L–PEP: A Logic to Reason about Privacy–Enhancing Cryptography Protocols

Almudena Alcaide, Ali E. Abdallah[†], Ana I. González–Tablas and José M. de Fuentes

Department of Computer Science,
University Carlos II of Madrid, Spain.
{aalcaide, aigonzal, jfuentes}@inf.uc3m.es
[†]Institute for Computing Research,
London South Bank University, UK.
a.abdallah@lsbu.ac.uk

Abstract. In recent years, many cryptography protocols have been designed, for many different scenarios, with the purpose of preserving security of communications as well as privacy and anonymity of participant entities. In general, every proposed solution has possed a real challenge to the existing formal methods of protocol analysis and verification. The main goal of this work is the proposal of a logic to reason about privacy-enhancing monotonic and non–monotonic cryptography protocols. The new logic will be called L-PEP and it extends the existing Rubin's logic of beliefs.
The original publication is available at www.springerlink.com

1 Introduction

This work has been mainly motivated by the following two factors. On the one hand, organizations are increasingly introducing privacy preserving mechanisms to comply with laws and regulations and, to satisfy users' and employees' privacy requirements. As as result, newly designed system architectures require of additional privacy and security features such as anonymity and pseudonymity, unlinkability and unobservability of transactions, selective disclosure of information, privacy policies embedded in data, etc [1]. Many attempts have been made to design new cryptography protocols to provide such strong privacy properties. In particular, the set of building block primitives used in privacy–enhancing cryptography protocols include anonymous credentials, blind signatures, dual signatures, recursive hashing, zero–knowledge proofs, identity–based encryption, anonymous biometrics, etc. As it happens, in every occasion the proposed cryptographic solution has possed a real challenge to the existing formal methods of protocol analysis and verification as none of the existing methods include the necessary inference rules to formally reason about those operations [2].

To this regard, the formal modeling of protocol privacy requirements represents a significant step forward, as it will allow us to validate protocol privacy

properties, in the same way as it is done with security properties such us confidentiality, integrity or accessibility. However, not very much work has been done in this direction so far.

On the other hand, privacy–enhancing technologies have placed the emphasis on determining the so called *Private Data Life Cycle*. Private data life cycle starts when data is created and finishes when the data is destroyed. The aim of those new technologies and the security protocols deployed to such effect is to control the whole cycle, including implicit and explicit disclosures of information, deletion of data, custody constraints, etc. For example, a common privacy constraint describes users' requirements to determine how long for entities are entitled to use or store someone else's private information, which might have been received in one of the steps of a cryptographic protocol. That is, via privacy policies and agreements, users could force other entities to delete and destroy users' own personal data after a fixed amount of time or, if data are not further required for any of the purposes which they were stored for [3]. Formally, a *non–monotonic* security protocol refers to the non–incremental acquisition of knowledge by participants along the execution of a security protocol. In fact, participant entities might have to *forget* information or data which they once knew or held, during a particular protocol instance. Again, formal representation and verification of non–monotonic schemes is still pending a global solution and not very much work has been carried out in this direction so far.

Since the design of BAN logic by Burrows, Abadi and Needham [4], many derivatives of BAN, as well as new logics, have been defined to formally and monotonically reason about cryptographic protocols. They are also known as *Logics of Beliefs*, as protocol goal verification is based on the set of beliefs and possessions that entities hold at the end of a protocol execution (for example, entity A *believes* K_{AB} is a secret key to communicate with entity B). In this context, monotonicity refers to the way in which the knowledge and the beliefs of a particular entity increase as the protocol execution progresses. In a monotonic protocol, once something is known, it is known forever.

By contrast, a non–monotonic logic would allow us to reason about non–monotonic protocols, in which entities might *stop believing* in a piece of information, or *forget* knowledge during the execution of a protocol. This is, in fact, the type of reasoning it is needed for the formal verification of many of today's privacy–enhancing cryptography protocols which rely on the deletion of information. Only a few attempts have been made to elaborate on this type of reasoning [5–7]. In [5] authors define a *not* operator to construct negation. In [6, 7] authors include actions to directly delete beliefs from entities' belief sets. We have chosen Rubin's logic [7] to be the focus of our study. The main reasons being that Rubin's logic is more tractable than Moser's [6] and furthermore, it allows us to differentiate between non–monotonicity of knowledge and non–monotonicity of belief. In addition, Rubin's logic does not require protocol *idealization* and the notation and the reasoning are similar to the original BAN logic [4].

1.1 Overview of Our Work

The main goal of this work is the proposal of a logic to reason about privacy-enhancing cryptographic protocols. The new logic will be called L-PEP and it extends Rubin's logic of beliefs [7]. Rubin's logic is extended by adding new inference rules and actions and by enhancing some of the original actions and rules. L-PEP, as presented in this paper, is defined to allow us to formally reason about cryptographic hashing, recursive cryptographic hashing, concatenation of hashes, dual signatures, identity–based encryption and blind signatures, all these being primitives of existing privacy–enhancing cryptography protocols. In this paper, and to illustrate how the formalism works, we formally analyze two different protocols. First, we apply the new L–PEP Logic to the analysis of the dual signature mechanism defined in SET (Secure Electronic Transaction protocol) [8]. Formal verification of dual signatures will be very relevant to other scenarios such as, privacy–preserving authentication and access control mechanisms. Second, we will formally reason about a privacy–enhancing *non–monotonic* protocol, applied to an electronic speed ticketing system in vehicular ad–hoc networks (VANETs). We will give a brief description of the system and analyze vehicular privacy related goals.

