Research Plan:

(Advancing) Formal Verification of Authentication Protocols

Student:
Felix E. Linker
flinker@inf.ethz.ch

Supervisor:
Prof. Dr. David Basin
basin@inf.ethz.ch

Start of Matriculation and Employment: July 1st, 2020

Abstract

This document provides a research plan for the doctoral dissertation of Felix E. Linker. We aim for two main lines of research in this dissertation. First, the design and verification of An Authenticated Digital EMblem (ADEM) [18] and a messaging app authentication protocol. We have already begun the design and analysis of the system ADEM. ADEM allows one to verify digital infrastructure as protected under international humanitarian law. Additionally, ADEM will provide a specification of how to endorse parties to protect their infrastructure accordingly. The latter was inspired by the Automatic Certificate Management Environment (ACME) protocol [1]. ACME likewise inspired us to pursue a messaging app authentication protocol, where users can authenticate each other by proving control over social media accounts. Our second line of research is the advancement of the protocol verification tool Tamarin [23]. Here, we plan to add better support for recursive models to Tamarin utilizing algebraic data types and structural induction. Moreover, we will investigate the compartmentalisation of Tamarin models into multiple modules, facilitating the verification of infrastructure comprising multiple protocols.

1 Introduction

The Needham–Schroeder (NS) protocol [26] and the ACME protocol [1] illustrate the relevance of protocol verification: For the NS protocol, a critical bug was only found years later [20], and for the ACME protocol formal verification managed to uncover a subtle flaw [11]. These two examples together show that formal verification is not only powerful, but also that if not employed, critical bugs can go unnoticed to the public for years.

Yet, automated protocol verification tools such as Tamarin [23] struggle in two areas: the verification of unbounded, looping, or recursive protocols, e.g., the Signal protocol [21], and succinctly expressing and verifying the composition of protocols. To illustrate the latter problem, take the ACME protocol as an example. The security of the modern Internet relies on many factors: how certificate authorities (CAs) issue certificates, e.g., using ACME, how certificates are checked by modern browsers, how certificates are distributed via TLS [29], how certificate management is observed via certificate-transparency, how certificate-revocation is performed, etc. Attackers are not bound to break any single protocol's security properties to achieve their means. Security is well-known to be a non-monotonic property: Although system A might satisfy P, and system B might satisfy P, it is not guaranteed that all compositions of these two systems will satisfy P. Consequently, although the ACME protocol may have been verified to meet certain properties, these properties could be the *wrong* ones in context of the whole web public key infrastructure (PKI) [8].

We hit both of these barriers of protocol verification while designing and verifying the system ADEM [18]. ADEM tackles the challenge of transferring the idea of protective emblems, e.g., the red cross, into the digital world: ADEM enables one to mark digital entities, such as virtual servers or laptops, as protected under international humanitarian law. The recursive aspect of ADEM's verification are certificate chains that are used to endorse legitimate claimants of protection. The composition aspect of ADEM's verification is its heterogenous nature. Showing a digital alternative of a red cross to the *public* requires ADEM not only to support a wide range of entities to mark as protected, but also a wide range of protocols to communicate these markings. Additionally, ADEM relies on the security of other protocols such as TLS, ACME, and certificate transparency, and hence, composed analysis, taking all these parts into account would be desirable.

While analyzing the interplay of ADEM and the ACME protocol, another research gap became apparent: while it is well known that using public key encryption, authentic channels can be transformed into secret channels (e.g., formalized in the *channel calculus* [22]), a similar, canonical way to bootstrap authentic channels is not known. The well-known systematization of how to perform entity authentication for users by *something you know* (e.g., passwords), *something you have* (e.g., authenticators), or *something you are* (e.g., biometrics) [28] provides a starting point. However, both this systematization and the channel calculus do not allow one to fully grasp what ACME practically embodies: authentication by *something you control* (in case of the ACME protocol one must control a Domain Name System (DNS) entry or IP address).

