LOKI-2: An Improved Lightweight Cryptographic Key Distribution Protocol for Automotive Systems

Teri Lenard
Centre Universitaire d'Informatique,
Geneva School of Economics and Management,
University of Geneva, Switzerland
teri.lenard@unige.ch

Bèla Genge
Department of Electrical Engineering and
Information Technology, George Emil Palade
University of Medicine, Pharmacy, Science, and
Technology of Targu Mures, Romania
bela.genge@umfst.ro

Anastasija Collen
Centre Universitaire d'Informatique,
Geneva School of Economics and Management,
University of Geneva, Switzerland
anastasija.collen@unige.ch

Niels A. Nijdam
Centre Universitaire d'Informatique,
Geneva School of Economics and Management,
University of Geneva, Switzerland
niels.nijdam@unige.ch

***Abstract*—The Lightweight Cryptographic Key Distribution Protocol (LOKI) was initially proposed as a solution that enables computationally restricted automotive control units to periodically update short-term authentication keys. After analysing the protocol messaging from a security and a formal point of view, we found that the protocol lacks critical security properties. The work at hand introduces an improved protocol version that intents to address the initial design flaws. Moreover, a formal automated verification was conducted on both protocols, first to demonstrate the shortcomings of LOKI, and secondly, the correctness of the improved variant. Finally, the improved protocol is formally verified with a Burrows–Abadi–Needham (BAN) logic analysis.**

***Index Terms*—group key distribution protocol, automotive systems, controller area network, scyther, Burrows–Abadi–Needham logic**

I. Introduction

Automotive system encompass a wide variety of communication protocols that enable control units and digital sensors to exchange information, with the purpose of automating and controlling different subsystems of a vehicle. The cybersecurity dimension of automotive systems received significant attention in the past decade [1] [2], motivated by the lack of built-in security features these systems are equipped with [3]. This phenomenon stimulated researchers to design and build specially tailored security protocols that intent to securely distribute cryptographic keys, and authenticate message exchanges.

While novel protocol ideas can be gathered from the literature, a security protocol can not be considered trustworthy until it was formally verified and its correctness demonstrated. One protocol example that illustrates the issues that can occur when formal validation is omitted is the previously proposed Lightweight Cryptographic Key Distribution Protocol (LOKI) [4]. The LOKI protocol was proposed as a solution to distribute short-term authentication keys for a group of resource constrained automotive control units that communicate over Controller Area Network (CAN). LOKI considers the presence of a key distributor and a group of key receivers. In LOKI, the key distributor does not explicitly send the next short-term key to the group, but it generates an update event via a Message Authentication Code (MAC) that triggers the generation of the new key on each group member. Each member generates the new key locally based on a Key Derivation Function (KDF) constructed with the help of a reverse hash-chain traversal algorithm [5]. The protocol assumes that each group member is bootstrapped with a symmetric long-term key, the first short-term symmetric key, and several constant values (e.g., group id). Lastly, in terms of freshness, LOKI considers timestamps obtained from a time synchronisation protocol.

By analysing the messaging structure of LOKI from a security design and formal point of view, with the Scyther [6] modelling tool and Burrows–Abadi–Needham (BAN) logic [7], we discovered the following protocol drawbacks. Firstly, the broadcast message leveraged by the key distributor to trigger the generation of the new short-term key lacks sufficient freshness. The message contains only a timestamp freshness value in the order of seconds, which may not be timely received via a secure channel. The MAC message contains along the timestamp, a constant group id, frame id, and the current short-term key. Secondly, since each control unit in the group shares the same long-term symmetric key, it is impossible to determine the source of the key generation message (e.g., key distributor). Unfortunately, this fact enables an arbitrary member of the group to triggering the key generation process. Consequently, an attacker is only required to compromise one receiver member, not even the distributor, to disrupt the protocol.

To address these shortcomings, the current paper introduces an Imporved Lightweight Cryptographic Key Distribution Protocol (LOKI-2). The new version of the protocol corrects the initial design mistakes, by introducing proper freshness

979-8-3503-7035-5/23$31.00

usage, key and message source identification. Moreover, both, LOKI and LOKI-2 are formally analysed using the Scyther protocol verification tool. Lastly, we prove the correctness of the improved protocol version with a BAN logic analysis.

The paper continues with the related work in Section II. Afterwards, our proposed work is presented in Section III. Subsequently, the formal analysis we conducted is described in Section IV, followed by a BAN logic analysis of the improved protocol in Section V. We conclude the paper in Section VI.

II. Related Work

Automotive systems include multiple communication protocols, leveraged by control units and digital sensors to exchange frames. The CAN protocol is widely used between digital sensors and control units, Single Edge Nibble Transmission (SENT) and Local Interconnected Network (LIN) is strictly for digital sensors, and Media Oriented System Transport (MOST) for media oriented units. A telematic control unit interconnects these sub-networks and acts as a gateway between them, routing and translating frames from one sub-network to another. In CAN, the packet type is determined based on its CAN identifier field, lacking concrete naming in terms of source and destination. The frame identifier additionally plays the role in determining the frame priority in the bit wise bus access arbitration.

