flicker suspends the host OS and initiates a late-launch. It then executes a small user-defined section of code called a Piece of Application Logic (PAL). Due to the partial CPU reset, the PAL is protected from any software that was executed on the system before the late-launch and so can be said to execute in a Trusted Execution Environment (TEE). Once the PAL has completed execution, its memory is cleared, the PCRs are extended with a well-known value to indicate that the TEE is no longer active and the host OS is resumed from its suspended state. By using late-launch and the dynamic PCRs, Flicker also makes it possible to seal and unseal data within a specific PAL irrespective of the overall state of the host system. However, the functionality of a PAL is inherently limited because it cannot use any libraries or hardware drivers from the host OS since these may have been compromised before the late-launch. The performance of this framework is also limited by the time required to set up and initiate the DRTM late-launch.

2.4.2 ARM TrustZone

One of the most widely-deployed examples of a TEE is ARM TrustZone technology[4], which is a set of system capabilities and CPU extensions that provide two logical execution environments on the same core. In this *split-world* architecture, the feature-rich OS and applications are run in the *normal world* whilst security-sensitive code is executed in the *secure-world*. The secure world can therefore be used to provide secure services to the normal world. For example, the secure world could run a software-based smartcard or TPM that provides security guarantees to the normal world as if it were implemented on dedicated hardware. However, TrustZone does not provide a mechanism for establishing the trustworthiness of the software in the secure world. The software in the secure world is *trusted* by virtue of the fact that at present only trusted entities, such as the manufacturer, can run code in the secure world and this code is always executed before the software in the normal world. This approach means that most applications on the device cannot make use of TrustZone directly, although there are various ongoing efforts to change this [279, 280, 278, 91, 90]. Since the software in the secure world constitutes the root of trust for the platform, it must be trusted to the same degree as the TPM is trusted in TC.

2.4.3 Intel SGX

Intel Software Guard Extensions (SGX) are a set of CPU instructions and memory access mechanisms that can be used to establish a TEE, called an *enclave*, on the platform [183, 133]. Traditionally, privileged code (e.g. the OS kernel) and its associated data is protected from unprivileged code (e.g. user-space applications) by the x86 CPU's protected mode mechanism. Application code in an outer ring (ring-3) cannot read or access memory associated with a different application or code in an inner, more privileged ring (ring-0). However, protected mode does not protect unprivileged ring-3 code from other code

[4] `http://www.arm.com/products/processors/technologies/trustzone/index.php`

running in the same ring, or from privileged code on the same platform. SGX aims to protect the integrity of certain unprivileged code and the integrity and confidentiality of its associated data from all other software on the platform. One of the main use cases for this technology is therefore to protect user-space applications against an untrusted OS or other malware on the platform. In SGX, code and data to be protected are encapsulated in a secure *enclave*. Unlike TrustZone's static split-world architecture, applications can create arbitrarily many SGX enclaves, limited only by the capacity of the platform. Once the enclave has been launched, the CPU enforces that the enclave's memory region can only be read or accessed from within the enclave itself. Whenever enclave data is written to system memory, it is encrypted by the CPU to protect its confidentiality and integrity, even against adversaries that can read system memory. SGX still enables the OS to manage resources and schedule applications as usual, so the OS can still interrupt an enclave similarly to a normal application. An important feature of SGX is that it provides a mechanism similar to TC remote attestation whereby the CPU can provide a signed assertion, that can be verified by either a local or remote verifier, stating precisely what code and data is contained in an enclave [12]. Unlike TrustZone, this mechanism provides a root of trust for measurement for the TEE, making it trustworthy rather than merely trusted. Some initial research efforts have begun to explore how SGX could be used to enhance security in the context of cloud computing [31, 235]. Although hardware-supported TEE mechanisms are increasingly seen as a potential solution to many security challenges, they are not the primary focus of this thesis. TEEs aim to protect and ensure the trustworthiness of specific pieces of code on an untrusted platform, whereas TC has the broader goal of establishing trust in the platform as a whole. Given the diversity and complexity of modern client platforms, it appears that TEE-based approaches will succeed where TC has not. However for a system like the TRE, TC approaches and technologies still hold significant potential, as will be shown in this thesis.

2.5 Trusted Third Parties

The concept of a trusted-third-party (TTP) is an architectural construct based on the notions of trust described above. An interaction between two participants may require some degree of trust between the parties for the interaction to succeed. This can be one-way trust where only one participant trusts the other, or mutual trust between both parties. In some cases it may not be possible to establish this trust directly, or it may be the case that one or both parties specifically distrust the other for various reasons. In such cases, it may still be possible to facilitate the interaction by introducing a TTP. By definition, a TTP is distinct from the two participants in the interaction. Instead it is an entity that is trusted by both parties, even if they cannot or do not trust each other. The TTP concept is not limited to two-party interactions; a TTP could be trusted by an arbitrary number of participants in order to facilitate interaction between them.

There are various examples of the use of TTPs in modern computational and com-

munication systems. A common example is a Certificate Authority (CA) in a Public Key Infrastructure (PKI). The CA is said to provide a trust anchor for the PKI by verifying participants' real identities and signing digital certificates to this effect (e.g. the CA provides a TLS certificate for a specific website). However, this only works if that particular CA is also trusted by the other participants (e.g. the clients have the CA's root certificate installed in their browsers). To achieve mutual authentication using this mechanism, each participant must trust the CA used by the other participant, or a mutually trusted CA must be used. TTPs are also used in other contexts, such as in the smart grid [191, 42] (as discussed in Chapter 3) and in location-based services [110, 123, 189] (as discussed in Chapter 8).

Ajmani et al. [9] proposed the Trusted Execution Platform (TEP), a TTP that can perform general-purpose computations on behalf of two or more mutually-distrusting participants. TEP uses the Java language security mechanisms and specially designed cryptographic protocols to provide certain assurances of its trustworthiness. However, since these assertions are not supported by hardware-based trust mechanisms, they are dependent on the trustworthiness of the OS and other software on the platform (e.g. the Java runtime environment).

Trusted entities, which could be considered TTPs, are also used in differential privacy and in TC as discussed above. In differential privacy, the true dataset is always held by a trusted entity which answers statistical queries using a differentially private mechanism [87]. The individuals in the dataset must trust that this entity will not compromise their privacy by revealing too much information. The participants querying the dataset must also trust that this entity is operating correctly in order to have confidence in the results. Similarly, the framework by van den Braak et al. [47] makes use of a TTP to facilitate privacy-preserving sharing of personal data between public organizations.

In TC, the process of remote attestation could compromise the privacy of the prover unless additional steps are taken. If the prover used the same TPM key to sign quotes of the PCR values and then sent these to different verifiers, it would be possible for these verifiers to link these quotes to the same prover based on the signatures. In some cases this might be desirable, but in most cases this would compromise privacy because, for example, all actions of the prover could be linked together and tracked. In order to overcome this challenge, a TPM has the capability to generate arbitrarily many *Attestation Identity Keys (AIKs)* and use these for different remote attestation instances. However, in order to be sure that a particular key pair is a valid TPM-bound AIK, the TPM must obtain a certificate for each AIK from a special *Privacy CA* [215]. Since the TPM uses its primary identity to authenticate itself to the Privacy CA, it must trust this entity not to compromise its privacy by revealing the link between these AIKs and its real identity. The verifiers must also trust that the Privacy CA will only issue certificates for legitimate AIKs.

Although a TTP is trusted by two or more other participants, it is not necessarily trustworthy. Security breaches of trusted CAs, such as Comodo and DigiNotar [158] are

examples of how trust in a TTP can be undermined. Ideally, a TTP should provide some form of assurance or proof of its trustworthiness. TC technologies and approaches represent a possible technical mechanism for achieving this objective.

Most of the initial TC use cases focussed on end-point or client systems. For example, Trusted Network Connect (TNC) [254] is a mechanism for determining the software state of an end-point system before it is allowed to join a network. Lyle [172] evaluated the use of TC remote attestation in the context of a service-oriented architecture, focussing on the servers instead of the client systems. Although some challenges remain, the evaluation showed that TC remote attestation is a viable possibility for enhancing the trustworthiness of services, and that it can be made significantly more practical through various proposed enhancements. Moving beyond clients and servers, Pirker et al. [215] have provided one of the few examples of the use of TC approaches and technologies to establish the trustworthiness of a third-party system. They showed how TC remote attestation can be used to provide assurances about the behaviour of a Privacy CA [215]. However, beyond this example, the use of TC technologies and approaches for establishing the trustworthiness of third party systems has not previously been studied. To fill this gap, this thesis proposes and investigates the concept of the Trustworthy Remote Entity (TRE), a system that exhibits similar functional characteristics to a TTP whilst providing strong assertions of its trustworthiness.

Chapter 3

Security and Privacy in the Smart Grid

This chapter draws on research described in the following publication:

- A. J. Paverd, A. P. Martin, & I. Brown, "Security and Privacy in Smart Grid Demand Response Systems" In: *Second Open EIT ICT Labs Workshop on Smart Grid Security (SmartGridSec'14)*, 2014 [208].

3.1 Smart Grid Architecture 32
3.2 Consumers' Perspective of the Smart Grid 34
3.3 Information Flows 35
3.4 Security and Privacy Threats 39
3.5 Existing Solutions 40
3.6 Gap Analysis and Summary 51

This chapter presents background information about the smart grid, based on recent scientific literature, published standards, and national specification documents. It introduces the smart grid architecture and explains which aspects are within the scope of this research. In this context, the three primary smart grid information flows are described along with relevant examples of each. Based on recent scientific literature, various threats to security and privacy arising from smart grid systems are presented. The remainder of this chapter discusses existing solutions that have been proposed to mitigate these threats and enhance users' privacy in the smart grid. The chapter concludes with a gap analysis of previous research highlighting the gap addressed by this thesis.

3.1 Smart Grid Architecture

Figure 3.1 is the widely cited conceptual overview of the smart grid from the *NIST Framework and Roadmap for Smart Grid Interoperability Standards* [198]. This figure shows the flow of energy as well as the communication links between the various entities in the smart grid. The communication links in this figure represent the flow of information rather than physical communication links. In particular, this model highlights that consumers exchange information with energy suppliers, Distribution Network Operators (DNOs) and other entities such as operations centres and energy markets.

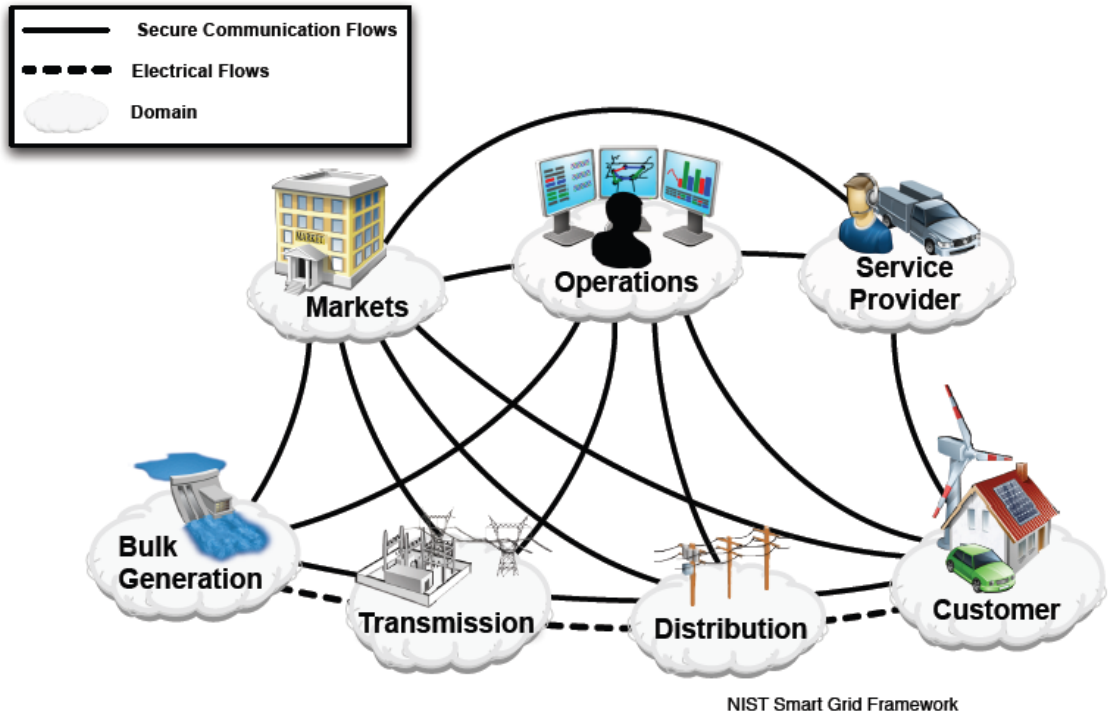

Figure 3.1: Widely cited NIST overview of smart grid entities [198].

As a concrete implementation of the NIST model, a section of the proposed smart grid communication architecture in the UK is shown in Figure 3.2. This illustrates how the communication between consumers and other entities will be realized. In this architecture, all information from consumers passes through an intermediate entity called the *Data Communications Company (DCC)* [270], which encompasses the communication service providers and the data service provider. The communication service providers provide the physical communication links between consumers and the centralized data service provider. The data service provider stores the consumers' data and makes it accessible to authorized entities, such as energy suppliers and DNOs. The services provided by the DCC to these authorized entities are defined in the UK Smart Energy Code [269].

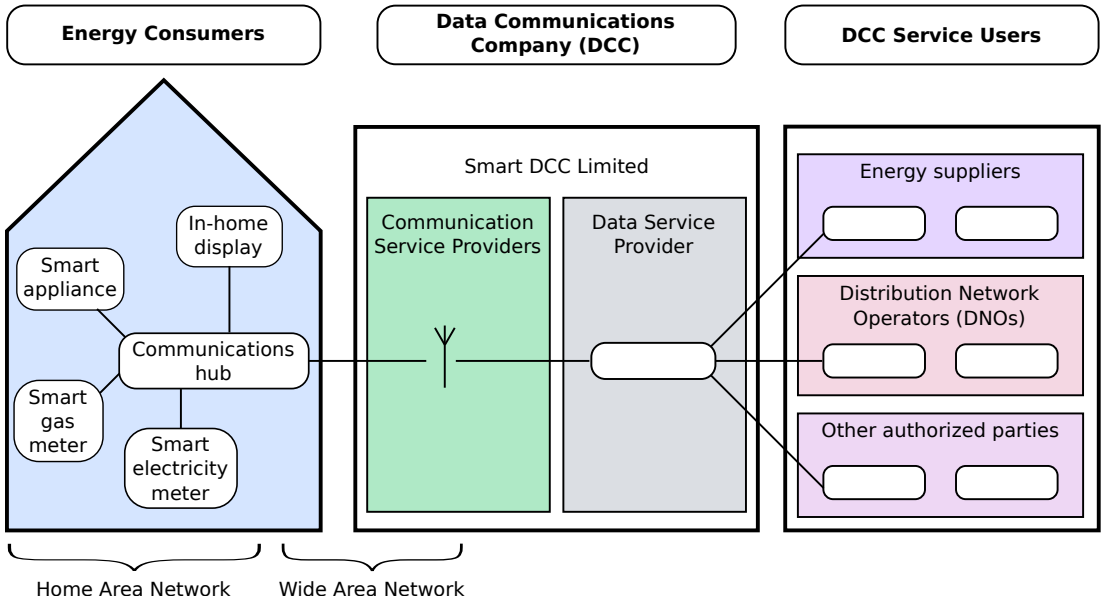

Figure 3.2: Planned UK smart grid model (adapted from [270]).

As explained by Darby [79], from the suppliers' perspective, electrical energy has always had a *time* dimension because, in the ideal case, electricity supply should match demand in order to minimize wastage. This time dimension is becoming increasingly apparent as more time-variable renewable energy resources are integrated into the grid. Ideally, some type of *active demand-side* is required to balance this time-variability in supply. Previously, only large consumers (e.g. industrial consumers) have been active participants in the grid, since these consumers could bring about relatively large changes in demand with minimal coordination effort. However, residential demand often makes up a significant percentage of overall consumption. For example, in 2014, 30% of the total electricity consumption in the UK was from residential consumers [264]. One of the objectives of the smart grid is to provide the communication systems that enable all consumers (including residential consumers) to become active participants in the grid. Darby and McKenna [78, 80] have investigated a number of possible approaches for residential demand response, including static price response, load response, dynamic price response, and frequency

response strategies. However, as explained by Darby [79], the extent to which small-scale residential consumers are able and/or willing to participate remains an open question. Darby also notes that policy questions around consumers' privacy have arisen with respect to residential demand response.

3.2 Consumers' Perspective of the Smart Grid

As shown in the previous section, the smart grid is a large interconnected system of systems that encompasses multiple domains such as energy suppliers, distribution network operators (DNOs), energy markets and consumers. Each domain has a different role within the smart grid and a different set of interfaces for communicating with other domains. From the perspective of residential consumers, there are two major aspects of the smart grid: smart metering and energy management services.

3.2.1 Smart Metering

The first aspect is the measurement and reporting of energy usage facilitated by smart energy meters and the Advanced Metering Infrastructure (AMI). The primary function of a smart meter is to measure energy usage at an end-point and communicate this information to various entities such as the energy supplier, DNO and third party energy service providers. Depending on the particular implementation of the smart grid, smart meters could communicate directly with these entities or could exchange information via intermediate entities such as the DCC in the UK.

In all cases, the distinguishing feature of smart meters in comparison to previous generations of meters is that smart meters allow more frequent measurement and reporting of energy consumption. In modern systems, the intervals between measurements are in the order of minutes and this could be decreased further to provide even finer temporal granularity in the future. These frequent measurements from smart meters form the basis of the new functionality provided by the smart grid. These measurements are used by the DNO to monitor the distribution network and by energy suppliers to improve the accuracy of demand forecasting and enable new billing approaches. In some countries, smart meters are also capable of measuring the flow of energy in both directions, thus allowing consumers to be remunerated at the applicable feed-in tariff for energy fed back to the grid. It order to obtain the full benefits of the smart grid, smart meters must be deployed to the majority of consumers. From the consumers' perspective, smart metering is usually the first aspect to be implemented since it is a pre-requisite for more advanced services.

3.2.2 Energy Management

The second major aspect from the consumers' perspective involves energy-related services and applications that are facilitated by smart meters and the smart grid. In addition

to sending measurements to external entities, smart meters provide this information to consumers. Initially, this will be facilitated by In-Home Displays (IHDs) in consumers' premises that display information such as the current consumption as well as current and future price information [265]. Some smart meters can also provide price information to other systems such as energy-aware smart appliances. It is anticipated that these appliances will use this information to optimize their energy consumption, for example by delaying their operation to off-peak periods when possible. In future, some of these smart appliances could provide remote control capability to allow remote power management by users or other systems.

Automated energy management systems can be used to manage multiple such appliances in order to optimize the overall energy consumption of the home. For example, a Home Energy Management System (HEMS) [19, 157, 200, 214] is a central entity within the home that communicates with the energy supplier as well as with smart home appliances. Home appliances report their current power requirements as well as other information such as the minimum level to which they could reduce consumption if required. The HEMS can activate or deactivate specific appliances or change appliance settings based on information received from the smart meter. This management functionality could also be provided by external entities such as the energy supplier or third party energy service providers. The capability to manage energy consumption allows the consumer to become an active participant in Demand Response (DR) activities. The consumer or the HEMS can reduce demand or shift load in response to increases in price or other DR signals.

In addition to managing the consumption of appliances, energy management is important for consumers who have the ability to produce or store energy. For example, some consumers install residential photovoltaic solar panels, wind turbines or other generating capacity. These consumers are sometimes referred to as producer-consumers or *prosumers* but, unless otherwise specified, in this thesis they are still referred to as consumers since this is their primary activity. It is also becoming increasingly common for consumers to have significant energy storage capacity in the form of Plug-in Electric Vehicles (PEVs). Since the smart grid enables bi-directional flow of energy, consumers can opt to feed energy back into the grid at specific times. This capability would also be managed by the HEMS.

All aspects of the smart grid described in this section involve communication between the consumer (via either the smart meter or the HEMS) and other smart grid entities such as the DNO, energy supplier or other service providers. The next section presents the details of these information flows between consumers and other entities.

3.3 Information Flows

In the smart grid, there are three primary information flows between consumers and external entities. In this context, the term *information flow* is used to refer to any exchange of information between communicating parties in order to achieve a specific objective. As an abstract construct, the information flow defines only *what* information is exchanged

between *which* communicating parties. It does not include details such as the communication protocols or technologies used to exchange the information. A one-to-one information flow can be uni-directional or bi-directional whilst a one-to-many information flow can be a multicast or a broadcast.

3.3.1 Network Monitoring

The monitoring information flow facilitates detailed monitoring of the distribution network by the DNO and monitoring of demand by the energy supplier. In this information flow, the frequent energy measurements taken by smart meters are sent to the DNO or energy supplier.

Network monitoring by the DNO is viewed as one of the main features of the smart grid because the DNO can monitor individual sectors of the distribution network at a high temporal resolution. The fine spatial granularity of individual smart meter measurements is not usually required since the DNO is primarily interested in monitoring specific sectors of the network (e.g. those served by a specific substation). However, the high temporal granularity of the measurements is very important for network monitoring so that the DNO can observe changes in consumption and take appropriate action. This allows the DNO to detect and diagnose faults and to balance the flow of energy throughout the network. Active network management is becoming increasingly important due to the rise of Distributed Energy Resources (DERs). DERs are small-scale power generation technologies (typically ranging from 1 kW to 10,000 kW) and include systems such as local generating capacity and PEVs that can supply stored energy back to the grid.

Similarly, the monitoring information flow allows energy suppliers to improve the accuracy of their demand forecasting. Using the frequent measurements from smart meters, the supplier can more accurately predict the likely future demand and plan to meet this demand. The complexity of demand forecasting will increase as more DERs are added to the grid since the predictions must take into account the likely output from each DER. Furthermore, PEVs add an additional spatial dimension since they could be charged at one location and then driven to another location, physically moving their stored energy to a different location in the grid. The monitoring information flow is a uni-directional flow from consumers to either the DNO or the energy supplier.

3.3.2 Billing

Another main feature of the smart grid is that it enables new types of billing for energy consumption. The frequent measurements performed by smart meters can facilitate strategies such as Time-of-Use (ToU) billing or dynamic pricing [11]. In a ToU scenario, the price of energy varies predictably depending on the time of day in an attempt to reduce consumption at peak times. With dynamic pricing, the price varies over time depending on the prevailing supply and demand situation. The price can also vary depending on location in order to balance the flow of energy in the grid.

In order to implement these new types of billing, the energy supplier requires frequent consumption measurements from each consumer. This information is sent from smart meters to the energy supplier constituting the billing information flow. For these billing approaches to be effective in reducing or shifting demand, the prevailing price information must also be communicated from the energy supplier to consumers. However, this is usually sent as a multicast or broadcast and so is not included in the one-to-one billing information flow. Therefore, the billing information flow is a uni-directional flow from consumers to the energy supplier.

It may be suggested that the cumulative bill could be calculated internally by the smart meter and only the result sent to the energy supplier at the end of the billing period. This would reduce the amount of information transferred in this information flow, but would require the energy supplier to place a greater degree of trust in the smart meter. Not only must the smart meter be trusted to obtain accurate measurements, but it must also now be trusted to multiply these measurements by the correct price, and to store the cumulative total. This requires the smart meter to have integrity-protected and rollback-protected non-volatile data storage to prevent the adversary (who may be the consumer and thus have physical access to the device) from modifying the cumulative value, or reverting it to a previous (lower) value. These additional security capabilities would significantly increase the cost of each smart meter, but would not be required if the billing computation were performed externally.

3.3.3 Demand Response

The third primary information flow is used to facilitate Demand Response (DR) communication. The US Department of Energy (DoE) defines DR as:

"Changes in electric usage by end-use customers from their normal consumption patterns in response to changes in the price of electricity over time, or to incentive payments designed to induce lower electricity use at times of high wholesale market prices or when system reliability is jeopardized" [271].

Albadi et al. [11] have presented a classification of DR approaches. In this classification, there are two main types: price-based and incentive-based DR approaches. In price-based approaches, the energy price changes as a function of time. Examples of price-based approaches include ToU billing and dynamic pricing. As described above, these price-based approaches are implemented using the billing information flow. Incentive-based DR approaches are subdivided by Albadi et al. [11] into classical approaches and market-based approaches. Classical approaches include direct load control or forced curtailment (load shedding) whilst market-based approaches offer financial incentives to consumers who participate in DR events. Similar approaches have been discussed by Darby and McKenna [78, 80]. One of the most promising market-based DR approaches is *demand bidding* in which consumers bid to reduce demand when required. In this approach, consumers interact with the Demand Side Manager (DSM) as follows: When a shortage in supply is expected, the

DSM creates a DR event and notifies all consumers. Consumers send bids to the DSM stating the amount of consumption they are willing to reduce and the desired incentive price for this reduction. The DSM selects the winning bids and communicates its decision to individual consumers. After the event, the respective incentives are credited to the successful bidders. Unlike other DR approaches, demand bidding requires full bi-directional communication between consumers and the DSM so that consumers can submit bids and the DSM can respond to individual consumers to accept or reject their bids. This provides a closed feedback loop allowing the DSM to monitor and control the DR process. Standards such as Open Automated Demand Response (OpenADR) version 1.0 [213] and the subsequent OASIS Energy Interoperation (EI) standard [202] specify data models for demand bidding. Although initially targeted at large industrial consumers, demand bidding can also be applied to residential consumers. In a residential setting, a HEMS or feature-rich smart meter would place bids and control energy-consuming systems according to a user-defined policy.

Since certain DR approaches, such as demand bidding, cannot be achieved using only the billing information flow, a third primary information flow in the smart grid must be considered. This demand response information flow represents an exchange of information between the consumers and the DSM. The nature of the information exchanged in this flow depends on the type of DR approach in use. This information flow must support full bi-directional communication required for DR approaches such as demand bidding.

The above description of participative DR is widely recognized as one of the main benefits of a smart grid. However, it should be noted that this assumes a particular time scale (i.e. in the order of minutes or hours) for a DR action to take effect. Darby [78] suggests that there is a *spectrum* of DR activities, operating at different time scales. For example, it may sometimes be desirable to have DR actions take effect on a much shorter time scale (e.g. in the order of seconds or hundreds of milliseconds). In order to achieve this response time, explicit communication between consumers and the DSM is generally not possible. Instead, one possibility would be to use smart appliances that monitor the frequency of the AC electricity and reduce consumption if this frequency drops below a certain threshold, since this is usually an indication of high demand. This could be called *cooperative DR* since consumers cooperate with each other to achieve the DR objective. Although this type of DR is theoretically feasible, it presents various practical challenges. For example, the inclusion of frequency sensing capabilities would increase the cost of appliances. Furthermore, since there is no automated enforcement mechanism, this type of *cooperative DR* assumes that all users are trustworthy and will not circumvent the DR mechanism. Although this type of cooperative DR could complement the participative DR approaches described above, it is not within the scope of this research since it does not involve explicit communication between the consumers and the DSM.

3.4 Security and Privacy Threats

Although the smart grid provides numerous benefits, it raises a number of serious security and privacy concerns. In terms of security, the use of digital communication networks throughout the smart grid presents a new vector for attacking the system. A *false data injection* attack involves modifying or falsifying measurement data sent over the network in order to negatively affect the system. For example, modification or falsification of measurements from a large number of smart meters could affect the demand forecasting process and potentially jeopardize the stability of the grid. In some cases, certain smart grid components can be remotely controlled over the network. For example, in the UK, smart meters are required to support remote commands that switch the consumer onto a prepaid tariff plan or temporarily interrupt the energy supply during periods of high demand [13]. As explained by Anderson and Fuloria [13], these capabilities pose a significant security risk if they can be misused by unauthorized entities. From the perspective of the DNOs and energy suppliers, this is also an important concern because unauthorized control of multiple homes could destabilize sections of the grid. A further security consideration is that the energy management systems (either HEMS or external systems) will likely be software-based. Since these systems are inherently network-connected, they are likely to be vulnerable to many of the threats that are currently prevalent in the PC domain.

From the consumers' perspective, the frequent energy measurements performed by smart meters also lead to privacy concerns. Before smart meters, the only energy-related information about a consumer that could be obtained was the consumer's total consumption over the previous billing period. Although this revealed some information about the consumer (e.g. approximate size of the residence), it was not generally considered to be a privacy concern (except in some cases where consumption information was used to investigate marijuana growth and drug manufacturing [165, 218]). However, the frequent energy measurements performed by smart meters can be used to infer detailed information about consumers, including behavioural patterns, occupancy details and possibly even medical conditions [218, 50]. Techniques such as Non-Invasive Load Monitoring (NILM) [125, 124, 86, 285, 30, 29, 38] aim to identify individual energy-consuming appliances in a home based on frequent measurements of the home's total energy usage. In addition to identifying appliances, it has been shown that frequent energy consumption measurements can be used to infer other types of private information. Using real energy consumption patterns at 30-minute measurement intervals, Beckel et al. [32] have shown that it is possible to perform automatic socio-economic classification of households with over 70% accuracy. It seems likely that the accuracy of these inference techniques will continue to improve.

Consumers generally consider this type of information to be private and thus, if it were obtained by an unauthorized or untrusted entity, it would constitute a privacy breach. Various legitimate participants in the smart grid, such as the energy supplier and DNO as well as intermediaries such as the communication service providers (e.g. the DCC in the

UK) may have access to the energy measurements from smart meters. However, consumers may not want these entities to be able to infer private information about them from these measurements.

Unlike the security threats described above, consumers' privacy is threatened by internal adversaries who are legitimate participants in the communication protocol. Furthermore, even if entities such as the DNO and energy supplier are trusted with the frequent measurements, there is still a possibility that this data could be stolen by an external adversary, which would also lead to a privacy breach. Formal definitions and explanations of these types of attacks as well as the relevant adversary models are presented in Chapter 4. As mentioned in Chapter 1, these privacy concerns are sufficiently serious to have caused delays to the deployment of smart meters.

In addition to the privacy concerns raised by smart meters, the bi-directional communication in the DR information flow also leads to serious privacy concerns [208]. In particular, protocols such as demand bidding could reveal private information about consumers. If bids placed by consumers were visible to an untrusted entity and could be linked to individual consumers, the untrusted entity would learn information such as the consumer's chosen energy supplier and tariff plan. Furthermore, the energy reduction specified in the bid reveals some information about the consumer's total energy consumption. In the same way that frequent energy measurements from smart meters can be used to make inferences, DR bids could also be used to infer private information. For example, a bid to decrease a large load, equal to that of a PEV, indicates that the consumer probably owns such a vehicle and would otherwise be recharging it. The ability to link the bids to individual consumers also allows the untrusted entity to build up a profile of the consumer's behaviour. For example, it could be inferred that a specific consumer usually charges an electric vehicle after returning from work in the evening. Any deviations from this profile could lead to further inferences about the consumer's behaviour. Continuing the previous example: if a particular consumer regularly bids to stop recharging an electric vehicle at peak times, any deviation from this pattern could indicate that the electric vehicle and its owner are away from home at that time. The messages in the bidding protocol can be encrypted to prevent interception by external adversaries or untrusted intermediaries (e.g. communication service providers), but if consumers do not trust the DSM itself, the bidding protocol could result in a breach of consumers' privacy.

3.5 Existing Solutions

This section reviews recent technical proposals for enhancing users' privacy in the smart grid. The proposals are grouped according to their fundamental approaches. Jawurek et al. have also presented a comprehensive survey of recent privacy technologies for smart grids [140].

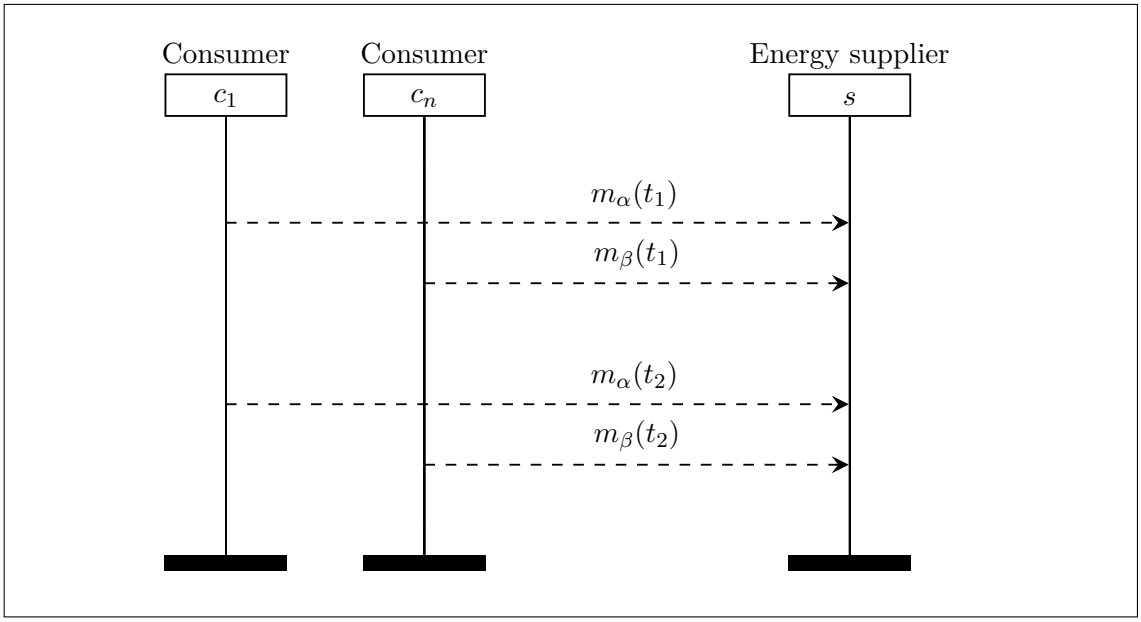

Figure 3.3: Abstract model of a pseudonymization scheme

3.5.1 Pseudonymization

A very basic approach is for consumers to use random pseudonyms when reporting energy measurements. Figure 3.3 shows an abstract model of this type of approach in which consumers c_1 to c_n send their energy measurements to the energy supplier under the pseudonyms α and β. Each consumer can submit a measurement $m(t_n)$ for each time period t_n. In Figure 3.3, the messages are shown as dashed lines to indicate that these cannot be sent directly to the energy supplier since this would reveal the consumers' network addresses. Instead these messages must be sent over some type of anonymous communication channel, such as a anonymity network. Alternatively, instead of using a unique pseudonym for each consumer, a common pseudonym could be used be all consumers in a particular location, such as the area served by a specific substation. Therefore, this approach could theoretically be used in the monitoring information flow. However, systems based on this approach must provide an additional mechanism to support the billing information flow, which requires that measurements must be associated with individually named consumers. Real protocols based on this approach must provide some mechanism to ensure that only legitimate consumers can submit measurements and that each consumer can only submit one measurement per time period.

A serious drawback of this type of approach is that the frequent energy measurements can be linked together through the pseudonym. This time-series of measurements forms a pattern of behaviour that can be used to de-anonymize the consumer. Furthermore, this approach introduces potential security concerns. Since it is possible that some number of smart meters could be compromised, some mechanism must be provided to detect these compromised smart meters and mitigate the risk of them injecting falsified measurements.

In turn, this method of de-anonymizing individual smart meters must be adequately protected to ensure that it cannot be used by untrusted entities.

Efthymiou and Kalogridis [89] have proposed an approach based on the use of two separate identifiers for each smart meter. The meter uses the high frequency identifier (HFID) to report frequent energy measurements (e.g. every 15 or 30 minutes) and the low frequency identifier (LFID) to report an aggregated measurement at the end of the billing period (e.g. once a month). The key contribution of their system is that the HFID is not linked to a specific user but only to a general location such as a group of houses or apartments. Their system introduces a third-party escrow provider as the only entity that can link a particular HFID to a corresponding LFID. This role could be fulfilled by the smart meter manufacturer. The various smart grid entities can query the third-party escrow provider to verify the authenticity of a particular HFID. However, for this approach to work, the third-party escrow provider must be completely trusted by all entities in the smart grid. An untrustworthy escrow provider can easily compromise user privacy or facilitate the falsification of HFID measurements. By colluding with an untrusted escrow provider, a compromised smart meter could launch false data injection attacks whilst remaining anonymous. Additionally, the external entities must trust the smart meter to correctly compute the aggregated result reported using the LFID. Finally, this approach only deals with privacy at the application layer of the system and does not consider the networking layer. Even if separate application layer identifiers are used, it may be possible to link these identifiers based on characteristics such as network addresses. For example, if this protocol were used over an IP network, the IP addresses of the smart meters could be used to link the HFID to the corresponding LFID. Similar pseudonymization schemes are described by Finster and Baumgart [101], Borges et al. [43] and Rottondi et al [224].

Jawurek et al. [139] argue that even though the measurements are only associated with a pseudonym, the adversary might still be able to link this pseudonym to a specific consumer. Given a set of energy measurements linked to a pseudonym (such as the HFID above), Jawurek et al. propose two potential attacks: The first is *linking by behaviour anomaly* in which unusual events are correlated to link the measurements to certain external information such as delivery of a new home appliance or a period in which the home is unoccupied. The second is *linking by behaviour pattern* in which the overall pattern is used to link two data sets. This could be used to link different pseudonyms to the same home if the user changes energy supplier or if the supplier changes pseudonyms each day. This demonstrates that basic pseudonymization of measurement data is generally not sufficient to protect user privacy. However, these attacks are not applicable to systems in which each measurement is anonymized independently without the use of pseudonyms.

3.5.2 Temporal Aggregation

Another approach is to aggregate measurement information over time within the home before sending it to any external entities. By sufficiently reducing the measurement fre-

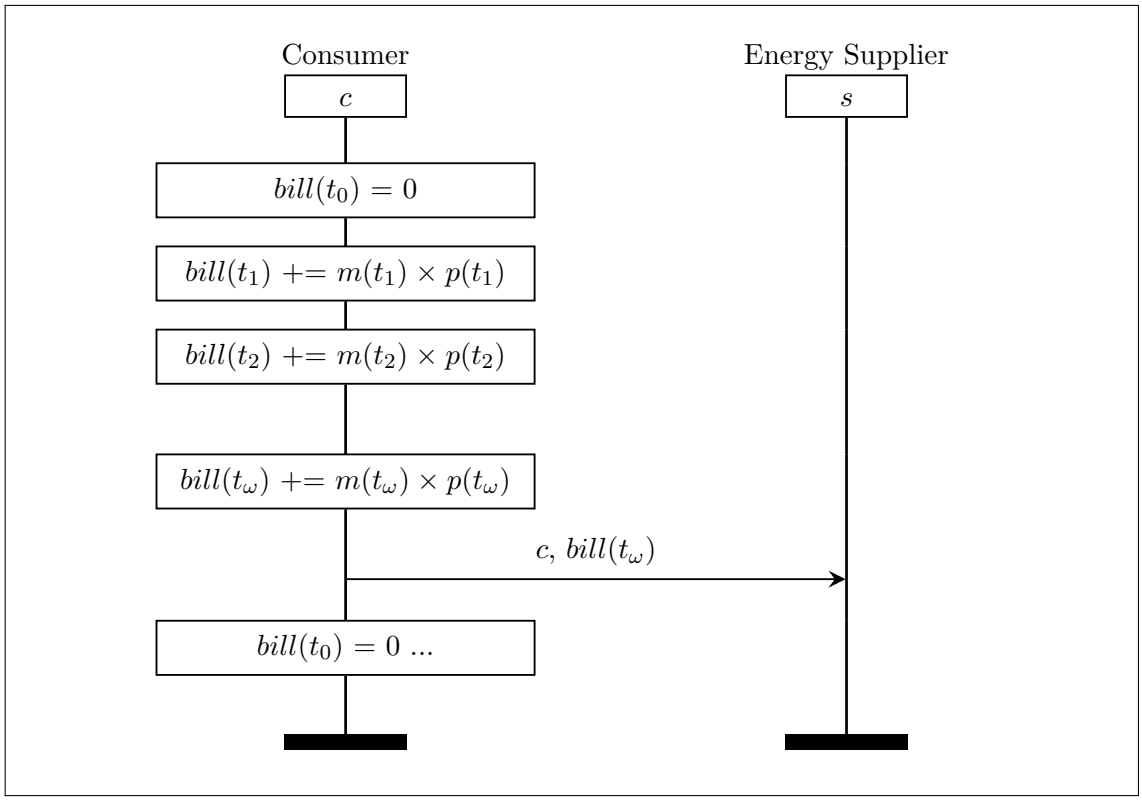

Figure 3.4: Abstract model of a temporal aggregation scheme

quency, the use of NILM techniques is prevented and this essentially results in the same level of privacy that was available before the introduction of smart meters. An abstract model of this type of scheme is shown in Figure 3.4. As shown in this figure, for each time period t_n, each consumer takes the consumption measurement for that period $m(t_n)$, multiplies it by the price per unit for that period $p(t_n)$, and then adds the result to an internal running total $bill(t_n)$. After the last time period t_ω in a particular billing cycle, the total is sent to the energy supplier using the consumer's real identity. However, in order to maintain some of the benefits of the smart grid such as time-variable energy pricing, and to avoid the need for costly secure storage on the smart meters (as explained in Section 3.3.2), real temporal aggregation protocols often use complex cryptographic techniques. Fundamentally, these protocols cannot be used in the network monitoring or DR information flows since they reduce the temporal resolution of the measurements.

Rial and Danezis [221] have presented a privacy-friendly metering system that facilitates ToU billing without leaking any information. In their system, the smart meter still records frequent energy measurements but does not send these to any external entities. Instead, the meter uses an additively homomorphic commitment scheme to generate cryptographic commitments for each measurement. At the end of a specified billing period, these measurements are transferred to the consumer's PC in order to calculate the energy bill. The privacy of the home user is protected because the measurements never leave the

home network and the cryptographic commitment scheme allows the provider to verify that the bill calculation has been performed correctly. However, a disadvantage of this approach is that the provider does not receive frequent measurement data for use in other operations such as demand forecasting or network optimization. Furthermore, the cryptographic calculations required to verify the bills are non-trivial and could take a significant amount of time when multiplied by the potential number of smart meters operated by a large energy provider.

Another approach proposed by Danezis et al. [77] is to use differentially private billing with rebates. This is based on the concept of differential privacy as explained in the previous chapter. In the system proposed by Danezis et al. [77], the user voluntarily increases each measurement made by the smart meter before it is sent to the DNO or energy supplier. Since these increases are random, this allows the user to mask the energy signatures of appliances and thus protect his or her privacy. The system also includes a cryptographic protocol to prove the total amount of random noise added by the user so that a rebate can be issued at the end of the billing period. Although this system allows for frequent communication of measurements, the addition of random noise decreases the usefulness of these measurements from the perspective of the DNO or energy supplier.

Jawurek et al. [138] have proposed a privacy-preserving protocol that facilitates billing by introducing an entity which they call a *Privacy Component*. The privacy component is distinct from the smart meter but still resides within the home. It is logically placed in the communication path between the smart meter and the external smart grid entities. They envision that this component could be implemented using common off-the-shelf hardware such as a network router or home gateway. The function of this component is to intercept measurements from the smart meter and calculate the user's energy bill at the end of each billing period using a ToU or dynamic pricing scheme from the energy supplier. This is essentially a form of temporal aggregation because the fine-grained measurements do not leave the user's home. However, this system still requires modification of the smart meters so that they will provide cryptographic commitments (Pederson commitments) with each energy measurement. The privacy component is not necessarily trusted by the energy supplier so the authors propose a protocol in which these commitments are used to verify the calculation of the user's energy bill without revealing the actual measurements. The authors implicitly assume that the privacy component is trusted by the home user because it is part of the user's home network. However, if this component were compromised, an attacker would gain access to all energy measurements from the home. This system only deals with billing and does not facilitate other information flows such as the network monitoring or DR information flows. Jawurek et al. have modelled the behaviour of the system by implementing the core functionality of the privacy component in Java and simulating various input parameters.

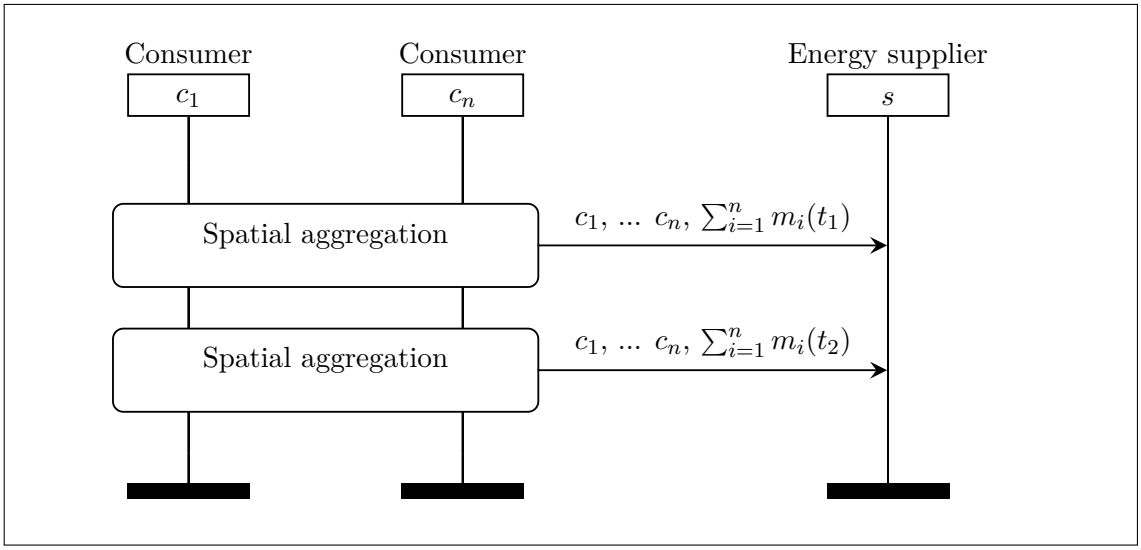

Figure 3.5: Abstract model of a spatial aggregation scheme

3.5.3 Spatial Aggregation

A third approach is to aggregate information from multiple consumers before it is sent to any external smart grid entities. An abstract model of this type of scheme is shown in Figure 3.5 in which the measurements from n consumers (c_1 to c_n) are spatially aggregated using some type of sub-protocol and the result is sent to the supplier. Since the aggregation is performed on the communication links, it is possible to define arbitrary groups of meters spanning multiple geographic areas. For example, measurements from all consumers in a specific area who are on a particular tariff plan could be aggregated if this provides useful information to the energy supplier. By including a sufficient number of consumers, this aggregation prevents an external entity from obtaining detailed information about specific homes. This can be considered to be a stronger form of k-anonymity (as described in the previous chapter) since the recipient cannot determine the contribution of each consumer to the total. Since measurements are still reported with a high frequency, it is possible to use NILM techniques but the results of these cannot be associated with a particular consumer. Spatial aggregation provides stronger privacy protection than basic anonymization of measurements because it is not possible to link specific appliances together under a pseudonym. However, if an external entity is used to perform this spatial aggregation, it is important to consider the degree to which this entity must be trusted by the consumers as well as the external smart grid entities. Used in isolation, spatial aggregation cannot support the billing information flow in which measurements must be connected to a named consumer. Similarly it cannot support the bi-directional DR information flow since it only provides uni-directional communication from the consumers to the energy supplier or DNO.

Garcia and Jacobs [107] have proposed a privacy-friendly metering protocol based on this approach that relies on an additive homomorphic encryption scheme. In their sys-

tem, if there are N meters in a group, each measurement from each meter is split into N components that can be added to obtain the actual measurement. Each component is encrypted with the public key of another meter in the group. Since homomorphic encryption is used, an untrusted external entity can be used to calculate the sum of the components for each meter without learning the actual values. Each meter can then decrypt this summation using its private key. At this point, each meter knows a randomized portion of the total measurement for the group, but this does not correspond to a measurement from any specific meter. These portions can then be sent to the untrusted entity and added to obtain the total measurement for the group. The authors have proved that their protocol is correct and does not leak information. However, compared to the baseline non-privacy preserving protocol, in which each smart meter communicates directly with the energy supplier, this protocol is significantly more complex in terms of its computational requirements. In the baseline protocol, each smart meter performs a single encryption operation per energy measurement, whereas in this protocol, each smart meter must perform N encryption operations per measurement. Since smart meters are generally not high-performance systems, this requirement severely limits the group size N, thus limiting the scalability of the protocol. This protocol also requires the smart meters to perform additive homomorphic encryption and requires external entities, such as the energy supplier, to trust that this will be performed correctly. Barcellona et al. [27] have presented a similar solution in which each measurement is split over multiple consumers. However, due to the high computational requirements of homomorphic encryption, they have instead used a linear threshold secret sharing scheme.

Deng and Yang [82] have presented a similar system that also uses homomorphic encryption and aggregation of measurements to protect user privacy. Each measurement is encrypted using a Paillier homomorphic encryption scheme and sent to the collector via an aggregation path or aggregation tree consisting of other smart meters. At each point in the path or tree, the measurement from the individual meter is added to the total using the additive property of homomorphic encryption. In this approach, every smart meter needs to have the identifiers and public encryption keys of its children and parent. This would require significant configuration effort and so would impact the scalability of this system. Another disadvantage of this approach is that the failure of any node in the aggregation path would result in loss of information from multiple nodes and necessitate reconfiguration of the system. Various similar spatial aggregation protocols based on homomorphic encryption have also been proposed [162, 234, 170, 46, 44, 45, 54].

Ács and Castelluccia [7] have proposed a system that aggregates measurements from multiple consumers and provides differential privacy guarantees for this aggregate. In their system, each meter adds an amount of random noise to mask each measurement. This random noise must be specifically generated using their proposed Distributed Laplacian Perturbation Algorithm so that the noise in the group aggregate follows the correct statistical distribution. Their system then uses a modulo addition-based encryption scheme to encrypt the measurement on each smart meter so that an external entity can only

decrypt the sum of the measurements. Although each meter adds random noise to each measurement, this is not sufficient to provide the required privacy guarantees for each home thus necessitating the use of spatial aggregation. It is argued that the aggregated result received by the external smart grid entities must still be masked by random noise to satisfy the definition of differential privacy. However, this decreases the usefulness of this information for demand forecasting or network optimization. This system only deals with the monitoring information flow in which the frequent spatially-aggregated measurements are sent to the supplier. In order to support the billing and DR information flows, this approach would have to be combined with other techniques.

Kursawe et al. [156] have proposed a system that aggregates measurements over multiple homes and provides an exact result of this aggregation without using a trusted third party. Similarly to the system described by Ács and Castelluccia [7], this system adds a random value to each measurement before it leaves the smart meter. The difference is that these random values are generated such that their sum will be zero or an arbitrary known constant. Kursawe et al. [156] present four protocols for generating these random values including basic interaction between smart meters and Diffie-Hellman key exchange protocols. They have demonstrated the feasibility of their system with a proof of concept implementation on real-world smart meters. Their proposed system does not support the billing or DR information flows and so would also have to be combined with other mechanisms. In order to ensure user privacy, the random values used to mask the measurements must be changed for each measurement. This added complexity could also have an impact on the scalability of the system.

Shi et al. [238] have proposed a similar system in which the random noise added to each measurement sums to zero when aggregating measurements from the whole group. They also acknowledge that these values must be changed for every measurement. However, instead of requiring interaction between the meters to change these masking values, their system relies on cryptographic techniques and a trusted setup process for the group of meters. This process must be performed whenever the group of meters changes. They also argue that revealing the exact aggregate result to an external entity violates differential privacy. Since the aggregating entity is untrusted, they propose a variant of their cryptographic protocol in which random noise is added by each meter. This system does not deal with the billing or DR information flows.

3.5.4 Trusted Third Parties

In some smart grid systems a trusted third party (TTP) is used to facilitate the aggregation of measurements. Since the TTP has access to the exact measurement data from each smart meter, consumers must trust that this entity will not compromise their privacy. Additionally, the other smart grid entities such as the DNO and energy supplier rely on the TTP to provide accurate responses to their queries and so must also trust the TTP in this regard. Given its role in the system, the TTP is a valuable target for attack by an

external party. Therefore all the smart grid entities must also trust that the TTP itself has not been compromised.

Molina-Markham et al. [191] have described a smart grid architecture in which neighbourhood gateways are placed between a group of smart meters and the external entities. These gateways are computational systems that receive the frequent energy measurements from the smart meters, remove the identifying information and then forward the measurements to the smart grid entities. They assume that the TTP is trustworthy but do not provide details of its operation. This system is similar to the basic anonymization approaches discussed above but the use of the TTP strengthens the anonymization since the anonymous measurements originate from the TTP rather than individual consumers. Their system also facilitates billing without compromising user privacy by using cryptographic commitments from the smart meters. In the initialization phase, the smart meter cryptographically commits to a set of tags that will be associated with the subsequent energy measurements. At the end of the billing period, the user computes his or her bill based on the applicable tariff structure and proves the correctness of this to the energy supplier based on the cryptographic commitments.

Bohli et al. [42] have presented a privacy model for smart metering and two sample architectures for protecting user privacy. Their privacy model uses a game-based approach to measure the privacy level provided by a particular smart metering application. Although it uses similar concepts of indistinguishability, their model is less detailed than related work on differential privacy. The first architecture they present uses a TTP to aggregate measurements from a group of smart meters before sending the result to the external smart grid entities. Using their privacy model, they explain that this architecture provides *perfect privacy* since it does not provide any statistical advantage to the external entities for learning individual energy measurements. They also assume that the TTP is trustworthy but do not describe its functionality. However, they note that the TTP may itself be a threat to privacy given the information it contains. Their second proposed architecture does not use a TTP but instead relies on the addition of random noise by each meter. In the summation of the contributions from all the meters, this noise is assumed to approach its expectation value of zero.

A real-world example of the use of a TTP in the smart grid is a fully centralized national smart grid architecture, such as that being developed in the UK. In this architecture, measurements from all smart meters are collected and processed by DCC. As discussed by Anderson and Fuloria [14], this level of centralization introduces various challenges such as the management of large amounts of data.

As seen from the above proposals, the use of a TTP in the smart grid architecture necessarily involves some notion of trust as discussed in the previous chapter.

3.5.5 Other Solutions

Although the majority of research has focussed on the above approaches, there have also been other proposals for protecting consumers' privacy in the smart grid.

McLaughlin et al. [184] have described an algorithm called Non-Invasive Load Levelling (NILL) that aims to eliminate the possibility of a remote entity using NILM techniques to obtain private information. Their system uses an in-residence battery to mask the energy signatures of the various appliances in the home. Their simulations have shown that this can be achieved even with a battery capacity which is lower than the residence's average load.

Similarly, Kalogridis et al. [147] have proposed a power management model using rechargeable batteries and a power mixing algorithm to protect user privacy. Their algorithm is implemented on an entity they call a Load Signature Moderator (LSM), the purpose of which is to detect potential privacy threats and take appropriate actions. In this case, a privacy threat refers to an energy signature that could be used to infer private information. These threats are detected by monitoring the overall energy consumption or by communicating with the various smart home appliances. When a potential privacy threat is detected, the system reconfigures the routing of power within the home, making use of the rechargeable batteries in order to mask the energy signature. Various other approaches using rechargeable batteries have been proposed [145, 146].

Although these approaches demonstrably enhance user privacy, they require significant infrastructure in terms of the batteries and associated control systems. Furthermore, electricity from the grid is supplied as alternating current (AC) whereas batteries of this size are direct current (DC) devices. The AC-DC and subsequent DC-AC conversions required to charge and discharge these batteries incur energy losses and thus reduce the overall benefit of the smart grid.

3.5.6 Trusted Computing in the Smart Grid

From recent literature, there have been various proposals to use TC technologies and approaches in the smart grid. Some of these proposals are specifically related to protecting user privacy.

Zic et al. [286] use the basic security features of TC to provide security for energy services in the home environment. In their approach, each consumer has a USB key with an embedded TPM. By using this key, the consumer's HEMS can be unambiguously identified and securely authenticated by remote entities. This key also facilitates secure communication channels between the HEMS and external smart grid entities.

In the protocol proposed by Garcia and Jacobs [107], it is assumed that the smart meters themselves contain secure hardware elements. These elements provide the basic cryptographic functionality, including asymmetric cryptography, required for their protocol. Specifically, they assume that the private asymmetric keys are not accessible from outside the secure hardware element. This could be achieved using functionality provided

by a TPM. However, they implicitly assume that this functionality will be trusted by all relying parties and do not make use of techniques such as remote attestation.

LeMay et al. [161] have proposed an architecture for an *attested meter* based on TC and virtualization. One of the primary goals of their system is to protect user privacy by reducing the amount of information that must be sent to external smart grid entities. In their system, every smart meter has a TPM and runs multiple virtual machines (VMs). One of the VMs facilitates billing by calculating the consumer's energy bill locally using pricing information from the energy supplier. At the end of the billing period, only the final output is sent to the external entities. Remote attestation is used to prove the correct operation of this VM and thus guarantee the correctness of the output. Additional VMs support DR commands and facilitate user interaction with the smart meter. However, this system does not provide a means to communicate high frequency measurements to the DNO for purposes of demand forecasting or network optimization without compromising user privacy. Even if the attested meter provided these measurements under a pseudonym, this could still present a threat to user privacy as explained in Section 3.5.1.

Another approach using TPMs in smart meters is presented by Petrlic [211]. Similarly to LeMay et al. [161], this system performs the calculation of the user's bill locally in the smart meter. Remote attestation is used to assure the external smart grid entities of the integrity of the software running on the meter. This system attempts to provide a means of reporting the high frequency measurements to the external entities in real time by introducing a TTP. Measurements are sent from the smart meters to the TTP which authenticates that the measurement originated from a legitimate source. The TTP then removes the source's identifying information and forwards the measurement to the DNO and energy supplier. However, the TTP does not perform any aggregation of measurements and so must provide the measurements to the external entities under a pseudonym, thus leaving them vulnerable to the attacks described by Jawurek et al. [139]. Furthermore, the TTP is assumed to be trusted by all entities with which it communicates. The users trust that it will not disclose their identifiable information with the measurements and the external entities trust that the TTP will not accept or send any unauthenticated measurements. The TTP does not provide any technological means of supporting this trust.

Simo Fohm et al. [239] have proposed a *privacy manager* system that allows consumers to be involved in the management of their privacy in the smart grid context. This system consists of a software component running on a so-called Smart Energy Gateway (SEG) within the consumers' premises. TC technologies are used to provide data minimization in order to limit the amount of information that leaves the home. The SEG provides a virtualized environment in which multiple agents can be run, such as a billing agent or home automation agent. The SEG contains all the energy measurements for the home and controls access to these by the various agents through a user-defined privacy policy. The TPM is used to attest to the correct operation of the SEG. It is claimed that the SEG can protect user privacy whilst still reporting high frequency energy measurements using

pseudonyms but no further details or protocols are provided. Inherently, the privacy-preserving characteristics of such a system would be limited since it only has access to measurement information from a single home and so cannot perform spatial aggregation.

A tangentially related use of TC in the broader context of energy distribution and transmission is presented by Wallom et al. [276, 275] as part of the *myTrustedCloud* project. They address the issue of trustworthiness in the cloud computing paradigm and demonstrate how the integration of TC technologies into this paradigm can facilitate security-critical applications such as those of the energy sector. They argue that the lack of trustworthy public cloud infrastructure has limited the use of this paradigm for business-critical applications. They explain how cloud computing can benefit the energy industry by providing the scalable and elastic computational resources required for the complex simulation of energy generation and distribution. However, the high data integrity requirements necessitate the use of trusted infrastructure for this application. Their system facilitates remote attestation of Virtual Machines (VMs), Node Controllers (NCs) and Storage Controllers (SCs) since the integrity of a VM is dependent on all three components. By using an iterative attestation mechanism, the integrity of these three components can be attested using a single attestation session. They have developed a prototype implementation of the system and demonstrated that it is a viable solution for the security requirements of the UK transmission and distribution network business sector. Although this system is related to a separate area of the smart grid, it highlights two important points relevant to this research: Firstly, it highlights the high security requirements for information processing in the energy sector and secondly, it demonstrates that TC technologies and approaches can be used to create systems that meet these requirements.

3.6 Gap Analysis and Summary

Based on the state-of-the-art literature discussed above, this section presents a gap analysis and provides motivation for further research in this field.

The protection of user privacy in the smart grid is widely acknowledged to be an important objective. The risks to user privacy in this context have been explored in various publications such as the NIST Privacy Impact Assessment (PIA) [199]. Various potential solutions for protecting user privacy have been proposed based on different approaches. Basic anonymization approaches have been suggested but have been shown to be vulnerable to specific attacks. Temporal aggregation of measurement information within a consumer's home can protect privacy but significantly reduces the functionality of the smart grid. Aggregation of measurement information over multiple homes has been shown to be a viable option but requires either complex cryptographic operations on the smart meters or the use of a TTP. In cases where a TTP is used, it is assumed to be trusted by all parties without providing any technological basis for this trust. These existing solutions all deal with the flow of information from the smart meter to the external entities since

this is the most obvious threat to user privacy at present. However, very little research addresses the problem of protecting user privacy whilst facilitating fully bi-directional communication between the consumers and the external smart grid entities as required for the DR information flow.

The technologies and approaches from the field of Trusted Computing (TC) have been developed for use mainly in the enterprise context. Very recent literature has suggested the use of these technologies in the smart grid context but is focussed almost exclusively on the use of TC in the smart meters themselves. Apart from the practical challenges this presents, these approaches have not provided a clear solution for using TC to protect user privacy.

Therefore a viable research opportunity exists to advance the state-of-the-art by combining these two streams of research. It is hypothesized that TC approaches and technologies can provide significant benefit in the smart grid context if applied to a Trustworthy Remote Entity (TRE) rather than to individual smart meters. As an intermediary in the communication paths between the consumers and the external entities, the TRE can be used to protect user privacy whilst maintaining the primary functionality of the smart grid. Using TC technologies and approaches, this system can provide a technological basis to support the trust placed in it by the relying parties. Furthermore, it is anticipated that the concept of the TRE can be used to provide benefit in contexts beyond the smart grid. The smart grid serves as a relevant real-world case study to support the investigation, development and evaluation of this technology.

Chapter 4

Modelling and Analysing Privacy Properties

This chapter is based on the following technical report:

- A. J. Paverd, A. P. Martin, & I. Brown, "Modelling and Automatically Analysing Privacy Properties for Honest-but-Curious Adversaries with Applications in the Smart Grid" Technical Report, 2013 [209].

4.1 Background and Related Work 56
4.2 Definitions 57
4.3 Adversary Model 62
4.4 Integration with Existing Methods 70
4.5 Evaluation 75
4.6 Summary 84

As shown in the previous chapter, security and privacy are of critical concern in the smart grid. However, most of the security and privacy vulnerabilities described in that chapter arose from the overall design of the communication system, rather than from flaws in the implementation. The design of such a system necessarily includes at least one *communication protocol*, which specifies how information is transferred between participants. If the communication protocols used in a system do not provide certain security and privacy properties, all implementations of the system will exhibit security and privacy vulnerabilities. Of course vulnerabilities could also arise from the implementation of the system but this is arguably an orthogonal issue. In order to enhance communication privacy, it is necessary to design and use communication protocols that provide adequate privacy guarantees. At the same time, these protocols must also provide sufficient security guarantees to support the functionality of the system. It is therefore necessary to have concrete definitions of these security and privacy properties and a means of evaluating communication protocols in terms of these definitions.

Two important properties of communication privacy are *undetectability* and *unlinkability*, since these can be used to reason about the more complex concepts of anonymity and privacy [212, 273, 52]. For example, anonymous communication systems aim to prevent adversaries from detecting any identifying information or linking items of interest to specific participants [281]. These types of privacy properties are relevant to the smart grid [98, 226] and other application domains, such as Radio Frequency Identification (RFID) communication [17, 51].

As communication protocols become more complicated, it becomes more difficult to ensure that they provide the required privacy properties. This has been the case for security properties, such as secrecy and authentication, and has led to the development of various security protocol analysis methodologies and tools. Similarly, there is a need for automated formal analysis of privacy properties [180] for which various approaches have been proposed and used, as described in Section 4.1.

In the symbolic paradigm, privacy properties such as undetectability and unlinkability have previously been modelled using the observational equivalence approach, in which two or more sequences of events (traces) are compared from the perspective of the adversary. Only one of the traces contains the event or item of interest. If the traces are observationally equivalent from the adversary's perspective, the event or item is undetectable. Similar formulations are used to model unlinkability and other privacy properties. An alternative approach is to formulate these properties as reachability assertions. In this approach, the adversary is modelled as a deductive system with a set of inference rules. The privacy properties are violated if the adversary can use these rules to make specific deductions about the item of interest. In contrast to the observational equivalence approach, the reachability approach requires a more complicated adversary model in order to accurately capture the adversary's capabilities. However, the reachability approach has been successfully used for modelling and analysing security properties, such as secrecy and authentication. A significant advantage of using the reachability approach for analysing

privacy properties is that it can be combined with the analysis of security properties. Furthermore, if correctly constructed, these two types of analysis can be performed using the same formal model of the protocol in order to capture the interplay between the security and privacy requirements.

In some cases, the security and privacy goals of a single communication protocol appear to conflict with each other. For example, certain smart grid privacy challenges could be overcome by simply anonymizing all measurements from smart meters. However, without authentication of the smart meters, the system would be vulnerable to attacks such as false data injection. In an ideal voting system (which can be modelled as a communication protocol [194]) the privacy requirement is for the votes to be anonymous. However, some form of authentication is needed to enforce that only registered voters can vote and that each voter can only cast a single vote. In digital subscription services, the service provider will only allow authorized users to access the content, but to protect users' privacy, the provider should not learn what content a particular user has accessed [245]. In the above examples, the security and privacy goals are not mutually exclusive, but the tension between them could lead to either security or privacy flaws in the communication protocols if they are not analysed correctly. Existing approaches for verifying privacy properties, especially those based on observational equivalence, are often performed separately from the analysis of security properties. However, independent verification might not always capture this interplay between security and privacy. Therefore, it is advantageous to define a single model of a communication protocol and analyse this with respect to both security and privacy properties. The interplay between security and privacy properties also gives rise to the need for multiple different adversary models in the analysis. For example, it is desirable to verify the security properties with respect to a strong adversary model, such as the Dolev-Yao (DY) model [85], but when considering privacy properties, the adversary is often a legitimate participant in the protocol with whom the other participants are required to communicate and so is more accurately modelled as a *semi-honest* [117] or *honest-but-curious (HBC)* adversary.

This chapter aims to fill this gap by presenting a new methodology and a supporting software tool for modelling and automatically analysing privacy properties in communication protocols, such as those used in the smart grid. It begins by presenting definitions for the properties of undetectability and unlinkability and explains how these can be applied with respect to the HBC adversary. It then describes an approach for formalizing these properties as reachability assertions in a deductive system and representing them in the process algebra of Communicating Sequential Processes (CSP). In order to generate these CSP representations, the Casper/FDR protocol analysis tool, originally developed by Lowe [167], has been enhanced to model and analyse these two privacy properties. In this new *Casper-Privacy*[1] tool, both the security and privacy properties are analysed with respect to the same CSP protocol model. The HBC adversary model and the Casper-

[1]Although not indicated in the name, the Casper-Privacy tool still uses the same FDR model checker as the Casper/FDR tool.

Privacy tool have been evaluated by re-analysing three protocols from the literature and comparing the results. In order to evaluate the effectiveness of this tool in the smart grid context, a further four smart meter communication protocols have been analysed in terms of their security and privacy properties.

4.1 Background and Related Work

Formal analysis of the security properties of communication protocols is an established area of research. In addition to the work by Roscoe [223] and Lowe [168, 167] using CSP and the Casper/FDR tool [167] there have been many other developments in this field [53, 4, 240, 69, 41, 185]. There have also been related research efforts to formalize the concepts of privacy and anonymity and their constituent properties, including undetectability and unlinkability. A comprehensive review of the formalization of anonymity has been presented by Wright et al. [281].

Previous research has focussed on the formalization and analysis of privacy properties in both the computational and symbolic paradigms. In the computational paradigm, the notion of computational indistinguishability has been used to reason about properties such as unlinkability and anonymity [251, 244, 55, 105, 102, 24, 23]. There are also various examples of the use of the symbolic paradigm: Mauw et al. have developed a formalization of anonymity in onion routing using observational equivalence [180]. Berthold and Clauss [39] use formal concept analysis techniques to reason about unlinkability. The applied pi calculus and the ProVerif tool have been used to verify privacy-type properties for various types of protocols including voting protocols [154, 81] and the Direct Anonymous Attestation (DAA) protocol [48, 81]. There have been efforts to verify privacy properties in the protocols used by RFID tags [17, 16, 51]. In terms of modelling undetectability and unlinkability, the methodology presented in this chapter is most closely related to the work by Veeningen et al. [274, 273, 272], in which undetectability and unlinkability are modelled as reachability assertions. This chapter builds on their definitions and extends these to include other deductions that could be made by the HBC adversary. However, unlike previous work, the methodology in this chapter has been specifically designed to be directly integrated with the analysis of the security properties.

Fournet and Abadi [103] presented one of the earliest examples of combining the analyses of security and privacy properties. They used the applied pi calculus [3] to analyse a private authentication protocol. They verified the security properties (authentication and secrecy) and one specific privacy requirement: that external observers are unable to learn the identities of the protocol's participants. This requirement can be represented in terms of undetectability and unlinkability. The methodology presented in this chapter differs in that it enables the analysis of undetectability and unlinkability properties for any information item with respect to any participant, thus allowing it to be applied to other types of communication protocols.

More recently, Luu et al. [171] presented *SeVe*, a tool for automatically verifying se-

curity properties that has been implemented as a module of the Process Analysis Toolkit (PAT). They consider the security properties of secrecy and authentication as well as three privacy properties: anonymity, receipt freeness and coercion resistance. These properties can be verified automatically with respect to an external intruder. Although closely related, the methodology presented in this chapter differs in terms of the specific privacy properties that can be analysed, and that this analysis takes place with respect to an internal HBC adversary, who is a legitimate participant in the protocol, rather than an external intruder.

4.2 Definitions

In order to analyse communication protocols, it is necessary to have concrete formal definitions of the desired privacy properties and of the relevant adversary model. This section presents the formal definitions of the *honest-but-curious adversary* and the privacy properties of *undetectability* and *unlinkability*.

4.2.1 Honest-But-Curious Adversary

For protocol analysis in general, the most well-known adversary model is the so-called Dolev-Yao (DY) model [85]. The DY model is also one of the strongest possible models in terms of adversary capabilities. In the ideal case, security and privacy properties would be maintained even in the presence of a DY adversary. However, in some cases, the DY model is too strong to be used as a realistic representation of certain participants. For example, in the smart grid, the energy supplier could not realistically be modelled as a DY adversary. In reality, various factors limit the actions that the energy supplier may perform including regulations, audits, oversight and desire to maintain reputation. However, although a DY model is not appropriate in this case, it does not necessarily mean that the energy supplier is not adversarial. To capture this adversarial behaviour, this agent can be modelled as a *semi-honest* [117] or *honest-but-curious (HBC)* adversary, which is defined as follows:

Definition 6 (Honest-But-Curious Adversary)**.** *The honest-but-curious (HBC) adversary is a legitimate participant in a communication protocol who will not deviate from the defined protocol but will attempt to learn all possible information from legitimately received messages.*

In comparison to the DY model, the HBC model is more limited in that it will not deviate from the protocol and cannot send any falsified messages. The HBC adversary is also more limited than a passive DY adversary, since the HBC adversary cannot eavesdrop on arbitrary communication channels and can only receive messages for which it is the intended recipient. However, the strength of the HBC model is that, unlike a DY adversary, many protocols will require participants to communicate directly with the HBC adversary.

Arapinis et al. [15] and Cheval et al. [63] have both suggested that the HBC adversary model is too weak to represent certain types of real world adversaries. They both propose

to use the *malicious-but-cautious* adversary model, in which the adversary can perform active attacks but will only perform attacks that do not leave any verifiable[2] evidence of the adversary deviating from the protocol. This is similar to the *covert* adversary model [21], which is sometimes used in the analysis of Secure Multiparty Computation (SMC) protocols. The malicious-but-cautious adversary model is useful because it affords the adversary more capabilities than the HBC model but is more realistic than the DY model. If a protocol is secure against the malicious-but-curious adversary, it will be secure against all weaker adversaries, including the HBC adversary.

However, one difficulty with using the malicious-but-curious model in a real world context is to determine the adversary's precise capabilities. In reality, the detectability of actions is usually probabilistic and it is very unlikely that any action is perfectly undetectable for all possible observers. For example, the adversary may perform certain deviant actions that are undetectable to the other participants in the protocol because these participants can only interact with the adversary over the network. However, if the adversary's systems were physically audited, many of these actions are likely to be detected. Going a step further, even more of these actions will be discovered if the adversary's employees are required to testify in a court of law. Therefore, the actions this adversary may perform are dependent on the capabilities of all observers, and thus these must be specified in the model. Furthermore, in a real world context, the resources of the adversary affect which of its actions will be detected. An adversary with minimal resources is required to carefully select actions that cannot be detected whereas a well resourced adversary could, for example, bribe the auditors or the final adjudicator in order to avoid detection. Certain powerful adversaries are even able to use legal mechanisms to silence anyone who may be able to detect the their deviant actions. Therefore, in this model, the adversary's capabilities are also determined by its level of resources and/or influence, thus making this adversary model relatively complicated to use in a realistic setting.

In contrast, the weaker HBC model avoids this complexity since the HBC adversary's capabilities are not defined in terms of detectability. If the malicious actions of an HBC adversary were detected, it will make the adversary unpopular with the other participants, but since the adversary has not deviated from the protocol, further consequences are usually unlikely. If the other participants dislike the HBC adversary's actions, it indicates that the protocol itself is flawed or that these participants have unrealistically high expectations of the actual security and privacy guarantees provided by the protocol. Therefore, the methodology presented in this chapter uses the HBC adversary model, but future work may extend this by including the malicious-but-cautious and covert adversary models.