A promising application of *something you control* paradigm could be to bridge the gap of end-to-end authenticity in messaging applications such as Signal or WhatsApp. Both messengers provide strong confidentiality guarantees, *given* that the channel is established with the right client. The client authentication, however, is only performed via SMS one-time passwords (OTPs). One can argue, that for most users, SMS authentication - although insecure [12] - is sufficient to provide them with high chances of connecting to the right clients. However, this not only assumes the security of SMS authentication, but also that users are able to exchange authentic phone numbers. A final line of research is to add stronger authenticity guarantees to messaging applications by providing clients with the ability to prove control over social media accounts. This way, a chat in a messaging app would not only be authenticated via a phone number, but also via Twitter, Google, or Facebook accounts. We believe that this not only improves the security of messaging applications, but also provides end-users with more understandable security guarantees: a messaging app chat should be secure as long as all authentication factors are secure, and end-users likely have a clear mental model for what it means to control a social media account.

2 State-of-the-Art

2.1 Authentication Protocols

First, we distinguish two meanings of the term "authentication protocol". It could mean to use a protocol as a channel to perform entity authentication, i.e., verifying who it is that one is communicating with (usually utilizing *something you* . . .), or it could mean to provide protocols with authentication properties [19], which express in varying degrees that the communication partners agree on certain values if they successfully finish a run of their protocol. Authentication as a property is especially critical for authentication and key agreement (AKA) protocols such as the 5G-AKA protocol [31].

Kaufman, Perlman, and Speciner [13] additionally discusses entity authentication from a different viewpoint than *something you . . .*, namely the delegation of entity authentication to third parties by *trusted intermediates*. In short, such intermediates do the job of entity authentication for you, e.g., using the ACME protocol, and give you a "receipt" of this authentication, e.g., a certificate. Kaufman, Perlman, and Speciner distinguish three settings of trusted intermediates: CAs, as in the well-known web PKI, and both single and multiple key distribution centers (KDCs). The level of trust put into this intermediate depends on the concrete system.

The authentication and delegation systems OAuth 2.0 [9], OpenID Connect [30], Kerberos [34], and CONIKS [24] all implement a central trusted intermediate. Kerberos additionally supports the model of multiple trusted intermediates, supporting access ticket delegation. OAuth 2.0, OpenID Connect, and Kerberos are all concerned with granting users access to resources or services. In this model, a user requests access to a resource or service hosted by a so called *relying party*. The relying party has offloaded the duty of authentication to one of these system. To gain access, the users authenticates to the system and is handed an access token or ticket that can be checked by the relying party in collaboration with the trusted intermediate.

CONIKS on the other hand focusses on providing users with authentic public keys for other users of a same service. In doing so, it tries to tackle a similar problem as the web PKI. ClaimChain [16] tries to solve the same problem as CONIKS, but is fully distributed and, hence, does not implement any trusted intermediate. Both CONIKS and ClaimChain aim to provide authentic public keys by focussing on key consistency, i.e., that everyone agrees on the key associated with an identity. In CONIKS, the consistency is expected to be maintained by a central provider, whereas in ClaimChain, every user is monitored to ensure that consistent claims are made about other users' keys.

2.2 Protocol Verification

There are three main protocol verification tools in the symbolic model: Tamarin [23], ProVerif [4], and DY* [3]. In the symbolic model, messages are represented by terms. Atomic messages are represented by *symbolic* variables and operations on messages are represented by functions, e.g., an encrypted message `m` is represented by the term `enc(m, k)`. To reason about the effects of computation, usually, equational theories are employed. Equational theories induce equivalence classes among terms. For example, the term `dec(enc(m, k), k)` resembling the decryption of the encryption of message `m` under key `k` is axiomatized to be equal to `m`.

Protocol verification tools in the symbolic model make the *perfect cryptography assumption*, which, for example, states that messages can be decrypted if and only if one knows the decryption key. These assumptions are not made when protocols are proven to be correct in the *computational model*. In this model, though, proofs are much more subtle, complex, and manual. Security proofs in the symbolic model should not be interpreted as proofs of definite security, but the symbolic model has proven its value by uncovering real bugs in real protocols [11, 2].

