Human Centric Security and Privacy for the IoT using Formal Techniques

Florian Kammüller
Middlesex University London, Department of Computer Science, UK
f.kammueller@mdx.ac.uk

Abstract. In this paper, we summarize a new approach to make security and privacy issues in the Internet of Things (IoT) more transparent for vulnerable users. As a pilot project, we investigate monitoring of Alzheimer's patients for a low-cost early warning system based on bio-markers supported with smart technologies. To provide trustworthy and secure IoT infrastructures, we employ formal methods and techniques that allow specification of IoT scenarios with human actors, refinement and analysis of attacks and generation of certified code for IoT component architectures.

Keywords: Human Factors · Security and Privacy in the IoT · Formal Methods

1 Introduction

The Internet of Things (IoT) denotes the combination of physical objects with their virtual representation in the Internet. It consists not only of human participants but "Things" as well. The IoT has a great potential to provide novel services to humans in all parts of our society. Amongst the biggest problems for this technology to catch on in critical applications are security flaws, due to technical restrictions, immaturity of software applications, and mainly a lack of transparency. The main trigger for security problems is human behaviour, either unintentional or malicious.

In this paper, we give an overview of how we apply formal techniques to enhance security and privacy of human centric IoT systems. We focus on healthcare aiming to support low-cost Alzheimer's diagnosis. We outline the process we use in the CHIST-ERA project SUCCESS. In detail, we report on using interactive theorem proving with Isabelle. We use this proof assistant for the modeling and attack analysis of infrastructures with humans and for the formal definition cryptographic. We apply the Isabelle Insider framework for human centric infrastructure analysis and the inductive approach for security protocol verification to support the secure IoT system development in the early security requirement phase as well as the technical network security level.

2 Background

This section provides a short summary of the techniques used in the process of formal development that we use in SUCCESS before highlighting the contributions of the current paper.

2.1 Overview of SUCCESS Project

The core idea of our approach is to use formal methods and verification tools to provide more transparency of security risks for people in given IoT scenarios.
SUCCESS will validate the scientific and technological innovation through pilots, one of which will be in collaboration with a hospital and will allow all stakeholders (e.g. physicians, hospital technicians, patients and relatives) to enjoy a safer system capable to appropriately handle highly sensitive information on vulnerable people while making security and privacy risks understandable and secure solutions accessible.
This international collaboration is funded by the European programme CHIST-ERA [1]. We apply techniques from hardware and software, user behaviour and human-computer interaction to a research pilot from the healthcare sector on supporting IoT monitoring techniques that are human understandable and can be certified by automated techniques.
• specification and verification techniques for secure IoT components and their composition [2],
• verification methods and risk assessment techniques [3] for IoT scenarios with models of human behavior [4], social interactions and human-system interactions,
• implementation and modeling languages with algorithms for the certification of safety, availability, secrecy, and trustworthiness across from the model to the platform [5].

2.2 Contribution of this Paper and Overview

This paper summarizes how the requirements for the IoT healthcare system lead to a high level formal specification in the Isabelle insider framework [6] which also allows attack tree analysis [4]. This first phase of the SUCCESS approach is summarized in Section 3 illustrated on a simplified architecture and use cases for the pilot case study. As a result of this first phase, the input to the BIP-based analysis and component architecture design and certified code generation is provided. We omit details on this phase since it is not within the scope of the paper. The output of this process however is a Java Script smartphone app capable of synchronous communication within the phone and via Bluetooth with sensors in the environment of the phone in the patient's home. The communication of the smartphone app with data servers in hospitals and other institutions (like research centers) is asynchronous and channeled via the Internet. It cannot be part of the certified code generation in BIP (which is restricted to synchronous communication). Therefore, we show up in Section 4 what is the state of the art of technically realizing secure communication for privacy sensitive data using web services and data interchange formats. In Section 5, we illustrate how to formally verify this communication using the inductive approach of security protocol verification in Isabelle (which is compatible with the Isabelle Insider framework as has recently been shown [7]).