The CAN is a bus-oriented broadcast communication protocol, that was designed to be reliable, and resistant to errors. Nevertheless, the CAN protocol was not designed with built-in security features. In consequence, this sparked a wide variety of security solutions to be proposed in the literature as pointed out by surveys conducted by [2] and [8]. Moreover, studies similar to the one from [9] define concrete threats and countermeasures specifically for CAN.

Among the earliest works that target security protocols for in-vehicle communication networks is Van Herrewege et al. [10]. The authors developed a data authentication protocol for the CAN that consider MACs as a basis. Likewise, data authentication received an impressive amount of attention from works such as [11] [**?**]. Similar works, mostly leverage symmetric encryption schemes along freshness nonces. Their scope is to achieve low network and computational impact on control units, while maintaining a level of compliance with security standard recommendations from Automotive Open System Architecture (AUTOSAR) [12].

Particularly in the context of key distribution protocols, Groza et al. [13] offered a comparative analysis for different modalities of distributing keys over CAN, covering Diffie–Hellman approaches and identity-based authentication key agreement protocols. In a different direction, Palaniswamy et al. [14] developed a protocol suite covering authentication, key management, and secure session key update for in-vehicle units. Jain and Guajardo [15] leveraged the physical properties of the CAN bus to demonstrate how control units can securely exchange and derive group keys. Likewise, works such as [16] [17] address the problem of group key distribution over CAN.

Compare to the related work, the current paper targets to address several security design mistakes of a previously proposed group key distribution protocol. While the main paper focused on design and neglected formal validation, the current work accentuates the latter point through an automated formal analysis under the Dolev-Yao adversary model, and through a BAN logic analysis.

III. Proposed Work

A. Initial Protocol

Let I denote the protocol initiator which triggers the key derivation process in a group of protocol receivers R. In LOKI, I and R are bootstrapped with a shared symmetric long-term key K, corresponding to the group they are members of. Each group, is identified by a group id g_{id}. Furthermore, both I and R store an array of hash values z generated from a hash-chain, and the first short-term key k, with $k \in z$. Meaning that k is the first value of the hash-chain. The array of hashes z corresponds to the KDF algorithm named Speed2Pebbling [5]. Accordingly, a sequence of intermediary hash values are stored to traverse in reverse a hash-chain with low computation complexity. More information can be found in the LOKI paper [4] and in the algorithm original work [5].

LOKI contains two sub-protocols, one for key generation, and the other for key synchronisation. To generate a short-term new key k, I broadcasts a MAC computed with K, over k, t, c_{id} and the g_{id}, where t is a timestamp and c_{id} is the CAN frame identifier. If R successfully verifies the received MAC, R runs the KDF algorithm over z to obtain the new key $k + 1$. This process is displayed in Fig. 1.

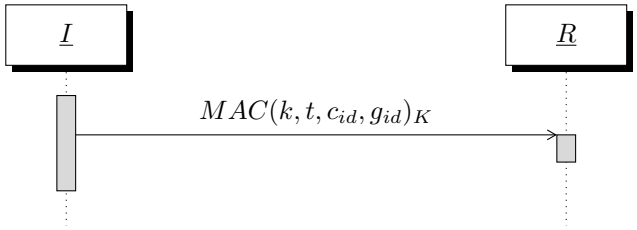

Fig. 1. LOKI message exchange to generate a new key.

To avoid frequent miss-synchronisations and errors, LOKI assumes that t is in the order of seconds. This fact leaves an open window for an attacker to try to predict or brute-force the timestamp value. Moreover, as can be seen in Fig. 1, LOKI does not contain sufficient identifiable information to determine the source of the message. Furthermore, I would know that R successfully updated its key k only indirectly, by observing messages authenticated by R with k, or by using k in communication with R. If not, I can not acknowledge if the protocol was executed correctly. This leaves I vulnerable to trivial man-in-the-middle attacks.

In the synchronisation protocol, LOKI follows a challenge-response pattern along an authentication test [18]. If R wants to synchronise its key k and hash values z with I, R sends a challenge to I containing a nonce n, and $MAC(n, t, c_{id}, g_{id})_K$

computed with K. If I receives the challenge, it responds with the encrypted $\{z\}_K$ hash values, along a MAC that additionally includes $\{z\}_K$ and the incremented nonce $n+1$. This process is outlined in Fig. 2. A drawback here is with regard to K, as R can not distinguish if the response to the challenge came from I, or from another group member. Consequently, a form of addressing or identification is missing.

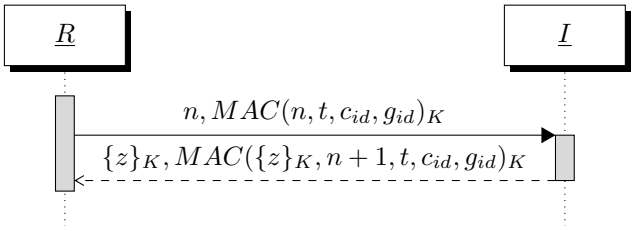

Fig. 2. LOKI message exchange in the synchronisation protocol.

B. Improved Protocol

The improved variant LOKI-2 addresses the security issues mentioned so far in LOKI, while maintaining its philosophy. The improvements target the message exchange between I and R, and exclude the KDF. LOKI-2 switches from a symmetric long-term key to a pair of long-term asymmetric keys. Additionally, it removes redundant communication terms such as c_{id}, introduces a shared secret nonce n_s instead of timestamps, a key id k_{id}, and confirmation messages for the key generation process.