Tamarin uses multiset rewriting in the symbolic model and evaluates security properties as constraint systems using backwards search. ProVerif uses the applied pi calculus as an input language and evaluates security properties as Horn clauses. There exists a compiler from the applied calculus to Tamarin multiset rewriting theories, making Tamarin and ProVerif highly comparable. Hence, in this paragraph, we focus on the differences of ProVerif and Tamarin. In contrast to Tamarin, ProVerif supports symbolic model of bit strings, e.g., allowing the detection of type-flaw attacks. In ProVerif, the decryption of an arbitrary term can return "garbage", whereas in Tamarin, it merely is represented as a term to which no further equations can be applied. In contrast to

ProVerif, however, Tamarin is more interactive. Tamarin allows manual proofs of models, which facilitates debugging, but also allows to complete proofs that could not have been found by any of Tamarin's heuristics.

Whereas ProVerif was less interactive than Tamarin, DY* is on the other end of the spectrum and almost completely manual. DY* is written in the dependently typed, functional programming language F* [33]. DY* provides a framework within F* to encode and verify protocols. DY* is powerful, but only supports one proof strategy that must be supported by manually annotating invariants, shifting the majority of the burden to the modeller.

2.3 Formal Methods & Automated Induction

Protocol verification is a subfield in the vast area of formal methods. In this section, we investigate three other lines of research in the area of formal methods: program verification, interactive theorem proving, and general purpose model checkers.

Program verification, for example as performed by the Viper tool [25], is related to our field of research for two reasons: First, models in Tamarin could be read both as non-deterministic functional or logic programs. Second, program verification has long been tackling the problems of automated induction and modular verification. An immediate difference between protocol and program verification is that in the latter case, a machine-readable model "comes for free", i.e., someone has written code that must now be verified. This makes programs usually much more complex, facilitating the need for modular verification. Program verification tools usually tackle this complexity additionally by requiring manual annotations from programmers, e.g., in the case of Viper, pre- and post-conditions. Due to this high reliance on manual annotations, proofs of induction about, e.g., loops in programs are mostly interactive. The user is required to annotate loops with sufficiently strong invariants or the proof will not succeed.

The works of Kraan [15] on logic programming, and the theorem provers NQTHM [5] and ACL2 [14] for Common LISP can also be considered as program verification tools. However, Kraan [15] puts special emphasis on logic program synthesis alongside the synthesized program's verification. All three tools, though, focus on automated proofs by induction, for example, the theorem provers NQTHM and ACL2 famously introduced the Boyer-Moore-Waterfall, a scheme to automate induction.

General purpose model checkers, such as Alloy [10], nuXmv [7], or TLA+ [17], are similar to Tamarin - a model checker itself. Interestingly, these model checkers all support partitioning formal models, e.g., into modules, which is not supported by Tamarin. Although these model checkers are general purpose, they each come with a targeted verification domain. Alloy focusses on object oriented programming models, nuXmv on hardware verification, and TLA+ on concurrent algorithms.

3 Goals of Thesis

We plan two follow two main lines of research: the design and verification of authentication protocols, and advancing the verification of such protocols. These two lines go hand-in-hand, as the gaps in protocol verification were uncovered by the design of the new authentication protocols.

3.1 Formal Verification of Authentication Protocols

Research questions. The development of ADEM was inspired by the web PKI, and - in part - relies on it. It follows the model of CAs as trusted intermediates. ADEM builds on top of well-established technology, e.g., ACME, the web PKI, and TLS, but combines them in a unique way due to its unique constraints that arise from the humanitarian context. The question that has not been answered so far: How can such novel combinations of existing technology can be accounted for in protocol verification?