Previous related work can be found in [9–11]. In [9] and [10] authors use Rubin's Logic to formally analyze an e–commerce protocol and two authentication protocols respectively. However, the analysis is monotonic (i.e. only incremental knowledge is considered) and no signatures applied to hashed values (signatures with appendix) are formally considered during the protocols' formal analysis. Finally, in [11], a non–monotonic authenticated routing protocol is analyzed in which entities must *forget* other participant's public key certificates once these are revoked by the corresponding authorities. In that work, neither privacy related goals nor signatures applied to hashed values are formally analyzed.

The rest of the paper is organized as follows. In Section 2 we describe Rubin's logic main components including the additional features and enhancements proposed in this work. In Section 3 we illustrate the new formalism giving formal proof of privacy related goals regarding the Dual signature mechanism in the SET protocol. In Section 4 we illustrate the new formalism giving formal verification of a vehicular speed ticketing electronic protocol. Finally, Sections 5 and 5.1 are devoted to establish final considerations and future work, respectively.

2 Rubin's Logic

In this section we present an extension as well as few amendments to the non–monotonic logic introduced by Rubin in [7]. Some of the concepts and notation have been synthesized, please refer to [4] and [7] for a detailed description.

2.1 Non–monotonicity of Knowledge vs. Non–monotonicity of Belief

Rubin introduced a logic in which it is easy to difference between non–monotonicity of knowledge and non–monotonicity of belief in the following way: while entities

know things such as keys, nonces, or data, entities *believe* in metadata, such as freshness of data, or in other entities' possessions. When a principal stops believing in something it does not imply it also stops possessing such an item or knowing the data. By contrast, if an entity stops possessing or knowing an item, it does automatically stop believing in it as it does not longer hold it. This fact is very relevant in today's schemes. Principals are forced, via privacy agreements, to *forget* (stop storing and destroy) certain information and some protocols base their correctness on that premise. Furthermore, *forgetting* information implies readjustments in other principals' sets of beliefs. For example, a merchant *forgets* the user's card details after a transaction is processed, and the user *stops believing* that the merchant possesses such information.

2.2 Special notation:

$\sharp X$	Freshness of item X.
X *contains* x	Denotes x is part of item X.
$:=$	Denotes assignment.
$k_{P_i}^{-1}$	Denotes entity P_i private key.
k_{P_i}	Denotes entity P_i public key.
$\{X\}_k$	Denotes token X encrypted with key K.

2.3 Global definitions:

The specification of a protocol is the starting point of the analysis.

- **Set of participants:** W denotes the whole *World* of entities. $P = \{P_1, \ldots, P_n\} \subseteq W$ denotes the set of entities participating in the protocol.
- **Set of inference rules:** R denotes the set of inference rules. Each entity applies these rules to its sets of beliefs and knowledge.
- **Set of secrets:** S denotes the set of secrets. Data which is meant to be kept secret from the *World*. For each secret $s \in S$ the set ***Observers(s)*** denotes the set of entities which know secret s. Every communication is assumed to be listened by the whole *World* so after each communication, a function ***Update*** will recursively update the appropriate *Observers* sets. The Update functions also operates over the set of possessions of each entity.
- **TRUST Matrix :** A matrix TRUST is used to represent the trust relationship between every pair of principals. An entity P_i is said to trust entity P_j if P_i behaves as though P_j will follow the protocol.
- **Free variables:** FV denotes the set *free* variables to be instantiated along the formal proof.
- **Bound variables:** BV denotes the set *bound* variables which must hold the same value throughout the whole formal proof.

2.4 Definitions local to each protocol participant:

Local sets are private to each protocol participant.

- **Set of Possessions:** $POSS(P_i)$ denotes the possessions of participant P_i. For example: $POSS(P_i) = \{X, k_{P_i}^{-1}\}$. This is, P_i possesses (*knows*) X and private key $k_{P_i}^{-1}$.

- **Set of Beliefs:** $BEL(P_i)$ denotes the beliefs of participant P_i. For example: $BEL(P_i) = \{\sharp k, \{k_{P_j}^{-1}, k\} \in POSS(P_j)\}$. This is, P_i believes that k is a *fresh* key and that P_j is in possession of private key $k_{P_j}^{-1}$ and secret key k.

