Explanation by Automated Reasoning Using the Isabelle Infrastructure Framework

Florian Kammüller

Middlesex University London and
Technische Universität Berlin
f.kammueller@mdx.ac.uk

Abstract. In this paper, we propose the use of interactive theorem proving for explainable machine learning. After informally motivating our proposition, we illustrate it on the dedicated application of explaining security attacks using the Isabelle Infrastructure framework and its process of dependability engineering. This formal framework and process provides the logics for specification and modeling. Attacks on security of the system are explained by specification and proofs in the Isabelle Infrastructure framework. Existing case studies of dependability engineering in Isabelle are used as feasibility studies to illustrate how different aspects of explanations are covered by the Isabelle Infrastructure framework. Finally, we propose a research agenda on how first-class explanation integrated with automated reasoning will solve the problem.

Keywords: Explainable AI, Automated Reasoning, Dependability Engineering, First Class representation of Attack Trees, Isabelle Infrastrucure framework

1 Proposing Interactive Theorem Proving for Explainable Machine Learning

Machine Learning (ML) is everywhere in Computer Science now. One may almost say that all of Computer Science has now become a part of ML and is viewed as a technique within the greater realm of Data Science or Data Engineering. But while this major trend like many other trends prevails, we should not forget that Artificial Intelligence (AI) is the original goal of what was the starting point of machine learning and that Automated Reasoning has been created as a means to provide for artificial intelligent systems a mechanical way of imitating human reasoning by implementing logics and automatizing proof.

When we think of how to explain why a specific solution for a problem is a solution, the purest way to do so is to explain it by way of mathematically precise arguments – which is equivalent to providing a logically sound proof in a mathematical model of the solution domain or context. An ML algorithm would do the same, for example, by providing a decision tree to explain a solution, but usually the ML explanations which are generated by the ML model itself are

very close to the ML implementation. So, they often fail to give a satisfactory, i.e. human understandable explanation.

This paper shows our point of view on a tangible way forward to combining interactive theorem proving with machine learning (ML). Different from the main stream of using ML to improve automated verification, we propose an integration at a higher level, using logical modeling and automated reasoning for explainability of machine learning solutions. The main idea of our proposal is based on one major fact about logic and proof:

> *Reasoning is not only a very natural way of explanation but it is also the most complete possible one since it provides a mathematical proof on a formal model.*

In the spirit of this thought, we provide a proof of concept on a framework that has been established for security and privacy analysis, the Isabelle Infrastructure framework. In this paper, we thus first introduce this framework by summarizing its basic conepts and various applications (Section2). After contrasting to some other conceptual approaches to ML and theorem proving including explanation (Section 3), we highlight the aspects that the Isabelle Infrastructure framework already provides (Section 4), before we finally sketch our conceptual proposal for using first-class representations of explanations in the logic to enable automated reasoning (Section 5).

2 Isabelle Infrastructure Framework

The Isabelle Infrastructure framework is implemented as an instance of Higher Order Logic in the interactive generic theorem prover Isabelle/HOL [24]. The framework enables formalizing and proving of systems with physical and logical components, actors and policies. It has been designed for the analysis of insider threats. However, the implemented theory of temporal logic combined with Kripke structures and its generic notion of state transitions are a perfect match to be combined with attack trees into a process for formal security engineering [3] including an accompanying framework [11].

Kripke structures, CTL and Attack Trees A number of case studies have contributed to shape the Isabelle framework into a general framework for the state-based security analysis of infrastructures with policies and actors. Temporal logic and Kripke structures are deeply embedded into Isabelle's Higher Order logic thereby enabling meta-theoretical proofs about the foundations: for example, equivalence between attack trees and CTL statements have been established [8] providing sound foundations for applications. This foundation provides a generic notion of state transition on which attack trees and temporal logic can be used to express properties for applications. The logical concepts and related notions thus provided for sound application modeling are:

– Kripke structures and state transitions:
  A generic state transition relation is $\rightarrow_i$; Kripke structures over a set of states `t` reachable by $\rightarrow_i$ from an initial state set `I` can be constructed by the `Kripke` constructor as

  $$\texttt{Kripke}\ \{\texttt{t}.\ \exists\ \texttt{i} \in \texttt{I}.\ \texttt{i} \rightarrow_i^*\ \texttt{t}\}\ \texttt{I}$$