ACME pioneered the approach of establishing authenticity-over-control. What has been considered so far, is how this approach can be generalized, applied in different contexts, and systematized in comparison to other entity authentication mechanisms. In our research, we plan to expand the scope of authenticity-over-control, by prototyping the approach for end-user authentication in messaging apps. Additionally, we plan to investigate other usage contexts in which ADEM could be used. If the semantics of emblems in ADEM would be changed to not signal protection, but rather prove affiliation to arbitrary entities (to companies, groups, the government, . . .), the architecture might be deployed in scenarios that provide no possibility for automated authentication so far.

Methods and contributions. We will found the development of these systems on rigorous requirements engineering, system specification, prototype implementation, and formal verification of the underlying models. We expect to make three main contributions: the design, implementation and verification of ADEM and a messaging app authentication protocol, and a systematization of the notion authenticity-over-control.

3.2 Advancing Protocol Verification

Research questions. The goal of this thesis is to advance protocol verification in two ways: supporting more powerful induction schemes to improve the handling of recursive models, and supporting better was to modularize Tamarin models such that the composition of protocols can be analyzed more clearly. We decided to extend Tamarin, because it strikes the perfect balance between interactivity and automation in comparison to ProVerif and DY*. It is well-known that no scheme for automated induction can be complete [6]. Furthermore, our induction scheme will be based on heuristics. Leveraging the interactive mode of Tamarin will allow modelers to gain insight into the attempts of Tamarin to proof lemmata by induction, noticing non-termination or manually guiding the tool whenever heuristics fail.

The works on logic programming and Common LISP theorem proving have shown that powerful heuristics for automated induction exist, however, given the nature of the problem, they cannot be expected to work at every time. Our goal is to build on this research and apply it to Tamarin, with the hopes of not only providing better induction schemes, but maybe also uncovering particular types of models for which automated induction works especially well.

Modular verification is implemented in Viper, and was performed using the theorem prover Isabelle [27, 32]. At the same time, many general purpose model checkers provide the option to modularize and plug together different models. However, specification composition is not straightforward. Although Tamarin supports lemmata marked as "reuse", which could facilitate model compositions, it remains unclear, how approaches of modularization could be adopted to Tamarin models. A final goal of this thesis is to investigate if both notions of modular verification (separating Tamarin into modules itself, and performing verification piece by piece) are feasible in Tamarin.

Methods and contributions. The contributions of a more powerful protocol verification tool are obvious: protocols can be verified more accurately and more efficiently, both in terms computational power and modeling effort.

Extending Tamarin with new features requires four kinds of work. For every addition to Tamarin, in a first step, its syntax and semantics need to be defined, and how automated verification strategies can deal with these features. Second, the new features must be implemented. Third, their soundness must be proven. And fourth, the new features need to be evaluated, preferably on case studies.

4 Work Plan

We divide the goals of our thesis into four projects: the design and analysis of ADEM, the design and analysis of a messaging app authentication protocol, improving Tamarin's handling of recursive models, and adding support for modularization to Tamarin. We lay out a detailed work schedule at the end of this section in Table 1.

4.1 ADEM

The ADEM project comprises three work packages. These cannot be clearly separated from each other, but should overall not take more than 12 months. The design and implementation of ADEM constitutes the first work package. The second work package is given by the formal verification of ADEM as a whole. To pursue the standardization of ADEM in an international context marks the third and final work package. We note, that the time needed for this final work package cannot be estimated accurately, as it is highly dependent on external factors. We expect, though, that standardization will go in parallel with all other work packages.

The design of ADEM has seen good progress, and we have taken first steps towards a standardization of the protocol. ADEM comprises multiple protocols, and each of them has been at least sketched as a design. Partially, the protocols have already been subject to external feedback, e.g., from the International Comittee of the Red Cross (ICRC). In parallel, we are investigating user stories helping us to design ADEM's practical aspects, e.g., deployment in context of a humanitarian mission.