[2]In this context, the term *verifiable* is taken to mean that the adversary's deviant actions can be detected by some observer and that the evidence of these actions is sufficient to convince a reasonable adjudicator. It does not imply formal verification in the mathematical sense.

HBC Formalization

Formally, the knowledge of an HBC adversary $\mathcal{A}$ is represented by two disjoint sets of information called *views*:

- The adversary's *external view* $\mathcal{A}_\mathcal{E}$ contains all the information that the adversary receives from external sources as well as all permitted decompositions of this information. This set therefore represents all information that, from the adversary's perspective, is known to at least one other participant in the protocol. At the start of the analysis, this view is empty.
- The adversary's *internal view* $\mathcal{A}_\mathcal{I}$ contains the adversary's initial knowledge and any information the adversary infers during the protocol. This set represents the adversary's private knowledge and inferences, which are not known to other participants. At the start of the analysis, this view contains only the adversary's initial knowledge.

Therefore, both $\mathcal{A}_\mathcal{E}$ and $\mathcal{A}_\mathcal{I}$ are dependent on the current state of the system. As the HBC adversary receives new messages or completes more runs of the protocol, more information items and inferences are added to these views according to the set of inference rules described in the next section. The external and internal views are always disjoint ($\mathcal{A}_\mathcal{E} \cap \mathcal{A}_\mathcal{I} = \emptyset$) and a shorthand for the union of these two views is $\mathcal{A}_{\mathcal{EI}} = \mathcal{A}_\mathcal{E} \cup \mathcal{A}_\mathcal{I}$.

4.2.2 Undetectability

Pfitzmann and Hansen have proposed a widely-cited terminology for privacy [212]. They define undetectability as: *"Undetectability of an item of interest from an attacker's perspective means that the attacker cannot sufficiently distinguish whether it exists or not."*

Veeningen et al. [273, 272] use this definition to develop a formal model of undetectability. They divide all information items into three disjoint sets: the set of participating entities (e.g. natural persons), the set of identifiers (e.g. usernames or network addresses) and the set of data items included in the exchanged messages. A data item is detectable by a participant if it forms part of that participant's view of the system [273, 272].

The methodology in this chapter is based on the definition of undetectability by Pfitzmann and Hansen [212] and uses a similar approach to Veeningen et al. [273, 272] to reach a concrete instantiation of this definition. However, no distinctions are made between identifiers and data items as these distinctions would be specific to the protocol and the context in which it is used. Therefore, in this methodology, any distinct piece of information in the communication protocol is simply referred to as an *information item*. Since the HBC adversary is the legitimate recipient, the definition is modified slightly to focus on *sender undetectability*, which is achieved if the adversary cannot sufficiently distinguish whether or not an item of interest exists outside of its own knowledge (i.e. is known by another participant). Unless otherwise specified, the term *undetectability* refers to *sender undetectability*. The term *exists* is interpreted to mean that the item is known by at least

one of the participants in the protocol. The adversary can determine that an item exists either by direct observation (e.g. the item appears in $\mathcal{A}_{\mathcal{E}}$) or by some process of deduction using the inference rules described in the next section. This definition does not require the adversary to know the information item in order to detect that it exists. Using the notation $detect(i_1)$ to mean that the adversary has detected or inferred the existence of i_1, the property of undetectability is defined as a reachability assertion as follows:

Definition 7 (Undetectability)**.** *From the perspective of an HBC adversary $\mathcal{A}$, an information item i_1 is undetectable if and only if a deductive system accurately representing the capabilities of $\mathcal{A}$ cannot reach the conclusion $detect(i_n)$.*

4.2.3 Unlinkability

Pfitzmann and Hansen [212] define unlinkability as: *"Unlinkability of two or more items of interest from an attacker's perspective means that, within the system, the attacker cannot sufficiently distinguish whether these items are related or not."*

Veeningen et al. [273, 272] again use this definition to present a formal model of unlinkability. In their model, each identifier is associated with a single entity and data items can be associated (linked) with a particular identifier. Therefore, if an agent can link two data items to the same identifier, these data items can be linked to each other. By extension, if two data items are linked to each other and one of these can be linked to a specific identifier, then the other can also be linked to that identifier. The methodology in this chapter uses a similar approach to reach a concrete instantiation of the definition by Pfitzmann and Hansen [212], but as with undetectability, it does not distinguish between data items and identifiers.

As with undetectability, the HBC adversary is the legitimate recipient and thus this definition is modified slightly to focus on *sender unlinkability*, which means that the adversary cannot sufficiently distinguish whether or not two or more items of interest are related because they originated from the same sender. Unless otherwise stated, the term *unlinkability* is used to refer to *sender unlinkability*. Sender unlinkability is often a necessary requirement for achieving sender anonymity, which is the main objective of most anonymous communication protocols. For example, in the smart metering protocols described in the previous chapter, the concern is that fine-grained energy usage measurements from smart meters could be linked to specific individuals, thus compromising their privacy [125, 191, 50, 122, 71, 218]. Various proposed smart meter communication protocols aim to achieve sender anonymity [89, 43, 243]. Sender anonymity is trivially compromised if items can be linked to a specific individual. Furthermore, it might also be compromised if multiple items can be linked to each other as having originated from the same sender. Even if the identity of the sender is initially unknown, these linked items might form a pattern of behaviour that can sometimes be used to de-anonymize the sender. For example, in the smart metering protocol by Efthymiou and Kalogridis [89], all measurements from a specific smart meter are linked together by a unique pseudonym. Jawurek et al. [139]

have explained how these linked measurements form a pattern of behaviour that can be compared to that of real users (e.g. by correlating periods when the house is unoccupied) and used to de-anonymize the measurements. There have been various other examples of de-anonymization based on behavioural patterns, which ultimately result from a lack of sender unlinkability [193, 192].

In order to model unlinkability in protocols, it is necessary to assume that all information is explicitly represented in the description of the protocol. For example, if a communication protocol is implemented in a real system, the implementation might introduce additional identifying information, such as an IP address or device identifier, that is not specified in the protocol and thus cannot be evaluated in the analysis. However, this could be avoided by running the protocol over an anonymity network such as TOR [84], or some other type of anonymous communication channel that does not include inherent network identifiers. In the protocol models presented in this thesis, if network identifiers are not explicitly included, it should be assumed that the protocol takes place over an anonymous communication channel.

Given a set of information items $\mathcal{I}$, the aim of the adversary is to determine which of these items can be linked together as originating from the same sender. If the adversary receives a message, m, which is a sequence of information items $i_1...i_n$, the adversary can link these information items together as originating from the same sender by virtue of being in the same message. Two messages, m_1 and m_2, can be linked to the same sender (and thus to each other) if they share one or more information items that are only known to a single participant other than the HBC adversary by the following deductive reasoning: Given a set of participating entities $\mathcal{E}$, if messages m_1 and m_2 both contain a specific information item that is only known by a subset of entities ($\mathcal{E}' \subseteq \mathcal{E}$), then both m_1 and m_2 must have originated from members of $\mathcal{E}'$. If $\mathcal{E}'$ contains only a single entity, all the information items in m_1 and m_2 can be definitively linked to each other and to this entity. If $\mathcal{E}'$ contains two entities, one of which is the HBC adversary, the adversary can exclude its own messages and link all remaining messages to the other entity in $\mathcal{E}'$. Similar relationships can exist for larger cardinalities of $\mathcal{E}'$ if there are multiple colluding HBC adversaries. This interpretation of unlinkability is similar to that used by Berthold and Clauss [39] and Veeningen et al. [273, 272].

Using the notation $link(i_m, i_n)$ to mean that the adversary has observed or inferred that items i_m and i_n originated from the same sender and are thus linked to each other, the property of unlinkability is defined as a reachability assertion as follows:

Definition 8 (Unlinkability)**.** *Information items i_m and i_n are unlinkable by HBC adversary $\mathcal{A}$ if and only if a deductive system accurately representing the capabilities of $\mathcal{A}$ cannot reach the conclusion $link(i_m, i_n)$.*

Where $link(i_m, i_n)$ represents a symmetric, transitive binary relation between two information items, indicating that they originated from the same sender:

$$\forall\, i_m, i_n \bullet link(i_m, i_n) \Rightarrow link(i_n, i_m)$$
$$\forall\, i_x, i_y, i_z \bullet link(i_x, i_y) \wedge link(i_y, i_z) \Rightarrow link(i_x, i_z)$$

4.3 Adversary Model

This section presents the formal model of the HBC adversary's capabilities. The adversary is represented as a deductive system consisting of a set of inference rules. Section 4.3.1 defines the components and notation used in this model. It also discusses two important concepts used in the model, namely anonymous and probabilistic encryption. Section 4.3.2 describes the deductive system and presents the set of inference rules that form the core of this model.

4.3.1 Components and Notation

Since undetectability and unlinkability are formulated as reachability assertions, the fundamental components of this model are similar to those used in other reachability-based methodologies, such as the analysis of the security properties of secrecy and authentication. As is usually the case when analysing security properties, this model assumes ideal representations of all cryptographic primitives. The inference rules in the model are represented using *structural operational semantics* and the signatures of these are shown in Table 4.1.

As shown in Table 4.1, an inference rule consists of a set of premises (statements above the line) and a set of conclusions (statements below the line). If all premises of a particular rule are true, then all conclusions of that rule are also true. The notation $\mathcal{A}_\mathcal{X}$ denotes any one of the views of adversary $\mathcal{A}$, namely the external view $\mathcal{A}_\mathcal{E}$, the internal view $\mathcal{A}_\mathcal{I}$ or the union of these two $\mathcal{A}_{\mathcal{EI}}$. The term i_n denotes an information item, which can be any piece of information that is present in the protocol, including participant identities, cryptographic keys, plaintext data, ciphertext data, cryptographic hashes or any combination thereof. $\mathcal{A}_\mathcal{X} \vdash i_n$ denotes that i_n is contained within view $\mathcal{A}_\mathcal{X}$. $receive(i_n)$ denotes the adversary receiving a message i_n. As previously defined, $detect(i_n)$ denotes that the adversary has detected the existence of i_n and $link(i_m, i_n)$ means that the adversary has established a link between i_m and i_n. $E(k, i_n)$ denotes symmetric encryption of i_n using key k so that it can only be decrypted using the same key k. $E(k_+, i_n)$ denotes asymmetric encryption of i_n using key k_+ so that it can only be decrypted using key k_-. Since k_+ is used to represent a public key, k_- is a private key and thus $E(k_-, i_n)$ denotes a signature of item i_n that

Table 4.1: Components and notation used in the model

$$rule ::= \frac{premises}{conclusions}$$

$$premises ::= statement\ [;\ premises]$$
$$conclusions ::= statement\ [;\ conclusions]$$

$$statement ::= \mathcal{A}_{\mathcal{X}} \vdash i_n \mid receive(i_n) \mid detect(i_n) \mid link(i_m, i_n)$$

$$\mathcal{A}_{\mathcal{X}} ::= \mathcal{A}_{\mathcal{E}} \mid \mathcal{A}_{\mathcal{I}} \mid \mathcal{A}_{\mathcal{EI}}$$

$$i_n ::= bit\ string \mid \{i_n,\ i_n\ [,\ i_n]\} \mid E(k, i_n) \mid E(k_-, i_n) \mid E(k_+, i_n) \mid E_A(k_+, i_n) \mid E_P(k_+, i_n) \mid E_{AP}(k_+, i_n) \mid H(i_n)$$

$$k \in symmetric\ keys$$
$$k_-,\ k_+ \in asymmetric\ keys$$

can be verified using key k_+. $E_A(k_+, i_n)$ denotes anonymous (key-private) encryption, $E_P(k_+, i_n)$ denotes probabilistic encryption and $E_{AP}(k_+, i_n)$ denotes the combination of the two. $H(i_n)$ denotes the one-way cryptographic hash of i_n.

Anonymous Encryption

Anonymous encryption refers to an encryption scheme which, in addition to the usual security properties, also provides *anonymity* or *key-privacy* as described by Bellare et al. [34]. In the context of a public-key encryption scheme, the adversary will usually have access to the set of public keys with which a message could have been encrypted. In an anonymous encryption scheme, the adversary is unable to determine which of these keys was used to encrypt the message. Symmetric encryption schemes satisfy this property by default because the adversary should not have access to the secret key. As described by Kohlweiss et al. [152], this property is highly relevant for receiver unlinkability and receiver anonymity since the adversary is unable to deduce the intended recipient of an encrypted message. However, this is also relevant to sender unlinkability in a more general asymmetric encryption scheme since it ensures that an adversary is unable to make linkability deductions about two messages that might have been encrypted using the same key. In this chapter, anonymous encryption is denoted using the $E_A(...)$ notation.

Probabilistic Encryption

Probabilistic encryption, as introduced by Goldwasser and Micali [118], describes an encryption scheme with the property that: *"Whatever is efficiently computable about the cleartext given the ciphertext, is also efficiently computable without the ciphertext."* Prob-

abilistic encryption introduces a degree of randomness into the encryption scheme so that multiple encryptions of the same message with the same key will result in different ciphertexts. In order to be considered *semantically secure* [118, 119], an encryption scheme must be probabilistic. This concept can be applied to both symmetric and asymmetric encryption. Probabilistic encryption is important when considering undetectability and unlinkability. If a deterministic (i.e. non-probabilistic) encryption scheme is used, an adversary who observes the same ciphertext multiple times could deduce that these represented the same message encrypted under the same key, even without decrypting the messages. If a probabilistic scheme were used in this scenario, the adversary would instead observe multiple different ciphertexts. Some encryption schemes, such as ElGamal and Paillier, are probabilistic by default whereas others, such as RSA, can be made probabilistic by adding randomized padding, such as the Optimal Asymmetric Encryption Padding (OAEP) scheme [33], to the message before encryption. In this chapter, probabilistic encryption is denoted using the $E_P(...)$ notation and the combination of anonymous and probabilistic encryption uses the $E_{AP}(...)$ notation.

4.3.2 Deductive System

In this reachability-based approach, the HBC adversary is represented as a deductive system consisting of a set of axioms and a set of inference rules that are used to reach logical deductions. At any point in the protocol, the set of axioms consists of the adversary's initial knowledge and the knowledge the adversary has gained through receiving messages. Based on these axioms, the adversary applies the inference rules in an attempt to reach new conclusions. A particular inference rule can be applied if all the premises of the rule are satisfied. The resulting conclusions from a successful application of an inference rule can be used to satisfy the premises of other inference rules. For example, due to the transitive nature of the link relationship, the adversary can create a graph of links, based on multiple deductions, in order to test for the existence of a path between items that are supposedly unlinkable. The overall aim of this deductive system is to determine whether the properties of undetectability and unlinkability hold for specific information items.

Honest Participant Rules

The following inference rules describe the permissible behaviour of an honest participant in the protocol, using the semantics defined in the previous section. As a legitimate participant in the protocol, the HBC adversary also has these capabilities and thus the rules below are presented from the perspective of the adversary:

Honest participant - rule 1 (Received message)

$$\frac{receive(i_1)}{\mathcal{A}_{\mathcal{E}} \vdash i_1}$$

Any message received by the adversary must have originated from another participant (honest or otherwise) in the protocol and so is added to the adversary's external view.

Honest participant - rule 2 (Decomposition - external)

$$\frac{\mathcal{A}_{\mathcal{E}} \vdash \{i_1, i_2\}}{\mathcal{A}_{\mathcal{E}} \vdash i_1;\ \ \mathcal{A}_{\mathcal{E}} \vdash i_2}$$

The adversary can decompose messages in the external view into their sub-terms, which remain in the external view since they must be known by another participant.

Honest participant - rule 3 (Decomposition - internal)

$$\frac{\mathcal{A}_{\mathcal{I}} \vdash \{i_1, i_2\}}{\mathcal{A}_{\mathcal{I}} \vdash i_1;\ \ \mathcal{A}_{\mathcal{I}} \vdash i_2}$$

A message in the internal view must have been composed by the adversary and thus can only be decomposed into the internal view.

Honest participant - rule 4 (Composition)

$$\frac{\mathcal{A}_{\mathcal{EI}} \vdash i_1;\ \ \mathcal{A}_{\mathcal{EI}} \vdash i_2}{\mathcal{A}_{\mathcal{I}} \vdash \{i_1, i_2\}}$$

The adversary can create compositions of terms but can only add these to the internal view.

Honest participant - rule 5 (Symmetric decryption - external)

$$\frac{\mathcal{A}_{\mathcal{E}} \vdash E(k, i_1);\ \ \mathcal{A}_{\mathcal{EI}} \vdash k}{\mathcal{A}_{\mathcal{E}} \vdash i_1}$$

If the correct key is known, the adversary can perform symmetric decryption of a received term (i.e. in the external view), adding the decrypted information item to the external view.

Honest participant - rule 6 (Symmetric encryption)

$$\frac{\mathcal{A}_{\mathcal{EI}} \vdash i_1;\ \ \mathcal{A}_{\mathcal{EI}} \vdash k}{\mathcal{A}_{\mathcal{I}} \vdash E(k, i_1)}$$

The adversary can create symmetrically encrypted terms and add these to the internal view.

Honest participant - rule 7 (Symmetric decryption - internal)

$$\frac{\mathcal{A}_{\mathcal{I}} \vdash E(k, i_1);\ \ \mathcal{A}_{\mathcal{EI}} \vdash k}{\mathcal{A}_{\mathcal{I}} \vdash i_1}$$

A symmetrically encrypted term in the internal view must have been generated by the adversary and thus can only be decrypted into the internal view.

Honest participant - rule 8 (Asymmetric decryption - external)

$$\frac{\mathcal{A}_{\mathcal{E}} \vdash E_{[AP]}(k_+, i_1);\ \ \mathcal{A}_{\mathcal{EI}} \vdash k_-}{\mathcal{A}_{\mathcal{E}} \vdash i_1}$$

If the correct key is known, the adversary can perform asymmetric decryption of a received term (i.e. in the external view), adding the decrypted information item to the external view.

Honest participant - rule 9 (Asymmetric encryption)

$$\frac{\mathcal{A}_{\mathcal{EI}} \vdash i_1;\ \ \mathcal{A}_{\mathcal{EI}} \vdash k_+}{\mathcal{A}_{\mathcal{I}} \vdash E_{[AP]}(k_+, i_1)}$$

The adversary can create asymmetrically encrypted terms and add these to the internal view.

Honest participant - rule 10 (Asymmetric decryption - internal)

$$\frac{\mathcal{A}_{\mathcal{I}} \vdash E_{[AP]}(k_+, i_1);\ \ \mathcal{A}_{\mathcal{EI}} \vdash k_-}{\mathcal{A}_{\mathcal{I}} \vdash i_1}$$

An asymmetrically encrypted term in the internal view must have been generated by the adversary and thus can only be decrypted into the internal view.

Honest participant - rule 11 (Hash)

$$\frac{\mathcal{A}_{\mathcal{EI}} \vdash i_1}{\mathcal{A}_{\mathcal{I}} \vdash H(i_1)}$$

The adversary can perform deterministic cryptographic hash operations on known information items and add the results to the internal view.

HBC Adversary Rules

In addition to the capabilities of an honest participant, the HBC adversary is also capable of making inferences based on received information. These additional capabilities are modelled by the additional HBC inference rules below, which are derived from the definitions of undetectability and unlinkability in the previous section:

HBC adversary - rule 1 (Detectability)

$$\frac{\mathcal{A}_{\mathcal{E}} \vdash i_1}{detect(i_1)}$$

By definition, any information item in the adversary's external view can be detected by the adversary since it must be known to at least one other participant in the protocol.

HBC adversary - rule 2 (Link symmetry)

$$\frac{link(i_1, i_2)}{link(i_2, i_1)}$$

The ordering of terms in a link can be reversed since this is defined as a symmetric relation.

HBC adversary - rule 3 (Link transitivity)

$$\frac{link(i_1, i_2);\ \ link(i_2, i_3)}{link(i_1, i_3)} \quad \forall\, i_2 \neq E_P(...)$$

The transitive nature of the link relation allows the construction of chains of links. However, undecryptable probabilistically encrypted terms cannot be used as the basis for links because their values change on each encryption, even if the same information item is encrypted with the same key (as explained in Section 4.3.1).

HBC adversary - rule 4 (Compositional link)

$$\frac{\mathcal{A}_{\mathcal{E}} \vdash \{i_1, i_2\}}{link(i_1, i_2)}$$

A sequence of terms in the adversary's external view represents a message. All direct subterms of a message can be linked together as they must have been known by the message's sender. This type of rule is also used by Berthold and Clauss [39].

HBC adversary - rule 5 (Symmetric decryptable)

$$\frac{\mathcal{A}_{\mathcal{E}} \vdash E(k, i_1);\ \ \mathcal{A}_{\mathcal{EI}} \vdash k}{detect(k);\ \ link(i_1, k);\ \ link(k, E(k, i_1))}$$

From a symmetric encryption term that can be decrypted, the adversary detects both the decrypted information item i_1 (by the symmetric decryption and detectability rules above) and the key k, and links these to each other and to the encryption term since these must all have been known by the participant who created the encryption term. By linking the key and the decrypted item to the encrypted term, these items can be linked to any other items in the same message by the link transitivity and compositional link rules above.

HBC adversary - rule 6 (Asymmetric decryptable)

$$\frac{\mathcal{A}_{\mathcal{E}} \vdash E_{[AP]}(k_+, i_1);\ \ \mathcal{A}_{\mathcal{EI}} \vdash k_-}{detect(k_+);\ \ link(i_1, k_+);\ \ link(k_+, E_{[AP]}(k_+, i_1))}$$

From an asymmetric encryption term that can be decrypted, the adversary detects both the decrypted information item i_1 (by the asymmetric decryption and detectability rules above) and the encryption key k_+, Event though k_+ might not be known, the adversary can be sure it exists. The decrypted item and the key are linked to each other and to the encryption term because these must all have been known by the agent who created the encryption term.

HBC adversary - rule 7 (Asymmetric undecryptable)

$$\frac{\mathcal{A}_{\mathcal{E}} \vdash E(k_+, i_1);\ \ \mathcal{A}_{\mathcal{EI}} \vdash k_+}{detect(k_+);\ \ link(E(k_+, i_1), k_+)}$$

From an asymmetric encryption term that cannot be decrypted, the adversary detects the encryption term by the detectability rule above. Since anonymous encryption (key-privacy) has not been used, the adversary can also determine which key k_+, out of a known set of keys, was used to create the encryption term and thus k_+ is detected and linked to the encryption term. If anonymous encryption had been used, it would not be possible to detect or link k_+.

HBC adversary - rule 8 (Probabilistic asymmetric undecryptable)

$$\frac{\mathcal{A}_{\mathcal{E}} \vdash E_P(k_+, i_1);\ \ \mathcal{A}_{\mathcal{EI}} \vdash k_+}{detect(k_+)}$$

From a probabilistic asymmetric encryption term that cannot be decrypted, the adversary can again determine which key k_+, out of a known set of keys, was used to create the encryption term and thus k_+ is detected. Unlike above, the adversary cannot use the encryption term as the basis for any links due to the use of probabilistic encryption. Again, if anonymous encryption had been used, the adversary would not have been able to detect k_+.

HBC adversary - rule 9 (Probabilistic asymmetric undecryptable in a message)

$$\frac{\mathcal{A}_{\mathcal{E}} \vdash i_1, E_P(k_+, i_2);\ \ \mathcal{A}_{\mathcal{EI}} \vdash k_+}{detect(k_+);\ \ link(i_1, k_+)}$$

This is similar to the previous rule, except that if the encryption term is received as part of a message, the adversary can still link the detected encryption key k_+ to the rest of the message since this is not covered by the probabilistic encryption. This additional rule is required in the non-anonymous case since the link transitivity rule does not generally apply to probabilistic encryption.

HBC adversary - rule 10 (Known hash)

$$\frac{\mathcal{A}_{\mathcal{E}} \vdash H(i_1);\ \ \mathcal{A}_{\mathcal{EI}} \vdash i_1}{link(H(i_1), i_1)}$$

Given a deterministic cryptographic hash of an information item, if the adversary knows (or can guess) an information item that hashes to the same value, the adversary can detect that i_1 exists externally and can link i_1 to the hash value. As an idealized cryptographic primitive, it is assumed that hash collisions do not occur.

Additional Inference Rules

In addition to the above rules, the model can also include user-specified rules that capture additional information about a protocol or the context in which it is used. For example, additional rules can be used to model mathematical equivalences. In one of the smart meter communication protocols analysed in Section 4.5, this capability is used to model the additive property of energy measurements from smart meters. Although the half-hourly energy consumption measurements are represented as distinct information items, the summation of all half-hourly measurements from a single user is equal to the total consumption measurement for that user. Provided that the half-hourly measurements can all be linked to each other, their summation can be linked to the total consumption measurement through an additional inference rule. In comparison to the observational equivalence approach, the reachability-based approach used in this model makes it significantly easier to capture such behaviour in the form of additional inference rules.

4.3.3 Soundness and Completeness

As a deductive system, it is important to consider the soundness and completeness of this adversary model. As explained above, at any point in the protocol, the set of axioms consists of the adversary's initial knowledge and the knowledge the adversary has gained through receiving messages. These axioms are therefore always true representations of the real system. The inference rules are valid because they are faithful representations of well-known cryptographic concepts. Therefore it is claimed that the deductive system is sound with respect to a real HBC adversary.

However, due to various factors, it cannot be claimed that this deductive system exhibits completeness with respect to a real HBC adversary. Firstly, the axioms may not be a complete representation of the adversary's knowledge. For example, a real adversary may learn additional information from received messages, such as network identifiers or timing information. Secondly, since this analysis takes place in the symbolic paradigm, it may be possible for the adversary to make additional inferences, beyond those allowed by the inference rules, based on properties that are not represented in this model (e.g. the length of certain terms). Ideally, these additional inferences would be captured through the specification of additional inference rules, but it is not always possible to recognize the

adversary capabilities that should be represented as additional rules. Finally, the context in which the protocol is used may provide additional information to the adversary. For example, if the adversary knows that there is only a single participant, all received messages can be linked to this participant. However, since the primary aim of this methodology is to systematically identify vulnerabilities, rather than to provide complete proofs of the privacy properties, the fact that this deductive system does not exhibit completeness does not detract from its usefulness. When considering vulnerability detection rather than proving correctness, Bai [25] argues that it may even be worth sacrificing both soundness and completeness rather than not using formal methods.

4.4 Integration with Existing Methods

The HBC model presented in this chapter has been integrated into the Casper/FDR tool, developed by Lowe [167], to create the Casper-Privacy tool. The following subsections give brief background on the Casper/FDR tool, present the enhanced syntax for modelling privacy properties and describe the implementation of the model. Although the HBC model has been integrated into this particular tool, this does not restrict the applicability of the model itself, which could be integrated into various protocol analysis methods and tools.

4.4.1 Casper/FDR Tool

Roscoe [223] and Lowe [168] developed a method for verifying the security properties of protocols (i.e. secrecy and authentication) using the process algebra of Communicating Sequential Processes (CSP) [126, 127] and its model checker FDR [222]. As a process algebra, CSP provides a formal method for modelling concurrent systems. In CSP, a system is modelled as a set of processes where each process can perform a defined sequence of events. Processes can be placed in parallel with each other and synchronized on specific events such that these events can only occur when all synchronized processes are ready to perform them. The possible sequences of events of a process or a parallel combination of processes are referred to as the event traces. In the existing analysis method [223, 168], the communication protocol is modelled as a CSP process and placed in parallel with an intruder process. The security properties are expressed as reachability assertions for the intruder process. The FDR tool is used to perform trace refinement on the combined system to check if any possible event trace violates the security properties. If any property is violated, FDR outputs the relevant event trace as a counter-example. The Casper/FDR tool developed by Lowe [167, 169] greatly simplifies this analysis by taking abstracted descriptions of protocols (e.g. in so-called *Alice & Bob* notation), compiling them into CSP models and interpreting the FDR output. It should be noted that this method is limited to bounded analysis of protocols in terms of the number of participants and the number of runs of the protocol that can be analysed. Whilst this does not affect scenarios

in which an attack is found, it limits the generality of the claims that can be made when no attack is found. The HBC adversary model presented in the previous section can be used in either bounded or unbounded analysis, but in the case where this model is integrated with the Casper/FDR tool, the overall analysis will be bounded.

4.4.2 Enhanced Casper-Privacy Specifications

In the Casper-Privacy input script, the properties to be analysed are called specifications. In order to analyse undetectability and unlinkability, two new types of specifications have been added to the tool, under the heading `#Privacy`. Each specification is fully described by a single line as shown in Table 4.2.

Table 4.2: New Casper-Privacy Specifications

```
#Privacy

- Undetectable (HBC, {Information items})
Undetectable (A, {X, Y})

- Unlinkable (HBC, {Items}, {Excluded items}, {Extra links})
Unlinkable (A, {X, Y}, {K}, { ({M, N},{Y}) })
```

The undetectability specification begins with the keyword `Undetectable` and takes two parameters, the identity of the HBC adversary and a set of information items. The first parameter specifies which of the participants in the protocol will take the role of the HBC adversary. The second parameter is a list of any information items (e.g. identifiers, data items or keys) that are supposed to be undetectable by the HBC adversary. The undetectability specification will fail if the HBC adversary can detect one or more of the specified information items. For example, the specification in Table 4.2 will fail if participant `A` detects either of the information items `X` or `Y`.

The unlinkability specification begins with the keyword `Unlinkable` and takes four parameters. The first parameter specifies the identity of the HBC adversary. The second parameter lists the information items that this adversary will attempt to link. The third parameter is a list of information items that should be excluded from the linking algorithm. This is used to represent information items that could be shared by multiple participants (e.g. shared keys) and thus should not be used as the basis of links. The fourth parameter allows the specification of additional inference rules that the HBC adversary can use in that specific protocol. These are specified as tuples containing a left set and a right set. If the HBC adversary can link together all the information items in the left set, then the adversary can deduce a link between all items in the union of both sets. In Table 4.2, if the adversary can link information items `M` and `N`, then links between items `M`, `N`, and `Y` can be deduced. The unlinkability specification will fail if the adversary can establish

a definitive link between all the information items in the second parameter taking into account the exclusions and additional links.

In contrast to the existing Casper/FDR secrecy and authentication specifications, the new Casper-Privacy specifications deal with the actual variables in the system rather than the free variables. The specifications can therefore be used in systems where multiple participants take the same role or where multiple repetitions of the protocol take place.

In order to fully represent this model, certain existing Casper/FDR specifications have also been enhanced. Specifically, the encryption specifications have been augmented to model anonymous encryption, probabilistic encryption and any combination thereof as shown in Table 4.3. The original encryption specification is considered to represent non-anonymous non-probabilistic (i.e. deterministic) encryption. Anonymous or probabilistic encryption can be modelled by adding the `-A` or `-P` modifiers respectively, and the combination thereof is represented by the `-AP` modifier.

Table 4.3: Enhanced Casper-Privacy Specifications

```
- Original encryption specification
1.  a → b :  {message}{key}

- Anonymous encryption
1.  a → b :  {message}{key-A}

- Probabilistic encryption
1.  a → b :  {message}{key-P}

- Anonymous probabilistic encryption
1.  a → b :  {message}{key-AP}
```

The Casper/FDR compilation step has been enhanced to automatically compile these new and enhanced specifications into the CSP model, run the analysis of the privacy properties, and integrate the results into the existing Casper/FDR output.

4.4.3 Implementation in CSP

The deductive system and the inference rules presented above have been implemented in CSP and integrated with the existing Casper/FDR tool. The CSP implementation of the deductive system representing the HBC adversary is very similar to the deductive system used to represent the external intruder in the existing analysis method by Roscoe [223] and Lowe [168]. The core aspects of the HBC adversary model are shown diagrammatically in Figure 4.1.

In this context, the term *fact* is used to refer to any information item in the protocol or any statement about one or more information items. For example, the facts `Detect.X` and `Link.(Y,Z)` represent the detection of information item `X` and the linking of items

`Y` and `Z`. A *deduction* (i.e. an inference rule applied to specific facts) is represented as a tuple consisting of a single fact, the subject, and a set of facts from which the subject can be deduced.

In the first phase, the overall set of possible deductions is constructed. For each fact, the set of messages from which the fact can be learned and the sets of other facts from which it can be deduced are calculated. The set of deductive rules in which each fact could be used is also determined.

In the second phase, the `HBC` process is constructed as the parallel combination of processes representing the state of each fact in the system from the perspective of the HBC adversary. Each fact that is not yet known to the HBC adversary is modelled as a process starting in the `UNKNOWN` state. These processes can transition to the `KNOWN` state either through the HBC adversary receiving a message containing the information item, or through the successful application of an inference rule. Once in the `KNOWN` state, the process is willing to participate in any relevant deduction event. These processes are synchronized on all deduction events so that deductions will only succeed if all the required facts are known. Processes in the `KNOWN` state representing a `Detect.X` or `Link.(X,Y)` fact can generate an `HBC` event if the information items they represent have been specified as undetectable or unlinkable.

The `HBC` process is placed in parallel with the original Casper/FDR `SYSTEM` process that represents the behaviour of honest participants in the protocol. These two processes are synchronized on events performed by the HBC adversary. In the final phase, trace refinement is used to determine if any sequence of events could violate any of the undetectability or unlinkability specifications. If such a trace is found, the Casper-Privacy tool outputs the sequence of events as a counter-example showing how the respective undetectability or unlinkability specification was violated.

4.4.4 Alternative CSP Implementations

The following approaches have been tested and found to be unsuitable or less efficient than the implementation described in the previous section.

Receive then Check

When the HBC adversary receives a message, the links contained in the message are parsed and added to the adversary's set of links. After every new message, the set of links is analysed to determine if it contains a path between the specified information items. If a link can be established, an `HBC` event is generated. This approach generates the correct event traces but results in a state explosion in the refinement checking phase. In the full transition system, a separate event is created for each possible set based on a power set construction (i.e. for a set of n elements, the resulting power set contains 2^n sets).

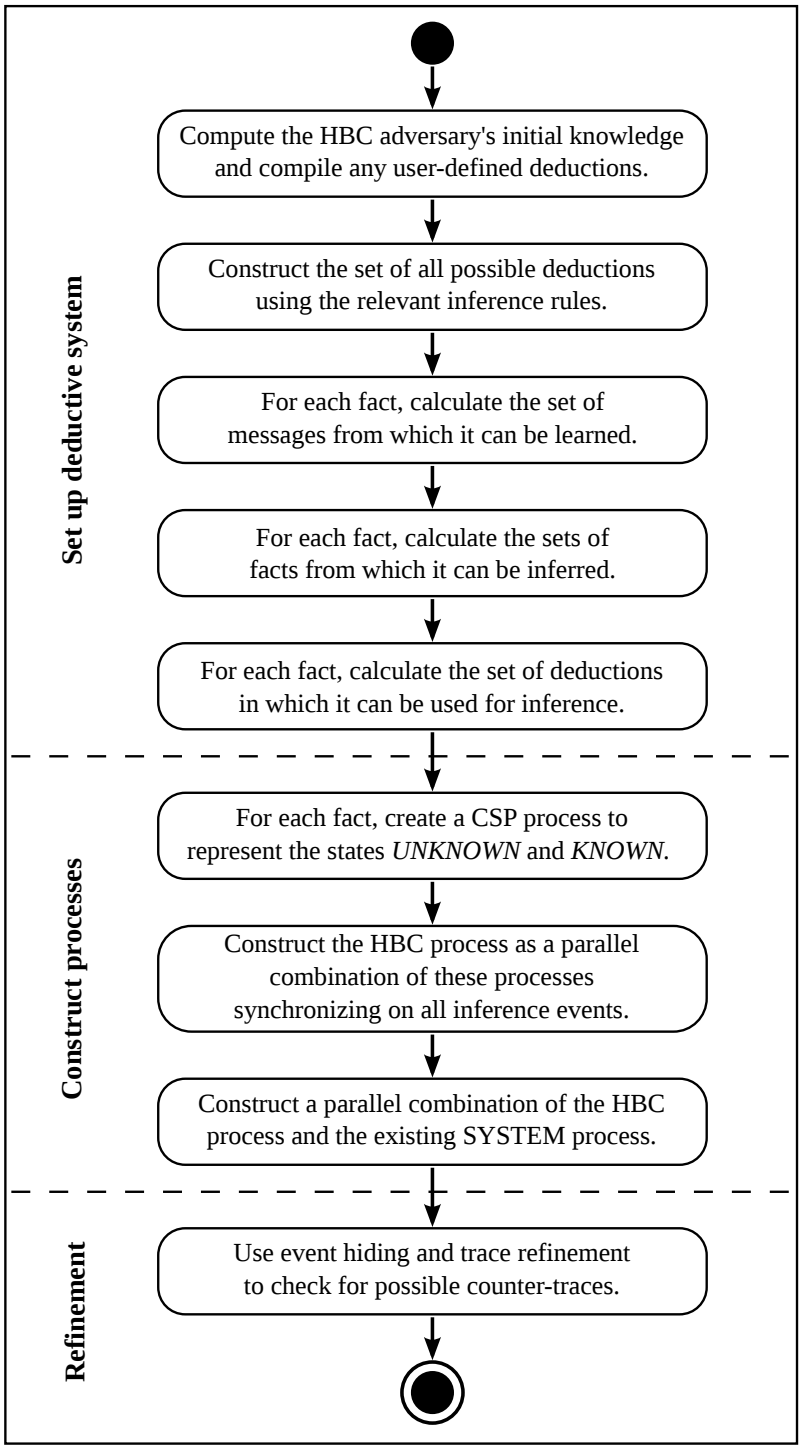

Figure 4.1: CSP implementation of the deductive system

Accumulate then Check

This is similar to the above approach but avoids the need to check for links every time a message is received. The messages received by the adversary are added to a set that is checked for links after the protocol completes. Again, this approach produces the correct event traces but leads to state explosion given the large number of possible permutations.

Check-lists using Arguments

The system first calculates the sets of messages that, if received by the HBC adversary, would cause a specification to fail (i.e. a check-list). The system then attempts to detect any event trace that contains all the events on a given check-list. The system creates a separate process for each check-list and whenever a message is received by the HBC adversary, it is checked-off the list. Once all events on a list have taken place, the relevant `HBC` event is performed causing the trace refinement to fail. In this approach, the construction of the check-lists is relatively efficient. However, the method of detecting traces corresponding to these lists again causes a state explosion because of the high number of possible permutations that arise from this set-based construction.

Check-lists using Parallel Composition

This approach is an enhancement of the previous check-list approach. As in the previous approach, the system first calculates the sets of messages that would cause the specifications to fail. The system then detects the corresponding traces using a parallel composition of processes. Instead of creating a process for each list, the system creates a separate process for each information item. These processes are composed in parallel according to the check-lists such that when all events on a list have occurred, the combined process generates the relevant `HBC` event. This avoids the state explosion by eliminating the use of sets. Instead, it relies on a very large number of processes and achieves the desired behaviour through parallel composition. Although this approach is feasible to implement, the main implementation described in the previous section is significantly more flexible and scalable.

4.5 Evaluation

In order to evaluate the functionality of this HBC adversary model and its implementation, the Casper-Privacy tool has been used to re-analyse three different protocols that have already been analysed in the literature. The tool correctly recreated the results from the published analyses and, in some cases, identified new attacks. In addition, four smart meter communication protocols from the literature have been analysed to evaluate the effectiveness of the HBC model in this specific context. For each protocol, the security properties (secrecy and authentication) have been analysed using the existing Dolev-Yao

```
#Free variables
idp1, idp2 : IDPAgent     -- Identity providers
ls : LSAgent              -- Linking service
sp : SPAgent              -- Service provider
idsess : SessionID        -- session identifier
n1, n2 : Nonce            -- nonces
pid1, pid2 : PID          -- permanent user identifiers (ls <-> idp)
d1, d2 : Attributes       -- user attributes from idp1 and idp2
pubidp2 : IDPPublicKey    -- IDP2's public key
prividp2 : IDPPrivateKey -- IDP2's private key
publs : LSPublicKey       -- LS's public key
privls : LSPrivateKey     -- LS's private key
InverseKeys = (publs,privls), (pubidp2,prividp2)

#Protocol description
-- All messages are sent over confidential authenticated channels
1. idp1 -> sp : idsess, d1, ls, {pid1,n1}{publs} % x1
2. sp -> ls : idsess, d1, x1 % {pid1,n1}{publs}
3. ls -> sp : idp2, {pid2,n2}{pubidp2} % x2
4. sp -> idp2 : idsess, d1, x2 % {pid2,n2}{pubidp2}
5. idp2 -> sp : idsess, d2
```

Listing 4.1: Part of the model for the TAS^3 attribute aggregation protocol [57]

adversary whilst the privacy properties (undetectability and unlinkability) were concurrently analysed using the new HBC adversary model. The following subsections present selected parts of the Casper-Privacy models and summarize the analysis results. The full Casper-Privacy input scripts are included in Appendix A.

4.5.1 TAS^3 Attribute Aggregation Protocol

The purpose of the TAS^3 protocol by Chadwick [57] is to allow a user U to supply a service provider SP with identity-related attributes from multiple identity providers $IdPs$ in a single session without the user having to authenticate to each provider during the session. To achieve this, the protocol introduces a linking service LS. Once U has been authenticated by the first identity provider IdP_1, the SP receives the address of the LS and a token for the LS from IdP_1. The SP contacts the LS which responds with the address of IdP_2 and another authorization token. The SP then contacts IdP_2 to obtain further attributes about U. Veeningen et al. [273] have analysed this section of the protocol in terms of undetectability and unlinkability. Listing 4.1 shows part of the Casper-Privacy model for this protocol[3] and the full model is shown in Listing A.1 in Appendix A. This model only includes the interaction between two identity providers, the service provider and the linking service, since this is the section in which the previously identified privacy flaw arises.

Security Properties: In this protocol, all communication takes place over TLS and

[3]In these compact representations, the comments (denoted by --) are placed on the same lines as the input, but when compiling these models, each comment must be moved its own line.

strong identity guarantees are provided by each of the communicating nodes. The security properties analysed were the secrecy of data items and identifiers and the authentication of the communicating parties. This analysis did not reveal any compromises of these properties in the presence of an external DY adversary.

Privacy Properties: Veeningen et al. [273] defined three undetectability properties and one unlinkability property for this protocol. Their analysis showed that two undetectability properties (P2 and P3 in [273]) do not hold, allowing both the LS and IdP_2 to detect and observe attributes of U that should only be available to the SP. The analysis with the Casper-Privacy tool produced the same results for all four properties and, in addition, showed that IdP_2 can link the detected attributes to a specific user, thus constituting a serious privacy flaw. This attack was found by analysing the unlinkability of attribute D_1 and identifier PID_2 from the perspective of IdP_2 (P5 in Listing A.1 in Appendix A). The root cause of this attack is as follows: in message 4, SP includes attribute D_1 in the message sent to $IdP2_2$. However, this message also contains PID_2 in encrypted form, and thus, if IdP_2 is an HBC adversary, it can use the *asymmetric decryptable* and *compositional link* rules to link D_1 to PID_2. This means that IdP_2 learns new information about the user through this protocol. This particular unlinkability property was not analysed by Veeningen et al. [273], however, it is reasonable to expect that, had they analysed it, they would have arrived at the same conclusion.

4.5.2 Unlinkability of RFID e-Passports

Arapinis et al. [17] have used the applied pi calculus to analyse one of the RFID communication protocols used in electronic passports (e-passports). The Basic Access Control (BAC) protocol is a four-message protocol designed to establish a shared session key between the RFID reader and the tag in the passport. Before the protocol begins, the reader scans the optical data on the passport to obtain the tag's long-term encryption key k_e and message authentication key k_m. Using k_e and k_m, the parties exchange encrypted messages and message authentication codes (MACs). Arapinis et al. [17] have described a linkability flaw in the BAC protocol that could allow an adversary to identify and track the passport over RFID without the consent of the user. It must be noted that exploitation of this flaw requires an active adversary. Therefore, for this analysis the capabilities of the HBC adversary were extended with user-specified rules to recreate the analysis by Arapinis et al. [17]. Since the Casper-Privacy tool cannot model branching in protocols, two separate models were created to capture the normal protocol as well as the case in which the adversary replays an old message to the tag as shown in Listing 4.2. The full models are shown in Listing A.2 and Listing A.3 in Appendix A.

Security Properties: The primary security requirements for this protocol, with respect to an external DY adversary, are the secrecy of the exchanged keys and the authentication between the tag and the reader. As in the published analysis [17], it is assumed that k_e and k_m are shared by the tag and the reader over a secure channel. Analysis with the

```
#Free variables
t, r, hbc : Agent          -- tag, reader and adversary
nt : NonceTag              -- nonce generated by tag
nr : NonceReader           -- nonce generated by reader
kt : KeyMaterialTag        -- key material generated by tag
kr : KeyMaterialReader     -- key material generated by reader
e6A80 : ErrorCode          -- error code used in the system
ke : KeyEncrypt            -- encryption key derived from passport
km : KeyMAC                -- MAC key derived from passport data
h : HashFunction
InverseKeys = (ke,ke), (km,km)

#Protocol description      -- normal protocol
0.   -> t : r
1. t -> r : nt
2. r -> t : {nr,nt,kr}{ke}, h({nr,nt,kr}{ke})
3. t -> r : {nt,nr,kt}{ke}, h({nt,nr,kt}{ke})

#Protocol description      -- invalid nonce sent to tag
0.   -> t : hbc
1. t -> hbc : nt
-- HBC replays message previously overheard
2. hbc -> t : {ni,nt,kr}{ke}, h({ni,nt,kr}{ke})
3. t -> hbc : nt, e6A80
```

Listing 4.2: Part of the model for the ePassport Basic Access Control protocol [17]

Casper-Privacy tool confirms that an external DY adversary is unable to compromise the security properties of this protocol without first learning these long-term secret keys.

Privacy Properties: As described by Arapinis et al. [17], the adversary records the second RFID message (m_2) from a legitimate run of the protocol for a target passport. This message from the reader to the tag contains the original nonce generated by the tag and a MAC using k_m. The adversary then runs the protocol with any tags in range and replays m_2. The tags first check the MAC and then the nonce in m_2 and will send an error message if either check fails. Critically, in the French implementation of this protocol, the tag produces different error messages for each check. As confirmed by the Casper-Privacy tool, the nonce check will always fail, but if the MAC check passes the adversary learns that this is the target passport.

4.5.3 Protecting location privacy using k-anonymity

Gedik and Liu [111] have proposed a protocol to enhance privacy in a location-based service (LBS). In a LBS, a user's current location is sent to a service provider SP in order to receive some information or service relevant to that particular location. However, users do not always trust the SP and thus the SP is represented as an HBC adversary. The aim of this protocol is to prevent the SP from linking the submitted location information to a specific user. As shown in Listing 4.3, in this protocol the users send their requests containing their real identities and precise locations to a trusted anonymity server AS.

```
#Free variables
m : Agent               -- mobile device
sp : ServiceProvider    -- service provider
as : AnonymityServer    -- anonymity server
id : UserID             -- unique device ID (e.g. IMSI/IMEI)
rn : RequestNumber      -- unique request number
rd : RequestData        -- request data
lbd : LBData            -- request response
h : HashFunction

#Protocol description
-- All messages are sent over confidential authenticated channels
1. m -> as : m, id, rn, sp, rd
2. as -> sp : h(id,rn) % t, rd
3. sp -> as : t % h(id,rn), lbd
4. as -> m : id, rn, lbd
```

Listing 4.3: Part of the model for the location-based service protocol [111]

This server implements a spatial cloaking algorithm before forwarding parts of the requests to the SP. Even though it is assumed that no user identities are required by the SP, there is still a risk that the SP could link multiple requests together to create location patterns of specific users. Auxiliary information could then be used to link these location patterns to named users. The full model is shown in Listing A.4 in Appendix A.

Security Properties: Gedik and Liu [111] explained that in this protocol, the users are assumed to have secure connections to the trusted anonymity server (e.g. over TLS). The users identify themselves to this server using their real identities (u_{id}). The communication between the anonymity server and the SP also takes place over TLS because the privacy properties are not affected by the untrusted SP learning the identity of the anonymity server. The Casper-Privacy analysis does not reveal any compromises of the security properties with respect to an external DY adversary.

Privacy Properties: The two main privacy requirements in this protocol are that the untrusted SP should not be able to detect identifying information for individual users and that the SP should not be able to link multiple requests together. The Casper-Privacy analysis confirms that this protocol achieves these objectives with respect to a semi-honest SP. In particular, the step taken by the trusted anonymity server of replacing the user identity (u_{id}) and the request number (r_{no}) with a random string before sending the request to the SP is critical to the privacy properties. However, further analysis shows that the anonymity server itself could compromise both these privacy properties. The root cause of this attack is that in message 1, the mobile device sends its unique device ID (id) directly to the anonymity server (as). An untrusted anonymity server can trivially detect this information. Furthermore, since this is a long-term identifier, the anonymity server can link together multiple requests containing the same identifier in order to track the user. It is therefore critical that this service should be provided by a trustworthy entity (e.g. as described in Chapter 8).

```
#Free variables
sm : Agent          -- smart meter
ut : Utility        -- utility
agg : Aggregator    -- aggregator
hfid : HFID         -- high frequency identifier
lfid : LFID         -- low frequency identifier
ma : HFDataA        -- measurement from smart meter
mb : HFDataB        -- measurement from smart meter
t : LFData          -- total energy consumption from smart meter

#Protocol description
-- All messages are sent over confidential authenticated channels
1. sm -> agg : hfid, ma
2. agg -> ut : hfid, ma
3. sm -> agg : hfid, mb
4. agg -> ut : hfid, mb
5. sm -> ut  : lfid, t
```

Listing 4.4: Part of the model for the pseudonymous smart metering protocol [89]

4.5.4 Smart Meter Anonymization using Pseudonyms

The smart meter communication protocol by Efthymiou and Kalogridis [89] uses two unlinkable identifiers for reporting energy measurements from a smart meter to an energy utility. The high-frequency identifier ($HFID$) is a pseudonym for reporting frequent measurements and the low-frequency identifier ($LFID$), which contains the user's personal information, is used for infrequent communication such as reporting the total consumption over a billing period. The link between an $HFID$ and the corresponding $LFID$ should only be known by a trusted third party. However, in their analysis of this protocol, Jawurek et al. [139] have described multiple ways in which an $HFID$ could be linked to a real user based on correlation with secondary data sources. Listing 4.5 shows part of the Casper-Privacy model of this protocol in which a single consumer submits two pseudonymous consumption measurements using the consumer's $HFID$ followed by the total consumption measurement using the consumer's $LFID$. In this protocol, the aggregator simply forwards all received messages to the utility. The full model is shown in Listing A.5 in Appendix A.

Security Properties: The authors propose the use of asymmetric cryptography and digital certificates from a mutually trusted certificate authority (CA). In particular, each smart meter has a separate certificate and key pair for its $HFID$ and $LFID$. The Casper-Privacy analysis does not reveal any compromises in terms of the secrecy of the messages or the authentication of the communicating entities in the presence of an external DY adversary.

Privacy Properties: The Casper-Privacy tool was used to analyse the unlinkability between an $HFID$ and $LFID$ from the perspective of the energy utility. The results showed that this fundamental claim does not hold because the utility can use the $HFID$ as a pseudonym to link high frequency measurements together and obtain the total con-

```
#Free variables
ut : Utility  -- utility
sm : Agent    -- smart meter
ma : HFDataA  -- sequential measurements from smart meter
mb : HFDataB  -- (assumed to include timestamps)
t: LFData     -- total energy consumption from smart meter

GEpublic   : GridEncryptionPublicKey    -- grid operator keys
GEprivate  : GridEncryptionPrivateKey

-- Spublic assumed to be signed by utility using GSprivate
Spublic  : SmartMeterPublicKey          -- pseudonymous keys
Sprivate : SmartMeterPrivateKey
Rpublic  : RealPublicKey                -- smart meter keys
Rprivate : RealPrivateKey

h : HashFunction
InverseKeys = (GEpublic,GEprivate), (Spublic,Sprivate),
  (Rpublic,Rprivate)

#Protocol description
1. sm -> ut : { Spublic, ut, ma, {h(ma)}{Sprivate} }{GEpublic}
2. sm -> ut : { Spublic, ut, mb, {h(mb)}{Sprivate} }{GEpublic}
3. sm -> ut : { Rpublic, ut, t,  {h(t)}{Rprivate} }{GEpublic}
```

Listing 4.5: Part of the model for the pseudonymous smart metering protocol [101]

sumption for an $HFID$ as the summation of these values. Since these measurements are sufficiently detailed, the utility can uniquely match this total to the values reported using the $LFID$ and thus de-anonymize the consumer.

4.5.5 Pseudonymous Smart Metering without a Trusted Third Party

Finster and Baumgart [101] have proposed a similar protocol to anonymize energy measurements. In their protocol, each smart meter S generates an asymmetric key pair $\{S_{public}, S_{private}\}$. Initially S authenticates itself to the energy utility and sends a cryptographically blinded version of S_{public} to be signed. After unblinding the result, S uses this as a pseudonym to report high-frequency measurements. Similarly to the previous protocol, low-frequency measurements are still reported using the consumer's real identity. Again the fundamental requirement is that the adversary should not be able to link a specific S_{public} to a particular consumer. A proposed option in this protocol is that these pseudonyms could be re-issued on a more frequent basis (e.g. daily). Listing 4.5 shows part of the Casper-Privacy model of this protocol in which a single consumer submits two pseudonymous consumption measurements using S followed by the total consumption measurement using the consumer's real key. The full model is shown in Listing A.6 in Appendix A.

Security Properties: Similarly to the protocol above, the authors use asymmetric cryptography and digital certificates to encrypt and sign the measurement data. The high-

frequency measurements are signed by the smart meter's private key $S_{private}$ and can be verified using S_{public}, which is itself signed by the energy utility. The Casper-Privacy analysis did not reveal any compromises in terms of the secrecy of the messages or the authentication of the participants in the presence of an external DY adversary.

Privacy Properties: The Casper-Privacy tool was used to analyse the unlinkability between S_{public} and the consumer's identity ID_U from the perspective of an HBC energy utility. The tool presented a counter-example showing an attack against this unlinkability property. As in the previous protocol, the root cause of this attack is that the energy utility can establish links between the high-frequency consumption values based on the pseudonym S_{public}. If all the measurements in a billing period can be linked together, the total can be calculated, matched with one of the submitted totals (t), and thus linked to ID_U. However, further analysis shows that if the pseudonyms are changed during the billing period, it would not be possible to link all the required high-frequency measurements and thus the desired unlinkability property would be maintained.

4.5.6 Smart Meter Anonymization through Group Identifiers

Borges et al. [43] have described a generalized representation of a privacy-enhancing protocol for smart meter communication, based on anonymity networks. Similarly to the protocols above, the protocol distinguishes between high-frequency anonymized information and low-frequency identifiable information. A unique customer identifier IdC is used for low-frequency messages whilst an anonymous group identifier IdG is used for high-frequency messages. To avoid the pseudonym attacks described above, one IdG is used to represent a group of users. The adversary should not be able to link any high-frequency measurements to a specific IdC. Listing 4.6 shows part of the Casper-Privacy model of this protocol in which a single consumer submits two consumption measurements using IdG followed by the total consumption measurement using IdC. The full model is shown in Listing A.7 in Appendix A.

Security Properties: The combined security and privacy analysis with the Casper-Privacy tool identified an attack against the security properties of this protocol. It is realistic to assume that a Dolev-Yao adversary controls at least one of the smart meters in the system (e.g. using available open-source tools [236]). Since IdG is shared by all smart meters, this adversary can generate and send multiple falsified high-frequency measurements in each reporting period, thus invalidating the legitimate reporting from the whole group.

Privacy Properties: The Casper-Privacy analysis of the privacy properties did not reveal any further attacks. It confirmed that the HBC adversary cannot link any measurements reported using IdG to any unique IdC without additional auxiliary information. However, this strong anonymity guarantee exacerbates the consequences of the security flaw in this protocol, since it is not possible for the utility to identify which meters in the group have been compromised.

```
#Free variables
sm  : Agent           -- smart meter
ut  : Utility         -- utility
idg : IdGroup         -- group identifier
idc : IdPersonal      -- personal identifier
ma  : MeasurementA    -- sequential measurements from smart meter
mb  : MeasurementB    -- (assumed to include timestamps)
t   : Total           -- total energy consumption from smart meter
pubUt  : PublicKey    -- utility's public key
privUt : PrivateKey   -- utility's private key
InverseKeys = (pubUt,privUt)

#Protocol description
1.  sm -> ut : {idg, ma}{pubUt}
2.  sm -> ut : {idg, mb}{pubUt}
3.  sm -> ut : {idc, t}{pubUt}
```

Listing 4.6: Part of the model for the smart meter protocol using group identifiers [43]

4.5.7 OpenADR Standard

In the smart grid, the term demand response (DR) describes a set of actions to dynamically reduce energy demand at specific times and locations. Incentive-based DR schemes offer consumers some incentive to voluntarily participate in demand response events. One particular type of scheme, demand bidding, is based on a bidding process in which consumers place bids indicating the amount by which they are currently willing to reduce their consumption and, in some cases, the desired level of incentive. The Demand Side Manager (DSM) selects which bids to accept and communicates these acceptance decisions to the relevant bidders. Unlike smart metering, the demand bidding process requires bi-directional communication between the DSM and each individual consumer. OpenADR is a communication data model that can be used for incentive-based DR [213]. This standard introduces the concept of the Demand Response Automation Server (DRAS), an intermediary node that receives bids from consumers and forwards them to the energy supplier [213]. Listing 4.7 shows part of the model in which a consumer sends a bid to the DSM via the DRAS. The full model is shown in Listing A.8 in Appendix A.

Security Properties: The OpenADR protocol uses strong identity guarantees from all of the communicating nodes and uses TLS connections for all communication between the nodes [148]. The Casper-Privacy analysis did not reveal any potential attacks in terms of the secrecy of the messages or the authentication of the participants in the presence of an external DY adversary.

Privacy Properties: Previous work has described potential privacy concerns that arise in OpenADR based on the bi-directional flow of information [208, 148]. In this system, the energy supplier is modelled as an HBC adversary $\mathcal{A}$. Given that the DRAS forwards the messages directly to $\mathcal{A}$, the adversary can detect the bid amounts and can link these to individual users. Furthermore, given sufficient auxiliary information, such as a database of appliance energy signatures, $\mathcal{A}$ can match these bid amount to specific signatures to

```
#Free variables

sm : Agent        -- smart meter
ut : Utility      -- utility
dras : DRAS       -- DRAS
id : DREventID    -- identifier for a DR event
bid : BidAmount   -- amount bid by a specific smart meter

#Protocol description
-- All messages are sent over confidential authenticated channels
1. ut -> dras : ut, dras, id
2. dras -> sm : dras, sm, id
3. sm -> dras : sm, dras, id, bid
4. dras -> ut : ut, sm, dras, id, bid
```

Listing 4.7: Part of the model for the OpenADR smart meter communication protocol [213]

learn private information such as which appliances are being used. Further analysis shows that this flaw also exists if the DRAS is an HBC adversary [148].

4.6 Summary

In order to achieve communication privacy, it is critical that the communication protocols exhibit specific privacy properties. Formal methods can be used to model and reason about privacy properties such as undetectability and unlinkability, and as the protocols become more complicated, automated analysis of these properties is required. Various approaches have been used to model and analyse these properties, including indistinguishability in the computational paradigm and observational equivalence in the symbolic paradigm. This chapter has presented an approach for modelling and analysing these properties as reachability assertions in the presence of an honest-but-curious (HBC) adversary. This approach is similar to that used in the analysis of security properties, such as secrecy and authentication. The HBC adversary is modelled as a deductive system consisting of a set of inference rules derived from the definitions of undetectability and unlinkability. In comparison to other symbolic approaches, a significant advantage of this reachability approach is that it can be directly integrated with the analysis of security properties. Furthermore, these two types of analysis can be performed concurrently on a single formal model of the protocol, incorporating multiple concurrent adversary models, in order to analyse the interplay between security and privacy properties.

The HBC adversary model has been implemented in the process algebra of CSP and integrated into an established analysis method. An enhanced version of the Casper/FDR tool, the Casper-Privacy tool, has been developed in order to model and analyse both security and privacy properties of a protocol. This HBC adversary model and the Casper-Privacy tool have been evaluated by re-analysing three communication protocols that have been analysed in recent literature. In all cases, the Casper-Privacy analysis gave the same results as the published analyses and, in some cases, new attacks have been identified.

To demonstrate the effectiveness of this tool in the smart grid context, a further four smart meter communication protocols have been analysed and various attacks against the security and privacy properties have been identified. Although the Casper-Privacy analysis tool does not offer some of the advanced features of more recent protocol analysis tools (e.g. unbounded verification), these example analyses show that the tool can correctly identify these types of security and privacy flaws in communication protocols, such as those used in the smart grid. This tool can therefore be used to model and systematically analyse the enhanced smart grid communication architecture presented in the next chapter.

Chapter 5

Enhanced Smart Grid Architecture

This chapter is based on the following publications:

- A. J. Paverd, A. P. Martin, & I. Brown, "Security and Privacy in Smart Grid Demand Response Systems". In: *Second Open EIT ICT Labs Workshop on Smart Grid Security (SmartGridSec'14)*, 2014 [208].
- A. J. Paverd, A. P. Martin, & I. Brown, "Privacy-Enhanced Bi-Directional Communication in the Smart Grid using Trusted Computing". In: *Proceedings of the Fifth IEEE International Conference on Smart Grid Communications (SmartGridComm'14)*, 2014 [207].

5.1 Baseline System Model and Requirements 88
5.2 Privacy-Enhancing Communication Architecture 93
5.3 Implementation Considerations 102
5.4 Evaluation of Architecture 108
5.5 Summary 114

This chapter presents and evaluates an enhanced smart meter communication architecture based on the Trustworthy Remote Entity (TRE). The chapter begins by defining the specific requirements of the solution with reference to a baseline system model. An overview of the enhanced architecture is then presented, showing how the TRE can be integrated into the current smart grid architecture. This architecture covers all three information flows defined in Chapter 3: network monitoring, billing, and Demand Response (DR). For each information flow, a privacy-enhancing communication protocol using the TRE is presented. Each of these protocols represents a novel contribution and, in particular, the bi-directional communication protocol used in the DR information flow is one of the first solutions to be proposed for this particular problem. To evaluate the architecture, the security and privacy properties of each of the new communication protocols are analysed using the Casper-Privacy tool presented in the previous chapter. The efficiency and practicality of the new protocols are evaluated through comparisons with other proposals from recent literature in terms of the number of messages exchanged, the number of cryptographic operations performed, and the types of cryptographic operations required.

5.1 Baseline System Model and Requirements

As explained in Chapter 3, various countries and regions are currently designing and deploying smart grids. The aim of this research is to enhance communication privacy within these current smart grid designs. This significantly improves the deployability of this solution when compared with approaches that require a fundamental redesign of the smart grid. To achieve this, the architecture presented in this chapter is based on a baseline system model, as presented in this section, that captures the main principles and requirements of current smart grid designs. This baseline model is not a concrete communication architecture, but is instead a representation of the fundamental exchanges of information (i.e. communication) that take place between consumers and other participants in the smart grid. For example, as described in Chapter 3, in the UK smart grid architecture, the Data Communications Company (DCC) periodically queries each smart meter to retrieve consumption measurements, which can then be retrieved from the DCC by authorized entities, such as energy suppliers. However, in a different architecture, smart meters might send these measurements directly to the energy suppliers. In both cases, the fundamental exchange of information is that the measurements are communicated from the smart meters to the energy suppliers. The baseline system model captures these types of fundamental communication requirements that describe *what* communication takes place, without specifying *how* it takes place. This has the natural advantage that the baseline model and requirements are independent of any concrete communication architecture. Furthermore, the baseline model includes functionality such as residential demand bidding, which will likely be incorporated into future iterations of the smart grid [106, 8]. The functional, security and privacy requirements of this solution are defined with reference to this baseline system model.

5.1.1 Baseline System Model

In the set of all consumers $\mathcal{C}$, each consumer $c \in \mathcal{C}$ has a feature-rich smart meter or home energy management system capable of bi-directional communication. For each time period t, the consumer c produces a measurement $m_c(t)$ representing c's total energy consumption during that time period. All time periods have the same constant duration τ, which is a parameter of the implementation. In current implementations, such as the UK architecture, this duration ranges between $\tau = 30$ minutes and $\tau = 24$ hours [92].

In the monitoring information flow, consumption measurements from smart meters are communicated to the respective Distribution Network Operator (DNO). In order to monitor consumption in each physical segment of the distribution network, the DNO requires the sum $S_G(t)$ of the individual consumption measurements for time period t from a group of consumers G who are in a specific geographical area [100]:

$$S_G(t) = \sum_{c \in G} m_c(t)$$

For dynamic pricing, the energy supplier periodically broadcasts the price per unit of energy $p(t)$ (e.g. in £/kWh for electricity) that will apply at time period t. This is not included in the billing information flow because it is a broadcast message sent to multiple consumers. In the billing information flow, consumption measurements from smart meters are communicated to the respective energy suppliers. For a billing period that runs from *t-start* to *t-end*, the supplier requires the bill $L_c(\textit{t-end})$ for each consumer c, which is calculated by multiplying each consumption measurement $m_c(t)$ by the prevailing price per unit for that time period $p(t)$ and taking the summation over the billing period:

$$L_c(\textit{t-end}) = \sum_{t=\textit{t-start}}^{\textit{t-end}} m_c(t) \times p(t)$$

In the DR information flow, when a reduction in demand is required in a specific area, the Demand Side manager (DSM) creates a demand bidding event and invites bids from consumers in that area [106, 8]. For time period t, each consumer c may generate a bid $(\textit{bid-}q_c(t), \textit{bid-}p_c(t))$, which represents an offer to reduce demand in time period t by the bid quantity $\textit{bid-}q_c(t)$ (or, in some architectures, to sell back this quantity to the grid) in exchange for an incentive payment at the bid price per unit $\textit{bid-}p_c(t)$. Depending on the implementation, the bid quantity can be expressed as either energy or power. These bids are communicated from the consumers to the DSM. The DSM selects which bids to accept and creates a decision $d_c(t)$ (`accept` or `reject`) for each individual bidder. Unlike the other information flows, the DR information flow therefore requires bi-directional communication between consumers and the DSM. Depending on the specific implementation, the DNO, energy supplier and DSM might not all be independent participants (e.g. a single participant might fulfil two or more roles). Where the distinction between roles is not important, the term *service provider* is used to refer to any combination of these roles.

5.1.2 Functional Requirements

The following functional requirements are derived from the baseline system model:

FR-1: In the monitoring information flow, the DNO must receive or be able to compute the total consumption $S_G(t)$ for a group of consumers G in each time period t. To be of use, this information must be available to the DNO within the following time period and must be within an acceptable margin of error.

FR-2: In the billing information flow, the energy supplier must receive or be able to compute the total bill $L_c(t\text{-}end)$ for each consumer c over the billing period.

FR-3: In the DR information flow for a demand bidding protocol, the DSM must be able to use any algorithm to accept or reject bids. The DSM must therefore receive the bids within the same time period as the invitation to bid.

FR-4: For demand bidding, each bidder must receive an acceptance or rejection decision within the same time period as the bid was placed.

FR-5: For demand bidding, the incentives for accepted bids must be credited to the respective bidders within the same billing period.

These functional requirements define the scope of the solution presented in this chapter. This scope is broadly similar to that of current smart meter deployment plans (e.g. the UK smart grid architecture [270]) but also includes residential demand bidding functionality which is likely to be incorporated into the next generation of smart grid systems [106, 8]. These requirements are therefore applicable to current as well as future smart grid architectures.

5.1.3 Security Requirements

Various threats to the security of the smart meter communication architecture have been described in Chapter 3. These could be carried out by different classes of attackers, ranging from an adversarial consumer who has hacked into a single smart meter, through to a well-resourced external entity who controls a large percentage of the communication network. These threats must therefore be considered with respect to the strongest type of adversary which, in the context of a communication system such as this, can be represented using the Dolev-Yao (DY) adversary model [85]. The DY model gives the adversary the capability to eavesdrop on all communications in the system and to modify and falsify any messages. The DY adversary is limited only by the use of cryptography, since it is assumed that this adversary cannot break correctly implemented cryptographic primitives in a reasonable time. However, the DY adversary might have access to specific cryptographic keys (e.g. the keys available in a compromised smart meter). If the solution is secure against this type of adversary, it will be secure against adversaries with lesser capabilities. The solution must therefore fulfil the following security requirements in the presence of a DY adversary:

SR-1: The integrity and authenticity of all information communicated between consumers and the DNO, energy supplier and DSM must be protected.

SR-2: All aggregated quantities (i.e. $S_G(t)$ and $L_c(t\text{-}end)$) must be correctly calculated using the correct sets of consumers and time periods.

SR-3: All DR bids processed by the DSM must correspond to bids from legitimate consumers. Both the authenticity and integrity of the bids must be ensured.

SR-4: Incentives for successful bids must only be credited if the consumer has actually reduced consumption by the specified quantity.

It should be noted that these security requirements apply only to the communication system and not to the end points (e.g. the smart meters or service providers). In real-world implementations, other security requirements would apply to these end points. For example, smart meters should be sufficiently tamper-resistant to prevent theft of energy by adversarial consumers and energy suppliers should be secure against data breaches. However, the security of the end points is an orthogonal concern to the security of the communication architecture and is thus beyond the scope of this solution. This solution therefore assumes that the relevant end point security requirements have been met.

5.1.4 Privacy Requirements

As described in Chapter 3, privacy concerns arise from the possibility that certain participants might be honest-but-curious (HBC) adversaries who will follow the defined protocol but attempt to learn private information about the other participants from the received messages. In the worst-case scenario, the DNO, energy supplier and DSM might all be HBC adversaries. Furthermore, these three potential adversaries might collude with one another, especially if multiple roles are performed by the same participant. As explained in Chapter 3, it would be unrealistic to represent the DNO, energy supplier or DSM using the DY adversary model because, as highly-visible public participants, their behaviour can be scrutinized by all participants and controlled through mechanisms such as regulation and auditing. However, HBC behaviour by these entities would be significantly more difficult to detect and prevent since it does not involve any deviation from the defined protocol. Therefore, the solution must assume that these participants are HBC adversaries and provide adequate communication privacy for consumers, with respect to both the DY and HBC adversaries, by fulfilling the following privacy requirements:

PR-1: In the monitoring and billing information flows, the adversaries (both DY and HBC) must not be able to link any individual consumption measurements to specific consumers.

PR-2: In the monitoring and billing information flows, the adversaries must not be able to link together multiple consumption measurements from the same consumer.

This prevents the adversaries from building up a pattern of behaviour that could identify the consumer.

PR-3: In the DR information flow, the adversaries must not be able to detect whether or not a specific consumer has placed a bid. This prevents the adversaries from inferring private information about consumers based on bidding behaviour.

PR-4: In the DR information flow, the adversaries must not be able to link any bids to specific consumers.

PR-5: In the DR information flow, the adversaries must not be able to link together multiple bids from the same consumer. This prevents the DSM from building up a pattern of behaviour that could be linked to a specific consumer.

Even if the service providers are not actually adversarial, these privacy requirements are also actually beneficial to the service providers because they help to mitigate the consequences of data breaches. For example, if an energy supplier were fully trusted by consumers, the consumers might consent to share their frequent consumption measurements directly with this supplier. If this trusted supplier suffers a data breach as the result of an attack by an external adversary, consumers' private information might be leaked. In this way, an external adversary can retroactively become an HBC adversary with access to consumers' private information. For example, if the trusted supplier had stored individual consumption measurements from the previous billing period (e.g. for billing purposes) and these were leaked through the data breach, the result would be the same as if the consumers had been communicating directly with an internal HBC adversary for the previous billing period. Since this attack is retroactive, consumers cannot withhold their private information or defend against this adversary in any way. In addition to compromising consumers' privacy, this would undermine consumers' trust in the supplier and possibly lead to legal action against the supplier by the national data protection authority (e.g. the Information Commissioner's Office in the UK).

However, if these privacy requirements are met with respect to the supplier, then they are also met with respect to any adversaries who obtain this information through data breaches at the supplier (i.e. retroactive HBC adversaries). Firstly, this reduces the risk of attack by making suppliers less valuable targets and secondly, it reduces the consequences of the attack in terms of loss of personal information.

The deregulated nature of certain energy markets must also be taken into account when considering the various information flows in the smart grid. For example, in a deregulated market, a particular DNO could be responsible for a certain area in which it owns and maintains the physical distribution infrastructure, but consumers in that area are free to select any energy supplier. As explained by Nizar et al. [201], energy suppliers could gain a competitive advantage from having information about individual customer behaviour and preferences. For example, this information could enable suppliers to segment the market and use the most effective means of marketing to individual segments [201]. However, it

might be possible for energy suppliers to have close business relationships with DNOs, or even for a single company to be both a DNO and energy supplier. If the consumption information from the DNO were shared with a particular energy supplier (e.g. within the same company), this could give that supplier an unfair information advantage over its competitors. In practice, this type of information flow between DNOs and suppliers must be controlled. For example, in the case of a single company acting as both a DNO and energy supplier, the company could be required to establish a type of internal *Chinese wall* to prevent this information flow between the its DNO and energy supplier operations. However, if the DNO only receives the information in aggregated form, as described in the above privacy requirements, this automatically limits any potential flow of information from the DNO to the energy supplier. The aggregated information is sufficient for the DNO to perform its required function, but would not enable any energy supplier to gain an unfair competitive advantage. This again confirms that these privacy requirements are beneficial for consumers as well as the other participants in the smart grid. The following sections present and evaluate a solution that meets these functional, security and privacy requirements.