The roles from LOKI are preserved in LOKI-2. In LOKI-2, I is bootstrapped with a long-term asymmetric key. Let pk_I denote the public part and sk_I the secret part. On the other hand, a group member R is bootstrapped with pk_I. Moreover, I and R share a secret nonce n_s, the hash array z, the first short-term key k, the g_{id} and its id k_{id}.

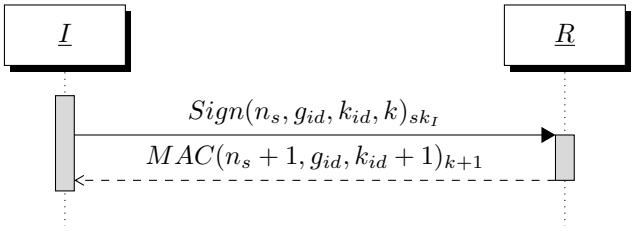

Fig. 3. LOKI-2 message exchange for key generation.

Compared to LOKI, where a MAC is broadcasted by I to trigger the key generation process, LOKI-2 broadcasts a digital signature computed over n_s, g_{id}, k_{id} and k with sk_I. Upon receiving this message, each R can determine the source of the message with pk_I. Afterwards, if the received n_s, g_{id}, k_{id} and k from I match up with the locally stored values, resulting in a successful verification of the signature, then R computes the next short-term key $k+1$. Upon completing the process, R responds back to I with a MAC tag computed with $k+1$ demonstrating to R that the new key was generated successfully. The MAC tag is computed over an incremented n_s+1, g_{id} and the new key id $k_{id}+1$. The mentioned steps are outlined in Fig. 3.

Next, the LOKI-2 synchronisation protocol is presented. Following the same challenge-response pattern, R sends I a challenge nonce n along the g_{id} encrypted with I's public key pk_I. After this event, I responds back with the current hash array z, the current k_{id}, g_{id} and the incremented nonce $n+1$, everything being encrypted with sk_I. As can be seen in Fig. 4, the asymmetric version of the synchronisation protocol consists in less terms used, with fewer cryptographic operations, while maintaining at the same time the necessary security properties.

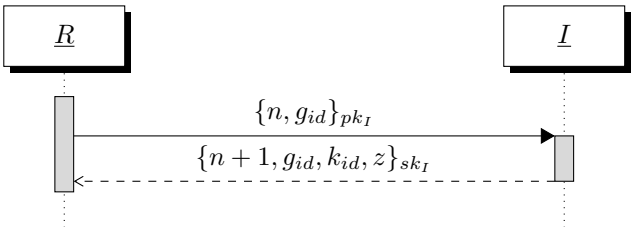

Fig. 4. LOKI-2 message exchange in the synchronisation protocol.

IV. Formal Analysis

The formal analysis was conducted with the protocol verification tool Scyther, developed by Cremers [6]. Scyther provides an abstract language for writing protocol specifications. Moreover, Scyther analyses protocol specifications under perfect encryption assumptions with the Dolev-Yao [19] adversary model. This implies that the adversary possesses complete control over the communication channel, it can eavesdrop, manipulate and replay network messages, and execute the same cryptographic operations as the protocol members. To prove the correctness of protocol $p(i, r)$ specification, where i denotes the protocol initiator and r the protocol responder, Scyther considers protocols claims [20] to demonstrate Proof of Correctness (PoC). The following claims were considered in the present analysis:

- *Secrecy (secret)*: A secrecy claim for a term $t \in T$ of an agent i is true, if and only if the term t never becomes known to the intruder.
- *Aliveness (alive)*: A protocol $p(i, r)$ satisfies the property of aliveness, if and only if after an agent i executes a round of $p(i, r)$, i is sure that it communicates with a trusted agent r, and r executed an event.
- *Weak agreement (weakagree)*: A protocol $p(i, r)$ satisfies the property of weak agreement, if and only if an agent i is sure that agent r executed the correct role in the protocol.
- *Non-injective synchronisation (nisynch)*: Represents a strong form of authentication in protocol $p(i, r)$. The claim is satisfied if all messages exchanged in $p(i, r)$ have been sent or received by i or r.
- *Non-injective agreement (niagree)*: A protocol $p(i, r)$ satisfies the claim of non-injective agreement, if whenever i

completes a run of the protocol believing it communicates with r, then r has run the protocol before believing to be communicating with i. Consequently i and r must agree on the contents of messages exchanged.

Subsequently, we define the protocol specification according to [20] [6]. Let $LOKI(i,r)$ denote the initial version of LOKI, $LOKI2(i,r)$ the improved version of $LOKI(i,r)$, $LOKIS(i,r)$ the key synchronisation protocol of LOKI, and $LOKIS2(i,r)$ the improved synchronisation protocol. First, we introduce the protocol specification for $LOKI(i,r)$:

$$\begin{aligned} LOKI(i,r) = (&\{i,r,t,g_{id},c_{id},k\},\\ &[send_1(i,r,\{t,g_{id},c_{id},k\}_k)\\ &\ recv_1(i,r,\{t,g_{id},c_{id},k\}_k)\\ &\ claim(i,r,secret,t)\\ &\ claim(i,r,secret,g_{id})\\ &\ claim(i,r,secret,c_{id})\\ &\ claim(i,r,secret,k)\\ &\ claim(i,r,alive)\\ &\ claim(i,r,weakagree)\\ &\ claim(i,r,niagree)\\ &\ claim(i,r,nisynch)]). \end{aligned} \quad (1)$$

In Equation (1), and likewise in Equations (2) (3) and (4) the term $send(i,r,m)$ denotes a message m sent from i to r, $recv(i,r,m)$ denotes that r received m from i, and $claim(i,r,c,x)$ refers to the claim c raised for both i and r in regards to x.