– CTL statements:
  We can use the Computation Tree Logic (CTL) to specify dependability properties as

  $$\texttt{K} \vdash\ \mathsf{EF}\ \texttt{s}$$

  This formula states that in Kripke structure `K` there is a path (E) on which the property `s` (given as the set of states in which the property is true) will eventually (F) hold.
– Attack trees:
  attack trees are defined as a recursive datatype in Isabelle having three constructors: $\oplus_\vee$ creates or-trees and $\oplus_\wedge$ creates and-trees. And-attack trees $l{\oplus_\wedge^s}$ and or-attack trees $l{\oplus_\vee^s}$ consist of a list of sub-attacks which are themselves recursively given as attack trees. The third constructor takes as input a pair of state sets constructing a base attack step between two state sets. For example, for the sets `I` and `s` this is written as $\mathcal{N}_{(\texttt{I},\texttt{s})}$. As a further example, a two step and-attack leading from state set `I` via `si` to `s` is expressed as

  $$\vdash\ [\mathcal{N}_{(\texttt{I},\texttt{si})},\mathcal{N}_{(\texttt{si},\texttt{s})}]{\oplus_\wedge^{(\texttt{I},\texttt{s})}}$$

– Attack tree refinement, validity and adequacy:
  Attack trees can be constructed also by a refinement process but this differs from the system refinement presented in the paper [13]. An abstract attack tree may be refined by spelling out the attack steps until a valid attack is reached:
  $\vdash$`A :: (`σ`:: state) attree)`.
  The validity is defined constructively so that code can be generated from it. Adequacy with respect to a formal semantics in CTL is proved and can be used to facilitate actual application verification. This is used for the stepwise system refinements central to the methodology called Refinement-Risk cycle developed for the Isabelle Infrastructure framework [13].

A whole range of publications have documented the development of the Isabelle Insider framework. The publications [20–22] first define the fundamental notions of insiderness, policies, and behaviour showing how these concepts are able to express the classical insider threat patterns identified in the seminal CERT guide on insider threats [2]. This Isabelle Insider framework has been applied to auction protocols [17, 18] illustrating that the Insider framework can embed the inductive approach to protocol verification [25]. An Airplane case study [15, 16] revealed the need for dynamic state verification leading to the

extension of adding a mutable state. Meanwhile, the embedding of Kripke structures and CTL into Isabelle have enabled the emulation of Modelchecking and to provide a semantics for attack trees [5–8, 11]. Attack trees have provided the leverage to integrate Isabelle formal reasoning for IoT systems as has been illustrated in the CHIST-ERA project SUCCESS [3] where attack trees have been used in combination with the Behaviour Interaction Priority (BIP) component architecture model to develop security and privacy enhanced IoT solutions. This development has emphasized the technical rather than the psychological side of the framework development and thus branched off the development of the Isabelle *Insider* framework into the Isabelle *Infrastructure* framework. Since the strong expressiveness of Isabelle allows to formalize the IoT scenarios as well as actors and policies, the latter framework can also be applied to evaluate IoT scenarios with respect to policies like the European data privacy regulation GDPR [9]. Application to security protocols first pioneered in the auction protocol application [17,18] has further motivated the analysis of Quantum Cryptography which in turn necessitated the extension by probabilities [4, 10, 12].

Requirements raised by these various security and privacy case studies have shown the need for a cyclic engineering process for developing specifications and refining them towards implementations. A first case study takes the IoT healthcare application and exemplifies a step-by-step refinement interspersed with attack analysis using attack trees to increase privacy by ultimately introducing a blockchain for access control [11]. First ideas to support a dedicated security refinement process are available in a preliminary arxive paper [23] but only the follow-up publication [14] provides the first full formalization of the RR-cycle and illustrates its application completely on the Corona-virus Warn App (CWA). The earlier workshop publication [19] provided the formalization of the CWA illustrating the first two steps but it did not introduce the fully formalised RR-cycle nor did it apply it to arrive at a solution satisfying the global privacy policy [13].

3 Machine Learning, Explanation and Theorem Proving

If theorem proving could automatically be solved by machine learning, we would solve the P=NP problem [28]. Nevertheless, ML has been successfully employed within theorem provers to enhance the decision processes. Also in Isabelle, the sledgehammer tool uses ML mainly to select lemmas.