4.2 Messaging App Authentication

Given that this project has a much narrower scope than ADEM, we group it as a single work package that comprises both the exploration of existing technologies that could solve similar problems, the formal analysis of our approach, and the implementation as a prototype. Overall, we expect this work package to take 6 months.

We already finished a rough analysis of the authentication process's requirements, especially the criticality of privacy. Additionally, we identified existing components in the Internet's infrastructure that we might be able to alter or use to deploy the protocol. Currently, we are investigating the trade-offs between these components, how much provider buy-in would be required, and what kind of properties they offer.

4.3 Verifying Recursive Models in Tamarin

As mentioned in the previous section, we plan to add better support for recursive models to Tamarin in two steps. First, we plan to add algebraic data types to Tamarin. Second, we plan to add structural induction schemes on algebraic data types to Tamarin. Each of these steps constitutes a work package. For both steps, we plan to first define clear motivational examples in the form of case studies that either cannot be modelled or proven in Tamarin currently, are very cumbersome to model, or require a considerate manual intervention or computing resources to be verified. In a subsequent step, we define the syntax and user interface of Tamarin's new features, and finally, how automated proving shall be performed. For these additions, we will provide soundness proofs, ensuring that the new theories deliver correct results. Finally, Tamarin's new features need to be implemented and evaluated both with respect to performance and to the case studies identified earlier. We expect both work packages to take 4 months each.

For both the addition of algebraic data types, and the support of structural induction, we identified motivating examples, sketched examples illustrating the syntax envisioned, and implemented case-study-level prototypes. All these steps demonstrate the feasibility of our idea. Next up are the detailed analysis of the case studies, especially in light of the current possibilities of protocol verification tools.

4.4 Modular Models in Tamarin

This research goal constitutes a single work package. In general, we expect the same steps to be taken as layed out in the previous paragraph. However, the design of the syntax needs to be considered with special caution. As a new first step, existing models that are often reused, such as TLS, will be investigated. We will face the questions: Can interfaces to protocol models be defined? Can Tamarin's modeling features be unified such that these interfaces are expected to behave uniformly for different models? Can existing lemmata be reused to facilitate new proofs? And most critically, does a new compositional approach provide any value over the naive approach of abstracting components manually as channels that are assumed to provide certain properties? We believe that this work can be tackled within 6 months.

		2021		2022				2023			
		Past	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
ADEM	Design Verification Standardization	▨	▨	▨	▨	▨					
Messaging Authentication		▨	. . .	. . .	. . .	▨	▨	▨			
Recursion	ADTs	▨	▨	▨							
	Induction	▨	. . .	▨	▨						
Modularization						▨	▨	▨			

Table 1: Detailed Work Schedule

4.5 Expected Publications

We estimate that each of our projects could result in a publication containing its results. From the outset, it is hard to judge how many results will come out of each of these projects. Maybe, different projects can be combined into a single publication, e.g., ADEM serves as a motivational example for better handling of recursive models in Tamarin. In total, we expect to end with the following *publishable results*:

- Design and verification of ADEM
- Design and verification of messaging app authentication
- Systematization of authentication-over-control
- Addition of algebraic data types to Tamarin
- Addition of structural induction schemes to Tamarin
- Addition of modularization to Tamarin

5 Additional Information

I have finished my study programme as a doctoral student by attending the courses 252-1414-00S *System Security* (7 ECTS), and 263-4660-00S *Applied Cryptography* (8 ECTS). I assist the teaching of the courses 252-0463-00L *Security Engineering*, and 252-0058-00L *Formal Methods and Functional Programming* where I give exercises session for half a semester each. I have no other duties.

References

[1] Richard Barnes et al. *Automatic Certificate Management Environment (ACME)*. Request for Comments RFC 8555. Num Pages: 95. Internet Engineering Task Force, Mar. 2019. DOI: 10.17487/RFC8555. URL: https://datatracker.ietf.org/doc/rfc8555 (visited on 09/23/2021).