- **Behavior List:** $BL(P_i)$ denotes the behavior list of principal P_i. This list dictates to entity P_i what actions to take along the protocol execution and in which order. For example: $BL(P_i) = \langle bvr_1, brv_2, \ldots, bvr_n \rangle$. Where each bvr_i is of the form: $bvr_i = \{msg.operation; AL\}$. Where:
  - $msg.operation$ denotes one of the following operations: $send(P_j, m)$ or $receive(P_j, m)$, to denote sending a message m to P_j or receiving a message m from P_j, respectively. After a $receive(P_j, m)$ message operation is performed the corresponding $POSS$ sets are modified accordingly. After a *send* message operation is performed the ***Update*** function is recursively called to modify the appropriate *Observers* sets.
  - AL denotes an ordered list of zero or more *actions* that participants have to perform along the protocol execution. These actions are described in detail in section 2.5.

2.5 List of Actions

Describe how entities perform operations over data. All variables involved in these formula are considered to be bound variables. Only those actions related to our scope are listed and only the new ones are described in detail. Please refer to [7] for a complete list.

A1. **Apply**(f, X, n)
Condition: $\{f, X\} \subset POSS(P)$
Result: $POSS(P_i) := POSS(P_i), \cup\{f^l(X)\ \forall l \leq n\}$
Description: This new action is used when P_i applies the function f to X in a recursive way. Function f could be a hash function, a polynomial, a biometric template, a boolean function, XOR, etc.

A2. **Concat**(x_1, x_2)
Condition: $\{x_1, x_2\} \subset POSS(P_i)$
Result: $POSS(P_i) := POSS(P_i) \cup \{(x_1 \| x_2)\}$

A3. **Decrypt**$(\{X\}_k, k)$
Condition: $\{\{X\}_k, k\} \subset POSS(P_i), P_i \in Observers(k)$
Result: $POSS(P_i) := POSS(P_i) \cup \{X\}$

A4. **Decrypt–asymm**$(\{X\}_{k_{P_j}}; k_{P_j}^{-1})$
Condition: $\{\{X\}_{k_{P_j}}; k_{P_j}^{-1}\} \subset POSS(P_j),\ P_j \in Observers(k_{P_j}^{-1})$
Result: $POSS(P_j) := POSS(P_j) \cup \{X\}$

Description: This new action is used when a principal decrypts an encrypted item with the corresponding private key.

A5. **Decrypt–signature**$(\{X\}_{k_{P_j}^{-1}}; k_{P_j})$
Condition: $\{\{X\}_{k_{P_j}^{-1}}, k_{P_j}\} \subset POSS(P_i)$, $P_i \in Observers(k_{P_j})$
Result: $POSS(P_i) := POSS(P_i) \cup \{X\}$, $BEL(P_i) := BEL(P_i) \cup \{X \in POSS(P_j)\}$
Description: This new action is used when a principal decrypts an encrypted item with the corresponding public key. Note that this rule refers to *message recovery* signature schemes.

A6. **Encrypt**(X; k)
Condition: $\{X, k\} \subset POSS(P_i), P_i \in Observers(k)$
Result: $POSS(P_i) := POSS(P_i) \cup \{X\}_k$

A7. **Forget**(X)
Condition: $X \in POSS(P_i)$
Result: $POSS(P_i) := POSS(P_i) - \{X\}$, $BEL(P_i) := BEL(P_i) - \{\sharp X\}$, $\forall j$ *such that* $TRUST[j,i] = 1$ then $BEL(P_j) := BEL(P_j) - \{X \in POSS(P_i)\}$

A8. **Forget–secret**(s)
Condition: $P_i \in Observers(s)$, $s \in POSS(P_i)$
Result: $POSS(P_i) := POSS(P_i) - \{s\}$, $BEL(P_i) := BEL(P_i) - \{\sharp s\}$, $Observers(s) := Observers(s) - \{P_i\}$, $\forall j$ *such that* $TRUST[j,i] = 1$ then $BEL(P_j) := BEL(P_j) - \{s \in POSS(P_i)\}$

A9. **Generate–secret(s)**
Result: $S := S \cup \{s\}$, $Observers(s) := \{P_i\}$, $POSS(P_i) := POSS(P_i) \cup \{s\}$, $BEL(Pi) := BEL(Pi) \cup \{\#(s)\}$

A10. **Split**$(\{(x_1 \| x_2)\})$
Condition: $\{(x_1 \| x_2)\} \in POSS(P_i)$
Result: $POSS(P_i) := POSS(P_i) \cup \{x_1, x_2\}$

2.6 Set of Inference Rules

After each action, each participant updates its BEL set using all possible inference rules. Note that, rules are used to propagate belief during the protocol run whereas actions are used to propagate knowledge. We will now describe the most relevant rules for the scope of our work. Only the new ones are fully described.