3 Healthcare Case Study in Isabelle Insider Framework

The case study we use as a running example in this paper is a simplified scenario from the context of the SUCCESS project for Security and Privacy of the IoT [1]. A central topic of this project for the pilot case study is to support security and privacy when using cost effective methods based on the IoT for monitoring patients for the diagnosis of Alzheimer's disease. As a starting point for the design, analysis, and construction, we currently develop a case study of a small device for the analysis of blood samples that can be directly connected to a mobile phone. The analysis of this device can then be communicated by a dedicated app on the smart phone that sends the data to a server in the hospital.

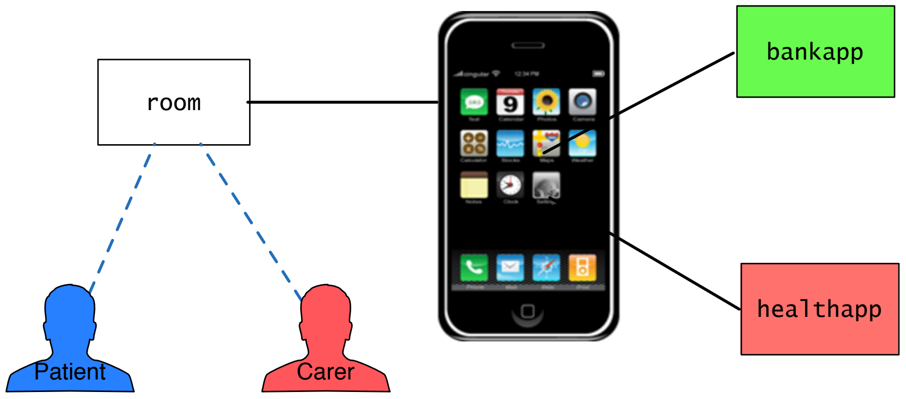

Fig. 1. Health care scenario: carer and patient in the room may use smartphone apps.

3.1 Healthcare Scenario

In this simplified scenario, there are the patient and the carer within a room together with the smart phone (see Figure 1).
The carer has access to the phone to support the patient in handling the special diagnosis device, the smart phone, and the app. The insider threat scenario has a second banking app on the smart phone that needs the additional authentication of a “secret key”: a small electronic device providing authentication codes for one time use common for private online banking.
Assuming that the carer finds this device in the room of the patient, he can steal this necessary credential and use it to get onto the banking app. Thereby he can get money from the patient’s account without consent.

3.2 Isabelle Insider Framework Analysis

The Isabelle Insider framework enables formalization of the infrastructure as a graph of locations, like room or smartphone, in which human actors reside in locations and local policies are attached to them as well. The details of this modeling and analysis of the case study is given in [4]. As a brief illustration we give some excerpts here. The local policies are given by the following Isabelle definition and explained below.

```
local_policies G ≡
(λ x. if x = room then {(λ y. True,{get, put, move}) }
      else (if x = sphone then{((λ y. has (y, ''PIN'')),
                   {put,get,eval,move}),(λy. True, {})}
            else (if x = healthapp then
                         {((λ y. (∃ n. (n @G sphone)∧
                           Actor n = y)),
                           {get,put,eval,move})}
                  else (if x = bankapp then
                          {((λy.(∃n.(n@G sphone)∧
                           Actor n=y ∧ has(y,''skey''))),
                           {get,put,eval,move})}
                        else {}))))
```

In this policy, any actor can move to the room and when in possession of the PIN can move onto the sphone and do all actions there. The following restrictions are placed on the two other locations.
• healthapp: to move onto the healthapp and perform any action at this location, an actor must be at the position sphone already;
• bankapp: to move onto the bankapp and perform any action at this location, an actor must be at the position sphone already and in possession of the skey.