[2] David Basin, Ralf Sasse, and Jorge Toro-Pozo. "The EMV Standard: Break, Fix, Verify". In: *arXiv:2006.08249 [cs]* (Feb. 17, 2021). arXiv: 2006.08249. URL: http://arxiv.org/abs/2006.08249 (visited on 09/24/2021).

[3] Karthikeyan Bhargavan et al. "DY* : A Modular Symbolic Verification Framework for Executable Cryptographic Protocol Code". In: EuroS&P 2021 - 6th IEEE European Symposium on Security and Privacy. Sept. 6, 2021. URL: https://hal.inria.fr/hal-03178425 (visited on 09/23/2021).

[4] Bruno Blanchet. "Modeling and Verifying Security Protocols with the Applied Pi Calculus and ProVerif". In: *Foundations and Trends® in Privacy and Security* 1.1 (Oct. 30, 2016), pp. 1–135. ISSN: 2474-1558, 2474-1566. DOI: 10.1561/3300000004. URL: https://www.nowpublishers.com/article/Details/SEC-004 (visited on 09/23/2021).

[5] R. S. Boyer, M. Kaufmann, and J. S. Moore. "The Boyer-Moore theorem prover and its interactive enhancement". In: *Computers & Mathematics with Applications* 29.2 (Jan. 1, 1995), pp. 27–62. ISSN: 0898-1221. DOI: 10.1016/0898-1221(94)00215-7. URL: https://www.sciencedirect.com/science/article/pii/0898122194002157 (visited on 09/24/2021).

[6] Alan Bundy. "Chapter 13 - The Automation of Proof by Mathematical Induction". In: *Handbook of Automated Reasoning*. Ed. by Alan Robinson and Andrei Voronkov. Handbook of Automated Reasoning. Amsterdam: North-Holland, Jan. 1, 2001, pp. 845–911. ISBN: 978-0-444-50813-3. DOI: 10.1016/B978-044450813-3/50015-1. URL: https://www.sciencedirect.com/science/article/pii/B9780444508133500151 (visited on 09/28/2021).

[7] Roberto Cavada et al. "The nuXmv Symbolic Model Checker". In: *Computer Aided Verification*. Ed. by Armin Biere and Roderick Bloem. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2014, pp. 334–342. ISBN: 978-3-319-08867-9. DOI: 10.1007/978-3-319-08867-9_22.

[8] Laurent Chuat et al. "SoK: Delegation and Revocation, the Missing Links in the Web's Chain of Trust". In: *2020 IEEE European Symposium on Security and Privacy (EuroS P)*. 2020 IEEE European Symposium on Security and Privacy (EuroS P). Sept. 2020, pp. 624–638. DOI: 10.1109/EuroSP48549.2020.00046.

[9] Dick Hardt. *The OAuth 2.0 Authorization Framework*. Request for Comments RFC 6749. Num Pages: 76. Internet Engineering Task Force, Oct. 2012. DOI: 10.17487/RFC6749. URL: https://datatracker.ietf.org/doc/rfc6749 (visited on 09/24/2021).

[10] Daniel Jackson. *Software Abstractions: Logic, Language, and Analysis*. Cambridge, MA, USA: MIT Press, Mar. 24, 2006. 366 pp. ISBN: 978-0-262-10114-1.

[11] Dennis Jackson et al. "Seems Legit: Automated Analysis of Subtle Attacks on Protocols that Use Signatures". In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. CCS '19. New York, NY, USA: Association for Computing Machinery, Nov. 6, 2019, pp. 2165–2180. ISBN: 978-1-4503-6747-9. DOI: 10.1145/3319535.3339813. URL: https://doi.org/10.1145/3319535.3339813 (visited on 09/23/2021).

[12] Roger Piqueras Jover. "Security Analysis of SMS as a Second Factor of Authentication: The challenges of multifactor authentication based on SMS, including cellular security deficiencies, SS7 exploits, and SIM swapping". In: *Queue* 18.4 (Aug. 31, 2020), Pages 20:37–Pages 20:60. ISSN: 1542-7730. DOI: 10.1145/3424302.3425909. URL: https://doi.org/10.1145/3424302.3425909 (visited on 09/23/2021).