R1. **Function Rule**
Condition: $\{\{f, X\} \subset POSS(P_i)\} \in BEL(P_j)$
Result: $\{Apply(f, X, n) \in POSS(P_i)\} \in BEL(P_j)$
Description: This new rule states that if P_j believes that P_i possesses f and X then, P_j believes that P_i possesses $f^i(X)$ $\forall 1 \leq i \leq n$

R2. **Message Meaning Rule**
Condition: $\{X\}_k \in POSS(P_i)$, $\{P_i, P_j\} \subseteq Observers(k)$
Result: $BEL(P_i) := BEL(P_i) \cup \{X \in POSS(P_j)\}$

R3. **Signature Verification Rule**

Condition: $\{\{h(X)\}_{k_{P_j}^{-1}}, X\} \in POSS(P_i), \{h(X) \in POSS(P_j)\} \in BEL(P_i)$

Result: $BEL(P_i) := BEL(P_i) \cup \{X \in POSS(P_j)\}$

Description: In [12] authors propose some new hash inference rules to add to BAN logic, however, these rules enforce sender authentication for every hash value received. By contrast, we propose a new signature verification rule such that, we are able to link a signed hash value to its original unsigned value by authenticating the signer entity of the hash. Indeed, the expression $\{h(X) \in POSS(P_j)\} \in BEL(P_i)$ is true if and only if the action Decrypt–signature($\{h(X)\}_{k_{P_j}^{-1}}$) has been applied successfully.

R4. **Split Rule**

Condition: $\{f, f(x_1) \parallel f(x_2)\} \in POSS(P_i)$

Result: $\{f(x_1), f(x_2)\} \in POSS(P_i)$

Description: This rule states that if P_i knows function f, then it can split concatenated items. This is common as knowing a function will give us information about the length of its output.

2.7 Formal Verification Process

The verification process is a recursive, automated process in which inference rules and the *Update* function are recursively applied until none of the rules can proceed any further and the *Observer* sets cannot be modified. Note that the processes described in the next sections are incomplete as they only highlight the most relevant steps.

3 SET's Dual Signature

To illustrate how the formalism described in previous sections works, and how it can be used to formally reason about privacy properties, we introduce the dual signature mechanism used in the SET protocol [8] for e-commerce transactions. The goal of the dual signature mechanism is to implement a *minimum information disclosure*[1] procedure. As previously mentioned, although various attempts have been made to formally reason about SET [14], the hash inference rule applied was not correct [15].

Notation:

$\{U, M, B\}$	Denotes the set of participant entities: User, Merchant and Bank.
K_U^{-1}	Denotes user U private key.
K_{UM}	Denotes a secret key between the user U and the merchant M.
K_{UB}	Denotes a secret key between the user U and the bank B.
h	Cryptographic hash function.

[1] All systems should comply to the policy of minimum information disclosure as stated in [13], Article 7.

- **Step 1:** User U wants to proceed with an electronic purchase. User U constructs an *Order Information* token denoted as OI describing the concept of the purchase, quantity, price, etc. Additionally, U also generates a *Payment Information* token denoted by PI, including the card details and the amount to be paid. Item OI is destined to the Merchant M and item PI is destined to the Bank B. Both items must be linked as the user must be able to prove that the payment is intended for the corresponding order, however, OI must be kept secret from B as well as PI must be kept secret from the merchant M.
- **Step 2:** User U sends Merchant M the following message:

$$m_1 = \{OI, h(PI), \{h(h(OI)\|h(PI))\}_{K_U^{-1}}\}_{K_{UM}}$$

  In receiving this item, entity M can verify the user's signature over item OI and can also establish the link between the OI and PI without getting to know the content of PI.
- **Step 3:** User U sends the Bank B the following message:

$$m_2 = \{h(OI), PI, \{h(h(OI)\|h(PI))\}_{K_U^{-1}}\}_{K_{UB}}$$

  In receiving this item, entity B can verify the user's signature over item PI and can also establish the link between the OI and PI without getting to know the content of OI.

3.1 SET's Dual Signature Formal Verification

As the formal reasoning of steps 2 and 3 are equivalent, we will only proceed with the formal verification of steps 1 and 2.

Global and local definitions:

- W, the world.
- $P = \{U, B, M\}$, is the set of participant entities.
- $S = \{K_{UM}, K_{UB}, K_U, K_U^{-1}\}$, is the set of secrets. Where $Observers(K_{UM}) = \{U, M\}$; $Observers(K_{UB}) = \{U, B\}$; $Observers(K_U) = \{W\}$; $Observers(K_U^{-1}) = \{U\}$;
- $R = \{R1, R2, R3, R4\}$ is the set of inference rules.
- $\forall i, j \in \{1, 2, 3\}, TRUST_{ij} = 0$
- $FV = \{x_1, x_2, x_3\}$
- $BV = \{h, OI, PI\}$. Where h represents a cryptographic hash function, OI the order information and PI the payment information.

Principal U's local sets:

$POSS(U) = \{K_{UM}, K_{UB}, K_U^{-1}, K_U, h\}$
$BEL(U) = \{\sharp K_{UM}, \sharp K_{UB}\}$
$BL(U) = \langle bvr_1^U, bvr_2^U \rangle$
Where:
$bvr_1^U = \{ \quad - \quad ; generate\_secret(OI), generate\_secret(PI)\}$
$bvr_2^U = \{Send(M, m_1); \quad - \quad \}$

Principal M's local sets:

$POSS(M) = \{K_U, K_{UM}, h\}$
$BEL(M) = \quad \{\sharp K_{UM}, \sharp K_U, \{K_U^{-1}, h\} \in POSS(U)\}$
$BL(M) = \quad \langle bvr_1^M \rangle$
Where:
$bvr_1^M = \quad \{Receive(U, m);$
$Decrypt(m, K_{UM})$ into x_1, x_2, x_3
$Apply(h, x_1, 1),$
$Concat(h(x_1), x_2),$
$Decrypt - signature(x_3, K_U)\}$

Note that entity M would believe that $\{K_U^{-1} \in POSS(U)\}$ and $\sharp K_U$ once the corresponding public key certificate has been properly verified.