3.3 Attack Tree Analysis

Attack Trees [8] are a graphical tree-based design language for the stepwise investigation and quantification of attacks. They have been integrated as an extension to the Isabelle Insider framework [6]. This integration extends the Insider model described in the previous section with a proof calculus and modelchecking semantics for attack trees. The extension allows stepwise refinement of attacks exhibiting possible attack paths. The refinement of attack trees is illustrated in Figure 2 with the refined attack path highlighted.

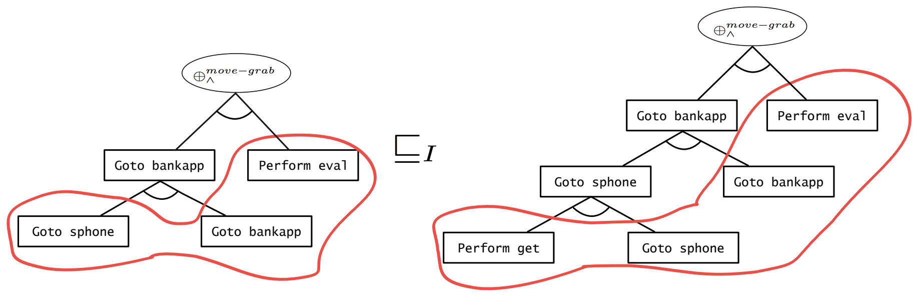

Fig 2. Attack tree refinement enables stepwise attack path discovery.

The following refinement shows the logical expression of this attack refinement. It expresses that the carer can evaluate the money transfer on the bankapp by first stealing the skey, getting on the phone, on the bankapp and then evaluating.

$$\texttt{[Goto bankapp, Perform eval]} \oplus_{\wedge}^{\texttt{move-grab}}$$
$$\sqsubseteq_{\texttt{hc\_scenario}}$$
$$\texttt{[Perform get, Goto sphone, Goto bankapp, Perform eval]} \oplus_{\wedge}^{\texttt{move-grab}}$$

The proof calculus uses the refinement to prove that the sequence of actions `[Perform get, Goto sphone, Goto bankapp, Perform eval]` represents an attack in the given infrastructure. The underlying semantics providing the notion of validity of an attack is based on the state transition relation defined in the modelchecking foundation (Kripke-structure over infrastructure states) we constructed in the Isabelle Insider framework.

The attack tree analysis enables formalizing the requirements and high level architecture of the pilot case study. The found attacks can be used to improve the security policies on the model to provide a security enhanced formal specification for the next phase of applying the BIP methodology to develop a component architecture for the target IoT infrastructure in which the security properties of the initial model are preserved and certified code for the components (sensors and smart phone) can be generated. We omit any details of this phase since they will be reported elsewhere. In addition, the attack trees and paths are naturally suited to visualize the security risks to users showing up potential attacks.

4 Security of Web Services for Mobile Devices

We now move to the level of the overall system architecture of SUCCESS in order to show up security and privacy risks of IoT devices connected to data servers via Internet and smart phone technology. In order to be compatible with existing standard technologies, the target code for the smartphone healthapp will be implemented in Java Script. This app represents the client side interface to the database servers in hospitals and other institutions, like research centers. Fortunately, the BIP methodology [2] is flexible enough to produce a Java Script app as certified target code for this component. However, BIP is designed for the formal development of synchronous systems. For the local scenario of sensors connected to a central hub like the smartphone either by physical link – like a blood sample sensor that can be connected via the micro usb or lightning port of the smartphone – or through close range networking protocols – like motion sensors communicating with the phone via Bluetooth [9], this is sufficient. Bluetooth is a packet-based protocol with a master-slave structure where all slaves share the master's clock, i.e., it is synchronous and thus amenable to the BIP code generation and certification process.

But the main data upload of the diagnosis data is to databases on external servers connected via Internet. This is asynchronous communication using web-services. The overall architecture is shown in Figure 3 showing yet another Insider attack by the carer (discussed further below).

