Analysis of some Famous Cryptographic Protocols Using the Interpretation-Function-based Method

Hanane Houmani Mohamed Mejri
LSFM Research Group,
Computer Science Department,
Laval University, Quebec, Canada
Hanane.Houmani@ift.ulaval.ca Mohamed.Mejri@ift.ulaval.ca

Abstract

This paper shows the efficiency and the flexibility of the interpretation function-based method [8, 9, 11] through the analysis of some famous cryptographic protocols. In fact, by using the notion of interpretation function, this method gives general and sufficient conditions allowing to guarantee the secrecy property of cryptographic protocols. This result holds under a large class of equational theories, that makes the approach flexible and general. Moreover, the approach does not only help to prove the secrecy property, but also helps a lot to discover protocols weaknesses and to correct them. These advantages are discussed and illustrated by cases study of some protocols that are analyzed in different equational theories.

1. Introduction

Cryptographic protocols are used in our daily life to make secure our communications and transactions as mailing, banking, e-commerce, ect. Hence, the guarantee of the security of cryptographic protocols is paramount. However, the verification of the security of cryptographic protocols is undecidable in general (see [5, 6]). Therefore, researchers have proposed a large variety of methods and tools that can help to find attacks or to prove the security for some classes of these protocols. A general survey of the most used approaches related to the verification of cryptographic protocols could be found in [2, 13, 16].

Due to the complexity of the problem, almost all existing approaches try to simplify it by making some restrictions or assumptions on protocols (like the perfect encryption assumption). In other words, they can deal with only some classes of cryptographic protocols under some restrictions and assumptions like a specific intruder model. For example, in [14], L. Paulson has proved that the Bull protocol preserves secrecy using an intruder model that does not take into account any algebraic property of cryptographic primitives. However, he proved that attacks are possible on this protocol if some algebraic properties of $\oplus$ or of exponentiation are considered.

In [8, 9, 11], we have gathered assumptions and restrictions that can be made on cryptographic protocols on what we called a *context of verification*. More specifically, a context of verification is basically the specification of messages algebra, the capacities of the intruder, and the equational theory. This representation allowed us to give sufficient conditions that do not depend on a specific context of verification (specific class of messages or a specific capacities of the intruder) and that guarantee the secrecy property of any protocol that respect them. These sufficient conditions state

intuitively that agents of a given protocol should not decrease the security level of messages when they send them over the network. To formalize these conditions, we have introduced a special functions called "interpretation function". An interpretation function gives a security level of a component of a message. Hence to verify if agents do not decrease security level of a message, we compare the security level of each message in sent messages with his security level in the received messages. However, the interpretation function should give the correct security level of a message and could not be misled by an intruder. Such interpretation functions are called "safe" interpretation functions.

These concepts give to the interpretation functions-based method a certain flexibility and make it more general. Indeed, the sufficient conditions of this approach do not depend on a context of verification. Therefore, the approach is powerful enough to be applied to analyze protocols in a large variety of contexts of verification (a large variety of equational theories). To illustrate this, we analyse and discuss in this paper, some famous cryptographic protocols in different context of verification.

The remainder of this paper is organized as follows. Section 2 presents the definition of a context of verification that deals with equational theories. Also, it gives the definition of some basic words used within this paper. Section 3 gives a formal definition for the secrecy property. Section 4 introduces the proposed conditions and proves that they are sufficient to ensure the secrecy property of cryptographic protocols. Section 5 presents a guideline to define "safe" interpretation functions and gives some some standard and practical examples of such functions. Section 6 put in practice the approach with some cases in point. Finally, section 7 provides some concluding remarks.

2. Basic definitions

Basically, this section extends the definition of a context of verification defined in [8, 9, 11], in order to deal with equational theories. Also, it gives the definition of a set of messages and the definition of the intruder capacities.

Context of verification: Parameters like the structure of messages exchanged during the protocol, the intruder capacities or the algebraic properties of cryptographic primitives, could affect the class of protocols that could be analyzed by an approach. We found therefore interesting to gather them in what we called a *context of verification*. A context of verification can have the following form $\mathcal{C} = \langle \mathcal{N},\ \Sigma,\ \mathcal{E},\ K,\ \mathcal{L}^{\sqsupseteq},\ \ulcorner \cdot \urcorner \rangle$, where:

- **The names** $\mathcal{N}$ is the set of names (nounce, keys, etc). For instance, let $\mathcal{N}_0$ be the set of names given by the the following BNF grammar:

$$\begin{array}{rcll} n & ::= & A & \text{(Principal Identifier)} \\ & | & N_a & \text{(Nonce)} \\ & | & k_{ab} & \text{(Shared key)} \end{array}$$

- **The signature** Σ is a signature and it contains all function symbols (encryption and pair symbol for example). For instance, let Σ_0 be the signature defined as follows:

$$\Sigma_0 = \{enc, dec, pair, fst, snd\}$$

As usual we write $\langle x, y \rangle$ instead of writing $pair(x, y)$.