5.2 Privacy-Enhancing Communication Architecture

Chapter 3 discussed various approaches that have already been proposed to enhance privacy in smart meter communication. However, many of those solutions have limitations (e.g. scalability) or cannot be used in all three information flows. Furthermore, the analyses in Chapter 4 show that, even when used for their designed purpose, some proposed protocols exhibit security or privacy flaws and thus do not meet the requirements defined above. As explained in the gap analysis section in Chapter 3, previous research in this area has not investigated the use of a trustworthy intermediary in the communication path between the consumers and the service providers.

To fill this gap, this section presents a new privacy-enhancing communication architecture based on the Trustworthy Remote Entity (TRE). As introduced in Chapter 1, the TRE provides similar functionality as that of a computational Trusted Third Party (TTP). However, unlike a TTP, which is often blindly trusted, the TRE provides strong guarantees of its trustworthiness. This architecture assumes that the TRE is mutually trusted by all participants and shows how this entity can be used to enhance communication privacy in the smart grid. Chapter 6 and Chapter 7 explain how the TRE is implemented and how its trustworthiness is established.

5.2.1 Overview

An overview of the new privacy-enhancing communication architecture is shown in Figure 5.1. In this architecture, all communication between consumers and service providers takes place via the TRE. Since there is no need for any participants to hide their identities

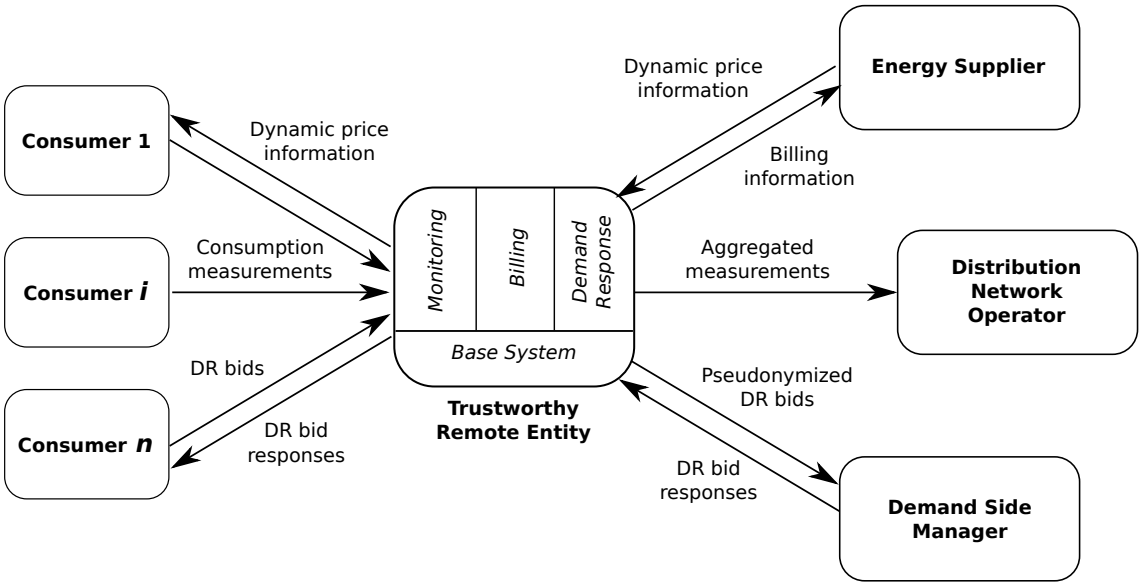

Figure 5.1: Overview of the privacy-enhancing communication architecture

from the TRE, all communication with the TRE takes place over secure authenticated channels providing confidentiality and integrity protection with respect to external adversaries as well as strong mutual authentication. This can be achieved using Transport Layer Security (TLS) with mutual authentication. For each information flow, the TRE performs specific information processing as part of a privacy-enhancing communication protocol. These protocols are described in the following subsections with reference to a single TRE. In some implementations, a single TRE might have sufficient capacity to handle all communications and might be trusted by all participants. For example, a *microgrid*, which is a small, localized and self-sufficient group of energy producers and consumers, could likely be supported by a single TRE. However, for larger systems, such as national electricity grids, it is envisaged that there would be a network of TREs distributed throughout the grid, each providing identical functionality. This would allow the communication workload to be distributed over multiple TREs and would provide multiple redundancy and failure recovery contingency plans to increase the availability of the system. It would also give participants a choice of which TRE to use, which is advantageous from the perspective of establishing trust but increases the complexity of the communication protocols in order to avoid leaking private information. The use of multiple TREs is addressed in each of the following subsections and other implementation considerations are discussed in Section 5.3.

5.2.2 Network Monitoring

As shown by the functional requirements in the previous section, the network monitoring information flow is a unidirectional information flow from consumers to the DNO. The main characteristics of the flow are that it requires high temporal resolution (i.e. measurements are required for each time period) but does not require the maximum spatial

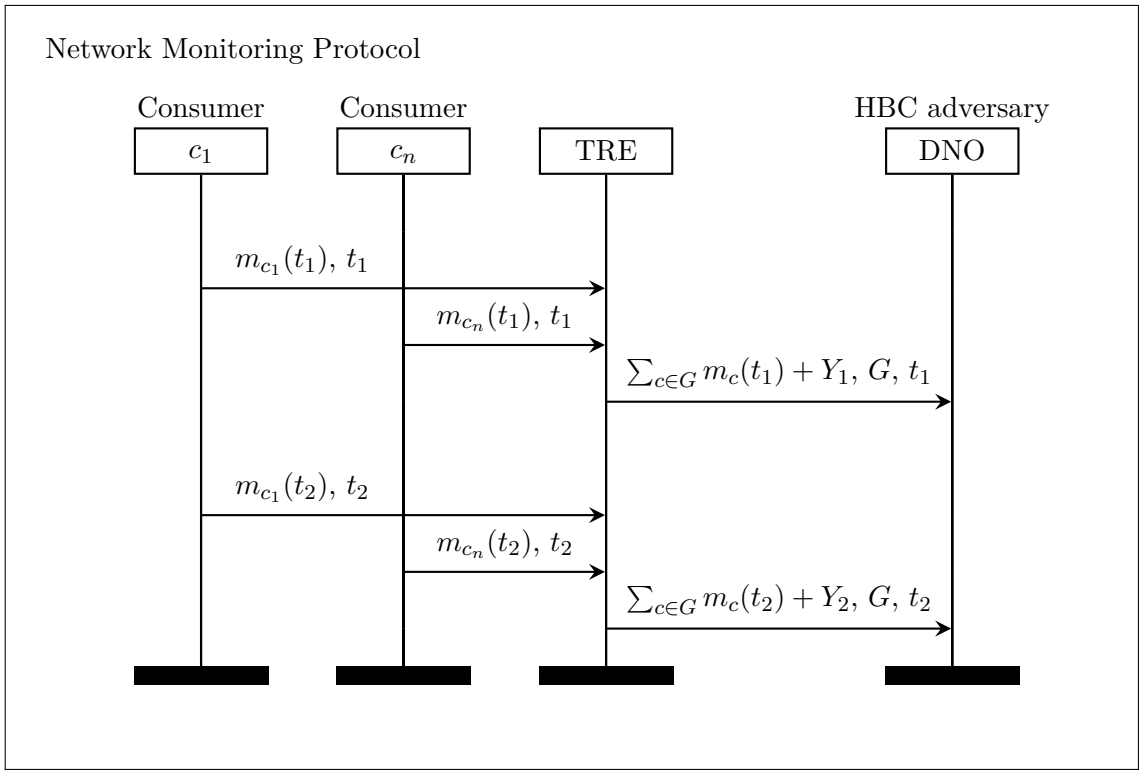

Figure 5.2: Spatial aggregation of consumption measurements

resolution (i.e. measurements from multiple consumers can be aggregated). Taking advantage of these characteristics, the new privacy-enhancing network monitoring protocol uses the TRE to perform spatial aggregation of consumption measurements from groups of consumers in order to prevent the DNO from linking individual measurements to specific consumers or linking together multiple measurements from the same consumer. The TRE is also used to apply data perturbation to the aggregated result to achieve differential privacy [87].

The network monitoring communication protocol is shown in Figure 5.2. In this protocol, all consumers are divided into aggregation groups defined by the DNO. Aggregation groups are usually defined such that each group represents a specific sector of the distribution network. In Figure 5.2, only two out of n consumers in a particular group are shown. At the end of each time period t, each consumer c sends the consumption measurement $m_c(t)$ for that period to the TRE. The TRE first performs bounds checking on these measurements to mitigate against false data injection attacks. For example, negative values or values that exceed a consumer's maximum possible consumption are excluded from the aggregation and an alert is raised.

At this point, the TRE now holds a dataset containing all the individual consumption measurements for time period t. For each aggregation group G, the TRE performs a statistical query on this dataset to compute the summation of the relevant consumption measurements using a differentially private mechanism. This query mechanism determines

the exact summation and then adds calibrated random noise that is drawn from a Laplace distribution, Lap(λ), which has the probability density function $h(y) \propto exp(-|y|/\lambda)$ (mean = 0, standard deviation = λ) [88]. The TRE then sends the result of this query to the DNO:

$$\text{TRE} \rightarrow \text{DNO:} \quad \left(\sum_{c \in G} m_c(t) \right) + Y \qquad \text{where: } Y \sim \text{Lap}(1/\epsilon)$$

This query mechanism is therefore ϵ-indistinguishable [88], where ϵ is a configurable privacy parameter. The addition of random noise is necessary to mitigate against a variant of the *set-difference* attack [241] in which the DNO compares two overlapping aggregation groups that differ by a single consumer in order to learn that individual's consumption. Whilst the TRE itself does not permit overlapping groups, the differential privacy guarantee ensures that this attack is not possible even if the adversary has arbitrary additional information. The sensitivity of the added noise is calibrated to mask the presence or absence of any single consumer in the aggregate [88]. Since the TRE only performs a single query per group from each dataset, there is no need to increase the amount of noise added or consider the DNO's privacy budget.

Consumers' privacy is technically preserved by the aggregation operation if there are at least two consumers in each group ($|G| \geq 2$). However, in practice larger aggregation groups would be used. As $|G|$ increases, the percentage error introduced by the random noise decreases since it is calibrated to mask the presence or absence of a single consumer. In all practical implementations, this error will be less than other errors, such as those caused by electrical losses in the distribution network. The maximum $|G|$ depends on implementation details such as the bandwidth and computational capacity of the TRE, as discussed in the next chapter.

It is not necessary for all consumers in a particular sector of the distribution network to communicate via the same TRE. If multiple TREs are used, consumers on each TRE are divided into aggregation groups (e.g. according to sector) and then, if need be, the DNO can compute the sum of the results from the different TREs (e.g. to determine the total consumption for a particular sector). However, if multiple TREs are used, consumers can only choose a TRE that is also used by other consumers in the same aggregation group, in order to ensure that there are always two or more consumers in each aggregation group on each TRE. In practice, provided that the required trust relationships can be established, consumers will choose the TRE that enables them to be part of the largest possible aggregation group in order to maximize their privacy. This means that the aggregation groups will naturally concentrate over as small a number of TREs as possible. Although membership of an aggregation group might appear to be similar to the type of quasi-identifier that would necessitate the use of k-anonymity, this is not the case because in this protocol, privacy is enhanced through aggregation of measurements rather than anonymization.

Overall, this protocol aims to achieve the same outcome as other spatial aggregation techniques [42, 107, 7, 156, 226, 162, 96, 99] without requiring any modification to the

consumers' smart meters and only minimal configuration changes to the DNO. Importantly, this protocol does not increase the number of messages sent by consumers or the number of cryptographic operations performed by smart meters and does not require any new cryptographic capabilities on the smart meters.

5.2.3 Billing

The billing information flow is a unidirectional information flow from consumers to the energy supplier. The main characteristics of this flow are that it requires high spatial resolution (i.e. measurements cannot be aggregated over multiple consumers) but does not require the maximum temporal resolution (i.e. measurements from a single consumer can be aggregated over time). The new privacy-enhancing billing protocol therefore uses the TRE to perform temporal aggregation of billing information from each consumer and reports each consumer's total bill to the energy supplier at the end of each billing period. Although this still reveals some information about the consumer (e.g. it might be possible to infer the number of occupants in the residence based on this total bill), it achieves the same level of privacy as was available before smart meters.

The billing protocol is shown in Figure 5.3. At time t the supplier notifies the TRE of the current energy price per unit p_t, which the TRE broadcasts to consumers. By verifying that p_t was sent by the TRE, consumers are assured that this is the price that will be applied. Although they are shown in the figure, these price information messages are not strictly part of the billing information flow since they are broadcast to all consumers. The billing information flow is unidirectional because it only includes the following messages sent from the consumers, via the TRE, to the supplier. Each consumer c sends a consumption measurement $m_c(t)$ for time period t to the TRE, which performs bounds checking as in the network monitoring protocol. The TRE then multiplies the consumption measurement with the applicable price per unit for the respective time period and adds the result to a running total bill for that consumer $L_c(t)$, which is held by the TRE:

$$L_c(t_n) \;=\; L_c(t_{n-1}) + (m_c(t) \times p(t))$$

At the end of a billing period (e.g. after time period t_n), the TRE sends each consumer's aggregated total $L_c(t_n)$ to the energy supplier and resets the running total to zero. For consumers who feed energy back into the grid, the same protocol can be used. The quantity fed back into the grid in each time period is multiplied by the respective feed-in tariff (which can be different from the consumption tariff) and the result is credited to the consumer's running total held by the TRE.

The temporal aggregation period is dynamically defined by the supplier but it must exceed the minimum value specified by the regulator and enforced by the TRE to protect privacy (e.g. weeks or months). The maximum temporal aggregation period again depends on the implementation of the architecture and the capacity of the TRE (i.e. how many consumers can be served by each TRE). In general, it is not necessary to apply differential

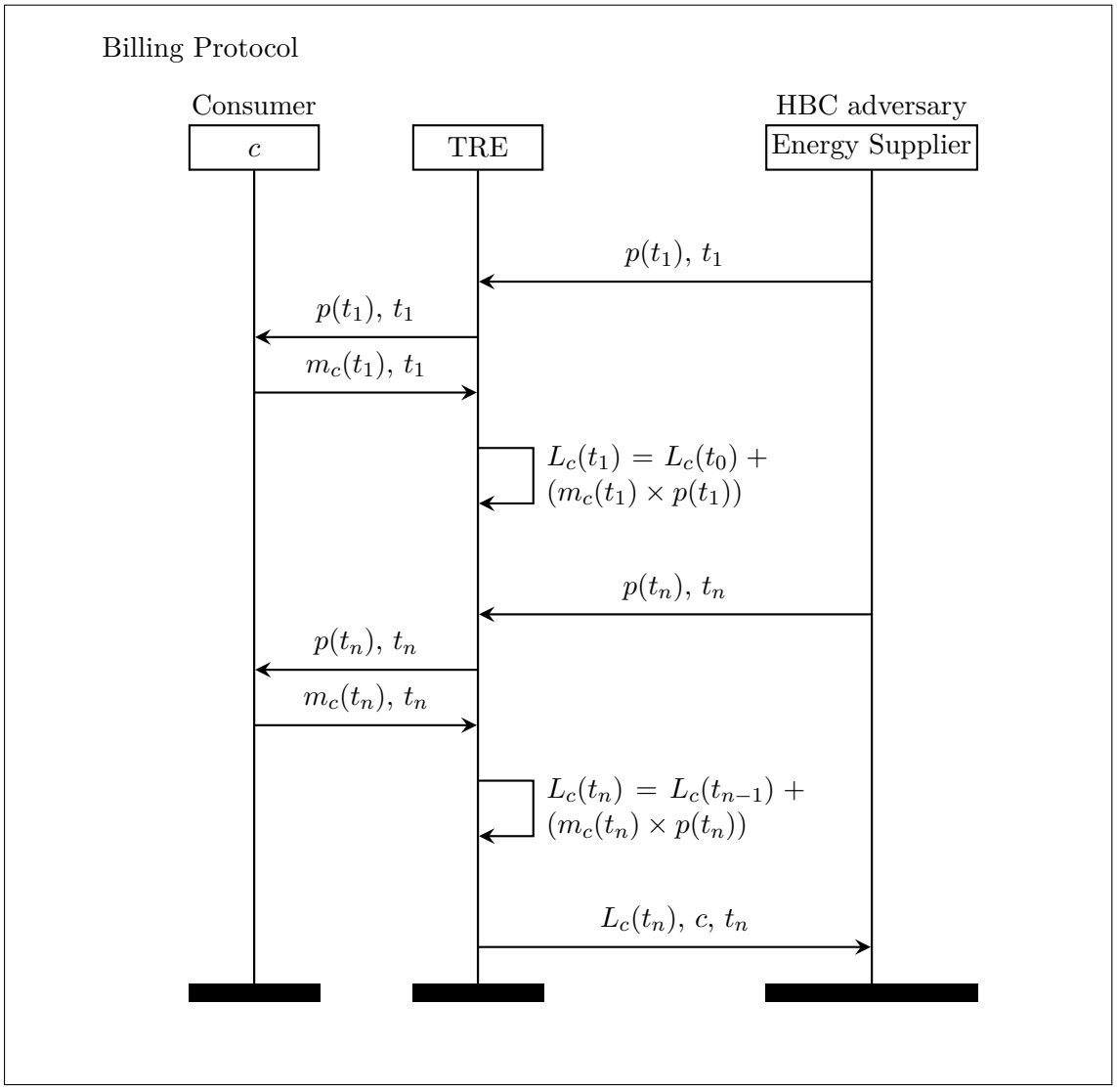

Figure 5.3: Temporal aggregation of billing information

privacy in this case because the supplier cannot define overlapping time periods and thus cannot learn anything other than the temporal aggregate. If differential privacy guarantees were required, this could be implemented in a similar manner to the network monitoring protocol described above. However, the addition of random noise would cause inaccuracies on consumers' bills and thus is not likely to be used in practice.

Since temporal aggregation is used as the primary mechanism for preserving privacy in the billing protocol, consumers may use any TRE for this protocol, regardless of how many other consumers are also using it. This allows consumers to use any TRE they consider to be trustworthy. If they wish, consumers can also switch TREs at any time, provided that consumers use the same TRE for a sufficient number of time periods in each billing period to create a temporal aggregate that adequately protects their privacy. In practice, consumers should use a single TRE for the entire billing period, if possible, since this maximizes their privacy. To maximize the efficiency of the system, this protocol can be combined with the network monitoring protocol since both use the same individual consumption measurements as inputs. The combination of these protocols must still satisfy all constraints on the choice of TRE from both protocols.

Overall, this protocol aims to achieve the same result as other privacy-preserving billing methods [221, 77, 138] without requiring modifications to the smart meters or increasing the number of messages sent by consumers.

5.2.4 Demand Response

Unlike the previous information flows, the DR information flow involves bi-directional communication between consumers and the DSM. In the DR information flow, techniques such as spatial or temporal aggregation cannot be used directly for multiple reasons: Firstly, in demand bidding, the bids cannot be spatially aggregated over multiple consumers because each bid contains both quantity and price information. Since each consumer can bid at a different price, the bid quantities cannot be aggregated. Secondly, consumers might only be able to reduce consumption by specific quanta (e.g. by disconnecting a particular load), which means that bids cannot be partially accepted and thus cannot be spatially aggregated. Thirdly, in the bidding process, each bidder must receive an individual decision from the DSM either accepting or rejecting the bid so that the bidder knows how to proceed and thus these decisions cannot be spatially aggregated. Fourthly, since the purpose of DR is to reduce demand at specific times, bids cannot be temporally aggregated because the DSM must make decisions based on the most recent bids in each time period. Furthermore, the addition of random noise to individual bid quantities would severely diminish the usefulness of these bids, and thus affect the functionality of the DR information flow. This means that most of the privacy-enhancing approaches used in previous proposals, including homomorphic encryption, secret splitting, consumer-side aggregation, and adding random noise, cannot be used in this information flow. Although cryptographic secure multiparty computation (SMC) protocols can in general be used for

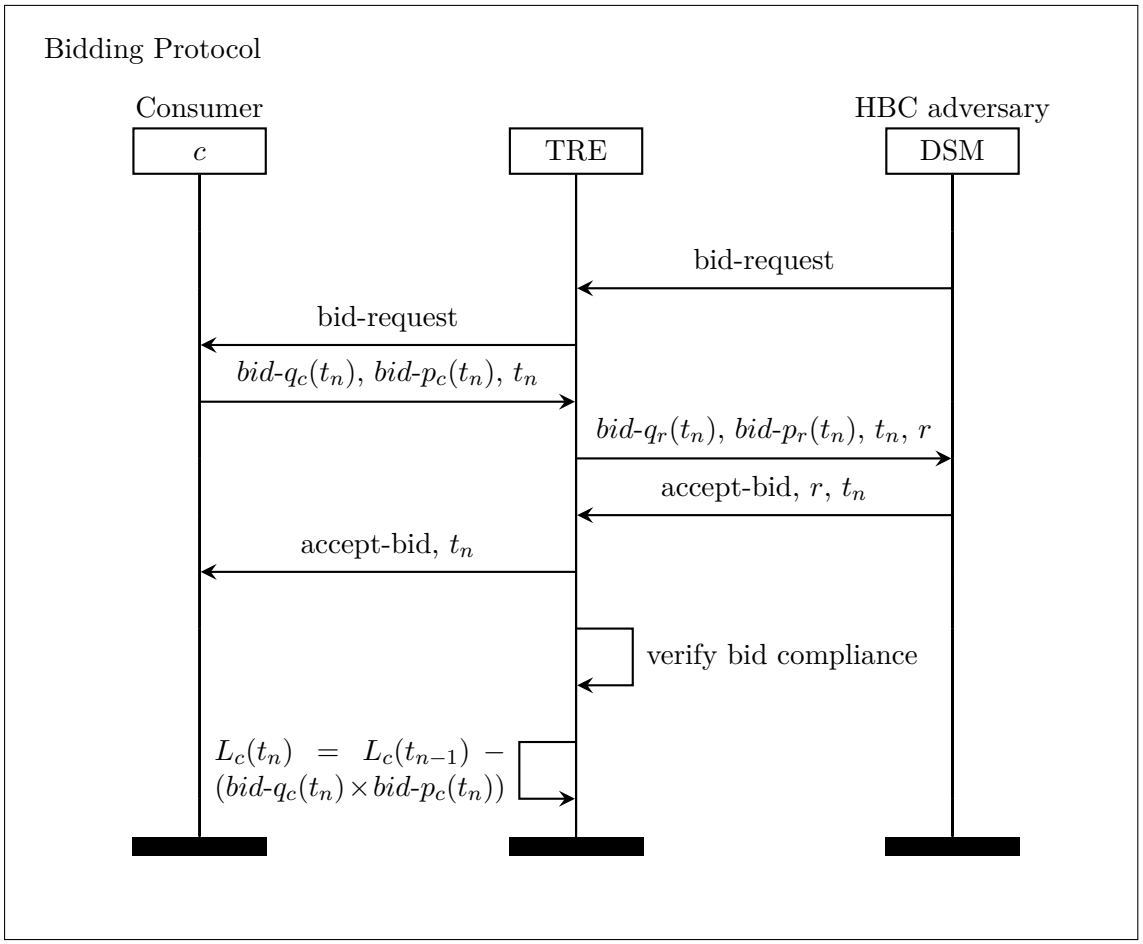

Figure 5.4: Anonymization and temporal aggregation for demand bidding

auctions, in this scenario these protocols would incur infeasibly high computational and communication costs since the auctions could involve tens of thousands of participants. Furthermore, to protect consumers' privacy, the bid decisions should only be communicated to the individual bidders rather than published to all participants as is usually the case for the results of cryptographic SMC protocols. However, as explained in the privacy requirements in Section 5.1.4, this bidding information is privacy-sensitive and thus cannot be sent directly to the DSM. Therefore, a new approach is required to to provide communication privacy in this bi-directional information flow.

The privacy-enhancing demand bidding protocol is shown in Figure 5.4. In this protocol, the TRE is used to provide a combination of pseudonymization and temporal aggregation. When the DSM initiates a DR event, it sends the TRE a request for bids from consumers in a specific area, which the TRE passes on to these consumers. The TRE ensures that at least some minimum number k of consumers are included in the request, where k is a configurable privacy parameter.

For time period t_n, each participating consumer c creates a bid consisting of a quantity

($bid\text{-}q_c(t_n)$) and a price per unit ($bid\text{-}p_c(t_n)$) and sends this directly to the TRE:

$$c \rightarrow \text{TRE:} \quad (bid\text{-}q_c(t_n), bid\text{-}p_c(t_n))$$

To mitigate against false bid injection, consumers must authenticate themselves to the TRE and the TRE performs bounds checking on all bids. Although not shown in Figure 5.4, this means that the consumer's identity is included in all bids submitted to the TRE. Since the bids could be requested at any time, an adversary might perform a traffic analysis attack to try to detect which consumers do not place bids immediately following a request for bids. To mitigate against this, all consumers should reply to the request for bids, even if they do not intend to place a bid (i.e. they respond with a *null bid*), and a probabilistic encryption scheme must be used so that null bids are indistinguishable from actual bids for anyone except the TRE. In order to prevent linking through traffic analysis, the TRE waits until all bids have been received and then randomizes the ordering of the received bids. The TRE then sends the DSM a set of *pseudo-bids* where each pseudo-bid corresponds to one of the consumers' bids:

$$\text{TRE} \rightarrow \text{DSM:} \quad (bid\text{-}q_r(t_n), bid\text{-}p_r(t_n), r, t_n)$$

Each pseudo-bid includes a single-use random number r that serves as a pseudonym for the original bid but can only be linked to the original bid by the TRE. Instead of using a pseudonym for each consumer, a different random pseudonym is used for each bid, in order to prevent bids from the same consumer being linked together. However, since the bids are pseudonymized rather than aggregated, various data privacy attacks must be considered. Since the consumers' area is a quasi-identifier, the TRE enforces k-anonymity by ensuring that at least k consumers are included in any bid request. If k is sufficiently large (e.g. at least tens or hundreds of consumers), it is highly unlikely that all bids will be identical and thus ℓ-diversity is essentially achieved through the inherent diversity of the system. In this scenario it is not necessary to consider t-closeness because average information about each area is already available through the monitoring information flow and thus does not diminish consumers' privacy. From the DSM's perspective, the TRE appears to be a single large consumer who submits multiple bids for each DR event. The DSM can therefore use its existing processes and algorithms to select which pseudo-bids to accept and then send these decisions to the TRE which in turn notifies the individual consumers. The TRE can verify compliance with bid obligations by requesting and analysing consumption measurements from successful bidders.

Up to this point in the protocol, pseudonymization has been effectively used to prevent the DSM from linking bids to specific consumers. However, pseudonymization cannot be used to protect privacy with respect to the incentive payments for successful bidders since these must ultimately be paid to named consumers. Instead, the protocol uses temporal aggregation, in a similar manner to the billing protocol, to support incentive payments.

For each successful bid, the respective incentive payment (i.e. $bid\text{-}q_c(t_n) \times bid\text{-}p_c(t_n)$) is credited to an internal running total held by the TRE for each bidder. At the end of each billing period, the total incentive due to each bidder is sent to the DSM. However, since the bids are pseudonymized rather than aggregated, the DSM might be able to deduce a unique set of bids that can be added together to reach the total incentive payment for a particular consumer, and thus link these bids to the consumer. The risk of this attack is increased if a consumer exhibits unusual bidding behaviour or if there are relatively few consumers placing bids via a particular TRE. It should be noted that, in the general case, this attack is not scalable and cannot be used on a large percentage of the bids. However, this attack can be completely mitigated if the bidding protocol is combined with the billing protocol on the same TRE. Instead of using a separate running total for the incentives, the TRE deducts these from the consumer's running bill $L_c(t)$ from the billing information flow:

$$L_c(t_n) \;=\; L_c(t_{n-1}) + (m_c(t) \times p(t)) - (bid\text{-}q_c(t_n) \times bid\text{-}p_c(t_n))$$

Since the individual consumption values are aggregated rather than pseudonymized, this type of attack is not possible in the billing information flow and therefore, if the bid incentives are deducted from this total, they are masked by the billing information. In this way, neither individual measurements nor individual bids can be linked to specific consumers by the energy supplier or DSM, even if these two HBC adversaries collude. If the DSM and supplier use separate billing systems, the TRE can be configured to provide a statement to the supplier at the end of each billing period showing the sum of the incentives that have been credited in that period. The optimal approach is for a consumer to use the same TRE for all three information flows, so that the consumption measurements used in the monitoring and billing information flows can also be used to verify bid compliance and a single running bill can be used in both the billing and DR information flows.

Overall, this protocol aims to support all the functional requirements of the demand bidding process whilst ensuring that the DSM is unable to link bids to individual consumers and is therefore unable to detect if specific consumers have placed bids. Table 5.1 summarizes the various privacy attacks and mitigation strategies presented in this section.

5.3 Implementation Considerations

Although any concrete implementation of this new communication architecture will have to be designed for the specific context in which it is used, this section discusses some of the important implementation considerations for this architecture.

Table 5.1: Privacy attacks and mitigation strategies in the bidding protocol

Attacks	Mitigation strategies
External adversary:	
Traffic analysis between consumers and the TRE is used to detect which consumers have or have not placed bids.	Require all consumers to respond to every bid request, even with a null bid, and use probabilistic encryption to make all responses indistinguishable.
Colluding adversaries:	
Compare traffic between consumers and the TRE with the order in which bids are received at the DSM to link bids to specific consumers.	The TRE waits to receive all bids before replacing the identities with random pseudonyms and sending them to the DSM in a random order.
HBC adversary:	
Link bids to a specific consumer by requesting bids from only that consumer.	The TRE enforces that bid requests must apply to at least k consumers.
Learn a particular consumer's bid if all k bids are identical to each other (homogeneity attack [175]).	The system inherently exhibits ℓ-diversity because, for sufficiently large values of k, it is highly unlikely for all k consumers to submit identical bids.
Learn average information about a particular group of consumers from their bids (skewness and similarity attacks [163]).	There is no need to consider t-closeness because average information for each area is already available through the monitoring information flow.
Deduce which pseudo-bids uniquely add up to the total incentive amount for a particular consumer, thus linking these bids to a specific consumer.	Deduct bid incentives from the consumer's running bill used in the billing information flow.

5.3.1 Protocol Implementation

The protocols described in the previous section can be implemented using existing smart meter communication specifications. As an example, this subsection describes how these protocols can be implemented using the DLMS-COSEM specification, which is a mandatory requirement for all smart meters in the UK [265, 266]. The Device Language Message Specification (DLMS) provides the abstract concepts for modelling communicating devices whilst the Companion Specification for Energy Metering (COSEM) defines rules for data exchange with smart energy meters. The DLMS-COSEM specification has been standardized as the IEC 62056 series of standards. However, in certain cases, the protocols used the UK will deviate from the standardized version of DLMS-COSEM, as explained in the Great Britain Companion Specification [268].

The privacy-enhancing communication protocols presented in the previous section can make use of many existing DLMS-COSEM messages and, where changes are required, these are relatively minor additions to the specification. Examples of various DLMS-COSEM messages from the new communication protocols are shown in Table 5.2. In these protocols, all communication takes place over mutually-authenticated secure channels that provide confidentiality and integrity protection (e.g. TLS connections). However, for the sake of clarity, these examples only show the DLMS-COSEM messages.

The monitoring and billing protocols can implemented entirely using existing DLMS-COSEM commands and messages. In DLMS-COSEM, the smart meters are viewed as servers that respond to requests for information. To retrieve consumption measurements from consumer c, the TRE uses the COSEM GET command type to request the sum of the active power integrated over the last billing period (object ID = 1.1.1.8.0.65, attribute = 2) [134, 135]. An example of this type of command is shown in hexadecimal notation in row 1 of Table 5.2. Each consumer responds to the TRE with the requested information (e.g. 5 kWh) as shown in row 2. These are exactly the same messages that would be exchanged between the DNO/supplier and the consumers if these entities were communicating directly. When the DNO or energy supplier communicate with the TRE, they prefix a 32 bit group identifier to each message indicating the aggregation group for which the message is intended, as shown in row 3. The respective entity identifier is also included in each response from the TRE, as shown in row 4. Where a message concerns an individual consumer (e.g. a query of the consumer's temporally aggregated bill), the group identifier is replaced by the consumer's public identifier.

For the bidding protocol, three new COSEM messages are defined: The first is a new GET message that allows the DSM to request bids from specific groups of consumers. This message is very similar to the GET request for consumption values except that it requests a different attribute value (object ID = 1.1.1.8.0.65, attribute = 4). When this message is sent from the DSM to the TRE, the group identifier is used to indicate which group of consumers are invited to submit bids. The second new message allows consumers to place bids consisting of quantity and price information, as shown in row 5 of Table 5.2.

Table 5.2: Examples of DLMS-COSEM commands for the new protocols

1. Request latest consumption measurement (TRE → c): $\underbrace{00\ 01\ 00\ 01\ 00\ 01}_{\text{Preamble}}\ \underbrace{00\ 0D}_{\text{Length}}\ \underbrace{C0\ 01\ 81}_{\text{GET}}\ \underbrace{00\ 03}_{\text{Type}}\ \underbrace{01\ 01\ 01\ 08\ 00\ 65\ 02}_{\text{Object ID and attribute}}\ 00$
2. Send latest consumption measurement (c → TRE): $\underbrace{00\ ...\ 01}_{\text{Preamble}}\ \underbrace{00\ 0A}_{\text{Length}}\ \underbrace{C4\ 01\ 81}_{\text{Response}}\ \underbrace{00\ 05}_{\text{Type}}\ \underbrace{xx\ xx\ xx\ xx}_{\text{Quantity}}\ 00$
3. Request latest consumption aggregate (DNO → TRE): $\underbrace{xx\ xx\ xx\ xx}_{\text{Group ID}}\ \underbrace{00\ ...\ 01}_{\text{Preamble}}\ \underbrace{00\ 0D}_{\text{Length}}\ \underbrace{C0\ 01\ 81}_{\text{GET}}\ \underbrace{00\ 03}_{\text{Type}}\ \underbrace{01\ 01\ 01\ 08\ 00\ 65\ 02}_{\text{Object ID and attribute}}\ 00$
4. Send latest consumption aggregate (TRE → DNO): $\underbrace{xx\ xx\ xx\ xx}_{\text{Group ID}}\ \underbrace{00\ ...\ 01}_{\text{Preamble}}\ \underbrace{00\ 0A}_{\text{Length}}\ \underbrace{C4\ 01\ 81}_{\text{Response}}\ \underbrace{00\ 05}_{\text{Type}}\ \underbrace{xx\ xx\ xx\ xx}_{\text{Quantity}}\ 00$
5. Place a bid (c → TRE): $\underbrace{00\ ...\ 01}_{\text{Preamble}}\ \underbrace{00\ 0F}_{\text{Length}}\ \underbrace{C4\ 01\ 81}_{\text{Response}}\ \underbrace{00\ 05}_{\text{Type}}\ \underbrace{xx\ xx\ xx\ xx}_{\text{Quantity}}\ \underbrace{05}_{\text{Type}}\ \underbrace{xx\ xx\ xx\ xx}_{\text{Price}}\ 00$
6. Place a pseudo-bid (TRE → DSM): $\underbrace{xx\ xx\ xx\ xx}_{\text{Group ID}}\ \underbrace{xx\ xx\ xx\ xx}_{\text{Entity ID}}\ \underbrace{00\ ...\ 01}_{\text{Preamble}}\ \underbrace{00\ 0F}_{\text{Length}}\ \underbrace{C4\ 01\ 81}_{\text{Response}}\ \underbrace{00\ 05}_{\text{Type}}\ \underbrace{xx\ xx\ xx\ xx}_{\text{Quantity}}\ \underbrace{05}_{\text{Type}}\ \underbrace{xx\ xx\ xx\ xx}_{\text{Price}}\ 00$

The format of the corresponding pseudo-bids placed by the TRE is very similar but is prefixed with the group identifier and a random entity identifier generated for that bid, as shown in row 6. The third new message is used by the DSM to communicate its decision about each bid. The 'accept' and 'reject' values are simply defined as constants. The DSM includes both the group identifier and the entity identifier in the decisions it sends to the TRE and the TRE uses these to forward the decisions to the correct consumers. Since these new messages only involve constant values or types that are already used in other COSEM messages (e.g. quantity of power/energy or price information), they are relatively simple additions to the existing DLMS-COSEM specifications.

Any of the above commands can be combined into a single message if they share the same sender and receiver. For example, the TRE can send an single message to a consumer that includes both the GET commands to retrieve the most recent consumption measurement and request bids for the next time period.

5.3.2 Owners and Operators of TREs

From a practical perspective, the first consideration is who should own and operate the TRE. If this type of functionality were provided by a TTP, this would be an important question because the reputation of the operator would have a significant influence on the trustworthiness of the TTP. However, unlike a TTP, the TRE provides technical guarantees of its trustworthiness that are independent of its owner and operator. The software design of the TRE and the use of Trusted Computing allow all TREs to provide the same type of guarantee irrespective of their owners and operators. The only exception to this is the class of entities who can mount advanced hardware-level attacks against the TRE, as described in the next chapter. However, entities with these specialized capabilities are relatively rare. Therefore, the TRE can be owned and operated by essentially any entity involved in the communication architecture.

From an efficiency perspective, since the TREs are used as intermediaries between all communicating parties, it would be advantageous for the TREs to be situated centrally in the communication networks. In general, communication network operators would therefore be well placed to operate the TREs as part of their communications infrastructure. In certain countries, the overall design of the smart metering system might also suggest other entities who could operate the TREs. For example, in the UK, the Data Communication Company (DCC) is mandated to provide communication services for all smart meters. As explained in Chapter 3, the DCC encompasses both the communication service providers, who provide the communication links to smart meters, as well as the data service provider, who is responsible for processing, storing and distributing smart meter measurement data. In the current design, all communication between smart meters and the DNO or energy supplier takes place via the DCC. This makes the DCC the ideal operator for the TREs in the UK, either as part of the communication infrastructure or the data processing functionality. In a system in which multiple TREs are used, there could be many different TRE owners and operators.

Another practical consideration is the owner or operator's motivation for providing the TRE functionality. Again this depends on the specific scenario, but in general, either regulation or financial incentives (or a combination thereof) could be used. A regulatory solution might involve the energy regulator mandating the use of TREs for specific information flows, whereas a market-based solution would see the DNO, energy supplier and DSM, as well as possibly the consumers paying for the use of the TRE.

5.3.3 Geographic Locations of TREs

Other practical considerations include the number of TREs required and where they will be geographically situated. The number of TREs required is determined by the number of consumers in the system and the capacity of each TRE. The capacity of each TRE depends on the hardware platform on which the TRE is run. Chapter 6 provides a precise evaluation of this capacity and shown that a TRE implemented on current consumer PC hardware

can support approximately 20,000 consumers. This means that in the UK, supporting the communication requirements of all the planned 53 million smart meters [270] would require fewer than 3,000 TREs (although this number could be reduced significantly by using higher capacity TREs). From a network perspective, the TRE is very similar to a generic head-end system for communicating with smart meters and thus similar design techniques could be used to dimension the TRE's computational capacity. Since the TREs communicate more frequently with consumers than with other entities, and since many of the consumers who connect to a specific TRE are in the same geographic area for spatial aggregation, it could be argued that the TRE should be situated in close geographic proximity to these consumers since this would minimize communication latency. However, the latency of the communication is not a critical parameter in this architecture, and so situating the TREs further away from the consumers they serve would not have a significant impact on the system. This means that other practical considerations, such as ease of maintenance and replacement can therefore be taken into account when deciding on the location of TREs. In terms of availability, as with other critical infrastructure, it is important to consider geographic redundancy so that an incident at a particular location would not affect the availability of all TREs. Since this architecture does not require major changes to smart meters, it could also be implemented incrementally with additional TREs being added as the need arises.

5.3.4 TRE Failure Recovery

The use of multiple TREs is advantageous in terms of the availability of the system. As described in the next chapter, the design of the TRE provides a mechanism for creating an encrypted backup of specific data held by the TRE (e.g. protocol state) and restoring this backup on another TRE that is in exactly the same software state. In the new smart meter communication architecture presented above, this mechanism would be used to create periodic backups of the aggregation groups that have been configured for the monitoring information flow as well as the running total for each consumer's bill in the billing information flow. These backups can be created frequently (e.g. once every time period) because the size of this protocol state is relatively small and does not increase over time. Since these backups are encrypted, they can be stored by any entity without having to establish additional trust relationships. The only requirement is that the integrity and availability of the encrypted backups must be maintained so that they can be successfully restored when needed. If a particular TRE fails, its most recent backup will be restored to another TRE that is in exactly the same state as the original TRE was in at the time the backup was created. The consumers who had previously used this TRE as well as the DNO, energy supplier and DSM must all be notified of this failure and given the address of the new TRE to which the protocol state has been restored. All participants would then continue to use the new TRE as if nothing had changed. For example, in the billing information flow, the consumers would not need to resend previous measurements

since their running totals would have been backed-up and restored to the new TRE. This process essentially transfers the trust relationships that had already been established with the failed TRE over to the new TRE. The technical details of the TRE's backup and restore mechanism are presented in the next chapter.

5.4 Evaluation of Architecture

This section presents the evaluation of the new protocols that constitute the privacy-enhancing communication architecture. Two different methodologies are used to evaluate each of the protocols: Firstly, the security and privacy properties of the protocols are modelled and automatically analysed using the Casper-Privacy tool presented in the previous chapter. Secondly, the protocols are compared to other proposals in terms of their efficiency and practicality.

For the protocol analysis, the full input script of each protocol for use with the Casper-Privacy tool is presented in Appendix A. The security and privacy properties that have been analysed were selected based on the requirements defined in Section 5.1. In each of these models, all communication is assumed to take place over secure channels that provide confidentiality and integrity protection with respect to an external adversary (e.g. mutually authenticated TLS connections). This is modelled using the channel specifications under the `#Channels` heading in the input scripts. In all cases, it would be possible to include more consumers and time periods in the models but these would increase the time and computational requirements of the analysis and there is nothing in the protocols to suggest that including additional consumers or continuing the analysis for more time periods would reveal any attacks. As explained in the previous chapter, one of the main limitations of the Casper-Privacy tool is that it can only perform bounded analysis. However, this type of analysis is still sufficient to identify security and privacy flaws in other smart meter communication protocols, as shown in the previous chapter.

For the efficiency and practicality evaluation, the new protocols are compared to other proposals in terms of three main aspects: Firstly, the number of messages sent by smart meters in each protocol is compared. This is an important efficiency metric because smart meters will usually be distributed over wide geographic regions and thus will not all be connected via high speed communication links. In some cases, relatively low bandwidth bearers, such as GPRS, will be used for communication with smart meters [137]. Secondly, protocols are compared based on the number of cryptographic operations performed on the smart meters for each consumption measurement. Since smart meters are designed to be cost-effective measurement devices, they have limited computational capabilities and thus minimizing the number of computationally expensive cryptographic operations they have to perform increases efficiency and reduces costs. Thirdly, the types of cryptographic operations required for each protocol are compared. Protocols requiring cryptographic operations that are not widely used are less practical to implement than those that only use standard cryptographic functions, which are usually already included in the smart

meter specifications [265, 266]. For example, cryptographic operations that are not supported by any of the cryptographic libraries available for embedded systems would have to be implemented before the protocol could be used on real smart meters. Furthermore, non-standard cryptographic primitives are significantly more likely to contain flaws than widely-used primitives that have benefited from more extensive testing and verification. For example, the Open Smart Grid Protocol (OSGP) [97] was recently shown to contain serious security vulnerabilities due to its use of non-standard cryptographic primitives [143].

5.4.1 Network Monitoring

Enhanced Casper-Privacy Analysis

The model used to analyse the network monitoring protocol is shown in Listing A.9 in Appendix A. All communication is assumed to take place over secure channels that provide confidentiality and integrity protection with respect to an external adversary (e.g. mutually authenticated TLS connections). Similarly to Figure 5.2, this model consists of two consumers, a TRE and a DNO and the analysis covers two consecutive time periods. There is an external DY adversary and the DNO is modelled as an HBC adversary. Since this analysis is done in the symbolic paradigm, it is not possible to accurately represent the addition of random noise to the aggregated measurements. However, since the aggregation groups are fixed for this model, the DNO cannot perform a set-difference type attack [241]. In this model, the following security properties are analysed:

- **Secrecy:** Individual consumption measurements are only known by the respective consumers and the TRE.
- **Authentication:** The consumers and the TRE agree on the individual measurement values that have been sent and the TRE and DNO agree on the aggregated measurement values.

The following privacy property is analysed:

- **Unlinkability:** The DNO cannot link any measurements to specific consumers and cannot link together any measurements from the same consumer.

The analysis has shown that all of these properties hold against both the external DY adversary and the internal HBC adversary. The fact that the analysis completes, shows that the protocol reaches its final state and thus satisfies all the functional requirements of the monitoring information flow (i.e. that the DNO receives the sum S_G of consumption measurements from group G). The security properties show that the encrypted and mutually authenticated communication channels between the consumers and the TRE both protect consumers' privacy with respect to external adversaries and prevent false data injection attacks caused by falsified measurements from external adversaries. It should be noted that falsified measurements from legitimate smart meters are not directly addressed

by this solution, since this is an orthogonal problem. However, the bounds checking performed by the TRE ensures that even falsified measurements from legitimate smart meters are restricted to realistic values, in order to prevent significant pollution of the aggregate. For the analysis of the privacy properties, an additional inference rule is added to capture the algebraic equivalence between the sum of the individual consumption measurements and the aggregate provided by the TRE for that time period. Even with this additional capability, the HBC adversary is unable to compromise consumers' privacy through this protocol. Therefore, this protocol meets all the functional, security and privacy requirements for the network monitoring information flow.

Efficiency and Practicality

Apart from the addition of an initial procedure to establish the trustworthiness of the TRE (explained in the next chapter), this protocol does not require any modification to current smart meter specifications [265, 266].

In terms of communication efficiency, this new protocol is almost equivalent to the non-privacy-preserving case of direct communication between consumers and the DNO. The number of messages sent by the smart meters does not increase. The size of the messages is also unchanged, except for a slight increase in the size of the first message between each participant and the TRE, through which the trustworthiness of the TRE is established (as explained in the next chapter). The new protocol therefore achieves higher communication efficiency than proposals that increase the number of messages sent by smart meters [107, 156, 96, 226, 99].

In terms of computational efficiency, the new protocol does not increase the number of encryption operations performed by the smart meters. In total, the TRE and DNO only perform one additional encryption and decryption operation per aggregation group in order to communicate the aggregate securely. This is therefore more computationally efficient than proposals in which the number of encryption or decryption operations performed by smart meters is increased [107, 226, 7, 156]. Furthermore, since consumers communicate only with TRE, the workload of the DNO is significantly reduced.

In the new protocol, the aggregation of consumption measurements is performed before the additional noise is added to achieve differential privacy guarantees. This avoids the need for each smart meter to add noise as in other protocols [42, 7, 220, 238] and ensures that only the correct amount of noise is added. This also allows the TRE to perform bounds checking on the individual consumption measurements before noise is added.

The new protocol only requires standard symmetric and asymmetric cryptographic primitives, which are widely used in other contexts and have relatively well-tested implementations available. These are also already included in some smart meter specifications [265, 266, 267]. This new protocol is therefore more practical to implement than proposals requiring new cryptographic functionality on smart meters, such as homomorphic encryption capabilities [162, 107, 46, 227, 170, 96, 137, 220, 229].

5.4.2 Billing

Enhanced Casper-Privacy Analysis

The model used to analyse the billing protocol is shown in Listing A.10 in Appendix A. Similarly to Figure 5.3, this model consists of a consumer, a TRE and an energy supplier and the analysis covers two consecutive time periods. There is an external DY adversary and the energy supplier is modelled as an internal HBC adversary. In this model, the following security properties are analysed:

- **Secrecy:** Individual consumption measurements are only known to the TRE and the respective consumers.
- **Authentication:** The consumer and the TRE agree on the prices and individual measurement values and the TRE and energy supplier agree on the temporally aggregated bills.

The following privacy property is analysed:

- **Unlinkability:** The supplier cannot link individual measurements to specific consumers and cannot link together multiple measurements from the same consumer.

The analysis has shown that all of these properties hold against both an external DY adversary and an internal HBC adversary. The fact that the analysis completes, shows that the protocol reaches its final state and thus satisfies all the functional requirements of the billing information flow (i.e. that the energy supplier receives the total bill $L_c(t_n)$ for consumer c up to time period t_n). As before, the security properties show that the encrypted and mutually authenticated communication channels between consumers and the TRE protect consumers' privacy and prevent false data injection attacks with respect to external adversaries. For the analysis of the privacy properties, an additional inference rule is again added to represent the fact that if the HBC adversary can link together multiple consumption measurements from a single consumer and knows the respective prices for those time periods, then the adversary can calculate the bill for that consumer and link this to the bill provided by the TRE. Even with this additional capability, the HBC adversary is unable to compromise consumers' privacy through this protocol. Therefore, this protocol meets all the functional, security and privacy requirements of the billing information flow.

Efficiency and Practicality

As with the network monitoring protocol, apart from establishing the trustworthiness of the TRE, this new billing protocol does not require any modification to current smart meter specifications. This protocol relies only on standardized and widely used cryptographic operations that are already included in smart meter specifications [265, 266]. In comparison, other proposals for privacy-preserving billing rely on less widely-used cryptographic capabilities, including commitment schemes [221], wavelet transforms [93] and

zero-knowledge proofs [191, 77, 190]. Molina-Markham et al. [190] have shown that it is computationally feasible to perform zero-knowledge proofs on current smart meters. However, they were required to develop a custom implementation of the required cryptographic primitives since these are not available in current embedded cryptographic libraries [190]. Since zero-knowledge proofs are not widely used, they are also not generally supported by the hardware cryptographic accelerators found in many modern microprocessors and microcontrollers. Therefore their performance is likely to be significantly slower than the standard symmetric and asymmetric operations that can be accelerated in hardware. Using zero-knowledge proofs would also involve changing the specifications and updating the cryptographic software on all smart meters. The new billing protocol is therefore at least as efficient and practical to implement as other proposals for achieving the same outcome.

5.4.3 Demand Response

Enhanced Casper-Privacy Analysis

The model used to analyse the demand bidding protocol in the DR information flow is shown in Listing A.11 in Appendix A. This model is similar to Figure 5.4 but includes two consumers, instead of one, as well as the TRE and DSM. The analysis covers two time periods in order to evaluate the unlinkability of multiple bids from the same consumer. For this analysis, various simplifications were made to the model without affecting the correctness of the analysis. The initial messages sent from the DSM and TRE requesting bids were removed because these are broadcast messages sent to all consumers. Since the DSM may use an arbitrary process for the selection of bids, this is not represented in the model. Instead the DSM responds to every bid with a bid decision, which could represent either acceptance or rejection. The steps in which the TRE verifies consumers' compliance with their bid obligations and credits the incentives to successful bidders' bills do not influence the outcome of this analysis because they are internal to the TRE and are therefore omitted from this model. In this model, the following security properties are analysed:

- **Secrecy:** Individual bids must only be known to the TRE and the respective consumers.
- **Authentication:** The consumer and the TRE must agree on the individual bids and bid decisions.

The following privacy properties are analysed:

- **Undetectability:** The DSM is unable to detect if a specific consumer has placed a bid.
- **Unlinkability:** The DSM cannot link pseudo-bids to specific consumers or to the actual bids placed by consumers.

- **Unlinkability:** The DSM cannot link together multiple bids placed by a single consumer.

The analysis has shown that all of these properties hold against both an external DY adversary and an internal HBC adversary. The fact that the analysis completes, shows that the protocol reaches its final state and thus satisfies the functional requirements of the DR information flow. In this protocol, the DSM receives the bids and the consumers receive the DSM's decisions all within the same time period, and the incentives for successful bids are credited to the respective bidders by the TRE. The security properties show that the encrypted and mutually authenticated communication channels between consumers and the TRE and between the TRE and the DSM prevent an external adversary from placing bids and protect the integrity and authenticity of the legitimate bids. Although not included in this model, the TRE would enforce that incentives are only credited to successful bidders who have complied with their bid obligations. The privacy properties show that even the DSM is unable to detect whether or not a specific consumer has placed a bid. The DSM is also unable to link any bids or pseudo-bids to any specific consumers and is unable to link together any bids or pseudo-bids from the same consumer.

Efficiency and Practicality

Apart from the initial steps to establish trust relationships with the TRE, this protocol does not require any changes to either the consumers or the DSM. From the consumers' perspective, the TRE appears to be fulfilling the role of a trustworthy DSM since consumers send their bids directly to the TRE and receive responses from this entity. From the DSM's perspective, the TRE appears to be a single large consumer who places multiple bids. The DSM can use any algorithm to select which bids to accept and it will receive notification of the total incentive payments for successful bids from the TRE at the end of the billing period. Therefore, the addition of the TRE to preserve privacy does not decrease the efficiency or practicality of this protocol compared to the non-privacy-preserving case in which consumers and the DSM communicate with each other directly.

The only other proposal from the literature that addresses a similar challenge is that by Rottondi et al. [225] in which they propose a privacy-friendly load scheduling framework based on the Shamir Secret Sharing scheme. Their framework allows a set of appliances to be scheduled by a set of potentially untrusted schedulers without revealing the time of use or energy consumption patterns of appliances, or disclosing the identities of the respective consumers [225]. However, their framework assumes a co-operative setting in which appliances will abide by the given schedule and thus they do not include the capability to verify compliance with the scheduling. Their framework also does not consider the use of incentives for consumers which, as discussed above, present various privacy challenges that necessitate the use of a trustworthy entity such as the TRE.

5.5 Summary

This chapter has presented a new privacy-enhancing communication architecture for the smart grid. The starting point for this architecture is a baseline system model that captures the functional requirements of current smart grid architectures as well as anticipated future requirements, such as residential demand response functionality. Using this baseline model as a reference, a set of functional, security and privacy requirements have been defined for the communication architecture. These requirements serve to protect consumers' privacy from internal HBC adversaries, and also protect service providers from data breaches through which an external adversary could retroactively become an internal HBC adversary.

To meet these requirements, a unified communication architecture combining all three major information flows has been proposed. In this architecture, all communications between consumers and the DNO, energy supplier and DSM are mediated by a TRE. In the network monitoring information flow, the TRE performs spatial aggregation of the consumption measurements from groups of consumers to prevent individual measurements being linked to specific consumers or to other measurements. The TRE adds a calibrated amount of Laplace-distributed random noise to this aggregate to achieve differential privacy guarantees and thus prevent set-difference type attacks. In the billing information flow, the TRE performs temporal aggregation of consumption measurements from each consumer to achieve the same level of privacy available before smart meters. To maximize efficiency, the monitoring and billing protocols can be combined by the TRE since they both use the consumption measurements from smart meters as inputs. In the bidirectional demand response (DR) information flow, spatial and temporal aggregation approaches cannot be used directly in protocols such as demand bidding. To enhance consumers' privacy in this information flow, the TRE combines pseudonymization and temporal aggregation. Consumers send bids to the TRE, which replaces their identities with random one-time pseudonyms. The DSM selects bids as usual, using any algorithm, and responds to the TRE. The TRE then notifies the respective consumers and credits the incentives to their running total bills from the billing information flow.

In terms of implementation considerations, TREs can be owned and operated by almost any entity since a TRE's trustworthiness is established through technical mechanisms independently of the reputation of the operator. Communication network operators or entities such as the DCC in the UK are ideally situated to operate the TRE, possibly as a paid service. Although small systems, such as microgrids, can be supported by a single TRE, larger systems such as the national smart metering infrastructure in the UK would require in the order of a thousand TREs to support communication with all 53 million smart meters. The use of multiple TREs increases consumers' trust in the system by giving them a choice of TREs. The TRE's encrypted backup and failure recovery capabilities allow existing trust relationships to be securely transferred from a failed TRE over to a new TRE without any additional burden on the relying parties.

The new communication architecture has been evaluated through automated formal analysis using the Casper-Privacy tool presented in the previous chapter, as well as through comparisons with other proposals in terms of efficiency and practicality. The formal analysis showed that all three protocols meet all the functional, security and privacy requirements defined at the start of this chapter, with respect to the concurrent combination of an external DY adversary and an internal HBC adversary in the role of the DNO, energy supplier or DSM. This systematic analysis confirms that these new protocols do not exhibit any of the security or privacy flaws that were identified by the tool in similar protocols in the previous chapter.

In terms of efficiency and practicality, the comparison with other proposals for achieving the same objectives showed that the new communication architecture is more efficient and practical to implement than various other approaches. The new architecture does not increase the number of messages sent by smart meters nor the number of computationally expensive cryptographic operations performed by smart meters. It also does not require any new types of cryptographic operations to be performed by smart meters and can thus be implemented on real smart meters with minimal modifications to the existing specifications.

Overall, the research in this chapter has shown that the new TRE-based communication architecture provides consumers with the same level of privacy as before the implementation of the smart grid and that due to the nature of this application domain, this does not diminish the functionality of the smart grid. This chapter has therefore confirmed part of the first primary research hypothesis, that the TRE can indeed be used to enhance communication privacy whilst maintaining the primary functionality of the smart metering system. In this chapter is has been assumed that the TRE can be realized and can establish trust relationships with all other participants. Chapter 6 explains how the TRE can be implemented and Chapter 7 presents a mechanism for establishing the trustworthiness of the TRE.

Chapter 6

TRE Design and Implementation

6.1 Establishing the Trustworthiness of the TRE 119
6.2 Adversary Model and Security Requirements 120
6.3 Abstract Reference TRE Architecture 123
6.4 x86-TPM TRE Architecture 126
6.5 Alternative TRE Architectures 139
6.6 Benchmarking and Evaluation 144
6.7 Using the TRE in the Smart Grid 152
6.8 Summary 154

The previous chapters have shown how the TRE can be used to enhance communication privacy in application domains such as the smart grid. This can be achieved due to the first fundamental characteristic of the TRE, which is that the TRE is trusted by all participants, even if the participants are mutually distrustful of one another. Up to this point, similar claims could have been made about a generic trusted third party and thus the pertinent question is: *why should participants trust the TRE?* Unlike a trusted third party, the second fundamental characteristic of the TRE is that it provides strong technical guarantees of its trustworthiness. These guarantees are realized through the design and implementation of the TRE itself as described in this chapter.

This chapter defines the requirements of the TRE and shows how this system can be realized using current technology. This complements the preceding chapter in testing the primary research hypothesis which is that the TRE can be used to enhance communication privacy in the smart grid, and can itself be realized. This chapter begins by describing the overall approach for establishing the trustworthiness of the TRE and then defines the adversary model and security requirements for the TRE itself. An abstract TRE reference architecture that satisfies these requirements is then presented. As an instantiation of this abstract architecture, Section 6.4 describes a concrete TRE architecture based on the x86 platform and using the Trusted Platform Module (TPM). In order to investigate the feasibility, security and efficiency of this design, a fully-functional prototype of this x86-TPM concrete architecture has been implemented. Various alternative architectures are discussed and used in the comparative evaluation and benchmarking of the prototype implementation. Figure 6.1 shows the relationships between the requirements, abstract reference architecture, concrete architectures and prototypes, indicating the extent to which each is discussed in this chapter.

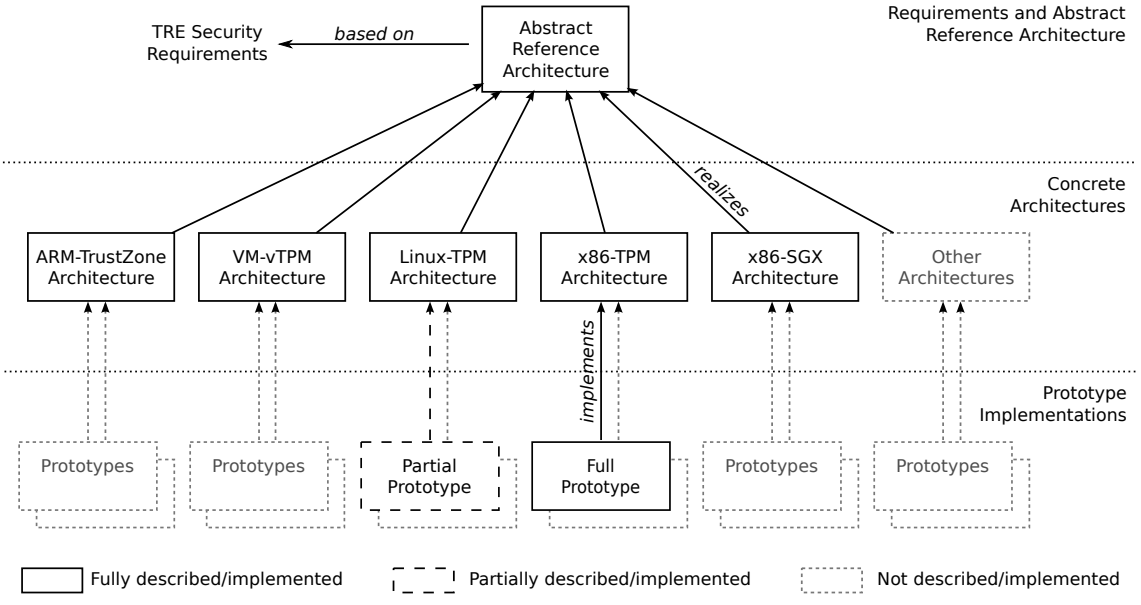

Figure 6.1: Relationships between requirements, architectures and prototypes

6.1 Establishing the Trustworthiness of the TRE

In current communication systems, the trust relationships between participants are based on the identities of the participants. For example, on the Internet, a digital certificate is used to confirm the identity of a web server or to indicate that the server is operated by a specific company. Similarly, client certificates or login credentials are used to distinguish between different clients. This type of *identity-based trust* is well suited for scenarios in which the communication must take place between specific participants. The mechanisms that support identity-based trust (e.g. digital certificates) allow a relying party to distinguish the remote entity from a set of similar entities. Therefore, as explained in Chapter 2, these mechanisms fulfil Proudler's first condition for establishing trust, which is that the entity can be unambiguously identified. However, these mechanisms do not provide any guarantees about the second or third conditions, which are that the entity must be known to operate unhindered and that the relying party must have experience of good behaviour by this entity. At best the identification mechanisms make it possible for a relying party to build up evidence in support of the second and third conditions. However, even extensive evidence does not constitute a strong guarantee of the trustworthiness of the entity because it does not account for the possibility that the entity's behaviour might change or that the entity might be performing actions that are undetectable by the relying party. For example, as explained in the previous two chapters, an HBC adversary is a legitimate participant in the protocol whose undesirable actions are undetectable by the other participants. Therefore, identity-based trust cannot protect against an HBC adversary.

In contrast to identity-based trust, which relies on the ability to distinguish the entity from a set of similar entities, *attestation-based trust* relies on the ability of the relying party to describe the precise nature of the entity. Instead of trusting an entity because of *which one* it is or *who* operates it, the entity is trusted because of *what* it is. For identity-based trust, the relying party only requires sufficient information to uniquely distinguish the entity from other similar entities, whereas for attestation-based trust, the relying party requires significantly more information to be able to fully describe the nature of the entity. Practically, attestation-based trust requires some form of attestation in which all the required information is provided to the relying party, usually by the entity itself.

If the precise nature of the entity can be described by the relying party, all three of Proudler's conditions [217], as described in Section 2.2, are met. For the first condition, the precise nature of the entity can now be identified. This type of identification is particularly appropriate for software because two copies of the same software are indistinguishable from each other but can be identified through a description of their nature (e.g. their source code or functional specifications). As with identity-based trust, entities that include hardware platforms can still be distinguished from each other. For the second condition, a complete and accurate description of the entity will, by definition, reveal whether or not the entity is operating unhindered. For the third condition, prior experience of good behaviour can be substituted by the description of the entity's current and future behaviour. This is more

reliable because the trust decision is based on current and future behaviour rather than on past behaviour. It is also more efficient because it allows relying parties to make trust decisions without having to build up experience of good behaviour by the entity. However, in practice, the effort required to make an informed attestation-based trust decision is infeasibly high for most systems. For example, as explained in Chapter 2, TC remote attestation can provide the relying party with a precise, complete and current description of the software running on a system. However, the effort required to make a trust decision based on this attestation increases with the amount of software on the system thus making attestation-based trust infeasible for most modern software-based systems.

In order to be considered trustworthy, the TRE is designed specifically to make use of attestation-based trust. Unlike trusted third parties, which rely on identity-based trust, the distinguishing feature of the TRE is that it provides relying parties with a precise, complete and current description of *what* it is. The primary objective of the overall TRE design is therefore to facilitate attestation-based trust relationships. This is partially achieved through the attestation subsystem as shown in the abstract reference architecture below, but also depends on various other aspects of the architecture as described throughout this chapter. In order to overcome the challenges of attestation-based trust described above, the primary evaluation criterion for the overall TRE design is the extent to which it minimizes the effort required to establish attestation-based trust relationships.

6.2 Adversary Model and Security Requirements

Since there is no restriction on who may own or operate the TRE, the operator might not be fully trusted by all participants in the communication system. In the worst case, the operator might be viewed as adversarial. This adversarial operator is distinct from the network-based DY and HBC adversaries described in the previous chapters because it has physical access to the TRE. However these adversaries may collude with one another. In order for the TRE to be considered *trustworthy* even if its operator is adversarial, the TRE must meet specific security requirements with respect to this type of adversary. These security requirements enable the mechanisms through which the TRE provides guarantees of its trustworthiness to relying parties. This section presents a precise description of the capabilities of the adversarial operator and defines the security requirements that must be met with respect to this type of adversary.

It is important to note that this chapter focuses on the security of the TRE itself. The security of other components, including the smart meters, is beyond the scope of this research. Various other research efforts have investigated the security of smart meters and proposed mechanisms through which this can be improved, including solutions based on Trusted Computing techniques, as described in Section 3.5.6. One such technique is described in detail in Section 7.6. As explained in the previous chapter, it is assumed that some percentage of smart meters could be compromised and should thus be considered adversarial. This type of threat is taken into account in the design of the enhanced smart

grid architecture presented in the previous chapter. Similarly, the security of the other participants in the protocol (e.g. the energy supplier, DNO and DSM) is also beyond the scope of this research. However, as explained in the previous chapter, by reducing the amount of sensitive personal information held by these participants, the enhanced smart grid architecture limits the impact of a data breach at any of these participants.

6.2.1 Adversary Model

In the worst case, the TRE operator could have similar motives to the DY and HBC adversaries, such as injecting false data into the communication and learning participants' private information. Since the operator has control of all communication channels with the TRE, it has all the capabilities of a DY adversary. The operator also has physical access to the TRE platform, thus giving this adversary the following additional capabilities:

- Load and execute any software on the platform;
- Reset the platform or modify system software (e.g. BIOS and boot loader);
- Add and remove hardware and peripherals;
- Read and write to non-volatile storage media;

The operator, like other adversaries, is assumed to be a rational agent who will only undertake attacks in which the gain (monetary or otherwise) exceeds the cost of mounting the attack. This calculation depends on the nature of the information handled by the TRE, which is in turn defined by the application domain and use case. In general, there are certain types of attacks that are economically infeasible, even for an adversarial operator. For example, although the operator could theoretically mount sophisticated hardware attacks against the TRE, the cost of these attacks is likely to exceed the value of the information obtained. The cost of a hardware attack is dependent on the concrete architecture of the TRE. For example, it is significantly less costly for the operator to read information from a discrete memory module (e.g. dual-ported DRAM) than to read memory inside a System on Chip (SoC). The protection mechanisms provided by the TRE must therefore be commensurate with the nature of the information handled by the TRE. The level of difficulty to mount a successful hardware attack is discussed in more detail alongside the concrete implementation in Section 6.4. As with the DY adversary, it is assumed that the operator is computationally bounded and thus cannot subvert correctly implemented cryptographic primitives in a reasonable time.

The adversary described in this section is arguably the strongest possible adversary that could realistically be encountered in this context. In many situations, the operator might not have all of these capabilities or might not even be adversarial. However, if the TRE achieves an acceptable level of security with respect to this strong adversary, it will achieve at least the same level of security against all other adversaries. The TRE's level of security is defined through a set of security requirements.

6.2.2 Mandatory Security Requirements

For a TRE that is used to enhance communication privacy, the following TRE Security Requirements (TSRs) are always applicable, irrespective of the application domain:

TSR-1: Confidential computation: Although all parties should know precisely what computational operations the TRE performs, the private inputs to these operations must not be disclosed to anyone. For example, if the TRE is used to aggregate private values, the confidentiality of these values must be maintained. The TRE therefore requires a mechanism that prevents external parties from observing or inferring anything about its private inputs.

TSR-2: Integrity-protected computation: To ensure that the TRE performs the expected computation, it requires a mechanism that prevents external parties from interfering with this computation. This corresponds to Proudler's second requirement for establishing trust, which says that the object or entity must be operating unhindered [217]. For example, the TRE must only include the correct input values in an aggregation operation.

TSR-3: Confidential communication: In contrast to other approaches for enhancing communication privacy, the trustworthy nature of the TRE means that participants can send their private information directly to the TRE in uncensored form, without compromising their privacy. In certain use cases, the TRE can also send new confidential information to individual participants (e.g. individualized results of the computation). To achieve this, the TRE must be able to establish confidential communication channels with each participant.

TSR-4: Integrity-protected communication: Similarly, the TRE must be able to establish integrity-protected communication channels with each participant to ensure that the TRE receives the intended, unmodified information and that participants receive the intended results.

TSR-5: Strong attestation: In order to establish trust relationships, the TRE must be able to attest to each participant that it satisfies all the above design requirements as well as any further requirements specific to the application domain. This attestation may make use of a suitable root of trust (RoT) if the RoT is *trusted* by all participants. With the exception of the RoT, this attestation should not rely on any other trust relationships between participants and the TRE, or require the involvement of any other entity. This is an adaptation of Proudler's first requirement for establishing trust, which says that the object or entity must be unambiguously identified [217]. In the case of the TRE, it might not be necessary for the participants to know the specific identity of a TRE as long as they can establish that it satisfies the above requirements. This idea is discussed in detail in the next chapter.

6.2.3 Application-Specific Security Requirements

Depending on the application domain or use case, the following application-specific requirements might also be applicable to the TRE:

TSR-6: TRE identification: For some use cases, the TRE must provide a strong assertion of its identity to relying parties (e.g. to allow participants to use a specific TRE for all communication). This corresponds directly to Proudler's first requirement regarding identification [217].

TSR-7: Client authentication: Certain use cases place strict limitations on the set of entities who may participate in a protocol. For example, only legitimate smart meters may submit consumption measurements or bids. To enforce this, the TRE must be able to authenticate all communicating entities using a suitable mechanism (e.g. pubic key authentication).

TSR-8: Protected storage: For use cases in which the TRE is required to store information (e.g. backups of protocol state information), the confidentiality and integrity of this information must be protected. Even if the TRE uses an external entity to store this data, the information represented by the data must not be disclosed to anyone and any modifications must be detectable.

6.3 Abstract Reference TRE Architecture

This section presents an abstract reference architecture for the TRE that meets the security requirements defined in the previous section. The components of this reference architecture are abstract representations of specific pieces of functionality that must be present in any concrete architecture. Since the removal of any of these components would compromise one or more of the TRE's security or functional requirements, this reference architecture represents the minimum set of required capabilities of the TRE. In any concrete implementation, each abstract component must be implemented as a mechanism or set of mechanisms that provide the equivalent functionality. A graphical representation of the abstract reference architecture is shown in Figure 6.2.

In this architecture, the **protected execution environment** isolates the TRE software from all other software on the platform. It is the primary mechanism through which the architecture protects the confidentiality (TSR-1) and integrity (TSR-2) of the TRE's computation. All code within the protected execution environment is said to be within the TRE's software Trusted Computing Base (TCB). This mechanism therefore also assists with attestation (TSR-5) by excluding other system software from the TCB, thus reducing the effort required to verify the attestation.

The **communications subsystem** is the primary interface through which information is transferred between the protected execution environment and external entities. Depending on the specific architecture, this could be either direct communication with remote

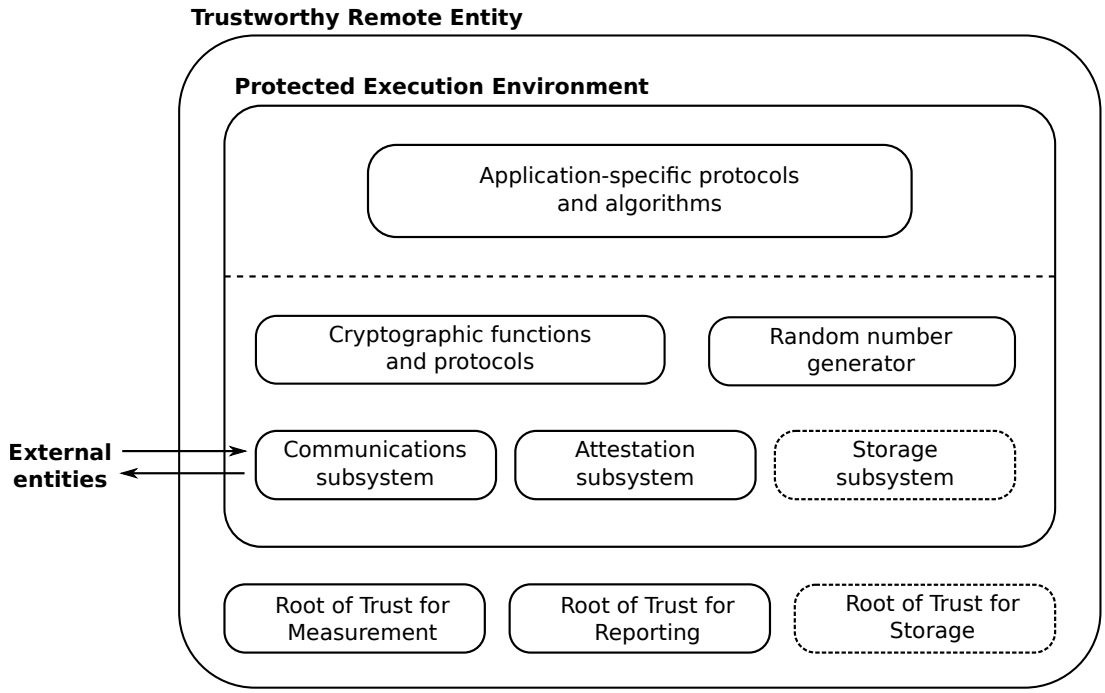

Figure 6.2: Abstract reference architecture of the TRE

entities over a network or communication with untrusted software outside the protected execution environment on the same platform through platform-specific mechanisms.