A very relevant work by Vigano and Magazzeni [27] focuses the idea of explainability on security, coining the notion of *XSec* or *Explainable Security*. The authors propose a full new research programme for the notion of explainability in security in which they identify the "Six Ws" of XSec: Who? What? Where? When? Why? And hoW? They position their paper clearly into the context of some earlier works along the same lines, e.g. [1, 26], but go beyond the earlier works by extending the scope and presenting a very concise yet complete description of the challenges. As opposed to XAI in general, the paper shows how already in understanding explanations only for the focus area of security (as op-

posed to all application domains of IT) is quite a task. Also they point out that XAI is merely concerned with explaining the technical solution provided by ML, whereas XSec looks at various other levels most prominently, the human user, by addressing domains like *usable security* and *security awareness*, and *security economics* [27][p. 294].

Our point of view is quite similar to Vigano's and Magazzeni's but we emphasize the technical side of explanation using interactive theorem proving and the Isabelle Infrastructure framework, while they focus on differentiating the notion of explanation from different aspects, for example, stake holders, system view, and abstraction levels. However, the notion of refinement defined for the process of dependability engineering for the Isabelle Infrastructure framework [13] allows addressing most of the Six Ws, because our model includes actors and policies and allows differentiation between insider and outsider attacks, expression of awareness [14]. Thus, we could strictly follow the Ws when explaining our proposition but we believe it is better to contemplate the Ws simply in the context of classical Software Engineering that has similar Ws. Moreover, the Refinement-Risk cycle of dependability engineering can be seen as specification refinement framework that employs the classical AI technique of automated reasoning. Surely, the human aspect versus the system aspect on the Six Ws of XSec brings in various different view points but these are inherent in if the contexts, that are needed for the interpretation are present in the model. Otherwise, they simply have to be added to it, for example, by using refinement to integrate these aspects of reality into the model. Then the Isabelle Infrastructure framework allows explanation for various purposes, audiences, technical levels (HW/SW). policies, localities and other physical aspects. Thus, we can answer all Six Ws and argue that is what human centric software, security, and dependability engineering are all about.

Moreover, despite contrasting from the approach by Vigano and Megazzini, we follow the classical engineering approach of Fault-tree analysis, more concretely using Attack Trees, and propose a dual process of attack versus security protection goal analysis which in itself offers a direct input to ML, for example to produce features that could be used for Decision trees as well as metrics that could provide feedback for optimization techniques as used in reinforcement learning.

4 Explaining (not only) Security by the Isabelle Infrastructure framework

This section describes the core ideas of explanation provided by applying the Isabelle Infrastructure framework.

4.1 State transition systems and attack trees as a dual way of explanation

One important aspect of explanation that is not restricted to security at all is to provide a step by step trace of state transitions to explain how a specific state

may appear. This can explain where a problem lies, for example, to explain how an ML algorithm arrived at a decision for a medical diagnosis by lining up a number of steps that lead to it.

In the Isabelle Infrastructure framework the notion of state transition systems is provided as a generic theory based on Kripke structures to represent state graphs over arbitrary types of states and using the branching time logic CTL to express temporal logical formulas over them. The correspondence between the CTL formulas of reachability and attack trees and the proof of adequacy are suitable to allow for a dual step by step analysis of a system dove-tailing the fault analysis with a specification refinement. This dove-tailing process leads to an elaborate process not only of explaining faults of system designs and how they can be reached practically by a series of actions but also an explanation of additional features of a system that are motivated by the detected fault. For example, when it comes to human awareness and usable security an explanation of a necessary security measure that is imposed on a user can be readily illustrated by an attack graph or its equivalent attack path that can be readily produced by the adequacy theory.

4.2 Human and Locality Aspects

The Isabelle Infrastructure framework has initially been designed to be merely focused on modeling and analyzing Insider threats before it became extended into what is now known as the Isabelle Infrastructure framework. Due to this initial motivation the framework explicitly supports the notion of human actors within networks of physical and virtual locations. These aspects are important to model various different stake holders to enable explanations to different audiences having different view points and needing different levels of detail and complexity in their explanations. For example, the explanation of a security threat will have a substantially different form if produced for a security analyst of to a system end user. Due to the explicit representation of human actors as well as their locations and other variable features, the Isabelle Infrastructure framework supports a fine grained control over the definition of applications thus enabling very flexible support of explanation about human aspects and suited to human understanding.

Also the human aspect necessitates consideration of the human condition, in particular psychological characterizations. The Isabelle Infrastructure framework, by augmenting the Isabelle Insider framework, provides for such characterization. For example, when considering insiderness, the state of the insider is characterized by a predicate that allows to use this state within a logical analysis of security and privacy threats to a system. Although these characterizations are axiomatic in the sense that the definition of the insider predicate is based on empirical results that have been externally input into the specification, it is in principle feasible to enrich the cognitive model of the human in the Isabelle Insider framework. A first step towards that has been done by experimenting with an extension of a notion of human awareness to support additionally anal-

ysis of unintentional insiders for human unawareness of privacy risks in social media [14].