Expected security and privacy goals:

G1: $\{OI\} \in POSS(M)$. M possesses item OI

G2: Observers(OI)={U,M} . Secret OI must only be known by U and M

G3: $\{OI \in POSS(U)\} \in BEL(M)$. M must believe that U possesses OI

G4: $\{\{h(OI) \| h(PI)\} \in POSS(U)\} \in BEL(M)$. M must believe that U possesses $\{h(OI) \| h(PI)\}$

G5: Observers(PI)={U,B} . Secret PI must only be known by U and B

Formal proof: Only those steps in relation to the satisfaction of goals are shown.

Step 1:
$bvr_1^U = \{ \quad - \quad ; generate\_secret(OI), generate\_secret(PI)\}$
$S := S \cup \{OI, PI\}$
$Observers(OI) = \{U\}$
$Observers(PI) = \{U\}$
$bvr_2^U = \{Send(M, m_1); \quad - \quad \}$

Step 2:
$bvr_1^M =$
$\{$**Receive**(U, m_1); $-$ $\}$
$POSS(M) := POSS(M) \cup \{m_1\}$
Decrypt$(\{OI, h(PI), \{h(h(OI)\|h(PI))\}_{K_U^{-1}}\}\}_{K_{UM}}, K_{UM}\})$
$POSS(M) := POSS(M) \cup \{x_1 = OI, x_2 = h(PI), x_3 = \{h(h(OI)\|h(PI))\}_{K_U^{-1}}\}\}$ **(G1)**
Message Meaning Rule R2:
$BEL(M) := BEL(M) \cup \{x_1 = OI, x_2 = h(PI), x_3 = \{h(h(OI)\|h(PI))\}_{K_U^{-1}}\} \in POSS(U)\}$ **(G3)**
Function Rule R1 for hash h and *concat* functions:
$BEL(M) := BEL(M) \cup \{h(h(OI)\|h(PI)) \in POSS(U)\}$
Apply$(h, x_1, 1)$
$POSS(M) := POSS(M) \cup \{h(OI)\}$
Concat$(h(x_1), x_2)$
$POSS(M) := POSS(M) \cup \{h(OI)\|h(PI)\}$
Decrypt-signature$(x_3 = \{h(h(OI)\|h(PI))\}_{K_U^{-1}}\}, K_U)$
$POSS(M) := POSS(M) \cup \{h(h(OI)\|h(PI))\}$,
$BEL(M) := BEL(M) \cup \{\{h(h(OI)\|h(PI))\} \in POSS(U)\}$
Signature Verification rule R3:
$BEL(M) := BEL(M) \cup \{\{h(OI)\|h(PI)\} \in POSS(U)\}$**(G4)**

The ***Update*** function, recursively applied until the end of
the verification process, renders the following results over the secrets OI and PI:
$Observers(OI) = \{U, M\}$ **(G2)**
$Observers(PI) = \{U, B\}$ **(G5)**

□

4 Private Speed Ticketing Electronic System

Vehicular ad–hoc networks (VANETs) allow nodes, including vehicles and road side units, to communicate with each other. These networks are used with the purpose of optimizing traffic (reducing congestion and accidents), obtaining real–time road information (the vehicles can serve as information collectors) and giving authorities the possibility to supervise vehicles electronically (speed control, vehicular permits, etc). In VANETs, vehicles are equipped with tamper proof devices known as On–Board Units (OBUs) used to communicate with other vehicles and with other Road–Side Units (RSUs) part of the road infrastructure. In VANETs, vehicles and drivers ought to maintain their identities private, at the same time as they must keep accountable for their actions and, the information processed in the VANET must be totally reliable. In this section we sketch a speed ticketing electronic system which allows the transportation authorities to enforce speed limits over a stretch of road, at the same time as the identity of the non–offenders vehicles is kept private[2].

[2] The system here described is being enhanced and developed further. For the subject of the scope of this article we only detail the most basic version.

m_1. $RSU_1 \rightarrow World$:	$\{request\_id\}$
m_2. $OBU \rightarrow RSU_1$:	$\{ECN\}_{K_{TS}}$
m_3. $RSU_1 \rightarrow TS$:	$\{sp\_evidence_1\}_{K^{-1}_{RSU_1}}$
m_4. $RSU_2 \rightarrow World$:	$\{request\_id\}$
m_5. $OBU \rightarrow RSU_2$:	$\{ECN\}_{K_{TS}}$
m_6. $RSU_2 \rightarrow TS$:	$\{sp\_evidence_2\}_{K^{-1}_{RSU_2}}$
m_7. $TS \rightarrow TA$:	$\{sp\_ticket\}$
	Where:
	$sp\_evidence_1 = \{\{ECN\}_{K_{TS}} \parallel t_1\}$
	$sp\_evidence_2 = \{\{ECN\}_{K_{TS}} \parallel t_2\}$
	$sp\_ticket = \{TS \parallel ECN \parallel t_1 \parallel t_2\}_{K_{TA}}$

Table 1. Speed Ticketing Electronic Protocol monitoring a stretch of road between road–side units RSU_1 and RSU_2.