- **The equational theory** $\mathcal{E}$ is the equational theory that represents the algebraic properties of function symbols (commutativity of the pair symbol for example). For instance, Let E_0 be the equational theory that contains the following equations:

$$\begin{aligned} fst(\langle x, y\rangle) &= x \\ snd(\langle x, y\rangle) &= y \\ dec(enc(x, y), y) &= x \end{aligned}$$

- **The intruder knowledge** K is the set of initial knowledge of the intruder. For instance, let K_0 be the set of knowledge of intruder, and it is the same initial knowledge as any other honest agent, *i.e.*he shares with other agents a key K_{ia}, has a public key k_i, has a private key K_i^{-1}, and he can generate fresh values.
- **The lattice of security** $\mathcal{L}^{\sqsupseteq}$ is a lattice of security that contains security levels (types) ($\{public,\ secret, any\}$ is a lattice of security for example). For instance, let $\mathcal{L}_0^{\subseteq}$ be the security lattice that is the power of the set of agents identities i.e $2^{\mathcal{I}}$.
- **The typed environment** $\ulcorner\cdot\urcorner$ is the function that returns the real security level of a message. For instance, let $\ulcorner\cdot\urcorner_0$ be this function and this function gives the set of identities of agent that can know a message, then ($\ulcorner k_{ab}\urcorner_0 = \{A, B\}$).

Messages: Given a context of verification $\mathcal{C}$, a set of messages $\mathcal{M}$ can be defined (this definition is inspired from [1].) by the following BNF grammar:

$$\begin{array}{rcll} m & ::= & N & \text{(Name)} \\ & | & X & \text{(Variable)} \\ & | & f(m_1, \ldots, m_n) & \text{(Function application)} \end{array}$$

Notice that the set of messages involved in the verification context will be denoted, in that follows, by $\mathcal{M}$, and the set $\mathcal{A}(M)$ denotes the set of atomic components (nonces, keys and principal identifiers) in M.

Deduction: Given a context of verification $\mathcal{C} = \langle \mathcal{N},\ \Sigma,\ \mathcal{E},\ K,\ \mathcal{L}^{\sqsupseteq},\ \ulcorner\cdot\urcorner\rangle$ and a set of message M that represents the information available to an intruder, the message m can be deduced from M in the context $\mathcal{C}$ and we write $M \models_{\mathcal{C}} m$ if m can be obtained by using these rules:

$$\begin{array}{ll} (\mathsf{Init}) & \dfrac{\square}{M \models_{\mathcal{C}} m}[m \in M \cup K_I] \\[2ex] (\mathsf{Eq}) & \dfrac{M \models_{\mathcal{C}} m_1 \quad m_1 =_{\mathcal{C}} m_2}{M \models_{\mathcal{C}} m_2} \\[2ex] (\mathsf{Op}) & \dfrac{M \models_{\mathcal{C}} m_1,\ \ldots,\ M \models_{\mathcal{C}} m_n}{M \models_{\mathcal{C}} f(m_1, \ldots, m_n)}[f \in \Sigma] \end{array}$$

Table 1. Generic capacities of intruder

Given a context of verification $\mathcal{C} = \langle \mathcal{N},\ \Sigma,\ \mathcal{E},\ K,\ \mathcal{L}^{\sqsupseteq},\ \ulcorner\cdot\urcorner\rangle$, we write $m_1 =_{\mathcal{C}} m_2$ if and only if $m_1 =_{\mathcal{E}} m_2$, and $m_1 =_{\mathcal{E}} m_2$ if the message m_1 and m_2 are equal under the equational theory $\mathcal{E}$.

Protocol: Basically, a protocol is specified by a sequence of communication steps given in the standard notation. More precisely a protocol p has to respect the following BNF grammar:

$$p \quad ::= \langle i : \ A \rightarrow B : m \rangle \mid p.p$$

The statement $\langle i : \ A \rightarrow B : m \rangle$ denotes the transmission of a message m from the principal A to the principal B in the step i. Let p_0 be a variant of the Woo and Lam [17] authentication protocol. This variant, given by Table 2, aims to distribute a new key that will be shared between two agents A and B.

$$\begin{array}{ll} p_0 = & \langle 1, A \rightarrow B : A \rangle. \\ & \langle 2, B \rightarrow A : N_b \rangle. \\ & \langle 3, A \rightarrow B : \{N_b, k_{ab}\}_{k_{as}} \rangle. \\ & \langle 4, B \rightarrow S : \{A, \{N_b, k_{ab}\}_{k_{as}}\}_{k_{bs}} \rangle. \\ & \langle 5, S \rightarrow B : \{N_b, k_{ab}\}_{K_{bs}} \rangle \end{array}$$

Table 2. Woo and Lam modified protocol.

We will denote in the sequel $\mathcal{R}_G(p)$ the formal specification of protocol p. It can be a specification in the pi-calculs or in strands spaces or any other specification. $[\![p]\!]$ will denote the executions of a protocol p.