After running the protocol specification from Equation (1) with Scyther, we obtained the output from Table I. Here, it can be seen two claims were not verified, aliveness and weak agreement for the role i. This is an expected outcome since $LOKI(i,r)$ does not contain acknowledgement messages, r knows it communicates with i, but i is not aware of r. This shortcoming is addressed in $LOKI2(i,r)$.

Compared to $LOKI(i,r)$, the $LOKI2(i,r)$ introduces an acknowledgement message from r to i, and a shared secret freshness nonce n_s to address the freshness problems identified in $LOKI(i,r)$. While the first message of the protocol uses the long-term sk_I, the acknowledgement response expected from r is computed with the new short-term authentication key $k+1$. This way, r can provide i a proof that the new key was generated and can be used successfully. Below the protocol specification for $LOKI2(i,r)$ is defined:

$$\begin{aligned} LOKI2(i,r) = (&\{i,r,n_s,g_{id},k_{id},k,k+1,sk_I\},\\ &[send_1(i,r,\{n_s,g_{id},k_{id},k\}_{sk_I})\\ &\ recv_1(i,r,\{n_s,g_{id},k_{id},k\}_{sk_I})\\ &\ send_2(r,i,\{n_s+1,g_{id},k_{id}+1\}_{k+1})\\ &\ recv_2(r,i,\{n_s+1,g_{id},k_{id}+1\}_{k+1})\\ &\ claim(i,r,secret,n_s)\\ &\ claim(i,r,secret,n_s+1)\\ &\ claim(i,r,secret,g_{id})\\ &\ claim(i,r,secret,k_{id})\\ &\ claim(i,r,secret,k_{id}+1)\\ &\ claim(i,r,secret,k)\\ &\ claim(i,r,secret,k+1)\\ &\ claim(i,r,alive)\\ &\ claim(i,r,weakagree)\\ &\ claim(i,r,niagree)\\ &\ claim(i,r,nisynch)]). \end{aligned} \quad (2)$$

All the claims raised in the improved key distribution protocol $LOKI2(i,r)$ hold, as can be seen in Table II.

TABLE I
SCYTHER ANALYSIS OUTCOME FOR THE INITIAL VERSION OF LOKI.

Protocol	Claim	Term	PoC
$LOKI(i,r)$	secret	t	Ok
	secret	g	Ok
	secret	c_{id}	Ok
	secret	k	Ok
	alive	-	Fail for i, Ok for r
	weakagree	-	Fail for i, Ok for r
	niagree	-	Ok
	nisynch	-	Ok

TABLE II
SCYTHER ANALYSIS OUTCOME FOR THE IMPROVED VERSION LOKI-2.

Protocol	Claim	Term	PoC
$LOKI2(i,r)$	secret	n_s, n_s+1	Ok
	secret	g_{id}	Ok
	secret	$k_{id}, k_{id}+1$	Ok
	secret	$k, k+1$	Ok
	alive	-	Ok
	weakagree	-	Ok
	niagree	-	Ok
	nisynch	-	Ok

Next, we verify the key synchronisation protocol $LOKIS(i,r)$, with the protocol specification:

$$\begin{aligned} LOKIS(i,r) = (&\{i,r,n,g_{id},c_{id},z,t,K\},\\ &[send_1(r,i,n,\{n,t,g_{id},c_{id}\}_K)\\ &\ recv_1(r,i,n,\{n,t,g_{id},c_{id}\}_K)\\ &\ send_2(i,r,\{z\}_K,\{n+1,t+1,g_{id},c_{id},z\}_K)\\ &\ recv_2(i,r,\{z\}_K,\{n+1,t+1,g_{id},c_{id},z\}_K)\\ &\ claim(i,r,secret,n+1) \end{aligned}$$

$$
\begin{aligned}
&claim(i, r, secret, t, t+1)\\
&claim(i, r, secret, g_{id})\\
&claim(i, r, secret, c_{id})\\
&claim(i, r, secret, z)\\
&claim(i, r, alive)\\
&claim(i, r, weakagree)\\
&claim(i, r, niagree)\\
&claim(i, r, nisynch)]). \qquad (3)
\end{aligned}
$$

As shown in Table III the claims raised in the initial version of $LOKIS(i, r)$ hold.

TABLE III
SCYTHER ANALYSIS OUTCOME FOR THE SYNCHRONISATION PROTOCOL OF LOKI.