The **cryptographic library and protocols** must be included within the protected execution environment to enable confidential (TSR-3) and integrity-protected (**TSR-4**) communication between the TRE and remote participants. This component is also used when the TRE is required to identify itself (TSR-6) or authenticate other participants (TSR-7).

The **Random Number Generator (RNG)** is used by the cryptographic library and the attestation subsystem. It is very likely that this functionality would also be used by the application-specific protocols and algorithms. Since the output of the RNG is used as a secret, it must not be possible for external entities or untrusted software on the same platform to learn, infer or manipulate this output. Alternatively, a pseudo-random number generator (PRNG) may be used, provided that its source of entropy cannot be observed or manipulated by untrusted software or entities.

The **attestation subsystem** consists of the components that are used to attest the state of the TRE to relying parties (TSR-5). The overall objective of remote attestation is to convey some type of *measurement* of the TRE's software state to relying parties. This requires a *measurement process* that performs these measurements and an *attestation protocol* that is used to convey these measurements to the relying parties. The nature of the measurements depends on the type of attestation mechanism used in the specific architecture. For example, a binary attestation mechanism, such as TC remote attesta-

tion, provides the relying party with cryptographic hashes of all binaries that have been executed on the platform since the last platform reset. In contrast, a property-based attestation mechanism might provide guarantees of specific properties of the system that allow the relying party to make an informed trust decision. Since the measurement process is closely integrated with the system architecture, it is discussed in this chapter. The remote attestation protocol used by the TRE is the subject of Chapter 7.

A *Root of Trust (RoT)* is an element that serves as the foundation or *trust anchor* for a trust relationship. If the relying party trusts the RoT to operate correctly (whatever its function), this can be used to build additional trust relationships that are more useful to the relying party. The RoT is the element that must be trusted by the relying party in order to obtain any further guarantees about the system. It is therefore critical that the relying parties have some reason to trust the RoT. One possibility for achieving this is to select a RoT that has been certified by a trusted entity. Another possibility is to use standardized technologies that are already in widespread use to maximize the likelihood that the relying parties already trust this type of element. The following three roots of trust use the same nomenclature as the TCG TPM specifications [121]. However, as long as the required functionality is provided, these elements can be implemented using any suitable technologies.

The **Root of Trust for Measurement (RTM)** is the functional element that must be trusted by relying parties in order to obtain accurate measurements of the software state. These measurements are conveyed to the relying parties through the remote attestation protocol.

Similarly, the **Root of Trust for Reporting (RTR)** is the element that must be trusted by relying parties in order to obtain accurate reports of the measurements. Whereas the RTM is trusted to ensure the measurements are an accurate representation of the system, the RTR is trusted to ensure that these measurements can be accurately conveyed to the relying parties. Since the TRE's functionality and software state is not a secret, there is no need for these measurements to be confidential. However, the integrity and authenticity of the measurements must be guaranteed in order to ensure that the attestation is valid.

For applications requiring confidential and integrity-protected non-volatile storage (TSR-8), the optional **Root of Trust for Storage (RTS)** is the element that must be trusted by the TRE to provide these guarantees. This is specifically required to mitigate against the adversarial operator's ability to read and write non-volatile storage media. The RTS is used by the **storage subsystem**, which is responsible for protecting any information that is stored outside the protected execution environment (TSR-8).

In Figure 6.2, these roots of trust are depicted as being external to the protected execution environment. This separation is critical because it allows the roots of trust to be used by the software in the protected execution environment without introducing a cyclic dependency. For example, if the RTM were inside the protected execution environment, it could not be used to give an accurate measurement of the software inside this environment

as this would include a measurement of itself. This separation also allows the roots of trust to be protected from the software running inside the protected execution environment and thus maintain their trustworthiness. An effective mechanism for achieving this separation is to use hardware-based roots of trust, which are also more likely to be trusted by the relying parties.

The **application-specific protocols and algorithms** provide the desired functionality of the TRE for a specific application domain. The privacy-enhancing protocols for the smart grid described in the previous chapter are examples of this type of component. Although there is a conceptual divide between the application-specific components and the application-invariant components of the TRE, there is no technical division between these components. This is in contrast to a traditional OS design in which the kernel is isolated from the user space applications. Since the TRE is a single-function system, this type of distinction between privileged and unprivileged software does not apply.

6.4 x86-TPM TRE Architecture

The x86-TPM[1] architecture is a concrete implementation of the abstract reference architecture defined in the previous section. It is based on the widely-available x86 hardware platform and makes use of the Trusted Platform Module (TPM) to provide the required roots of trust. This section presents an overview of this architecture and its design objectives and then provides specific details about each of the major components. To demonstrate the TRE's functionality, a fully-functional prototype of this architecture has been implemented and is described in parallel with the architecture in the following subsections. The specific implementation decisions made for this prototype (e.g. the choices of software libraries) are not restrictions on the architecture but rather serve as examples of how it can be implemented. Alternative implementations of this architecture using different components are possible but are beyond the scope of this research. The prototype includes implementations of the privacy-enhancing protocols for smart meter communication described in the previous chapter.

6.4.1 Overview of the Architecture

The main characteristics of this architecture are as follows:

- **Bare-metal:** The TRE is implemented as a single software executable that runs directly on the x86 platform without any OS or hypervisor. The TRE software is loaded by the boot loader in place of the OS and includes all necessary hardware drivers.
- **Single-function:** In contrast to an OS, the TRE performs only a single function and does not permit any further software to be loaded until the platform is reset. This is a very similar concept to that of a *unikernel* [176].

[1] In this chapter, architectures are referred to using the nomenclature: *"host platform - root of trust"*.

- **Event-driven:** Since the TRE's primary purpose is to perform privacy-enhancing operations in communication protocols, its functionality is primarily event-driven. The TRE's software architecture is therefore designed as an event-driven system that reacts to network-originating stimuli. To reduce complexity, the TRE uses a simplified execution model in which all operations are executed by a single thread.

- **No internal isolation:** In this architecture there is no internal isolation between components (e.g. no distinction between kernel and user space software). The primary purpose of such isolation would be to limit the impact of a malicious or compromised component on the overall system. However, since all included components are essential for achieving the TRE's functional and security requirements, a compromise of any component would undermine the functionality or trustworthiness of the TRE, irrespective of the use of isolation. This is similar to the design of an *exokernel* [95].

- **Binary attestation:** The TRE provides relying parties with an integrity-protected representation of all software that has been executed in the protected execution environment using the TPM and TC remote attestation. The Dynamic Root of Trust for Measurement (DRTM) is used to limit the scope of this attestation. Sadeghi and Stuble [231] argue that one of the problems with binary attestation is that it could be used to discriminate against certain operating systems or applications (i.e. enabling anti-competitive behaviour). However, their argument only considers the attestation of participants themselves. Since the TRE is a *remote* entity, its use of binary attestation does not impose any limitations on any other participants. It is important to note that binary attestation provides a representation of the actual binary instructions that constitute the TRE software. This is a stronger guarantee than simply attesting the TRE's source code, in which case the relying parties would have to blindly trust the process through which the source code was compiled [252].

- **Implemented in C:** Although this architecture could be implemented in various compiled languages, the most appropriate choice appears to be C as this provides a sufficient level of control without incurring the additional development effort required to implement the TRE in an assembly language. This also makes it possible to use a wide variety of externally-implemented software components, since C is the most common language for writing system-level software.

- **Open source software:** In order to use binary attestation to establish the trustworthiness of the TRE, the full source code of the TRE must be available to all relying parties. The prototype has therefore been implemented as open source software under the BSD software licence.

- **Current hardware:** This TRE architecture has been fully implemented using only currently available hardware. The current generation TPM 1.2 provides sufficient

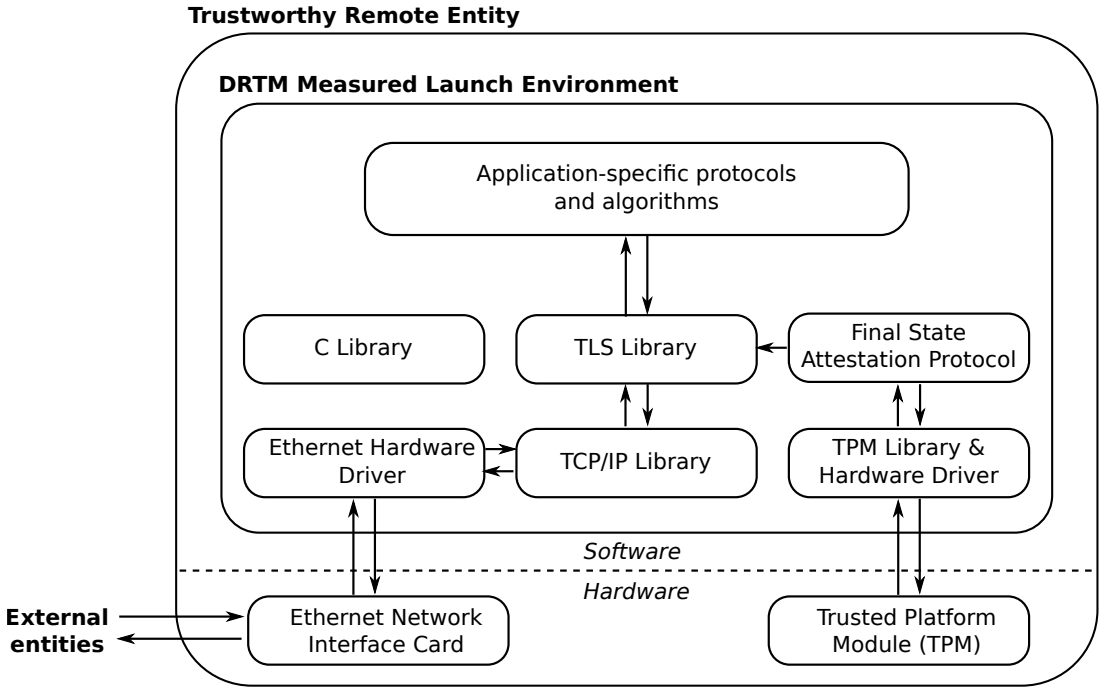

Figure 6.3: Overview of the x86-TPM TRE architecture

functionality for this architecture. However, various benefits could be derived from using the upcoming TPM 2.0 when available (e.g. greater choice of cryptographic algorithms and increased key lengths).

An overview of this x86-TPM TRE architecture is shown in Figure 6.3. With the exception of the attestation subsystem, which is the subject of Chapter 7, the design objectives and the specific components of this architecture are described in detail in the following subsections.

6.4.2 Design Objectives

Since the primary objective of the overall TRE design is to facilitate attestation-based trust relationships, the specific design objective for this concrete architecture is to provide sufficient functionality to implement the application-specific protocols (e.g. those presented in the previous chapter) whilst meeting all the security requirements and minimizing the TRE's software TCB. In order to establish the trustworthiness of the TRE, this architecture uses TC remote attestation to provide relying parties with an integrity-protected representation of the TRE's software TCB. The relying parties can then evaluate this TCB, or have it evaluated by another trusted entity, in order to make a decision about the trustworthiness of the TRE. As explained in Chapter 2, the effort required by the relying party to make a trust decision increases as the size of the software TCB increases. Depending on the mechanisms used to make the trust decision, the increase in required effort might be superlinear with respect to the TCB size. This would usually be the case

if automated formal analysis were used. Therefore, minimizing the software TCB serves to minimize the effort required to make a trust decision about the system. Furthermore, it has been shown that an increase in the lines of source code will probably result in an increase in the density of software defects [187], and thus the number of defects also increases superlinearly with respect to the size of the TCB. Some of these defects could be vulnerabilities that compromise the TRE's security requirements. Therefore, in general, minimizing the TCB serves to reduce the number and density of software defects, which in turn increases the trustworthiness of the system.

Theoretically, TC remote attestation can be used on any system that contains a TPM, including client PCs. However, in practice the effort required by the relying party to make a trust decision about all the software in the TCB of a client PC makes this infeasible. There have been various efforts to reduce the software TCB of systems in order to improve security and/or increase the feasibility of remote attestation: Lyle [172] showed that this type of remote attestation can be made feasible on systems such as web servers that have a smaller, more stable software TCB. Lyle and Martin [173] also proposed a *split service architecture* for web services that minimizes the TCB of the trusted components by partitioning the software on the platform into an untrusted front-end and a trustworthy back-end. Toegl et al. [253] use the DRTM late-launch to remove pre-OS components such as the BIOS and bootloader from the TCB. One of the first examples of a hardware-backed TEE on the PC platform was the *Flicker* research project [181], which demonstrated many of the advantages of minimizing the TCB, in terms of both security and remote attestation. TrustVisor [182] continued this idea and significantly improved performance using virtualization. At a fundamental level, security technologies such as ARM TrustZone and Intel SGX enhance security by reducing the TCB of the components that must be trusted. Building on these previous efforts, the x86-TPM TRE architecture aims to minimize the TRE's software TCB whilst meeting all the security requirements and providing sufficient functionality to run the application-specific protocols. Since the primary evaluation criterion for the overall TRE design is the extent to which it minimizes the effort required to establish attestation-based trust relationships, this specific architecture must be evaluated in terms of the size and composition of its TCB.

As part of the evaluation, the computational performance of the prototype implementation is also analysed. However, computational performance is only a secondary objective of the architecture since it sometimes conflicts with the primary objective. For example, although a multi-threaded software architecture would increase computational performance, the current prototype only uses a single execution thread in order to reduce complexity and minimize TCB size. Provided that the computational performance of the system is sufficient to meet the security requirements and achieve the functional objectives of the TRE, the optimization of this computational performance is beyond the scope of this research. In a production version of the TRE, certain trade-offs could be made to enhance performance levels.

6.4.3 Protected Execution Environment

Almost all previous approaches for reducing the size of the TCB have relied on isolating the TCB from other software on the platform. For example, Flicker [181], TrustVisor [182], ARM TrustZone and Intel SGX all isolate specific pieces of software from an untrusted commodity OS running on the same platform. Since these approaches have been designed to improve the security of existing systems (e.g. client PCs, servers etc.), this untrusted software is required to fulfil the functional requirements of the systems. In contrast, as a single-function remote entity, the TRE does not have to support any untrusted software on the platform. Instead of isolating the TCB as in other systems, the TRE can minimize the TCB by removing all untrusted software from the platform. The TRE's software TCB can therefore coincide with the platform boundary, such that the only software on the platform is part of the software TCB. Since the protected execution environment defines the boundary of the software TCB, in this architecture the protected execution environment can be the platform boundary.

The protected execution environment is established through the Dynamic Root of Trust for Measurement (DRTM) late-launch, which performs a partial reset of the platform in order to exclude any software that had previously been run on the platform. This means that the BIOS, the boot loader and all other software components necessary to boot the platform into the desired state are excluded from the TRE's software TCB. The late-launch hardware instruction resets PCRs 17-22 of the TPM and uses these to record the measurements of all software subsequently executed on the platform. If the TRE software is modified before launch, this will be detectable through these measurements. In the prototype, the late-launch is performed by Intel's *tboot* framework[2]. The prototype complies with the multiboot specification[3] and makes use of the E820 memory map provided by tboot to configure the available platform memory.

Once the TRE software is running, the protected execution environment is maintained by the platform's boundary since the TRE is the only software running on the platform and no further software will be loaded. Unlike other approaches, this means that the software in the protected execution environment is always active and always has full control of the platform. In the prototype, Protected Memory Regions (PMRs) are used to protect the TRE's internal data from malicious hardware peripherals. For example, Direct Memory Access (DMA) improves the performance of certain peripherals by allowing them to read and write directly to main memory without involving the CPU. Without the PMRs, a malicious DMA-capable peripheral could extract confidential data from main memory, thus violating TSR-1. Furthermore, these peripherals might be invisible to relying parties, so the possibility of a DMA attack would undermine the attestation guarantees provided by the TRE. To overcome this challenge, the PMRs designate specific areas of memory that are protected from DMA access. In the prototype, the PMRs are set up to cover

[2] `http://sourceforge.net/projects/tboot/`

[3] `http://www.gnu.org/software/grub/manual/multiboot/multiboot.html`

the majority of main memory, except for a small unprotected area that is used for interfacing to memory-mapped hardware peripherals. The full memory layout is explained in Section 6.4.8.

This architecture is still vulnerable to sophisticated hardware attacks in which the operator is able to read and/or modify data on the bus between the CPU and main memory or read and/or modify the data stored in main memory without using DMA. However, the cost of mounting these attacks is assumed to be higher than the value of the information obtained, making these type of attacks economically infeasible for the adversary as explained in Section 6.2.1. These type of attacks can be prevented using hardware-based memory encryption functionality, such as that provided by Intel's SGX technology [183, 133].

6.4.4 Communications Subsystem

In this architecture, the TRE communicates with external parties via a hardware network interface. The TCB therefore includes a driver for the Network Interface Card (NIC) as well as a network stack (e.g. a TCP/IP stack). In this prototype, communication takes place via a single Ethernet interface using a generic E1000 Ethernet NIC driver. This driver is relatively simple and is also compatible with many common NICs. Since this driver uses memory-mapped IO to interface with the NIC, the transmit and receive buffers are allocated outside of the PMRs so that these can be accessed by the NIC through DMA. This does not affect security since any confidential data will be encrypted before being passed to the NIC driver. The Lightweight IP[4] (lwip) library is used as the TCP/IP stack. This library was primarily designed for embedded systems and therefore has a relatively small TCB. In the prototype, it is responsible for setting up and managing TCP connections and performing auxiliary operations such as sending and responding to Address Resolution Protocol (ARP) messages. The lwip library is available under the BSD licence. The TCP/IP stack periodically polls the NIC driver for incoming packets. Although interrupts could be used, this would increase the complexity of the system and the size of the TCB. When a packet is received, it is passed to the cryptographic library and processed in an event-driven manner.

6.4.5 Cryptographic Library

The cryptographic library enables the TRE to communicate securely with remote parties over untrusted channels and thus satisfies the requirements for confidential (TSR-3) and integrity-protected (TSR-4) communication. When required by the use case, this component also allows the TRE to identify itself (TSR-6) and authenticate remote parties (TSR-7). This library must include the required cryptographic primitives (e.g. encryption, decryption, signing, hashing etc.) as well as the protocols in which they are used to

[4]Website: `http://savannah.nongnu.org/projects/lwip/`
Source code: `http://git.savannah.gnu.org/cgit/lwip/lwip-contrib.git`

provide secure communication channels (e.g. TLS). The specific cryptographic primitives and protocols used will depend on the application domain or use case. For example, in the enhanced smart grid communication architecture presented in the previous chapter, all communication between participants and the TRE takes place over mutually authenticated TLS connections which would be facilitated by the cryptographic library.

The prototype uses the *mbed TLS*[5] (formally PolarSSL) cryptographic library. The fact that this library is primarily intended for use in embedded systems also makes it well-suited for use in the TRE because it has a relatively small code size. It is also highly modular, thus allowing unused functionality to be disabled, and it does not require an OS since it is completely self-contained. This library supports current cryptographic protocols such as TLS 1.2 and all modern cipher suites. To minimize the prototype's TCB, all cipher suites are disabled except TLS 1.2 using a single cipher suite, namely TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256. This is a modern cipher suite that offers relatively good performance, shorter key lengths due to ECC, and perfect forward secrecy (PFS) because of the ephemeral Diffie-Hellman (DH) key exchange. The cryptographic primitives used in this cipher suite are mandatory for all smart meters in the UK [265, 266], thus suggesting that this will be one of the supported cipher suites. Only one named curve, the NIST-recommended prime256v1 (secp256r1) curve, is enabled. Although this library is primarily licensed under the GPL (version 2 or later), it also has a Free and Open Source Software (FOSS) licence exception that allows it to be licensed under the terms of the BSD licence[6].

In some application domains, it is possible, as an alternative, to use specialized cryptographic protocols or a purpose-built cryptographic library in order to further reduce the size of the TCB or improve performance. However, a custom protocol or library is likely to be significantly less trustworthy, or at least require significantly more effort to verify than a standardized, widely-used component. The advantage of using standardized protocols, such as TLS, and widely-used libraries, such as mbed TLS, is that these will have benefited from a much higher degree of scrutiny in terms of their security properties. Since mbed TLS is used in many systems, the cost of any security analysis of this library would be amortized over its large installed base.

6.4.6 Storage Subsystem

The storage subsystem in this architecture can be used to create secure backups of the TRE's state. The TRE can maintain two types of state information: *soft state* is any information that can be easily re-established from the communicating participants, whereas *hard state* cannot be easily re-established. For example, soft state includes a TRE's TLS session keys or participants' pseudonyms in the demand bidding protocol presented in the previous chapter. If a TRE were to fail, participants could simply connect to a different TRE and request new pseudonyms. Since it can be re-established, soft state does not need

[5]Website: `https://tls.mbed.org/Sourcecode:https://github.com/ARMmbed/mbedtls`
[6]`https://tls.mbed.org/foss-license-exception`

to be backed up, thus reducing the attack surface of the TRE. Most of the TRE's state is soft state. However, some protocols may require a limited amount of hard state that cannot be easily recovered and thus must be backed up. For example, in the smart grid billing protocol presented in the previous chapter, the consumers' running totals are hard state. Since this type of state must be maintained even if the TRE fails, the TRE provides two types of secure backups: an offline backup for the same platform or an online backup to a different TRE. In either case, these backups can only be used by a TRE that is in exactly the same state as the original TRE when the backup was created, thus preserving the trust relationships. Both types of backups are initiated by network commands. Although the design of both types of backups is presented, the storage subsystem is not implemented as part of the prototype since this is not a mandatory requirement.

Offline backups are designed to be stored outside the TRE and used to restore a TRE to a particular state. These backups can only be restored to the same platform on which they were created, since that platform's TPM is used as the root of trust for storage (RTS). The storage subsystem first generates a new AES key k_A and encrypts the target data with this key $E(k_A, \text{data})$ using AES Galois Counter Mode (AES-GCM). Since AES-GCM is an authenticated encryption mode, it ensures both the confidentiality and authenticity of the encrypted data and thus does not require an additional integrity-protection mechanism. The storage subsystem then uses the TPM to seal the AES key k_A using the TPM's Storage Root Key (SRK) K_{SRK} and the current value of the dynamic PCRs, which represent the current software state of the TRE. In this sealing operation, the TPM also includes the current values of the dynamic PCRs at the time the operation took place (i.e. `PCR_digest_at_creation`). The resulting encrypted blob $E(k_{SRK+}, k_A, \text{PCRs})$ can only be decrypted by the TPM on which it was created and will only be decrypted if the dynamic PCR values are the same as when k_A was sealed. Finally, the encrypted data $E(k_A, \text{data})$ and the encrypted AES key $E(k_{SRK+}, k_A)$ are sent as a response to the backup request. Since all keys and data are encrypted, this backup can be stored by any entity without compromising the trust relationships. To restore this backup, the process is reversed: the encrypted keys and data are sent to the TRE, the TPM unseals the AES key (if the dynamic PCRs are in the correct state) and the storage subsystem uses this to decrypt and restore the data. Since data can be sealed to any PCR state, and since the PCR state of the TRE is a well-known value, an adversary might try to inject false data into the system through this backup mechanism. To prevent this, the TRE will only restore an offline backup if the values of the dynamic PCRs included in the backup (i.e. the `PCR_digest_at_creation`) are the same as their current values. In comparison with using the TPM to seal the data directly, it is significantly more efficient to perform bulk data encryption using AES and to seal the AES key under the SRK (although this might change with the TPM 2.0). It is not sufficient to bind the AES key to the TPM since the bind operation cannot be used to enforce PCR state. This type of backup cannot be transferred to another TRE because a TPM may not seal data using a migratable key. Backups of this type would therefore be created periodically and used in case the TRE

loses power or has to be reset for any other reason, but does not fail completely.

In order to backup data to another TRE, both TREs must be online at the same time in order to perform the online backup operation. To create an online backup, the primary TRE establishes a secure connection to the secondary TRE and both perform the remote attestation protocol with each other (i.e. the attestation protocol is run once in each direction). If the values of the respective dynamic PCRs are the same on both TREs, the primary TRE sends the data to be backed up to the secondary TRE over the secure connection. This mutual attestation ensures that each TRE knows the current state of the other. This is required in order to migrate the trust relationships from one TRE to another and also to ensure that a malicious party cannot introduce false data into the system through this backup mechanism. This type of online backup would be performed periodically so that the secondary TRE could be used as an immediate replacement if the primary TRE were to fail and could not be recovered using an offline backup.

Even though these backup mechanisms use strong technical measures to protect the confidentiality and integrity of the information, they still result in a slight increase of the TRE's attack surface. However, it is anticipated that these mechanisms would only be required for hard state information, and thus used infrequently. Hard state is usually the result of long-running temporal aggregation, but temporal aggregation itself is a privacy-enhancing algorithm and thus the aggregated information becomes less sensitive over time. Therefore, even if the backup mechanism were somehow to be compromised, the adversary would only learn this aggregated hard state information. For example, obtaining a consumer's total bill half way through the billing period is unlikely to be considered a major privacy violation. Therefore, these backup mechanisms do not significantly affect the trustworthiness of the TRE.

6.4.7 Application-Specific Protocols

This architecture can support a wide variety of application-specific protocols that make use of the subsystems and components described above. Chapter 8 presents a broader discussion of the possible use cases for a TRE and the respective application-specific protocols. To demonstrate and evaluate the functionality of this architecture, the privacy-enhancing smart meter communication protocols presented in the previous chapter have been implemented as part of this prototype. In these protocols, all communication with the TRE takes place over mutually authenticated TLS connections, which are established using the communication subsystem and the cryptographic library. The protocols themselves have been implemented using the DLMS-COSEM specifications as described in the previous chapter. Following the event-driven architecture of the TRE, these protocols have been implemented to react to messages arriving over the network interface. Depending on the nature of the received message, the protocols might update their internal state, respond to the message, and/or open communication channels and send information to other participants.

```
void* memchr(const void *s, int c, size_t n);
int memcmp(const void*, const void*, size_t);
void* memcpy(void*, const void*, size_t);
void* memmove(void*, const void*, size_t);
void* memset(void*, int, size_t);
size_t strlen(const char*);
unsigned long strtoul(const char *nptr, char **endptr, int base);
long strtol(const char *nptr, char **endptr, int base);
int strcmp(const char *s1, const char *s2);
int strncmp(const char *s1, const char *s2, size_t n);
int strcasecmp(const char *s1, const char *s2);
char* strcpy(char* to, const char* from);
char* strncpy(char* dst, const char* src, size_t n);
char* strchr(const char *p, int ch);
int vscnprintf(char *buf, size_t size, const char *fmt, va_list a);
int snprintf(char *buf, size_t size, const char *fmt, ...);
int sscanf(const char *str, const char *fmt, ...);
```

Listing 6.1: C library functions included in the TRE prototype

6.4.8 Other Components

Random Number Generator

The RNG functionality is provided by the Counter mode Deterministic Random Byte Generator (CTR-DRBG) included in the mbed TLS library. The CTR-DRBG has been standardized by NIST and is described in NIST Special Publication 800-90A [28]. The entropy for this byte generator is obtained from the TPM's hardware RNG when required. However, this slightly decreases the performance of the system when the DRBG needs to be reseeded because the TPM is not a high-performance component. Using the default mbed TLS reseeding interval, the DRBG requires a new random seed from the TPM after approximately 3,000 ECC signature operations.

C Library Functions

In order to support the subsystems and components described above, various C library functions are included in the TCB. These functions provide memory control and string processing capabilities. The list of included C library functions is shown in Listing 6.1. The implementation of these functions is derived from the FreeBSD system[7] and is available under the BSD licence. These functions are all implemented in C except for the `memcpy` and `memset` functions, which are implemented in assembly language to improve performance.

Memory Management

Since there is no isolation between components, this architecture uses a single flat memory model with identity paging applied to all physical memory (i.e. each virtual memory address corresponds to the same physical memory address). This eliminates the overhead

[7] `https://www.freebsd.org/`

Table 6.1: TRE memory layout for a platform with 4 GB physical memory

Start	End	Size	Usage
0x00000000	0x000FFFFF	1 MB	Unused (BIOS and legacy peripherals)
0x00100000	0x002105CA	∼ 1 MB	TRE executable software
0x002105CB	0x003FFFFF	∼ 2 MB	TRE Stack
0x00400000	0xEFFFFFFF	∼ 3.75 GB	TRE Heap
–	–	–	Memory protection boundary
0xF0000000	0xFFFFFFFF	256 MB	Memory-mapped IO (unprotected region)

and complexity of context switching, thus reducing the size of the TCB. It also simplifies the process of porting the implementation to platforms that do not have sophisticated memory management capabilities.

Table 6.1 shows the memory layout used in the prototype implementation. Memory below 1 MB is not used by the TRE as it contains BIOS memory and various legacy peripheral interfaces (e.g. VGA buffers). Similarly to an OS, the TRE software is loaded into memory starting at 1 MB. The compiled TRE executable is approximately 1 MB in size. The bottom of the TRE's stack is placed at 4 MB and the stack grows downwards towards the lower memory addresses as is the convention on x86 platforms. The required stack size was determined experimentally. During testing, the stack size of the prototype implementation did not exceed 512 kB. Static analysis of the TRE prototype using the checkstack[8] utility showed that no individual function uses more than 4 kB of stack space. The allocated stack size of approximately 2 MB is therefore sufficient and includes a significant safety margin. As an event-driven system, this architecture makes use of dynamic memory allocation (i.e. heap allocation). The memory management functionality is provided by the Two Level Segmented Fit (TLSF) memory allocator [179]. Since this allocator is designed for real-time applications, allocations and deallocations occur in constant time (**O(1)**). A public domain implementation of the TLSF allocator[9] was used in the TRE prototype. As shown in Table 6.1, the memory region above 4 MB up to the memory protection boundary, is designated as the TRE's heap space and is therefore managed by this allocator. The memory protection boundary (`0xF0000000` in Table 6.1) delineates the end of the PMR (i.e. the DMA-protected memory region) that was set up during the late-launch. In the prototype, this boundary is set at 256 MB before the end of usable memory. All memory above this boundary is used for communicating with memory-mapped IO devices such as the NIC. Although this region is not protected against malicious DMA-capable peripherals, it does not contain any data that would not be available to the adversary through other means (e.g. by eavesdropping on the network interface). The TRE uses the Memory Type Range Registers (MTRRs) to configure the platform such that all protected memory (i.e. memory below the protection boundary)

[8] `http://lxr.free-electrons.com/source/scripts/checkstack.pl`
[9] `http://tlsf.baisoku.org/`

uses write-back caching to maximize computational performance, whilst all unprotected memory (i.e. above the protection boundary) is marked as uncachable to facilitate communication with memory-mapped IO devices.

6.4.9 TRE Configuration

To minimize the software TCB and simplify the remote attestation mechanism, this architecture removes all dynamic configuration of hardware peripherals from the TRE. In general, the x86 platform allows the running OS to dynamically detect and configure the available hardware peripherals. For example, an OS would usually scan the platform's PCI bus to detect the bus, slot and function numbers of the NIC. Similarly, the OS would query the peripherals for additional configuration information (e.g. the OS would query the NIC to determine its MAC address).

If this type of dynamic configuration capability were used on the TRE, it would allow the TRE software to run unmodified on multiple platforms, but would also increase the size of the software TCB. Furthermore, the results of this dynamic configuration would not be automatically reflected in the measurement of the TRE, since they are only determined after the TRE has commenced execution. Including this configuration information in the measurements is important as it allows relying parties to detect changes to the TRE's hardware platform, which could influence their trust relationships. The results of the dynamic configuration could be explicitly extended into the PCRs by the TRE, but this would also add unnecessary overhead to the TCB.

Instead of using dynamic configuration, the x86-TPM TRE architecture makes use of static preconfiguration. When the TRE software begins executing, it expects to find a specific configuration structure, which has been loaded into memory, containing all necessary hardware configuration information. This configuration structure is shown in Listing 6.2. It includes the PCI bus, slot and function numbers of the NIC as well as the NIC's MAC address. This structure is also used to supply the TRE with other configuration information, such as it's IPv4 address, and to store encrypted cryptographic keys, such as the Attestation Identity key (AIK), which is loaded into the TPM.

In order to generate this configuration structure, the `tre-config` Linux utility has been developed. This application is designed to run on a standard Linux installation on the same platform on which the TRE will be run. In the envisioned deployment scenario, the target platform will be able to dual-boot between the TRE and this Linux installation which is used for configuration and maintenance. The `tre-config` application generates the configuration structure by scanning the hardware configuration of the system and requesting input from the operator where necessary (e.g. the IPv4 address of the TRE). The configuration structure is then saved as a serialized binary file and made accessible to the boot loader. When the platform is booted as a TRE, the boot loader loads this binary file into memory as it would a kernel module. The tboot framework also measures all loaded modules and extends their measurements into a separate PCR from the main TRE

```
typedef struct __attribute__((packed)) tre_config
{
    //Hardware configuration
    uint8_t     e1000_pci_bus;
    uint8_t     e1000_pci_slot;
    uint8_t     e1000_pci_func;
    uint8_t     e1000_mac_address[6];

    //IPv4 configuration
    uint32_t    ipv4_address;
    uint32_t    ipv4_gateway;
    uint32_t    ipv4_netmask;

    //TPM-sealed AIK blob
    uint32_t    aik_blob_len;
    uint8_t     aik_blob[4096];

} tre_config_t;
```

Listing 6.2: TRE configuration structure

software. As explained in the next section, it is important to use a separate PCR for the configuration information so that the measurement of the TRE software will be the same across different platforms even though the hardware configuration of the platforms might be different. The OS that runs the `tre-config` application does not need to be trusted in any way. Since the configuration structure does not contain any unencrypted sensitive information, the only possible attack would be to compromise the availability of the TRE through misconfiguration of the platform hardware. However, availability is not a security requirement of the TRE because this can be trivially compromised by the adversarial operator who has physical access to the platform. Therefore, this preconfiguration approach reduces the TCB by eliminating the need for dynamic hardware configuration and ensures that all configuration information is automatically measured and extended into the PCRs before the TRE begins executing.

6.4.10 Measurement and Attestation

In order to use TPM remote attestation, all software that runs in the protected execution environment must be *measured* before it commences execution. This section describes the measurement process responsible for performing these measurements. Since the TPM 1.2 is used as the root of trust for reporting (RTR), a measurement is performed by taking a SHA1 cryptographic hash of the executable binary, recording this in the Secure Measurement Log (SML) and extending it into one of the TPM's PCRs. The measurement process for the x86-TPM TRE architecture always includes four such measurements. After the DRTM late-launch, the first piece of software that runs is the vendor-supplied Authenticated Code Module (ACM). As part of the late-launch, the ACM is measured by the DRTM firmware and extended into PCR 17 [132]. The DRTM firmware is the root of

trust for measurement (RTM) since it performs the first measurement in the chain. After performing the relevant initialization operations, the ACM measures the tboot software and extends it into PCR 18 [132]. The TRE software is then measured by tboot and also extended into PCR 18. Finally, all configuration structures are measured by tboot and extended into PCR 19. Since no further software is loaded until the platform is reset, no further measurements are required. In this way, the TRE configuration structure and any application-specific configuration structures are measured and included in the attestation. However, since these configuration structures are not security-sensitive, the value of PCR 19 can simply be compared to previous values to detect if the TRE's configuration has changed. Since the measurements of the configuration structures are extended into a separate PCR from the TRE software, the relying parties can base their trust decisions entirely on PCR 18. Furthermore, by excluding the configuration measurements from this PCR, the value of PCR 18 remains constant across all TREs (although in practice, a small number of variants might arise due to different versions of the tboot or TRE software). This allows trust decisions about the TRE to be delegated to external service providers if the relying parties do not have the required capabilities to make these decisions. For example, a regulatory body could publish a list of trustworthy TRE signatures (i.e. values of PCR 18), which could be independently verified by any other entity (e.g. an independent security auditor). Relying parties can therefore make the trust decisions themselves or delegate these to other trusted entities. The attestation protocol, which is responsible for communicating the measurements to relying parties, is the subject of Chapter 7.

If the platform uses a TPM 1.2 that is implemented as a discrete integrated circuit (IC), it may be possible for an adversary with physical access to the platform to subvert the measurement process by performing a hardware attack against the TPM. Kauer [149] and Sparks [242] have both described and implemented a *reset attack* against discrete TPM ICs. In this attack, the *LRESET#* pin of the TPM or the reset line of the low pin count (LPC) bus is temporarily connected to the platform ground voltage level. This causes the TPM to be reset without the platform being reset, which allows malicious software on the platform to avoid detection by extending falsified values into the TPM's PCRs. However, the recent TPM 2.0 specification allows the TPM to be implemented as part of the platform's firmware, thus making it very difficult, if not impossible, to perform this type of reset attack.

6.5 Alternative TRE Architectures

This section introduces various alternative architectures that could be used to implement the TRE. All of these are concrete instantiations of the abstract reference architecture presented in Section 6.3 and thus fulfil the security requirements defined in Section 6.2. These architectures differ from each other and from the x86-TPM TRE architecture presented in the previous section in terms of the platforms on which they are implemented and the roots of trust they use. This section introduces the alternative architectures and then the

relevant technical details are presented as part of the comparative evaluation in the next section. Although specific technical details are discussed, full prototype implementations of the alternative architectures are beyond the scope of this research.

6.5.1 Linux-TPM

The simplest architecture from a design and implementation perspective is to implement the TRE as an application running on a commodity OS such as Linux. In this architecture, most of the subsystems are provided directly by the OS. The communication subsystem and cryptographic library are both provided by the OS (e.g. the Linux networking stack and the OpenSSL cryptographic library). The OS includes all necessary hardware drivers as well as memory management functionality and a C library. The protected execution environment is provided by the operating system's process isolation mechanism which isolates applications from one another. However, this means that the OS itself is included in the TRE's software TCB, even though most of its functionality is not necessary for the TRE. The TPM is used as the root of trust and the system follows the same measured boot process as the x86-TPM TRE architecture, including the DRTM late-launch. Since binary attestation is used, all software in the TCB must be measured and extended into the TPM using a tool such as the Linux Integrity Measurement Architecture (IMA) [232]. Although various techniques have been proposed to improve OS attestation [94], this still remains an open challenge. If the relying party is capable of making a trust decision about the entire TCB, this architecture fulfils all the security requirements.

6.5.2 VM-vTPM

The main disadvantage of the Linux-TPM TRE architecture is that the commodity OS is included in the TCB. The VM-vTPM architecture is designed to avoid this by using virtualization to create and enforce the TRE's protected execution environment, whilst allowing other software to run outside the protected environment on the same platform. This architecture runs the TRE in some type of virtualized environment, such as a Virtual Machine (VM), and uses a virtualized TPM (vTPM) [37] as the root of trust. The main advantage of being able to run other software on the same platform is that any functionality that is not security critical can be delegated to this untrusted software, thus minimizing the TCB. For example, tasks such as setting up communication links with remote participants or interfacing with platform hardware can often be delegated to the untrusted software. The trusted software can then use a simplified protocol to communicate with the untrusted software rather than communicating directly with remote participants.

This type of approach has been proposed for other systems in recent literature: Lyle and Martin [173] have described how virtualization can be used to divide a web service into an untrusted front-end and a trustworthy back-end. In their split-service architecture, the untrusted front-end runs the web service middleware that communicates with other participants using SOAP and XML. This middleware communicates with software in the

trusted environment using Java RMI, which is significantly simpler than SOAP or XML, thus reducing the TCB of the trusted environment. TPM-based binary attestation can thus be used more effectively by attesting only the software in the trusted environment.

McCune et al. [182] have presented *TrustVisor*, a special-purpose hypervisor that can provide protected execution environments for sensitive code and data whilst minimizing the amount of code that the hypervisor itself adds to the TCB. TrustVisor is designed to protect legacy applications running on a commodity OS by isolating a specific Piece of Application Logic (PAL) from other PALs and from the untrusted OS. It can thus provide multiple protected environments and supports isolation at a very fine level of granularity by virtualizing the platform's physical memory, enforcing memory isolation between the different PALs and the untrusted OS, and protecting PALs against malicious DMA reads and writes. For each PAL, TrustVisor creates an instance of a software-based *micro* TPM (μTPM) that provides a reduced set of TPM functionality. Since the μTPM runs on the main CPU, it is significantly faster than current hardware TPMs and due to its reduced functionality, it adds less code to the TCB than a standard vTPM. TrustVisor itself is measured during the DRTM late-launch and extended into the hardware TPM. Before a PAL is executed, TrustVisor measures the PAL and extends it into the respective μTPM, for which TrustVisor serves as the root of trust. TPM-based binary attestation of individual PALs is achieved by providing the relying party with both a measurement of the TrustVisor software, backed by the hardware TPM, and a measurement of the specific PAL from the μTPM, backed by TrustVisor. There are various other examples of the use of virtualization to enhance security, including SecVisor [237] and the Qubes[10] OS, but these do not focus on minimizing the TCB or facilitating remote attestation.

TrustVisor is well-suited for implementing the VM-vTPM TRE architecture. Following the overall design philosophy of Lyle and Martin's split-service architecture [173], the security-critical parts of the TRE are run in a PAL, whilst all other functionality, such as communication with remote participants, is provided by the commodity OS that runs outside the TRE's protected execution environment. Similarly to the x86-TPM TRE architecture, the TRE's PAL still includes the application-specific protocols and the cryptographic library, since these are necessary for secure end-to-end communication with remote participants. The PAL also includes the required memory management functionality, C library functions, and a driver for the μTPM. The communication subsystem, including the network interface driver, is provided by the untrusted OS but the PAL still contains a minimized communication subsystem for communicating with the untrusted OS (e.g. marshalling data across the TrustVisor separation boundary). The measurement and attestation of the TRE software is all handled by TrustVisor.

[10] `https://www.qubes-os.org/`

6.5.3 ARM-TrustZone

ARM TrustZone technology is a set of CPU extensions and supporting components available in modern ARM Cortex-A series cores. TrustZone provides a Trusted Execution Environment (TEE), called the *secure world*, that can be used to run security-sensitive software in isolation from other software running in the *normal world* on the same platform[11]. When the platform is started, TrustZone guarantees that the secure world software will be executed first with full control of the platform. After the secure world initialization, control can be transferred to the normal world. The normal world can communicate with the secure world (and vice versa) via TrustZone's *Secure Monitor*. If the platform includes a secure element (e.g. a tamper-resistant storage co-processor), this can be used as a root of trust for storage. However, TrustZone does not provide roots of trust for measurement (RTM) or reporting (RTR). This means that although the secure world is isolated from the normal world, there is no way to measure the software running in the secure world or attest it to a remote relying party.

Since the TRE requires a RTM in order to fulfil the attestation requirement (TSR-5), the TRE software therefore cannot be run in the secure world. In the ARM-TrustZone TRE architecture, the TRE is run in the normal world and the secure world is used to run a software TPM, such as that described by Aaraj et al. [1], which provides the required RTM and RTR. The TRE's protected execution environment is therefore the boundary of the normal world and thus no other software can be run on the platform (unless some type of virtualization approach is used, as described above). The software that runs in the normal world is almost identical to that described in the x86-TPM architecture. The normal world contains the cryptographic library and application-specific protocols, as well as the communication and remote attestation subsystems. It also includes memory management functionality, C library functions and the relevant hardware drivers. Additionally, all software in the secure world, including the software TPM, also becomes part of the TRE's software TCB. Given the lack of a hardware RTM and RTR, the platform manufacturer must provide a strong guarantee that a correctly-implemented software TPM will be run in the secure world. This is comparable to a TPM manufacturer making an assertion that a particular component is a genuine TPM. However, since the TPM is a hardware device, it is arguably less vulnerable to compromise than a software TPM, particularly given the recent reports of other TrustZone software being compromised[12]. Overall, this architecture is very similar to the x86-TPM TRE architecture, except that it uses a software TPM running in the TrustZone secure world as its RTM and RTR. Provided that the relying parties consider these roots of trust to be trustworthy, this architecture meets all the TRE security requirements.

A variation of this architecture is to use a heterogeneous computing platform, such as the Xilinx Zync-7000 System on Chip (SoC), which combines a field-programmable gate

[11] `http://www.arm.com/products/processors/technologies/trustzone/`
[12] `http://bits-please.blogspot.co.uk/2015/03/getting-arbitrary-code-execution-in.html`

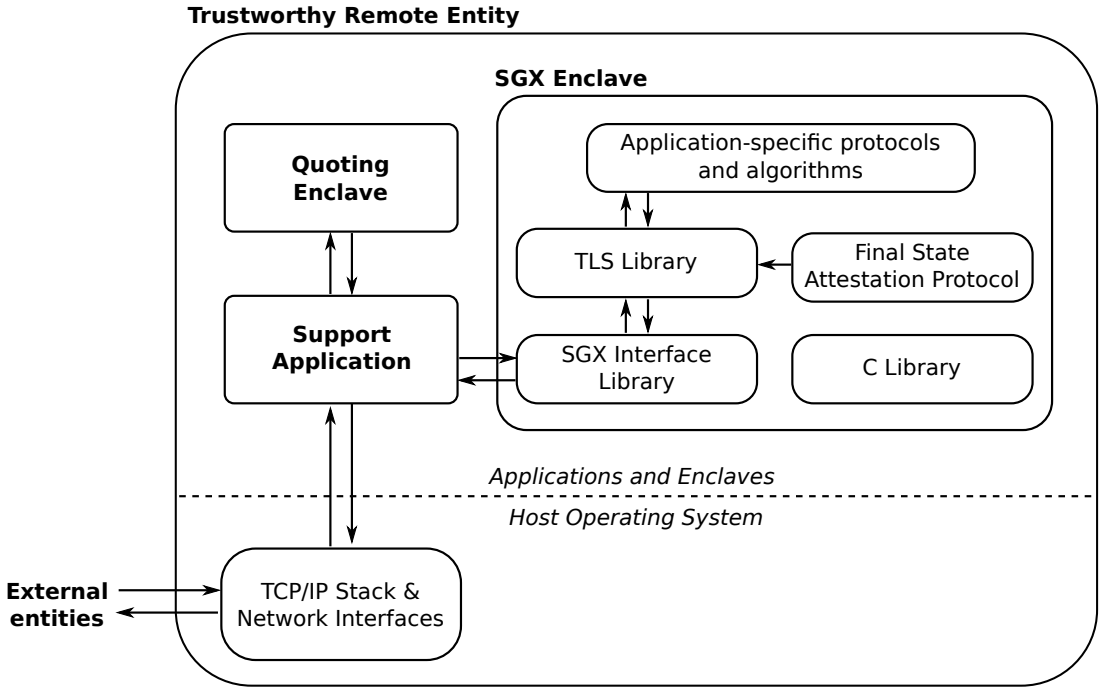

Figure 6.4: Overview of the x86-SGX TRE architecture

array (FPGA) with an ARM core. The TRE would be implemented on the FPGA, either as a direct FPGA design or by using the FPGA to emulate another type of processor. The TrustZone functionality of the ARM core would again be used to run a software TPM and provide the required RTM and RTR. Alternatively, the FPGA could be configured to provide TPM functionality [115].

6.5.4 x86-SGX

The recently-announced Software Guard Extensions (SGX) technology from Intel is designed to protect applications or parts of applications against an untrusted OS [183, 133]. These CPU extensions provide very similar functionality to TrustVisor but achieve even higher levels of assurance by doing this using hardware-based mechanisms. As explained in Chapter 2, security-sensitive software is run in an SGX *enclave*, which is executed in user mode (i.e. ring 3) and communicates with the untrusted OS using shared memory. Unlike TrustVisor, sensitive data from an SGX enclave never leaves the CPU unencrypted, and is thus protected against even an adversary who has physical access to the platform and can read the platform's memory. Unlike TrustZone, SGX provides a root of trust for measurement that can be used to measure and attest the software inside an enclave.

An overview of the x86-SGX TRE architecture is shown in Figure 6.4. In this architecture, the TRE is implemented in a single SGX enclave. The private information processed by the TRE is only available within this enclave and all components that do not require access to the private information are excluded from the enclave. For example, the commu-

nication subsystem (i.e. the TCP/IP stack and network interface driver) is provided by the untrusted OS. Although it would be possible to use multiple enclaves to provide isolation between the TRE's components, this would not increase the trustworthiness of the TRE. Since each enclave would require access to the private information, a compromise of any enclave would therefore still undermine the trustworthiness of the TRE, irrespective of the use of isolation. Furthermore, the use of multiple enclaves requires inter-enclave communication, which both increases the size of the software TCB and presents a new potential attack vector (e.g. the inter-enclave communication protocols could be manipulated by untrusted software on the platform).

In a similar manner to the VM-vTPM architecture, the SGX enclave includes the cryptographic library, application protocols, memory management and C library functionality. Again, a minimized communication subsystem is required within the enclave to communicate with the untrusted software on the platform. Since the SGX enclave cannot make system calls, it cannot communicate directly with the untrusted OS. Instead it requires an additional supporting application, as shown in Figure 6.4. However, since this supporting application is outside of the enclave, it does not increase the size of the TCB. The untrusted OS includes the relevant hardware drivers and handles the communication with remote participants. SGX provides a mechanism for measuring and attesting the contents of an enclave using the CPU as the root of trust (i.e. no TPM is required). Similarly to the TPM, the CPU itself is endorsed by the manufacturer as being a genuine SGX component. Since the SGX functionality is provided by the CPU, it does not increase the TRE's software TCB. SGX therefore meets all the security requirements for the TRE. Furthermore, since SGX encrypts any enclave data that leaves the CPU, it is even secure against a stronger adversary who is capable of reading data from the system bus or from main memory.

In the same paradigm, a precursor to SGX was the Flicker system by McCune et al. [181], which uses a DRTM late-launch to protect specific pieces of application logic (PALs) from an untrusted OS. Similarly to the x86-SGX architecture, the TRE could be implemented as a Flicker PAL (i.e. an x86-Flicker architecture). In this case, the PAL would also include a TPM driver and an attestation subsystem since these are not provided by Flicker. However, Flicker's use of the TPM imposes significant performance limitations when switching into and out of the protected environment. It is widely assumed that SGX will not suffer from similar limitations.

6.6 Benchmarking and Evaluation

The preceding sections have shown how the x86-TPM TRE architecture as well as the alternative TRE architectures described in Section 6.5 can all meet the security requirements defined at the start of this chapter. As explained in Section 6.1, the primary evaluation criterion for the overall TRE design is the extent to which it minimizes the effort required to establish attestation-based trust relationships. Since all of these architectures

use binary attestation, they all aim to minimize the TCB in order to minimize the effort required to establish trust relationships through this binary attestation mechanism. The TRE architectures could also be evaluated in terms of their computational performance. However, provided that it is sufficient to fulfil the TRE's functional requirements, the computational performance of the TRE is a secondary consideration because it does not affect the TRE's trustworthiness, which is the main requirement for enhancing communication privacy. The following subsections evaluate the x86-TPM TRE architecture and the accompanying prototype implementation through a comparative analysis of the size and composition of the TCB and, secondarily, its computational performance. The prototype implementation is compared against the alternative architectures described in the previous section. Although these alternative architectures have not been implemented in full, partial prototypes or representative samples from similar systems have been used for this evaluation.

6.6.1 TCB Size

x86-TPM TRE Prototype Implementation

To ensure a fair evaluation of the TCB size, the Linux Kernel coding style[13] was applied to all code in the prototype with the exception of the lwip and mbed TLS libraries which have their own coding styles. The SLOCCount[14] tool by David A. Wheeler was used to count the number of lines of code in the prototype's TCB. Since this tool only counts physical lines of code, the prototype's code was first passed through the C preprocessor to expand all preprocessor directives (e.g. `#include` and `#ifndef` statements). However, this expansion results in header files being counted multiple times if they are included in more than one file. To overcome this, a unity build[15] of each subsystem was generated (i.e. all code from the subsystem was merged into a single `.c` file) before being counted using SLOCCount. The code sizes of the overall software TCB and its various subsystems are shown in Table 6.2 and Figure 6.5.

In the prototype implementation, all non-essential features and options in the various subsystems have been turned off. However, no code optimization has been performed on the component libraries themselves. These measurements are therefore a true reflection of the number of lines of code required to implement this functionality using standard unmodified components. Although the TLS library (mbed TLS) has been configured to include only the cryptographic functions required for a single cipher suite, it still makes up the majority (58.3%) of the TCB. In comparison, the same mbed TLS library with all cryptographic functions and cipher suites enabled is more than 45,000 lines of code (as counted by the SLOCCount tool, excluding tests and examples). The widely used OpenSSL library contains over 400,000 lines of code[16]. The second largest contributor to

[13] `https://www.kernel.org/doc/Documentation/CodingStyle`
[14] `http://www.dwheeler.com/sloccount/`
[15] `http://buffered.io/posts/the-magic-of-unity-builds/`
[16] `https://www.openhub.net/p/openssl/`

Table 6.2: Lines of code in the TCB of the x86-TPM TRE prototype

Subsystem	**Component (Language)**	**Lines of code**
Cryptographic functions and protocols	mbed TLS library (C)	14,408
Communications	lwip TCP/IP stack (C)	5,135
	Ethernet hardware driver (C)	834
Memory management	TLSF allocator, MTRRs, PMRs (C)	1,035
Roots of Trust	TPM library hardware driver (C)	1,005
C Library	C library functions (C)	794
	C library functions (assembly)	60
Core	TRE core functionality (C)	685
	Early initialization (assembly)	35
Application-specific protocols	Smart grid privacy protocols (C)	507
Attestation	Final state attestation protocol (C)	221
Total		**24,719**

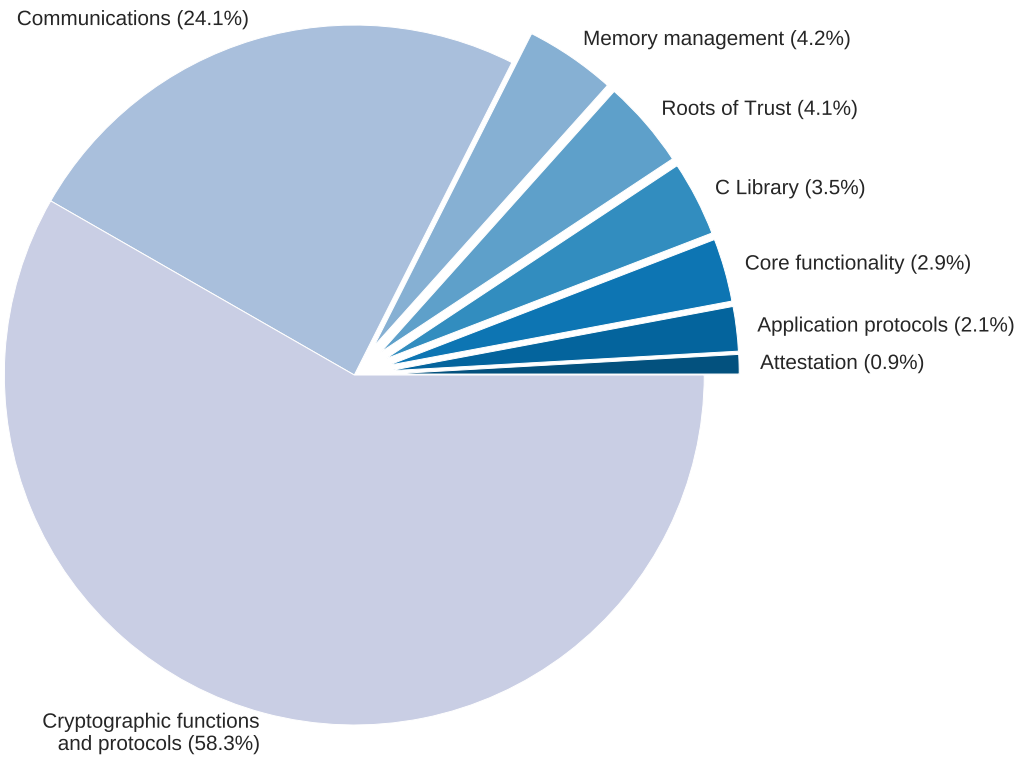

Figure 6.5: Relative sizes of TRE subsystem by lines of code

the TCB is the communication subsystem (24.1%), which consists of the lwip TCP/IP stack and the Ethernet hardware driver. Again, all unnecessary options have been disabled since the full lwip library is more than 87,00 lines of code[17]. The TPM hardware driver and library, which provide the various roots of trust, are 4.1% of the TCB. The supporting subsystems providing memory management and C library functionality contribute 4.2% and 3.5% respectively. The software that interconnects the various subsystems and components is referred to as the TRE core functionality and makes up 2.9% of the TCB. The fully-functional smart grid protocols, as described in the previous chapter, contribute 2.1% to the TCB and the attestation subsystem is 0.9% because it makes use of functionality already provided by the cryptographic library as explained in the next chapter.

Immediately after the DRTM late-launch, part of the tboot framework is executed in order to perform platform configuration and to measure the TRE software. Since this software is running in the protected execution environment, it must be counted as part of the TCB. Although tboot was used in this prototype, any suitable framework could have been used to perform the DRTM late-launch. For example, the component of the Flicker system [181] that executes after the late-launch is less than 250 lines of code. This could be used to launch the TRE and would require even fewer than 250 lines of code because some of the Flicker functionality (e.g. the TPM driver) is already included in the TRE. Therefore, even with this additional code, the TCB would be less than 25,000 lines of code.

Overall, at less than 25,000 lines of code, the TRE is three orders of magnitude smaller than the Linux kernel, which contains over 18 million lines of code[18]. The TRE is smaller than a *Mirage* unikernel, which is over 100,000 lines of code [176]. The TRE is still larger than the *seL4* microkernel (8,700 lines of code), which has been formally verified [151]. However, advances in verification techniques and computational capabilities could make full formal verification of the TRE a possibility in the near future. For example, a specific version of the TLS library, which constitutes 58% of the prototype implementation's TCB, has already been formally analysed and shown to be immune from certain common vulnerabilities [263]. Furthermore, a TCB of this size is also amenable to security audits and the TRE would be a particularly good candidate for auditing because many of its subsystems are well-tested libraries that are widely used in other systems, thus reducing the effort required to audit and allowing the costs to be amortized over a large number of users. The TLS library used in the TRE prototype has been selected over the widely-used OpenSSL library as the cryptographic core of the *OpenVPN-NL* software tool, which is a hardened version of Open-VPN that has been evaluated and accredited by the Dutch government[19]. Code reviews of the TLS library's source code were performed as part of this accreditation.

[17] `https://www.openhub.net/p/lwip`
[18] `https://www.openhub.net/p/Linux`
[19] `https://openvpn.fox-it.com/`

Linux-TPM

In the Linux-TPM TRE architecture, most of the functionality would be provided by the underlying OS. The only subsystems that would be required in the TRE application are the application-specific protocols and the attestation subsystem. However, since the OS kernel is included in the TCB, the size of the TCB would increase by at least 3 orders of magnitude. Furthermore, every update to the OS would cause the TCB to change, thus further increasing the effort required to make informed trust decisions about this type of system.

VM-vTPM

In the VM-vTPM TRE architecture implemented using TrustVisor, the TCP/IP stack (5135 lines) and the Ethernet hardware driver (834 lines) are provided by the untrusted OS and can therefore be replaced by a significantly smaller component in the TRE (although still non-zero in size). The attestation capability (221 lines) and the post-late-launch code (∼250 lines) are provided by TrustVisor. However, TrustVisor itself is 6,481 lines of code [182], which is also included in the TCB, thus resulting in a slightly larger TCB size than the x86-TPM architecture. Since TrustVisor has been specially designed to minimize the TCB, the use of any other virtualization technology in this architecture would result in a larger increase in TCB size.

ARM-TrustZone

In the ARM-TrustZone TRE architecture, there is no possibility to run untrusted code on the platform since the software TPM is run in the secure world and the protected execution environment in the normal world. Therefore, all subsystems listed in Table 6.2 will be part of the TCB, although the size of certain components (e.g. the Ethernet hardware driver) may change due to the differences between the x86 and ARM platforms. However, in this architecture, the software TPM and all other software running in the secure world will also be part of the software TCB. Therefore the software TCB in this architecture will always be larger than that of the x86-TPM architecture. Furthermore, since there is no root of trust for measurement for this secure world software, it must be blindly trusted by relying parties, thus making this architecture less trustworthy than the x86-TPM implementation.

x86-SGX

In the x86-SGX architecture, the components of the communication subsystem (i.e. the TCP/IP stack and Ethernet hardware driver) are provided by the untrusted OS. Since SGX is used as the root of trust, the TPM driver and attestation subsystems can also be removed from the TCB. These would be replaced by an SGX driver, which is expected to be significantly less complex and require fewer lines of code. Since all SGX functionality is provided by hardware, it does not contribute to the TCB. Therefore, when this architecture

becomes available, it is likely to result in a TCB that is at least 30% smaller than that of the x86-TPM architecture and provide even stronger security guarantees due to its use of enclave memory encryption.

6.6.2 Computational Performance

Although computational performance is only a secondary evaluation criterion, two benchmark tests were performed on the x86-TRE prototype to give a representative indication of its computational performance. The first benchmark measured the time taken to create and verify signatures since these are the most computationally expensive cryptographic operations performed during a TLS handshake. The second benchmark measured the overall time required to perform a DLMS-COSEM operation consisting of a single request and response, since this type of operation is used for all communication in the privacy-enhancing smart grid protocols.

Signature and Verification Benchmark

This benchmark consisted of creating an ECDSA signature of a 20 byte value using a 256 bit ECC key and the *prime256v1* curve, and then verifying this signature. This combination of signature creation and verification was performed between 50 to 1000 times and the total time for each run was measured using the CPU tick counter through the `RDTSC` instruction. Each run was repeated 100 times in order to obtain the average time and standard deviation.

The average times for the signature and verification benchmark are shown in Figure 6.6. In this figure, the performance of the x86-TPM TRE prototype is compared to that of a partial prototype of the Linux-TPM architecture. Both systems use the same cryptographic library (mbed TLS) but the Linux-TPM prototype uses the memory management functionality (i.e. `malloc` and `free`) and C library functions (e.g. `memcpy` and `memset`) provided by the host OS. The Linux-TPM prototype was run on two different platforms with different CPUs, one of which was also used to run the TRE. In all cases, the software was compiled using the GCC compiler[20]. For each system, two performance curves are shown representing different compiler optimization levels. At the `-O2` optimization level, the compiler performs nearly all supported optimizations that do not involve a space-speed trade-off. At the maximum (`-O3`) optimization level, the compiler performs all available optimizations that can be performed whilst still producing standards-compliant output. The standard deviations are too small to be visible in Figure 6.6, thus indicating that these performance levels can be consistently obtained.

On the same platform, the TRE is approximately 8% slower than the Linux implementation. This is primarily due to the fact that the Linux implementation's memory manipulation functions (i.e. `memcpy` and `memset`) have been optimized for the specific platform. For example, these functions make use of the SSE2 or later CPU extensions to

[20] `https://gcc.gnu.org/`

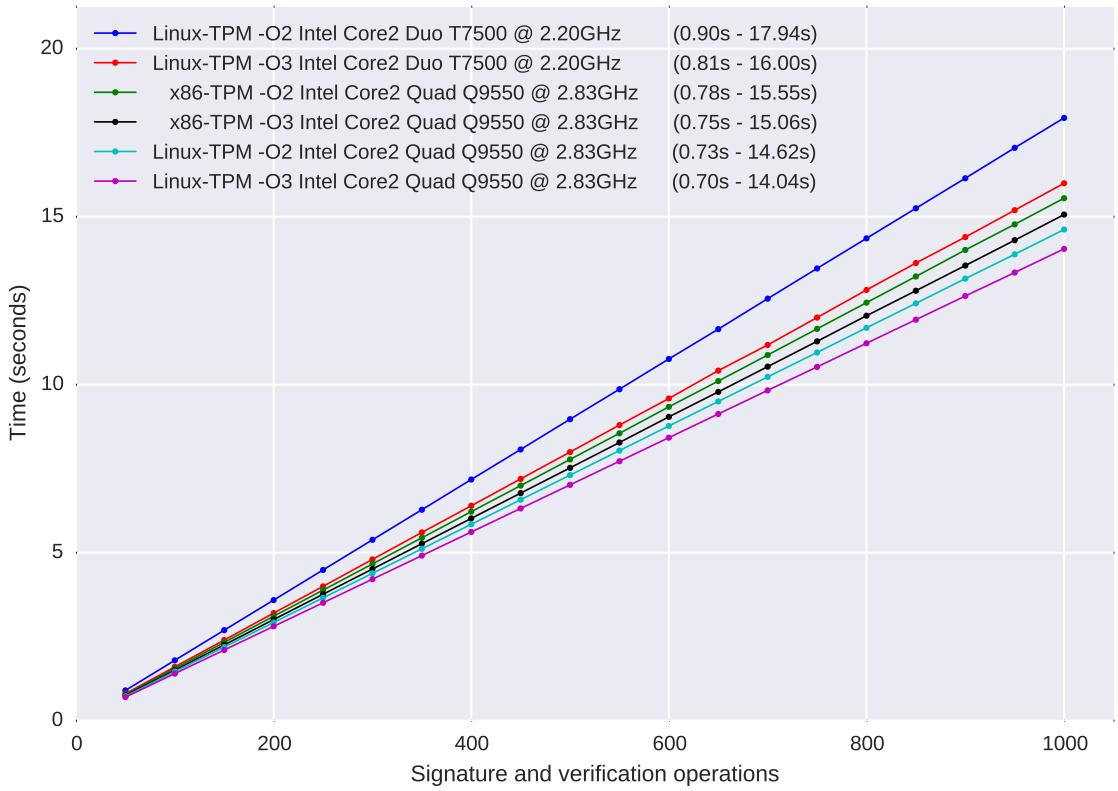

Figure 6.6: Computational performance of signature and verification operations

speed up memory operations. These functions are extensively used by the cryptographic library. When the Linux implementation is restricted to using the same `memcpy` and `memset` functions as the TRE, its performance curves are virtually identical to those of the TRE. It would be possible to use optimized versions of these functions in the TRE prototype, but this would add approximately 2,300 lines of assembly code to the TCB. This same trade-off between performance and TCB size is also applicable to the VM-vTPM and the x86-SGX TRE architectures.

DLMS-COSEM Benchmark

For this benchmark, the DLMS-COSEM `GET` operation was selected since this is used by the TRE to request the most recent consumption measurement from consumers. Examples of the actual request and response messages are shown in Table 5.2 in the previous chapter. This benchmark measured the time taken for the TRE to send a specified number of requests and receive all the responses. Each request-response pair required the TRE to set up a new TLS connection with a different participant. However, the remote attestation protocol was not included in these benchmarks since this is the subject of the next chapter. For this experiment, the TRE was connected via a local 100 MB/s Ethernet network to a server that emulates multiple smart meters. The average latency between the TRE and this server was 0.4 ms. The same server and network were used in all benchmarks.

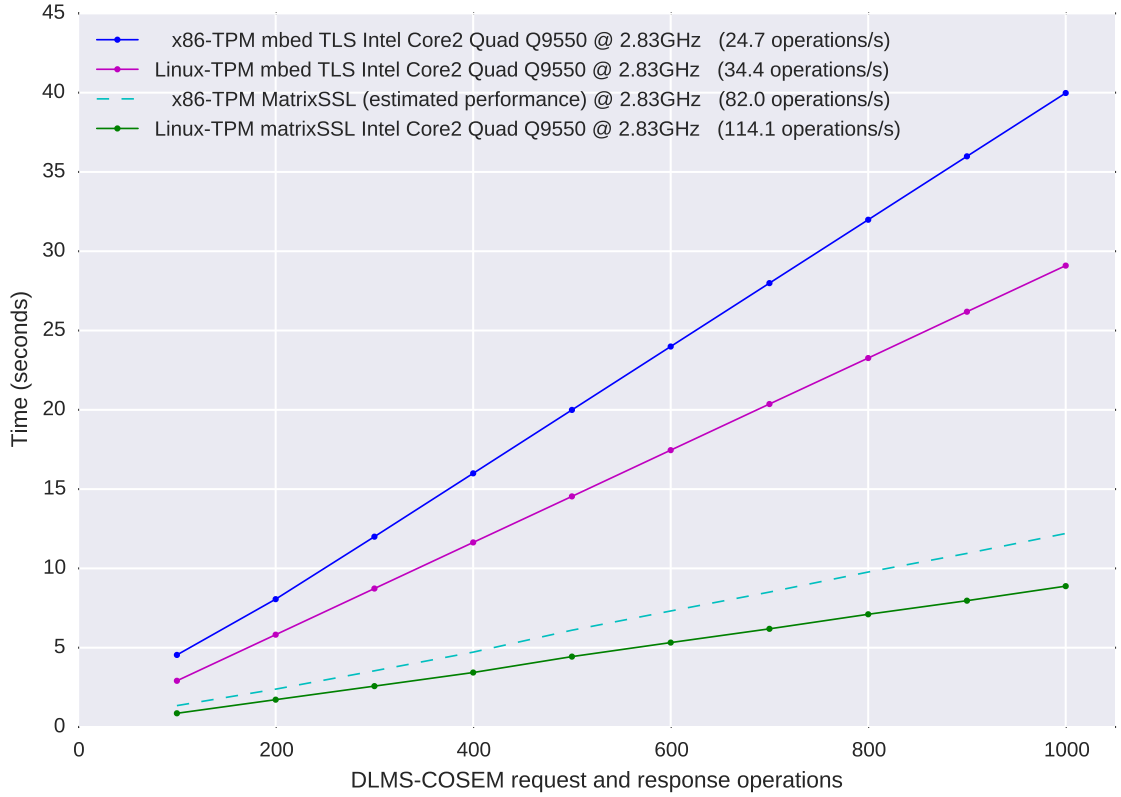

Figure 6.7: Overall performance of DLMS-COSEM operations

Although the response time of this server was necessarily included in the measured times, the TRE's computational performance remained the limiting factor in all benchmarks.

In this benchmark, the TRE sends the requests continuously and processes the responses as they arrive. When using the DLMS/COSEM standard, the smart meters are the servers and the TRE is the client. This means that the TRE may decide when to contact each smart meter. This benchmark is therefore an accurate representation of the TRE's real-world performance in this type of protocol. If an alternative architecture were used in which the smart meters initiate the connections, this would likely decrease the performance of the TRE because the timing of the requests would be suboptimal from the TRE's perspective. In this case, the benchmark would also need to account for the timing of the incoming requests (e.g. by modelling these using a Poisson distribution) and the effects of having a queue of requests (e.g. response latency and average queue length).

The average times taken to perform the specified number of DLMS-COSEM operations are shown in Figure 6.7. In this figure, each measurement is the average of 10 independent experiments. Again, the standard deviations are too small to represent in this figure. The maximum standard deviation was 0.57 seconds for 1000 operations in the x86-TPM TRE prototype. All systems were compiled with full (`-O3`) compiler optimizations. As explained in Section 6.4, the x86-TPM TRE prototype uses the mbed TLS cryptographic library. For comparison purposes, Figure 6.7 also shows the performance of two partial

prototypes of the Linux-TPM architecture, one using the same mbed TLS library and the other using the matrixSSL[21] library. When using the same cryptographic library, the x86-TPM prototype is approximately 37% slower than the Linux-TPM prototype, including the 8% difference in the signature and verification operations shown in Figure 6.6. The remainder of this performance difference is due to the optimized functionality provided by the host OS, including parallel processing (each operation runs in its own process), an optimized network interface driver, faster memory management (`malloc` and `free`) and faster C library functions (`memcpy` and `memset`). When the MatrixSSL library is used in place of the mbed TLS library in the Linux-TPM partial prototype, each operation takes less than a third of the original time. This is simply due to differences in efficiency between these two libraries for this particular set of cryptographic operations. Using the ratio between the two Linux-TPM curves for each value on the horizontal axis, the estimated performance of the x86-TPM prototype, if it were using matrixSSL, is shown as the dashed line in Figure 6.7. However, using the matrixSSL library in the x86-TPM prototype would add approximately 9,000 lines of code to the TCB size in Table 6.2. Again, a trade-off between performance and TCB size can be made in each architecture.

Although the other alternative architectures have not been prototyped, it is possible to speculate about their performance based on these results. The VM-vTPM architecture is likely to provide similar computational performance to the x86-TPM architecture because it will still be limited by the performance of the cryptographic library and the C library functions. Although it can take advantage of the optimized Ethernet driver and TCP/IP stack provided by the untrusted OS, this advantage will likely be lost due to the additional time taken to communicate across the VM boundary between the trusted and untrusted environments. The ARM-TrustZone TRE architecture will also be very similar to the x86-TPM architecture since it uses the same software and cryptographic library. The x86-SGX architecture is likely to provide better computational performance than the x86-TPM architecture. Like the VM-vTPM architecture, it can make use of the optimized Ethernet driver and TCP/IP stack provided by the host OS, but in this case, the overhead added by the hardware-supported mechanisms for transitioning between the secure enclave and the untrusted OS is likely to be minimal. However, this architecture will probably still be slower than the Linux-TPM architecture since the x86-SGX architecture cannot use the optimized memory manipulation functions from the host OS without increasing the size of the TCB.