Current standards of best practice for web services for mobile applications have settled on two combinations of technology (1) Java Script Object Notation (JSON) [10] over RESTful web services using http(s) or (2) eXtensible Markup Language XML over SOAP using Web Service Security (WSS) [11]. Solution (1) is more lightweight since the JSON data transfer standard is much less complex than XML. REST prescribes a standard format for web services that is also less complex than SOAP.

So from that perspective, it is a clear choice that in the context of mobile application the former is preferable to guarantee less resource consumption caused by an overhead of the SOAP/XML solution. The critical point is the consideration of security. While the combination of JSON over an https based RESTful web service is slick and appears sufficient it relies on the "s" in https, i.e. Transport Layer Security (TLS) (or Secure Socket Layer (SSL) how it was originally called and is still more widely known as).

TLS is a good standard solution providing point-to-point security between the http port or http proxy of the smart phone and its counterpart on the database servers. However, it does not provide end-to-end security. The difference is that in an end-to-end security connection the security protection would be between the healthapp and the database application on the server instead of in between the http socket of the smartphone usually on port 80 and the connected socket on the same port on the server as it is provided by a TLS connection. Do we need end-to-end security for SUCCESS?

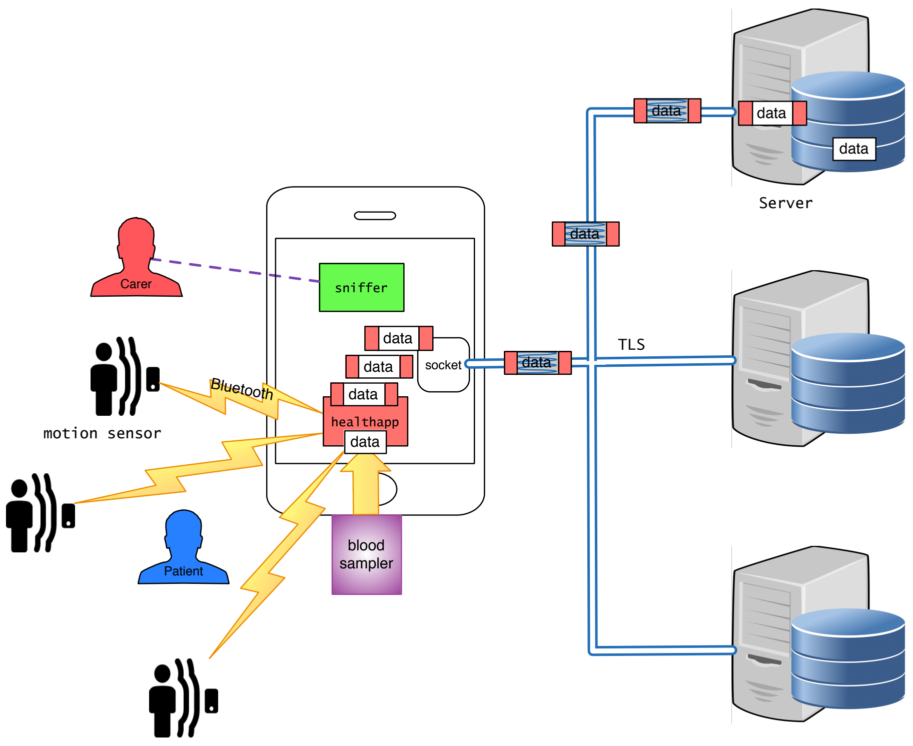

Fig 3. Carer puts sniffer on smart phone eavesdropping on cleartext TCP packets.