[13] Charlie Kaufman, Radia Perlman, and Mike Speciner. *Network security private communication in a public world*. 2nd ed., 14th printing. Prentice Hall series in computer networking and distributed systems The Radia Perlman series in computer networking and security. Upper Saddle River, NJ: Prentice Hall PTR, 2011. 713 pp. ISBN: 978-0-13-046019-6.

[14] M. Kaufmann and J.S. Moore. "An industrial strength theorem prover for a logic based on Common Lisp". In: *IEEE Transactions on Software Engineering* 23.4 (Apr. 1997), pp. 203–213. ISSN: 1939-3520. DOI: 10.1109/32.588534.

[15] Ina Kraan. "Proof planning for logic program synthesis". In: (1994). Accepted: 2019-02-15T14:32:46Z Publisher: The University of Edinburgh. URL: https://era.ed.ac.uk/handle/1842/34920 (visited on 09/24/2021).

[16] Bogdan Kulynych et al. “ClaimChain: Improving the Security and Privacy of In-band Key Distribution for Messaging”. In: *Proceedings of the 2018 Workshop on Privacy in the Electronic Society*. WPES’18. New York, NY, USA: Association for Computing Machinery, Jan. 15, 2018, pp. 86–103. ISBN: 978-1-4503-5989-4. DOI: `10.1145/3267323.3268947`. URL: `https://doi.org/10.1145/3267323.3268947` (visited on 09/24/2021).

[17] Leslie Lamport. “Hybrid Systems in TLA+”. In: *Hybrid Systems, Robert L. Grossman, Anil Nerode, Hans Rischel, and Anders P. Ravn, editors. Lecture Notes in Computer Science, Springer-Verlag* 736 (Apr. 6, 1993), pp. 77–102. URL: `https://www.microsoft.com/en-us/research/publication/hybrid-systems-tla/` (visited on 09/24/2021).

[18] Felix Linker and David Basin. *Signaling legal protection during cyber warfare: an authenticated digital emblem*. Humanitarian Law & Policy Blog. Sept. 21, 2021. URL: `https://blogs.icrc.org/law-and-policy/2021/09/21/legal-protection-cyber-warfare-digital-emblem/` (visited on 09/23/2021).

[19] G. Lowe. “A hierarchy of authentication specifications”. In: *Proceedings 10th Computer Security Foundations Workshop*. Proceedings 10th Computer Security Foundations Workshop. June 1997, pp. 31–43. DOI: `10.1109/CSFW.1997.596782`.

[20] Gavin Lowe. “An attack on the Needham-Schroeder public-key authentication protocol”. In: *Information Processing Letters* 56.3 (Nov. 10, 1995), pp. 131–133. ISSN: 0020-0190. DOI: `10.1016/0020-0190(95)00144-2`. URL: `https://www.sciencedirect.com/science/article/pii/0020019095001442` (visited on 09/23/2021).

[21] Moxie Marlinspike. *Advanced cryptographic ratcheting*. Signal Messenger. Nov. 26, 2013. URL: `https://signal.org/blog/advanced-ratcheting/` (visited on 09/23/2021).

[22] Ueli M. Maurer and Pierre E. Schmid. “A Calculus for Secure Channel Establishment in Open Networks”. In: *Proceedings of the Third European Symposium on Research in Computer Security*. ESORICS ’94. Berlin, Heidelberg: Springer-Verlag, Nov. 7, 1994, pp. 175–192. ISBN: 978-3-540-58618-0. (Visited on 09/23/2021).

[23] Simon Meier et al. “The TAMARIN Prover for the Symbolic Analysis of Security Protocols”. In: *Computer Aided Verification*. Ed. by Natasha Sharygina and Helmut Veith. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2013, pp. 696–701. ISBN: 978-3-642-39799-8. DOI: `10.1007/978-3-642-39799-8_48`.