The system consists of :

- A Ticketing Authority (TA) responsible for managing unique vehicle identifiers, as for example the Electronic Chassis Number (ECN).
- On–Board Units (OBUs). An OBU is a tamper proof device included in each vehicle on the road. The ECN of a vehicle is inserted in the vehicle's OBU such that when a vehicle receives a $\{request\_id\}$ token, enquiring about its identity, the OBU responds with the ECN encrypted.
- Road–Side Units (RSUs) which periodically broadcast messages over *Dedicated Short Range Channels* requesting the identity of the passing by vehicles. The RSUs compose *sp_evidence* tokens from the responses received from the passing vehicles by attaching the time to the vehicle identification token. This new token called *sp_evidence* is sent to a Ticketing Server (TS).
- The Ticketing Server (TS) receives *sp_evidence* tokens from two different RSUs and issue *sp_ticket* tickets for those vehicles which took less that the permitted time to cover a distance between the locations of the two RSUs. The TS is also responsible for informing the Ticketing Authority (TA) about which vehicles have over passed the speed limit over the stretch of road being monitored.

(Table 1 details the messages involved in the protocol.)

Although the messages exchanged between principals are detailed in Table 1, we will use the extension of Rubin's logic to formally represent each participant entity behavior list and the privacy related goals to be satisfied at the end of the message exchange.

4.1 Speed Ticketing Formal Verification

- **New action** Generate–ticket(ECN, t_1, t_2, t_l)
  Condition: $\{\{ECN\}_K, t_1, t_2, t_l\} \subset POSS(TS)$ such that: $(t_2 - t_1) < t_l$

Result: $POSS(TS) := POSS(TS) \cup \{TS \parallel ECN \parallel t_1 \parallel t_2\}_{K_{TA}}$

Description: This action is used to issue a *sp_ticket* when the difference between t_2 and t_1 is below a certain legal limit.

– **Global definitions:**

- W, the world.
- $P = \{TA, TS, RSU_1, RSU_2, OBU\}$, is the set of participant entities.
- $S = \{K_{TA}, K_{TS}, K_{RSU_1}, K_{RSU_2}, ECN, K_{TA}^{-1}, K_{TS}^{-1}, K_{RSU_1}^{-1}, K_{RSU_2}^{-1}\}$, is the set of secrets. Where $Observers(K_{TA}) = \{W\}$; $Observers(K_{TS}) = \{W\}$; $Observers(K_{RSU_1}) = \{W\}$; $Observers(K_{RSU_2}) = \{W\}$; $Observers(ECN) = \{OBU\}$; $Observers(K_{TA}^{-1}) = \{TA\}$; $Observers(K_{TS}^{-1}) = \{TS\}$; $Observers(K_{RSU_1}^{-1}) = \{RSU_1\}$; $Observers(K_{RSU_2}^{-1}) = \{RSU_2\}$;
- $R = \{R1, \ldots, R4\}$ is the set of inference rules.
- $\forall i, j \in \{1, 2, 3, 4, 5\}, TRUST_{ij} = 1$. Note that, unlike the previous example, all entities trust each other in following the protocol instructions. This is a reasonable assumption as all principals belong for the same authority, no tampering is possible on the OBU and the OBU trusts the authority responsible for the ticketing system.
- $BV = \{t_1, t_2, t_{limit}, sp\_evidence_1, sp\_evidence_2, sp\_ticket\}$. Where t_{limit} indicates the minimum legal time permitted in driving between units RSU_1 and RSU_2.

– **Principal RSU_1:**

$POSS(RSU_1) = \{K_{RSU_1}, K_{RSU_1}^{-1}\}$
$BEL(RSU_1) = \{\}$
$BL(RSU_1) = \langle bvr_1^{RSU_1}, bvr_2^{RSU_1}, bvr_3^{RSU_1} \rangle$
Where:
$bvr_1^{RSU_1} = \{Send(W, m_1);\ \text{-}\ \}$
$bvr_2^{RSU_1} = \{Receive(OBU, m_2);\ Encrypt(Concat(m_2, t_1), K_{RSU_1}^{-1})\}$
$bvr_3^{RSU_1} = \{Send(TS, m_3);\ \text{-}\ \}$

– **Principal RSU_2:**

$POSS(RSU_2) = \{K_{RSU_2}, K_{RSU_2}^{-1}\}$
$BEL(RSU_2) = \{\}$
$BL(RSU_2) = \langle bvr_1^{RSU_2}, bvr_2^{RSU_2}, bvr_3^{RSU_2} \rangle$
Where:
$bvr_1^{RSU_2} = \{Send(W, m_4);\ \text{-}\ \}$
$bvr_2^{RSU_2} = \{Receive(OBU, m_5);\ Encrypt(Concat(m_5, t_2), K_{RSU_2}^{-1})\}$
$bvr_3^{RSU_2} = \{Send(TS, m_6);\ \text{-}\ \}$