Consider again Figure 3: since the carer needs to have access to the smart phone to support the patient, he can still endanger privacy by the following attack. Suppose, we only use point-to-point security as given by TLS available on smart phones and servers by default. The carer can use his access to the smartphone to download a sniffer app from the app store, like Wireshark and thereby he can trace and intercept all message communication on the smartphone. This is again an insider attack since again the carer is the attacker. The CMU Insider Threat Guide provides the Insider Attack pattern of ambitious leader: if the carer would collaborate with an ambitious leader outside the home, he could install a specialized app on the phone that would forward intercepted packages from the healthapp to the server of the ambitious leader who could sell the data to interested parties or use it directly for blackmail.
Using the Isabelle Insider framework with its extension to attack trees [6,4] this attack can be discovered and proved in the attack tree calculus.
`[Goto sphone, Perform put, Goto sniffer, Perform eval]` $\oplus_{\wedge}^{\texttt{put-sniffer}}$

It exposes an interesting challenge for the Isabelle Insider framework since an actor extends the infrastructure (and thus implicitly the local policies) by adding the new location sniffer.

5 Suggested Solution for Security of Web-Services for Mobile Devices

Practically, to introduce end-to-end security we could use JSON as data-interchange format and a RESTful web service with http and TLS. The standard use of http does not foresee the use of an individual protocol for the authentication and key establishment between the healthapp and the database servers, we can design a protocol on top of the transport layer security that enables our application to establish end-to-end rather than point-to-point security.
Designing your own protocol bears risks since security flaws can be introduced. However, in SUCCESS, we use formal methods and in particular Isabelle offers the inductive approach to security protocol verification, e.g. [12], which we use to verify our protocol.

5.1 Isabelle Inductive Approach

A TLS formalization exists already in Isabelle [12]. It uses the inductive approach to security protocol verification that is compatible with the Isabelle Insider framework as we have found in recent applications to an auction protocol [7]. So, for our purposes we can simply assume TLS to be available and base the formal modeling and verification of the current end-to-end extension for SUCCESS on it. Such formal representation of security protocols in the inductive approach and other formal approaches, provide abstract descriptions of protocols. These abstract descriptions usually express what is formalized in standards like in the case of TLS. They serve to prove security properties with mathematical rigour and machine support making the assumption that keys are not lost, and cryptographic algorithms are not broken. Concerning communication channels they use the common strong Dolev-Yao attacker model: messages can be eavesdropped, intercepted and faked.
The Isabelle formalization uses inductive definitions. These definitions are contained in Isabelle theory files adding some modularity to the inductive approach. That is, for our application, we can use aspects of the TLS protocol and mainly its underlying theory of cryptographic keys and messages. To ensure end-to-end security, however, we define our own lightweight security protocol that runs within the TLS. The full TLS specification and proved properties can be found elsewhere [12]. Details on formalizing a protocol in the inductive approach are explained below when presenting the protocol.

5.2 End-to-End Security for Smartphone Apps over JSON and REST

We can now express a simple protocol that supports mutual authentication between the healthapp represented as P (for Patient) and a database server Server. In the process of this communication a shared key is negotiated that can then be used for future encrypted data upload to the server. We focus on two security goals: (A) authentication of the client to the server and of the server to the client (B) symmetric key exchange for future confidential communication.

The protocol we propose assumes public keys to be in place and trusted. This is a realistic simplifying assumption since the communication is usually to fixed institutions and additional public keys of new servers can be added on the smart phone app. An additional public key certification authority protocol in the style of the DNSsec protocol could be used to set this up [13].
 Nonces (Numbers only used once) are used for freshness in the key establishment and authentication phase. The goal of the protocol is the establishment of a shared key KP,Server for a future secure communication of private data from the app to the application on the Server. The healthapp is here referred to as P (for Patient) and the application on the server as Server. The protocol is specified as a set of event lists in the following inductive definition; the rules are explained below.

```
inductive_set sucsec :: event list set
where
Nil:  [] ∈ sucsec|
Fake: [| evsf ∈ sucsec; X ∈ synth (analz (spies evsf)) |]
         ⇒ Says Spy A X # evsf ∈ sucsec |
suc1: [| evs1 ∈ sucsec; ~(Nonce N_P ∈ used evs1);
        ~(Key K_P,Server ∈ used evs1) |]
      ⇒ Says P Server (Crypt(pubK Server)
            {Key K_P,Server, Nonce N_P}) # evs1 ∈ sucsec |
suc2: [| evs2 ∈ sucsec; ~(Nonce N_Server ∈ used evs2);
        Says P Server (Crypt(pubK Server)
        {Key K_P,Server, Nonce N_P}) ∈ set evs2 |]
      ⇒ Says Server P (Crypt(pubK P)
            {Nonce N_P, Nonce N_Server, Server})
        # evs2 ∈ sucsec |
suc3: [| evs3 ∈ sucsec; Says Server P (Crypt(pubK P)
        {Nonce N_P, Nonce N_Server, Server}) ∈ set evs3 |]
      ⇒ Says P Server
            (Crypt(pubK Server){Nonce N_Server})
        # evs2 ∈ sucsec
```