6.7 Using the TRE in the Smart Grid

This section describes how the design and implementation of the TRE presented in this chapter can be used in the enhanced smart grid communication architecture given in Chapter 5. In particular, it explains how the relevant trust relationships are established, and analyses the TRE's performance in the context of the smart grid.

[21] `http://www.matrixssl.org/`

6.7.1 Trust Relationships in the Smart Grid

As explained in the preceding chapter, consumers as well as service providers must establish trust relationships with the TRE. Consumers must trust that the TRE will not reveal their private information, whilst service providers must trust the TRE to authenticate consumers and perform the privacy-enhancing operations correctly. As explained in this chapter, these trust relationships are based on binary attestation of the TRE to all relying parties. Specifically, the relying parties base their trust decisions on the PCR values provided by the TRE. In theory, this system allows each consumer to make an individual assessment of the trustworthiness of the TRE based on this attestation. However, in practice, the majority of consumers do not have the capability to make such a decision. These consumers would delegate this decision to a trusted party. For example, the national energy regulator or national data protection authority could publish a list of trusted PCR values for the TRE. Consumers could input these values into their smart meters (either through a local communication interface or via a web service) to ensure that the smart meter may only communicate with an approved TRE. Although some consumers still rely on a trusted party, the significant advantage compared with the previous situation the attestation provided by the TRE allows *delegable trust decisions*. This means that the system does not specify *which* trusted party must be used. If certain consumers do not trust the energy regulator, they could just as easily delegate this trust decision to a different entity, such as a private security company or a university. This is possible because any entity with the relevant capabilities can evaluate the PCR values and provide a list of trusted values. Furthermore, consumers can delegate this trust decision to multiple entities, similarly to how medical test results can be shown to multiple doctors. This ensures that consumers can delegate these trust decisions to one or more *trustworthy* entities. In addition to the initial setup phase, this type of delegable trust decision can also be used if the software on the TRE is updated (i.e. causing the PCR values to change). However, as explained in this chapter, minimizing the TRE's software TCB reduces the likelihood of software defects and thus frequency of software updates.

This research primarily focusses on approaches and mechanisms through which relying parties can establish the trustworthiness of the TRE. In many application domains, a related challenge is for the TRE to establish the trustworthiness of the other participants. For example, the enhanced smart grid architecture could benefit from the TRE being able to ascertain whether individual smart meters have been compromised. However, this is beyond the scope of this research because it is an orthogonal challenge. For example, in a smart grid architecture *without* a TRE, it would be equally beneficial for the service providers to establish the trustworthiness of the smart meters. If there exists a mechanism for verifying the trustworthiness of smart meters, this same mechanism can be used either with or without the TRE. For example, if all smart meters included trusted hardware and had the capability to perform remote attestation, this attestation could be verified by either the service providers or by the TRE. The use of a TRE therefore does not have

any effect on this challenge. However, this is not necessarily the case in other proposals for enhancing privacy in the smart grid. For example, the proposals based on new cryptographic techniques such as homomorphic encryption or secret sharing, as described in Chapter 3, would likely preclude the use of smart meter attestation, and thus introduce potential security vulnerabilities.

TRE Performance in the Smart Grid

In absolute terms, both the prototypes evaluated in the preceding section meet the performance requirements of the smart grid. Assuming all three smart grid protocols proposed in the previous chapter are in use, each consumer can have at most two individual request-response interactions with the TRE in any time period:

1. The TRE requests the consumer's latest consumption measurement and requests a bid for the next time period. The request includes the price information for the next period. The consumer responds with the consumption measurement and a bid.

2. The TRE notifies the consumer of the DSM's decision about a particular bid.

When the TRE communicates with the service providers, the data from individual consumers is either aggregated (e.g. in the monitoring and billing protocols) or can be combined into a single operation (e.g. the set of all consumer bids and the set of all decisions). Therefore this communication requires only two DLMS-COSEM operations per aggregation group per service provider and is thus negligible compared to the number of operations between the TRE and consumers. Since the slowest x86-TPM TRE prototype can perform at least 40,000 DLMS-COSEM operations in each 30 minute time period, it can therefore support at least 20,000 individual consumers. If necessary, this capacity can be increased by using a faster cryptographic library, optimizing the software implementation, and/or using higher performance hardware. In order to minimize costs, the real-world communication links between smart meters and the TRE will have significantly lower bandwidths and higher latencies, and the computational performance of real smart meters will be significantly lower than in this experimental setup. Overall, these benchmarks show that the computational performance of the TRE will not be the limiting factor in the system and is therefore sufficient to fulfil the functional requirements of the smart grid context.

6.8 Summary

This chapter has described how the TRE itself can be designed, implemented and evaluated. Although most communication systems rely on identity-based trust relationships, these do not provide strong guarantees of trustworthiness. In particular, identity-based trust cannot defend against an HBC adversary. In contrast, the TRE enables relying parties to form significantly stronger attestation-based trust relationships due to the design of the TRE's software architecture.

Since the TRE can be operated by any entity, it must be secure against a potentially adversarial operator who has all the capabilities of a DY adversary as well as physical access to the platform. The TRE must therefore fulfil the mandatory security requirements of confidential and integrity-protected computation and communication as well as providing strong attestation guarantees. In some use cases, the TRE might also be required to identify itself or authenticate remote participants as well as provide secure storage capabilities for backup purposes. All of these requirements are met by the abstract TRE reference architecture, which defines the minimum set of required capabilities of the TRE.

A concrete instantiation of this abstract architecture has been designed and implemented on the x86 platform using the TPM as its root of trust. This x86-TPM TRE architecture is a single-function piece of software, similar to a unikernel, that runs directly on the hardware. It has been designed as an event-based system to match the event-driven nature of the TRE's functionality. This architecture uses the TPM to perform binary attestation of its TCB and therefore its specific design objective is to minimize the size of the TCB. Unlike other systems that reduce the TCB by isolating it from untrusted software, the fact that the TRE is an independent *remote* entity allows it to minimize the TCB by removing all untrusted software from the platform. Following the same principle as an exokernel, there is no isolation between different components of this architecture since all components are critical to the security and functionality of the TRE. All communication with the TRE takes place over TLS using the cipher suite that is likely to be mandated for all smart meters in the UK. The prototype is implemented in C and makes use of well-known widely-used libraries for cryptographic functions and protocols, the TCP/IP stack, memory management and C library functions. Where possible, unused functionality in these libraries has been disabled, but the libraries themselves are unmodified. To enable the relying parties to inspect this software, the prototype is implemented as open source software under the BSD licence. To overcome the unique security challenges introduced by having a backup and recovery system, this architecture describes two possible backup mechanisms: the first is an offline backup sealed to a specific TPM and PCR state and the second is an online backup involving mutual attestation between two TREs. Both mechanisms protect the confidentiality and authenticity of the backup and also defend against the injection of false data into the TRE through this mechanism. The online backup mechanism therefore enables the transferral of trust relationships between TREs. In order to reduce the complexity of dynamic hardware configuration, the prototype uses preconfiguration and includes a Linux utility to generate the required configuration structures. The measurements of the TRE software and configuration structures are performed by the DRTM late-launch and the tboot framework. Importantly, the configuration information, which is not security sensitive, is extended into a different PCR from the TRE software. This allows a regulator or similar trusted entity to publish a list of trustworthy TRE signatures, which can be verified by anyone, in order to simplify the attestation process.

Various alternative TRE architectures have been presented and discussed. In the Linux-TPM architecture, the TRE software runs as an application on a commodity OS.

Although it can improve performance by using the optimized functionality provided by the OS, this architecture results in the full OS kernel being included in the TCB, even though most of its functionality is unnecessary for the TRE. The VM-vTPM architecture, based on TrustVisor, provides a protected execution environment for the TRE whilst allowing it to delegate non-security-sensitive tasks to an untrusted commodity OS running on the same platform. Compared to the x86-TPM architecture, this architecture's TCB is almost identical in size and its computational performance is expected to be very similar. The ARM-TrustZone architecture lacks a root of trust for measurement in the secure world, thus necessitating the use of a software TPM in the secure world and moving the TRE software to the normal world. Since the software TPM is included in the TCB, this architecture will always have a larger TCB than the x86-TPM architecture and its computational performance is expected to be similar. Finally, the x86-SGX architecture will make use of SGX technology, when available, to isolate the TRE from a commodity OS in a similar manner to the VM-vTPM architecture. However, by providing this isolation functionality in hardware and making the CPU the root of trust, this architecture is likely to have a TCB that is at least 30% smaller than the x86-TPM architecture and offer superior computational performance.

The x86-TPM architecture and its prototype implementation were evaluated primarily in terms of TCB size and secondarily in terms of computational performance. The TCB of the prototype consists of approximately 25,000 lines of C code. The major contributors to this are the cryptographic library (58%) and the communication subsystem (24%). This TCB is three orders of magnitude smaller than a Linux kernel. Although full formal verification of this TCB is probably out of reach at present, advances in verification approaches and computational capabilities are likely to make this feasible in the near future. However, a TCB of this size is very amenable to security audits and the TRE is a particularly good target because of its use of widely-used libraries. The computational performance benchmarks show that there is sometimes a trade-off between TCB size and computational performance: components that have been optimized to provide higher performance often increase the size of the TCB. This can be seen when the x86-TPM architecture with its small TCB is compared to the Linux-TPM architecture which uses optimized components. However, as long as the computational performance is sufficient to enable the TRE's functionality, it does not affect the TRE's ability to enhance communication privacy. Even the slowest x86-TPM prototype can support at least 20,000 consumers in the three smart grid protocols from the previous chapter. Although this can be increased if necessary, it shows that the TRE's computational performance is sufficient for this application domain.

Overall, this chapter has complemented the previous chapter in confirming the first primary research hypothesis: the previous chapter showed how the TRE can be used to enhance communication privacy in the smart grid and this chapter has explained and demonstrated how the TRE itself can be realized. Although this chapter has discussed the overall attestation mechanism used by the TRE, one of the key elements of this mechanism, the remote attestation protocol, is the subject of the next chapter.

Chapter 7

TRE Remote Attestation

This chapter draws on research described in the following publication:

- A. J. Paverd and A. P. Martin, "Hardware Security for Device Authentication in the Smart Grid," In: *First Open EIT ICT Labs Workshop on Smart Grid Security (SmartGridSec12)*, 2012 [206].

7.1 Adversary Model and Security Requirements 159
7.2 Performance Challenges and Requirements 160
7.3 Current Remote Attestation Protocols 161
7.4 Final State Attestation Protocol 170
7.5 Evaluation 177
7.6 Verifying the Attestation 184
7.7 Summary 187

Remote attestation is the principal mechanism through which relying parties establish the trustworthiness of the TRE. As explained in the previous chapter, the purpose of remote attestation is to provide relying parties with sufficient information about a system in order to facilitate an attestation-based trust relationship. In this chapter, the system providing the information is referred to as the *prover*[1], the information itself as *measurements*, and the relying parties as *verifiers*[2]. The prover uses a *measurement process* to obtain the measurements, and a *remote attestation protocol*[3] to communicate these measurements to the verifiers in a trustworthy manner. The nature of the measurements could vary between systems: in binary attestation the measurements are cryptographic representations of all software that has been run on the platform, whereas in property-based attestation [231, 216, 196, 61, 153] the measurements are properties of the prover's system. Alam et al. [10] provide a summary of different types of measurement processes. As explained in the previous chapter, the measurement process requires a Root of Trust for Measurement (RTM). Since the TRE's measurement process has been discussed in Chapter 6, the focus of this chapter is primarily on the TRE's remote attestation protocol.

In an attestation protocol, the prover uses its Root of Trust for Reporting (RTR) to generate a *quote*[4] to convince remote verifiers that the measurements being communicated are authentic. Three of the four concrete TRE architectures presented in the previous chapter use a TPM (or vTPM or software TPM) as their RTR[5]. Therefore, without loss of generality, the techniques in this chapter are described using the TPM as the RTR.

This chapter presents an in-depth analysis of the remote attestation protocol used by the TRE. Section 7.1 defines the adversary model and security requirements for remote attestation protocols and Section 7.2 discusses the performance requirements in terms of attestation latency and scalability. In Section 7.3, current remote attestation protocols are summarized and evaluated with respect to the security and performance requirements. To overcome certain limitations of current protocols, Section 7.4 presents a new highly-scalable remote attestation protocol specifically designed for the TRE, called the Final State Attestation (FSA) protocol. As explained in Section 7.5, two types of comparative evaluation have been performed on the FSA protocol. Firstly, the security properties have been modelled and analysed using the TrustFound framework [26]. Secondly the protocol has been implemented as part of the TRE prototype and its performance has been evaluated. Section 7.6 discusses this protocol from the verifiers' perspective and describes a proof-of-concept mechanism for protecting the verification of the attestation. This chapter therefore extends and complements the previous chapter in describing the design and implementation of the TRE. However, the FSA protocol described in this chapter can also be used in various other systems and is thus described in its own chapter.

[1] Also called the attestor [232, 231, 60], prover [247, 104], or target [160, 67].

[2] Also called appraisers [67], challengers [232, 231, 160, 104], or verifiers [48, 60, 247].

[3] Also sometimes called an integrity reporting protocol.

[4] The term *quote* is used in both TPM and SGX-based remote attestation.

[5] To be precise, the TPM's cryptographically unique Endorsement Key (EK) is the RTR, but the EK is always bound to a specific TPM, which must be trusted to protect this key [255].

7.1 Adversary Model and Security Requirements

In the context of remote attestation, the adversary's objective is to subvert the attestation mechanism in order to abuse the resulting attestation-based trust relationship. For example, the adversary could attempt to masquerade as a legitimate prover and use attestation to falsely convince relying parties of its own trustworthiness, or interpose itself in a supposedly secure communication channel between the verifier and prover. Similarly to the adversarial TRE operator, this attestation adversary may control the communication channels and have physical access to a legitimate prover. Therefore, in addition to the capabilities of a DY adversary, the attestation adversary can load and execute any software on the prover's platform, reset the platform, modify system software, add and remove hardware peripherals, and read and write to non-volatile storage media. As with the adversarial TRE operator, it is assumed that the attestation adversary cannot subvert correctly implemented cryptographic primitives and will not perform sophisticated runtime or hardware attacks against the platform since the cost-benefit ratio of these attacks makes them economically infeasible.

The adversary has two primary vectors for subverting remote attestation: either to undermine the prover's measurement process (e.g. running malicious software on the prover without this being recorded) or to subvert the attestation protocol itself (e.g. exploiting flaws in the protocol). The trustworthiness of the measurement process is dependent on the overall system design and the RTM. The TRE's measurement process has been discussed in the previous chapter. Assuming that the measurement process is trustworthy, and that the measurements are thus accurate and complete representations of the prover, the remote attestation protocol must fulfil the following security requirements in order for the overall attestation mechanism to be trustworthy:

AR-1: Authenticity: Verifiers must be able to unambiguously determine that the measurements came from the entity with which they are communicating.

AR-2: Currentness[6]: Verifiers must be able to unambiguously determine that the measurements represent the current state of this entity.

A mechanism that fails to fulfil **AR-1** is vulnerable to masquerading attacks in which the adversary presents verifiers with measurements of a different entity. Since the objective of attestation is to eliminate the need for identity-based trust, it is insufficient to link the measurements to an identity unless the identity is uniquely and inextricably bound to a specific prover. For example, it is insufficient to link the measurements to the prover's long-term public key if the adversary can obtain the corresponding private key through physical access to the platform. A mechanism that fails to fulfil **AR-2** is vulnerable to replay attacks in which the adversary replays old measurements generated when the platform was previously in a trustworthy state. Similar security requirements have also been proposed by Coker et al. [67].

[6] In some cases, the term *freshness* is also used to refer to the temporal validity of the measurements [160].

These requirements are directly applicable to the TRE. For example, in the privacy-enhancing smart grid communication protocols presented in Chapter 5, the TRE uses remote attestation to establish trust relationships with consumers and service providers. The consumers and service providers must therefore be able to determine that the TRE's attestation actually originates from the entity with which they are communicating and that it represents the current software state of the TRE.

7.2 Performance Challenges and Requirements

As explained in Chapter 2, the purpose of the TPM-based remote attestation is to communicate the TPM's PCR values to the verifier. The TPM creates a *quote*, in which the current values of some subset of the PCRs are signed by Attestation Identity Key (AIK). The AIK is accompanied by a certificate asserting that its private key is securely held within a genuine TPM, thus assuring the verifier that, if the signature is valid, the quote was generated by a genuine TPM.

Since the TPM 1.2 is designed to be a low-cost component, the `TPM_Quote()` operation used to produce a quote is relatively slow. To quantify this, micro-benchmarks were performed on a TPM 1.2 from ST Microelectronics (chip version: 1.2.7.40). The average time to perform one quote operation is approximately 731 ms with a standard deviation of 0.7 ms over 3000 samples. Furthermore, the TPM 1.2 can usually only perform one `TPM_Quote()` operation at a time. Abd Aziz et al. [6] present a performance analysis of TLS connections that have been augmented with mutual TPM-based attestation. Their results show that the total time required to establish this type of connection is between 1.63 seconds and 3.04 seconds, depending on the implementation. In comparison, in their experiment, it takes only 0.20 seconds to establish a unilaterally-authenticated TLS connection without attestation.

In many remote attestation use cases there are many provers and a small number of verifiers. For example, in Trusted Network Connect (TNC), the network operator can request a TPM quote from each client before allowing the client to join the network [254]. Although each prover must perform the slow `TPM_Quote()` operation, this is only performed infrequently by each prover (e.g. once per TNC session) and thus does not have a major impact on performance. The verifier is required to make trust decisions about many provers, but since these decisions do not require a TPM, they can be performed significantly faster on the main CPU and parallelized as required.

In contrast, the TRE uses remote attestation in the opposite direction to prove its software state to all relying parties, thus resulting in a single prover and many verifiers. This is the case when the TRE is used in the smart grid, as described in Chapter 5, as well as in other application domains, as described in Chapter 8. Given the performance limitations of the TPM 1.2, an attestation protocol that requires a separate TPM quote for each verifier would severely limit the rate at which the TRE could establish trust relationships. For example, if such a protocol were used with the TRE prototype from

the previous chapter, the capacity of the prototype would be reduced from over 20,000 consumers to fewer than 1,250 consumers since the TRE would have to perform two `TPM_Quote()` operations per consumer per time period. Although the TPM is probably the most well-known example, this type of limitation also applies to any RTR that takes a non-trivial amount of time to produce the attestation statement. Therefore, in addition to the security requirements defined in the previous section, it is important to impose performance requirements on the remote attestation protocol when used in the TRE.

In the context of remote attestation protocols, the term *latency* refers to the time taken to complete a single attestation. The latency takes into account the number of messages that must be sent between the prover and verifier, the size of the messages, and the number of cryptographic operations that must be performed by the prover and verifier. Since the actual time required to complete an attestation will vary depending on the communication latency and bandwidth of the network as well as the parties' computational performance, attestation latency is always used as a relative measurement between two attestation protocols. In this context, *scalability* refers to the rate at which attestations can be performed by a single prover with multiple verifiers, as is the case for the TRE. Although it would also be possible to consider scalability in the context of multiple provers and a single verifier, this is beyond the scope of this research. If any of the attestation operations performed by the prover cannot be parallelized, the absolute time required to perform these operations limits the scalability of the protocol. Since the overall communication system in which the attestation protocol is used will have certain performance requirements in terms of latency and scalability, the attestation protocol itself must meet the following performance requirements:

AR-3: Latency: The time taken for a single attestation must be within the bounds of the context in which it is used.

AR-4: Scalability: The prover must be able to perform the attestation protocol at a sufficient rate to support the expected number of relying parties.

These performance requirements are also directly applicable to the TRE. For example, in the smart grid communication protocols, if the attestation latency were too high, the information could not be communicated in a timely manner. The TRE is required to perform the attestation protocol at least twice per time period with each relying party and thus the scalability of the attestation protocol limits the number of relying parties that can be supported by a single TRE. Therefore, provided an attestation protocol fulfils the security requirements defined in the previous section, the main design objectives are to minimize attestation latency and maximize the scalability of the protocol.

7.3 Current Remote Attestation Protocols

Various remote attestation protocols have been proposed in the literature. Although most fulfil the security requirements defined above, these protocols differ in terms of their

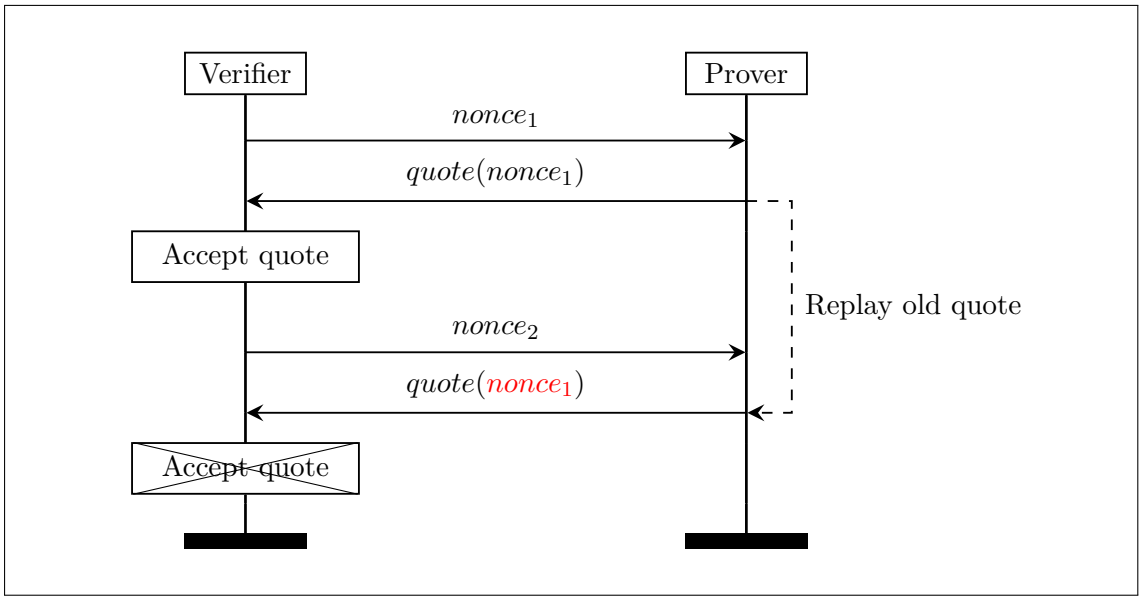

Figure 7.1: Nonce-challenge attestation protocol

latency and scalability in the context of a single prover and multiple verifiers. Since one of the primary performance constraints is the time taken to produce the TPM quote, some of these protocols use a single quote for multiple verifiers. It is therefore possible to distinguish between *one-to-one* attestation protocols, in which each verifier requires a separate quote, and *one-to-many* attestation protocols, in which a single quote can be used for multiple verifiers. It is also possible to consider *many-to-one* or *many-to-many* attestation protocols, but these are beyond the scope of this research since they are not relevant to the TRE.

7.3.1 One-to-One Protocols

Nonce-Challenge Attestation

One of the original TPM remote attestation protocols, as proposed by Sailer et al. [232], is a one-to-one attestation protocol that uses a nonce-based challenge. To prevent replay attacks, the TPM allows some externally-supplied data (160 bits on the TPM 1.2) to be included with the PCR values in the structure signed by the AIK. As shown in Figure 7.1, when requesting a quote, the verifier supplies a new unpredictable nonce. If this nonce is present in the quote, the verifier can be sure that the quote is not a replay of an earlier TPM quote. However, since the state of the platform can change immediately after the creation of the quote, this approach only partially satisfies AR-2 because even a recent quote does not necessarily represent the *current* platform state. To fulfil AR-1, the verifier needs some guarantee that the quote represents the state of the entity with which it is communicating. In the original protocol, this is achieved by including the identity of the prover (e.g. its long-term TLS public key) in the AIK certificate such that the verifier can

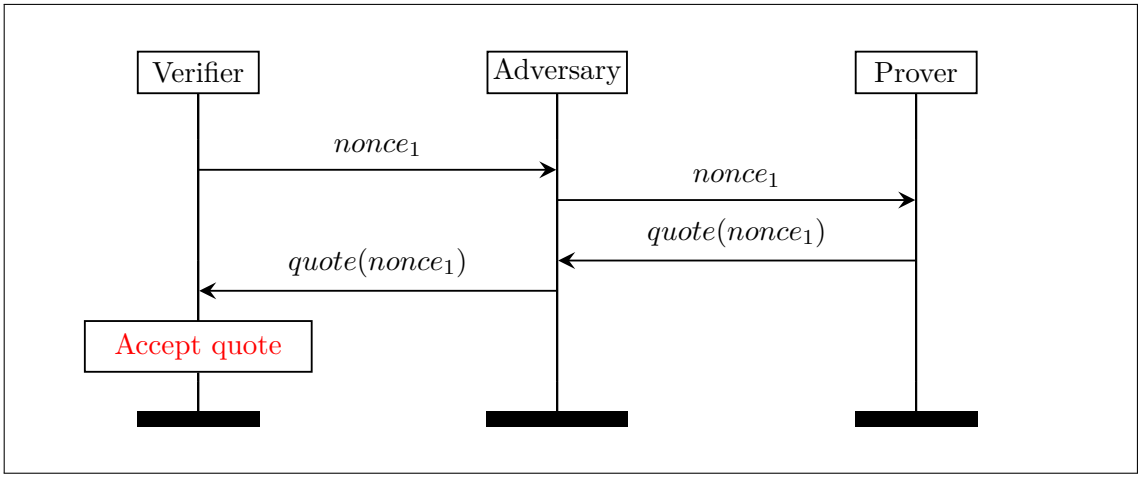

Figure 7.2: Masquerading attack against a nonce-challenge attestation protocol

compare this to the identity used to establish the communication channel. Alternatively, Goldman et al. [116] achieve a similar result by extending a measurement of the prover's identity into the PCRs.

Similar nonce-challenge attestation protocols have been used in various contexts. There have been proposals for using this type of protocol for property-based attestation [216, 153]. Sadeghi and Schulz [230] have proposed an extension to IPsec that allows the handshake phase to include nonce-challenge attestation. Chen et al. [62] use this type of protocol for authenticating devices and users in WiMAX networks. LeMay et al. [161] use it to authenticate smart meters to service providers and other entities. Notably, the TrustVisor [182] framework described in the previous chapter makes use of this type of protocol and acknowledges its impact on attestation latency.

Nonce-Challenge with Key Confirmation

Stumpf et al. [246] have pointed out that in some contexts, the adversary may be able to extract the long-term TLS keys from an honest prover under the adversary's control. As shown in Figure 7.2, the adversary can then perform a masquerading attack by forwarding the verifier's nonce to this honest prover and returning the correct quote to the verifier using the same identity, thus making the adversary appear trustworthy. They propose an improved protocol that prevents this attack by performing a Diffie-Hellman (DH) key exchange between the verifier and the prover in parallel with the attestation. The prover generates a new DH key pair and includes the public key in each quote. When requesting a quote, each verifier sends its own DH public key to the prover. After receiving and verifying the quote, the verifier computes a shared secret key with the prover based on the prover's DH public key, which has been authenticated by the quote. The verifier then performs a *key confirmation* step by sending a new unpredictable nonce to the prover. The prover demonstrates knowledge of the shared secret by using it to encrypt this nonce and returning

the result to the verifier, who checks that the encryption is correct. This therefore links the attestation measurements to the entity with which the verifier is communicating. This protocol assumes that, unlike a long-term key pair, the prover's DH private key cannot be extracted by the adversary since it is randomly generated for each attestation. This is a reasonable assumption since it can be enforced by the design of the honest prover.

Nonce-Challenge with Channel-Binding

If the attestation is carried out within a secure channel between the verifier and the prover, and if this secure channel provides Perfect Forward Secrecy (PFS), then the above protocol by Stumpf et al. [246] can be simplified into a single interaction between the verifier and the prover. In TLS, cipher suites that use ephemeral DH key exchanges (e.g. ECDHE_...) can provide PFS. Assuming that TLS is used, a masquerading attack requires two TLS connections: one between the verifier and the adversary and one between the adversary and the honest prover. Even if the adversary has extracted the long-term keys from the honest prover, the adversary does not know the ephemeral private values that are generated by the honest prover to achieve PFS for each connection. Therefore the TLS master secrets for these two connections will always be different. The prover must ensure that, in addition to the verifier's nonce, the quote also includes a hash of this master secret or some equivalent quantity derived from the master secret. If the quote verification is successful, it means that the verifier and prover are using the same TLS master secret and are therefore communicating with each other. This relies on the same assumptions as the protocol by Stumpf et al. [246] above. This modified protocol achieves *channel-binding* in which the attestation is linked to a specific secure communication channel.

Non-Migratable TPM Keys

In contrast, Löhr et al. [166] have proposed a protocol based on non-migratable TPM keys that does not require any `TPM_Quote()` operations. In their protocol, the TPM creates a non-migratable key that is locked to a particular set of PCR values (i.e. it can only be used when the platform is in a specific state). They use the `TPM_CertifyKey()` operation to produce an *attestation token*, signed by an AIK, linking the public part of this key to the PCR values for which it can be used. This token can then be distributed to all verifiers and inspected without involving the prover. To establish a trust relationship, a verifier sends the prover a challenge (i.e. an unpredictable nonce) to be signed using this key. If the signature is correct, the verifier can be sure that the prover was in the advertised state at the time the challenge was signed. To ensure authenticity, the signing key must be non-migratable and thus the signature operation can only be performed on the TPM. The `TPM_Sign()` and `TPM_Quote()` operations have similar durations since both create the same type of signature. Therefore, although this protocol does not use TPM quotes, it still requires the prover to perform a TPM signature operation for each verifier, and can thus be classified as a one-to-one protocol. Furthermore, although not stated in

the protocol, some type of individual channel-binding would have to be included in the signature to prevent masquerading attacks.

Latency and Scalability

Although most of the above one-to-one protocols meet the security requirements and might be sufficient for infrequent attestations, they would severely limit the communication performance of the TRE. By definition, the attestation latency of a one-to-one protocol is at least the time required to perform one RTR operation, which in the case of the TPM is a `TPM_Sign()` or `TPM_Quote()` operation. Since these TPM operations usually cannot be parallelized, the average latency will be significantly higher if the instantaneous rate of requests exceeds the rate at which these operations can be performed. Since each attestation can only be used by a single verifier, the latency of the attestation directly affects the scalability of these protocols. If the average rate of requests of a system exceeds the overall rate at which attestations can be performed, this type of one-to-one protocol cannot be used.

7.3.2 One-to-Many Protocols

To overcome the scalability limitations of one-to-one protocols, various one-to-many attestation protocols have been proposed.

Multiple-Hash Attestation

Another protocol by Stumpf et al. [247] amortizes the cost of a `TPM_Quote()` operation over multiple attestations. Verifiers still send individual nonces to the prover but instead of being sent directly to the TPM, these are merged together into a *NonceList*. At some point, the TPM produces a quote which includes the hash of the *NonceList*. The quote and the *NonceList* are provided to the relevant verifiers who can check that their individual nonces are included in the list. To increase scalability, the TPM produces quotes continuously and any requests that arrive while the TPM is busy are added to the *NonceList* for the next quote. To prevent masquerading attacks, this protocol uses the same key confirmation step as the nonce-challenge protocol with key confirmation proposed by some of the same authors [246]. As described above, this involves a DH key exchange in parallel with the quote request and then requires an additional challenge-response step to confirm the newly-established key. However, since the same DH key is used for multiple quotes, an implicit assumption is that the prover cannot transition into an untrusted state and then return to a trusted state because the DH key could have been extracted during the untrusted state. This is a reasonable assumption due to the extend-only nature of the TPM's PCRs (i.e. the prover would not be able to return to a trusted state without resetting the PCRs through a platform reset or DRTM late-launch). In theory, any number of nonces can be included in each *NonceList* and thus the scalability of the protocol is limited only by the rate at which the prover can communicate the quotes and *NonceLists* to verifiers and the

rate at which the key confirmation step can be performed. Assuming attestation requests arrive according to a uniform random distribution, the average time between arrival and the start of the next quote operation is half the time taken for one quote. The average latency of a single attestation in this protocol is therefore 1.5 `TPM_Quote()` operations plus the time required for the verifier to obtain the quote and *NonceList*, verify the quote, and perform the key confirmation step.

Timestamp-based Attestation

Moyer et al. [195] have presented *Spork*, a system that uses TPM-based remote attestation to link the content of web pages to the state of the web server from which they are served. Spork aims to allow web browsers to verify the integrity of the server before displaying web content. As with the TRE, this also leads to a single prover and many verifiers and thus one-to-one attestation protocols are insufficiently scalable given the performance of the TPM and the expected number of web clients. To overcome this, they propose a one-to-many attestation protocol based on trusted global timestamps from a separate trusted time server. Instead of creating a separate quote for each page, they use a cryptographic proof system based, on Merkle hash trees, such that all pages can be represented by a single root hash value. The information required by the verifier to check a particular page in this proof system grows logarithmically with the total number of pages. This means that the communication requirements for each attestation increase logarithmically with the total number of pages. The server periodically creates TPM quotes that include the most recent timestamp and the Merkle tree root. Verifiers can check the page content using the Merkle tree. In order to check that the timestamp is sufficiently recent, all verifiers must synchronize their clocks with the same time server, which is assumed to be globally trusted and thus becomes part of each participant's TCB [2]. Since a TPM 1.2 is used as the RTR, the rate at which the server can produce quotes is limited by the `TPM_Quote()` operation. For static content, the most recent quote is available immediately and thus the latency is only the time taken by the verifier to fetch this quote. For dynamically-generated content, the new content is included in the next server quote. Similarly to the multiple-hash protocol above, assuming attestation requests arrive according to a uniform random distribution, the average time between arrival and the start of the next quote operation is half the time taken for one quote. The average latency for attesting dynamic content is therefore 1.5 `TPM_Quote()` operations plus the time taken for the verifier to fetch and verify this quote. Since these quotes can be checked by any verifier, the only factor that limits scalability is the rate at which these quotes can be communicated to the verifier.

However, this protocol does not meet AR-1 because it does not guarantee that the content originated from the entity with which the verifier is communicating. It would not be possible to detect a masquerading attack in which the verifier is actually communicating with the adversary who is simply forwarding content from an honest prover, as shown in

Figure 7.2. This is not necessarily a problem if the communication is only uni-directional from the server to the client. However, if secure bi-directional communication is required (e.g. for the client to submit passwords or other private information), this protocol cannot be used because it gives no guarantees about the entity with which the verifier is communicating. Even if the server's long-term public key were included in the quotes, the adversary might have access to this key as explained by Stumpf et al. [246]. Thus this type of attestation protocol would be useful in an Information-Centric Networking (ICN) paradigm but cannot be used in the TRE as it does not meet the security requirements.

Timestamped Hash Chain Attestation

Stumpf et al. [247] have proposed a similar timestamp-based one-to-many protocol that prevents this type of masquerading attack. As in the multiple-hash attestation protocol by the same authors, this protocol includes a DH key exchange in parallel with the quote request and an additional key confirmation step. To reduce the load on the time server, the prover requests a single signed time value at the start of each session and then computes the hash of this value at set intervals. The most recent value in this hash chain is used in the quote as if it were a trusted timestamp. In each attestation, the verifier receives the initial signed timestamp, the number of elapsed time intervals and the most recent quote. If the verifier trusts that the prover will only update the hash chain at the specified rate (e.g. based on evidence from the PCRs), the verifier can determine when the quote was generated. Again it must be assumed that the prover cannot transition from an untrusted state to a trusted state without a platform reset or late-launch because, if this were possible, the untrusted state might have updated this hash chain at a faster rate, thus invalidating the relationship to the original signed timestamp. Since this is a one-to-many protocol, its scalability is only limited by the rate at which the prover can establish connections and perform the additional key confirmation step with each verifier. The attestation latency is the time it takes the verifier to fetch the quote and perform the key confirmation step. One limitation of this attestation protocol is that it relies on a separate entity to provide trusted timestamps, and thus this protocol cannot be used for attesting that entity.

Tickstamp Attestation

A variant of the above protocol, also proposed by Stumpf et al. [247], replaces the timestamp hash chain mechanism with the TPM's internal tick counter. In the setup phase, the prover creates a non-migratable TPM key that is locked to a specific set of PCR values and uses the `TPM_CertifyKey()` operation to create an assertion to this effect, signed by an AIK. Instead of creating a quote, the prover periodically uses the `TPM_TickStampBlob` operation to get the current value of the TPM's tick counter, which is signed by this non-migratable key. This tick counter is controlled by the TPM and is thus guaranteed to advance at a specified rate. To establish the relationship between the tick counter and

the current time, each verifier must initially perform a time-synchronization protocol with the prover, either using a one-to-one nonce-challenge protocol or a trusted time server. Again the prover's DH public key is included in the signature and a key confirmation step is performed to prevent masquerading attacks. Similarly to the previous protocol, the scalability is limited only by the rate at which the prover can perform the key confirmation step and the latency is the time it takes the verifier to fetch the quote and perform this key confirmation step. A limitation of this protocol is that it requires each verifier to perform an individual time synchronization with the prover or use a trusted time server.

Non-Migratable TPM Keys

Gasmi et al. [109] present a system architecture and remote attestation protocol for establishing trusted channels between two platforms. Their aim is to achieve mutual attestation between the participants whilst minimizing the time required to establish this trusted channel. In their system architecture, which must be applied to both sides of the trusted channel, a static TCB controls the encryption keys for the channel and decrypts data on behalf of other components on the same platform but outside the TCB. If the state of any of these components changes, the TCB will refuse to decrypt more data until the other communicating party has been informed and accepted this change. Quotes are created using non-migratable TPM keys in a similar manner to Löhr et al. [166], as discussed above. The TPM creates a non-migratable key $\mathcal{K}_{bind}$, which is locked to a particular set of PCR values. The `TPM_CertifyKey()` operation is used to produce an assertion to this effect signed by an AIK. Instead of using $\mathcal{K}_{bind}$ to sign verifiers' challenges, as in the one-to-one protocol, this protocol uses it to sign the public key of another key pair, $\mathcal{K}_{sign}$ (or $\mathcal{K}_{enc}$ if it is used for encryption rather than signing). This signature represents an implicit assertion that $\mathcal{K}_{sign}$ was generated when $\mathcal{K}_{bind}$ was available (i.e. when the PCRs were in the advertised state) and that the Trusted Computing Base (TCB) intends to protect $\mathcal{K}_{sign}$ from all other entities. To achieve this, $\mathcal{K}_{sign}$ is sealed to the current PCR state to protect it if the platform is reset. Since the TPM does not have the capability to attest to this sealing, the verifier must rely on the indirect evidence of the signature by $\mathcal{K}_{bind}$ and the PCR state to which $\mathcal{K}_{bind}$ is locked. The TLS handshake protocol and certificate are extended to include this attestation protocol. The TLS certificate includes the public key $\mathcal{K}_{bind+}$, the assertion linking $\mathcal{K}_{bind+}$ to specific PCR values, the AIK signature of this assertion (included using a Subject Key Attestation Evidence (SKAE) extension [262]), the public key $\mathcal{K}_{sign+}$, and the signature of this key by $\mathcal{K}_{bind-}$. This certificate can either be signed by a regular CA or self-signed by the TCB. The TLS handshake then proceeds as before using $\mathcal{K}_{sign}$ as the prover's public key. Since $\mathcal{K}_{sign}$ is not held by the TPM, it can be used to perform cryptographic operations on the main CPU as in a standard TLS handshake. If the TLS handshake succeeds, the prover must have had access to $\mathcal{K}_{sign}$, which means that the PCRs are in the advertised state. The proposal by Armknecht et al. [18] uses the same approach but sends all messages using standardized TLS extensions.

Although it is not specifically described as a one-to-many protocol, this type of approach only requires a single TPM operation (either signing or unsealing $\mathcal{K}_{sign}$) to achieve attestation with multiple verifiers. This is highly scalable because the only limitation on scalability is the rate at which the TLS handshakes can be completed when these additional attestation messages are included. The attestation latency of this protocol is only the additional time taken for the TLS handshake (since the actual channel establishment is not included in the latency measurement) and the time taken by the verifier to check the respective signatures. Gasmi et al. [109] make the assumption that the TCB will not change whilst Armknecht et al. [18] present a mechanism for updating the TCB. In this update mechanism, a platform reset is required to ensure that $\mathcal{K}_{sign}$, which has been unsealed by the old configuration, cannot be used by the possibly untrusted new configuration. Although $\mathcal{K}_{sign}$ can be sealed against the new PCR values, a new $\mathcal{K}_{bind}$ and AIK signature is required for each new configuration, since the PCR values will have changed. One limitation of this approach is that $\mathcal{K}_{sign}$ has a virtually unlimited validity period since no validity information is included in either of the signatures that connect this key to the PCR values (i.e. the AIK signature of $\mathcal{K}_{bind}$ and the $\mathcal{K}_{bind}$ signature on $\mathcal{K}_{sign}$). By definition, these signatures are public because they are sent to all verifiers. If $\mathcal{K}_{sign}$ were somehow to be compromised (e.g. through a vulnerability in the TCB software), the adversary would be able to masquerade as the trusted platform indefinitely. Once this is detected, either $\mathcal{K}_{sign}$ or $\mathcal{K}_{bind}$ must be added to some type of revocation list that is checked by all relying parties, thus increasing the latency of each attestation.

7.3.3 Other Types of Protocols

Direct Anonymous Attestation (DAA)

Direct Anonymous Attestation (DAA), as proposed by Brickell, Camenish, and Chen [48], is not strictly an attestation protocol but rather an alternative scheme for protecting privacy in remote attestation without the need of a Privacy CA. DAA is included as part of the TPM specifications [255]. After receiving a DAA credential from a *DAA issuer* (i.e. an entity similar to a privacy CA), a TPM can produce DAA signatures for any new AIKs it generates. From this point onwards, the AIKs can be used as before in any of the protocols above and thus do not affect the scalability of the protocol. However, if the verifier decides to provide a challenge for the DAA signature, then a new signature must be created with the `TPM_DAA_Sign()` operation which is a highly resource intensive, and thus relatively slow, TPM operation [255].

Intel SGX

When attesting an Intel SGX enclave (i.e. using SGX as the RTR), the process of producing a quote differs somewhat from the process of producing a TPM quote. SGX provides two types of enclave attestation: intra-platform attestation provides attestation between

different enclaves on the same platform whilst inter-platform attestation allows attestation of an enclave to a verifier on another platform [12]. Intra-platform attestation is managed completely by the CPU and uses only symmetric keys. The proving enclave uses the `EREPORT` instruction to create a *report* structure for a specific verifying enclave. The report is automatically encrypted using the verifying enclave's symmetric report key. Each enclave can obtain its own report key using the `EGETKEY` instruction. The security properties of the reports (i.e. **AR-1** and **AR-2**) are guaranteed by the CPU. The report structure contains a description of the properties of the proving enclave (i.e. the measurements) as well as a 256 bit user data field. This can be used to establish confidential mutually-authenticated communication channels between enclaves by including each enclave's pubic DH key in the respective reports.

For inter-platform attestation, a special *quoting enclave* is used. Similarly to the TPM, the remote verifier supplies a challenge, including an unpredictable nonce to ensure currentness, which is passed to the proving enclave. The proving enclave performs an intra-enclave attestation with the quoting enclave and includes the verifier's nonce in the report. The quoting enclave creates a quote by signing the report with the platform's Enhanced Privacy ID (EPID) key[7]. This quote is returned to the verifier who checks the signature using the platform's EPID public key certificate and confirms that the challenge nonce has been included.

Therefore, both intra-platform and inter-platform attestation are one-to-one in nature. Since all operations are performed on the main CPU, the performance is likely to be significantly higher than that of the TPM 1.2 described above. However, each inter-platform attestation still requires an intra-platform attestation and two context switches between the proving and quoting enclaves, which may limit performance if multiple attestations are required.

7.4 Final State Attestation Protocol

Building on the strengths of previous attestation proposals, the Final State Attestation (FSA) protocol has been specifically designed to meet the security and performance requirements of the TRE. To achieve this, the protocol takes advantage of the fact that the TRE reaches a *final state* from the perspective of the measurement process. The nature of the TRE allows it to make the assertion that once it has reached its final state, it will not leave this state voluntarily. Since this type of assertion cannot be made for every system, the FSA protocol is not a general-purpose attestation protocol. However, in addition to its use in the TRE, this protocol can be used in other systems that reach a final state, as explained in this section.

[7]EPID is a type of DAA scheme, similar to the original, but with enhanced revocation capabilities to deal with compromised provers [49].

7.4.1 FSA Protocol in the TRE

Since the TRE is a single-purpose system that does not load any additional software, it naturally reaches a final state which it will not leave voluntarily. In the x86-TPM TRE architecture from the previous chapter, this final state is reached after the TRE software has been loaded and measured by the DRTM late-launch and has commenced execution. The TRE extends a well-known value[8] into one of the PCRs to indicate that it has reached this state but the actual validity of this claim must be checked by a verifier by inspecting the TRE software. The verifier must be convinced that the TRE will not perform any action that changes the PCR values (i.e. it will not load any further software). This type of final state is also reached by other systems, including the system architectures by Gasmi et al. [109] and Armknecht et al. [18], and the TrustVisor system by McCune et al. [182].

Once the prover has reached its final state, the FSA protocol is used to convince verifiers that the prover is currently in this state. In conveying this information, the FSA protocol must fulfil the two security requirements defined in Section 7.1. To prevent the masquerading attack described above, the FSA protocol must provide channel-binding between the attestation information and the secure channels. The currentness requirement (**AR-2**) is simplified because once the prover is in the final state, by definition, it will not change its state voluntarily. Therefore this state can only be forcibly changed by the adversary. Since the TRE's protected execution environment prevents the adversary from modifying the state of a running TRE (except by sophisticated hardware attacks or runtime vulnerabilities, which are beyond the scope of this research), the only way the adversary can change this state is by resetting the platform and loading different software. Therefore, to fulfil **AR-2**, the FSA protocol has only to demonstrate that at some point in the past the prover reached its final state and that it has not been reset since that point.

The FSA protocol is shown in Figure 7.3. This figure assumes that the verifier and prover wish to establish a TLS connection, but any equivalent mechanism could be used. The figure shows the initiator of the communication (e.g. the TLS client) as the verifier but it is also possible for the protocol to be used in reverse such that the initiator is the prover. The main philosophy of this protocol is that once the prover reaches its final state, it generates a new ephemeral asymmetric key pair $\mathcal{K}_{FSA+}$ and $\mathcal{K}_{FSA-}$. The private key $\mathcal{K}_{FSA-}$ is stored in volatile memory and the verifier can confirm this from the quote. This means that if the prover's platform is reset, the private key is assumed to be irrecoverably lost. This assumption is supported by technologies such as Intel TXT, as used in the x86-TPM TRE prototype, which allows the prover to set the `TXT.CMD.SECRETS` flag to indicate that there are secrets in memory [132]. If the platform is reset without clearing this flag, TXT will scrub all memory before the platform is allowed to boot.

As shown in Figure 7.3, the setup phase of the protocol is performed when the system reaches its final state. The prover generates a new ephemeral key pair $\mathcal{K}_{FSA+}$ and $\mathcal{K}_{FSA-}$. The prover then creates a certificate $cert_{FSA}$ containing the public key $\mathcal{K}_{FSA+}$. This

[8] For example: `0x 46 69 6E 61 6C 20 53 74 61 74 65 00 00 00 00 00 00 00 00`

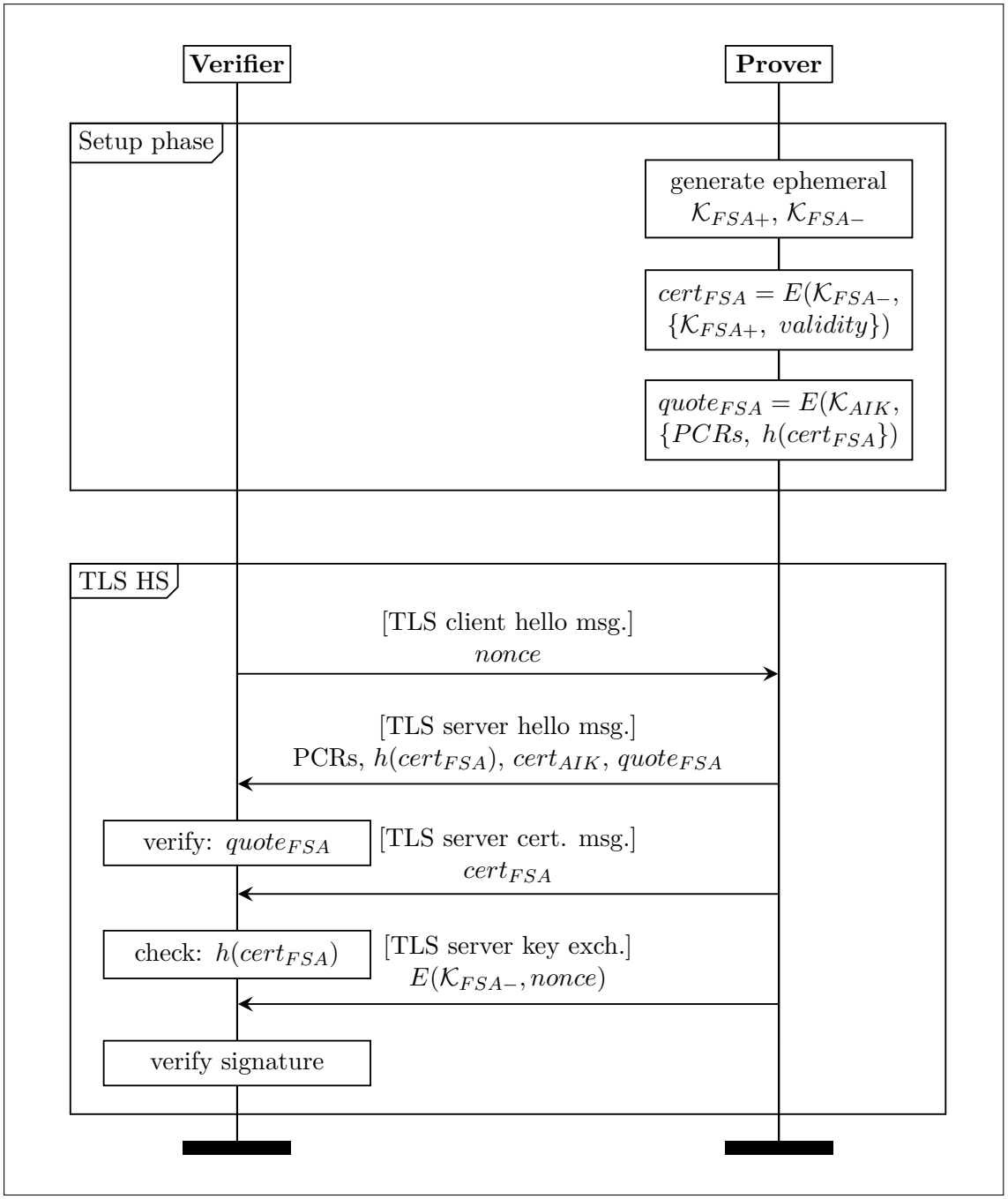

Figure 7.3: TLS handshake augmented with the Final State Attestation (FSA) protocol

```
Trustworthy Remote Entity
  Identity:                  Trustworthy Remote Entity
  Verified by:               Trustworthy Remote Entity
  Expires:                   18/03/15

Subject Name
  C (Country):               GB
  ST (County):               Oxfordshire
  L (Locality):              Oxford
  O (Organisation):          University of Oxford
  OU (Organisational Unit): Department of Computer Science
  CN (Common Name):          Trustworthy Remote Entity

Issuer Name
  C (Country):               GB
  ST (County):               Oxfordshire
  L (Locality):              Oxford
  O (Organisation):          University of Oxford
  OU (Organisational Unit): Department of Computer Science
  CN (Common Name):          Trustworthy Remote Entity

Issued Certificate
  Version:                   3
  Serial Number:             01
  Not Valid Before:          2015-03-17
  Not Valid After:           2015-03-18

Certificate Fingerprints
  SHA1:                      FA EC D8 57 08 E4 6B 3E 3A 9C BA 1F 52 32 B6 CE 66 5A BC 4F
  MD5:                       BD E3 4B 20 95 45 8B 29 84 06 9A 7D DD 13 72 20

Public Key Info
  Key Algorithm:             1.2.840.10045.2.1
  Key Parameters:            06 08 2A 86 48 CE 3D 03 01 07
  Key SHA1 Fingerprint:      8C 31 11 0D E4 43 CC 57 0C 4D A4 9C E8 F4 72 9A FA DA 40 C1
  Public Key:                04 87 F2 AB 92 D5 6C F7 5A 89 EE 2C 9E EA 9E 45 57 BA E0 20
                             4F 80 81 8B CE 2D AF 5F 6C 16 DD D1 32 1F 3C E9 D0 2E F7 3E
                             FC 71 3F EC 07 C6 97 70 74 F4 95 F0 1B 93 45 B6 D0 63 62 56
                             9E CE 63 7C 2F

Basic Constraints
  Certificate Authority:     No
  Max Path Length:           Unlimited
  Critical:                  No

Subject Key Identifier
  Key Identifier:            33 92 D0 C6 D6 F2 C2 2D A7 F7 F0 71 E2 1F CF C8 18 55 DB 74
  Critical:                  No

Signature
  Signature Algorithm:       1.2.840.10045.4.1
  Signature Parameters:      05 00
  Signature:                 30 45 02 20 14 8B 92 38 4C 96 1D 9C B4 AC E9 58 8D D6 0C 0D
                             19 9A FC 95 0C AE 62 A0 A8 80 ED 83 D2 03 39 AD 02 21 00 DE
                             58 18 89 5C 3B B4 5A DD C4 BB 39 8D F0 CF 2A F0 C0 FF 3C CF
                             DD 0B 23 BF 46 A4 B4 CD DD 68 94
```

Figure 7.4: Final State Attestation Certificate $cert_{FSA}$

certificate can be self-signed or signed by any other entity without affecting the security guarantees of this protocol. In addition to the public key, $cert_{FSA}$ also defines the temporal validity of the key pair. This does not require a trusted time source since the only security-critical property is the length of the validity period (i.e. the difference between the `Not Valid After` and `Not Valid Before` properties). If the TRE's internal clock is set incorrectly (maliciously or otherwise), this simply results in a denial of service attack, which can be trivially achieved by the attestation adversary through various other mechanisms. $cert_{FSA}$ can be in X.509 format if required. An example of a $cert_{FSA}$ generated by the TRE prototype is shown in Figure 7.4. After generating this certificate, the prover produces a single quote $quote_{FSA}$ of the current (final state) PCR values using the hash of $cert_{FSA}$ as the user-supplied data. As usual, the quote is signed by a TPM AIK. This quote therefore binds the ephemeral key pair and certificate to the current state of the system.

Table 7.1: Structure of the FSA TLS extension

Field name	**Description**	**Size (bytes)**
pcr_size	PCR composite structure size	2
pcr	PCR composite structure	PCR_size
$cert\_size$	Certificate hash size	2
$h(cert_{FSA})$	Certificate hash	$cert\_size$
aik_size	AIK certificate size	2
aik	AIK certificate	aik_size
$quote\_size$	Quote size	2
$quote_{FSA}$	TPM quote	$quote\_size$
Optionally:		
log_size	Secure measurement log size	2
log	Secure measurement log	log_size

Once the setup phase has been completed, the prover can begin accepting incoming connections from verifiers. Each verifier aims to establish a secure connection to the prover (e.g. a TLS connection) and to use remote attestation to verify that the entity with which it is communicating is in a trustworthy state (requirements **AR-1** and **AR-2**). To achieve this, the TLS handshake is augmented with the FSA protocol, as shown in Figure 7.3. The main difference compared with a standard TLS handshake is that the prover uses $cert_{FSA}$ and $\mathcal{K}_{FSA}$ instead of a long-term TLS certificate and key pair. When a verifier initiates a connection to the prover or vice-versa, a new type of TLS hello extension, the *FSA extension*, is added to the prover's first message (e.g. the TLS server hello message in Figure 7.3). The structure of this extension is shown in Table 7.1. This extension is used to communicate the PCR values, the hash of $cert_{FSA}$, $cert_{AIK}$, and $quote_{FSA}$ to the verifier. The FSA extension can also include the prover's measurement log and a link to the prover's source code, but this need not be included if it is well-known or available from other sources. For example, in the case of the TRE, before the attestation begins, the verifier could obtain the TRE's source code, analyse it, and calculate the expected PCR values. Alternatively, regulatory bodies or other trusted entities could publish lists of PCR values that they consider trustworthy. If there are restrictions on the length of the FSA extension, some or all of these pieces of information can be sent as TLS *supplementary data* messages. In the x86-TPM TRE prototype described in the previous chapter, with the measurement log omitted, the total length of the FSA extension was 924 bytes. The PCR structure was 29 bytes, the hash of $cert_{FSA}$ was 20 bytes, $cert_{AIK}$ was 611 bytes and $quote_{FSA}$ was 256 bytes.

After receiving the FSA extension, the verifier checks the signature on $cert_{AIK}$ to confirm that the AIK is held by a genuine TPM. It then uses the AIK, the PCR values and the hash of $cert_{FSA}$ to check the $quote_{FSA}$ signature. Finally, if the quote is valid,

the verifier uses these PCR values to make a trust decision about the prover, including whether or not the values represent the final state of the prover. If the PCRs represent a trustworthy final state, the verifier stores the hash of $cert_{FSA}$ and marks it as trustworthy. If attestation is not required, the client simply ignores the FSA extension. The TLS handshake then proceeds as usual. The prover provides the verifier with $cert_{FSA}$ in the TLS server certificate message and uses $\mathcal{K}_{FSA}$ to create the signature over the handshake digest (or performs the equivalent key exchange step if $\mathcal{K}_{FSA}$ is not a signing key). If the verifier accepts this signature, the TLS connection has been established and the attestation protocol has been completed successfully.

Since the secure communication channel is established using a TLS handshake (with the inclusion of some additional FSA information), it provides the same security guarantees as a standard TLS connection (e.g. protection against replay attacks and man-in-the-middle attacks). The additional FSA information is only used to verify the certificate presented by the prover during this handshake. In terms of the first attestation security requirement (authenticity), this protocol achieves channel-binding between the attestation and the secure channel through the link between $\mathcal{K}_{FSA+}$ and the TPM quote. The hash of $cert_{FSA}$ in the quote binds the attestation to $\mathcal{K}_{FSA+}$ and the use of $\mathcal{K}_{FSA-}$ in the handshake binds it to the channel. Even if the adversary can extract long-term keys from the prover, it cannot extract the ephemerally-generated $\mathcal{K}_{FSA-}$ (this is the same assumption made in previous proposals [246, 247, 109, 18]). If the private key $\mathcal{K}_{FSA-}$ could somehow be extracted from the prover (e.g. through an implementation flaw), it would allow the adversary to masquerade as the prover without even requiring the prover to be online. As a defence-in-depth measure, $\mathcal{K}_{FSA+}$ is given a relatively short validity period (e.g. hours) to reduce the economic feasibility of this attack. Frequent TLS key rotation is advantageous in most systems, but in systems that rely on identity-based trust, this is often limited by the cost of obtaining new CA certificates for each key. In contrast, the TRE's use of attestation-based trust and the FSA protocol allow it to generate new TLS keys as often as required and include them in new TPM quotes without any external interaction.

In terms of the second attestation security requirement (currentness), this protocol convinces the verifier that the PCR values represent the current state of the prover. The presentation of a valid quote demonstrates that the prover was in that state at the time the quote was produced (t_{quote}). The verifier can inspect this quote to ascertain that it represents the prover's final state. Since $\mathcal{K}_{FSA}$ is linked to the quote, this key must have been known to the prover at t_{quote}. If the TLS handshake completes successfully using $cert_{FSA}$, then this means that the prover still has access to the private key $\mathcal{K}_{FSA-}$. Since $\mathcal{K}_{FSA-}$ is generated ephemerally and stored in volatile memory, the fact that it is still accessible to the prover demonstrates that the prover has not been reset since t_{quote}. The protocol provides the verifier with proof that the prover is still in the final state represented by the quote, and thus fulfils both security requirements. The performance of this protocol is discussed as part of the evaluation in Section 7.5.

7.4.2 FSA Protocol in TrustVisor

In addition to its use in the TRE, the FSA protocol can also be used in other systems that exhibit this characteristic of reaching a final state. For example, this protocol can be used in TrustVisor [182]. TrustVisor uses a two-part attestation approach: a quote from the hardware TPM indicates the state of TrustVisor itself, and quotes from each μTPM attest to the state of the respective PAL. TrustVisor reaches a final state after it has been loaded and measured through the DRTM late-launch since it does not perform any further actions that change the measurement of its own TCB (the measurements of the PALs are extended into their own μTPMs, rather than the hardware TPM). As a special-purpose hypervisor, TrustVisor itself is never the end-point of a secure channel and thus a slightly different implementation of the FSA protocol, compared to that used for the TRE, must be used in this scenario. As with the TRE, when TrustVisor reaches its final state it generates the key pair $\mathcal{K}_{FSA+}$ and $\mathcal{K}_{FSA-}$ and stores the private key in volatile protected memory so that it cannot be accessed by the untrusted OS or any of the PALs. Since TrustVisor cannot use $cert_{FSA}$ to convey information to the verifier, it extends a hash of the public key $\mathcal{K}_{FSA+}$ into the FSA PCR (PCR 20). It also extends the validity period of $\mathcal{K}_{FSA+}$ into the FSA PCR, based on the platform's internal clock. As above, an incorrect clock value would result in denial of service but would not affect the security properties. Finally, TrustVisor extends a well-known value into the FSA PCR indicating that it is using the FSA protocol. All of these PCR extend operations must take place before the untrusted OS is launched. Two new hypervisor calls are added to TrustVisor: `HV_FSA_Sign` allows the untrusted OS to pass a value to TrustVisor to be signed by $\mathcal{K}_{FSA-}$ and `HV_FSA_Info` returns the public key $\mathcal{K}_{FSA+}$ and the validity period that have both been extended into the FSA PCR.

The original TrustVisor design uses a one-to-one nonce-challenge protocol in which the verifier supplies two nonces, $n1$ for the TPM quote and $n2$ of the μTPM quote. For each attestation request, the untrusted OS uses $n1$ to produce the TPM quote over the TrustVisor PCRs (PCR 17 and PCR 18), which contain the measurements of TrustVisor. The one-to-many FSA protocol can be implemented alongside this original protocol so that verifiers can use either protocol. To support the FSA protocol, the untrusted OS creates a single TPM quote $quote_{FSA}$ over the TrustVisor PCRs and the FSA PCR. Since the required information has been extended into the FSA PCR, the user-supplied data field in the quote is not used by the FSA protocol. The untrusted OS can use this field to achieve channel-binding since it is the end-point of the secure channel. The untrusted OS retrieves the FSA info, which would have been contained in $cert_{FSA}$, using the `HV_FSA_Info` operation. When the untrusted OS receives an attestation request, it passes the verifier's nonce $n1$ to TrustVisor using the `HV_FSA_Sign` operation, which returns the signature $E(\mathcal{K}_{FSA-}, n1)$. The untrusted OS sends the verifier $quote_{FSA}$ and the relevant AIK certificate, $\mathcal{K}_{FSA+}$ and its validity period, and the signature $E(\mathcal{K}_{FSA-}, n1)$. The verifier uses $quote_{FSA}$ to check that TrustVisor was in its final state. It checks that

$\mathcal{K}_{FSA+}$ is within its validity period and that this key and validity period were the first items extended into the FSA PCR, followed by the well-known value. This convinces the verifier that $\mathcal{K}_{FSA-}$ is held by TrustVisor and not any other software on the platform. Finally, the verifier checks the signature on nonce $n1$ to confirm that TrustVisor still has access to $\mathcal{K}_{FSA-}$. If this succeeds, the verifier is convinced that TrustVisor was previously in a trustworthy final state and that the platform has not been reset since then. Therefore the system must still be in the trustworthy final state represented by the quote.

Benchmarks by McCune et al. [182] show that when TrustVisor is used to protect the security-sensitive portions of an SSH server, it takes approximately ten times longer to connect to the server since the connection involves a TPM quote. However, if the FSA protocol were used, the time-consuming `TPM_Quote` operation would not be required for each attestation, thus significantly reducing the attestation latency (although there would still be some additional overhead due to the additional transfer of attestation information). This adaptation of the FSA protocol for TrustVisor demonstrates how this protocol can be used in other types of systems, even if the TCB is not the end-point of a secure communication channel.

7.5 Evaluation

Two main approaches have been used to evaluate the security and performance of the FSA protocol. Section 7.5.1 describes how the security properties were evaluated using formal methods and automated analysis. Section 7.5.2 presents the performance evaluation which consists of theoretical comparisons to other protocols and evaluation of a prototype implementation.

7.5.1 Formal Analysis of Security Properties

Bai et al. [26] have developed *TrustFound*[9], a framework for modelling and analysing trusted computing platforms using the Process Analysis Toolkit (PAT) [248]. PAT is capable of analysing models in a variety of different languages including CSP#, an extension of CSP (introduced in Chapter 4) that allows interaction with libraries written in C#. TrustFound introduces the Trusted CSP# (TCSP#) formalism which extends CSP/CSP# to include expressions for security-relevant functionality such as encryption, signatures and hash functions. TrustFound also includes a formal model of a TPM for use in TCSP# models. Similarly to the Casper-Privacy tool presented in Chapter 4, TrustFound models the security properties as reachability assertions in the symbolic paradigm. Although other frameworks and tools can be used to model trusted systems (e.g. AVISPA as used by Stumpf et al. [247]), these do not include a model of the TPM. This section describes how the TrustFound framework has been used to model and analyse the security properties of the FSA protocol.

[9] `http://www.comp.nus.edu.sg/~a0091939/TrustFound/`

In order to analyse the security properties of different attestation protocols, a model of a common attestation scenario has been created using TrustFound. Figure 7.5 gives an overview of this scenario and the full TCSP# model is shown in Listing B.1 in Appendix B. As shown in Figure 7.5, this scenario consists of a verifier who communicates with a prover over the network. The objective is for the verifier to use remote attestation to determine if the prover is in a trustworthy state and, if so, to send the prover a secret value. This could correspond to a relying party sending a piece of private information to the TRE. The prover is modelled as a sequence of three processes, each of which measures the subsequent process and extends it into the TPM to achieve a measured boot. For clarity, these processes are referred to as the platform's firmware, BIOS, and software. In this abstraction, the firmware is immutable and uncompromised but the adversary can choose to load compromised or uncompromised versions of the BIOS and software. In reality, this chain could contain more processes (e.g. the boot loader) or different processes (e.g. UEFI or a DRTM late launch). However, since the aim is to determine whether the final process is uncompromised and whether any of the preceding processes in the chain had been compromised, a chain of length three, beginning with an immutable uncompromised process, is sufficient to capture all possible permutations.

In order to compare different attestation protocols, the protocols themselves are modelled as interchangeable components for the common scenario. For comparison purposes two previous attestation protocols have been modelled: a one-to-one nonce-challenge protocol, similar to that presented by Sailer et al. [232] (Listing B.2 in Appendix B), and a one-to-many protocol based on timestamps, similar to that by Moyer et al. [195] (Listing B.3). The model of the FSA protocol is shown in Listing B.4. Since these are symbolic models, DH key exchanges cannot be modelled directly as this requires support for algebraic equivalences, which is not available in this tool. Therefore, key agreement between participants is simulated using a *key oracle* process in this analysis.

The adversary is modelled as a single process that interacts with both the prover and the verifier. To facilitate detailed analysis, each of the adversary's capabilities can be individually enabled or disabled. Conceptually, four different combinations of capabilities are considered in this analysis, since these correspond to different types of realistic adversaries. These are strictly ordered in that each combination includes all the capabilities of the previous combination. In order of increasing strength, the following combinations of adversary capabilities are considered:

1. **Passive adversary:** A very limited adversary who can only read messages from the network and cannot modify messages or send falsified messages.

2. **Network adversary:** A standard DY adversary who controls all communication links between the verifier and the prover. In addition to intercepting all communication, this adversary can modify network messages and send falsified messages.

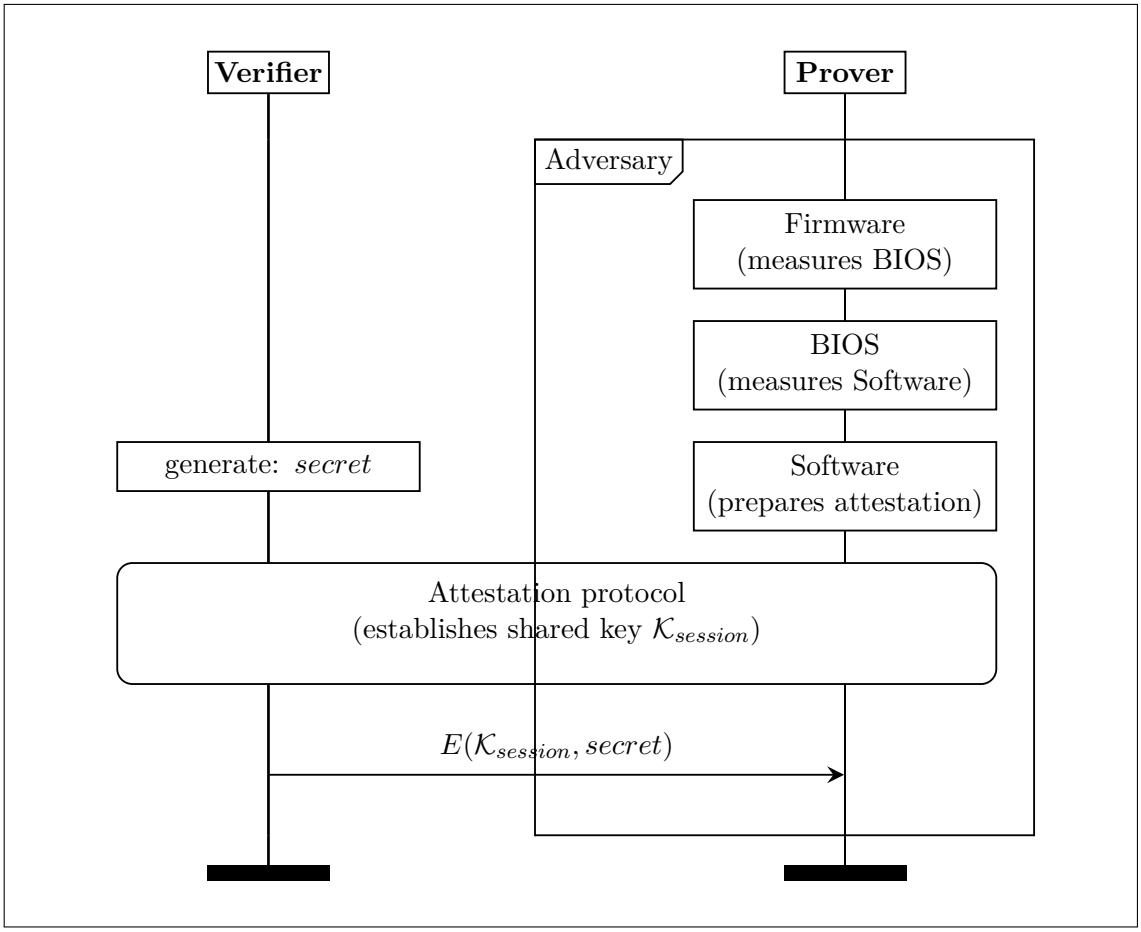

Figure 7.5: Common scenario for remote attestation protocol analysis

3. **Physical adversary:** This is the attestation adversary described in Section 7.1 or the adversarial operator of the TRE as described in the previous chapter. Since this adversary has physical access to the prover's platform, it controls all communication links to the prover and thus has all the capabilities of the network adversary. The physical adversary can also reset the platform at any time and can load compromised software.

4. **Advanced adversary:** In addition to the capabilities of the physical adversary, the advanced adversary can exploit runtime vulnerabilities in any software executing on the prover's platform. It can also reset the prover's TPM without resetting the platform [149].

As explained in Section 7.1, the capabilities of the advanced adversary are beyond the scope of this research. If the adversary can exploit runtime vulnerabilities in the prover's software, the prover's state will not be accurately represented by the PCR values. This is a common problem in TPM-based systems referred to as the *time-of-check, time-of-use* ($TOCTOU$) problem. However, since this is a deficiency of the measurement process, it cannot be mitigated through the attestation protocol. If the adversary can reset the

prover's TPM without resetting the platform, the adversary can simply load compromised software on the platform, reset the TPM, and extend the correct values into the PCRs. Again this is a deficiency in the measurement process since the measured PCR state does not represent the actual state of the platform. Since the TPM is not designed to be reset without resetting the platform, this must be considered a sophisticated hardware attack. The advanced adversary is included in each of the analysed protocols as a checking mechanism for the TCSP# adversary model. If the advanced adversary were not able to obtain the verifier's secret, this would indicate a problem with the model. As expected, the analyses show that the advanced adversary can obtain the verifier's secret in all three protocol models.

Two reachability assertions are used to check the relevant properties of each protocol. The first assertion is that it is possible for the prover to receive the verifier's secret value, thus confirming that the protocol can actually proceed to completion. The second assertion is that the adversary is unable to learn the verifier's secret value under any circumstances. By the construction of the model, this second assertion embodies both security requirements defined in Section 7.1, since failure to fulfil either requirement would allow the adversary to learn the secret value. The main analysis results for each protocol model, with respect to the different types of adversaries, are presented in the following subsections.