Protocol	Claim	Term	PoC
$LOKIS(i, r)$	secret	$n+1$	Ok
	secret	t	Ok
	secret	g_{id}	Ok
	secret	c_{id}	Ok
	secret	k_{id}	Ok
	secret	z	Ok
	alive	-	Ok
	weakagree	-	Ok
	niagree	-	Ok
	nisynch	-	Ok

Lastly, the protocol specification for the LOKI-2 synchronisation protocol $LOKI2S(i, r)$ is outlined:

$$
\begin{aligned}
LOKIS2(i, r) = (\{&i, r, n, g_{id}, z, k_{id}, pk_I, sk_I\},\\
&[send_1(r, i, \{n, g_{id}\}_{pk_i})\\
&\ recv_1(r, i, \{n, g_{id}\}_{pk_i})\\
&\ send_2(i, r, \{n+1, g_{id}, k_{id}, z\}_{sk_I})\\
&\ recv_2(i, r, \{n+1, g_{id}, k_{id}, z\}_{sk_I})\\
&\ claim(i, r, secret, n+1)\\
&\ claim(i, r, secret, g_{id})\\
&\ claim(i, r, secret, k_{id})\\
&\ claim(i, r, secret, z)\\
&\ claim(i, r, alive)\\
&\ claim(i, r, weakagree)\\
&\ claim(i, r, niagree)\\
&\ claim(i, r, nisynch)]). \qquad (4)
\end{aligned}
$$

To conclude the analysis, the claims raised in $LOKI2S(i, r)$ hold as shown in Table IV.

TABLE IV
SCYTHER ANALYSIS OUTCOME FOR THE SYNCHRONISATION PROTOCOL OF LOKI-2.

Protocol	Claim	Term	PoC
$LOKIS2(i, r)$	secret	$n, n+1$	Ok
	secret	g_{id}	Ok
	secret	k_{id}	Ok
	secret	z	Ok
	alive	-	Ok
	weakagree	-	Ok
	niagree	-	Ok
	nisynch	-	Ok

V. BAN LOGIC ANALYSIS

BAN logic [7] is a formal logic destined to analyse security protocols. The goal of BAN logic is to provide a framework through which a protocol designer can demonstrate the correctness of a security protocol from a formal point of view, abstracting from low level particularities. BAN logic is considered to verify the correctness from another formal point of view of the improved protocol LOKI-2.

In a BAN logic analysis there are two epochs in which a protocol is executed, the past and the present. Since a protocol execution begins in the present epoch, it is important to guarantee that messages exchanged in the past do not affect believes in the present epoch. Likewise, believes held in the present epoch should be stable enough to not be altered by past ones. Additionally, the BAN logic abstracts from low level particularities regarding the protocol implementation or the encryption scheme used.

In this logic of authentication, protocol participants are denoted as *principals*, each principal holding similar or different believes about protocol terms (e.g., secret keys, freshness nonces), denoted in BAN logic as *formulas* or *statements*. The notations for the aforementioned constructs can be viewed in Table V. Here, A, B or I, R denote principals participating in the protocol, K_{ab} a shared key between A and B, K_a and K_a^{-1} public and private keys of principal A, N_a a statement, and X, Y a range over statements.

TABLE V
BAN LOGIC NOTATION.

Notation	Definition
A, B, I, R	Principals
K_{ab}	Shared key between A and B
K_a	Public key of A
K_a^{-1}	Private key of A
N_a	Statement (e.g., nonce of a)
X, Y	Range over statements

BAN logic defines a series of constructs to express different assumptions and authentication goals about a protocol. The BAN logic constructs are outlined in Table VI. Along constructs, the logic leverages *postulates* or deduction rules for formal proofs. Burrows et al. [7] defines 5 main postulates.

The first rule is the *message meaning rule*, expressed as:

TABLE VI
BAN LOGIC CONSTRUCTS.

Notation	Definition
$A \mid\equiv X$	A **believes** X
$A \triangleleft X$	A **sees** X
$A \mid\sim X$	A **said** X
$A \Rightarrow X$	A **controls** X
#(X)	X **is fresh**
$A \overset{K}{\longleftrightarrow} B$	A **and** B **share** K
$\langle X \rangle_Y$	X **combines** Y
$\{X\}_K$	X **encrypted with** K
$\overset{k}{\longmapsto} B$	K **pubkey of** B

$$\frac{A \textbf{ believes } A \overset{K}{\longleftrightarrow} B, A \textbf{ sees } \{X\}_K}{A \textbf{ believes } (B \textbf{ said } X)},$$

which is equivalent to:

$$\frac{A \mid\equiv A \overset{K}{\longleftrightarrow} B, A \triangleleft \{X\}_K}{A \mid\equiv (B \mid\sim X)}.$$

The above rule can be adapted for public keys as follows:

$$\frac{A \mid\equiv A \overset{K}{\longleftrightarrow} B, A \triangleleft \{X\}_{K^{-1}}}{A \mid\equiv (B \mid\sim X)}.$$

The message meaning rule can be interpreted as follows. If a principal A believes it shares a secret key K with a principal B, and if A received a message X encrypted with key K from B, then A is allowed to believe that B once said X. The same rationale applies for the public/private key and shared secret variants of the rule.

The *nonce-verification rule* is the second formula denoted as:

$$\frac{A \textbf{ believes } X \textbf{ is fresh }, A \textbf{ believes } (B \textbf{ said } X)}{A \textbf{ believes } (B \textbf{ believes } X)},$$

which is equivalent to:

$$\frac{A \mid\equiv \#(X), A \mid\equiv (B \mid\sim X)}{A \mid\equiv (B \mid\equiv X)}.$$

Here, the rule states that if a principal A believes that a statement X is fresh, and if A additionally believes that B once said X, then in consequence A believes that B believes X.