This protocol is inspired by the improved version of the Needham-Schroeder public key protocol adding the symmetric session key $K_{P,Server}$ created by the healthapp using for example AES 256. The rule Nil initiates the set with the empty trace representing the point before any protocol session starts. The rule fake is the rule that introduces events created by the agent Spy who can synthesize and play into any event trace evsf messages based on what he analyses from all eavesdropped traffic: synth(analz(spies evsf)).
Note, that he "says" this message to an unspecified agent A which could be Server or P. The rule suc1 requires a fresh Nonce and a fresh symmetric key both created by the healthapp. Freshness of a Nonce or a key is expressed for example as ~(Nonce NP ∈ used evs1) meaning that this Nonce has not (~) been used in the trace evs1 before.

Agent P then sends these items encrypted with the public key of the server process Server. Consequently, those items can only be seen by Server. According to rule suc2, the server process responds with a message in which it packs its fresh Nonce and the unpacked Nonce of P submitted in the previous message thereby proving that it is in the possession of the private key priK Server. This corresponds to server authentication. Finally, rule suc3 is the client authentication in which the healthapp P proves that it is in possession of priK P by unpacking and repacking Nonce N_{Server} from the previous message.

Despite these arguments being seemingly obvious deductions from the protocol steps, they need to be verified to guarantee security. The inductive approach in Isabelle allows formal verification of these and other security properties of the protocol.

The provided abstract specification of the protocol can be implemented as initially mentioned using JSON or XML to encode the transmitted data (messages including keys) over https (thus automatically creating the TLS tunnel between the smart phone and the webserver of the hospital or research institution). The asymmetric cryptography for public and private key pairs can be implemented for example using RSA and the symmetric keys could be AES 256. For both JavaScript libraries exist.

The authentication protocol can be used for different servers. The data that is then sent by the healthapp (either in JSON of XML) can be preprocessed by sanitization of the data (e.g., delete names and address for scientific purposes when connecting to the database of the research center). Following instead the SOAP standard would require the use of XML and this is not suited to our mobile app. The practical ad hoc standard of using the lighter JSON data interchange format and combining it with a RESTful web service is practically sufficient and compatible with JavaScript as target language for the certified code generation of the healthapp as output of the BIP process.

6 Discussion and Conclusions

In this paper, we have given an overview of applying a range of formal techniques to the security and privacy sensitive scenario of healthcare focused on mobile Alzheimer's diagnosis. We only sketched the overall process as we envisage to use it in the CHIST-ERA project SUCCESS but detailed on the use of interactive theorem proving in Isabelle in two stages: (1) for a formal machine-supported analysis of attacks at early development stages and (2) for the formal definition of a dedicated end-to-end cryptographic protocol between a smart phone app and server database applications. Both stages are supported by Isabelle frameworks: (a) the Isabelle Insider framework for human centric infrastructure analysis and (b) the inductive approach for security protocol verification. The combination of both within the Isabelle framework is straightforward. A closer integration to formalize and prove deeper security properties involving both levels has been explored in a different context of auction protocols [7] and has procured interesting insights into collusion attacks and new notions of rational agents.