Nonce-Challenge Protocol

In this model there are three mitigation techniques that can be individually enabled: binding the quote to the prover's public key, binding the quote to the session key, and/or using a secure channel that provides Perfect Forward Secrecy (PFS). In all cases, the analysis confirms that the first assertion is valid and the protocol proceeds to completion.
Passive adversary: It is assumed that the adversary does not initially know the prover's long-term key pair and thus, even with all mitigation strategies disabled, this protocol is secure against the passive adversary.
Network adversary: This adversary can perform a masquerading attack by forwarding the verifier's nonce to a trustworthy prover and forwarding the quote to the verifier. Since the prover is in a trustworthy state, the verifier mistakenly believes that the adversary is trustworthy and sends it the secret value. This masquerading attack can be prevented by binding the quote to the prover's public key. In the model, this is achieved by extending the public key into the PCRs but it could also be achieved by including it as part of the nonce in the quote. The inclusion of the verifier's nonce in each quote prevents replay attacks.
Physical adversary: The analysis shows that a physical adversary can perform the following reset attack. When a verifier requests a quote, the protocol proceeds as usual until the trustworthy prover has created a quote using the correct nonce and sent it to the verifier. The adversary then resets the prover's platform and loads compromised software,

giving the adversary access to the prover's long-term key pair (assuming this is not sealed to a particular PCR value). The adversary then uses this long-term key to re-establish the communication channel with the verifier. If the verifier is configured to require a new quote for each handshake, the adversary can instead use the long-term key to recover the previous session key. This attack is also possible if the quote is bound to the session key instead of the long-term key. In all cases, the verifier accepts the quote and sends the secret to the compromised prover, thus leaking it to the adversary.

The analysis shows that this can be prevented by binding the quote to the session key and using a channel that provides PFS. Since the quote is bound to the session key, the verifier must always require a new quote whenever the session key changes. Even after gaining access to the prover's long-term keys, the adversary cannot recover a previous session key due to PFS. Furthermore, the model shows that even if the adversary has access to the long-term key *before* the protocol is run, the protocol is still secure. This differs from identity-based trust relationships in which the adversary would be able to use the long-term key to masquerade as the prover. This can be explained by recognizing that only the trustworthy prover can create a trustworthy quote and that the quote itself authenticates the session key being used by this prover. To obtain this quote, the adversary must establish a session key with the trustworthy prover. Since the adversary cannot extract the prover's ephemerally-generated DH private key, the adversary must use its own DH private key to establish a session with the verifier. The session key between the adversary and the verifier will therefore always be different from the session key between the adversary and the prover and thus the verifier will be able to detect this attack. This demonstrates that the prover's long-term keys are actually redundant for the purposes of establishing attestation-based trust relationships. Overall, this model confirms that the nonce-challenge protocol fulfils the security requirements defined in Section 7.1.

Global Timestamp Protocol

Since this is a one-to-many protocol it is not possible to bind the quote to individual session keys. This model therefore only includes two possible mitigation techniques: binding the quote to the prover's long-term public key and/or using PFS for the secure channels.

Passive adversary: Even with all mitigation techniques disabled, this protocol is secure against the passive adversary.

Network adversary: Against a network adversary, this protocol is vulnerable to the same type of masquerading attack as described for the nonce-challenge protocol above. As before, this attack can be prevented by binding the quote to the prover's public key. In this protocol, the currentness of the attestation is established through the inclusion of a globally-trusted timestamp in each quote.

Physical adversary: The analysis shows that a physical adversary can perform the following reset attack. The trustworthy prover obtains a timestamp and generates a quote bound to its long-term public key. The adversary requests this quote and receives it

from the prover (since the adversary may also be a participant in the protocol). The adversary then resets the prover and gains access to its long-term keys and can thus masquerade as a trustworthy prover by replaying this quote and using the prover's long-term keys, thus compromising **AR-2**. The prover might attempt to issue quotes more frequently so that the adversary does not have time to perform this attack. However, the adversary may have already obtained the prover's long-term keys and reset the prover back into a trustworthy state, thus undermining the authenticity guarantee. These keys could potentially be protected by sealing them to the trustworthy prover's PCR values but this would require additional mechanisms to convince the verifier that this guarantee is in place (e.g. those described by Armknecht et al. [18]). Since this is a one-to-many protocol, there is no possibility to bind the quote to the individual session keys as with the nonce-challenge protocol and thus the use of PFS in these channels cannot prevent this attack. Therefore, this protocol (as modelled in Listing B.3) does not fulfil the security requirements defined in Section 7.1.

Final State Attestation Protocol

By definition, the FSA protocol always binds the quote to the prover's public key ($\mathcal{K}_{FSA+}$) by including the hash of this key's certificate in the quote. Therefore, this model provides two mitigation techniques: using secure channels that provide PFS and using an ephemerally-generated public key that is irrecoverably lost if the prover is reset.

Passive adversary: Even with all mitigation techniques disabled, this protocol is secure against the passive adversary.

Network adversary: Against a network adversary, this protocol is not vulnerable to masquerading attacks because it already binds the quote to the prover's public key. In this protocol, currentness (**AR-2**) is established through the prover's assertion that it has reached its final state. The network adversary does not have sufficient capabilities to reset the prover's platform and force it to leave this final state.

Physical adversary: The model shows that once the prover has reached its final state, the physical adversary can request a quote and then reset the prover's platform. The adversary can load compromised software that uses the prover's long term public key and replays this quote to all verifiers, thus undermining the authenticity and currentness guarantees. However, if the prover's main key pair (i.e. its public and private keys) is generated ephemerally and stored in volatile memory (as required by the FSA protocol), it will be irrecoverably lost if the prover's platform is reset. This prevents the adversary from extracting the private key through a reset attack as was possible in the previous protocols. Since all quotes are bound to the prover's public key, resetting the prover automatically invalidates all previous quotes and thus prevents replay attacks. Without the prover's private key, the adversary also cannot recover any of the session keys from the secure channels between the prover and verifier, even if PFS is not used, thus ensuring authenticity. If a verifier accepts a quote, it means that the prover still has access to the

key included in the quote and thus the prover has not been reset since the quote was created. Therefore, due to the final state assertion made by the prover, the verifier can be sure that the quote represents the current and authentic state of the prover.

Overall, the analysis of the nonce-challenge and timestamp-based attestation protocols shows that the TCSP# adversary model is a correct representation of the respective adversary capabilities and that these capabilities are sufficient to subvert those protocols unless specific mitigation techniques are used. The analysis of the FSA protocol with respect to the same physical adversary has confirmed that this adversary is not able to subvert the FSA protocol.

7.5.2 Performance Evaluation

Latency

As a one-to-many protocol, the FSA protocol allows the prover to use a single RTR operation (e.g. TPM quote) to support multiple verifiers. Furthermore, during the attestation, this protocol does not require the prover to perform any additional cryptographic operations beyond those already required to establish the secure channel. The fact that this channel is established using $\mathcal{K}_{FSA}$ provides the required authenticity and currentness guarantees to the verifier. In contrast to the proposals by Stumpf et al. [247], the FSA protocol does not require any additional messages exchanges (e.g. key confirmation) since all attestation information can be included in an existing message through the FSA extension. From the prover's perspective, the only latency added by the FSA protocol, compared to a secure channel without attestation, is the time required to send the FSA extension. Since this extension is in the order of 924 bytes (assuming 256 bit ECC keys), this overhead is minimal. When the TPM is used as the RTR, this latency is therefore significantly less than any of the one-to-one protocols, including the widely-used nonce-challenge protocol.

From the verifier's perspective, a single attestation operation requires only three signature verification operations: verifying the signature on the AIK certificate, using the AIK to verify the signature in the quote, and using $\mathcal{K}_{FSA}$ to verify the signature in the handshake. The third signature verification is always required to establish the secure channel, even if attestation is not used. In terms of latency, the closest competitors to the FSA protocol are the protocols by Gasmi et al. [109] and Armknecht et al. [18]. Both protocols require the verifier to perform four signature operations: verifying the signature on the AIK certificate, verifying the AIK signature on $\mathcal{K}_{bind}$, verifying the $\mathcal{K}_{bind}$ signature on $\mathcal{K}_{sign}$, and using $\mathcal{K}_{sign}$ to verify the signature in the handshake. Therefore, in the general case, the FSA protocol requires one fewer signature verification operation. On a multi-core CPU this is a negligible difference in latency, but if the verifier is an embedded system, such as a smart meter, any additional signature verification operations are likely to have a more noticeable effect. In the special case of a verifier re-attesting the same prover (assuming no platform resets), the latency of these two protocols becomes equivalent because the verifier need only check the final handshake signature.

Scalability

The scalability of the FSA protocol has been evaluated by implementing this protocol as part of the TRE prototype described in the previous chapter. The emulated smart meters with which the TRE communicates were augmented to verify the attestation. Following the DLMS-COSEM protocol, the smart meters are the servers and the TRE is the client. This is the reverse of the scenario shown in Figure 7.3, but demonstrates that the FSA protocol can be used by both clients and servers. The DLMS-COSEM benchmark shown in Figure 6.7 was repeated for the x86-TPM TRE with the FSA protocol enabled. As expected, the scalability of the TRE was virtually unchanged, achieving the same rate of approximately 24.7 DLMS-COSEM request-response operations per second. In comparison, using the one-to-one nonce-challenge attestation protocol on the same platform would limit this rate to 1.4 operations per second.

The TPM 1.2 is known to have very high latencies for cryptographic operations but this is likely to improve in the next generation of attestation technologies. For example, a TPM 2.0 implemented in firmware would offer significantly faster cryptographic performance. Similarly, Intel SGX will use the main CPU to create quotes. However, even in these cases, it is still possible that the RTR will introduce some additional latency. For example, in SGX the creation of a quote requires an intra-platform attestation between the prover and the quoting enclave and at least two context switches between these enclaves. Although these overheads may be small, they can be completely eliminated by using a single quote for multiple verifiers and thus it may be possible to further improve the performance of these next-generation attestation technologies by using the FSA protocol.

7.6 Verifying the Attestation

When evaluating a system that makes use of remote attestation, it is also important to consider the process of verifying the attestation performed by the verifier. In practice, the attestation will be verified by some type of computer system on behalf of the user and thus the term *verifier* encompasses both the user and this verifying system. For the overall process to be meaningful, the user must trust that the verifying system itself is operating correctly. Compared to a remote system, it is usually less difficult for the user to establish the trustworthiness of a local system since the user has physical access to and control over this system. However, experience has shown that a local verifying system may still be vulnerable to malware or runtime attacks. It is therefore critical to ensure that the verification of the attestation cannot be undermined by these vulnerabilities.

In the context of the smart grid, the verifying system could be a smart meter or Home Energy Management System (HEMS). This system is required to establish a secure communication channel to the TRE and verify the TRE's remote attestation. In this context, the verifying system is also required to authenticate itself to the TRE. To prevent private data being sent to an untrusted entity, the user requires assurance that the software

performing the verification operations (e.g. the cryptographic library) has not been compromised and that the correct root key is being used (e.g. to verify the $cert_{AIK}$). The user also requires assurance that the authentication key pair, which uniquely identifies the user, is protected from any malware or compromised software on the verifier's platform, and that this key will only be used for communicating with trustworthy entities. Although there are various approaches for providing this type of assurance, the smart grid context imposes an additional constraint in that the verifying system must be able to operate without requiring user interaction. A widely-used method for protecting cryptographic keys on PCs is to encrypt them using the user's password, but for a smart meter or HEMS, the user cannot be expected to enter a password every time the key is required.

Trusted Computing could be used to protect the verification process in this context. A naive approach would be to simply seal the authentication key to a specific state of the verifying system. This means that if the verification algorithms or root key were modified, the authentication key would become inaccessible and the compromised verifying system would not be able to communicate with the TRE. The main disadvantages of this approach are that, unlike the TRE, the verifying system is likely to have a large, frequently changing software base, which makes sealing less practical and exposes the key to more security vulnerabilities because of the large TCB. A better approach is to use a Trusted Execution Environment (TEE) to isolate the authentication key from the rest of the software on the system. A proof-of-concept implementation of this type of TEE-based mechanism has been developed [206]. Although this implementation has been designed to protect the verification steps and authentication key in a standard TLS handshake, it could easily be adapted for use with the FSA protocol.

The TEE in this implementation is provided by the Flicker framework by McCune et al. [181], but other types of TEE, such as ARM TrustZone, could also be used [197]. In this implementation, the authentication key is generated within a Flicker PAL and sealed to the state of that PAL using the TPM. The software outside the PAL can therefore be changed without affecting the security of the PAL.

The enhanced handshake protocol is shown in Figure 7.6. In this figure, the mechanism is used on the client side but it could also be used on the server side. The TLS handshake proceeds as usual until the client is required to sign the handshake digest. The client's cryptographic library outside the PAL has been modified to invoke the PAL at this point and input the preceding handshake messages. The PAL parses these messages and uses its own signature verification algorithm and root certificate to check the server certificate. If the FSA protocol were used, the PAL could instead verify the AIK certificate and $cert_{FSA}$ which would have been sent in the server hello message. If this verification step succeeds, the PAL computes the digest of the handshake messages, signs this with the authentication key, and returns it to the client. The TLS handshake then completes as usual.

This mechanism protects the user's authentication key by ensuring that this key never leaves the PAL unencrypted. It also protects the verification process by sealing the authentication key to a specific root certificate and set of software algorithms in the PAL.

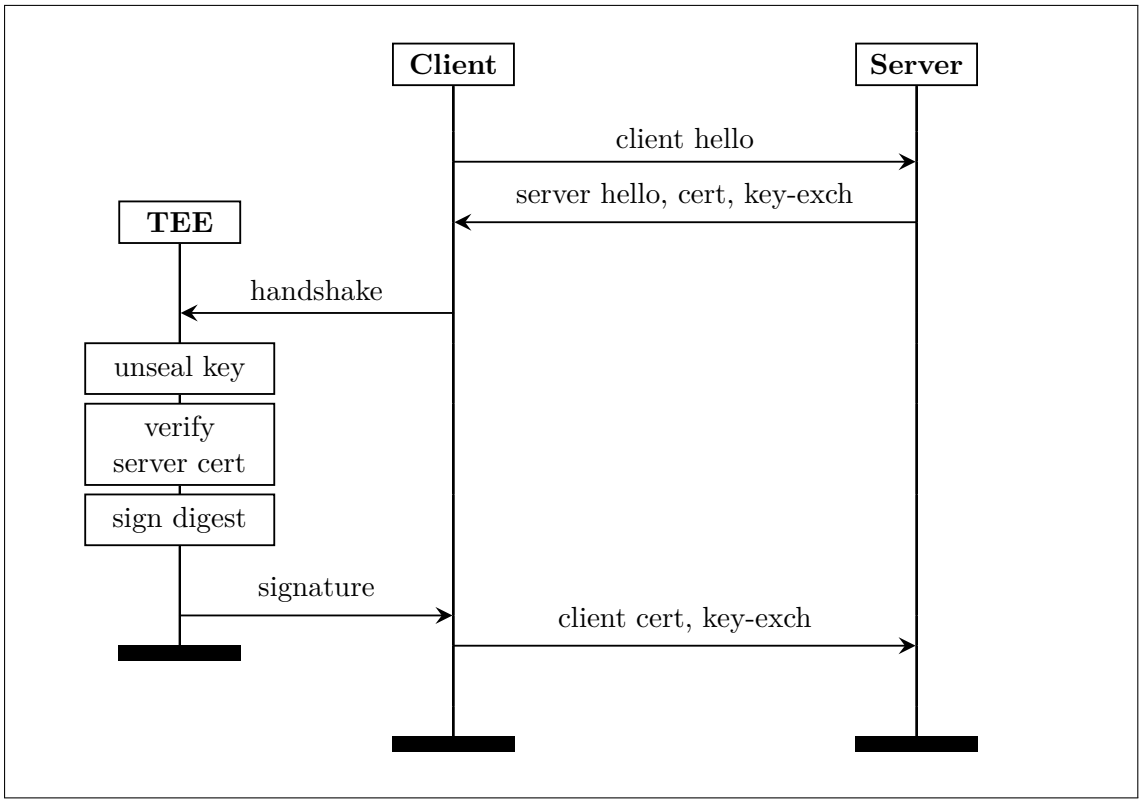

Figure 7.6: TEE-based mechanism for protecting an authentication key

The PAL itself has a minimal software TCB consisting of the signature creation and verification algorithms and the root verification key. As with the TRE, this minimal TCB reduces the probability of security vulnerabilities and is more amenable to formal verification or security audits. If the adversary modifies the verification algorithms or substitutes the root key, the authentication key cannot be unsealed and thus the compromised verifier cannot complete the attestation protocol. Although the adversary may be able to obtain private information from the verifier through other attacks, the adversary cannot subvert the verification of the attestation protocol. Since the TPM is used as the Root of Trust for Storage (RTS), this mechanism does not require any user interaction and can thus be used in contexts such as the smart grid. Using this type of protection mechanism and a secure remote attestation protocol, such as the FSA protocol, the user is assured that the received measurements are an accurate representation of the prover's current state. Although the actual process through which the user makes trust decisions is beyond the scope of this research, the mechanisms and protocols presented in this chapter provide the user with all the information necessary to make well-informed trust decisions.

7.7 Summary

This chapter has presented an in-depth analysis of remote attestation in the context of the TRE. Remote attestation is the primary mechanism through which relying parties establish the trustworthiness of the TRE. Fundamentally, remote attestation is the process of conveying sufficient information about a remote prover to a verifier in order for the verifier to make a well-informed trust decision. The attestation adversary aims to subvert this process to gain the verifier's trust under false pretences. This adversary has similar capabilities to the adversarial TRE operator described in the previous chapter in that it controls all network communication between the verifiers and the prover and has physical access to the prover, which allows it to reset the prover's platform and load compromised software.

The overall remote attestation mechanism consists of a measurement process (as described in the previous chapter) and a protocol for communicating the measurements to verifiers. Assuming a secure measurement process is used, the two security requirements for the attestation protocol are that the verifier be able to determine that the measurements are an authentic representation of the entity with which the verifier is communicating and that they represent the current state of this entity. In addition to these security requirements, the protocol must also meet the performance requirements in terms of attestation latency and scalability for the system in which it is used. These performance requirements are particularly important for the TRE since it is designed to establish attestation-based trust relationships with all participants.

Early TPM remote attestation protocols, such as the widely-used nonce-challenge protocol, are one-to-one in the sense that they required a separate RTR operation for each verifier. However, the performance of these protocols is severely limited by the time taken to perform this RTR operation. Benchmarks of a TPM 1.2 showed that it requires approximately $731ms$ to generate a single TPM quote and this operation usually cannot be parallelized. To overcome these performance limitations, various one-to-many protocols have been proposed in which a single quote can be used for multiple verifiers. Since these protocols do not have the implicit currentness and channel-binding guarantees of one-to-one protocols, they rely on techniques such as aggregating multiple hashes, using trusted timestamps, or using non-migratable TPM keys.

This chapter has presented a new highly-scalable one-to-many attestation protocol specifically designed for the TRE. The main philosophy of this Final State Attestation (FSA) protocol is that the prover must reach a final state and provide a guarantee that it will not leave this state voluntarily. Once the verifier has determined that the prover was in this final state at some point in the past, the verifier need only determine that the prover's state has not subsequently been involuntarily changed due to a platform reset. To achieve this, the FSA protocol binds the quote to a new key pair that is ephemerally-generated by the prover and stored in volatile memory such that if the prover is reset, the private key is irrecoverably lost. The prover also generates the necessary certificates for

this key pair and uses these to establish secure communication channels with each verifier. If the channel establishment succeeds, the verifier can be sure that the prover is in the state represented by the quote.

A formal model of the FSA protocol has been created and analysed using the Trust-Found framework. For comparison purposes, a one-to-one nonce-challenge protocol and a one-to-many protocol based on global timestamps were also analysed in the same scenario. Although the nonce-challenge protocol fulfils the security requirements, its performance limitations make it unsuitable for use in the TRE. The global timestamp protocol does not meet the security requirements due to its lack of channel-binding. The analysis showed that the FSA protocol does not suffer from this vulnerability and therefore meets the security requirements with respect to the defined adversary model. Analysis of the performance requirements showed that the FSA protocol generally requires fewer signature verification operations than the protocols using non-migratable TPM keys. The FSA protocol also provides better defence-in-depth by limiting the validity period of the attestation credentials. The scalability of the FSA protocol has been evaluated by repeating the DLMS-COSEM benchmark from the previous chapter. This showed that the FSA protocol adds virtually no overhead and allows the TRE to maintain a rate of 24.7 DLMS-COSEM request-response operations per second. This is a significant improvement over the widely-used nonce-challenge protocol which can achieve a maximum of 1.4 operations per second in the same scenario.

Finally, a mechanism has been presented for protecting the verification of the attestation on the verifier's system. A proof-of-concept implementation of this mechanism, using the Flicker framework, showed how the verifier's authentication key can be sealed to a specific set of algorithms and a root key. If the adversary modifies these algorithms or substitutes the root key, the authentication key becomes unavailable, thus preventing a compromised verifier from successfully completing the attestation protocol with a trustworthy prover.

Overall, the mechanisms and protocols presented in this chapter provide the verifier with all the information necessary to make well-informed trust decisions about systems like the TRE whilst meeting the performance requirements of these systems. This chapter therefore complements the previous chapter in showing how the TRE can be realized, thus confirming the first primary research hypothesis.

Chapter 8

Formalization and Application to Other Domains

This chapter draws on research described in the following publications:

- A. J. Paverd, "Applications of a Trustworthy Remote Entity (Abstract)" *University of Oxford Department of Computer Science Student Conference*, 2015.
- A. J. Paverd, "Poster: Enhancing Privacy in Location-Based Services using Trustworthy Remote Entities," Presented at the *7th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec'14)*, 2014.

8.1 Framework for Enhancing Communication Privacy using the TRE 190
8.2 Location-Based Services 195
8.3 Wireless Network Roaming 200
8.4 Secure Multiparty Computation using the TRE 204
8.5 Summary 208

The preceding chapters have demonstrated how the TRE can be realized and used to enhance communication privacy in the smart energy grid. This chapter formalizes the TRE concept and demonstrates its applicability to other application domains. The formalization of the TRE concept takes the form of an application-independent framework that describes the fundamental functionality provided by the TRE as an architectural building block for enhancing communication privacy. Using this framework, two smaller case studies are investigated: enhancing communication privacy in location-based services (LBS) and in wireless network roaming. Finally, this chapter presents a discussion of why the TRE is an effective and efficient mechanism for achieving secure multiparty computation (SMC).

8.1 Framework for Enhancing Communication Privacy using the TRE

In order to use the TRE to enhance communication privacy in a specific scenario, three broad aspects must be considered: the types of participants and adversaries, the form of communication privacy required, and the functionality to be provided by the TRE. This framework provides the underlying structure and tools for reasoning about each of these three aspects. Although the TRE may be used for various other purposes, this framework focuses solely on scenarios in which the TRE is used to enhance communication privacy. Since this framework only deals with the application layer of the scenario, all lower layer components (e.g. the participants' software implementations) are assumed to be ideal.

8.1.1 Participants and Adversaries

The TRE is always used in an environment in which there are multiple entities. Figure 8.1 is a graphical representation of the different types of entities relevant to this framework. The communication protocol defines the exchange of information between entities[1]. Some of these entities are subjects who generate and/or own information that they consider private (e.g. the consumers in the smart grid protocols). The set of all subjects is denoted $\mathcal{S}$. The set $\mathcal{D}$ is the set of entities who are willing to deviate from this protocol. The set $\mathcal{N}$ is the set of entities who have full control over the portion of the communication network used in this protocol (e.g. network service providers). As shown in Figure 8.1, there are overlaps between these various sets, which represent either entities with multiple capabilities or collusion between entities. Region 1 represents subjects who are also willing to deviate from the protocol. Region 2 represents subjects who are willing to deviate from the protocol and have full control of the network (if such an entity were adversarial, it could be represented as the classic Dolev-Yao adversary). Entities in region 3 do not have legitimate subject credentials whilst those in region 4 are not willing to deviate from the protocol. The set of honest-but-curious (HBC) entities $\mathcal{H}$ is strictly disjoint from

[1] In reality, there may be multiple communication protocols in use. If these protocols are interconnected, they can be represented as a single large protocol. If they are independent, each can be considered individually.

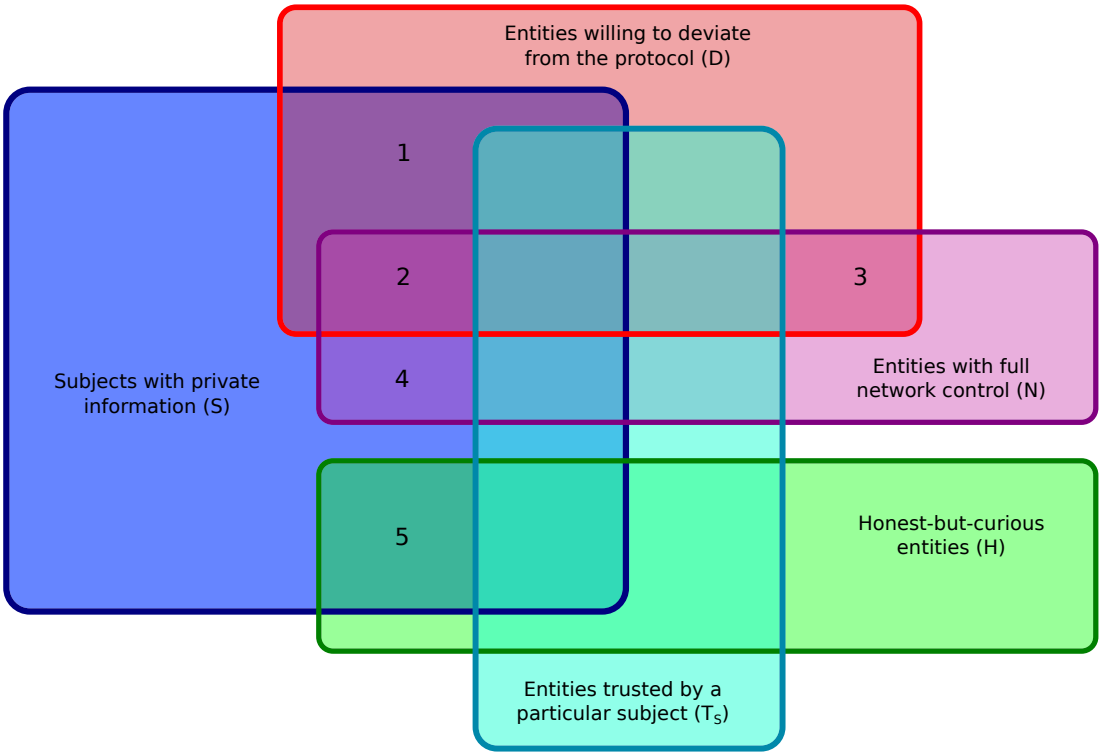

Figure 8.1: Types of entities in the TRE formalization

sets $\mathcal{D}$ and $\mathcal{N}$ since the HBC entities will not deviate from the protocol and will not receive messages for which they are not the intended recipients, as explained in Chapter 4. Some subjects may also be honest-but-curious (region 5). Each entity may trust a set of other entities to protect certain information or to perform certain actions. For example, subject $s \in \mathcal{S}$ may trust the set of entities $\mathcal{T}_s$ to store or process the subject's private information. From a particular entity's perspective, all other entities are untrusted and are thus considered to be adversarial. As shown in the figure, any type of entity may be trusted.

The communication protocol may involve any number of entities of any type. Entities who may legitimately participate in the protocol constitute the set of participants $\mathcal{P}$. The subject's communication privacy is compromised if any adversary can learn or infer the subject's private information and definitively link this information to the subject as a result of the communication protocol. As explained at the start of Chapter 1, communication privacy is threatened by two types of adversaries: internal adversaries, who are legitimate participants in the protocol (i.e. entities within $\mathcal{P}$), and external adversaries, who are not meant to participate in the protocol. In general, the privacy threat from external adversaries can be mitigated by performing the protocol over secure communication channels, which provide confidentiality and authentication. At the application layer, the main threat to communication privacy arises from internal adversaries (i.e. $\mathcal{P} \setminus \mathcal{T}_s$). If the protocol requires the subject's private information to be communicated to an untrusted

participant, it may be possible to preserve the subject's privacy by using an anonymous communication channel. However, if for any reason the protocol requires that an untrusted participant must be able to link this private information to the subject, it would not be possible to use an anonymous channel. In this case, if the protocol can be modified appropriately, the TRE can be used to ensure the subject's privacy.

8.1.2 Forms of Communication Privacy

The second aspect of the framework considers the form of privacy applicable to a given scenario. As defined in Chapter 1, communication privacy is the ability of individuals to control what personal information related to them may be collected or inferred as a result of communication. Fundamentally, there are two possible forms of communication privacy: either the communicated information does not compromise the individual's privacy requirements or the information cannot be associated with the individual. In this context, the first form is referred to as *information privacy* whilst the second is termed *identity privacy*.

Information privacy usually involves modifying the communicated information or computing some quantity derived from this information such that the result is still useful to the recipient but does not reveal the subject's private information. The subject's identity is still connected to this information. For example, in the enhanced smart grid architecture proposed in Chapter 5, the network monitoring and billing protocols provide information privacy with respect to the energy supplier and DNO. In general, some information about the subject is still revealed to the untrusted party, but ideally this should not diminish the subject's privacy. For example, in the smart grid billing protocol, the energy supplier still learns the consumer's total bill for each billing period, but as explained in Chapter 3, it is generally considered that this does not diminish consumers' privacy. The metrics by which information privacy is measured are specific to the type of information being protected. When analysing communication protocols, information privacy can be verified through properties such as undetectability of private information items, as described in Chapter 4. The objective of Secure Multiparty Computation (SMC), as described in Chapter 2, is to ensure information privacy by securely computing some function on the subjects' private inputs without revealing these inputs to any untrusted participants. The subjects' identities must be known and some type of authentication must be used in order to prevent external adversaries from participating in the computation.

In contrast, identity privacy or anonymity focusses on dissociating the subject's identity from the information. Even without modifying the communicated information, communication privacy can still be achieved if the information can no longer be linked to the individual. For example, the new demand bidding protocol presented in Chapter 5 leaves the content of the bid unchanged but dissociates the identity of the bidder, thus ensuring identity privacy. Identity privacy is generally measured in terms of the size of the anonymity set [212], which is the set of all participants to whom the information could

possibly be linked. When analysing communication protocols, identity privacy can be verified through properties such as unlinkability between individuals and information items, as described in Chapter 4. Even though the subject's identity may be hidden, there is still a requirement to ensure that the information originates from the correct subject and not an adversary. This has led to techniques such as private authentication [103, 5].

8.1.3 TRE Functionality

The final aspect of the framework is to consider the functionality that can be provided by the TRE to achieve the desired form of communication privacy with respect to the relevant participants and adversaries. The TRE can be used to ensure either information privacy, identity privacy, or some combination thereof. In general, the TRE performs actions that cannot be performed by the subject.

Information Privacy

To ensure information privacy, the TRE can compute any function on multiple private inputs. These inputs can either be from multiple subjects or from the same subject (e.g. over a period of time). For example, the TRE can perform the following types of privacy-enhancing operations:

- **Differentially-private aggregation over multiple subjects:** As demonstrated by the network monitoring protocol in Chapter 5, the TRE can receive private information from multiple subjects and answer statistical queries about this information (e.g. computing the aggregate). A differentially private query mechanism can be used to ensure that no subject's individual input can be inferred from the results. Since the information is centralized, the TRE can add the minimum amount of noise to the result. The TRE can prevent external adversaries from participating by authenticating all participants. Depending on the context, the TRE can also perform bounds-checking on inputs to prevent attacks from adversarial subjects (i.e. regions 1 and 2 in Figure 8.1).

- **Differentially-private temporal aggregation:** In certain scenarios, aggregation over multiple subjects does not produce a useful result and therefore cannot be used. For example, most scenarios involving billing cannot be aggregated over multiple individuals since each individual must receive a separate bill. In these circumstances, the TRE can still ensure information privacy by performing temporal aggregation over multiple inputs from the same subject, as demonstrated by the billing protocol in Chapter 5. The TRE can answer queries about this consolidated information (e.g. computing the total) in a differentially private manner such that no individual input can be inferred. Again the TRE can authenticate all participants and perform bounds-checking on the inputs.

Identity Privacy

The TRE can also ensure identity privacy by hiding the link between the subject's identity and the communicated information. Since the TRE can maintain persistent state across multiple protocol runs, this link can be stored by the TRE and used to facilitate bi-directional communication with the subject. Furthermore, if desirable for the specific scenario, the TRE can enable *asynchronous* bi-directional communication such that, once the initial communication has taken place, the other participants can asynchronously initiate communication with the subject via the TRE, without knowing the subject's real identity. For example, the TRE can perform the following types of operations to ensure identity privacy:

- **Pseudonymization of communication:** As demonstrated by the new demand bidding protocol, presented in Chapter 5, the subject can communicate with an untrusted participant and maintain identity privacy by using the TRE as an intermediary in the communication path. The TRE generates a random pseudonym for the subject and protects the link between this pseudonym and the subject's identity. The pseudonym can be refreshed as often as required (e.g. the subject can achieve anonymity by using each pseudonym only once).
- **Delegated authentication:** In addition to performing pseudonymization, the TRE can also perform delegated authentication of subjects in order to ensure identity privacy. If a particular scenario requires a subject to be authenticated, this process can be performed by the TRE, which then provides the relevant assertion if the authentication was successful. The TRE does not reveal the subject's identity at any time during this process.

One of the main advantages of the TRE is that all the functionality it provides is composable. Many real world scenarios have multiple functional requirements that usually cannot be solved with a single type of privacy-enhancing operation. For example, the demand bidding protocol composes pseudonymization with temporal aggregation in order to ensure identity privacy within the protocol and avoid leaking identifying information after the protocol has been completed. This also leads to efficiency benefits by reducing the number of messages required. For example, a single consumption value from the consumer can be used as input for both the network monitoring (spatial aggregation) and billing (temporal aggregation) protocols. Using this framework as a starting point, the following sections present two smaller case studies in which the TRE is used to enhance communication privacy in other application domains.

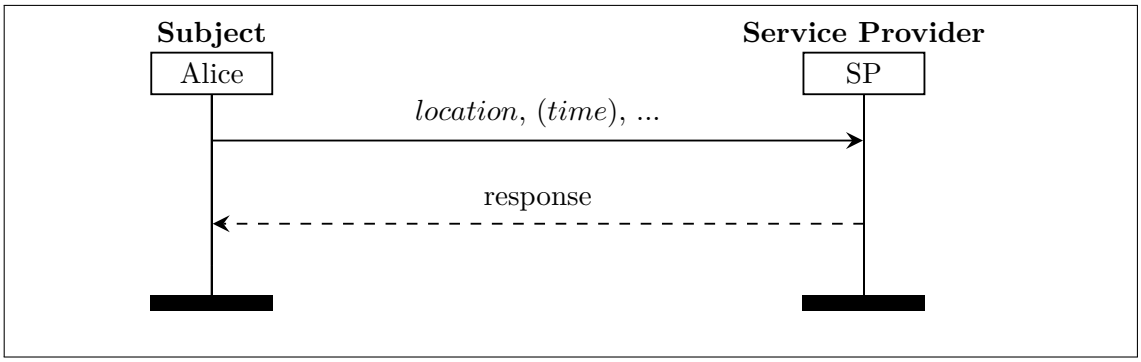

Figure 8.2: Generalized model of a location-based service (LBS)

8.2 Location-Based Services

A location-based service (LBS) refers to any exchange of information between a mobile user and a service provider that includes information about the user's location. The prevalence of location capabilities in mobile devices (GPS and network assisted GPS) has given rise to various new types of mobile applications that provide geographically relevant information or enable users to perform location-based searches (e.g. *"Find the nearest ..."*). This has also allowed mobile users to contribute to participatory sensing projects such as monitoring road conditions and traffic [188] or creating high-accuracy cellular network coverage maps [177]. For the purpose of this section, it is sufficient to consider a generalized model of a LBS as shown in Figure 8.2. In this model, a particular mobile user, *Alice*, sends a message to a *service provider* containing some form of location information. Optionally, the message can also contain information indicating the time at which Alice was at the specified location (e.g. for participatory sensing applications). In certain types of LBS, the service provider may send a *response* to Alice based on the supplied spatio-temporal information.

Despite their usefulness, it is widely acknowledged that these services introduce various privacy concerns. In this context, the strongest possible adversary, from a privacy perspective, is the service provider since this entity is the recipient of the users' messages. In terms of the framework presented above, the service provider would therefore be an internal adversary (i.e. an element of $\mathcal{P} \setminus \mathcal{T}_s$). Depending on the nature of the service and the assumed adversary knowledge, it is possible to consider two types of privacy: *location privacy* [141] and *identity privacy* [40, 83]. According to the framework, location privacy is a form of information privacy since it aims to conceal Alice's precise location from the service provider. In this case it can be assumed that the service provider already knows Alice's real identity (e.g. because she has been authenticated). On the other hand, identity privacy in this context aims to ensure that Alice is anonymous with respect to the service provider. When considering identity privacy, it is usually assumed that the messages themselves do not contain any directly identifying information, such as user names. It can also be assumed that the messages do not include any identifying network informa-

tion, such as Alice's network address, since this can be avoided by using an anonymous communication channel (e.g. an anonymity network [84, 59]). However, through the use of auxiliary information, the adversary may still be able link certain messages to particular individuals based on their location. For example, Gruteser and Grunwald [123] have explained how this may be possible when Alice is in a restricted location (e.g. a private residence), or when the adversary can match the location from the message to the location of a known user. Furthermore, if the adversary can link together multiple queries from the same user, the resulting *mobility trace* can be used to de-anonymize the user. As shown by de Montjoye et al. [193], four linked spatio-temporal points are sufficient to uniquely identify 95% of users from a dataset. Location information is therefore considered a *quasi-identifier* [40]. Therefore, in terms of the framework presented above, both information privacy and identity privacy are desirable in this context.

8.2.1 Location Cloaking

The most widely-used approach for ensuring both location privacy and information privacy is to use *spatial cloaking* and *temporal cloaking* algorithms to reduce the specificity of the spatio-temporal information received by the service provider [123, 36]. In this section, this approach is referred to as *location cloaking*. The aim of location cloaking is to hide Alice within a group of users by replacing her precise location with a larger area and/or replacing her precise time information with a time period. The resulting spatio-temporal region is referred to as the *cloaking region*. However, since the results are based on this spatio-temporal information, any reduction in specificity degrades the quality of the results.

In a naive location cloaking algorithm, Alice would select an arbitrary cloaking area and/or time period. However, there is no guarantee that this will ensure either location or identity privacy depending on the adversary's auxiliary information[2]. When ensuring location privacy, it is only assumed that the adversary does not already know Alice's precise location. The adversary may know the precise locations of multiple anonymous users. For example, the adversary could be observing user locations, without being able to identify the users. If the adversary observes that Alice's chosen cloaking region only includes one user, the adversary can conclude that this is Alice and thus learn her precise location. When considering identity privacy, the only assumption is that the adversary does not already know that the message was sent by Alice. The adversary may know that certain locations are associated with specific users (e.g. Alice's home and work addresses). In the extreme case, the adversary may know the precise locations of all users (e.g. as assumed by Kalnis et al. [144]). For example, this might be the case if the service provider colludes with the mobile network operator (MNO). Again, if Alice is the only user in the selected cloaking region, the adversary can conclude that she sent the message.

Gruteser and Grunwald [123] have proposed an approach to mitigate against these attacks by making the location cloaking algorithm k-anonymous. This is similar to the

[2] *Auxiliary information* refers to knowledge that the adversary obtains from other sources.

data privacy concept of k-anonymity [249], as described in Chapter 2, in that it requires the cloaking region to include Alice and at least $k-1$ other users who might have sent the same message. If implemented correctly, the adversary's probability of correctly identifying Alice is $\frac{1}{k}$ where k is a tunable privacy parameter selected by Alice for each message. In this context, it is not necessary to consider ℓ-diversity and t-closeness because the service provider does not aim to learn any information about Alice other than her location or the fact that she sent the anonymous message. Once the value of k has been specified, the objective is to find the smallest cloaking region that meets this requirement in order to maximize the quality of the results.

Various algorithms for determining a k-anonymous cloaking region have been proposed. Gruteser and Grunwald [123] use an approach based on *quadtrees* in which the region around Alice is repeatedly subdivided until it contains fewer than k users and then the previous iteration, which is guaranteed to contain at least k users, is used as the cloaking region. Gedik and Liu [110, 111] describe the *CliqueCloak* algorithm that uses constraint graphs to allow each user to select a different value for k. Kalnis et al. [144] propose a randomized nearest neighbour algorithm and an algorithm based on the Hilbert space-filling curve. Jensen et al. [141] provide a comprehensive summary of these and other approaches.

Many approaches require a centralized trusted third party (TTP) in order to determine the k-anonymous cloaking region [123, 110, 189, 144, 111, 20]. All users report their locations to the TTP, which is assumed to be *trusted* by all users, even though it does not provide any evidence that it is *trustworthy*. Two main disadvantages of using a TTP have been highlighted [64, 114, 40]. Firstly, since all cloaking requests are handled by this centralized entity, the TTP becomes both a performance bottleneck and a single point of failure. Secondly, since this TTP knows the location of every user, it becomes a valuable target for attacks. Since the TTP is blindly trusted, it could be compromised and become adversarial without the compromised state being detected by the users. Ghinita et al. [114] also highlight the fact that all the information held by the TTP is subject to the legal framework of the jurisdiction in which the TTP resides.

To overcome these disadvantages, peer-to-peer (P2P) protocols have been proposed that allow users to determine the cloaking groups amongst themselves [64, 114]. However, these P2P protocols also have disadvantages in terms of efficiency and security. For example, in the protocol by Chow et al. [64], users communicate with each other using direct P2P wireless links (e.g. Bluetooth). This means that Alice must establish $k-1$ new connections whenever she wishes to determine her cloaking region and must also be prepared to accept such connections from other users. This approach therefore cannot be used when no other users are within the range of Alice's P2P communication technologies. Furthermore, although not mentioned in the protocol, Alice will have to authenticate all her peers to ensure that only legitimate users are counted towards the k requirement. By contrast, in a centralized system, users are only required to communicate with the TTP when they change location or request a cloaking region. All user authentication is

performed by the TTP, thus improving security and decreasing the number of energy-consuming cryptographic operations performed on the mobile devices. As an alternative to using P2P protocols, the TRE can be used to overcome the limitations of the TTP whilst maintaining the security and efficiency benefits of the centralized architecture.

8.2.2 Enhancing LBS Privacy using TREs

Figure 8.3 shows a LBS communication architecture in which the TRE is used to enhance communication privacy. The main philosophy is to replace the TTP with a distributed set of TREs. The objective of this architecture is to ensure at least the same level of privacy as the TTP-based location cloaking system, whilst avoiding the two main disadvantages of the TTP.

As explained in the previous chapters, the TRE provides strong guarantees of its trustworthiness using remote attestation. Whenever a user establishes a secure connection to the TRE, the user can verify the TRE's software state using the FSA protocol, thus eliminating the need for an additional attestation step before the communication commences. This is the case for User A in Figure 8.3. As in the smart grid context, the source code of the TRE is available for verification by any user. It is again anticipated that there will be a small number of TRE configurations, thus making it feasible for trusted entities to publish complete lists of trustworthy PCR values. The use of the TRE therefore ameliorates the security and trust issues that would affect a TTP. Furthermore, since the TRE can also convince the service provider of its trustworthiness, the TRE can perform user authentication on behalf of the service provider, thus ensuring identity privacy even in applications that require user authentication. As shown in Figure 8.3, the TRE serves as an anonymizing proxy between the users and the service provider to avoid users being identified based on their network addresses. Therefore, in terms of the framework presented in Section 8.1, in this scenario, the TRE performs computations on multiple private inputs and provides delegated authentication functionality.

In order to overcome the performance bottleneck and single point of failure issues, the protocol is distributed over multiple TREs. The overall capacity of the system (i.e. number of users and rate of requests) can be increased by adding more TREs. If a TRE fails, its state can either be restored to a backup TRE using the backup and restore mechanisms described in Chapter 6, or users can simply connect to a different TRE. However, since the previously centralized functionality is now distributed over multiple TREs, a new *privacy-preserving dynamic partitioning* algorithm is required to ensure that this does not reduce users' privacy.

8.2.3 Privacy-Preserving Dynamic Partitioning

As shown in Figure 8.3, each TRE has a specific geographic point called its *centroid*. As part of the application-specific protocols, each TRE extends its centroid into one of its PCRs. Similarly to the global TTP, each TRE maintains a list of recent users near its

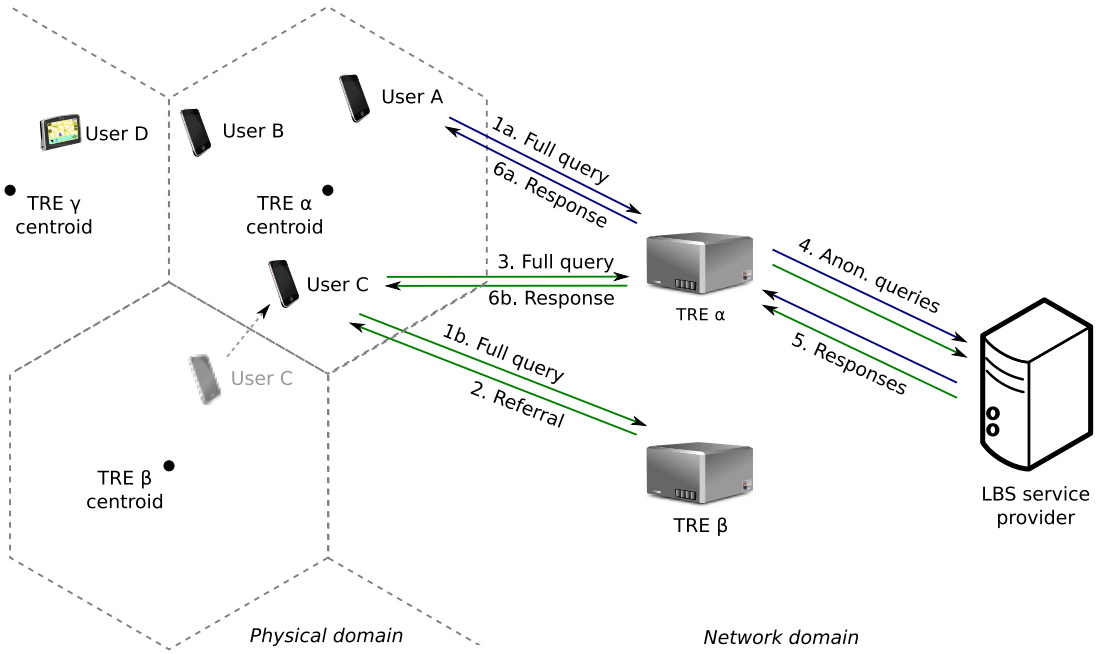

Figure 8.3: Communication architecture using the TRE to enhance LBS privacy

centroid, which it uses to compute the cloaking regions. The exact mechanisms through which users report their locations to the relevant TRE (e.g. continuously or only when changing location) depend on the protocol, but in all cases, the TRE is assumed to have at least the same level of information as the adversary about user locations. The TRE may use any non-P2P algorithm to determine the k-anonymous cloaking region. Although users can choose the TRE with which they communicate, it is advantageous for all users in the same geographic area to use the same TRE, since this minimizes the size of the cloaking regions. In the ideal case, all users would be perfectly partitioned into groups based on their current location. However, this introduces the following two new challenges:

1. How do users discover the nearest TRE centroid without revealing their locations to untrusted entities?

2. Can this partitioning be achieved without leaking information about users' locations?

For the first challenge, existing systems such as the Domain Name System (DNS) could not be used to look up the nearest TRE since this would reveal the user's location to the DNS servers, which may not be trusted with this information. Instead, each TRE maintains a list of nearby centroids, as well as some further afield. These lists are refreshed regularly (e.g. daily) and, to protect the lists' integrity, the remote attestation protocol is used to verify the state and confirm the centroid of any TREs added to the list. If the TRE receives a message from a user who is nearer to one of the other known centroids, the TRE will respond to the user with a referral message. The referral message contains all TREs from the list whose centroids are closer to the user's location than the current

TRE. The purpose of sending multiple TREs is to provide the user with other options in case one of the TREs is unavailable or is no longer in a trustworthy state. As shown in Figure 8.3, User A is nearest the centroid of TRE α and thus the user's query (message 1a) is processed by TRE α. User C had previously used TRE β but has now moved closer to the centroid of TRE α resulting in the referral (message 2). This user could choose to ignore the referral and still use TRE β by setting a flag in the query. Although this user will receive a sub-optimal result, this flexibility is necessary in case TRE α becomes unavailable or is not trusted by the user. Using this trustworthy referral service, users can therefore find the nearest TRE without disclosing their locations to any untrusted entities.

In terms of the second challenge, if incorrectly designed, the use of multiple TREs could leak information about the user's location. Upon receiving a query from TRE α, the service provider can infer that the user is both within the supplied location area and is also closer to TRE α than TRE β. Since the locations of the TRE centroids are public knowledge, the adversary can eliminate any possible locations that do not satisfy both these conditions. Therefore, if the cloaking region crosses the mid-point between two TREs' centroids, the adversary can possibly reduce the value of k. For example, in Figure 8.3, the ideal $k = 2$ cloaking region for User B contains User B and User D. However, if the query originates from TRE α, the adversary can eliminate User D, who is closer to TRE γ, and thus de-anonymize User B. However, by using the trustworthy referral mechanism described above, a given TRE will only store the locations of users nearest to its own centroid. Therefore the cloaking regions produced by a TRE will never include users nearer to another TRE, thus inherently preventing this information leak. An inevitable consequence of using multiple TREs is therefore that users near the mid-points between TREs receive slightly sub-optimal results, but these users can still adjust their k parameters to compensate for this. An external adversary who is able to monitor a user's network traffic might be able to learn something about the user's location by identifying the TRE with which the user is communicating, but this attack could be avoided by using anonymous communication channels between the users and the TREs.

8.3 Wireless Network Roaming

Wireless network roaming allows users to connect to and receive service from a *visited network (VN)* whilst away from their *home network (HN)*. Before providing service to a roaming user, the VN requires some type of assurance that this user is a legitimate subscriber of a HN with which the VN has a roaming agreement. Once service has been provided, the VN requires some mechanism to bill the user's HN for that service. However, this type of roaming also gives rise to various privacy concerns. When considering privacy in this context, there are three different types of adversaries: the VN, the HN and other network eavesdroppers. Using the framework in Section 8.1, the VN and HN are classified as internal adversaries whilst the eavesdroppers are external adversaries. By definition, the HN knows users' identities and the VN knows users' approximate locations

(e.g. within the VN's coverage area). However, since VNs are usually associated with specific locations, users may wish to conceal their location histories (i.e. details of VNs used) from their HN. This is a form of information privacy as defined in the framework. Similarly, the user may desire identity privacy with respect to the VN to prevent tracking by this potentially untrusted network. Samfat et al. [233] have presented a general classification of untraceability (privacy) requirements in network roaming and have identified five common classes of requirements. The classes are cumulative (e.g. C_2 includes all the requirements of C_1) and are related to information privacy and identity privacy as follows:

C_1 **Hiding user identities from eavesdroppers:** As the first class of requirements, all roaming protocols should protect the user's identity from external eavesdroppers. If this requirement is not met, an eavesdropper could track specific users and perform traffic analyses on their communications. This requirement can be achieved using secure communication channels (i.e. encapsulating the authentication messages).

C_2 **Hiding user identities from the VN:** This provides a degree of identity privacy with respect to the VN since the VN cannot distinguish between different users from the same HN. However, it does not ensure full identity privacy because the VN still learns the users' HN.

C_3 **Hiding the identity of the HN from eavesdroppers:** An eavesdropper might have auxiliary information about which users from a particular HN are in a particular VN. If the eavesdropper can determine a roaming user's HN, it becomes easier to de-anonymize the user.

C_4 **Hiding the identity of the HN from the VN:** Similarly to the attack above, the VN may have auxiliary information about which users from a specific HN are roaming on the VN. However, this attack is prevented if the VN cannot identify the user's HN. This corresponds to full identity privacy with respect to the VN.

C_5 **Hiding user behaviour from the HN:** Finally, users' may wish to hide their behaviour (i.e. which VNs they have used) from the HN. This corresponds to ensuring users' information privacy with respect to the HN.

8.3.1 Existing Approaches for Ensuring Privacy

Samfat et al. [233] present three protocols for authenticating a roaming user to a VN. In their first protocol, the user authenticates to the VN using an alias to protect the user's identity. However, the identity of the user's HN must also be provided so that the VN can contact the HN to authenticate the user. This therefore provides classes C_1 and C_2 of untraceability. The second protocol involves a small change to hide the identity of the user's HN from eavesdroppers and thus achieves class C_3. The third protocol removes the need for each VN to contact the user's HN by creating a chain of authentications starting from the HN. Whenever the user moves to a new VN, the user's previous VN is contacted

to authenticate the user. Although this chain begins with the HN, the first VN has no way of knowing that the preceding network was the user's HN. The HN only learns the identity of the user's first VN. Therefore the authors claim that this protocol ensures a level of privacy between that of C_4 and C_5. However, the protocol assumes that the user will visit multiple VNs before returning to the HN, which may not always be the case. These three protocols only deal with user authentication and thus do not provide mechanisms through which the VN can send a bill to the HN for services provided. This functionality could be added in the first and second protocols but would undermine the privacy guarantees of the third protocol. The authors even state that classes C_4 and C_5 conflict with the need for billing, since the VN cannot send the user's bill to the HN.

Jiang et al. [142] have presented two authentication protocols that are similar to the first protocol above in that they ensure identity privacy using pseudonyms (i.e. aliases).

Wan et al. [277] argue that class C_4 does not mitigate against a serious privacy threat. They therefore define a weaker class C_5 in which the VN may know the identity of the user's HN. They propose an authentication protocol that satisfies this weaker C_5 using pseudonyms and identity-based encryption. In their protocol, before leaving the HN, the user obtains a number of authenticated pseudonyms that can be used to authenticate to VNs based on a common root authority, which is assumed to be trusted by all networks. This protocol ensures information privacy with respect to the HN because the VN can authenticate the user without contacting the HN. However, this protocol also does not provide a mechanisms through which the HN can bill the user and pay the VN.

8.3.2 Privacy-Enhanced Wireless Network Roaming using TREs

Figure 8.4 shows a communication architecture for a wireless network roaming scenario in which the TRE is used to enhance communication privacy. With reference to the framework, this architecture aims to ensure both identity privacy (e.g. class C_4) and information privacy (e.g. class C_5). The protocol used in this architecture is divided into the authentication phase (message flows 1-3) and the billing phase (message flows 3-5). Before the protocol commences, each HN that is willing to participate, registers its master signing key with all TREs the HN considers trustworthy. Alice (i.e. User A) is assigned a primary credential by the HN that uniquely identifies the combination of the user and the HN. For example, this credential could be an asymmetric key pair with a public key certificate signed by the HN (message flow 1). Alice also obtains a list of TREs (or TRE configurations) that are trusted by the HN. For the protocol to succeed, there must exist at least one TRE that is simultaneously trusted by the VN, the HN and the user.

Authentication Phase

The protocol begins when Alice leaves the HN and connects to the VN. She provides the VN with a random subset of the list of TREs trusted by the HN, in order to avoid revealing her HN. The VN either already trusts some of these TREs, or performs the

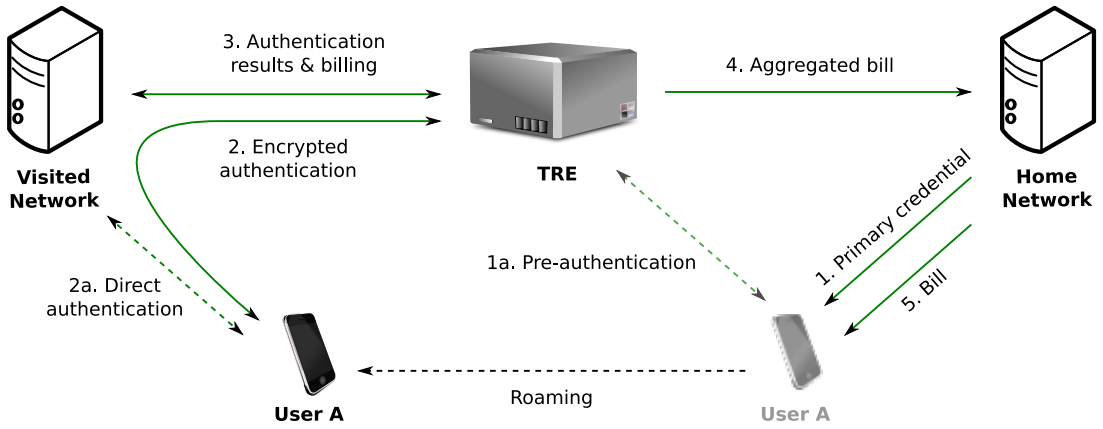

Figure 8.4: Privacy-enhancing wireless network roaming architecture using the TRE

remote attestation protocol with them to establish their trustworthiness. The VN then presents Alice with the subset of TREs it trusts and allows her to establish connections to any of these TREs. Alice connects to these TREs and performs the attestation protocol until she finds a TRE that is trusted by all three participants (message flow 2).

As described in the framework, a type of delegated authentication protocol is then used to ensure identity privacy whilst authenticating Alice to the VN. Alice establishes a secure channel with the chosen TRE through the VN and authenticates herself to the TRE using her primary credential. If the credential is signed by a participating network, the TRE can verify the signature using the network's master key. The TRE responds to the VN with an authentication token asserting that this user is authenticated and that a billing arrangement with her HN is in place (message flow 3). If Alice's credential specifies that she has a credit limit, the TRE queries her HN to determine her available credit and includes this in the authentication token. To avoid double spending, her HN will not allow other TREs to query Alice's available credit limit until this TRE has completed the billing phase of the protocol. Since Alice establishes a secure channel directly to the TRE, the VN cannot obtain even the public part of Alice's primary credential, since this would identify Alice's HN. Since the TRE protects Alice's identity from the VN, this protocol ensures identity privacy with respect to the VN, thus meeting the requirements of class C_4. During the authentication phase, the HN and VN both trust the TRE to authenticate users correctly whilst Alice trusts that the TRE will not reveal her identity to the VN.

If the destination VN is known in advance, a pre-authentication mechanism can be used to avoid having to contact the TRE during the authentication phase. In this mechanism, Alice contacts one of the TREs trusted by the VN and authenticates herself using her primary credential as usual (message flow 1a). The TRE issues Alice with a single-use temporary authentication token containing the equivalent information the TRE would have provided to the VN in the online protocol. Alice can use this token to authenticate directly to the VN without involving the TRE (message flow 2a). She can also give this temporary credential to a friend, but usage will still be billed to her account.

Billing Phase

Once Alice leaves the VN, or at pre-determined intervals, the VN sends her bill to the TRE that provided her authentication token. The TRE does not forward this directly to Alice's HN but instead uses it as input to a privacy-preserving temporal aggregation function, similar to that used in the smart meter billing protocol. In this case, the TRE is performing a computation on multiple private inputs from the same user. At the end of the billing period, the TRE provides the HN with the total amount to bill each user and the total amount to pay each VN. This mechanism prevents the HN from linking Alice to a specific VN by matching the values of the bills and payments. The only case in which this mechanism does not ensure privacy is if Alice was the only user from her HN to visit a specific VN and she did not visit any other VNs during that billing period. However, for any reasonably sized system, this case is very unlikely. During the billing phase, the HN and VN both trust the TRE to calculate the bills correctly whilst Alice trusts that the TRE will not reveal her behaviour to her HN. Although it has been argued that fulfilling the requirements for classes C_4 and C_5 would conflict with billing functionality [233], this did not take into account the possibility of using a TRE. The combination of delegated authentication and temporal aggregation made possible by the TRE allows this architecture to support both authentication and billing whilst ensuring full information and identity privacy (class C_5) with respect to all adversaries.

8.4 Secure Multiparty Computation using the TRE

Various cryptographic protocols have been proposed to achieve SMC using techniques such as garbled circuits [282, 283], oblivious transfer [219, 150] and homomorphic encryption [113, 112, 75], as described in Chapter 2. In theory, these cryptographic protocols allow the participants to jointly compute some function on their private inputs without disclosing their inputs to any other entity. Figure 8.5 shows the idealized setting for a cryptographic SMC protocol, consisting of two participants: Alice and Bob. However, a more realistic representation of this setting is shown in Figure 8.6, in which Alice and Bob (the users), being unable to perform 30+ digit mathematical operations in their heads, delegate these computations to their local computer systems. Alice's secret is therefore held by a system in which, for example, the hardware is produced by Harry, the CPU by Charlie, the memory by Megan, the OS by Linus et al., the SMC cryptographic library by Sam and the final application software by Dave. The users' private inputs can be accessed by most of these components, including the CPU, memory, OS, SMC cryptographic library and the SMC application[3]. Nevertheless, for various reasons, Alice still *trusts* her local system to protect her private inputs and thus she is willing to provide these inputs to her system. If Alice did not trust her local system and refused to provide her inputs, the

[3]Even if the user's private inputs are protected by new technologies, such as Intel SGX, these inputs are still accessible to at least the CPU, SMC cryptographic library, and SMC application.

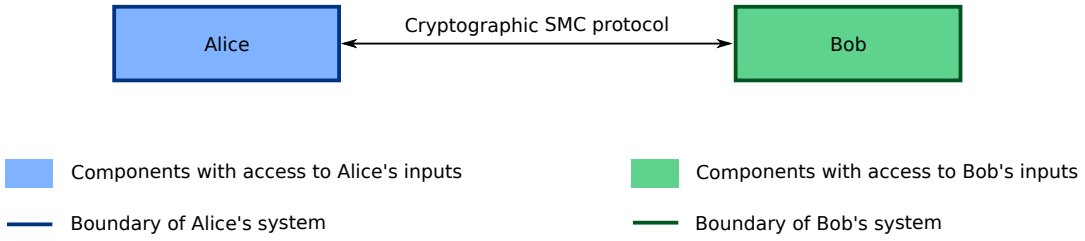

Figure 8.5: Idealized setting of a cryptographic SMC protocol

protocol could not continue. There are two main reasons why Alice may trust her local system.

Possibility for inspection: Firstly, it is possible for Alice to inspect her local system in great detail. Should she wish, Alice can usually inspect the software state of her local system to determine exactly which pieces of software are being executed. In some cases, Alice can also inspect the source code of this software[4]. Alice can use TC technologies, such as authenticated boot, to ensure that all software that executes on her local system is measured and can thus be inspected. Since she has physical access to the platform, Alice can inspect the hardware configuration of her local system. For example, she can determine exactly which hardware components constitute her system. However, the extent of hardware inspection is usually limited by cost constraints (e.g. it is infeasibly expensive to inspect the internal circuitry of the CPU).

Degree of control: Secondly, Alice has a relatively high degree of control over her local system in terms of both hardware and software. Alice may be able to select the hardware and software components that constitute her local system (within the constraints of interoperability). For example, having selected a CPU and platform architecture, Alice can select any OS that supports the chosen architecture. Alice can use TC technologies, such as authenticated boot and sealed storage, to ensure that her private information can only be accessed when her local system is in a trustworthy state. Since she has physical access to the platform, Alice may be able to add or remove hardware modules and reset the platform. Alice can also control physical access to her platform (e.g. through physical security). However, there are limitations on Alice's degree of control over her local system. It is assumed to be infeasibly expensive for her to substantially modify the behaviour of hardware components. Furthermore, if Alice is attempting to perform a specific cryptographic protocol with Bob, it is generally infeasible for her to change the cryptographic operations that must be performed on her local system and still complete the protocol successfully. Therefore, Alice's degree of control is usually limited to selecting the hardware and software, configuring the software, and choosing the inputs to provide to the software.

It can be argued that, of the two reasons above, the first is always mandatory whilst the

[4]However, the usefulness of inspecting source code depends on Alice trusting or verifying the compiler.

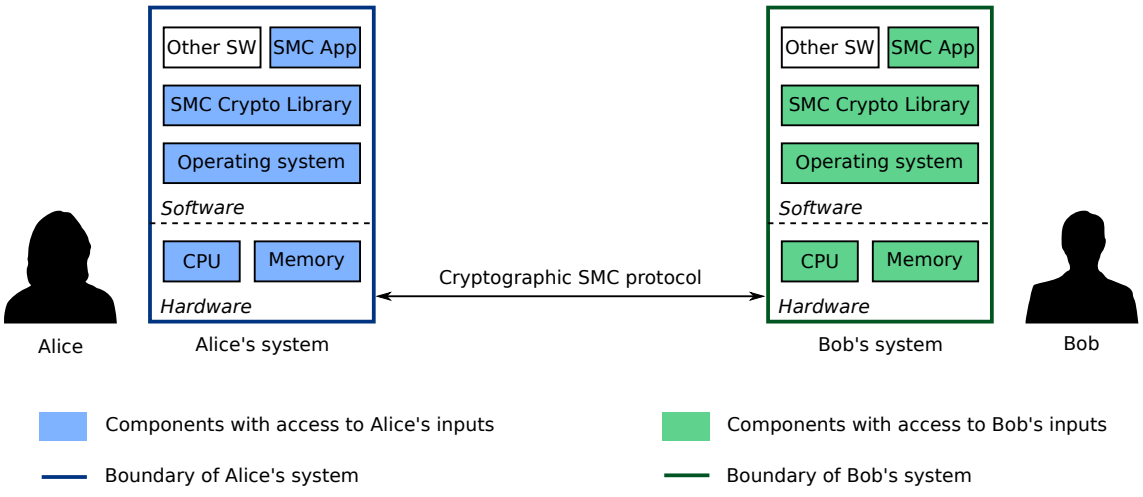

Figure 8.6: Real setting of a cryptographic SMC protocol

second is only required in specific circumstances. In order to trust her local system, Alice must be able to inspect it to determine that it is composed of the expected components and behaving in the expected manner. If, through inspection, she determines that she does not trust her local system in its present state, she requires a sufficient degree of control to bring the system into a trustworthy state. Once her system is in a trustworthy state, Alice only requires the ability to choose the inputs she provides.

In the cryptographic SMC protocols, the *boundary* of Alice's system is still largely viewed as the physical boundary of her platform (or perhaps a virtual boundary within her system if she uses a secure VM or an OS such as Qubes[5]). However, with the emergence of increasingly-distributed computing architecture, it is not guaranteed that Alice's private information will always remain within this boundary. For example, Alice's private information may be synchronized across her various devices or may be backed up outside of her local system. If she is using a mobile or resource-constrained device, certain computation may be outsourced to the cloud to improve performance and reduce energy consumption [65, 66, 70, 128, 203, 210]. Therefore, the boundary of Alice's system should be redefined to encompass all components that share the same level of trust from Alice's perspective.

The TRE is designed such that it can be inspected by Alice to exactly the same level of detail as she can inspect her local system. This is made possible by the TRE's system architecture, as described in Chapter 6, and the remote attestation protocol, as presented in Chapter 7. Since the TRE uses a minimized software TCB, it is arguably even easier for Alice to inspect the TRE's software than it is for her to inspect the software on her local system. This remote inspection capability is dependent on Alice trusting the root of trust used by the TRE (e.g. the TPM). It is reasonable to claim that Alice will usually trust this component because it is likely that she uses a similar root of trust on her local

[5] `https://www.qubes-os.org/`

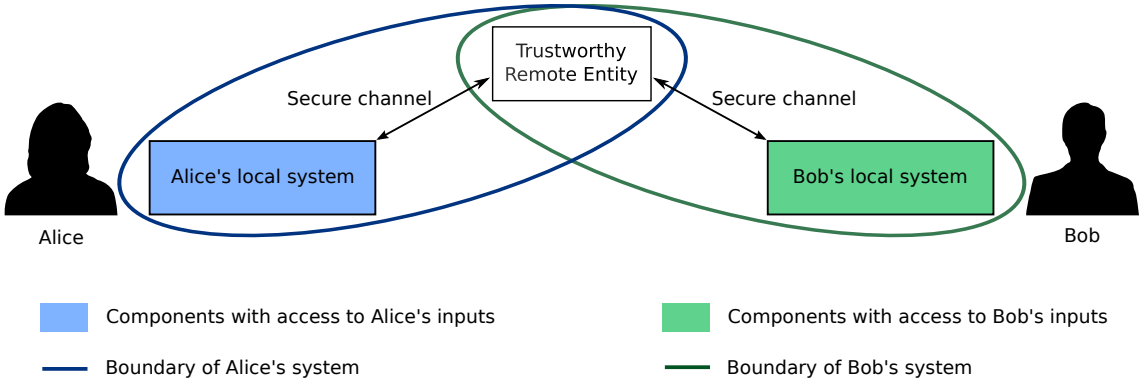

Figure 8.7: Secure multiparty computation using the TRE

system. For example, Alice is likely to use a TPM to support the authenticated boot that allows her to inspect all software on her local system. Although Alice does not have the same degree of control over the TRE as she may have over her local system, this need not affect her choice to trust the TRE if it is already in a trustworthy state. Alice can still choose which inputs she provides to the TRE. The remote attestation protocol is designed to allow Alice to detect whenever a change is made to the TRE's hardware or software, even by someone with physical access to the TRE (e.g. platform resets can be detected). Although the TRE may not be trustworthy in the absolute sense, it can be inspected and thus trusted by Alice to the same extent as her local system. Therefore, the boundary of Alice's overall system should be extended to include the TRE as shown in Figure 8.7. The same arguments apply to Bob and thus the boundary of Bob's system should also be redefined to include the TRE.

As shown in Figure 8.7, both Alice and Bob's systems, which were previously disjoint, can now both include the TRE as a common component. Although Alice and Bob may not be able to inspect each other's local systems, they can both inspect and therefore trust the TRE to the same extent as their respective local systems. This partial overlap between the participants' systems presents a new approach for achieving SMC. Instead of using her local system to perform a cryptographic protocol, Alice simply delegates her private input and the relevant computation to this shared component within her system, just as she would delegate data and instructions to a co-processor on the same physical platform. Since Bob performs the same type of delegation, the TRE has access to both participants' private inputs and since the TRE is a Turing-complete component, it can compute any function on these inputs. The TRE is not limited to two-party scenarios and, by the same arguments, can be included as a shared component by any number of subjects, limited only by the TRE's computational and communication capacity.

The TRE fulfils all the requirements for SMC given by Lindell and Pinkas [164] and explained in Chapter 2. It ensures *privacy* and *independence of inputs* since all participants communicate directly with the TRE and the TRE does not reveal their private

inputs to any other entities. It provides *correctness* because all computation is performed by the TRE and no entity can influence the TRE's behaviour. It provides *guaranteed output delivery* and *fairness* to the same extent as the cryptographic protocols in that all participants are guaranteed to receive their respective outputs in all cases except those in which the adversary has control of the communication network. This type of adversary (i.e. an entity in set $\mathcal{N}$ in the framework) can mount a denial of service attack against both the cryptographic SMC protocols and the TRE-based approach by blocking the respective participants' network traffic. Even though an adversary with physical access to the TRE can mount a complete denial of service against this TRE (e.g. by blocking all network communication), this only creates a temporary delay in the overall protocol while the participants connect to a different TRE. It can be trivially shown that the TRE exhibits identical behaviour to the ideal system in the *ideal/real simulation* used to evaluate the security of SMC protocols.

The use of the TRE as a shared component between multiple participants provides significant advantages in terms of efficiency by minimizing the computational requirements of the participant's local systems as well as the number of messages sent over the network (i.e. the computational and communication complexity of the system). As explained in Chapter 2, for a computation involving n participants, cryptographic SMC protocols could require $O(n^2)$ operations (or $O(n^2/s)$ operations of pre-computation) per multiplication. However, as shown in the performance analyses graphs in Chapter 6, the time taken by the TRE to perform multiparty computations always scales linearly with the number of participants involved (i.e. $O(n)$). Therefore, since the TRE can be trusted to the same extent as the participants' local systems, it can be included as a common component in all participants' systems and can thus be used to achieve effective, efficient and secure multiparty computation.

8.5 Summary

This chapter has presented a formalized view of the TRE concept and demonstrated how the TRE can be used to enhance communication privacy in application domains beyond the smart grid.

In order to use the TRE to enhance communication privacy, it is necessary to consider three aspects of the scenario. A framework has been developed that provides the underlying structure and tools for reasoning about these three aspects. Firstly, the participants and adversaries in the scenario must be identified and categorized according to their capabilities. In particular, it is important to distinguish between external and internal adversaries, since the latter pose a significantly greater threat to communication privacy. Secondly, the required forms of communication privacy with respect to these adversaries must be identified. Communication privacy can be achieved by either preventing untrusted parties from obtaining private information (information privacy), or by dissociating the subject's identity from the information (identity privacy). Thirdly, the functionality that

the TRE can perform to ensure the identified forms of privacy must be specified. The primary privacy-enhancing operations provided by the TRE include differentially-private spatial and temporal aggregation, pseudonymization and delegated authentication. This framework is demonstrated through two smaller case studies in which the TRE is used to enhance communication privacy.