4.3 Dependability Engineering: Specifying Protection Goals and Quantifying Attackers

The process of Dependability Engineering – the Refinement-Risk (RR) cycle – conceived for the Isabelle Infrastructure framework [23] allows a human centric system specification to be refined step-by-step following an iteration of finding faults within a system specification and refining this specification by more sophisticated data types or additional rules or changes to the semantics of system functions. The data type refinement allows integrating for example, more restrictive measures to control data, for example, using blockchains to enhance data consistency, or data labeling for access control. This refinement is triggered by previously found flaws in the system and thus provides concrete motivations for such design decisions leading to constructive explanations. Similarly, additional constraints on rules that are introduced in a refinement step of the RR-cycle are motivated by previously found attacks, for example, the necessity to change the ephemeral id of every user when they move to a new location instantaneously at moving time for the Corona-Warn-App is motivated by an identification attack [13, 19].

Since the RR-cycle is based on the idea of refinement, another requirement for a flexible explanation comes in for free: if we want to explain to different audiences or at different technical levels, we equally need to refine (or abstract) definitions of data-types, rules for policies, or descriptions of algorithms. The Isabelle Infrastructure framework directly supports these expressions at different abstraction levels and from different view points.

4.4 Quantification

An important aspect is quantification for explanation. Very often an explanation will not be possible in a possibilistic way. A quantification could be given by adding probabilities as well as other quantitative data, like costs, to explanations. For example, for a security attack the cost that an attacker is estimated to invest maximally on a specific attack step is an inevitable ingredient for a realistic attacker model. Similarly, the likelihood of a successful attack of a certain attack step could be needed for an analysis. Attack trees support these types of quantification. Naturally, the Isabelle Infrastructure framework also supports them. The application to the security analysis of Quantum Cryptography, i.e., the modeling and analysis of the Quantum Key Distribution protocol (QKD) lead to the extension for probabilistic state transition systems [4, 10, 12].

Quantification can also be a useful explanation for the process of learning for example by quantifying a distance to an attack goal. In that sense, quantified explanation can be a useful feedback for machine learning itself.

4.5 Explanation trees, attack trees and first-class representation

Pieters uses explanation trees to visualise the relation between explanation goals and subgoals. An explanation tree according to Pieters is "a tree in which the goals and subgoals of an explanation are ordered systematically" [26]. Explanation trees resemble very much attack trees, as already has been observed by Pieters. An attack tree explains an attack by a process that can be characterized as "attack tree refinement" in the Isabelle Insider framework [7, 8, 10]: a subtree "explains" the more refined steps that lead to the parent attack. Ultimately, the attack tree refinement leads to a valid explanation. Since attack trees are fully embedded as "first-class citizens" into the logic in the Isabelle Insider framework, it is not only possible to provide a formal semantics for such valid attacks based on Kripke structures and the temporal logic CTL but also to derive an efficient decision procedure (this means that code is generated in programming languages like Scala for deciding the validity of attack trees).

Similarly, first class explanations of explanation trees are well suited to provide semantically sound explanations. Since explanation trees are similar to attack trees a slight adaptation of their existing first-class representation suffices. Due to the first-class representation, sound justifications can be provided by proof. Also transparency of explanations can be achieved because the concepts of the Isabelle Infrastructure framework allow consistent translation of these first-class explanation trees at different levels of refinement. The conceptual inclusion of the human actor in the Isabelle Infrastructure framework additionally ensures that mere technical explanations can be made transparent for human centric contexts.

5 A Proposal: First-Class Explanation by Automated Reasoning

Based on the stock-taking in the previous subsection, we propose to use expressive formal logical models to provide explanations at all levels for different purposes and to different users. Explainability is a hot topic of Artificial Intelligence (AI). There is even a dedicated US research agenda called XAI (for eXplainable AI) by DARPA. The focus there is on providing a technical justification by explaining how a black box learning algorithm arrives at a decision. However, explanations are equally needed for other purposes, for example, to explain to a surgeon why the expert system suggests he should remove suspect tissue during an operation, but also in security, for example, to raise awareness for users of social networks about their privacy risks, as well as security experts, of what is going on in a network under attack. Generally, explanations may be used to (a) justify legal or more generally ethical decisions and (b) to describe something in detail to explain to humans how and why a decision is correct [26]. Purpose (b) is very important to create trust by enabling transparency. Explanations in the wider sense may be organized as explanation trees containing explanation goals as root nodes and subgoals as subtrees. Such trees can be related to a