3. Secrecy property

Who does not care about his own privacy and confidentiality. The confidentiality property, called also the secrecy property, is among the most important security properties of cryptographic protocols. That is why, in this paper we care about this property.

Intuitively, the formal definition of the secrecy property given hereafter states that the intruder cannot learn from any executions of a protocol more than what he is eligible to know. As saying in the introduction, the analysis depends always on the considered context of verification. That is why the definition of secrecy presented speaks about the correctness according to a context of verification instead of speaking of correctness only.

Definition 3.1 (Secrecy Property) *Let p be a protocol and $\mathcal{C} = \langle \mathcal{N}, \Sigma, \mathcal{E}, K, \mathcal{L}^{\sqsupseteq}, \ulcorner \cdot \urcorner \rangle$ a verification context. The protocol p is $\mathcal{C}$-correct with respect the secrecy property, if:*

$$\forall \alpha \in \mathcal{A}(\mathcal{M}) \cdot \quad [\![p]\!] \models_{\mathcal{C}} \alpha \quad \Rightarrow \quad \ulcorner K \urcorner \sqsupseteq \ulcorner \alpha \urcorner$$

where $\sqsupseteq$ is an order under the set of security levels and the notation $\ulcorner K_I \urcorner \sqsupseteq \ulcorner \alpha \urcorner$ is an abbreviation of: $\exists \beta \in K \cdot \ulcorner \beta \urcorner \sqsupseteq \ulcorner \alpha \urcorner$. Notice that this abbreviation will be used throughout the rest of this paper.

4. Main result

Now, it is time to give the sufficient conditions allowing to guarantee the secrecy property of a cryptographic protocol. Informally, these conditions state that honest agents should never decrease the security level of any atomic message. However, these conditions depend on having a "good"

way to calculate the security level of a message. By a "good" way, we mean basically that we should have a way that can never be misled by the intruder. The way to calculate a security level of message will be called an interpretation function and a "good" one will be called a "safe" interpretation function. Formally:

Definition 4.1 (Safe Interpretation Function)
Let $\mathcal{C} = \langle \mathcal{N},\ \Sigma,\ \mathcal{E},\ K,\ \mathcal{L}^{\sqsupseteq},\ \ulcorner\cdot\urcorner \rangle$ be a context of verification. A $\mathcal{C}$-interpretation function (a function from $\mathcal{A}(\mathcal{M}) \times \mathcal{M}$ to $\mathcal{L}$ F, is called $\mathcal{C}$-Safe if the following conditions hold:

1. F *is well formed, i.e:*
   $\mathsf{F}(\alpha, \{\alpha\}) = \bot$ *and* $\mathsf{F}(\alpha, M_1 \cup M_2) = \mathsf{F}(\alpha, M_1) \sqcap \mathsf{F}(\alpha, M_2)$ *and* $\mathsf{F}(\alpha, M) = \top$ *where* $\alpha \notin \mathcal{A}(M)$
2. F *is $\mathcal{C}$-full-invariant by substitution, i.e: for all M_1 and M_2 two set of messages in $\mathcal{M}$ such that $\forall \alpha \in \mathcal{A}(M_1) \cdot \ \mathsf{F}(\alpha, M_1) \sqsupseteq \mathsf{F}(\alpha, M_2)$ we have:*

$$\forall \sigma \in \Gamma, \forall \alpha \in \mathcal{A}(M_1) \cdot \ \mathsf{F}(\alpha, M_1\sigma) \sqsupseteq \mathsf{F}(\alpha, M_2\sigma)\sigma$$

   where Γ is the set of possible substitution form $\mathcal{X}$ to close messages in $\mathcal{M}$.
3. F *is $\mathcal{C}$-full-invariant by intruder, i.e:*
   $\forall M \subseteq \mathcal{M}$, $\forall \alpha \in \mathcal{A}(M)$ such that $\mathsf{F}(\alpha, M) \sqsupseteq \ulcorner\alpha\urcorner$ and $\forall m \in \mathcal{M}$ such that $M \models_{\mathcal{C}} m$ we have:

$$\forall \alpha \in \mathcal{A}(m) \cdot \ (\mathsf{F}(\alpha, m) \sqsupseteq \mathsf{F}(\alpha, M) \vee (\ulcorner K_I^{-1}\urcorner \sqsupseteq \ulcorner\alpha\urcorner)$$

As an example of a safe interpretation function, let $\mathcal{C}_0 = \langle \mathcal{N}_0,\ \Sigma_0,\ \mathcal{E}_0,\ K_0,\ \mathcal{L}_0^{\sqsupseteq},\ \ulcorner\cdot\urcorner_0 \rangle$ be the context of verification defined in section 2 and F_0 is the function that attributes a security level of a message according to what we have as neighbors to this message and what keys encrypt it directly. For instance, $\mathsf{F}_0(k_{ab}, \{A, \{B, N_b, k_{ab}\}_{k_{as}}\}_{k_{bs}}) = \{B\} \cup \ulcorner k_{as}^{-1}\urcorner_0 = \{A, S, B\}$. This function is called the DEKAN function is $\mathcal{C}_0$-safe (see section 5).