Thirdly, there is the *jurisdiction rule*:

$$\frac{A \textbf{ believes } (B \textbf{ controls } X), A \textbf{ believes } (B \textbf{ believes } X)}{A \textbf{ believes } X},$$

that can be written formally as:

$$\frac{A \mid\equiv (B \Rightarrow X), A \mid\equiv (B \mid\equiv X)}{A \mid\equiv X}.$$

In the jurisdiction statement, if a principal A believes the there is a principal B that has control over X, and B also believes X, then in consequence A will believe X. Forth postulate concerns the fact that if a principal is able to see a formula, then it also sees its components since it knows the necessary keys:

$$\frac{A \textbf{ sees } (Y, X)}{A \textbf{ sees } X}, \quad \frac{A \textbf{ sees } \langle X \rangle_Y}{A \textbf{ sees } \mathbf{X}},$$

$$\frac{A \textbf{ believes } B \overset{K}{\longleftrightarrow} A, A \textbf{ sees } \{X\}_K}{A \textbf{ sees } \mathbf{X}},$$

$$\frac{A \textbf{ believes } \overset{K}{\longmapsto} B, A \textbf{ sees } \{X\}_K}{A \textbf{ sees } X},$$

$$\frac{A \textbf{ believes } \overset{K}{\longmapsto} B, A \textbf{ sees } \{X\}_{K^{-1}}}{A \textbf{ sees } X}.$$

Lastly, a rule is given stating that if a formula is fresh, the entire formula must also be fresh:

$$\frac{A \textbf{ belives fresh}(X)}{A \textbf{ belives fresh}(X, Y)}.$$

Analysing a protocol with BAN logic implies the following steps:

- **Step I:** For each protocol principal define a set of assumption about the protocol.
- **Step II:** Identify for each principal the authentication goals that must hold.
- **Step III:** Define the list of messages of the protocol.
- **Step IV:** Define an idealised version of the protocol by replacing protocol terms with formulas.
- **Step V:** Apply on each protocol message postulates to verify the protocol.

Let I denote a principal which initiates the protocol, and R a principal which receives and responds to messages from I. Additionally, let K_i denote the I's public key shared with R, and K_i^{-1} the private key part. For convenience, we preserve the notation from Section IV for other statements. The BAN logic analysis begins with LOKI-2 and afterwards continues with the LOKI-2 synchronisation protocol. Accordingly, the following assumptions are raised for LOKI-2:

LOKI-2 assumptions	
I believes:	R believes:
$R \overset{K_i}{\longleftrightarrow} I$	$I \overset{K_i}{\longleftrightarrow} R$
$R \overset{N_s}{\longleftrightarrow} I$	$I \overset{N_s}{\longleftrightarrow} R$
$R \overset{k}{\longleftrightarrow} I$	$I \overset{k}{\longleftrightarrow} R$
$\#(N_s + 1)$	$\#(N_s)$
	$\overset{K_i}{\longmapsto} I$
$R \overset{k+1}{\longleftrightarrow} I$	$I \Rightarrow K_i^{-1}$
$I \Rightarrow k + 1$	$I \Rightarrow k + 1$

The LOKI-2 assumption for I infer that I believes that R shares with I the nonce N_s, the public key K_i, and the short-term key k. Additionally, I believes the incremented nonce

$N_s + 1$ is fresh. From R's perspective, R shares the first three assumptions as I, with the addition of the following. R believes that the nonce N_s is fresh, K_i is the public key of I and K_i^{-1} is the private key of I and is controlled by I, and I controls the new short-term key $k+1$. Subsequently, the authentication goals are specified:

LOKI-2 authentication goals	
I believes:	R believes:
	$I \overset{k+1}{\longleftrightarrow} R$

The authentication goals entail that after the protocol is executed, I and R share the same $k+1$ short-term key. The LOKI-2 analysis continues by defining the protocol message sequence and the idealised protocol. The Message sequence is:

1. $I \rightarrow R : \{N_s, g_{id}, k_{id}, k\}_{K_i^{-1}}$
2. $R \rightarrow I : \{N_s + 1, g_{id}, k_{id} + 1\}_{k+1}$,

and the idealised version:

1. $I \rightarrow R : \{I \overset{N_s}{\longleftrightarrow} R, I \overset{g_{id}}{\longleftrightarrow} R, I \overset{k_{id}}{\longleftrightarrow} R, I \overset{k}{\longleftrightarrow} R\}_{K_i^{-1}}$
2. $R \rightarrow I : \{N_s + 1, I \overset{g_{id}}{\longleftrightarrow}, k_{id} + 1\}_{k+1}$.

