Provable Entity Accountability for Protocols with Modifiable Signed Messages

Reyhane Falanji
Linköping University
Sweden
reyhane.falanji@liu.se

Mikael Asplund
Linköping University
Sweden
mikael.asplund@liu.se

Niklas Carlsson
Linköping University
Sweden
niklas.carlsson@liu.se

***Abstract*—This study focuses on a formal analysis of protocols that utilize chameleon signatures. These protocols allow modifying signed messages while keeping the same signature valid. We provide definitions of unforgeability, non-repudiation, and non-frameability (ensuring accountability) at the protocol level, complementing earlier definitions applicable for cryptographic primitives. Protocol-level properties are essential to allow symbolic protocol verification and support ensuring accountability for all relevant entities involved in the message exchange. Furthermore, we propose a basic protocol for transferring modifiable signed messages, and formally verify accountability properties of the protocol. To enable analysis, we define an equational theory for chameleon signatures in the Tamarin theorem prover.**

I. Introduction

Conventional signature schemes provide source authentication and message integrity. However, these constructs are inefficient in applications where a signed message needs to be modified. Krawczyk and Rabin [1] introduce chameleon signatures that allow a designated modifier to modify signed data. Unforgeability, non-repudiation, and non-frameability of the signer are proven for chameleon signatures as cryptographic primitives. However, in general, it is not trivial to determine whether protocols that rely on sound cryptographic primitives can also provide the corresponding guarantees. There are cases where, despite the computational security guarantees of the underlying primitives, these guarantees are broken in the protocol layer in certain scenarios [2]. Moreover, as chameleon signatures introduce an additional entity (i.e., modifier), it is crucial to assess the properties that can be guaranteed for the modifier as well. Although later studies on chameleon signatures provide enhancements of the primitive [4], to the best of our knowledge, no study covers symbolic analysis of protocols built upon these signatures.

A potential use case of chameleon signatures is illustrated in Figure 1 based on an edge-assisted vehicular network [5]. A vehicle (signer) is willing to share a piece of information (e.g., a picture taken by its camera) with another vehicle (final receiver) in the network. However, the picture may contain information that needs to be modified (e.g., blurring a part of the picture to protect the privacy of pedestrians), and the computational load should be offloaded to a less resource-constraint device, such as an edge server.

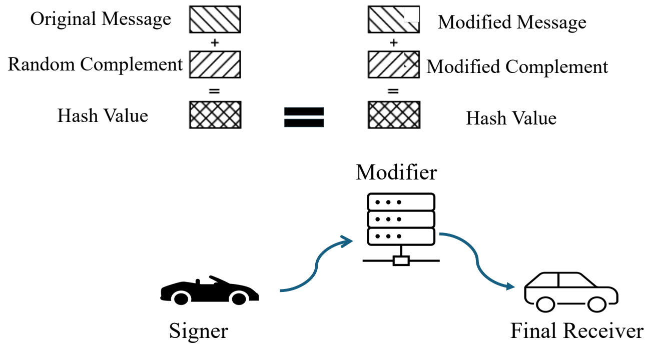

Fig. 1. An application of modifiable signed data in an edge-assisted vehicular network scenario.

As expected from any protocol utilizing signatures, unforgeability should hold for protocols relying on chameleon signatures as well. However, considering the legitimate modification allowed by the primitive, the definition of unforgeability differs from that in conventional signature schemes and includes the modifier as an additional source of the signed data. Moreover, since a modified version of the original data with the same signature might exist, it is essential to reason about the accountability of the participating parties in the protocol. Traditionally, a signer would be entirely responsible for a signature generated in a protocol unless the message has been tampered with. In the case of chameleon signatures, while the signer should be responsible for the signature on the original data (non-repudiation), it should not be framed for the modified data, even though the data has its signature on it (non-frameability). Similarly, the modifier should not have a way of repudiating the actions that it has performed (non-repudiation). At the same time, it should not be held responsible for the non-modified data (e.g., due to the signer's claim).

In this study, we formally analyze the entities' accountability in chameleon-signature-based protocols. We define protocol properties inspired by definitions of properties for chameleon signatures. The properties defined by [1] are appropriate for computational analysis, but cannot be applied at the protocol level. Our definitions are applicable in a labeled multiset rewriting formalism. Moreover, we propose a basic protocol that utilizes chameleon signatures for message passing. We model the protocol it in the Tamarin verifier [3]. While simple, the protocol is proven to meet all the expected accountability requirements. The proofs are achieved by introducing new equational theories in Tamarin for chameleon signatures.

II. Protocol Overview

This short paper only provides an intuitive overview of our proposed protocol. As illustrated in Figure 1, the protocol has three participating entities: a signer, a designated modifier, and a final receiver. To transfer a modifiable message, the signer calculates the chameleon hash on the message and a complement random value. Then, it calculates the signature of the hash and transfers the message, the random, and the signature to the modifier.

Upon receiving a signed message, the modifier creates a new message by modifying the old one. As allowed by the particular design of chameleon signatures, the modifier can then calculate a modified complement, which combined with the new message, results in the same hash value as the original message. Since the signature was on the hash value, and as the hash value remains intact, the signature remains valid on the new data. The modifier then transfers the new message, the modified complement, and the old signature to the final receiver, who can further verify the signature.