– **Principal OBU:**

$POSS(OBU) = \{ECN, K_{TS}\}$
$BEL(OBU) = \{\sharp K_{TS}, K_{TS}^{-1} \in POSS(TS)\}$. Note that these believes indicate that the vehicle's OBU has got a valid public key certificate from TS.
$BL(OBU) = \langle bvr_1^{OBU}, bvr_2^{OBU}, bvr_3^{OBU}, bvr_4^{OBU} \rangle$
Where:
$bvr_1^{OBU} = \{Receive(RSU_1, m_1); Encrypt(ECN, K_{TS})\}$
$bvr_2^{OBU} = \{Send(RSU_1, m_2); \quad - \quad\}$
$bvr_3^{OBU} = \{Receive(RSU_2, m_4); Encrypt(ECN, K_{TS})\}$
$bvr_4^{OBU} = \{Send(RSU_2, m_5); \quad - \quad\}$

– **Principal TS:**

$POSS(TS) = \{K_{TS}, K_{TS}^{-1}, K_{RSU_1}, K_{RSU_2}, t_{limit}\}$
$BEL(TS) = \{\sharp K_{RSU_1}, \sharp K_{RSU_2}, K_{RSU_1}^{-1} \in POSS(RSU_1), K_{RSU_2}^{-1} \in POSS(RSU_2)\}$.
$BL(TS) = \langle bvr_1^{TS}, bvr_2^{TS}, bvr_3^{TS} \rangle$
Where:
$bvr_1^{TS} = \{Receive(RSU_1, m_3); \quad - \quad\}$
$bvr_2^{TS} = \{Receive(RSU_2, m_6);$
$Decrypt - signature(m_3, K_{RSU_1}),$
$\boldsymbol{Forget}(\{sp\_evidence_1\}_{K_{RSU_1}^{-1}}),$
$Decrypt - signature(m_6, K_{RSU_2}),$
$\boldsymbol{Forget}(\{sp\_evidence_2\}_{K_{RSU_2}^{-1}}),$
$split(sp\_evidence_1),\ split(sp\_evidence_2),$
$decrypt - asymm(\{ECN\}_{K_{TS}}, K_{TS}^{-1}),$
$generate\_ticket(ECN, t_1, t_2, t_{limit}),$
$\boldsymbol{Forget}(\{sp\_evidence_1\},$
$\boldsymbol{Forget}(\{sp\_evidence_2\},$
$\boldsymbol{Forget}(\{ECN\}_{K_{TS}}),$
$\boldsymbol{Forget\_secret}(\{ECN\})\}$
$bvr_3^{TS} = \{Send(TA, m_7); \quad - \quad\}$

Note that, although the value $sp\_ticket = \{TS \parallel ECN \parallel t_1 \parallel t_2\}_{K_{TA}}$ is not forgotten, the content can only be retrieved by TA.

Expected security and privacy goals: The following list enumerates the goals in relation to vehicular identity privacy.

G1: $POSS(TS)\_Final = POSS(TS)\_Initial \cup \{sp\_ticket\}$. POSS(TS) has only increased with $\{sp\_ticket\}$.

G2: $Observers(ECN) = \{OBU\}$. Secret ECN must only be known by OBU.

Formal proof: Only those steps in relation to the satisfaction of goals are shown.

bvr_2^{TS}= {$Receive(RSU_2, m_6)$;
$POSS(TS) = POSS(TS) \cup \{sp\_evidence_1\}_{K^{-1}_{RSU_1}}\}$
$Decrypt - signature(m_3, K_{RSU_1})$,
$POSS(TS) = POSS(TS) \cup \{sp\_evidence_1\}$
Forget$(\{sp\_evidence_1\}_{K^{-1}_{RSU_1}})$,
$POSS(TS) = POSS(TS) - \{sp\_evidence_1\}_{K^{-1}_{RSU_1}}\}$
$Decrypt - signature(m_6, K_{RSU_2})$,
$POSS(TS) = POSS(TS) \cup \{sp\_evidence_2\}$
Forget$(\{sp\_evidence_2\}_{K^{-1}_{RSU_2}})$,
$POSS(TS) = POSS(TS) - \{sp\_evidence_2\}_{K^{-1}_{RSU_2}}\}$
$split(sp\_evidence_1)$,
$POSS(TS) = POSS(TS) \cup \{\{ECN\}_{K_{TS}}, t_1\}$
Note that the *Update* function renders the following result:
$Observers(ECN) = Observers(ECN) \cup \{TS\}$
$split(sp\_evidence_2)$,
$POSS(TS) = POSS(TS) \cup \{\{ECN\}_{K_{TS}}, t_2\}$
$generate\_ticket(ECN, t_1, t_2, t_{limit})$,
$POSS(TS) = POSS(TS) \cup \{TS \parallel ECN \parallel t_1 \parallel t_2\}$
Forget$(\{sp\_evidence_1\}$,
$POSS(TS) = POSS(TS) - \{sp\_evidence_1\}$
Forget$(\{sp\_evidence_2\}$, □
$POSS(TS) = POSS(TS) - \{sp\_evidence_2\}$
Forget$(\{ECN\}_{K_{TS}})$,
$POSS(TS) = POSS(TS) - \{ECN\}_{K_{TS}}$
Forget_secret$(\{ECN\})$,
$POSS(TS) = POSS(TS) - \{ECN\}$, **(G1)**
$Observers(ECN) = Observers(ECN) - \{TS\}$, **(G2)**