The sufficient condition that states that agents of a protocols should not decrease the security level of a messages according to a context of verification, can be formalized as follows:

Definition 4.2 (Increasing Protocol)
Let $\mathcal{C} = \langle \mathcal{N},\ \Sigma,\ \mathcal{E},\ K,\ \mathcal{L}^{\sqsupseteq},\ \ulcorner\cdot\urcorner \rangle$ be a verification context, F a $\mathcal{C}$-interpretation function and p a protocol. The protocol p is said to be F-increasing if:

$$\forall r \in \mathcal{R}_G(p), \forall \alpha \in \mathcal{A}(r^+) \cdot \ \mathsf{F}(\alpha, r^+) \ \sqsupseteq \ \ulcorner\alpha\urcorner \sqcap \mathsf{F}(\alpha, r^-)$$

where r^+ is a set containing the messages sent during the last step of r and r^- contains the set of messages received by the honest agent in r.

Now, the main theorem could be formalized as follows:

Theorem 4.3 *Let p be a protocol, $\mathcal{C}$ a verification context and F a $\mathcal{C}$-interpretation function . If F is $\mathcal{C}$-safe and p is F-increasing, then p is $\mathcal{C}$-correct with respect to the secrecy property.*

Proof:
The detailed proof could be found in [8, 9, 10, 11].

□

5. Guideline to Define a Safe Interpretation Function

According to theorem 4.3, the first step of the verification of secrecy property is to find a *safe interpretation function* and this is the delicate part of the approach. In fact, a safe interpretation function is basically a function that is full-invariant by substitutions and full-invariant by intruder. These properties are not easy to check or to respect. For that reason, this section provides a guideline allowing to easily construct a safe interpretation function. This guideline concerns only *convergent theories* [1, 7].

Basically, we show hereafter that any interpretation function that has a specific form is safe. In particular, we focus on interpretation functions having the following form:

$$\mathsf{F}(\alpha, M) = \mathsf{I} \circ \mathsf{S}(\alpha,\ M)$$

where S is a function that selects from M some atomic components having some links with α and where I is a function that interprets what S returns as a security type.

Intuitively, if the selection function S is full-invariant by intruder and by substitutions, and the function I is an homomorphism, then we could have more chance that the function $\mathsf{I} \circ \mathsf{S}$ is full-invariant by intruder and substitutions (hence safe). This fact will be studied and proved in the following sections.

5.1. Restrictions on the Selection Function S

If we consider a message, which is a term in the message algebra, as a tree and we attach to each arc of this tree an integer value (reflecting costs or distance between nodes), then it will be easy to define S that selects some components that are at some distance from a given component. To formalize the notion of a distance, we introduce what we call a Trees Assignment.

Definition 5.1 (Tree Assignment) *We introduce a function $\mathcal{T}$ that takes a message and returns a tree in which nodes are function symbols and leafs are atomic messages and in which each arc has a cost (integer value). This function will be called a* Tree Assignment.

Let be the message $\langle B, \{N_b, k_{ab}\}_{k_{as}}\rangle$. An example of $\mathcal{T}(m)$ is given by Fig. 1.

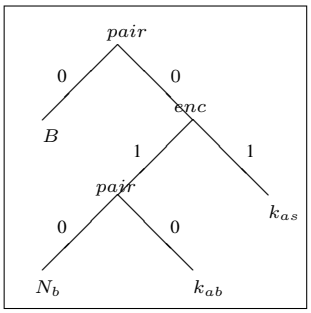

Figure 1. Message $\mathcal{T}(\langle B, \{N_b, k_{ab}\}_{k_{as}}\rangle)$

Where enc is the encryption operator, $pair$ is the concatenation operator. In the rest of this paper we will denote the cost between two nodes (n_1, n_2) in a message tree $\mathcal{T}(m)$ by $\mathcal{T}_m(n_1, n_2)$.

As in the previous example, a *Tree Assignment* function may attach to a message his corresponding tree with some costs in arcs. Such functions will be called *Corresponding Tree Assignment* functions. A *Corresponding Tree Assignment* functions will be denoted by $\mathcal{G},\ \mathcal{G}_1,\ \mathcal{G}_2,\ \ldots$.

In the rest of this paper, $d_{\mathcal{T}(m)}(\alpha,\, \beta)$ (or simply $d_m(\alpha,\, \beta)$ if $\mathcal{T}(m)$ is clear from the context) denotes the smallest distance between α and β in $\mathcal{T}(m)$. This notion can be easily extended to a set of messages M as follows: $d_M(\alpha,\, \beta) = \underset{m\in M}{Min}(d_m(\alpha,\, \beta))$. For instance, by Fig. 1, we have $m = \langle B, \{N_b, k_{ab}\}_{k_{as}}\rangle$, $d_m(N_b,\, k_{as}) = d_m(k_{ab},\, k_{as}) = 2$, $d_m(B,\, N_b) = d_m(B,\, k_{ab}) = 1$ and $d_m(k_{ab},\, N_b) = 0$.