In Location-Based Services (LBS), the user communicates her location information to a potentially untrusted service provider (an internal adversary). The user may not wish to reveal her precise location to the service provider (information privacy). Alternatively, the user may wish to remain anonymous with respect to the service provider (identity privacy) and thus cannot provide her precise location since this could be used to de-anonymize her. In this context, both information privacy and location privacy can be ensured by hiding the user within a spatio-temporal cloaking region containing at least $k-1$ other users. To provide the most accurate results, the cloaking region should be as small as possible. Various protocols have been proposed that use a TTP to calculate the cloaking region. However, two main disadvantages of this approach have been identified: since the TTP is a single centralized entity, it may become a performance bottleneck, and since it is blindly trusted, the TTP may compromise users' privacy. To overcome these challenges, a new architecture is proposed in which the TTP is replaced by a distributed set of TREs. Using the system architecture and remote attestation protocol presented in the preceding chapters, these TREs can provide guarantees of their trustworthiness to all participants. A new privacy-preserving dynamic partitioning protocol is proposed that allows users to discover the nearest TRE without revealing their locations to any untrusted entities and allows TREs to create cloaking regions that are guaranteed to meet the privacy requirements.

In wireless network roaming, the user wishes to make use of a visited network (VN) whilst away from her home network (HN). The user wants to conceal her identity from the VN (identity privacy) and prevent the HN from learning her behaviour (information privacy). However, the VN requires assurance that the user is a legitimate HN customer and the HN requires a mechanism to bill the user and pay the VN for services provided. Previous solutions have relied on aliases or pseudonyms to authenticate the user anonymously. However, none have provided mechanisms for achieving the billing requirement. A new protocol is presented in which authentication of a roaming user is delegated to a TRE that is mutually trusted by the VN, the HN and the user. The TRE assures the VN of the user's authenticity and possible credit limits without revealing the user's identity or HN. Afterwards, the VN sends the user's bill to the TRE which performs temporal aggregation to prevent the HN from inferring the user's behaviour. At the end of the billing period, the TRE provides the HN with total bills for each user and total payments due to each VN, thus ensuring both identity privacy and information privacy.

Finally, this chapter presents a discussion of how the TRE can be used to achieve secure multiparty computation (SMC) whilst providing the same level of security as cryptographic protocols with respect to plausible threat models. In a real world setting, the

cryptographic protocols are performed by the users' local systems which consist of a variety of hardware and software components supplied by different entities. Each user trusts her local system because she can inspect it in great detail and can modify its configuration until it is in a trustworthy state. It is reasonable to assume that users trust the TRE's root of trust, and can thus inspect the TRE's software to the same extent as their local systems. As new distributed computing architectures emerge, the boundary of a user's system should no longer be defined as the physical boundary of the platform but rather as the set of components that share the same level of trust from the user's perspective. By redefining these boundaries, the TRE can be included as a common component in multiple participants' systems. Instead of performing a cryptographic protocol, participants delegate their computation and private inputs to this shared component. Since the TRE can be included as a common component by multiple participants and can compute any function on the participants' private inputs, it can perform effective, efficient and secure multiparty computation.

Overall, this chapter therefore confirms the second primary research hypothesis by demonstrating that the concept of the TRE can be formalized and used to enhance communication privacy in application domains beyond the smart grid.

Chapter 9

Conclusion and Future Work

9.1 Research Context and Hypotheses 212
9.2 Main Contributions 214
9.3 Future Work 217
9.4 Summary 219

9.1 Research Context and Hypotheses

Communication privacy refers to the ability of individuals to control what personal information related to them can be collected or inferred as a result of their communication. It is primarily threatened by other legitimate but untrusted participants in the communication exchange (i.e. internal adversaries). Each entity aims to achieve the communication objectives without revealing private information to any untrusted entities. This differs from the classical security property of secrecy, which aims to protect information exchanged between mutually trusting participants. A comprehensive example of the need for communication privacy is the current deployment of smart energy meters to create a smart grid. The frequent (e.g. half-hourly) consumption measurements provided by smart meters are necessary to facilitate detailed network monitoring as well as time-of-use (ToU) billing. However, if these measurements are communicated directly to an untrusted service provider, they could potentially be used to infer details of consumers' behaviour, thus diminishing consumers' privacy, as explained in Chapter 3. An emerging aspect of the smart grid is residential demand bidding, in which consumers place bids to reduce energy consumption or sell back energy to the grid. If bids can be linked to consumers, this information can also be used to infer private information about consumers.

Communication privacy considers both *how* the communication takes place as well as *what* information is communicated, and thus includes aspects of both anonymous communication and data privacy. Fundamentally, communication privacy can be achieved in two ways. The first, referred to as *information privacy*, uses data privacy techniques, such as k-anonymity and differential privacy, to ensure that no private information can be learned or inferred from the communicated information. In the smart grid scenario, information privacy can be ensured by aggregating the frequent consumption measurements from multiple consumers for purposes of network monitoring, or aggregating billing information from a single consumer over time to facilitate ToU billing. The concept of secure multiparty computation (SMC) aims to achieve information privacy by allowing multiple participants to jointly compute a function on their private inputs (e.g. aggregation) without revealing their inputs to anyone. However, in some cases, the nature of the communicated information precludes the use of information privacy techniques. For example, the bids placed by smart grid consumers cannot be spatially aggregated since they require individual responses, and cannot be temporally aggregated since they are time-sensitive. In these cases, the second approach for enhancing communication privacy, referred to as *identity privacy*, is required. Identity privacy uses techniques from anonymous communication, such as network intermediaries and pseudonymization, to dissociate the subject's identity from the communicated information.

Various approaches for enhancing communication privacy in the smart grid have been proposed in recent literature, using techniques such as pseudonymization [89, 101, 43], homomorphic encryption [107, 82, 162, 234, 170, 46, 44, 45, 54], blinding schemes [156, 238], cryptographic commitments [221, 138], and differential privacy [77, 7]. However,

previous research has not considered the use of a trustworthy intermediary.

As a new approach for enhancing communication privacy, this thesis introduces the concept of the *Trustworthy Remote Entity (TRE)*. The TRE is a computational and communication system situated as an intermediary in the communication path between mutually distrusting participants. The TRE exhibits two fundamental characteristics. The first is that the TRE is trusted by all participants to the same extent as they trust their own local systems. Participants can therefore send their private inputs directly to the TRE without compromising their privacy. This enables the TRE to compute privacy-enhancing functions on the inputs from multiple participants (e.g. differentially-private spatial aggregation), or on multiple inputs from a single participant (e.g. differentially-private temporal aggregation), thus ensuring information privacy. It also allows the TRE to perform privacy-enhancing actions such as pseudonymization or delegated authentication to ensure identity privacy. The second characteristic of the TRE is that it provides strong technical guarantees of its trustworthiness. This means that the TRE is not blindly trusted, as is the case for a Trusted Third Party (TTP), but instead uses Trusted Computing (TC) technologies and, in particular, remote attestation to establish its trustworthiness. Relying parties are therefore able to inspect the behaviour of the TRE in great detail, but none can interfere with its operation without being detected.

9.1.1 Research Hypotheses

Within this context, this thesis describes the investigation of two primary research hypotheses:

1. *In the context of the smart grid, the concept of a Trustworthy Remote Entity (TRE) can be realized and used to enhance the privacy of consumers whilst maintaining the primary functionality of the system.*

2. *The concept of a TRE can be formalized and used to enhance communication privacy in other application domains.*

This research endeavour has focussed not only on verifying or falsifying these hypotheses, but also on determining the *extent* to which each hypothesis is true as well as identifying any constraints or necessary assumptions. This research has included both a top-down and a bottom-up perspective of this topic. The top-down perspective begins with a specific problem in the application domain, namely enhancing communication privacy in the smart grid, and leads to a set of functional requirements for the TRE. From this perspective, this research has answered questions such as the extent to which the TRE can be realized using current technologies, and determined which technologies are most suitable for realizing the TRE. Conversely, the bottom-up perspective begins with a specific technological capability, namely the TRE, and investigates how this capability can be used to solve relevant application domain problems. From this perspective, this research has shown how the TRE concept can be formalized and used to enhance communication privacy in

application domains beyond the smart grid. The primary research hypotheses span both the top-down and bottom-up perspectives and thus require a superposition of the results from each perspective. This research has confirmed both hypotheses and, in the process, has resulted in five main contributions, as summarized in the next section.

9.2 Main Contributions

9.2.1 Analysing Privacy Properties

In order to enhance communication privacy using the TRE, the application-layer communication protocols must provide the required privacy properties. The first main contribution of this thesis, presented in Chapter 4, is the development of a systematic analysis methodology for the privacy properties of undetectability and unlinkability. Since internal adversaries are the primary threat to communication privacy, this methodology considers an internal honest-but-curious (HBC) adversary, who will not deviate from the protocol but will attempt to learn or infer private information about other participants. This is a realistic representation of entities who are constrained by public scrutiny and/or regulatory controls. By modelling the adversary as a deductive system and the privacy properties as reachability assertions, this methodology can be directly integrated with existing methodologies for analysing security properties, which are also based on reachability. Although completeness cannot be claimed, the deductive system is sound and can thus be used to systematically test for vulnerabilities.

This new methodology has been implemented in the CSP process algebra and used to enhance the Casper/FDR security protocol analysis tool. This enhanced Casper-Privacy tool can analyse both security and privacy properties using the same formal model. The Casper-Privacy tool has been evaluated by re-analysing three protocols from the literature and analysing a further four smart grid protocols. In all cases, the tool identifies all known vulnerabilities and, in some cases, identifies new vulnerabilities arising from the tension between the security and privacy properties. This demonstrates the importance of analysing the interplay between security and privacy properties.

9.2.2 Privacy-Enhancing Smart Grid Communication Architecture

The second main contribution, presented in Chapter 5, is a new privacy-enhancing communication architecture for the smart grid, based on the TRE. For each smart grid information flow, this architecture includes a new protocol that provides consumers with the same level of privacy as before the implementation of the smart grid. For network monitoring, the TRE performs differentially-private spatial aggregation of consumption measurements from multiple consumers to ensure information privacy. For billing purposes, the TRE performs temporal aggregation on each consumer's billing information to ensure information privacy whilst facilitating dynamic pricing. Unlike the other information flows, the demand response (DR) information flow requires bi-directional communication between

each consumer and the service provider, and the communicated information cannot be spatially or temporally aggregated. In this flow, the TRE combines identity privacy and information privacy by pseudonymizing consumers' bids and performing temporal aggregation on the incentives paid to successful bidders to avoid leaking information. Due to the nature of this application domain, none of these protocols diminish the functionality of the smart grid.

The security and privacy properties of these protocols have been analysed using the Casper-Privacy tool, which showed that they do not exhibit any of the vulnerabilities found in the previously analysed protocols. Compared to other proposals, the new protocols are significantly more efficient and practical since they require fewer messages and cryptographic operations, and they do not require any changes to the cryptographic capabilities defined in current smart meter specifications. These protocols have been prototyped using elliptic curve cryptography (ECC) and the DLMS-COSEM specification, which are both mandatory for all smart meters in the UK. This therefore confirms part of the first research hypothesis.

9.2.3 TRE Design, Implementation and Evaluation

The third main contribution, presented in Chapter 6, is the design, prototype implementation and evaluation of the TRE. Starting from a set of TRE security requirements, an abstract TRE reference architecture has been defined and five concrete architectures have been described and analysed. The x86-TPM TRE architecture has been implemented as a fully-functional open source prototype. In this architecture, the TRE software runs directly on the platform, without an OS, and is measured by the DRTM late-launch. To facilitate the establishment of attestation-based trust relationships, the primary design objective is to minimize the size of the trusted computing base (TCB).

With a TCB of $24,719$ lines of C code, the x86-TPM TRE prototype is three orders of magnitude smaller than the Linux kernel. This TCB size, and the fact that it consists almost entirely of widely-used software libraries, make the TRE an ideal candidate for security audits and suggest that formal verification may be feasible in the near future. Of the five concrete architectures, the x86-TPM architecture achieves the smallest TCB using current technologies, whilst the x86-SGX architecture is anticipated to provide a further 30% TCB reduction when SGX technology becomes available. A partial prototype of the Linux-TPM TRE architecture provided marginally faster performance than the x86-TPM prototype, showing that some trade-offs between performance and TCB size arise at this level. All prototypes achieved a rate of at least 24.7 DLMS-COSEM request-response operations per second, which allows a single TRE to support over $20,000$ smart grid consumers. This proves that the TRE can be realized using current technologies and thus confirms another part of the first research hypothesis.

9.2.4 Final State Attestation (FSA) Protocol

The scalability of many remote attestation protocols is limited by their use of the root of trust for reporting (RTR). For example, the `TPM_Quote()` operation was measured to take on average 731 ms and cannot be parallelized. If the classic nonce-challenge attestation protocol were used for the TRE, this would severely limit the TRE's scalability. To overcome this challenge, the fourth main contribution, presented in Chapter 7, is a new highly scalable one-to-many remote attestation protocol designed specifically for systems like the TRE. To use this Final State Attestation (FSA) protocol, the attested system (i.e. the *prover*) must reach a *final state*, which can then only be changed by a platform reset. To detect platform resets, the prover generates an ephemeral TLS key pair, for which the private key $\mathcal{K}_{FSA-}$ is stored in volatile memory and is thus irrecoverably lost if the prover is reset. The prover includes a hash of the public key $\mathcal{K}_{FSA+}$ in all TPM quotes. Therefore, if a TPM quote represents the final state of the prover, and if the prover still has access to the private key $\mathcal{K}_{FSA-}$, then the quote must represent the current state of the prover. In addition to its use in the TRE, this protocol can be used in other final state systems, including *TrustVisor* [182], and may benefit next-generation attestation technologies such as SGX [12].

The security properties of the FSA protocol and two other attestation protocols have been analysed using the *TrustFound* framework. In a typical attestation scenario, the nonce-challenge protocol is secure but insufficiently scalable, whilst the one-to-many protocol based on global timestamps is vulnerable to reset attacks due to its lack of individual channel-binding. This analysis confirms that the FSA protocol fulfils the security requirements. Compared to other one-to-many protocols, the FSA protocol does not require a trusted time server and requires 25% fewer signature verification operations than its closest competitor. Benchmarks show that this protocol does not add any measurable overhead to the TRE prototype. In conjunction with the two preceding contributions, this therefore fully confirms the first research hypothesis.

9.2.5 Application-Independent Framework and Case Studies

The fifth main contribution, as presented in Chapter 8, is an application-independent formalization of the TRE concept and two smaller case studies demonstrating how the TRE can be used to enhance communication privacy in application domains beyond the smart grid. The formalization takes the form of a framework that provides the underlying structure and tools for reasoning about the three main aspects that must be considered when using the TRE to enhance communication privacy: 1) the classification of participants and adversaries based on their capabilities; 2) the choice between information privacy, identity privacy, or some combination thereof; and 3) the selection of TRE capabilities necessary to achieve the chosen form of privacy with respect to the identified adversaries.

The location-based service (LBS) case study demonstrates how the TRE can be used to ensure either information privacy or identity privacy in this context by determining k-

anonymous location cloaking regions. By replacing the commonly-used trusted third party with a distributed set of TREs, this new communication architecture provides guarantees of the TREs' trustworthiness and avoids single points of failure or performance bottlenecks.

The wireless network roaming case study shows how the combination of identity privacy and information privacy provided by the TRE can enable new functionality. By providing delegated user authentication, the TRE conceals the roaming user's identity from the visited network (VN). Unlike previous proposals, this architecture also facilitates privacy-preserving billing by using the TRE to perform temporal aggregation on each user's billing information, thus ensuring information privacy.

In comparison to cryptographic secure multiparty computation (SMC) protocols, it may be suggested that TRE-based solutions are less secure because participants' private inputs leave their local systems. However, in reality, participants can inspect the TRE to the same level of detail as they can their own local systems. The participants' trust boundaries should therefore be redefined to include the TRE, thus allowing participants to delegate private inputs to the TRE as if it were a local co-processor. In terms of performance, cryptographic SMC protocols with n participants usually require $O(n^2)$ operations whereas the TRE's performance has been shown to scale as $O(n)$. This shows that the TRE enables effective, efficient and secure multiparty computation. This contribution therefore confirms the second research hypothesis.

9.3 Future Work

9.3.1 Verifiable Execution

In addition to enhancing privacy, the TRE can also be used to provide guarantees that a specific computation has been performed correctly. This type of guarantee is already used in the new smart grid communication architecture presented in Chapter 5, since the service providers trust the TRE to perform the aggregation operations correctly (e.g. calculating consumers' bills correctly). This functionality can also be used as a standalone feature of the TRE.

For example, the Automatic Certificate Management Environment (ACME) protocol[1] allows web servers to interact with certificate authorities (CAs) to request public key certificates. The *Let's Encrypt* service[2] uses the ACME protocol to allow an agent on any web server to obtain a domain certificate without any human interaction. The certificate, which is trusted by the major web browsers, binds the website domain to a specific public key. In order to ensure that the corresponding private key is held by the agent that controls the website domain, the CA sends the agent one or more challenges, such as provisioning a specific DNS record or HTTP resource in the website's domain. The CA also provides a challenge for the agent to sign to prove ownership of the private key. If these steps succeed, the CA issues the certificate.

[1] `https://tools.ietf.org/html/draft-barnes-acme-03`
[2] `https://letsencrypt.org/howitworks/technology/`

As with all CAs, relying parties must trust that the CA performed the domain validation correctly. In normal CAs, this usually involves trusting that the CA's employees have done their job correctly, which is difficult to verify remotely. However, in the case of services like *Let's Encrypt*, in which this process is completely automated, remote attestation could be used to provide stronger guarantees about the CA's trustworthiness. Ideally, relying parties should be able to verify that the CA has performed the domain validation steps correctly and that the information in the certificate corresponds to the validated domain and public key. This could be achieved by using a TRE as the CA and using a modified version of the FSA protocol to establish the trustworthiness of this TRE-CA. Instead of using its ephemerally generated FSA key in TLS handshakes, the TRE-CA uses it to sign the domain certificates. In this way, the CA's long-term private key becomes redundant because a chain of trust can be established from the TPM, through the quote and the ephemeral FSA key, to the domain certificate. This relieves the TRE-CA from having to protect an online long-term signing key. When a web browser receives a certificate, it first verifies the TPM quote (which is likely to be one of a small number of published trustworthy values) and uses the FSA key embedded in that quote to verify the certificate. Using this modified FSA protocol, the quote does not give any indication of the TRE-CA's current state but instead attests to its state *at the time it signed the certificate*. In this scenario, the TRE is not being used to enhance privacy, but instead to provide verifiable and thus trustworthy execution of this important functionality.

9.3.2 Localized TREs

In this thesis, the TRE is described as a *remote* entity in the sense that it is physically distinct from the other communicating entities. The advantage of this is that the TRE is not required to run any other software on its platform. In contrast, Trusted Execution Environments (TEEs) are specifically designed to operate alongside untrusted software on the same platform. As explained in Chapter 6, the x86-SGX and ARM-TrustZone TRE architectures use TEEs to implement the TRE. With these architectures, the TRE could either be physically remote or could be implemented as a component of a specific participant's local system.

This type of localized TRE does not change the trust relationships since all parties are still able to inspect it to the same level of detail as they can their own local systems. The hosting participant, like the adversarial TRE operator, is still unable to modify the TRE's behaviour without being detected. However, localized TREs could change the performance characteristics of the system since the hosting participant would have a significantly higher bandwidth and lower latency communication channel to the TRE. This could be useful in scenarios in which there is an asymmetry between participants in terms of the amount of information exchanged with the TRE. For example, if one participant holds the majority of a dataset whilst the other participants each hold a small percentage, it would be advantageous from a performance perspective for the TRE to be hosted by the

first participant. A possible line of further research would be to investigate these localized TRE architectures and identify scenarios in which they could be beneficial.

9.3.3 Dynamic User-Defined TRE Functionality

In the scenarios described in this thesis, the behaviour of the TRE has been statically defined in the design of the communication architecture. However, since the TRE can perform any type of computation, its functionality could be dynamically defined by its users. In addition to sending private information to this *dynamic TRE (D-TRE)*, users could also upload functionality in the form of algorithms to be performed on specific pieces of information. The user who uploads the functionality would make use of the D-TRE's verifiable execution functionality, as discussed above, to ensure the functionality is performed correctly. As in the current TRE architectures, all participants could inspect the specific operations that will be performed by the D-TRE and verify that these provide the desired level of security and privacy. In a more advanced scenario, participants could upload information and permit arbitrary operations on their information, provided that certain security and/or privacy properties are maintained. This is similar to research projects such as TEP [9], Airavat [228] and πBox [159], but uses TC and remote attestation to provide a high level of assurance of the D-TRE's state and configuration.

Further research would be required in order to define the exact requirements for this type of dynamic functionality and develop protocols for uploading functionality to the D-TRE. Since a single D-TRE would provide multiple different types of functionality, it would be important to investigate the composability (amalgamation) of functionality. The privacy-enhancing smart grid architecture in Chapter 5 already demonstrates the advantages of composing different types of functionality by using the same information for both the network monitoring and billing protocols. There is therefore reason to believe that further advantages could be achieved through the composition of functionality, but it must be ensured that this does not undermine the security and privacy properties. Another line of research would be to investigate how the D-TRE could attest its state to relying parties efficiently, given that its functionality is no longer static. This type of scenario might benefit from isolation between TRE components, as mentioned in Chapter 6, allowing the D-TRE's static base system to be attested separately from the dynamic functionality. Overall, many new and innovative types of functionality could be provided by the D-TRE.

9.4 Summary

The objective of this research endeavour has been to introduce the concept of the TRE and investigate its use in enhancing communication privacy in the smart energy grid and in other application domains. In order to analyse and reason about communication privacy, Chapter 4 has introduced a new approach to modelling and automatically analysing the privacy properties of undetectability and unlinkability in communication protocols.

Chapter 5 has presented a new communication architecture for the smart grid, in which the TRE is used to enhance communication privacy in all three information flows, including those in which bi-directional communication is required. To demonstrate that the TRE can be realized using current technologies, Chapter 6 has described various system architectures for the TRE and shown that Trusted Computing technologies can be used to provide technical guarantees of the TRE's trustworthiness. To overcome the inherent scalability challenges of current remote attestation protocols, Chapter 7 has proposed the Final State Attestation (FSA) protocol, which is a highly scalable remote attestation protocol specifically designed for systems like the TRE. Moving beyond the smart grid, Chapter 8 has presented an application-independent formalization of the TRE concept and shown how the TRE can be used to enhance communication privacy in location-based services and wireless network roaming. Building on the research described in this thesis, Chapter 9 has discussed various possibilities for further research, including verifiable execution and the TRE-CA, asymmetric protocols using localized TRE architectures, and the possibility of a dynamic TRE.

This thesis confirms both of the primary research hypotheses and therefore demonstrates that the Trustworthy Remote Entity can be realized and used to enhance communication privacy in the smart energy grid and in other application domains.

Bibliography

[1] N. Aaraj, A. Raghunathan, and N. K. Jha. "Analysis and design of a hardware/-software trusted platform module for embedded systems". In: *ACM Transactions on Embedded Computing Systems* 8.1 (Dec. 2008), pp. 1–31 (cited on page 142).

[2] M. Abadi and R. Needham. "Prudent engineering practice for cryptographic protocols". In: *IEEE Transactions on Software Engineering* 22.1 (1996) (cited on page 166).

[3] M. Abadi and C. Fournet. "Mobile values, new names, and secure communication". In: *Proc. ACM SIGPLAN-SIGACT symposium on Principles of programming languages (POPL '01)*. 2001 (cited on page 56).

[4] M. Abadi and A. D. Gordon. "A calculus for cryptographic protocols: the spi calculus". In: *Proc. ACM conference on Computer and communications security*. CCS '97. 1997 (cited on page 56).

[5] M. Abadi and C. Fournet. "Private authentication". In: *Theoretical Computer Science* 322.3 (Sept. 2004), pp. 427–476 (cited on page 193).

[6] N. Abd Aziz, N. I. Udzir, and R. Mahmod. "Performance analysis for extended TLS with mutual attestation for platform integrity assurance". In: *The 4th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent*. 2014 (cited on page 160).

[7] G. Ács and C. Castelluccia. "I have a DREAM!: differentially private smart metering". In: *Proceedings of the 13th international conference on Information hiding*. IH'11. 2011 (cited on pages 46, 47, 96, 110, and 212).

[8] C. Adika and L. Wang. "Demand-Side Bidding Strategy for Residential Energy Management in a Smart Grid Environment". In: *IEEE Transactions on Smart Grid* 5.4 (July 2014), pp. 1724–1733 (cited on pages 88, 89, and 90).

[9] S. Ajmani, R. Morris, and B. Liskov. *A Trusted Third-Party Computation Service*. Tech. rep. MIT, 2001. URL: `http://pmg.csail.mit.edu/pubs/ajmani01trusted-abstract.html` (cited on pages 28 and 219).

[10] M. Alam et al. "Analysis of existing remote attestation techniques". In: *Security and Communication Networks* 5.9 (Sept. 2012), pp. 1062–1082 (cited on page 158).

[11] M. Albadi and E. El-Saadany. “A summary of demand response in electricity markets”. In: *Electric Power Systems Research* 78.11 (Nov. 2008), pp. 1989–1996 (cited on pages 36 and 37).

[12] I. Anati et al. “Innovative technology for CPU based attestation and sealing”. In: *Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy*. 2013 (cited on pages 27, 170, and 216).

[13] R. Anderson and S. Fuloria. “Who Controls the off Switch?” In: *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*. 2010 (cited on page 39).

[14] R. Anderson and S. Fuloria. “On the security economics of electricity metering”. In: *The Ninth Workshop on the Economics of Information Security (WEIS 2010)*. 2010 (cited on page 48).

[15] M. Arapinis, S. Bursuc, and M. Ryan. “Privacy-supporting cloud computing by in-browser key translation”. In: *Journal of Computer Security* 21.6 (Nov. 2013), pp. 847–880 (cited on page 57).

[16] M. Arapinis, V. Cheval, and S. Delaune. “Verifying Privacy-Type Properties in a Modular Way”. In: *2012 IEEE 25th Computer Security Foundations Symposium*. 2012 (cited on page 56).

[17] M. Arapinis et al. “Analysing Unlinkability and Anonymity Using the Applied Pi Calculus”. In: *Proc. IEEE Computer Security Foundations Symposium (CSF)*. 2010 (cited on pages 54, 56, 77, 78, 249, and 251).

[18] F. Armknecht et al. “An Efficient Implementation of Trusted Channels based on OpenSSL”. In: *Proceedings of the 3rd ACM workshop on Scalable trusted computing - STC ’08*. 2008 (cited on pages 168, 169, 171, 175, 182, and 183).

[19] B. Asare-Bediako, P. F. Ribeiro, and W. L. Kling. “Integrated energy optimization with smart home energy management systems”. In: *2012 3rd IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*. 2012 (cited on page 35).

[20] R. Assam and T. Seidl. “A Model for Context-Aware Location Identity Preservation Using Differential Privacy”. In: *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*. 2013 (cited on page 197).

[21] Y. Aumann and Y. Lindell. “Security against covert adversaries: efficient protocols for realistic adversaries”. In: *Proceedings of the 4th conference on Theory of cryptography (TCC’07)*. 2007 (cited on page 58).

[22] A. Avizienis et al. “Basic concepts and taxonomy of dependable and secure computing”. In: *IEEE Transactions on Dependable and Secure Computing* 1.1 (Jan. 2004), pp. 11–33 (cited on page 21).

[23] M. Backes et al. “AnoA: A Framework for Analyzing Anonymous Communication Protocols”. In: *2013 IEEE 26th Computer Security Foundations Symposium*. 2013 (cited on page 56).

[24] M. Backes et al. “Provably Secure and Practical Onion Routing”. In: *2012 IEEE 25th Computer Security Foundations Symposium*. 2012 (cited on page 56).

[25] G. Bai. “Formally Analyzing and Verifying Secure System design and Implementation”. PhD thesis. 2015 (cited on pages 70 and 269).

[26] G. Bai et al. “TrustFound: Towards a Formal Foundation for Model Checking Trusted Computing Platforms”. In: *FM 2014: Formal Methods*. Ed. by C. Jones, P. Pihlajasaari, and J. Sun. Lecture Notes in Computer Science. Springer International Publishing, 2014, pp. 110–126 (cited on pages 158, 177, and 269).

[27] C. Barcellona et al. “Multi-party metering: An architecture for privacy-preserving profiling schemes”. In: *2013 Sustainable Internet and ICT for Sustainability (SustainIT)*. 2013 (cited on page 46).

[28] E. Barker and J. Kelsey. “NIST Special Publication 800-90A Recommendation for Random Number Generation Using Deterministic Random Bit Generators”. In: January (2012) (cited on page 135).

[29] N. Batra et al. “NILMTK: An Open Source Toolkit for Non-intrusive Load Monitoring”. In: *Fifth International Conference on Future Energy Systems (ACM e-Energy)*. 2014 (cited on page 39).

[30] G. Bauer, K. Stockinger, and P. Lukowicz. “Recognizing the use-mode of kitchen appliances from their current consumption”. In: *Proceedings of the 4th European conference on Smart sensing and context*. EuroSSC’09. 2009 (cited on page 39).

[31] A. Baumann, M. Peinado, and G. Hunt. “Shielding applications from an untrusted cloud with Haven”. In: (Oct. 2014), pp. 267–283 (cited on page 27).

[32] C. Beckel, L. Sadamori, and S. Santini. “Automatic Socio-economic Classification of Households Using Electricity Consumption Data”. In: *Proceedings of the Fourth International Conference on Future Energy Systems*. e-Energy ’13. 2013 (cited on page 39).

[33] M. Bellare and P. Rogaway. “Optimal asymmetric encryption”. In: *Advances in Cryptology - EUROCRYPT’94*. Lecture Notes in Computer Science. 1995, pp. 92–111 (cited on page 64).

[34] M. Bellare et al. “Key-Privacy in Public-Key Encryption”. In: *Advances in Cryptology (ASIACRYPT)*. Ed. by C. Boyd. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2001, pp. 566–582 (cited on page 63).

[35] R. Bendlin et al. “Semi-homomorphic encryption and multiparty computation”. In: *Proceedings of the 30th Annual international conference on Theory and applications of cryptographic techniques: advances in cryptology - EUROCRYPT'11*. 2011 (cited on page 20).

[36] A. Beresford and F. Stajano. “Location privacy in pervasive computing”. In: *IEEE Pervasive Computing* 2.1 (Jan. 2003), pp. 46–55 (cited on page 196).

[37] S. Berger et al. “vTPM: virtualizing the trusted platform module”. In: *USENIX-SS'06 Proceedings of the 15th conference on USENIX Security Symposium*. 2006 (cited on page 140).

[38] D. C. Bergman et al. “Distributed non-intrusive load monitoring”. In: *ISGT 2011*. 2011 (cited on page 39).

[39] S. Berthold and S. Clauss. “Linkability estimation between subjects and message contents using formal concepts”. In: *Proc. ACM workshop on Digital identity management*. DIM '07. 2007 (cited on pages 56, 61, and 67).

[40] C. Bettini et al. “Anonymity and Historical-Anonymity in Location-Based Services”. In: *Privacy in Location-Based Applications*. Ed. by C. Bettini et al. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2009, pp. 1–30 (cited on pages 195, 196, and 197).

[41] B. Blanchet. “Security Protocol Verification: Symbolic and Computational Models”. In: *Proceedings of the First International Conference on Principles of Security and Trust*. POST'12. 2012 (cited on page 56).

[42] J.-M. Bohli, C. Sorge, and O. Ugus. “A Privacy Model for Smart Metering”. In: *2010 IEEE International Conference on Communications Workshops*. 2010 (cited on pages 28, 48, 96, and 110).

[43] F. Borges, L. A. Martucci, and M. Muhlhauser. “Analysis of privacy-enhancing protocols based on anonymity networks”. In: *Proc. IEEE Third International Conference on Smart Grid Communications (SmartGridComm)*. 2012 (cited on pages 42, 60, 82, 83, 212, and 259).

[44] F. Borges and M. Muhlhauser. “EPPP4SMS: Efficient Privacy-Preserving Protocol for Smart Metering Systems and Its Simulation Using Real-World Data”. In: *IEEE Transactions on Smart Grid* 5.6 (Nov. 2014), pp. 2701–2708 (cited on pages 46 and 212).

[45] F. Borges, F. Volk, and M. Muhlhauser. “Efficient, verifiable, secure, and privacy-friendly computations for the smart grid”. In: *2015 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*. 2015 (cited on pages 46 and 212).

[46] F. Borges et al. “A privacy-enhancing protocol that provides in-network data aggregation and verifiable smart meter billing”. In: *2014 IEEE Symposium on Computers and Communications (ISCC)*. 2014 (cited on pages 46, 110, and 212).

[47] S. W. van den Braak et al. “Trusted third parties for secure and privacy-preserving data integration and sharing in the public sector”. In: *Proceedings of the 13th Annual International Conference on Digital Government Research*. dg.o ’12. 2012 (cited on page 28).

[48] E. Brickell, J. Camenisch, and L. Chen. “Direct anonymous attestation”. In: *Proc. ACM conference on Computer and communications security (CCS’04)*. 2004 (cited on pages 56, 158, and 169).

[49] E. Brickell and J. Li. “Enhanced Privacy ID: A Direct Anonymous Attestation Scheme with Enhanced Revocation Capabilities”. In: *IEEE Transactions on Dependable and Secure Computing* 9.3 (May 2012), pp. 345–360 (cited on page 170).

[50] I. Brown. “Britain’s Smart Meter Programme: A Case Study in Privacy by Design”. In: *International Review of Law, Computers & Technology* (Feb. 2013) (cited on pages 39 and 60).

[51] M. Brusó, K. Chatzikokolakis, and J. Den Hartog. “Formal Verification of Privacy for RFID Systems”. In: *Proc. IEEE Computer Security Foundations Symposium (CSF)*. 2010 (cited on pages 54 and 56).

[52] M. Brusó et al. “Linking Unlinkability”. In: *Proc. International Symposium on Trustworthy Global Computing (TGC)*. 2012 (cited on page 54).

[53] M. Burrows, M. Abadi, and R. M. Needham. “A Logic of Authentication”. In: *Proc. Royal Society of London -Mathematical and Physical Sciences* 426.1871 (1989), pp. 233–271 (cited on page 56).

[54] N. Busom et al. “Efficient smart metering based on homomorphic encryption”. In: *Computer Communications* (Sept. 2015) (cited on pages 46 and 212).

[55] J. Camenisch and A. Lysyanskaya. “A Formal Treatment of Onion Routing”. In: *25th Annual International Cryptology Conference (CRYPTO’05)*. Ed. by V. Shoup. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, pp. 169–187 (cited on page 56).

[56] R. Canetti. “Security and Composition of Multiparty Cryptographic Protocols”. In: *Journal of Cryptology* 13.1 (Apr. 2000), pp. 143–202 (cited on page 18).

[57] D. Chadwick. *Design of Identity Management, Authentication and Authorization Infrastructure*. Tech. rep. 2010. URL: `http://homes.esat.kuleuven.ac.be/~decockd/tas3/final.deliverables/pm36/TAS3%5C_D07p1%5C_IDM-Authn-Authz%5C_V3p0.pdf` (cited on pages 76 and 247).

[58] D. Challener et al. *A practical guide to trusted computing*. Pearson Education, 2007 (cited on page 23).

[59] C. Chen et al. “HORNET: High-speed Onion Routing at the Network Layer”. In: *Proceedings of the 2015 ACM SIGSAC conference on Computer & Communications security - CCS ’15*. 2015 (cited on page 196).

[60] L. Chen et al. “A protocol for property-based attestation”. In: *Proceedings of the first ACM workshop on Scalable trusted computing - STC ’06*. 2006 (cited on page 158).

[61] L. Chen et al. “Property-Based Attestation without a Trusted Third Party”. In: *Proceedings of the 11th international conference on Information Security (ISC’08)*. Ed. by T.-C. Wu et al. Lecture Notes in Computer Science. 2008 (cited on page 158).

[62] Y.-T. Chen, A. Studer, and A. Perrig. “Combining TLS and TPMs to Achieve Device and User Authentication for Wi-Fi and WiMAX Citywide Networks”. In: *Wireless Communications and Networking Conference, 2008. WCNC 2008. IEEE*. 2008 (cited on page 163).

[63] V. Cheval, S. Delaune, and M. Ryan. “Tests for Establishing Security Properties”. In: *Trustworthy Global Computing*. Ed. by M. Maffei and E. Tuosto. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2014, pp. 82–96 (cited on page 57).

[64] C.-Y. Chow, M. F. Mokbel, and X. Liu. “A peer-to-peer spatial cloaking algorithm for anonymous location-based service”. In: *Proc. ACM international symposium on Advances in geographic information systems*. GIS ’06. 2006 (cited on page 197).

[65] B.-G. Chun and P. Maniatis. “Dynamically partitioning applications between weak devices and clouds”. In: *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services Social Networks and Beyond - MCS ’10*. 2010 (cited on page 206).

[66] B.-G. Chun et al. “CloneCloud: elastic execution between mobile device and cloud”. In: *Proceedings of the sixth conference on Computer systems - EuroSys ’11*. 2011 (cited on page 206).

[67] G. Coker et al. “Principles of remote attestation”. In: *Int. J. Inf. Secur.* 10.2 (June 2011), pp. 63–81 (cited on pages 158 and 159).

[68] *Constitution of the Republic of South Africa*. 1996 (cited on page 2).

[69] C. J. F. Cremers. “The Scyther Tool: Automatic Verification of Security Protocols”. In: *Computer Aided Verification*. Ed. by A. Gupta and S. Malik. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2008, pp. 414–418 (cited on page 56).

[70] E. Cuervo et al. “MAUI: making smartphones last longer with code offload”. In: *Proceedings of the 8th international conference on Mobile systems, applications, and services - MobiSys ’10*. 2010 (cited on page 206).

[71] C. Cuijpers and B.-J. Koops. “Smart Metering and Privacy in Europe: Lessons from the Dutch Case”. In: (Feb. 2012) (cited on page 60).

[72] C. Cuijpers and P. B.-j. Koops. *Het wetsvoorstel 'slimme meters': een privacytoets op basis van art . 8 EVRM Onderzoek in opdracht van de Consumentenbond*. Tech. rep. Universiteit van Tilburg, 2008 (cited on page 4).

[73] T. Dalenius. *Finding a needle in a haystack-or identifying anonymous census record*. 1986. URL: `http://www.jos.nu/Articles/abstract.asp?article=23329` (cited on page 13).

[74] T. Dalenius. “Towards a methodology for statistical disclosure control”. In: *Statistik Tidskrift* 15 (1977), pp. 429–444 (cited on page 16).

[75] I. Damgård et al. “Multiparty Computation from Somewhat Homomorphic Encryption”. In: *32nd Annual Cryptology Conference on Advances in Cryptology (CRYPTO'12)*. Ed. by R. Safavi-Naini and R. Canetti. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, pp. 643–662 (cited on pages 20 and 204).

[76] G. Danezis, R. Dingledine, and N. Mathewson. “Mixminion: Design of a Type III Anonymous Remailer Protocol”. In: *19th International Conference on Data Engineering*. 2003 (cited on page 12).

[77] G. Danezis, M. Kohlweiss, and A. Rial. “Differentially private billing with rebates”. In: *Proceedings of the 13th international conference on Information hiding*. IH'11. 2011 (cited on pages 44, 99, 112, and 212).

[78] S. J. Darby. “Load management at home: advantages and drawbacks of some 'active demand side' options”. In: *Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy* (2012), pp. 1–9 (cited on pages 33, 37, and 38).

[79] S. J. Darby. “the 'Time' Dimension of Electricity, Options for the Householder, and Implications for Policy”. In: *The World Renewable Energy Congress* (2011), pp. 1086–1093 (cited on pages 33 and 34).

[80] S. J. Darby and E. McKenna. “Social implications of residential demand response in cool temperate climates”. In: *Energy Policy* 49 (Oct. 2012), pp. 759–769 (cited on pages 33 and 37).

[81] S. Delaune, M. Ryan, and B. Smyth. “Automatic Verification of Privacy Properties in the Applied pi Calculus”. In: *Trust Management II*. Ed. by Y. Karabulut et al. IFIP - The International Federation for Information Processing. Springer US, 2008, pp. 263–278 (cited on page 56).

[82] P. Deng and L. Yang. “A secure and privacy-preserving communication scheme for Advanced Metering Infrastructure”. In: *Innovative Smart Grid Technologies (ISGT), 2012 IEEE PES*. 2012 (cited on pages 46 and 212).

[83] R. Dewri. “Local Differential Perturbations: Location Privacy under Approximate Knowledge Attackers”. In: *IEEE Transactions on Mobile Computing* 12.12 (Dec. 2013), pp. 2360–2372 (cited on page 195).

[84] R. Dingledine, N. Mathewson, and P. F. Syverson. “Tor: The Second-Generation Onion Router”. In: *Proceedings of the 13th USENIX Security Symposium, 2004*. 2004 (cited on pages 12, 61, and 196).

[85] D. Dolev and A. Yao. “On the security of public key protocols”. In: *IEEE Transactions on Information Theory* 29.2 (Mar. 1983), pp. 198–208 (cited on pages 55, 57, and 90).

[86] S. Drenker and A. Kader. “Nonintrusive monitoring of electric loads”. In: *Computer Applications in Power, IEEE* 12.4 (Oct. 1999), pp. 47–51 (cited on pages 4 and 39).

[87] C. Dwork. “Differential Privacy”. In: *33rd International Colloquium on Automata, Languages and Programming (ICALP'06)*. Ed. by M. Bugliesi et al. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2006, pp. 1–12 (cited on pages 16, 17, 28, and 95).

[88] C. Dwork et al. “Calibrating Noise to Sensitivity in Private Data Analysis”. In: *Third Theory of Cryptography Conference (TCC'06)*. Ed. by S. Halevi and T. Rabin. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2006, pp. 265–284 (cited on pages 16 and 96).

[89] C. Efthymiou and G. Kalogridis. “Smart Grid Privacy via Anonymization of Smart Metering Data”. In: *Proc. IEEE International Conference on Smart Grid Communications (SmartGridComm)*. 2010 (cited on pages 42, 60, 80, 212, and 255).

[90] J.-E. Ekberg, K. Kostiainen, and N. Asokan. “The Untapped Potential of Trusted Execution Environments on Mobile Devices”. In: *IEEE Security & Privacy* 12.4 (July 2014), pp. 29–37 (cited on page 26).

[91] J.-E. Ekberg et al. “Scheduling execution of credentials in constrained secure environments”. In: *Proceedings of the 3rd ACM workshop on Scalable trusted computing - STC '08*. 2008 (cited on page 26).

[92] Energy UK. *Data Guide for Smart Meters*. 2013. URL: `http://www.energy-uk.org.uk/files/docs/Policies/Smart%20Meter%20policies%20consultation%20responses/2013/smart%5C_meter%5C_data%5C_guide%5C_version%5C_1-june-13.pdf` (cited on pages 3 and 89).

[93] D. Engel. “Wavelet-based load profile representation for smart meter privacy”. In: *2013 IEEE PES Innovative Smart Grid Technologies Conference (ISGT)*. 2013 (cited on page 111).

[94] P. England. “Practical Techniques for Operating System Attestation”. In: *Proceedings of the 1st international conference on Trusted Computing and Trust in Information Technologies - TRUST ’08*. Ed. by P. Lipp, A.-R. Sadeghi, and K.-M. Koch. Lecture Notes in Computer Science. 2008 (cited on page 140).

[95] D. R. Engler, M. F. Kaashoek, and J. O’Toole. “Exokernel: An Operating System Architecture for Application-Level Resource Management”. In: *Proceedings of the fifteenth ACM symposium on Operating systems principles (SOSP’95)*. 1995 (cited on page 127).

[96] Z. Erkin and G. Tsudik. “Private Computation of Spatial and Temporal Power Consumption with Smart Meters”. In: *10th International Conference on Applied Cryptography and Network Security (ACNS’12)*. 2012, pp. 561–577 (cited on pages 96 and 110).

[97] European Telecommunications Standards Institute (ETSI). *ETSI GS OSG 001 - Open Smart Grid Protocol*. 2012. URL: `http://www.etsi.org/deliver/etsi%5C_gs/OSG/001%5C_099/001/01.01.01%5C_60/gs%5C_osg001v010101p.pdf` (cited on page 109).

[98] H. S. Fhom and K. M. Bayarou. “Towards a Holistic Privacy Engineering Approach for Smart Grid Systems”. In: *Proc. IEEE International Conference on Trust, Security and Privacy in Computing and Communications*. 2011 (cited on page 54).

[99] S. Finster. “Smart Meter Speed Dating, short-term relationships for improved privacy in Smart Metering”. In: *2013 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. 2013 (cited on pages 96 and 110).

[100] S. Finster and I. Baumgart. “Privacy-Aware Smart Metering: A Survey”. In: *IEEE Communications Surveys & Tutorials* PP.99 (2014), pp. 1–14 (cited on page 89).

[101] S. Finster and I. Baumgart. “Pseudonymous Smart Metering without a Trusted Third Party”. In: *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*. 2013 (cited on pages 42, 81, 212, and 257).

[102] L. Fischer, S. Katzenbeisser, and C. Eckert. “Measuring unlinkability revisited”. In: *Proc. ACM workshop on Privacy in the electronic society (WPES’08)*. 2008 (cited on page 56).

[103] C. Fournet and M. Abadi. “Hiding Names: Private Authentication in the Applied Pi Calculus”. In: *Proc. Mext-NSF-JSPS international conference on Software security: theories and systems*. Ed. by M. Okada et al. Lecture Notes in Computer Science. 2003 (cited on pages 56 and 193).

[104] A. Francillon et al. “Systematic Treatment of Remote Attestation”. In: *Cryptology ePrint Archive* (2012) (cited on page 158).

[105] M. Franz, B. Meyer, and A. Pashalidis. “Attacking Unlinkability: The Importance of Context”. In: *Privacy Enhancing Technologies*. Lecture Notes in Computer Science. 2007, pp. 1–16 (cited on page 56).

[106] J. C. Fuller, K. P. Schneider, and D. Chassin. “Analysis of Residential Demand Response and double-auction markets”. In: *2011 IEEE Power and Energy Society General Meeting*. 2011 (cited on pages 88, 89, and 90).

[107] F. D. Garcia and B. Jacobs. “Privacy-Friendly Energy-Metering via Homomorphic Encryption”. In: *Proceedings of the 6th international conference on Security and trust management*. Ed. by J. Cuellar et al. Lecture Notes in Computer Science. 2011 (cited on pages 45, 49, 96, 110, and 212).

[108] S. Garfinkel, G. Spafford, and A. Schwartz. *Practical Unix & Internet Security, 3rd Edition*. O’Reilly Media, Inc., 2003 (cited on page 21).

[109] Y. Gasmi et al. “Beyond secure channels”. In: *Proceedings of the 2007 ACM workshop on Scalable trusted computing - STC ’07*. 2007 (cited on pages 168, 169, 171, 175, and 183).

[110] B. Gedik and L. Liu. “Location Privacy in Mobile Systems: A Personalized Anonymization Model”. In: *Proc. IEEE International Conference on Distributed Computing Systems (ICDCS)*. 2005 (cited on pages 28 and 197).

[111] B. Gedik and L. Liu. “Protecting Location Privacy with Personalized k-Anonymity: Architecture and Algorithms”. In: *IEEE Transactions on Mobile Computing* 7.1 (2008), pp. 1–18 (cited on pages 78, 79, 197, and 253).

[112] C. Gentry. “A Fully Homomorphic Encryption Scheme”. PhD thesis. 2009. URL: `papers2://publication/uuid/894E7CEA-E13A-4034-9EC1-317B0D47C79D` (cited on page 204).

[113] C. Gentry. “Fully homomorphic encryption using ideal lattices”. In: *Proceedings of the 41st annual ACM symposium on Symposium on theory of computing - STOC ’09*. 2009 (cited on pages 19 and 204).

[114] G. Ghinita, P. Kalnis, and S. Skiadopoulos. “PRIVE: Anonymous Location-Based Queries in Distributed Mobile Systems”. In: *Proceedings of the 16th international conference on World Wide Web - WWW ’07*. 2007 (cited on page 197).

[115] B. Glas et al. “A System Architecture for Reconfigurable Trusted Platforms”. In: *2008 Design, Automation and Test in Europe*. 2008 (cited on page 143).

[116] K. Goldman, R. Perez, and R. Sailer. “Linking remote attestation to secure tunnel endpoints”. In: *Proceedings of the first ACM workshop on Scalable trusted computing - STC ’06*. 2006 (cited on page 163).

[117] O. Goldreich. *Foundations of Cryptography: Volume 2, Basic Applications*. Vol. 2. Cambridge University Press, 2009 (cited on pages 55 and 57).

[118] S. Goldwasser and S. Micali. “Probabilistic encryption”. In: *Journal of Computer and System Sciences* 28.2 (1984), pp. 270–299 (cited on pages 63 and 64).

[119] S. Goldwasser and S. Micali. “Probabilistic encryption & how to play mental poker keeping secret all partial information”. In: *Proc. ACM symposium on Theory of computing (STOC’82)*. 1982 (cited on page 64).

[120] P. Golle. “Revisiting the uniqueness of simple demographics in the US population”. In: *Proceedings of the 5th ACM workshop on Privacy in electronic society (WPES’06)*. 2006 (cited on page 13).

[121] D. Grawrock. *Dynamics of a Trusted Platform: A building block approach*. Intel Press, 2009 (cited on page 125).

[122] U. Greveler et al. “Multimedia Content Identification Through Smart Meter Power Usage Profiles”. In: *5th International Conference on Computers, Privacy and Data Protection* (2012) (cited on page 60).

[123] M. Gruteser and D. Grunwald. “Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking”. In: *Proc. International conference on Mobile systems, applications and services*. MobiSys ’03. 2003 (cited on pages 28, 196, and 197).

[124] G. W. Hart. “Nonintrusive appliance load monitoring”. In: *Proceedings of the IEEE* 80.12 (Dec. 1992), pp. 1870–1891 (cited on pages 4 and 39).

[125] G. W. Hart. “Residential energy monitoring and computerized surveillance via utility power flows”. In: *Technology and Society Magazine, IEEE* 8.2 (June 1989), pp. 12–16 (cited on pages 4, 39, and 60).

[126] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985 (cited on page 70).

[127] C. A. R. Hoare and J. Davies. *Communicating Sequential Processes*. 2004. URL: http://www.usingcsp.com/cspbook.pdf (cited on page 70).

[128] D. Huang et al. “MobiCloud: Building Secure Cloud Framework for Mobile Computing and Communication”. In: *2010 Fifth IEEE International Symposium on Service Oriented System Engineering*. 2010 (cited on page 206).

[129] IBM Research. *Integrity Measurement Architecture*. URL: http://researcher.watson.ibm.com/researcher/view%5C_project.php?id=2851 (cited on page 23).

[130] IEEE Standards Association. *IEEE 2030-2011: IEEE Guide for Smart Grid Interoperability of Energy Technology and Information Technology Operation with the Electric Power System (EPS), End-Use Applications , and Loads*. 2011 (cited on page 3).

[131] IETF. *RFC4949: Internet Security Glossary, Version 2*. Tech. rep. 2. 2007. URL: http://tools.ietf.org/html/rfc4949 (cited on page 21).

[132] Intel. *Intel Trusted Execution Technology (Intel TXT): Measured Launch Environment Developer's Guide*. 2015. URL: http://www.intel.co.uk/content/dam/www/public/us/en/documents/guides/intel-txt-software-development-guide.pdf (cited on pages 25, 138, 139, and 171).

[133] Intel Corporation. *Intel Software Guard Extensions Programming Reference*. 2014. URL: https://software.intel.com/sites/default/files/managed/48/88/329298-002.pdf (cited on pages 26, 131, and 143).

[134] International Electrotechnical Commission. *IEC 62056-5-3:2013: Electricity metering data exchange - The DLMS/COSEM suite: Part 5-3: DLMS/COSEM application layer*. 2014 (cited on page 104).

[135] International Electrotechnical Commission. *IEC 62056-6-1:2013: Electricity metering data exchange - The DLMS/COSEM suite Part 6-1: Object Identification System (OBIS)*. 2013 (cited on page 104).

[136] International Telecommunication Union. *ITU-T Recommendation X.1252 - Baseline identity management terms and definitions*. Tech. rep. International Telecommunication Union, 2010 (cited on page 2).

[137] M. Jawurek. "Privacy in Smart Grids". PhD thesis. 2013 (cited on pages 108 and 110).

[138] M. Jawurek, M. Johns, and F. Kerschbaum. "Plug-in privacy for smart metering billing". In: *Proceedings of the 11th international conference on Privacy enhancing technologies*. PETS'11. 2011 (cited on pages 44, 99, and 212).

[139] M. Jawurek, M. Johns, and K. Rieck. "Smart metering de-pseudonymization". In: *Proc. Computer Security Applications Conference*. ACSAC '11. 2011 (cited on pages 42, 50, 60, and 80).

[140] M. Jawurek, F. Kerschbaum, and G. Danezis. "SoK : Privacy Technologies for Smart Grids - A Survey of Options". In: (2013) (cited on page 40).

[141] C. Jensen, H. Lu, and M. L. Yiu. "Location Privacy Techniques in Client-Server Architectures". In: *Privacy in Location-Based Applications*. Ed. by C. Bettini et al. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2009, pp. 31–58 (cited on pages 195 and 197).

[142] Y. Jiang et al. "Mutual Authentication and Key Exchange Protocols for Roaming Services in Wireless Mobile Networks". In: *IEEE Transactions on Wireless Communications* 5.9 (Sept. 2006), pp. 2569–2577 (cited on page 202).

[143] P. Jovanovic and S. Neves. *Dumb Crypto in Smart Grids: Practical Cryptanalysis of the Open Smart Grid Protocol*. Tech. rep. 2015. URL: https://eprint.iacr.org/2015/428 (cited on page 109).

[144] P. Kalnis et al. “Preventing Location-Based Identity Inference in Anonymous Spatial Queries”. In: *IEEE Transactions on Knowledge and Data Engineering* 19.12 (Dec. 2007), pp. 1719–1733 (cited on pages 196 and 197).

[145] G. Kalogridis, Z. Fan, and S. Basutkar. “Affordable Privacy for Home Smart Meters”. In: *Parallel and Distributed Processing with Applications Workshops (ISPAW), 2011 Ninth IEEE International Symposium on*. 2011 (cited on page 49).

[146] G. Kalogridis and S. Dave. “PeHEMS: Privacy enabled HEMS and load balancing prototype”. In: *2012 IEEE Third International Conference on Smart Grid Communications (SmartGridComm)*. 2012 (cited on page 49).

[147] G. Kalogridis et al. “Privacy for Smart Meters: Towards Undetectable Appliance Load Signatures”. In: *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*. 2010 (cited on page 49).

[148] M. Karwe and J. Strüker. “Maintaining Privacy in Data Rich Demand Response Applications”. In: *First Open EIT ICT Labs Workshop on Smart Grid Security (SmartGridSec12)*. Ed. by J. Cuellar. Lecture Notes in Computer Science. 2013, pp. 85–95 (cited on pages 83 and 84).

[149] B. Kauer. “OSLO: improving the security of trusted computing”. In: *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*. SS’07. 2007 (cited on pages 139 and 179).

[150] J. Kilian. “Founding crytpography on oblivious transfer”. In: *Proceedings of the twentieth annual ACM symposium on Theory of computing (STOC’88)*. 1988 (cited on pages 19 and 204).

[151] G. Klein et al. “seL4: Formal Verification of an OS Kernel Gerwin”. In: *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles - SOSP ’09*. 2009 (cited on page 147).

[152] M. Kohlweiss et al. “Anonymity-Preserving Public-Key Encryption: A Constructive Approach”. In: *13th International Symposium on Privacy Enhancing Technologies (PETS 2013)*. Ed. by E. Cristofaro and M. Wright. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013, pp. 19–39 (cited on page 63).

[153] K. Kostiainen, N. Asokan, and J.-E. Ekberg. “Practical property-based attestation on mobile devices”. In: *Proceedings of the 4th international conference on Trust and trustworthy computing - TRUST ’11*. 2011 (cited on pages 158 and 163).

[154] S. Kremer and M. Ryan. “Analysis of an Electronic Voting Protocol in the Applied Pi Calculus”. In: *14th European Symposium on Programming (ESOP’05)*. Ed. by M. Sagiv. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, pp. 186–200 (cited on page 56).

[155] U. Kühn et al. “Secure data management in trusted computing”. In: *Proceedings of the 7th international conference on Cryptographic hardware and embedded systems*. CHES’05. 2005 (cited on page 23).

[156] K. Kursawe, G. Danezis, and M. Kohlweiss. “Privacy-friendly aggregation for the smart-grid”. In: *Proceedings of the 11th international conference on Privacy enhancing technologies*. PETS’11. 2011 (cited on pages 47, 96, 110, and 212).

[157] M. Kuzlu, M. Pipattanasomporn, and S. Rahman. “Hardware Demonstration of a Home Energy Management System for Demand Response Applications”. In: *IEEE Transactions on Smart Grid* 3.4 (Dec. 2012), pp. 1704–1711 (cited on page 35).

[158] N. Leavitt. “Internet Security under Attack: The Undermining of Digital Certificates”. In: *Computer* 44.12 (2011), pp. 17–20 (cited on page 28).

[159] S. Lee et al. “πBox: A Platform for Privacy-Preserving Apps.” In: *Proceedings of the 10th USENIX conference on Networked Systems Design and Implementation (NSDI’13)*. 2013 (cited on page 219).

[160] A. Lee-Thorp. *Attestation in Trusted Computing: Challenges and Potential Solutions*. Tech. rep. Royal Holoway, University of London, 2010. URL: `https://repository.royalholloway.ac.uk/items/49558ca0-a73b-9550-886b-214165f08563/1/` (cited on pages 158 and 159).

[161] M. LeMay et al. “Unified Architecture for Large-Scale Attested Metering”. In: *2007 40th Annual Hawaii International Conference on System Sciences (HICSS’07)*. 2007 (cited on pages 50 and 163).

[162] F. Li, B. Luo, and P. Liu. “Secure Information Aggregation for Smart Grids Using Homomorphic Encryption”. In: *2010 First IEEE International Conference on Smart Grid Communications*. 2010 (cited on pages 46, 96, 110, and 212).

[163] N. Li, T. Li, and S. Venkatasubramanian. “t-Closeness: Privacy Beyond k-Anonymity and l-Diversity”. In: *23rd IEEE International Conference on Data Engineering*. 2007 (cited on pages 15 and 103).

[164] Y. Lindell and B. Pinkas. “Secure multiparty computation for privacy-preserving data mining”. In: *Journal of Privacy and Confidentiality* 1.1 (2009), p. 5 (cited on pages 18, 19, 20, and 207).

[165] M. A. Lisovich and S. B. Wicker. “Privacy Concerns in Upcoming Residential and Commercial Demand-Response Systems”. In: *IEEE Proceedings on Power Systems* 1.1 (2008), pp. 1–10 (cited on page 39).

[166] H. Löhr et al. “Enhancing Grid Security Using Trusted Virtualization”. In: *4th International Conference on Autonomic and Trusted Computing (ATC’07)*. Ed. by B. Xiao et al. Lecture Notes in Computer Science. 2007 (cited on pages 164 and 168).

[167] G. Lowe. “Casper: a compiler for the analysis of security protocols”. In: *Proc. Computer Security Foundations Workshop*. 1997 (cited on pages 55, 56, and 70).

[168] G. Lowe. “Breaking and fixing the Needham-Schroeder Public-Key Protocol using FDR”. In: *Second International Workshop on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'96)*. Ed. by T. Margaria and B. Steffen. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1996, pp. 147–166 (cited on pages 56, 70, and 72).

[169] G. Lowe. *Casper : A Compiler for the Analysis of Security Protocols*. 2009. URL: `http://www.cs.ox.ac.uk/gavin.lowe/Security/Casper/` (cited on page 70).

[170] R. Lu et al. “EPPA: An Efficient and Privacy-Preserving Aggregation Scheme for Secure Smart Grid Communications”. In: *IEEE Transactions on Parallel and Distributed Systems* 23.9 (Sept. 2012), pp. 1621–1631 (cited on pages 46, 110, and 212).

[171] A. T. Luu et al. “SeVe: automatic tool for verification of security protocols”. In: *Frontiers of Computer Science* 6.1 (2012), pp. 57–75 (cited on page 56).

[172] J. Lyle. “Trustworthy Services Through Attestation”. PhD thesis. University of Oxford, 2011 (cited on pages 24, 25, 29, and 129).

[173] J. Lyle and A. Martin. “Engineering attestable services”. In: *Proceedings of the 3rd international conference on Trust and trustworthy computing (TRUST '10)*. 2010 (cited on pages 25, 129, 140, and 141).

[174] J. Lyle and A. Martin. “On the Feasibility of Remote Attestation for Web Services”. In: *International Conference on Computational Science and Engineering*. 2009 (cited on page 5).

[175] A. Machanavajjhala et al. “l-Diversity: Privacy Beyond k-Anonymity”. In: *ACM Transactions on Knowledge Discovery from Data* 1.1 (Mar. 2007) (cited on pages 13, 14, and 103).

[176] A. Madhavapeddy et al. “Unikernels: library operating systems for the cloud”. In: *ACM SIGARCH Computer Architecture News* 41.1 (May 2013), pp. 461–472 (cited on pages 126 and 147).

[177] J. D. Mankowitz and A. J. Paverd. “Mobile device-based cellular network coverage analysis using crowd sourcing”. In: *IEEE EUROCON - International Conference on Computer as a Tool*. 2011 (cited on page 195).

[178] A. Martin. *The ten-page introduction to Trusted Computing*. Tech. rep. University of Oxford Computing Laboratory, 2008 (cited on pages 21 and 23).

[179] M. Masmano et al. “TLSF: a new dynamic memory allocator for real-time systems”. In: *Proceedings. 16th Euromicro Conference on Real-Time Systems, 2004. ECRTS 2004.* 2004 (cited on page 136).

[180] S. Mauw, J. H. S. Verschuren, and E. P. Vink. “A Formalization of Anonymity and Onion Routing”. In: *9th European Symposium on Research in Computer Security (ESORICS'04)*. Ed. by P. Samarati et al. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2004, pp. 109–124 (cited on pages 54 and 56).

[181] J. M. McCune et al. “Flicker: an execution infrastructure for TCB minimization”. In: *Proceedings of the 3rd ACM SIGOPS/EuroSys European Conference on Computer Systems (Eurosys'08)*. 2008 (cited on pages 25, 129, 130, 144, 147, and 185).

[182] J. M. McCune et al. “TrustVisor: Efficient TCB Reduction and Attestation”. In: *IEEE Symposium on Security and Privacy*. 2010 (cited on pages 129, 130, 141, 148, 163, 171, 176, 177, and 216).

[183] F. McKeen et al. “Innovative instructions and software model for isolated execution”. In: *Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy (HASP'13)*. 2013 (cited on pages 26, 131, and 143).

[184] S. McLaughlin, P. McDaniel, and W. Aiello. “Protecting consumer privacy from electric load monitoring”. In: *Proceedings of the 18th ACM conference on Computer and communications security*. CCS '11. 2011 (cited on page 49).

[185] S. Meier et al. “The TAMARIN Prover for the Symbolic Analysis of Security Protocols”. In: *Proceedings of the 25th international conference on Computer Aided Verification (CAV'13)*. Ed. by N. Sharygina and H. Veith. Lecture Notes in Computer Science. 2013 (cited on page 56).

[186] A. Meyerson and R. Williams. “On the complexity of optimal K-anonymity”. In: *Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS'04)*. 2004 (cited on page 13).

[187] S. C. Misra and V. C. Bhavsar. “Relationships between selected software measures and latent bug-density: guidelines for improving quality”. In: *Proceedings of the 2003 international conference on Computational science and its applications*. 2003 (cited on page 129).

[188] P. Mohan, V. N. Padmanabhan, and R. Ramjee. “Nericell: Rich Monitoring of Road and Traffic Conditions using Mobile Smartphones”. In: *Proceedings of the 6th ACM conference on Embedded network sensor systems (SenSys'08)*. 2008 (cited on page 195).

[189] M. F. Mokbel, C.-Y. Chow, and W. G. Aref. “The new Casper: query processing for location services without compromising privacy”. In: *Procedings of the International Conference on Very Large Databases*. VLDB '06. 2006 (cited on pages 28 and 197).

[190] A. Molina-markham et al. “Designing Privacy-preserving Smart Meters with Low-cost Microcontrollers”. In: *16th International Conference on Financial Cryptography and Data Security*. 2012 (cited on page 112).

[191] A. Molina-Markham et al. “Private memoirs of a smart meter”. In: *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building*. BuildSys ’10. 2010 (cited on pages 28, 48, 60, and 112).

[192] Y.-A. de Montjoye, L. Radaelli, and V. K. Singh. “Unique in the shopping mall: On the reidentifiability of credit card metadata”. In: *Science* 347.6221 (2015), pp. 536–539 (cited on pages 13 and 61).

[193] Y.-A. de Montjoye et al. “Unique in the Crowd: The privacy bounds of human mobility.” In: *Scientific reports* 3 (Jan. 2013), p. 1376 (cited on pages 13, 61, and 196).

[194] M. Moran, J. Heather, and S. Schneider. “Verifying anonymity in voting systems using CSP”. In: *Formal Aspects of Computing* 26.1 (2012), pp. 63–98 (cited on page 55).

[195] T. Moyer et al. “Scalable Web Content Attestation”. In: *IEEE Transactions on Computers* 61.5 (May 2012), pp. 686–699 (cited on pages 166 and 178).

[196] A. Nagarajan et al. “Property Based Attestation and Trusted Computing: Analysis and Challenges”. In: *Third International Conference on Network and System Security*. 2009 (cited on page 158).

[197] C. Namiluko, A. J. Paverd, and T. De Souza. “Towards Enhancing Web Application Security Using Trusted Execution”. In: *Workshop on Web Applications and Secure Hardware - WASH’13*. 2013 (cited on page 185).

[198] National Institute of Standards and Technology (NIST). *NIST Special Publication 1108R2: NIST Framework and Roadmap for Smart Grid Interoperability Standards, Release 2.0*. Tech. rep. 2012 (cited on pages 3 and 32).

[199] NIST Smart Grid Interoperability Panel - Cyber Security Working Group. *NISTIR 7628: Guidelines for Smart Grid Cyber Security: Vol. 2, Privacy and the Smart Grid*. Tech. rep. August. 2010 (cited on page 51).

[200] D. Niyato, L. Xiao, and P. Wang. “Machine-to-machine communications for home energy management system in smart grid”. In: *IEEE Communications Magazine* 49.4 (Apr. 2011), pp. 53–59 (cited on page 35).

[201] A. Nizar, Z. Dong, and J. Zhao. “Load profiling and data mining techniques in electricity deregulated market”. In: *2006 IEEE Power Engineering Society General Meeting*. 2006 (cited on page 92).

[202] OASIS. *Energy Interoperation Version 1.0*. 2013. URL: `http://docs.oasis-open.org/energyinterop/ei/v1.0/cos01/energyinterop-v1.0-%20cos01.html`. (cited on page 38).

[203] J. Oberheide et al. “Virtualized in-cloud security services for mobile devices”. In: *Proceedings of the First Workshop on Virtualization in Mobile Computing (MobiVirt’08)*. 2008 (cited on page 206).

[204] C. Orlandi. "Is multiparty computation any good in practice?" In: *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2011 (cited on page 20).

[205] P. Paillier. "Public-key cryptosystems based on composite degree residuosity classes". In: *Proceedings of the 17th international conference on Theory and application of cryptographic techniques (EUROCRYPT'99)*. 1999 (cited on page 19).

[206] A. J. Paverd and A. P. Martin. "Hardware Security for Device Authentication in the Smart Grid". In: *First Open EIT ICT Labs Workshop on Smart Grid Security (SmartGridSec12)*. 2012 (cited on pages 157 and 185).

[207] A. J. Paverd, A. P. Martin, and I. Brown. "Privacy-Enhanced Bi-Directional Communication in the Smart Grid using Trusted Computing". In: *Fifth IEEE International Conference on Smart Grid Communications (SmartGridComm'14)*. 2014 (cited on page 87).

[208] A. J. Paverd, A. P. Martin, and I. Brown. "Security and Privacy in Smart Grid Demand Response Systems". In: *Second Open EIT ICT Labs Workshop on Smart Grid Security (SmartGridSec14)*. 2014 (cited on pages 31, 40, 83, and 87).

[209] A. J. Paverd, A. Martin, and I. Brown. *Modelling and Automatically Analysing Privacy Properties for Honest-but-Curious Adversaries*. Tech. rep. 2014. URL: `https://www.cs.ox.ac.uk/people/andrew.paverd/casper/casper-privacy-report.pdf` (cited on page 53).

[210] A. J. Paverd. *Enhanced Mobile Computing Using Cloud Resources*. 2011. URL: `http://open.uct.ac.za/handle/11427/11063` (cited on page 206).

[211] R. Petrlic. "A privacy-preserving Concept for Smart Grids". In: *Sicherheit in vernetzten Systemen: 18. DFN Workshop*. 2010 (cited on page 50).

[212] A. Pfitzmann and M. Hansen. *A Terminology for talking about privacy by data minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management*. Tech. rep. Aug. 2010. URL: `http://dud.inf.tu-dresden.de/literatur/Anon%5C_Terminology%5C_v0.34.pdf` (cited on pages 54, 59, 60, and 192).

[213] M. A. Piette et al. *Open Automated Demand Response Communications Specification (Version 1.0)*. Tech. rep. April. California Energy Commission, PIER Program, 2009 (cited on pages 38, 83, 84, and 261).

[214] M. Pipattanasomporn, M. Kuzlu, and S. Rahman. "An Algorithm for Intelligent Home Energy Management and Demand Response Analysis". In: *IEEE Transactions on Smart Grid* 3.4 (2012), pp. 2166–2173 (cited on page 35).

[215] M. Pirker et al. "A PrivacyCA for Anonymity and Trust". In: *Proc. International conference on Trust and trustworthy computing (TRUST '09)*. Lecture Notes in Computer Science. 2009 (cited on pages 28 and 29).

[216] J. Poritz et al. *Property Attestation - Scalable and Privacy-friendly Security Assessment of Peer Computers*. Tech. rep. IBM Research, 2004 (cited on pages 158 and 163).

[217] G. Proudler. "Concepts of Trusted Computing". In: *Trusted Computing*. Ed. by C. J. Mitchell. 2005. Chap. 2, pp. 11–13 (cited on pages 21, 119, 122, and 123).

[218] E. L. Quinn. "Privacy and the New Energy Infrastructure". In: *Privacy and the New Energy Infrastructure* (2009) (cited on pages 39 and 60).

[219] M. O. Rabin. *How To Exchange Secrets with Oblivious Transfer*. Tech. rep. Aiken Computation Lab, Harvard University, 1981. URL: `https://eprint.iacr.org/2005/187.pdf` (cited on pages 19 and 204).

[220] V. Rastogi and S. Nath. "Differentially private aggregation of distributed time-series with transformation and encryption". In: *Proceedings of the 2010 international conference on Management of data (SIGMOD '10)*. 2010 (cited on page 110).

[221] A. Rial and G. Danezis. "Privacy-preserving smart metering". In: *Proceedings of the 10th annual ACM workshop on Privacy in the electronic society (WPES'11)*. 2011 (cited on pages 43, 99, 111, and 212).

[222] A. Roscoe. "Model-checking CSP". In: *A classical mind: essays in honour of CAR Hoare*. 1994 (cited on page 70).

[223] A. Roscoe. "Modelling and verifying key-exchange protocols using CSP and FDR". In: *Proceedings of the IEEE Computer Security Foundations Workshop*. 1995 (cited on pages 56, 70, and 72).

[224] C. Rottondi, G. Mauri, and G. Verticale. "A data pseudonymization protocol for Smart Grids". In: *IEEE Online Conference on Green Communications (GreenCom'12)*. 2012 (cited on page 42).

[225] C. Rottondi and G. Verticale. "Privacy-friendly appliance load scheduling in smart grids". In: *IEEE International Conference on Smart Grid Communications (SmartGridComm'13)*. 2013 (cited on page 113).

[226] C. Rottondi, G. Verticale, and A. Capone. "Privacy-preserving smart metering with multiple data Consumers". In: *Computer Networks* 57.7 (2013), pp. 1699–1713 (cited on pages 54, 96, and 110).

[227] C. Rottondi, G. Verticale, and C. Krauss. "Distributed Privacy-Preserving Aggregation of Metering Data in Smart Grids". In: *IEEE Journal on Selected Areas in Communications* 31.7 (July 2013), pp. 1342–1354 (cited on page 110).

[228] I. Roy et al. "Airavat: security and privacy for MapReduce". In: *Proceedings of the 7th USENIX conference on Networked systems design and implementation (NSDI'10)*. 2010 (cited on page 219).

[229] S. Ruj, A. Nayak, and I. Stojmenovic. *A Security Architecture for Data Aggregation and Access Control in Smart Grids*. Tech. rep. Dlc. 2011, pp. 1–12. URL: `http://arxiv.org/pdf/1111.2619v1` (cited on page 110).

[230] A.-R. Sadeghi and S. Schulz. "Extending IPsec for Efficient Remote Attestation". In: *Financial Cryptography and Data Security*. Ed. by R. Sion et al. Springer Berlin Heidelberg, 2010, pp. 150–165 (cited on page 163).

[231] A.-R. Sadeghi and C. Stuble. "Property-based attestation for computing platforms: caring about properties, not mechanisms". In: *Proceedings of the 2004 workshop on New security paradigms (NSPW'04)*. 2004 (cited on pages 127 and 158).

[232] R. Sailer et al. "Design and implementation of a TCG-based integrity measurement architecture". In: *Proceedings of the 13th conference on USENIX Security Symposium*. USENIX Association. 2004 (cited on pages 23, 140, 158, 162, and 178).

[233] D. Samfat, R. Molva, and N. Asokan. "Untraceability in mobile networks". In: *Proceedings of the 1st annual international conference on Mobile computing and networking (MobiCom'95)*. 1995 (cited on pages 201 and 204).