5 Conclusions

In this paper we have described what we believe is tractable and powerful formalism for the formal verification of privacy–enhancing cryptography protocols. The original logic introduced by Rubin offered a major contribution for the reasoning of non–monotonic protocols although it needed a few amendments and updates to it, for example, the concept of signature verification used in Rubin's logic was, in our opinion, invalid. The new extended logic L–PEP has served to formally verify the dual signature of the protocol SET and a speed ticketing electronic *non–monotonic* system. The logic has enabled us to reason about private data protection laws by which transportation authorities can only store information about offender vehicles and where tracing of non–offenders is not permitted.

5.1 Future Work

Currently, we are analyzing a privacy–enhancing authentication and access control protocol based on blind signatures and recursive hashing [16].

Additionally, a more advanced version of the speed ticketing electronic service here described, including identity based encryption, is under verification using the described formalism.

Finally, the extended logic also intends to set the basis for future automated design of multi-party privacy–enhancing cryptographic protocols by means of evolutionary computation and artificial intelligence techniques. Previous works on this area are [17, 18]. The logic just presented will allow for similar techniques to synthesize privacy related protocols.

References

1. Pfitzmann, A., Hansen, M.: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management - a consolidated proposal for terminology. Technical report (February 2008)
2. Meadows, C.: Formal methods for cryptographic protocol analysis: emerging issues and trends. IEEE journal on selected areas in communications **21**(1) (2003)
3. : Privacy and identity management for europe (prime). https://www.prime-project.eu Privacy and Identity Management for Europe (PRIME).
4. Burrows, M., Abadi, M., Needham, R.: A logic of authentication. ACM Transactions on Computer Systems **8**(1) (1990) 18–36
5. Abadi, M., Tuttle, M.: A semantics for a logic of authentication. In: Proceedings of the ACM Symposium of Principles of Distributed Computing, ACM Press (1991) 201–216
6. Moser, L.: A logic of knowledge and belief for reasoning about computer security. In: Computer Security Foundations Workshop II, 1989., Proceedings of the. (Jun 1989) 57–63
7. Rubin, A.D.: Nonmonotonic cryptographic protocols. In: Proceedings of the Computer Security Foundations Workshop. (1994) 100–116
8. : A Formal Specification of Requirements for Payment Transactions in the SET Protocol. Volume 1465/1998. Springer Berlin / Heidelberg (1998)
9. Xu, Y., Xie, X.: Analysis of electronic commerce protocols based on extended rubin logic. In: Proceedings of the 9th International Conference for Young Computer Scientists, Washington, DC, USA, IEEE Computer Society (2008) 2079–2084
10. Xu, Y., Xie, X.: Analysis of authentication protocols based on rubin logic. In: WiCOM '08. 4th International Conference on Wireless Communications, Networking and Mobile Computing. (2008) 1–5
11. Xu, Y., Xie, X.: Security analysis of routing protocol for manet based on extended rubin logic. In: ICNSC 2008. IEEE International Conference on Networking, Sensing and Control. (2008) 1326–1331
12. Teepe, W.: Reconciling information excahnge and confidentiality–a formal approach. Technical report, Rijksuniversiteit Groningen (2006) Ph.D. Thesis.
13. : Protection of individuals with regard to the processing of personal data and on the free movement of such data. http://eur-lex.europa.eu Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995.

14. Agray, N., Hoek, W.v.d., Vink, E.P.d.: On ban logics for industrial security protocols. In: CEEMAS '01: Revised Papers from the Second International Workshop of Central and Eastern Europe on Multi-Agent Systems. (2002) 29–36
15. Teepe, W.: On ban logic and hash functions or: how an unjustified inference rule causes problems. Autonomous Agents and Multi-Agent Systems **19**(1) (2009) 76–88
16. Ren, K. Wenjing Lou Kwangjo Kim Deng, R.: A novel privacy preserving authentication and access control scheme for pervasive computing environments. In: IEEE Transactions on Vehicular Technology. Volume 55. (2006) 1373–1384
17. Chen, H., Clark, J., Jacob, J.: Automatic design of security protocols. Computational Intelligence **20**(3) (2004) 503–516 Special Issue on Evolutionary Computing in Cryptography and Security.
18. Alcaide, A., Estévez-Tapiador, J., Hernandez Castro, J., Ribagorda, A.: Nature-inspired synthesis of rational protocols. In: Proceedings of the 10th International Conference On Parallel Problem Solving from Nature (PPSN 2008). (2008) LNCS Vol. 5199, pp. 981–990.