Using the notion of Tree Assignment and notion of distance, the n-selection function $\mathsf{S}^n_{\mathcal{T}}$ could be defined as follows:

$$\mathsf{S}^n_{\mathcal{T}}(\alpha,\, M) \;=\; \begin{cases} \mathcal{A} & \text{if } \alpha \in \mathsf{Clear}(M) \\ \{\beta \in \mathcal{A}(M) \mid d_{\mathcal{T}(M)}(\alpha,\, \beta) \;=\; n\} & \text{else} \end{cases}$$

The function Clear is defined as follows:

$$\mathsf{Clear}(m) = \{\alpha \mid \forall f \in \Sigma \cdot \; f \text{ is a predecessor of } \alpha \text{ in } \mathcal{T}(m_{\downarrow}) \;\Rightarrow\; f : \overline{e}\}$$

Where $f : \overline{e}$ means that the symbol f is not an operator of encryption, and $f : e$ means that the symbol f is an operator of encryption.

We have said that we will attempt to have a selection function that is full Invariant by Intruder and by substitutions. First, let's study the full invariance by Intruder property. The full invariance property means intuitively that we should select elements that could not be misled by an intruder. So, to guaranty that property, we have to select always the encryption keys because we need to ensure that every message is well-encrypted, and so it can not be sent to who is not eligible to know it. Also, we need that algebraic properties do not add information to a message. Indeed, the algebraic properties of function symbols are part of intruder capacities. Also, constructing new messages with function symbols is also part of intruder capacities. Hence, a function symbols should not add more information. These three conditions that are sufficient to the full invariance by intruder of a selection function, are formalized follows:

Definition 5.2 (Safe Costs Assignment (SCA)) *Let $\mathcal{C}$ be a context of verification and $\mathcal{T}$ be a tree assignment function. We say that $\mathcal{T}$ is a $\mathcal{C}$-SCA if:*

1. *At least direct encryption keys must be selected,* i.e. *for every k that encrypt directly α in a message m we have:*
   - $\mathcal{T}_m(\alpha, k) = n$
2. *Function symbols do not add more information:*
   - $\forall f \in \Sigma, \forall m_1, \ldots, m_n \in \mathcal{M} \cdot \; \mathcal{T}(f(m_1, \ldots, m_n)) \preceq \mathcal{T}(m_1, \ldots, m_n)$
     where $\mathcal{T}(m) \preceq \mathcal{T}(m')$ holds if and only if for any $n \in \mathbb{N}$ and for any $\alpha, \beta \not\in \mathsf{Clear}(m, m')$, we have $\mathcal{T}_m(\alpha, \beta) = \mathcal{T}_{m'}(\alpha, \beta)$
3. *Messages that are equivalent under the equational theory must gives the same information:*
   - $m_1 =_{\mathcal{C}} m_2 \;\Rightarrow\; \mathcal{T}(m_1) = \mathcal{T}(m_2)$

Now, let's study the full-invariance by substitutions. The definition of $\mathsf{S}^n_{\mathcal{T}}$ is extended to $\mathsf{S}^{\overrightarrow{n}}_{\mathcal{T}}$, called the n-polynomial selection function, as follows:

$$\mathsf{S}^{\overrightarrow{n}}_{\mathcal{T}}(\alpha,\, M) \;= \sum_{i=0}^{n-1} a_i x^i$$

where $a_i = \mathsf{S}^i_{\mathcal{T}}(\alpha,\, M)$. Coefficients a_i are in the set $2^{\mathcal{A}}$ and this set with the standard inclusion and intersection operators represent a ringoid. Let extend the order $\subseteq$ to $\subseteq^n$ is as follows:

$$\sum_{i=0}^{n-1} a_i x^i \subseteq^n \sum_{i=0}^{n-1} b_i x^i \;\Leftrightarrow\; \forall i \in \{1,\ldots,n\} \cdot\; a_i \subseteq b_i$$

Any n-polynomial selection function $\mathsf{S}^{\overrightarrow{n}}_{\mathcal{T}}$ is full-invariant by substitutions (see proofs in [8, 9, 10, 11]).

At this step, we have shown that to have a selection function that is full invariant by intruder and substitutions, it is sufficient to define S as a polynomial selection function with using a $n-$SCA tree assignment function.