[234] N. Saputro and K. Akkaya. "Performance evaluation of Smart Grid data aggregation via homomorphic encryption". In: *IEEE Wireless Communications and Networking Conference (WCNC'12)*. 2012 (cited on pages 46 and 212).

[235] F. Schuster et al. *VC3: Trustworthy Data Analytics in the Cloud*. Tech. rep. MSR-TR-2014-39. Feb. 2014. URL: `http://research.microsoft.com/apps/pubs/default.aspx?id=210786` (cited on page 27).

[236] SecureState. *Termineter Framework: Open Source Smart Meter Hacking Tool*. 2012. URL: `http://blog.securestate.com/termineter-framework-open-source-smart-meter-hacking-tool/` (cited on page 82).

[237] A. Seshadri et al. "SecVisor: A Tiny Hypervisor to Provide Lifetime Kernel Code Integrity for Commodity OSes". In: *Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles (SOSP'07)*. 2007 (cited on page 141).

[238] E. Shi et al. "Privacy-Preserving Aggregation of Time-Series Data". In: *Proceedings of the 18th Annual Network and Distributed System Security (NDSS'11)*. 2011 (cited on pages 47, 110, and 212).

[239] H. Simo Fhom et al. "A user-centric privacy manager for future energy systems". In: *International Conference on Power System Technology (POWERCON)*. 2010 (cited on page 50).

[240] D. X. Song, S. Berezin, and A. Perrig. "Athena: a novel approach to efficient automatic security protocol analysis". In: *Journal of Computer Security* 9.1 (2001), pp. 47–74 (cited on page 56).

[241] T. de Souza et al. “Set Difference Attacks in Wireless Sensor Networks”. In: *8th International ICST Conference, SecureComm 2012*. 2012 (cited on pages 16, 96, and 109).

[242] E. R. Sparks. *A Security Assessment of Trusted Platform Modules*. Tech. rep. 2007. URL: `http://129.170.213.101/reports/TR2007-597.pdf` (cited on page 139).

[243] M. Stegelmann and D. Kesdogan. “GridPriv: A Smart Metering Architecture Offering k-Anonymity”. In: *Proceedings of the IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. 2012 (cited on page 60).

[244] S. Steinbrecher and S. Köpsell. “Modelling Unlinkability”. In: *Third International Workshop on Privacy Enhancing Technologies (PET'03)*. Lecture Notes in Computer Science. 2003, pp. 32–47 (cited on page 56).

[245] S. G. Stubblebine, P. F. Syverson, and D. M. Goldschlag. “Unlinkable serial transactions: protocols and applications”. In: *ACM Transactions on Information Systems Security* 2.4 (Nov. 1999), pp. 354–389 (cited on page 55).

[246] F. Stumpf et al. “A Robust Integrity Reporting Protocol for Remote Attestation”. In: *Second Workshop on Advances in Trusted Computing (WATC'06)*. 2006 (cited on pages 163, 164, 165, 167, and 175).

[247] F. Stumpf et al. “Improving the scalability of platform attestation”. In: *Proceedings of the 3rd ACM workshop on Scalable trusted computing (STC'08)*. 2008 (cited on pages 158, 165, 167, 175, 177, and 183).

[248] J. Sun, Y. Liu, and J. Dong. “Model Checking CSP Revisited: Introducing a Process Analysis Toolkit”. In: *Third International Symposium on Leveraging Applications of Formal Methods, Verification and Validation*. Ed. by T. Margaria and B. Steffen. Communications in Computer and Information Science. Springer Berlin Heidelberg, 2008, pp. 307–322 (cited on page 177).

[249] L. Sweeney. “k-Anonymity: A Model for Protecting Privacy”. In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10.05 (Oct. 2002), pp. 557–570 (cited on pages 13 and 197).

[250] L. Sweeney. *Simple demographics often identify people uniquely*. Tech. rep. 2000. URL: `http://dataprivacylab.org/projects/identifiability/paper1.pdf` (cited on page 13).

[251] P. Syverson et al. “Towards an Analysis of Onion Routing Security”. In: *International Workshop on Design Issues in Anonymity and Unobservability*. Ed. by H. Federrath. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2001, pp. 96–114 (cited on page 56).

[252] K. Thompson. “Reflections on trusting trust”. In: *Communications of the ACM* 27.8 (Aug. 1984), pp. 761–763 (cited on page 127).

[253] R. Toegl, M. Pirker, and M. Gissing. "acTvSM: A Dynamic Virtualization Platform for Enforcement of Application Integrity". In: *Proceedings of the 2nd International Conference on Trusted Systems*. 2011 (cited on pages 25 and 129).

[254] Trusted Computing Group. *TCG Trusted Network Connect*. 2012. URL: http://www.trustedcomputinggroup.org/resources/tnc%5C_architecture%5C_for%5C_interoperability%5C_specification (cited on pages 29 and 160).

[255] Trusted Computing Group. *TPM Main Specifications, Version 1.2, Revision 116, Part 1: Design Principles*. 2011 (cited on pages 21, 22, 158, and 169).

[256] Trusted Computing Group. *TPM Main Specifications, Version 1.2, Revision 116, Part 2 TPM Structures*. 2011 (cited on pages 21 and 22).

[257] Trusted Computing Group. *TPM Main Specifications, Version 1.2, Revision 116, Part 3: Commands*. 2011 (cited on pages 21 and 22).

[258] Trusted Computing Group. *Trusted Platform Module Library Part 1: Architecture*. 2013 (cited on pages 22 and 23).

[259] Trusted Computing Group. *Trusted Platform Module Library Part 2: Structures*. 2013 (cited on page 22).

[260] Trusted Computing Group. *Trusted Platform Module Library Part 3: Commands*. 2013 (cited on page 22).

[261] Trusted Computing Group. *Trusted Platform Module Library Part 4: Supporting Routines*. 2013 (cited on page 22).

[262] Trusted Computing Group Infrastructure Workgroup. *Subject Key Attestation Evidence Extension*. 2005 (cited on page 168).

[263] TrustInSoft. *PolarSSL 1.1.8 verification kit*. Tech. rep. 2015. URL: http://trust-in-soft.com/polarSSL%5C_demo.pdf (cited on page 147).

[264] United Kingdom Department of Energy & Climate Change. *Digest of United Kingdom Energy Statistics (DUKES) 2015*. 2015 (cited on page 33).

[265] United Kingdom Department of Energy & Climate Change. *Smart Metering Equipment Technical Specifications Version 1.1*. 2014. URL: https://www.gov.uk/government/uploads/system/uploads/attachment%5C_data/file/299395/smets.pdf (cited on pages 35, 104, 109, 110, 111, and 132).

[266] United Kingdom Department of Energy & Climate Change. *Smart Metering Equipment Technical Specifications Version 2*. 2013. URL: https://www.gov.uk/government/uploads/system/uploads/attachment%5C_data/file/68898/smart%5C_meters%5C_equipment%5C_technical%5C_spec%5C_version%5C_2.pdf (cited on pages 104, 109, 110, 111, and 132).

[267] United Kingdom Department of Energy & Climate Change. *Smart Metering Implementation Programme Communications Hub Technical Specifications Version 1.46*. 2014. URL: http://www.rightmove.co.uk/house-prices/detailMatching.html?prop=43612978%5C&sale=50972279%5C&country=england (cited on page 110).

[268] United Kingdom Department of Energy & Climate Change. *Smart Metering Implementation Programme Great Britain Companion Specification (GBCS)*. 2014 (cited on page 104).

[269] United Kingdom Department of Energy & Climate Change and Ofgem. *Smart Energy Code - Draft Schedule of Core Communication Services*. 2013. URL: https://www.gov.uk/government/publications/notification-of-core-communication-services-schedule-and-invitation-for-elective-communication-service-requests (cited on page 33).

[270] United Kingdom Smart Grid Forum. *Smart Grid Vision and Routemap*. 2014. URL: https://www.gov.uk/government/uploads/system/uploads/attachment%5C_data/file/285417/Smart%5C_Grid%5C_Vision%5C_and%5C_RoutemapFINAL.pdf (cited on pages 33, 90, and 107).

[271] United States Department of Energy. *Benefits of Demand Reponse in Electricity Markets and Recommendations for Achieving Them*. 2006. URL: http://energy.gov/oe/downloads/benefits-demand-response-electricity-markets-and-recommendations-achieving-them-report (cited on page 37).

[272] M. Veeningen, B. de Weger, and N. Zannone. "Data minimisation in communication protocols: a formal analysis framework and application to identity management". In: *International Journal of Information Security* 13.6 (Apr. 2014), pp. 529–569 (cited on pages 56, 59, 60, and 61).

[273] M. Veeningen, B. Weger, and N. Zannone. "Formal Privacy Analysis of Communication Protocols for Identity Management". In: *Information Systems Security*. Lecture Notes in Computer Science. 2011, pp. 235–249 (cited on pages 54, 56, 59, 60, 61, 76, and 77).

[274] M. Veeningen, B. Weger, and N. Zannone. "Modeling Identity-Related Properties and Their Privacy Strength". In: *Formal Aspects of Security and Trust*. Lecture Notes in Computer Science. 2011, pp. 126–140 (cited on page 56).

[275] D. Wallom et al. "myTrustedCloud: trusted cloud infrastructure for security-critical computation and data management". In: *Proceedings of the WICSA/ECSA 2012 Companion Volume*. WICSA/ECSA '12. 2012 (cited on page 51).

[276] D. Wallom et al. "myTrustedCloud: Trusted Cloud Infrastructure for Security-critical Computation and Data Managment". In: *2011 IEEE Third International Conference on Cloud Computing Technology and Science*. 2011 (cited on page 51).

[277] Z. Wan, K. Ren, and B. Preneel. "A secure privacy-preserving roaming protocol based on hierarchical identity-based encryption for mobile networks". In: *Proceedings of the first ACM conference on Wireless network security (WiSec'08)*. 2008 (cited on page 202).

[278] J. Winter. "Experimenting with ARM TrustZone – Or: How I Met Friendly Piece of Trusted Hardware". In: *IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*. 2012 (cited on page 26).

[279] J. Winter. "Trusted computing building blocks for embedded linux-based ARM trustzone platforms". In: *Proceedings of the 3rd ACM workshop on Scalable trusted computing (STC'08)*. 2008 (cited on page 26).

[280] J. Winter et al. "A flexible software development and emulation framework for ARM TrustZone". In: *Proceedings of the Third international conference on Trusted Systems (INTRUST'11)*. 2011 (cited on page 26).

[281] J. Wright et al. *Formalizing Anonymity: A Review*. Tech. rep. June. University of York Technical Report YCS 389, 2005. URL: `http://eprints.whiterose.ac.uk/72501/1/YCS%5C_2005%5C_389.pdf` (cited on pages 54 and 56).

[282] A. C. Yao. "Protocols for Secure Computations". In: *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*. SFCS '82. 1982 (cited on pages 20 and 204).

[283] A. C.-C. Yao. "How to generate and exchange secrets". In: *27th Annual Symposium on Foundations of Computer Science (SFCS'86)*. 1986 (cited on pages 20 and 204).

[284] H. Zang and J. Bolot. "Anonymization of location data does not work". In: *Proceedings of the 17th annual international conference on Mobile computing and networking - MobiCom '11*. 2011 (cited on page 13).

[285] M. Zeifman and K. Roth. "Nonintrusive appliance load monitoring: Review and outlook". In: *IEEE Transactions on Consumer Electronics* 57.1 (Feb. 2011), pp. 76–84 (cited on pages 4 and 39).

[286] J. J. Zic et al. "Hardware Security Device Facilitated Trusted Energy Services". In: *Mobile Networks and Applications* 17.4 (Mar. 2012), pp. 564–577 (cited on page 49).

Appendix A

Casper-Privacy Protocol Models

This appendix contains the full input scripts for the Casper-Privacy tool representing the protocols analysed in Chapter 4. In many of these models, it should be noted that checking all the undetectability and unlinkability specifications concurrently will incur very high processing and memory requirements. The semantics of the models, and thus the outcomes of the analysis, are unchanged by evaluating only a subset of the privacy specifications in each analysis run.

A.1 TAS^3 Attribute Aggregation Protocol

```
-- Part of TAS3 attribute aggregation protocol
-- Consisting of two identity providers, a linking service and a
   service provider
-- Assuming perfectly anonymous communication network

#Free variables
-- Identity providers 1 and 2, linking service and service provider
idp1, idp2 : IDPAgent
ls : LSAgent
sp : SPAgent
-- session identifier
idsess : SessionID
-- nonce
n1, n2 : Nonce
-- permanent user identifiers (ls <-> idp1) and (ls <-> idp2)
pid1, pid2 : PID
-- user attributes from idp1 and idp2
d1, d2 : Attributes
-- Public and private key pairs
pubidp2 : IDPPublicKey
prividp2 : IDPPrivateKey
publs : LSPublicKey
privls : LSPrivateKey
InverseKeys = (publs,privls), (pubidp2,prividp2)

#Processes
IdentityProvider1(idp1, sp, idsess, d1, ls, pid1, n1, publs)
IdentityProvider2(idp2, d2, pid2, prividp2)
LinkService(ls, idp2, pid1, pid2, n2, pubidp2, privls)
ServiceProvider(sp, idsess)

#Protocol description
1. idp1 -> sp : idsess, d1, ls, {pid1,n1}{publs} % x1
2. sp -> ls : idsess, d1, x1 % {pid1,n1}{publs}
3. ls -> sp : idp2, {pid2,n2}{pubidp2} % x2
4. sp -> idp2 : idsess, d1, x2 % {pid2,n2}{pubidp2}
5. idp2 -> sp : idsess, d2

#Channels
-- Assuming mutually authenticated TLS connections
1 C NF NRA- NR-
2 C NF NRA- NR-
3 C NF NRA- NR-
4 C NF NRA- NR-
5 C NF NRA- NR-


#Specification
Secret(idp1, d1, [sp])
Secret(ls, pid2, [idp2])
Secret(idp2, d2, [sp])
Agreement(idp1, sp, [d1])
Agreement(idp2, sp, [d2])
```

```
------------------------------

#Actual variables
IDP1, IDP2, I : IDPAgent
LS : LSAgent
SP : SPAgent
IDsess1, IDsess2 : SessionID
N1, N2, N3 : Nonce
PID1, PID2 : PID
D1, D2, D3: Attributes
PubIDP2 : IDPPublicKey
PrivIDP2 : IDPPrivateKey
PubLS : LSPublicKey
PrivLS : LSPrivateKey
InverseKeys = (PubLS,PrivLS), (PubIDP2,PrivIDP2)

#Functions

#System
IdentityProvider1(IDP1, SP, IDsess1, D1, LS, PID1, N1, PubLS)
IdentityProvider2(IDP2, D2, PID2, PrivIDP2)
LinkService(LS, IDP2, PID1, PID2, N2, PubIDP2, PrivLS)
ServiceProvider(SP, IDsess1)

#Intruder Information
Intruder = I
IntruderKnowledge = {I, LS, IDP1, IDP2, SP}

#Privacy

-- P1: Pass, Pass
Detectable( SP, {D1,D2} )
Linkable( SP, {D1,D2}, {}, {} )

-- P2: Pass, Fail
Undetectable( IDP1, {PID2,D2} )
Undetectable( IDP2, {PID1,D1} )

-- P3: Fail
Undetectable( LS, {D1,D2} )

-- P4: Pass
Unlinkable( SP, {D1,D3}, {LS}, {} )

-- P5: Fail, Fail
Unlinkable( IDP2, {D1,PID2}, {}, {} )
```

Listing A.1: Casper-Privacy script for the TAS^3 attribute aggregation protocol [57]

A.2 Unlinkability of RFID e-Passports

```
-- Part of Basic Access Control (BAC) protocol of the French
   ePassport
-- Consisting of a tag and a reader

-- WARNING: Very high processing requirements

#Free variables
-- Tag and reader
t, r : Agent
-- nonces generated by the tag and reader
nt : NonceTag
nr : NonceReader
-- Key material generated by the tag and reader
kt : KeyMaterialTag
kr : KeyMaterialReader
-- Keys derived from machine-readable data on the passport
ke : KeyEncrypt
km : KeyMAC
h : HashFunction
InverseKeys = (ke,ke), (km,km)


#Processes
Tag(t, nt, kt, ke, km)
Reader(r, nr, kr, ke, km)


#Protocol description
0.    -> t : r
1. t -> r : nt
2. r -> t : {nr,nt,kr}{ke}, h({nr,nt,kr}{ke})
3. t -> r : {nt,nr,kt}{ke}, h({nt,nr,kt}{ke})


#Channels
-- All confidentiality properties described using public keys in
   protocol


#Specification
Secret(r, kr, [t])
Secret(t, kt, [r])
Agreement(t, r, [kr,kt])


--------------------------------

#Actual variables
T, R, I : Agent
Nt1, Nt2 : NonceTag
Nr1, Nr2 : NonceReader
Kt1, Kt2 : KeyMaterialTag
Kr1, Kr2 : KeyMaterialReader
```

```
Ke : KeyEncrypt
Km : KeyMAC
InverseKeys = (Ke,Ke), (Km,Km)


#Functions


#System
Tag(T, Nt1, Kt1, Ke, Km); Tag(T, Nt2, Kt2, Ke, Km)
Reader(R, Nr1, Kr1, Ke, Km); Reader(R, Nr2, Kr2, Ke, Km)


#Intruder Information
Intruder = I
IntruderKnowledge = {I, T, R}


#Privacy

-- The tag should be linkable between sessions based on Ke and Km
-- even though different keys and nonces are used.
Linkable( R, {Kt1,Kt2}, {}, {} )
```

Listing A.2: Casper-Privacy script for the French ePassport Basic Access Control protocol [17]

A.3 Unlinkability of RFID e-Passports - Error Condition

```
-- Part of Basic Access Control (BAC) protocol of the French
   ePassport
-- Protocol for error 6A80 when message 2 contains correct MAC but
   incorrect nonce
-- Consisting of a tag and a reader

#Free variables
-- Tag and reader
t, hbc : Agent
-- nonces generated by the tag and reader
nt : NonceTag
ni : NonceIncorrect
-- Key material generated by the reader
kr : KeyMaterialReader
-- Error codes used in the system
e6A80 : ErrorCode
-- Keys derived from machine-readable data on the passport
ke : KeyEncrypt
km : KeyMAC
h : HashFunction
InverseKeys = (ke,ke), (km,km)


#Processes
Tag(t, nt, ke, km, e6A80)
Reader(hbc, ni, kr, ke, km)


#Protocol description
0.    -> t : hbc
1. t -> hbc : nt
-- HBC replays message previously overheard
2. hbc -> t : {ni,nt,kr}{ke}, h({ni,nt,kr}{ke})
3. t -> hbc : nt, e6A80


#Channels
-- All confidentiality properties described using public keys in
   protocol


#Specification
Secret(hbc, kr, [t])
Agreement(hbc, t, [kr])


--------------------------------

#Actual variables
T, HBC, I : Agent
Nt1, Nt2 : NonceTag
Ni1, Ni2 : NonceIncorrect
Ki : KeyMaterialReader
```

```
E6A80 : ErrorCode
Ke : KeyEncrypt
Km : KeyMAC
InverseKeys = (Ke,Ke), (Km,Km)


#Functions


#System
Tag(T, Nt1, Ke, Km, E6A80); Tag(T, Nt2, Ke, Km, E6A80)
Reader(HBC, Ni1, Ki, Ke, Km); Reader(HBC, Ni2, Ki, Ke, Km)


#Intruder Information
Intruder = I
IntruderKnowledge = {I, T, HBC}


#Privacy

-- The tag should not be linkable between sessions to an intruder
-- Ke and Km cannot be used for linking since they are not known to
    the intruder
Unlinkable( HBC, {Nt1,Nt2}, {Ke,Km}, {} )
```

Listing A.3: Casper-Privacy script for the Basic Access Control protocol error 6A80 branch [17]

A.4 Protecting location privacy using k-anonymity

```
-- Location-Based Service (LBS) privacyenhancing protocol based on
   Gedik2008
-- Consisting of two mobile devices, one HBC service provider and
   one trusted anonymity server
-- Assuming perfectly anonymous communication network

-- WARNING: This analysis has very high computational requirements!

#Free variables
-- mobile device, service provider and anonymity server
m : Agent
sp : ServiceProvider
as : AnonymityServer
-- user ID (unique to each mobile device e.g. IMSI/IMEI)
id : UserID
-- request number (unique to each request)
rn : RequestNumber
-- request data
rd : RequestData

-- request response
lbd : LBData

k : Key
h : HashFunction
InverseKeys = (k,k)

#Processes
MOBILE(m, as, sp, id, rn, rd)
ANONSERVER(as)
LBSPROVIDER(sp, lbd)

#Protocol description
1. m -> as : m, id, rn, sp, rd
2. as -> sp : h(id,rn) % t, rd
3. sp -> as : t % h(id,rn), lbd
4. as -> m : id, rn, lbd

#Channels
-- Assuming confidentiallity for messages from the mobile (m) to
   the anonymity server (as)
-- based on the use of TLS connections and the public key of as.
-- Assuming confidentially, integrity and authenticity for messages
    from the
-- anonymity server (as) to the service provider (sp)
-- based on mutual TLS authentication.
1 C NF NRA- NR-
2 C NF NRA- NR-
3 C NF NRA- NR-
4 C NF NRA- NR-


#Specification
```

```
Secret(m, id, [as])
Secret(m, rn, [as])
Secret(m, rd, [as, sp])
Secret(sp, lbd, [as])
Agreement(m, as, [id, rn, rd])
Agreement(sp, as, [rd, lbd])

--------------------------------


#Actual variables
-- mobile device, service provider and anonymity server
M1, I : Agent
SP : ServiceProvider
AS : AnonymityServer
-- user ID (unique to each mobile device e.g. IMSI/IMEI)
ID1 : UserID
-- request number (unique to each request)
RN1, RN2 : RequestNumber
-- request data
RD1, RD2 : RequestData
-- request response
LBD1, LBD2 : LBData

K : Key
InverseKeys = (K,K)

#Functions

#System
MOBILE(M1, AS, SP, ID1, RN1, RD1); MOBILE(M1, AS, SP, ID1, RN2, RD2
    )
ANONSERVER(AS); ANONSERVER(AS)
LBSPROVIDER(SP, LBD1); LBSPROVIDER(SP, LBD2)

#Intruder Information
Intruder = I
IntruderKnowledge = {I, M1, AS, SP}

#Privacy
Unlinkable( SP, {ID1,M1}, {}, {} )
Unlinkable( SP, {ID1,RN1}, {}, {} )
Unlinkable( SP, {ID1,RD1}, {}, {} )
Unlinkable( SP, {RD1,RD2}, {}, {} )
```

Listing A.4: Casper-Privacy script for the privacy-enhancing location-based service protocol [111]

A.5 Smart Meter Anonymization through Pseudonyms

```
-- Advanced Metering Infrastructure in the Smart Grid based on the
   protocol in Efthymiou2010
-- Consisting of two homes with smart meters, one energy utility
   and one communication aggregator
-- Assuming perfectly anonymous communication network

#Free variables
-- smart meter, utility and aggregator
sm : Agent
ut : Utility
agg : Aggregator
-- high frequency identifier
hfid : HFID
-- low frequency identifier
lfid : LFID
-- sequential measurements, A then B, from smart meter
ma : HFDataA
mb : HFDataB
-- total energy consumption from smart meter
t : LFData
k : Key
InverseKeys = (k,k)

#Processes
SMARTMETER(sm, agg, ut, hfid, lfid, ma, mb, t)
AGGREGATOR(agg, ut, hfid)
UTILITY(ut, lfid)

#Protocol description
1. sm -> agg : hfid, ma
2. agg -> ut : hfid, ma
3. sm -> agg : hfid, mb
4. agg -> ut : hfid, mb
5. sm -> ut  : lfid, t

#Channels
-- Assuming confidentiallity for messages from the smart meter to
   the aggregator (AGG) and utility (U)
-- based on the use of TLS connections and the public keys of AGG
   and U.
-- Assuming confidentially, integrity and authenticity for messages
    from AGG to U
-- based on mutual TLS authentication.
1 C NF NRA- NR-
2 C NF NRA- NR-
3 C NF NRA- NR-
4 C NF NRA- NR-
5 C NF NRA- NR-


#Specification
Secret(sm, ma, [agg,ut])
Secret(sm, mb, [agg,ut])
```

```
Secret(sm, t, [ut])
Agreement(sm, agg, [ma, mb])
Agreement(sm, ut, [t])

--------------------------------

#Actual variables
SM1, SM2, I : Agent
UT : Utility
AGG : Aggregator
HFID1, HFID2 : HFID
LFID1, LFID2 : LFID
Ma1, Ma2 : HFDataA
Mb1, Mb2 : HFDataB
T1, T2 : LFData
K : Key
InverseKeys = (K,K)

#Functions

#System
SMARTMETER(SM1, AGG, UT, HFID1, LFID1, Ma1, Mb1, T1)
SMARTMETER(SM2, AGG, UT, HFID2, LFID2, Ma2, Mb2, T2)
AGGREGATOR(AGG, UT, HFID1)
AGGREGATOR(AGG, UT, HFID2)
UTILITY(UT, LFID1)
UTILITY(UT, LFID2)

#Intruder Information
Intruder = I
IntruderKnowledge = {I, SM1, SM2, AGG, UT}

#Privacy
Undetectable( AGG, {LFID1,T1} )
Unlinkable( AGG, {Ma1,Mb1}, {}, {} )
Unlinkable( AGG, {HFID1,LFID1}, {}, {(<Ma1,Mb1>,<T1>),(<Ma2,Mb2>,<
    T2>)} )
Unlinkable( UT, {HFID1,LFID1}, {}, {(<Ma1,Mb1>,<T1>),(<Ma2,Mb2>,<T2
    >)} )
```

Listing A.5: Casper-Privacy script for the smart meter communication protocol using pseudonyms [89]

A.6 Pseudonymous Smart Metering without a TTP

```
-- Advanced Metering Infrastructure in the Smart Grid based on the
   protocol in Finster2013
-- Consisting of two homes with smart meters and one energy utility
-- Assuming perfectly anonymous communication network


#Free variables
-- utility
ut : Utility
-- smart meter
sm : Agent

-- sequential measurements, A then B, from smart meter
-- (assumed to include timestamps)
ma : HFDataA
mb : HFDataB
-- total energy consumption from smart meter
t: LFData

-- grid operator keys
GEpublic   : GridEncryptionPublicKey
GEprivate  : GridEncryptionPrivateKey

-- pseudonymous keys for a smart meter
-- (Spublic assumed to be signed by utility using GSprivate)
Spublic  : SmartMeterPublicKey
Sprivate : SmartMeterPrivateKey

-- real keys for a smart meter
Rpublic  : RealPublicKey
Rprivate : RealPrivateKey

h : HashFunction
InverseKeys = (GEpublic,GEprivate), (Spublic,Sprivate), (Rpublic,
   Rprivate)

#Processes
SMARTMETER(sm, ut, GEpublic, Spublic, Sprivate, Rpublic, Rprivate,
   ma, mb, t)
UTILITY(ut, GEpublic, GEprivate, Spublic, Rpublic)

#Protocol description
1. sm -> ut : { Spublic, ut, ma, {h(ma)}{Sprivate} }{GEpublic}
2. sm -> ut : { Spublic, ut, mb, {h(mb)}{Sprivate} }{GEpublic}
3. sm -> ut : { Rpublic, ut, t,  {h(t)}{Rprivate} }{GEpublic}

#Specification
Secret(sm, ma, [ut])
--Secret(sm, mb, [ut])
--Secret(sm, t, [ut])
Agreement(sm, ut, [ma])
--Agreement(sm, ut, [t])
```

```
--------------------------------

#Actual variables
-- utility
UT : Utility
-- smart meter
SM1, I : Agent

-- sequentiual measurements, A then B, from smart meter
-- (assumed to include timestamps)
Ma1, Ma9 : HFDataA
Mb1, Mb9 : HFDataB
-- total energy consumption from smart meter
T1, T9  : LFData

-- grid operator keys
GEpublic1   : GridEncryptionPublicKey
GEprivate1  : GridEncryptionPrivateKey

-- pseudonymous keys for a smart meter
Spublic1  : SmartMeterPublicKey
Sprivate1 : SmartMeterPrivateKey

-- real keys for a smart meter
Rpublic1  : RealPublicKey
Rprivate1 : RealPrivateKey

InverseKeys = (GEpublic1,GEprivate1), (Spublic1,Sprivate1), (
   Rpublic1,Rprivate1)


#Functions

#System
SMARTMETER(SM1, UT, GEpublic1, Spublic1, Sprivate1, Rpublic1,
   Rprivate1, Ma1, Mb1, T1)
UTILITY(UT, GEpublic1, GEprivate1, Spublic1, Rpublic1)

#Intruder Information
Intruder = I
IntruderKnowledge = {I, Ma9, Mb9, T9}

#Privacy
Unlinkable( UT,  {Spublic1,Rpublic1}, {GEpublic1}, {(<Ma1,Mb1>,<T1
   >)} )
```

Listing A.6: Casper-Privacy script for the smart meter communication protocol using pseudonyms [101]

A.7 Smart Meter Anonymization through Group Identifiers

```
-- smart meter and utility
sm : Agent
ut : Utility
-- group identifier
idg : IdGroup
-- personal identifier
idc : IdPersonal
-- sequentiual measurements, A then B, from smart meter
ma : MeasurementA
mb : MeasurementB
-- total energy consumption from smart meter
t  : Total
pubUt  : PublicKey
privUt : PrivateKey
InverseKeys = (pubUt,privUt)

#Processes
SMARTMETER(sm, ut, idg, idc, ma, mb, t, pubUt)
UTILITY(ut, privUt, idg, idc)

#Protocol description
1.  sm -> ut : {idg, ma}{pubUt}
2.  sm -> ut : {idg, mb}{pubUt}
3.  sm -> ut : {idc,  t}{pubUt}

#Specification
Secret(sm, ma, [ut])
Secret(sm, mb, [ut])
Secret(sm, t,  [ut])
AnonymousAgreement(sm, ut, [ma, mb])
AnonymousAgreement(sm, ut, [t])

#Actual variables
SM1, SM2, I      : Agent
UT               : Utility
IdG              : IdGroup
IdC1, IdC2, IdC9 : IdPersonal
Ma1, Ma2, Ma9    : MeasurementA
Mb1, Mb2, Mb9    : MeasurementB
T1, T2, T9       : Total
PubUT            : PublicKey
PrivUT           : PrivateKey
InverseKeys = (PubUT,PrivUT)

#System
SMARTMETER(SM1, UT, IdG, IdC1, Ma1, Mb1, T1, PubUT)
SMARTMETER(SM2, UT, IdG, IdC2, Ma2, Mb2, T2, PubUT)
UTILITY(UT, PrivUT, IdG, IdC1)
UTILITY(UT, PrivUT, IdG, IdC2)

#Intruder Information
Intruder = I
```

```
IntruderKnowledge = {I, SM1, SM2, UT, Ma9, Mb9, T9, PubUT, IdC9,
   IdG}

#Privacy
Unlinkable( UT, {Ma1,IdC1}, {PubUT}, {} )
Unlinkable( UT, {Ma2,IdC2}, {PubUT}, {} )
Unlinkable( UT, {Mb1,IdC1}, {PubUT}, {} )
Unlinkable( UT, {Mb2,IdC2}, {PubUT}, {} )
```

Listing A.7: Casper-Privacy script for the smart meter communication protocol using group identifiers [43]

A.8 OpenADR Standard

```
-- Advanced Metering Infrastructure in the Smart Grid based on the
   OpenADR protocol
-- Consisting of two homes with smart meters, one energy utility
   and one demand response automation server (DRAS)
-- Assuming perfectly anonymous communication network

#Free variables
-- smart meter, utility and DRAS
sm : Agent
ut : Utility
dras : DRAutomationServer
-- identifier for a DR event
id : DREventID
-- amount bid by a specific smart meter
bid : BidAmount
k : Key
InverseKeys = (k,k)

#Processes
SMARTMETER(sm, dras, bid)
AUTOMATIONSERVER(dras, ut, sm)
UTILITY(ut, dras, id)

#Protocol description
1. ut -> dras : ut, dras, id
2. dras -> sm : dras, sm, id
3. sm -> dras : sm, dras, id, bid
4. dras -> ut : ut, sm, dras, id, bid

#Channels
-- Assuming confidentiallity for messages between the smart meter
   and the DRAS
-- based on the use of TLS connections and the public keys of SM
   and DRAS.
-- Assuming confidentially, integrity and authenticity for messages
    between the DRAS and UT
-- based on mutual TLS authentication.
1 C NF NRA- NR-
2 C NF NRA- NR-
3 C NF NRA- NR-
4 C NF NRA- NR-


#Specification
Secret(ut, id, [dras])
Secret(sm, bid, [dras])
Agreement(sm, dras, [id, bid])
Agreement(dras, ut, [id, bid])

------------------------------

#Actual variables
SM, I : Agent
```

```
UT : Utility
DRAS : DRAutomationServer
ID : DREventID
BID: BidAmount
K : Key
InverseKeys = (K,K)

#Functions

#System
SMARTMETER(SM, DRAS, BID)
AUTOMATIONSERVER(DRAS, UT, SM)
UTILITY(UT, DRAS, ID)

#Intruder Information
Intruder = I
IntruderKnowledge = {I, SM, DRAS, UT}

#Privacy
Unlinkable( DRAS, {SM,BID}, {}, {} )
Unlinkable( UT, {SM,BID}, {}, {} )
```

Listing A.8: Casper-Privacy script for the OpenADR smart meter communication protocol [213]

A.9 Privacy-Enhancing Network Monitoring Protocol

```
-- This protocol uses a Trustworthy Remote Entity (TRE) to
-- perform spatial aggregation of the frequent (15-30 minute)
-- consumption measurements from multiple consumers in
-- an area. The result is sent to the distribution network
-- operator (DNO) for network monitoring purposes.

-- Note: This script requires at least 6GB of memory.


#Free variables

-- consumers A and B, utility and TRE
ca, cb, dno, tre : Agent

-- sequential measurements from consumers at times 1 and 2
ma1, mb1 : Measurement1
ma2, mb2 : Measurement2

-- spatial aggregates of measurements from smart meters
-- A and B at times 1 and 2
agg1 : SpatialAggregate1
agg2 : SpatialAggregate2

-- required for the Casper analysis but not used
k : Key
InverseKeys = (k,k)

#Processes
CONSUMERA(ca, tre, ma1, ma2)
CONSUMERB(cb, tre, mb1, mb2)
REMOTEENTITY(tre, dno, ca, cb, agg1, agg2)
SERVICEPROVIDER(dno, tre, ca, cb)

#Protocol description
1.  ca -> tre : ca, ma1
1b. cb -> tre : cb, mb1
2.  tre -> dno : agg1
3.  ca -> tre : ca, ma2
3b. cb -> tre : cb, mb2
4.  tre -> dno : agg2

#Channels
-- Assuming mutually authenticated TLS connections
-- on all channels using separate key pairs.
1  C NF NRA- NR-
1b C NF NRA- NR-
2  C NF NRA- NR-
3  C NF NRA- NR-
3b C NF NRA- NR-
4  C NF NRA- NR-

#Specification
-- Pass
```

```
Secret(ca, ma1, [tre])
-- Pass
Secret(ca, ma2, [tre])
-- Pass
Secret(cb, mb1, [tre])
-- Pass
Secret(cb, mb2, [tre])
-- Pass
Agreement(ca, tre, [ma1, ma2])
-- Pass
Agreement(cb, tre, [mb1, mb2])
-- Pass
Agreement(tre, dno, [agg1, agg2])

#Actual variables
CA, CB, DNO, TRE, I : Agent
MA1, MB1, MI1 : Measurement1
MA2, MB2 : Measurement2
AggAB1, AggI1 : SpatialAggregate1
AggAB2, AggI2 : SpatialAggregate2
K : Key
InverseKeys = (K,K)

#System
CONSUMERA(CA, TRE, MA1, MA2)
CONSUMERB(CB, TRE, MB1, MB2)
REMOTEENTITY(TRE, DNO, CA, CB, AggAB1, AggAB2)
SERVICEPROVIDER(DNO, TRE, CA, CB)

#Intruder Information
Intruder = I
IntruderKnowledge = {I, CA, CB, TRE, DNO, MI1, AggI1, AggI2}
-- IntruderKnowledge = {I, SMA, SMB, TRE, DNO, MI1, MI2, AggI1,
   AggI2}


#Privacy
-- Pass
Unlinkable( DNO, {MA1,CA}, {}, {(<MA1,MB1>,<AggAB1>)} )
-- Pass
Unlinkable( DNO, {MB1,CB}, {}, {(<MA1,MB1>,<AggAB1>)} )
-- Pass
Unlinkable( DNO, {MA2,CA}, {}, {(<MA2,MB2>,<AggAB2>)} )
-- Pass
Unlinkable( DNO, {MB2,CB}, {}, {(<MA2,MB2>,<AggAB2>)} )
-- Pass
Unlinkable( DNO, {MA1,MA2}, {}, {(<MA1,MB1>,<AggAB1>), (<MA2,MB2>,<
   AggAB2>)} )
-- Pass
Unlinkable( DNO, {MB1,MB2}, {}, {(<MA1,MB1>,<AggAB1>), (<MA2,MB2>,<
   AggAB2>)} )
```

Listing A.9: Casper-Privacy script for the Network Monitoring Protocol

A.10 Privacy-Enhancing Billing Protocol

```
-- This protocol uses a Trustworthy Remote Entity (TRE) to
-- perform temporal aggregation of the frequent (15-30 minute)
-- consumption measurements from a single consumer
-- mutiplied by the price per unit sent by the supplier for
-- that time interval. The total is sent to the supplier
-- at the end of each period for billing purposes.


#Free variables

-- consumer A, energy supplier and TRE
ca, sup, tre : Agent

-- sequential measurements from consumer A at times 1 and 2
ma1 : Measurement1
ma2 : Measurement2

-- temporal aggregates of measurements at times 1 and 2
-- multiplied by prices at times 1 and 2
agga : TemporalAggregate

-- price values for time periods 1 and 2
price1, price2 : PriceValue

-- time stamps for time periods 1 and 2
t1, t2 : TimeValue

-- required for the Casper analysis but not used
k : Key
InverseKeys = (k,k)

#Processes
CONSUMERA(ca, tre, ma1, ma2, t1, t2)
REMOTEENTITY(tre, sup, ca, agga, t1, t2)
SERVICEPROVIDER(sup, tre, ca, t1, t2, price1, price2)

#Protocol description
1.  sup -> tre : t1, price1
2.  tre -> ca : t1, price1
3.  ca -> tre : ca, t1, ma1
4.  sup -> tre : t2, price2
5.  tre -> ca : t2, price2
6.  ca -> tre : ca, t2, ma2
7.  tre -> sup : ca, agga

#Channels
-- Assuming mutually authenticated TLS connections
-- on all channels using separate key pairs.
1  C NF NRA- NR-
2  C NF NRA- NR-
3  C NF NRA- NR-
4  C NF NRA- NR-
5  C NF NRA- NR-
```

```
6  C NF NRA- NR-
7  C NF NRA- NR-

#Specification
-- Pass
Secret(ca, ma1, [tre])
-- Pass
Secret(ca, ma2, [tre])
-- Pass
Agreement(ca, tre, [ma1, ma2])
-- Pass
Agreement(sup, tre, [price1, price2])
-- Pass
Agreement(ca, tre, [price1, price2])
-- Pass
Agreement(tre, sup, [agga])

#Actual variables
CA, SUP, TRE, I : Agent
MA1, MI1 : Measurement1
MA2, MI2 : Measurement2
AggA, AggI : TemporalAggregate
Price1, Price2 : PriceValue
T1, T2 : TimeValue
K : Key
InverseKeys = (K,K)

#System
CONSUMERA(CA, TRE, MA1, MA2, T1, T2)
REMOTEENTITY(TRE, SUP, CA, AggA, T1, T2)
SERVICEPROVIDER(SUP, TRE, CA, T1, T2, Price1, Price2)

#Intruder Information
Intruder = I
IntruderKnowledge = {I, CA, TRE, SUP, T1, T2, Price1, Price2, MI1,
   MI2, AggI}


#Privacy
-- Pass
Unlinkable( SUP, {MA1,CA}, {}, {(<MA1,Price1,MA2,Price2>,<AggA>)} )
-- Pass
Unlinkable( SUP, {MA2,CA}, {}, {(<MA1,Price1,MA2,Price2>,<AggA>)} )
-- Pass
Unlinkable( SUP, {MA1,MA2}, {}, {(<MA1,Price1,MA2,Price2>,<AggA>)}
   )
```

Listing A.10: Casper-Privacy script for the Billing Protocol

A.11 Privacy-Enhancing Demand Bidding Protocol

```
-- This protocol uses a Trustworthy Remote Entity (TRE) to
-- facilitate a demand bidding protocol between consumers
-- and the Demand Side Manager (DSM). Upon receiving a bid
-- from a legitimate consumer, the TRE creates a
-- pseudo-bid and sends it to the DSM. If the bid is
-- accepted, the DSM notifies the TRE which in turn
-- notifies the consumer.

-- Note: This script requires at least 6GB of memory


#Free variables

-- consumers A and B, Demand Side Manager (DSM) and TRE
ca, cb, dsm, tre : Agent

-- DR event from the DSM
-- dre1 : DREvent

-- bids from consumers A and B
bida1, bidb1 : Bid

-- pseudo-bids A and B generated by the TRE
pseudobida1, pseudobidb1 : PseudoBid

-- notifications for bids A and B from the DSM
nota1, notb1 : Notification

-- time stamps for time period 1
t1 : TimeValue

-- required for the Casper analysis but not used
k : Key
InverseKeys = (k,k)

#Processes
CONSUMERA(ca, tre, bida1, t1)
CONSUMERB(cb, tre, bidb1, t1)
REMOTEENTITY(tre, dsm, ca, cb, pseudobida1, pseudobidb1, t1)
SERVICEPROVIDER(dsm, tre, ca, cb, nota1, notb1, t1)

#Protocol description
1.  ca -> tre : ca, t1, bida1
1b. cb -> tre : cb, t1, bidb1
2.  tre -> dsm : t1, pseudobida1, pseudobidb1
3.  dsm -> tre : t1, nota1, notb1
4.  tre -> ca : t1, nota1
4b. tre -> cb : t1, notb1

#Channels
-- Assuming mutually authenticated TLS connections
-- on all channels using separate key pairs.
1  C NF NRA- NR-
```

```
1b C NF NRA- NR-
2  C NF NRA- NR-
3  C NF NRA- NR-
4  C NF NRA- NR-
4b C NF NRA- NR-

#Specification
-- Pass
Secret(ca, bida1, [tre])
-- Pass
Secret(cb, bidb1, [tre])
-- Pass
--Secret(ca, bida2, [tre])
-- Pass
--Secret(cb, bidb2, [tre])
-- Pass
Agreement(ca, tre, [bida1])
-- Pass
Agreement(tre, ca, [nota1])
-- Pass
Agreement(cb, tre, [bidb1])
-- Pass
Agreement(tre, cb, [notb1])
-- Pass
--Agreement(ca, tre, [bida2])
-- Pass
--Agreement(tre, ca, [nota2])
-- Pass
--Agreement(cb, tre, [bidb2])
-- Pass
--Agreement(tre, cb, [notb2])
-- Pass
--Agreement(tre, dsm, [pseudobida1, pseudobidb1, pseudobida2,
   pseudobidb2])
-- Pass
--Agreement(dsm, tre, [nota1, notb1, nota2, notb2])

#Actual variables
CA, CB, DSM, TRE, I : Agent
BidA1, BidB1, BidA2, BidB2, BidI : Bid
PseudoBidA1, PseudoBidB1, PseudoBidA2, PseudoBidB2, PseudoBidI :
   PseudoBid
NotA1, NotB1, NotA2, NotB2, NotI : Notification
T1, T2 : TimeValue
K : Key
InverseKeys = (K,K)

#System
CONSUMERA(CA, TRE, BidA1, T1);CONSUMERA(CA, TRE, BidA2, T2)
CONSUMERB(CB, TRE, BidB1, T1);CONSUMERB(CB, TRE, BidB2, T2)
REMOTEENTITY(TRE, DSM, CA, CB, PseudoBidA1, PseudoBidB1, T1);
   REMOTEENTITY(TRE, DSM, CA, CB, PseudoBidA2, PseudoBidB2, T2)
SERVICEPROVIDER(DSM, TRE, CA, CB, NotA1, NotB1, T1);SERVICEPROVIDER
   (DSM, TRE, CA, CB, NotA2, NotB2, T2)

#Intruder Information
```

```
Intruder = I
IntruderKnowledge = {I, CA, CB, TRE, DSM, T1, T2, BidI, PseudoBidI,
    NotI}

#Privacy
-- Pass
Undetectable( DSM, {BidA1} )
-- Pass
Undetectable( DSM, {BidB1} )
-- Pass
Undetectable( DSM, {BidA2} )
-- Pass
Undetectable( DSM, {BidB2} )
-- Pass
Unlinkable( DSM, {BidA1,CA}, {}, {(<NotA1>,<BidA1>)} )
-- Pass
Unlinkable( DSM, {BidB1,CB}, {}, {(<NotB1>,<BidB1>)} )
-- Pass
Unlinkable( DSM, {PseudoBidA1,CA}, {}, {(<NotA1>,<BidA1>)} )
-- Pass
Unlinkable( DSM, {PseudoBidB1,CB}, {}, {(<NotB1>,<BidB1>)} )
-- Pass
Unlinkable( DSM, {PseudoBidA1,PseudoBidA2}, {TRE}, {(<NotA1>,<BidA1
   >),(<NotA2>,<BidA2>)} )
-- Pass
Unlinkable( DSM, {PseudoBidB1,PseudoBidB2}, {TRE}, {(<NotB1>,<BidB1
   >),(<NotB2>,<BidB2>)} )
-- Pass
Unlinkable( DSM, {BidA1,BidA2}, {}, {(<NotA1>,<BidA1>),(<NotA2>,<
   BidA2>)} )
-- Pass
Unlinkable( DSM, {BidB1,BidB2}, {}, {(<NotB1>,<BidB1>),(<NotB2>,<
   BidB2>)} )
```

Listing A.11: Casper-Privacy script for the Demand Bidding Protocol

Appendix B

TCSP# System Models

This appendix contains the TCSP# models of the remote attestation protocols described in Chapter 7. Bai et al. [26, 25] have presented a full description of the TrustFound framework, on which these models are based, as well as an explanation of the TCSP# syntax. In these models, each of the following adversary capabilities can be individually enabled to construct the different types of attestation adversaries described in Chapter 7:

1. **Attack_net_intercept:** Read messages from the network
2. **Attack_net_modify:** Modify messages on the network
3. **Attack_net_interrupt:** Interrupt (block) messages on the network
4. **Attack_load_comp_sw:** Load compromised software on the prover's platform
5. **Attack_reset_platform:** Reset the prover's platform
6. **Attack_corrupt_bios:** Exploit a runtime vulnerability in the BIOS
7. **Attack_corrupt_sw:** Exploit a runtime vulnerability in the software
8. **Attack_reset_tpm:** Reset the prover's TPM without resetting the platform

B.1 Attestation Scenario

```
#import "PAT.Lib.EntryList";
#import "TPMLib2";

var<Cryptography> crypt;  var <Generator> TPMGenerator;  var <PCR> good_state_pcr_value;

channel a2n 0;  //alice->net
channel n2b 0;  //net->bob
channel b2n 0;  //bob->net
channel n2a 0;  //net->alice
channel platform_reset 0; //Bob's machine reset signal

Initialization()=
    generator{TPMGenerator=new Generator(new Sym(g_tpm), 1);}
    -> cryptinit {
        crypt=new Cryptography();
        key_alice_current=new AKey(key_alice,0);
        key_bob_current=new AKey(key_bob,0);
        key_eve_current=new AKey(key_eve,0);}
    -> tpminit {
        tpm=new SimpleTPM(new AKey(tpm_ek,0), new AKey(tpm_ek,1), new SKey(tpm_srk), TPMGenerator);
        tpm.installEKcert(new AKey(privacy_ca,1));
        tpm.reset();}
    -> pcrcompute {
        good_state_pcr_value=new PCR(new Value(0));
        good_state_pcr_value.extend(new PCR(new Prog(P_BIOS_GENUINE)));
        good_state_pcr_value.extend(new PCR(new Prog(P_SW_GENUINE)));}
    -> initknowledge {
        eve_knowledge.addKnowledge(new SKey(key_alice_eve));
        eve_knowledge.addKnowledge(new SKey(key_bob_eve));
        eve_knowledge.addKnowledge(new Prog(P_BIOS_GENUINE));
        eve_knowledge.addKnowledge(new Prog(P_SW_GENUINE));}
    -> ( alice_talks_to_bob{alice_correspondant=new Sym(bob);} -> Skip()
      [] alice_talks_to_eve{alice_correspondant=new Sym(eve);} -> Skip()    );
       ( bob_talks_to_alice{bob_correspondant=new Sym(alice);} -> InitTargetPCR()
      [] bob_talks_to_eve{bob_correspondant=new Sym(eve);} -> InitTargetPCR());
    InitOther();

// Alice
var <SKey> alice_session_key;  var key_alice_current;  var alice_correspondant;

// Eve
var <Knowledge> eve_knowledge;  var <SKey> eve_session_key;
var key_eve_current;  var reset_platform_window = false;  var corrupt_bios_window = false;  var
     corrupt_sw_window = false;  var reset_tpm_window = false;

Eve() = (
   [attack_reset_platform==true && max_platform_resets>0 && reset_platform_window==true]platform_reset!0{
     reset_platform_window=false;max_platform_resets--} -> Eve()
[] [attack_corrupt_bios==true && corrupt_bios_window==true]eve_corrupt_bios{bios_mem=new Prog(
     P_BIOS_COMPROMISED);corrupt_bios_window=false;} -> Eve()
[] [attack_corrupt_sw==true && corrupt_sw_window==true]eve_corrupt_sw{sw_mem=new Prog(P_SW_COMPROMISED);
     corrupt_sw_window=false;} -> Eve());

// Bob
var <Prog> bios_binary;  var <Prog> sw_binary;  var <Prog> bios_mem;  var <Prog> sw_mem;
var <SKey> bob_session_key;
var bob_correspondant;  var key_bob_current;  var bob_receives_secret = false;  var bob_leaked_keys = false;

BobTPM() = TPM();

BobFirmware() =
    start_tpm{reset_platform_window=false} -> (
          gen_bios_binary{bios_binary=new Prog(P_BIOS_GENUINE)}->Skip
       [] [attack_load_comp_sw==true]comp_bios_binary{bios_binary=new Prog(P_BIOS_COMPROMISED)}->Skip);
    TPM_Extend!(new Value(0)).(new PCR(new Prog(bios_binary))) -> TPM_Extend?0
    -> load_bios{bios_mem=new Prog(bios_binary);corrupt_bios_window=true}
    -> run_bios{corrupt_bios_window=false} -> (
          [bios_mem==new Prog(P_BIOS_GENUINE)]BobBIOSgood()
       [] [bios_mem==new Prog(P_BIOS_COMPROMISED)]BobBIOScompromised() );

BobBIOScompromised() = compromised_bios -> load_sw{sw_mem=new Prog(P_SW_COMPROMISED)} -> run_sw ->
     BobSWcompromised();

BobBIOSgood() =
    genuine_bios -> (
          gen_sw_binary{sw_binary=new Prog(P_SW_GENUINE)} -> Skip
       [] [attack_load_comp_sw==true]comp_sw_binary{sw_binary=new Prog(P_SW_COMPROMISED)} -> Skip);
    TPM_Extend!(new Value(0)).(new PCR(new Prog(sw_binary))) -> TPM_Extend?0
    -> load_sw{sw_mem=new Prog(sw_binary);corrupt_sw_window=true}
    -> run_sw{corrupt_sw_window=false} -> (
          [sw_mem==new Prog(P_SW_GENUINE)]BobSWgood()
       [] [sw_mem==new Prog(P_SW_COMPROMISED)]BobSWcompromised() );

BobSWcompromised() = compromised_sw{bob_correspondant=new Sym(alice);} -> BobSWcompromised1();

BobSWcompromised1() = BobSWcompromised2(eve_knowledge.getCount());

BobSWcompromised2(m) = [] x:{0..m-1} @ (
   [bob_leaked_keys==false]bob_leaks_keys{bob_leaked_keys=true;eve_knowledge.addKnowledge(new SKey(
     bob_session_key));eve_knowledge.addKnowledge(key_bob_current);eve_knowledge.addKnowledge(tpm.getEKcert());
     eve_knowledge.deduceKnowledge()} -> BobSWcompromised1()
[] bob_key_agreement_start -> bob_key_agreement_end -> BobSWcompromised1()
```

```
[] [max_bob_messages>0]b2n!(new CiphertextS(eve_knowledge.at(x),bob_session_key)){max_bob_messages--} ->
     BobSWcompromised1()
[] n2b?data{eve_knowledge.addKnowledge(crypt.decrypt(data,bob_session_key));eve_knowledge.deduceKnowledge()} ->
      BobSWcompromised1()
[] [attack_reset_tpm==true && max_tpm_resets>0]reset_tpm{tpm.reset();max_tpm_resets--} -> InitTargetPCR();
     BobSWcompromised1()
[] [target_pcr.count()>0 && new PCR(eve_knowledge.at(x))==target_pcr.head()]TPM_Extend!(new Value(0)).(new PCR(
     eve_knowledge.at(x))) -> TPM_Extend?0 -> BobSWcompromised1()
[] [target_pcr.count()==0 && eve_knowledge.at(x)==expected_quote_nonce]TPM_Quote!(new Value(0)).eve_knowledge.
     at(x) -> TPM_Quote?tpm_quote{eve_knowledge.addKnowledge(tpm_quote);eve_knowledge.deduceKnowledge()} ->
     BobSWcompromised1()
[] [target_pcr.count()==0 && new Nonce(eve_knowledge.at(x))==expected_quote_nonce]TPM_Quote!(new Value(0)).new
     Nonce(eve_knowledge.at(x)) -> TPM_Quote?tpm_quote{eve_knowledge.addKnowledge(tpm_quote);eve_knowledge.
     deduceKnowledge()} -> BobSWcompromised1() );

Bob() = ( BobFirmware() ||| BobTPM() ) interrupt platform_reset?0{tpm.reset()} -> BobReset(); Bob();

// Network
var <GeneratedMsg> gm;
N2B(n)=([] x:{0..n-1} @ (n2b!gm.at(x){gm.clear();}->Skip));
N2A(n)=([] x:{0..n-1} @ (n2a!gm.at(x){gm.clear();}->Skip));

Network() =
   a2n?data -> (
       [attack_net_intercept==true]intecept{eve_knowledge.addKnowledge(data);eve_knowledge.deduceKnowledge()
     ;}->n2b!data->Network()
    [] [attack_net_modify==true]modify{eve_knowledge.addKnowledge(data);eve_knowledge.deduceKnowledge();gm=
     eve_knowledge.generateMsg(data);}->N2B(gm.getCount());Network()
    [] [attack_net_interrupt==true]interrupt{eve_knowledge.addKnowledge(data);eve_knowledge.deduceKnowledge()
     ;}->Network()
    [] n2b!data->Network() )
[] b2n?data -> (
       [attack_net_intercept==true && sw_mem!=new Prog(P_SW_COMPROMISED)]intercept{eve_knowledge.addKnowledge(
     data);eve_knowledge.deduceKnowledge();}->n2a!data->Network()
    [] [attack_net_modify==true && sw_mem!=new Prog(P_SW_COMPROMISED)]modify{eve_knowledge.addKnowledge(data);
     eve_knowledge.deduceKnowledge();gm=eve_knowledge.generateMsg(data);}->N2A(gm.getCount());Network()
    [] [attack_net_interrupt==true && sw_mem!=new Prog(P_SW_COMPROMISED)]interrupt{eve_knowledge.addKnowledge(
     data);eve_knowledge.deduceKnowledge();}->Network()
    [] n2a!data->Network() );

KeyOracle() = (
   [alice_correspondant==new Sym(bob) && bob_correspondant==new Sym(alice) && key_alice_current==new AKey(
     key_alice,0) && key_bob_current==new AKey(key_bob,0)]alice_key_agreement_start -> bob_key_agreement_start
     -> alice_bob{alice_session_key=new SKey(key_alice_bob);bob_session_key=new SKey(key_alice_bob);} ->
     alice_key_agreement_end -> bob_key_agreement_end -> KeyOracle()
[] [alice_correspondant==new Sym(bob) && bob_correspondant==new Sym(alice) && key_alice_current==new AKey(
     key_alice,0) && key_bob_current==new AKey(key_bob2,0)]alice_key_agreement_start -> bob_key_agreement_start
      -> alice_bob{alice_session_key=new SKey(key_alice_bob2);bob_session_key=new SKey(key_alice_bob2);} ->
     alice_key_agreement_end -> bob_key_agreement_end -> KeyOracle()
[] [alice_correspondant==new Sym(eve)]alice_key_agreement_start -> alice_eve{alice_session_key=new SKey(
     key_alice_eve);} -> alice_key_agreement_end -> KeyOracle()
[] [bob_correspondant==new Sym(eve)]bob_key_agreement_start -> bob_eve{bob_session_key=new SKey(key_bob_eve);}
     -> bob_key_agreement_end -> KeyOracle() );

// Main system
Protocol() = Initialization(); ( ( Alice() || Bob()  || KeyOracle() ) ||| Eve() ||| Network() );
#define bobReceivesSecret (bob_receives_secret == true);
#define secretLeaks (eve_knowledge.knows(new Sym(alice_secret)));
#assert Protocol() reaches bobReceivesSecret;
#assert Protocol() reaches secretLeaks; //if true, secret has leaked

//TPM model
var <SimpleTPM> tpm; var <GDBase> sKeyBlob;  var <GDBase> tpmKeyHandle;  var <GDBase> tpmKeyCer;
var <GDBase> tpmPlain;  var <GDBase> tpmQuote;
channel TPM_Extend 0;  channel TPM_CreateWrapKey 0;  channel TPM_LoadKey 0;  channel TPM_CertifyKey 0;
channel TPM_EvictKey 0; channel TPM_LoadKey_fake 0;  channel TPM_UnBind 0;  channel TPM_Quote 0;

TPM() =
   [max_tpm_extends>0]TPM_Extend?index.data{tpm.TPM_PCRExtend(index, data);max_tpm_extends--;if(target_pcr.
     count()>0){target_pcr.remove_head()};} -> TPM_Extend!0 -> TPM()
[] TPM_CreateWrapKey?index.value{tpm.TPM_CreateWrapKeyV2(index,value);sKeyBlob=tpm.getRet();} ->
     TPM_CreateWrapKey!sKeyBlob -> TPM()
[] TPM_LoadKey?tpmKeyBlob{tpm.TPM_LoadKey2(tpmKeyBlob);tpmKeyHandle=tpm.getRet();} -> TPM_LoadKey!tpmKeyHandle
     -> TPM()
[] TPM_CertifyKey?tpmKeyHandle2.tpmCerNonce{tpm.TPM_CertifyKey(tpmKeyHandle2,tpmCerNonce);tpmKeyCer=tpm.getRet
     ();} -> TPM_CertifyKey!tpmKeyCer -> TPM()
[] TPM_EvictKey?tpmKeyHandle3{tpm.TPM_EvictKey(tpmKeyHandle3);} -> TPM_EvictKey!0 -> TPM()
[] TPM_UnBind?tpmEncData.tpmKeyHandle4{tpm.TPM_UnBind(tpmKeyHandle4,tpmEncData);tpmPlain=tpm.getRet();} ->
     TPM_UnBind!tpmPlain -> TPM()
[] [max_tpm_quotes>0]TPM_Quote?index.tpmQuoteNonce{tpm.TPM_Quote(index,tpmQuoteNonce);tpmQuote=tpm.getRet();
     max_tpm_quotes--} -> TPM_Quote!tpmQuote -> TPM();
```

Listing B.1: TCSP# model of an attestation scenario

B.2 Nonce-Challenge Attestation Protocol

```
#include "attestation.csp";

enum {nothing, g_tpm, alice, bob, eve, alice_nonce, alice_secret, tpm_ek, tpm_srk, privacy_ca, key_alice,
     key_bob, key_bob2, key_eve, key_alice_bob, key_alice_bob2, key_alice_eve, key_bob_eve, tls_session_key,
     P_BIOS_GENUINE, P_BIOS_COMPROMISED, P_SW_GENUINE, P_SW_COMPROMISED};

var <Seq> target_pcr = new Seq();

InitTargetPCR() =
    initialize_target_pcr {
        target_pcr.clear_all();
        target_pcr.add(new PCR(new Prog(P_BIOS_GENUINE)));
        target_pcr.add(new PCR(new Prog(P_SW_GENUINE)));
        if(mitigation_bind_quote_public_key==true){
            if(alice_correspondant==new Sym(bob)){key_alice_correspondant=key_bob_current};
            if(alice_correspondant==new Sym(eve)){key_alice_correspondant=key_eve_current};
            target_pcr.add(new PCR(key_alice_correspondant)); }
        if(mitigation_bind_quote_session_key==true){
            if(alice_correspondant==new Sym(bob)){target_pcr.add(new PCR(new SKey(key_alice_bob)))};
            if(alice_correspondant==new Sym(eve)){target_pcr.add(new PCR(new SKey(key_alice_eve)))};}} -> Skip;

InitOther() = Skip;

// Alice
var expected_quote_nonce = new Nonce(alice_nonce);  var alice_received_ek_cert;  var alice_received_quote;
var key_alice_correspondant;

Alice() =
    alice_key_agreement_start -> alice_key_agreement_end
    -> a2n!(new CiphertextS(new Nonce(alice_nonce), alice_session_key)){
        if(mitigation_bind_quote_public_key==true){
            if(alice_correspondant==new Sym(bob)){key_alice_correspondant=key_bob_current};
            if(alice_correspondant==new Sym(eve)){key_alice_correspondant=key_eve_current};
            good_state_pcr_value.extend(new PCR(key_alice_correspondant)) };
        if(mitigation_bind_quote_session_key==true){
            good_state_pcr_value.extend(new PCR(alice_session_key)) }; }
    -> n2a?enc_ek_cert{alice_received_ek_cert = crypt.decrypt(enc_ek_cert,alice_session_key);}
    -> n2a?enc_quote{alice_received_quote = crypt.decrypt(enc_quote,alice_session_key)} -> (
        ifa(crypt.verifyPCRCert(alice_received_quote, alice_received_ek_cert, new AKey(privacy_ca,0))) {
            accept_quote
            -> ifa(crypt.getQuotePCRIndex(alice_received_quote) == (new Value(0))
                && crypt.getQuotePCRValue(alice_received_quote) == good_state_pcr_value
                && crypt.getQuoteNonce(alice_received_quote) == expected_quote_nonce)   {
                bob_in_good_state
                -> a2n!(new CiphertextS(new Sym(alice_secret), alice_session_key)) -> Skip }
            else { bob_not_in_good_state -> Skip } }
        else { reject_quote -> Skip } );

// Bob
var <Nonce> bob_received_nonce;

BobSWgood() =
    genuine_sw
    -> bob_key_agreement_start -> bob_key_agreement_end
    -> n2b?enc_nonce{bob_received_nonce=crypt.decrypt(enc_nonce, bob_session_key)}
    -> b2n!(new CiphertextS(tpm.getEKcert(),bob_session_key)) -> Skip;
    if(mitigation_bind_quote_public_key==true){TPM_Extend!(new Value(0)).(new PCR(key_bob_current))
        -> TPM_Extend?0 -> Skip};
    if(mitigation_bind_quote_session_key==true){TPM_Extend!(new Value(0)).(new PCR(bob_session_key))
        -> TPM_Extend?0 -> Skip};
    TPM_Quote!(new Value(0)).(new Nonce(bob_received_nonce)) -> TPM_Quote?bob_tpm_quote
    -> b2n!(new CiphertextS(bob_tpm_quote,bob_session_key)){reset_platform_window=true}
    -> n2b?data{if(crypt.decrypt(data,bob_session_key)==new Sym(alice_secret)){bob_receives_secret=true};
     reset_platform_window=false} -> Skip;

BobReset() = bob_reset{ if(mitigation_use_pfs==true){bob_session_key=new SKey(nothing)}; } -> Skip;

// Verification bounds
var max_tpm_extends = 6;
var max_tpm_quotes = 1;
var max_bob_messages = 2;
var max_platform_resets = 1;
var max_tpm_resets = 1;

// Permitted attacks
var attack_net_intercept = false;
var attack_net_modify = false; //true allows masquerading attack
var attack_net_interrupt = false;
var attack_load_comp_sw = false; //true allows loading compromised BIOS or software
var attack_reset_platform = false; //true allows reboot attack (requires attack_load_compromised_sw)
var attack_corrupt_bios = false; //true allows runtime exploit of the BIOS
var attack_corrupt_sw = false; //true allows runtime exploit of the software
var attack_reset_tpm = false; //true allows TPM reset attack

// Attack mitigations
var mitigation_bind_quote_public_key = false; //true prevents masquerading attack
var mitigation_bind_quote_session_key = false; //true prevents masquerading attack
var mitigation_use_pfs = false; //true prevents reboot attack (requires mitigation_bind_quote_session_key)
```

Listing B.2: TCSP# model of a nonce-challenge attestation protocol

B.3 Global Timestamp Attestation Protocol

```
#include "attestation.csp";

enum {nothing, g_tpm, g_timestamp, alice, bob, eve, alice_secret, tpm_ek, tpm_srk, privacy_ca, key_alice,
     key_bob, key_bob2, key_eve, key_alice_bob, key_alice_bob2, key_alice_eve, key_bob_eve, tls_session_key,
     P_BIOS_GENUINE, P_BIOS_COMPROMISED, P_SW_GENUINE, P_SW_COMPROMISED};

var <Seq> target_pcr = new Seq();

InitTargetPCR() =
    initialize_target_pcr{
        target_pcr.clear_all();
        target_pcr.add(new PCR(new Prog(P_BIOS_GENUINE)));
        target_pcr.add(new PCR(new Prog(P_SW_GENUINE)));
        if(mitigation_bind_quote_public_key==true){
            if(alice_correspondant==new Sym(bob)){key_alice_correspondant=key_bob_current};
            if(alice_correspondant==new Sym(eve)){key_alice_correspondant=key_eve_current};
            target_pcr.add(new PCR(key_alice_correspondant)); } } -> Skip;

InitOther() = add_eve_knowledge{eve_knowledge.addKnowledge(new Nonce(g_timestamp))} -> Skip;


// Alice
var expected_quote_nonce = new Nonce(g_timestamp);
var alice_received_ek_cert;
var alice_received_quote;
var key_alice_correspondant;

Alice() =
    alice_key_agreement_start -> alice_key_agreement_end
    -> n2a?enc_ek_cert{alice_received_ek_cert = crypt.decrypt(enc_ek_cert,alice_session_key);}
    -> n2a?enc_quote{alice_received_quote = crypt.decrypt(enc_quote,alice_session_key);} -> (
        ifa(crypt.verifyPCRCert(alice_received_quote, alice_received_ek_cert, new AKey(privacy_ca,0))) {
            accept_quote{
                if(mitigation_bind_quote_public_key==true){
                    if(alice_correspondant==new Sym(bob)){key_alice_correspondant=key_bob_current};
                    if(alice_correspondant==new Sym(eve)){key_alice_correspondant=key_eve_current};
                    good_state_pcr_value.extend(new PCR(key_alice_correspondant))}; } ->
            ifa(crypt.getQuotePCRIndex(alice_received_quote) == (new Value(0))
                && crypt.getQuotePCRValue(alice_received_quote) == good_state_pcr_value
                && crypt.getQuoteNonce(alice_received_quote) == expected_quote_nonce)   {
                bob_in_good_state
                -> a2n!(new CiphertextS(new Sym(alice_secret), alice_session_key)) -> Skip }
            else {bob_not_in_good_state -> Skip} }
        else {reject_quote -> Skip} );


// Bob
BobSWgood() =
    genuine_sw ->
    if(mitigation_bind_quote_public_key==true){TPM_Extend!(new Value(0)).(new PCR(key_bob_current))
        -> TPM_Extend?0 -> Skip };
    TPM_Quote!(new Value(0)).(new Nonce(g_timestamp)) -> TPM_Quote?bob_tpm_quote{reset_platform_window=true}
    -> bob_key_agreement_start -> bob_key_agreement_end
    -> b2n!(new CiphertextS(tpm.getEKcert(),bob_session_key))
    -> b2n!(new CiphertextS(bob_tpm_quote,bob_session_key))
    -> n2b?data{if(crypt.decrypt(data,bob_session_key)==new Sym(alice_secret)){bob_receives_secret=true};
     reset_platform_window=false} -> Skip;

BobReset() = bob_reset{ if(mitigation_use_pfs==true){bob_session_key=new SKey(nothing)}; } -> Skip;

// Verification bounds
var max_tpm_extends = 6;
var max_tpm_quotes = 1;
var max_bob_messages = 2;
var max_platform_resets = 1;
var max_tpm_resets = 1;

// Permitted attacks
var attack_net_intercept = false;
var attack_net_modify = false; //true allows masquerading attack
var attack_net_interrupt = false;
var attack_load_comp_sw = false; //true allows loading compromised BIOS or software
var attack_reset_platform = false; //true allows reboot attack (requires attack_load_compromised_sw)
var attack_corrupt_bios = false; //true allows runtime exploit of the BIOS
var attack_corrupt_sw = false; //true allows runtime exploit of the software
var attack_reset_tpm = false; //true allows TPM reset attack

// Attack mitigations
var mitigation_bind_quote_public_key = false; //true prevents masquerading attack
// Cannot bind quote to session key in one-to-many protocol
var mitigation_use_pfs = false;
```

Listing B.3: TCSP# model of an attestation protocol using global timestamps

B.4 Final State Attestation Protocol

```
#include "attestation.csp";

enum {nothing, g_tpm, alice, bob, eve, alice_secret, tpm_ek, tpm_srk, privacy_ca, key_alice, key_bob, key_bob2,
      key_eve, key_alice_bob, key_alice_bob2, key_alice_eve, key_bob_eve, tls_session_key, P_BIOS_GENUINE,
     P_BIOS_COMPROMISED, P_SW_GENUINE, P_SW_COMPROMISED};

var <Seq> target_pcr = new Seq();

InitTargetPCR() =
    initialize_target_pcr{
        target_pcr.clear_all();
        target_pcr.add(new PCR(new Prog(P_BIOS_GENUINE)));
        target_pcr.add(new PCR(new Prog(P_SW_GENUINE))); } -> Skip;

InitOther() = Skip;


// Alice
var expected_quote_nonce = new Nonce(new AKey(key_bob,0));
var alice_received_ek_cert;
var alice_received_quote;

Alice() =
    alice_key_agreement_start -> alice_key_agreement_end
    -> n2a?enc_ek_cert{alice_received_ek_cert = crypt.decrypt(enc_ek_cert,alice_session_key);
        if(alice_correspondant==new Sym(bob)){expected_quote_nonce=new Nonce(key_bob_current)};
        if(alice_correspondant==new Sym(eve)){expected_quote_nonce=new Nonce(key_eve_current)}; }
    -> n2a?enc_quote{alice_received_quote = crypt.decrypt(enc_quote,alice_session_key);} -> (
        ifa(crypt.verifyPCRCert(alice_received_quote, alice_received_ek_cert, new AKey(privacy_ca,0))) {
            accept_quote ->
            ifa(crypt.getQuotePCRIndex(alice_received_quote) == (new Value(0))
                && crypt.getQuotePCRValue(alice_received_quote) == good_state_pcr_value
                && crypt.getQuoteNonce(alice_received_quote) == expected_quote_nonce)   {
                bob_in_good_state
                -> a2n!(new CiphertextS(new Sym(alice_secret), alice_session_key)) -> Skip }
            else {bob_not_in_good_state -> Skip} }
        else {reject_quote -> Skip} );


// Bob
BobSWgood() =
    bob_key_agreement_start -> bob_key_agreement_end
    -> TPM_Quote!(new Value(0)).(new Nonce(key_bob_current)) -> TPM_Quote?bob_tpm_quote
    -> b2n!(new CiphertextS(tpm.getEKcert(),bob_session_key))
    -> b2n!(new CiphertextS(bob_tpm_quote,bob_session_key)){reset_platform_window=true}
    -> n2b?data{if(crypt.decrypt(data,bob_session_key)==new Sym(alice_secret)){bob_receives_secret=true};
     reset_platform_window=false} -> Skip;

 BobReset() = bob_reset{
    if(mitigation_use_pfs==true){bob_session_key=new SKey(nothing)};
    if(mitigation_ephemeral_public_key==true){key_bob_current=new AKey(key_bob2,0)}; } -> Skip;


// Verification bounds
var max_tpm_extends = 6;
var max_tpm_quotes = 1;
var max_bob_messages = 2;
var max_platform_resets = 1;
var max_tpm_resets = 1;

// Permitted attacks
var attack_network_interception = false;
var attack_network_modification = false; //true allows masquerading attack
var attack_network_interruption = false;
var attack_load_compromised_sw = false; //true allows loading compromised BIOS or software
var attack_reset_platform = false; //true allows reboot attack (requires attack_load_compromised_sw)
var attack_corrupt_bios = false; //true allows runtime exploit of the BIOS
var attack_corrupt_sw = false; //true allows runtime exploit of the software
var attack_reset_tpm = false; //true allows TPM reset attack

// Attack mitigations
// Protocol automatically binds quote to Bob's public key
// Cannot bind quote to session key in one-to-many protocol
var mitigation_use_pfs = false; //does not have an effect
var mitigation_ephemeral_public_key = false; //true prevents reboot attack
```

Listing B.4: TCSP# model of the Final State Attestation (FSA) protocol