It seems promising and a future challenge for SUCCESS to explore this integration on privacy of IoT solutions for vulnerable agents. Initial challenges like dynamic extension of the infrastructure graph and local policies (example of the sniffer app download) have already been identified in this paper.

The suggested use of the Bluetooth protocol [9] for the short distance communication in the patients home offers an additional security vulnerability due to symmetric key agreement protocols. However, there is a stronger implementation that uses asymmetric key establishment and that is feasible for certain devices including smart phones [14]. Starting from Bluetooth version 2.1 it is required to use Secure Simple Pairing (SSP) for pairing which is the public key based pairing method. If the attack analysis will show that a Bluetooth based attack is a risk SUCCESS needs to address, then we have to verify whether this asymmetric solution is feasible between the motion sensors and smartphone. Otherwise, we need to integrate weaker mitigation stategies, e.g. enable Bluetooth only when required, in the patient diagnosis policy. This part is addressed in the central part of the formal development of a component based architecture using the BIP methodology and is not covered in this paper.

Acknowledgments. Part of the research leading to these results has received funding from the European Union (CHIST-ERA 2015) under grant agreement no. 102112 (SUCCESS). This publication reflects only the authors' views and the Union is not liable for any use that may be made of the information contained herein.

References

1. SUCCESS: SecUre aCCESSibility for the internet of things. http://www.chistera.eu/projects/success. CHIST-ERA 2016.
2. A. Basu, S. Bensalem, M. Bozga, J. Combaz, M. Jaber, T.-H. Nguyen, and J. Sifakis. Rigorous Component-Based System Design Using the BIP Framework. IEEE Software, volume 28, No. 3, 2011.
3. F. Arnold, H. Hermanns, R. Pulungan, M.I.A. Stoelinga: Time-Dependent Analysis of Attacks. Principles of Security and Trust (POST'14), LNCS, pages 285–305, 2014.
4. F. Kammüller. Formal Modeling and Analysis with Humans in Infrastructures for IoT Healthcare Systems. 5th International Conference on Human Aspects of Information Security, Privacy, and Trust, HAS'17, co-located with HCII 2017. Springer LNAI, 2017.
5. N. Ben Said, T. Abdellatif, S. Bensalem, M. Bozga. Model-driven Information Flow Security for Component-Based Systems. ETAPS Workshop `From Programs to Systems', FPS@ETAPS 2014: 1-20, 2014.
6. F. Kammüller and C.W. Probst. Modeling and Verification of Insider Threats Using Logical Analysis. IEEE Systems Journal, 2016.
7. F. Kammüller, M. Kerber, C. W. Probst. Insider Threats for Auctions: Formal Modeling, Proof, and Certified Code. Special Issue of Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA), 8(1), 2017.
8. B. Schneier. Secrets and Lies: Digital Security in a Networked World. John Wiley & Sons, 2004.
9. Wikipedia. Bluetooth. https://en.wikipedia.org/wiki/Bluetooth. Accessed 4.3.2017.
10. JSON. ECMA-404 The JSON Data Interchange Standard. http://www.json.org, 2017.
11. OASIS. Web Services Security: SOAP Message Security. Working Draft 13, Document identifier: WSS: SOAP Message Security -13. Location: http://www.oasis-open.org/committees/documents.php. OASIS Open 2002.
12. L. C. Paulson. Inductive analysis of the Internet protocol TLS. ACM Transactions on Information and System Security. 2(3): 332–351, 1999.

13. F. Kammüller. Verification of DNSsec Delegation Signatures. 21st International Conference on Telecommunication, IEEE 2014.
14. F.-L. Wong, F. Stajano, and J. Clulow. Repairing the Bluetooth pairing protocol. Security Protocols '05. LNCS 4631: 31-45, Springer, 2007.