verification task equivalent to breaking the overall goal (the root goal) into its subgoals. Explanation trees resemble attack trees as used in security analysis. Such trees can be supported by automated reasoning by representing them explicitly as first-class citizens of the logic. Thereby the goal/subgoal-creation as well as their disjunctive or conjunctive composition can be assigned to a formal semantics and adequacy can be proved by automated reasoning. The expressiveness of some logics allows providing such a first class representation of (attack or explanation) trees in such a generic (polymorphic) way that the tree as well as its semantics can be instantiated to different scenarios. First class representation allows thus meta-logical reasoning while also using the representation to verify applications. For example, we could use explanation trees to represent decision trees - a common machine learning model suitable for technical explanations.

In XAI advances are being made on verification of non-symbolic AI approaches, such as Feed-Forward and Convolutional Neural Networks. Probabilistic Model Checking and Abstract Interpretation techniques promise to guarantee robustness, that is, explanations of which inputs are mapped to outputs which allow some reliable predictions by modelling closely the machine learning algorithms. The level of explanation that can be reached by such verification techniques lacks expression of relevant higher-level concepts present in the application which are necessary for justification in non-technical contexts, like laws, and detailed descriptions for humans. For example, the success of automated language translation tools like, the encoder-decoder pairs of networks used in Google Translate, are grounded on exploiting large data sets of government documents from bilingual countries computing large tables of probabilities between phrases of the different languages rather than using syntax and grammar rules as symbolic AI did. Explaining why these translations are good matches necessitates representing contextual information of the matched examples as concepts in the logical language, that is, make them first class citizens.

The potential reward is transparency and justifiability of automated decision systems that employ non-symbolic approaches, ranging from explanations in safety critical areas (why did the airplane crash? – was it a fault or an attack?) to security and privacy (who can see your private data on Facebook and how and why does it change if you change your settings?). Abstraction permits explanation that is consistent with a logic, for example temporal logic, to ascertain verifiability and consistency of the model. The explanation can be done consistently in a rich model where important concepts of the application context are explicit part of the formal model underlying the explanation tree thus guaranteeing soundness. Such a consistent and sound explanation can be used as a technical explanation for non-symbolic AI, for example as a decision tree, but it can also be used to provide an explanation for transparency to humans. Detailed descriptions on how a decision was arrived at can be constructed from the rich model of the application. For justifications, the semantic embedding for the first-class representation plays a key role as it permits to transfer the justification goal of the explanation via the underlying semantics of the tree. Thus, the justification can be formally proved in the logic again with respect to the

rich expressive model and relevant domain specific rules from the application. Additionally, justifications are guaranteed to be verifiably sound and consistent. In essence, chaining a symbolic approach based on first-class explanations to non-symbolic approaches will provide a higher level of abstraction that is closer to human understanding increasing awareness and trust.

6 Conclusions

In this paper we have proposed the use of Automated Reasoning in the particular instance of the Isabelle Infrastructure framework for Explanation. We summarized the work that lead to the creation of the Isabelle Infrastructure framework highlighting the existing applications and extensions. After studying some related work on explanation, we provided a range of conceptual points that argued why and how the Isabelle Infrastructure framework already supports explanation and can be used as a basis for a dedicated explanation framework. Finally, we propose a new research agenda that outlines how explanation can be achieved using first-class representations for explanations in automated reasoning systems extending existing concepts of the Isabelle Infrastructure framework.

References

1. G. Bender, L. Kot, and J. Gehrke. Explainable security for relational databases. In C. E. Dyreson, F. Li, and M. T. Özsu, editors, *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*, pages 1411–1422. ACM, 2014.
2. D. M. Cappelli, A. P. Moore, and R. F. Trzeciak. *The CERT Guide to Insider Threats: How to Prevent, Detect, and Respond to Information Technology Crimes (Theft, Sabotage, Fraud)*. SEI Series in Software Engineering. Addison-Wesley Professional, 1 edition, Feb. 2012.
3. CHIST-ERA. Success: Secure accessibility for the internet of things, 2016. http://www.chistera.eu/projects/success.
4. F. Kammüller. Formalizing probabilistic quantum security protocols in the isabelle infrastructure framework. Informal Presentation at Computability in Europe, CiE 2019.
5. F. Kammüller. Formal models of human factors for security and privacy. In *5th International Conference on Human Aspects of Security, Privacy and Trust, HCII-HAS 2017*, volume 10292 of *LNCS*, pages 339–352. Springer, 2017. Affiliated with HCII 2017.
6. F. Kammüller. Human centric security and privacy for the iot using formal techniques. In *3d International Conference on Human Factors in Cybersecurity*, volume 593 of *Advances in Intelligent Systems and Computing*, pages 106–116. Springer, 2017. Affiliated with AHFE'2017.
7. F. Kammüller. A proof calculus for attack trees. In *Data Privacy Management, DPM'17, 12th Int. Workshop*, volume 10436 of *LNCS*. Springer, 2017. Co-located with ESORICS'17.
8. F. Kammüller. Attack trees in isabelle. In *20th International Conference on Information and Communications Security, ICICS2018*, volume 11149 of *LNCS*. Springer, 2018.