In Message 1, it is stated that $R \triangleleft \{I \overset{N_s}{\longleftrightarrow} R, I \overset{g_{id}}{\longleftrightarrow} R, I \overset{k_{id}}{\longleftrightarrow} R, I \overset{k}{\longleftrightarrow} R\}_{K_i^{-1}}$. The message meaning rule for asymmetric keys is applied on $M = (I \overset{N_s}{\longleftrightarrow} R, I \overset{g_{id}}{\longleftrightarrow} R, I \overset{k_{id}}{\longleftrightarrow} R, I \overset{k}{\longleftrightarrow} R)$:

$$\frac{R \mid\equiv I \overset{K_i}{\longleftrightarrow} R, R \triangleleft \{M\}_{K_i^{-1}}}{R \mid\equiv I \mid\sim M},$$

because $R \mid\equiv \overset{K_i}{\longmapsto} I$. The rule holds for N_s, g_{id}, k_{id} and k since $N_s \in M$, $g_{id} \in M$, $k_{id} \in M$ and $k \in M$. Likewise, the following postulate is verified:

$$\frac{R \mid\equiv \overset{K_i}{\longmapsto} R, R \triangleleft \{M\}_{K_i^{-1}}}{R \triangleleft M}.$$

Moreover, since $N_s \in M$ we can apply the freshness postulate with $R \mid\equiv \#(N_S)$ and obtain $R \mid\equiv \#(M)$. Furthermore, the the nonce-verification rule holds:

$$\frac{R \mid\equiv \#(N_s), R \mid\equiv (I \mid\sim N_s)}{R \mid\equiv (I \mid\equiv N_s)}.$$

Lastly, the jurisdiction rule states that:

$$\frac{R \mid\equiv (I \Rightarrow K_i^{-1}), R \mid\equiv I \mid\equiv (I \Rightarrow K_i^{-1})}{R \mid\equiv K_i^{-1}},$$

and

$$\frac{R \mid\equiv (I \Rightarrow I \overset{k+1}{\longleftrightarrow} R), R \mid\equiv (I \mid\equiv I \overset{k+1}{\longleftrightarrow} R)}{R \mid\equiv I \overset{k+1}{\longleftrightarrow} R}.$$

This concludes the analysis of Message 1. We continue with Message 2 where $I \triangleleft \{N_s + 1, I \overset{g_{id}}{\longleftrightarrow}, k_{id} + 1\}_{k+1}$. Likewise Message 1, we first apply the massage meaning rule on Message 2 and obtain $I \mid\equiv R \mid\sim M$, where $M = (N_s + 1, I \overset{g_{id}}{\longleftrightarrow}, k_{id} + 1)$. Since $I \mid\equiv \#(N_s + 1)$ we postulate that:

$$\frac{I \mid\equiv \#(N_s + 1)}{I \mid\equiv \#(M)},$$

given that $N_s + 1 \in M$. The nonce verification rule results in $I \mid\equiv (R \mid\equiv N_s + 1)$, and and from the jurisdiction rule that $I \mid\equiv R \overset{k+1}{\longleftrightarrow} I$. From the assumption $I \Rightarrow k + 1$ we obtain $I \mid\equiv R \overset{k+1}{\longleftrightarrow} I$. This concludes the analysis of LOKI-2. The analysis continues with the synchronisation protocol. Here, the protocol assumptions are:

LOKI-2S assumptions	
I believes:	R believes:
$\#(N)$	$\#(N+1)$
$R \overset{K_i}{\longleftrightarrow} I$	$I \overset{K_i}{\longleftrightarrow} R$
$R \overset{g_{id}}{\longleftrightarrow} I$	$I \overset{g_{id}}{\longleftrightarrow} R$
	$\overset{K_i}{\longmapsto} I$
	$I \Rightarrow z$

In this case, the assumption for I state that I believes that I shares the public key K_i and g_{id} with R, and additionally I believes N is fresh. As for R, R believes it shares K_i and g_{id} with I, $N+1$ is fresh, K_i is the public key of I, and I controls z. Likewise, the authentication goals are defined:

LOKI-2S authentication goals	
I believes:	R believes:
$R \overset{z}{\longleftrightarrow} I$	$I \overset{z}{\longleftrightarrow} R$

The authentication goals state that after the protocol is executed, I and R should share the same hash values z from which R can deduce the short-term key k. The analysis continues by defining the protocol message sequence:

1. $R \rightarrow I : \{N, g_{id}\}_{pk_I}$
2. $I \rightarrow R : \{N + 1, g_{id}, k_{id}, z\}_{sk_I}$,

and the idealised version as:

1. $R \rightarrow I : \{\#(N), I \overset{g_{id}}{\longleftrightarrow} R\}_{pk_I}$
2. $I \rightarrow R : \{\#(N+1), I \overset{g_{id}}{\longleftrightarrow} R, k_{id}, z\}_{sk_I}$.

The message meaning rule for asymmetric key is applied on Message 1 with the assumption that $I \mid\equiv (I \overset{K_i}{\longleftrightarrow} R)$ giving $I \mid\equiv (R \mid\sim M)$, where $M = (\{\#(N), I \overset{g_{id}}{\longleftrightarrow} R\}_{K_i})$. Since I believes $\#(n)$ the nonce-verification rule gives $I \mid\equiv (R \mid\equiv \#(N))$, and since $n \in M$ we obtain that $I \mid\equiv (R \mid\equiv \#(M)$. This concludes the analysis of Message 1. For Message 2, the message meaning rule gives $R \mid\equiv (I \mid\sim M_{K_i})$ because $R \mid\equiv$

$(I \overset{K_i}{\longleftrightarrow} R)$ and $R \mid\equiv\overset{K_i}{\longmapsto} I$. The nonce-verification rule states that $R \mid\equiv (I \mid\equiv \#(N+1))$, resulting in $R \mid\equiv (I \mid\equiv \#(M))$, where $M = (\{\#(N+1), I \overset{g_{id}}{\longleftrightarrow} R, k_{id}, z\}_{sk_I})$. Lastly, from the jurisdiction rule we can conclude that $R \mid\equiv I \overset{z}{\leftrightarrow} R$.