[24] Marcela S. Melara et al. “CONIKS: Bringing Key Transparency to End Users”. In: 24th {USENIX} Security Symposium ({USENIX} Security 15). 2015, pp. 383–398. ISBN: 978-1-939133-11-3. URL: `https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/melara` (visited on 09/24/2021).

[25] Peter Müller, Malte Schwerhoff, and Alexander J. Summers. “Viper: A Verification Infrastructure for Permission-Based Reasoning”. In: *Proceedings of the 17th International Conference on Verification, Model Checking, and Abstract Interpretation - Volume 9583*. VMCAI 2016. Berlin, Heidelberg: Springer-Verlag, Jan. 17, 2016, pp. 41–62. ISBN: 978-3-662-49121-8. DOI: `10.1007/978-3-662-49122-5_2`. URL: `https://doi.org/10.1007/978-3-662-49122-5_2` (visited on 09/23/2021).

[26] Roger M. Needham and Michael D. Schroeder. “Using encryption for authentication in large networks of computers”. In: *Communications of the ACM* 21.12 (Dec. 1, 1978), pp. 993–999. ISSN: 0001-0782. DOI: `10.1145/359657.359659`. URL: `https://doi.org/10.1145/359657.359659` (visited on 09/23/2021).

[27] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*. Lecture Notes in Computer Science. Berlin Heidelberg: Springer-Verlag, 2002. ISBN: 978-3-540-43376-7. URL: `https://www.springer.com/gp/book/9783540433767` (visited on 09/23/2021).

[28] L. O'Gorman. "Comparing passwords, tokens, and biometrics for user authentication". In: *Proceedings of the IEEE* 91.12 (Dec. 2003), pp. 2021–2040. ISSN: 1558-2256. DOI: `10.1109/JPROC.2003.819611`.

[29] Eric Rescorla. *The Transport Layer Security (TLS) Protocol Version 1.3*. Request for Comments RFC 8446. Num Pages: 160. Internet Engineering Task Force, Aug. 2018. DOI: `10.17487/RFC8446`. URL: `https://datatracker.ietf.org/doc/rfc8446` (visited on 09/23/2021).

[30] N. Sakimura et al. *OpenID Connect Core 1.0 incorporating errata set 1*. Nov. 8, 2014. URL: `https://openid.net/specs/openid-connect-core-1_0.html` (visited on 09/24/2021).

[31] *Security architecture and procedures for 5G System*. TS 133 501 V16.7.1. 3GPP, Aug. 2021. URL: `https://www.etsi.org/deliver/etsi_ts/133500_133599/133501/16.07.01_60/ts_133501v160701p.pdf` (visited on 09/28/2021).

[32] Christoph Sprenger et al. "Igloo: soundly linking compositional refinement and separation logic for distributed system verification". In: *Proceedings of the ACM on Programming Languages* 4 (OOPSLA Nov. 13, 2020), 152:1–152:31. DOI: `10.1145/3428220`. URL: `https://doi.org/10.1145/3428220` (visited on 09/23/2021).

[33] Nikhil Swamy et al. "Dependent types and multi-monadic effects in F*". In: *Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. POPL '16. New York, NY, USA: Association for Computing Machinery, Jan. 11, 2016, pp. 256–270. ISBN: 978-1-4503-3549-2. DOI: `10.1145/2837614.2837655`. URL: `https://doi.org/10.1145/2837614.2837655` (visited on 09/23/2021).

[34] Brian Tung and Larry Zhu. *Public Key Cryptography for Initial Authentication in Kerberos (PKINIT)*. Request for Comments RFC 4556. Num Pages: 42. Internet Engineering Task Force, June 2006. DOI: `10.17487/RFC4556`. URL: `https://datatracker.ietf.org/doc/rfc4556` (visited on 09/23/2021).

Signatures

Date Felix E. Linker (*Student*)	Date Prof. Dr. David Basin (*Supervisor*)