5.2. Restrictions on the Function I

Having a selection function that is full-invariant by substitutions and full-invariant by intruder is not enough to construct a safe function $\mathsf{F} = \mathsf{I} \circ \mathsf{S}$ that is also full-invariant by substitutions and full-invariant by intruder. The function I needs also to be restricted so that it can preserves the full-invariance properties of S. For instance, the selection function $\mathsf{S}^n_{\mathcal{T}}$ returns a polynomial that will be interpreted by I as a security type. These atomic components could contain variables and therefore the function I needs to carefully handle them so that the full-invariance properties of $\mathsf{S}^n_{\mathcal{T}}$ are preserved. Also, the notion of variables in security types are introduced. To that end, there is a need to extend the set of constant security types $\mathcal{L}$ to $\mathcal{L}_{\overline{\mathcal{X}}} = \mathcal{L} \cup \overline{\mathcal{X}}$, where $\overline{\mathcal{X}}$ is the set of variables that will be used to range over security types. As a consequence, the ordering relation $\sqsupseteq$ is extended to $\sqsupseteq_{\overline{\mathcal{X}}}$ so that it can deal with security variables. The relation $\sqsupseteq_{\overline{\mathcal{X}}}$ is defined as follows:

- $(\mathcal{L}_{\overline{\mathcal{X}}},\, \sqsupseteq_{\overline{\mathcal{X}}})$: where $\overline{\mathcal{X}}$ is a set of variables that range over security types, $\mathcal{L}_{\overline{\mathcal{X}}} = \mathcal{L} \cup \overline{\mathcal{X}}$ and the relation $\sqsupseteq_{\overline{\mathcal{X}}}$ is defined as follows:

$$\tau_1 \sqsupseteq_{\overline{\mathcal{X}}} \tau_2 \Leftrightarrow \forall \sigma \in \overline{\mathcal{X}} \times \mathcal{L} :\; \tau_1\sigma \sqsupseteq \tau_2\sigma$$

- $\mathsf{I}_{\overline{\mathcal{X}}}$ is the extended mapping of I, from $\mathcal{A} \cup \mathcal{X}$ to $\mathcal{L} \cup \overline{\mathcal{X}}$ as follows:

$$\mathsf{I}_{\overline{\mathcal{X}}}(\alpha) \;=\; \begin{cases} \overline{x} & \text{if } \alpha = x \\ \mathsf{I}(\alpha) & \text{if } m \in \mathcal{A} \backslash \mathcal{X} \end{cases}$$

Also, the selection function S will return a polynomial, so in order to deal with a polynomial, I should be extended. To that end, let introduce the two following useful extensions, given a security lattice $(\mathcal{L},\, \sqsupseteq)$ and the mapping I from $\mathcal{A}$ to $\mathcal{L}$.

- I^n is the extended mapping of I as follows:

$$\mathsf{I}^n(\sum_{i=0}^{n-1} a_i x^i) = \sum_{i=0}^{n-1} \mathsf{I}(a_i) x^i$$

- Consequently, the security lattice so that it can be dealt with a polynomial of basic types is needed to be extended. If $(\mathcal{L},\, \sqsupseteq)$ is the basic security lattice, then its extended version will be $(\mathcal{L}^n,\, \sqsupseteq^n)$, where $\mathcal{L}^n$ is the set of polynomials that have degree equal to $n-1$ and coefficients in $\mathcal{L}$, and $\sqsupseteq^n$ is as follows:

$$\sum_{i=0}^{n-1} \tau_i x^i \sqsupseteq^n \sum_{i=0}^{n-1} \tau'_i x^i \;\Leftrightarrow\; \forall i \in \{0,\ldots,n-1\} \cdot\; \tau_i \sqsupseteq \tau'_i$$

Having a selection function that is full-invariant by intruder is not enough to construct a safe function $\mathsf{F} = \mathsf{I} \circ \mathsf{S}$ that is also full-invariant by intruder. The function I needs to preserve the full-invariance by intruder property of S. To that end, it necessary that the mapping I be an homomorphism.

Moreover, the function I associates security levels to what the selection function selects. Hence, the function I should not misinterpret the security level of a selected messages. Indeed, let consider the message $\{\alpha\}_k$, α in this message is well protected if it is encrypted with a key that has the security level that is superior or equal to the security level of α (i.e $\ulcorner k \urcorner \sqsupseteq \ulcorner \alpha \urcorner$). Hence, we should have that $\mathsf{F}(\alpha, \{\alpha\}_k) \sqsupseteq \ulcorner \alpha \urcorner$. From that, we can deduce that $\mathsf{I}(k) \sqsupseteq \ulcorner \alpha \urcorner$. Now, let consider that $\mathsf{I}(k) \sqsupseteq \ulcorner k \urcorner$, in this case we can allow to encrypt a message α with k such that we have $\mathsf{I}(k) \sqsupseteq \ulcorner \alpha \urcorner \sqsupseteq \ulcorner k \urcorner$, and this is not correct because the real security level of k should not be superior or equal to the one of α and so in this case α is not well protected. Hence, to not misinterpret the security level of a message, the function I should associate to a message (specially keys), a security level that is at least equal to the real security level of this message. This last is the security level that associates the function $\ulcorner \cdot \urcorner$. This fact is formalized by the following definition.