9. F. Kammüller. Formal modeling and analysis of data protection for gdpr compliance of iot healthcare systems. In *IEEE Systems, Man and Cybernetics, SMC2018*. IEEE, 2018.
10. F. Kammüller. Attack trees in isabelle extended with probabilities for quantum cryptography. *Computer & Security*, 87, 2019.
11. F. Kammüller. Combining secure system design with risk assessment for iot healthcare systems. In *Workshop on Security, Privacy, and Trust in the IoT, SPTIoT'19, colocated with IEEE PerCom*. IEEE, 2019.
12. F. Kammüller. Qkd in isabelle – bayesian calculation. *arXiv*, cs.CR, 2019.
13. F. Kammüller. Dependability engineering in isabelle, 2021. arxiv preprint, `http://arxiv.org/abs/2112.04374`.
14. F. Kammüller and C. M. Alvarado. Exploring rationality of self awareness in social networking for logical modeling of unintentional insiders, 2021. arxiv preprint, `http://arxiv.org/abs/2111.15425`.
15. F. Kammüller and M. Kerber. Investigating airplane safety and security against insider threats using logical modeling. In *IEEE Security and Privacy Workshops, Workshop on Research in Insider Threats, WRIT'16*. IEEE, 2016.
16. F. Kammüller and M. Kerber. Applying the isabelle insider framework to airplane security. *Science of Computer Programming*, 206, 2021.
17. F. Kammüller, M. Kerber, and C. Probst. Towards formal analysis of insider threats for auctions. In *8th ACM CCS International Workshop on Managing Insider Security Threats, MIST'16*. ACM, 2016.
18. F. Kammüller, M. Kerber, and C. Probst. Insider threats for auctions: Formal modeling, proof, and certified code. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, 8(1), 2017.
19. F. Kammüller and B. Lutz. Modeling and analyzing the corona-virus warning app with the isabelle infrastructure framework. In *20th International Workshop of Data Privacy Management, DPM'20*, volume 12484 of *LNCS*. Springer, 2020. Co-located with ESORICS'20.
20. F. Kammüller and C. W. Probst. Invalidating policies using structural information. In *IEEE Security and Privacy Workshops, Workshop on Research in Insider Threats, WRIT'13*, 2013.
21. F. Kammüller and C. W. Probst. Combining generated data models with formal invalidation for insider threat analysis. In *IEEE Security and Privacy Workshops, Workshop on Research in Insider Threats, WRIT'14*, 2014.
22. F. Kammüller and C. W. Probst. Modeling and verification of insider threats using logical analysis. *IEEE Systems Journal, Special issue on Insider Threats to Information Security, Digital Espionage, and Counter Intelligence*, 11(2):534–545, 2017.
23. F. Kammüller. A formal development cycle for security engineering in isabelle, 2020. arxiv preprint, `http://arxiv.org/abs/2001.08983`.
24. T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL – A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer-Verlag, 2002.
25. L. C. Paulson. The inductive approach to verifying cryptographic protocols. *Journal of Computer Security*, 6(1-2):85–128, 1998.
26. W. Pieters. Explanation and trust: What to tell the user in security and ai? *Ethics and Information Technology*, 13(1):53–64, 2011.
27. L. Viganó and D. Magazzeni. Explainable security. In *IEEE European Symposium on Security and Privacy Workshops, EuroS&PW*. IEEE, 2020.

28. D. Windridge and F. Kammüller. Edit distance kernelization of np theorem proving for polynomial-time machine learning of proof heuristics. In *Future of Information and Communications Conference, FICC 2019*, volume 70 of *Advances in Information and Communication. Lecture Notes in Networks and Systems*. Springer, 2019.