VI. Conclusion

The work at hand addresses several security flaws of the key distribution protocol LOKI. The LOKI protocol was designed to offer a lightweight security protocol that can generate fresh short-term group keys, without the need to communicate the actual short-term key. We have identified that LOKI lacks proper freshness usage, key distributor identification, and other security properties (e.g., aliveness, agreement) that makes it vulnerable to trivial attacks. These issues were addressed in an improved protocol named LOKI-2. A formal automated analysis was conducted to prove the correctness of the improved protocol variant, together with a BAN logic analysis to demonstrate the protocol's security properties.

Acknowledgment

This work is supported by the OPEVA project that has received funding within the Key Digital Technologies Joint Undertaking (KDT JU) from the European Union's Horizon Europe Programme under grant agreement No 101097267 and was co-funded by the Swiss State Secretariat for Education, Research and Innovation (SERI) and the Innosuisse – Swiss Innovation Agency. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or KDT JU. Neither the European Union nor the granting authority can be held responsible for them.

References

[1] C. Young, J. Zambreno, H. Olufowobi, and G. Bloom, "Survey of Automotive Controller Area Network Intrusion Detection Systems," *IEEE Design & Test*, vol. 36, no. 6, pp. 48–55, 12 2019.

[2] A. Martínez-Cruz, K. A. Ramírez-Gutiérrez, C. Feregrino-Uribe, and A. Morales-Reyes, "Security on in-vehicle communication protocols: Issues, challenges, and future research directions," *Computer Communications*, vol. 180, pp. 1–20, 12 2021.

[3] I. Pekaric, C. Sauerwein, S. Haselwanter, and M. Felderer, "A taxonomy of attack mechanisms in the automotive domain," *Computer Standards & Interfaces*, vol. 78, p. 103539, 2021.

[4] T. Lenard, R. Bolboaca, and B. Genge, "LOKI: A Lightweight Cryptographic Key Distribution Protocol for Controller Area Networks," in *2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing (ICCP)*. IEEE, 9 2020, pp. 513–519.

[5] B. Schoenmakers, "Explicit Optimal Binary Pebbling for One-Way Hash Chain Reversal," 2017, pp. 299–320.

[6] C. J. F. Cremers, "The Scyther Tool: Verification, Falsification, and Analysis of Security Protocols," in *Computer Aided Verification*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 414–418.

[7] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," *ACM Transactions on Computer Systems*, vol. 8, no. 1, pp. 18–36, 2 1990.

[8] R. S. Rathore, C. Hewage, O. Kaiwartya, and J. Lloret, "In-Vehicle Communication Cyber Security: Challenges and Solutions," *Sensors*, vol. 22, no. 17, p. 6679, 9 2022.

[9] H. J. Jo and W. Choi, "A Survey of Attacks on Controller Area Networks and Corresponding Countermeasures," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6123–6141, 7 2022.

[10] A. Van Herrewege, D. Singelee, and I. Verbauwhede, "CANAuth-a simple, backward compatible broadcast authentication protocol for CAN bus," in *ECRYPT workshop on Lightweight Cryptography*, vol. 2011, 2011, p. 20.

[11] B. Groza, S. Murvay, A. V. Herrewege, and I. Verbauwhede, "LiBrA-CAN," *ACM Transactions on Embedded Computing Systems*, vol. 16, no. 3, pp. 1–28, 8 2017.

[12] AUTOSAR, "Specification of Secure Onboard Communication AUTOSAR CP Release 4.3.1," *AUTOSAR*, 2017.

[13] B. Groza, L. Popa, and P.-S. Murvay, "CarINA - Car Sharing with IdeNtity Based Access Control Re-enforced by TPM," 2019, pp. 210–222.

[14] B. Palaniswamy, S. Camtepe, E. Foo, and J. Pieprzyk, "An Efficient Authentication Scheme for Intra-Vehicular Controller Area Network," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3107–3122, 2020.

[15] S. Jain and J. Guajardo, "Physical Layer Group Key Agreement for Automotive Controller Area Networks," 2016, pp. 85–105.

[16] M. D. Pesé, J. W. Schauer, J. Li, and K. G. Shin, "S2-CAN: Sufficiently Secure Controller Area Network," in *Annual Computer Security Applications Conference*. New York, NY, USA: ACM, 12 2021, pp. 425–438.

[17] Z.-A. Zhao, Y. Sun, D. Li, J. Cui, Z. Guan, and J. Liu, "A Scalable Security Protocol for Intravehicular Controller Area Network," *Security and Communication Networks*, vol. 2021, pp. 1–13, 12 2021.

[18] J. Guttman, "Security protocol design via authentication tests," in *Proceedings 15th IEEE Computer Security Foundations Workshop. CSFW-15*. IEEE Comput. Soc, 2002, pp. 92–103.

[19] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 3 1983.

[20] C. J. F. Cremers, *Scyther : semantics and verification of security protocols.* Technische Universiteit Eindhoven, 2006.