Definition 5.3 ($\mathcal{C}$-Coherent) *Let $\mathcal{C} = \langle \mathcal{N},\ \Sigma,\ \mathcal{E},\ K,\ \mathcal{L}^{\sqsupseteq},\ \ulcorner \cdot \urcorner \rangle$ be a context of verification and I be a mapping, then I is $\mathcal{C}$-coherent if and only if:*

$$\forall k \in \mathsf{Keys} \cdot \ulcorner k \urcorner \sqsupseteq \mathsf{I}(k)$$

The following theorem shows the condition that is needed to be respected by I so that we get a safe interpretation function.

Theorem 5.4 *Let $\mathbb{M}$ be a context of verification and $\mathsf{S}_{\mathcal{T}}^{\overrightarrow{n}}$ an n-vector selection function. If:*

- *$\mathcal{T}$ be a SCA, and*
- *I is a $\mathcal{C}-$coherent homomorphism from $((2^{\mathcal{A}})^n, \subseteq)$ to $(\mathcal{L}_{\overrightarrow{\mathcal{X}}}^n, \sqsupseteq_{\overrightarrow{\mathcal{X}}}^n)$*

Then, $\mathsf{F} = \mathsf{I} \circ \mathsf{S}_{\mathcal{T}}^{\overrightarrow{n}}$ is $\mathcal{C}$-safe.

Proof:
Detailed proofs could be found in [8, 9, 10, 11]. □

5.3. Simple and Practical Steps to Construct F

This section summarizes results of the previous sections in order to show how to build a safe interpretation in a simple way. In fact, we have supposed that F is a composition of two functions I and S, where S allows to select elements on what depend the security level of message and I is the function that associates to an element a security level. This decomposition allowed us to prove that in order to have a selection function that is full-invariant by intruder and by substitutions, it is sufficient that the selection is a n-polynomial selection function and this selection function is SCA. Also, we have proved that I need to be an homomorphism that is coherent in order to preserve the full-invariance properties of S. Indeed, theorem 5.4 presents sufficient conditions that allow to have a safe interpretation function. Hence, a safe interpretation function F can be constructed by just giving an integer, a tree assignment and a mapping from $(\mathcal{A}, \subseteq)$ to $(\mathcal{L}, \sqsupseteq)$ and after that making some manipulations on these elements to obtain a safe interpretation function. More precisely, building a safe interpretation function can be summarized in the following steps:

1. Define a n-polynomial selection function $\mathsf{S}_{\mathcal{T}}^{\overrightarrow{n}}$. To that end, we should define the following elements:

- Choose a positive integer value for n.
- Choose $\mathcal{T}$ such that it is a n-SCA. If it is not evident, we can built one from non-n-SCA function as follows:
  - Define $\mathcal{G}$ as a corresponding tree assignment that is not necessary n-SCA, but in which direct encryption keys are always selected, the selection beyond encryption is forbidden, and the theory of the context of verification do not add more information. More precisely, the distance between any message and its direct encryption keys should be inferior to n, and the cost between any operator and an encryption operator should be equal to ∞. Finally, costs do not allow to theory to add information (this last point will be detailed in the examples).
  - Construct $\mathcal{T}$ as $\mathcal{T}(m) = \mathcal{G}(m_{\downarrow})$.

  The function $\mathcal{T}$ constructed in that way will be n-SCA.

2. Define a $\mathcal{C}$-coherent homomorphism I from $((2^{\mathcal{A}})^n, \subseteq)$ to $(\mathcal{L}^n_{\overline{\mathcal{X}}}, \sqsupseteq^n_{\overline{\mathcal{X}}})$. Given a security lattice $(\mathcal{L}, \sqsupseteq)$, a $\mathcal{C}$-coherent homomorphism can be built simply by following these steps:
   - Define mappings $i_0, i_1, \ldots, i_{n-1}$ from $(2^{\mathcal{A}}, \subseteq)$ to $(\mathcal{L}, \sqsupseteq)$ that are $\mathcal{C}$-coherent or simply choose for $j = 0$ to $n-1$ $i_j = \ulcorner \cdot \urcorner$ defined in the context of verification.
   - Extend the mappings $i_0, i_1, \ldots, i_{n-1}$ to deal with sets of messages. The extended mappings are homomorphism.
   - Define the mapping I as follows:

     $$\mathsf{I}(M_0 + M_1 * x \ldots + M_{n-1} * x^{n-1}) = i_0(M_0) + i_1(M_1) * x \ldots + i_{n-1}(M_{n-1}) * x^{n-1}$$

   - Extend this mapping to deal with variables *i.e.*$\mathsf{I}_{\overline{\mathcal{X}}}$
3. Define a $\mathcal{C}$-safe function F as:

   $$\mathsf{F} = \mathsf{I}_{\overline{\mathcal{X}}} \circ \mathsf{S}^{\overrightarrow{n}}_{\mathcal{T}}$$

This guideline makes easy the construction of a safe interpretation function. However in this paper, we did not study the existence of such functions specially when a protocol respects the secrecy property, but this would be a in our future works.