III. Desired Properties

In this section, we delve into the properties of a signer in the context of chameleon signatures. Later in section IV, we show how these properties can be applied to the modifier.

*1) **Unforgeability**:*

Among the three fundamental properties we discuss (unforgeability, non-repudiation, and non-frameability), unforgeability takes on a distinct meaning in chameleon signatures. In the context of traditional signatures, unforgeability translates into the inability of any entity other than the signer to produce a valid signature on a data (when validated on the signer's public key). However, unlike traditional signatures, chameleon signatures allow for legitimate modifications by a designated modifier. This legitimate modification should not be considered as a forgery, when analysing the unforgeability of the protocol. Simultaneously, the existence of a trapdoor secret for the modifier should not allow any third party (other than the signer and the modifier) to forge the signature by any means. Thereby we state unforgeability for a protocol that transfers modifiable signed messages as follows:

- *Unforgeability: Any existing data with a valid signature has either been produced by the signer or is a modification of an original message performed by the designated modifier.*

*2) **Non-repudiation**:*

Even though the unforgeability property implies that only the signer and the modifier can create a valid signature, it does not convey their accountability. In particular, it is unclear which entity is responsible for a particular signature. Providing the signer with a modified message rightfully gives it enough evidence to deny having produced and signed the modified message. However, the signer can misuse the same reasoning to repudiate an action that it has preformed. Suppose a signer can create a message and a complement value that results in the same signature as a previously produced one (i.e., an action only supposed to be done by the modifier). In that case, it can repudiate having produced the original message. Since trapdoor secret is an additional element of chameleon signatures compared to conventional ones, the existence of this secret should not give the signer an opportunity to repudiate its actions. Therefore, we define the non-repudiation of the signer as follows:

- *Non-repudiation: If two messages with the same valid signature exist, one must be the modification of another, performed by the designated modifier.*

*3) **Non-frameability**:*

The two former properties clarify responsible entities but do not state anything about entities being framed. The ability of the modifier to create new information with a valid signature of the signer should not allow the modifier to frame the signer for the message that the signer did not produce. Non-frameability is a less discussed property of signatures due to its context-dependent nature. However, it is important in chameleon signatures since the existence of a modifier creates a high probability of the signer being framed. Therefore, we define the property as:

- *Non-frameability: If the signer has not produced any signature, no entity (including the modifier) should be able to produce a message with a valid signature (on the signer's public key).*

IV. Verification and Results

We model the protocol mentioned in section II in the Tamarin formal verifier. Tamarin explores the whole state spaces of all traces of the protocol, attempting to find a counter-example for each property. It considers a Dolev-Yao attacker model, who can listen to, intercept, or inject messages. Under these assumptions, Tamarin does not find any counter-example for the properties, therefore, all three properties hold for the protocol.

Since unforgeability holds, we can claim that the non-repudiation and non-frameability also hold for the modifier. This is true because any valid signature has either been produced by the signer or kept valid by the modifier on modified data. Therefore, if the signer can not repudiate a signature (non-repudiation of the signer), and if only the signer and the modifier can be responsible (unforgeability), it means that the modifier can not be framed for a signature on data that it has not modified. Similarly, we can prove that non-repudiation holds for the modifier.

V. Conclusion

While existing works on chameleon signatures focus on computational analysis of these primitives, no study considers the properties of protocols that rely on chameleon signatures. Moreover, no existing definition of unforgeability, non-repudiation, and non-frameability can be used for symbolic verification of this class of protocols.

In this study, we consider a basic protocol built on the functionalities provided by chameleon signatures. We provide protocol-level definitions of unforgeability, non-repudiation, and non-frameability. Furthermore, we model the protocol in the Tamarin formal verifier and prove that the properties hold for both the signer and the modifier.

Acknowledgment

This research was financially supported by the Swedish Foundation for Strategic Research (SSF), project FUS21-0033 (ASTECC).

References

[1] Krawczyk, Hugo, and Tal Rabin. ”Chameleon hashing and signatures.” Cryptology ePrint Archive (1998).

[2] Cheval, Vincent, Cas Cremers, Alexander Dax, Lucca Hirschi, Charlie Jacomme, and Steve Kremer. ”Hash Gone Bad: Automated discovery of protocol attacks that exploit hash function weaknesses.” In 32nd USENIX Security Symposium (USENIX Security 23), pp. 5899-5916. 2023.

[3] Meier, Simon, Benedikt Schmidt, Cas Cremers, and David Basin. ”The TAMARIN prover for the symbolic analysis of security protocols.” In Computer Aided Verification: 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings 25, pp. 696-701. Springer Berlin Heidelberg, 2013.

[4] Camenisch, Jan, David Derler, Stephan Krenn, Henrich C. Pöhls, Kai Samelin, and Daniel Slamanig. ”Chameleon-hashes with ephemeral trapdoors: And applications to invisible sanitizable signatures.” In Public-Key Cryptography–PKC 2017: 20th IACR International Conference on Practice and Theory in Public-Key Cryptography, Amsterdam, The Netherlands, March 28-31, 2017, Proceedings, Part II 20, pp. 152-182. Springer Berlin Heidelberg, 2017.

[5] Xu, Junpeng, Haixia Chen, Xu Yang, Wei Wu, and Yongcheng Song. ”Verifiable image revision from chameleon hashes.” Cybersecurity 4 (2021): 1-13.