5.4. Example 1: The DEK Function

In this example, we define a $\mathcal{C}_0$-safe interpretation function where $\mathcal{C}_0$ is the verification context described in example 2. To that end, we follow the previous steps:

1. Define a n-polynomial selection function $\mathsf{S}^{\overrightarrow{n}}_{\mathcal{T}}$:
   - Choose a value for n: In this example, we choose $n = 0$ in order to select keys that encrypt directly a message.
   - Let $\mathcal{G}_0$ be the function that sets values to nodes as shown by Fig. 2. In this example, Intuitively, $\mathcal{G}_0$ allows to select the innermost keys that encrypt directly α in M and these keys will be called the direct encrypting keys.

     Where $n = \begin{cases} \infty & \text{if } x \in \{enc, dec\} \\ 0 & \text{if } x = pair \\ -1 & \text{else} \end{cases}$

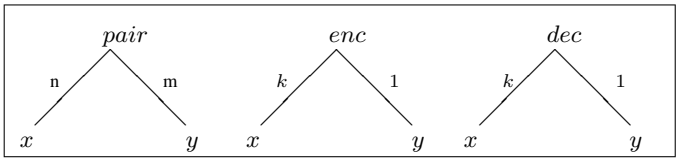

Figure 2. Costs Assignment $\mathcal{G}_0$

and $m = \begin{cases} \infty & \text{if } y \in \{enc, dec\} \\ 0 & \text{if } y = pair \\ -1 & \text{else} \end{cases}$

and $k = \begin{cases} \infty & \text{if } x \in \{enc, dec\} \\ 0 & \text{if } x = pair \\ -1 & \text{else} \end{cases}$

we can easily verify that $\mathcal{G}_0$ satisfy the required conditions: direct encryption keys are always selected, the selection beyond encryption is forbidden, and the theory of the context of verification do not add more information.

- Define $\mathcal{T}_0$ as $\mathcal{T}_0(m) = \mathcal{G}_0(m_\downarrow)$.

2. Define a $\mathcal{C}$-coherent homomorphism I from $((2^{\mathcal{A}}), \subseteq)$ to $(\mathcal{L}_{\overline{\mathcal{X}}}, \sqsupseteq_{\overline{\mathcal{X}}})$.
   - Define a mapping i_0 from $(2^{\mathcal{A}}, \subseteq)$ to $(\mathcal{L}, \sqsupseteq)$ that are $\mathcal{C}$-coherent or simply choose $i_0 = \ulcorner \cdot \urcorner_0$, where $\ulcorner \cdot \urcorner_0$ is defined in the context of verification $\mathcal{C}_0$.
   - Let extend this mapping with steps described previously to the mapping $\mathsf{I}^0_{\overline{\mathcal{X}}}$ that is from $((2^{\mathcal{A}})^0, \subseteq^0)$ to $((2^{\mathcal{T}}_{\overline{\mathcal{X}}})^0, \subseteq^0_{\overline{\mathcal{X}}})$.
3. The interpretation function is now as follows:

$$\mathsf{F} = \mathsf{I}_{\overline{\mathcal{X}}} \circ \mathsf{S}^{\overrightarrow{0}}_{\mathcal{T}_0}$$

For instance:

$$\begin{aligned} \mathsf{F}(\alpha, \{\{\alpha, B, X\}_{k_{as}}\}_{k_{bs}}) &= \mathsf{I}^0_{\overline{\mathcal{X}}}(\{k_{as}\}) \\ &= \{A, S\} \end{aligned}$$

This function is called the DEK function (**D**irect **E**ncrypting **K**eys). This function is $\mathcal{C}_0$-safe (theorem 5.4). The definition of DEK function could be easily extended when dealing with other syntaxes as syntax in which signature is possible.

Now, let study in what theory the DEK function is safe. The safety condition that makes the theory involved is: the theory do not allow to add more information:

$$\forall m \rightarrow_{\mathcal{E}} m' \in \mathcal{W}_{\mathcal{E}} \cdot \mathcal{G}(m') \preceq \mathcal{G}(m)$$

This condition means intuitively that the selected elements in the right side of an equation in the convergent rewriting system of $\mathcal{E}$, should be selected also in the left side of this equation. Hence, the DEK function will not be safe in the theory that has a convergent rewriting system that allows to select more elements in the right side of a rewriting rule than the right side. In fact, the selection of the DEK function concerns the encryption keys. More precisely, it concerns the encryption symbols.

Hence, the algebraic properties that could affect the safety of the DEK function are the properties in which the encryption symbols are involved. Specially, properties that allow to change keys. The commutativity of encryption property $enc(enc(x, y), z) = enc(enc(x, z), y)$, is an example of such properties. Thus, if the theory of the context of verification contains the commutativity of encryption property for one of encryption function symbols, then the DEK function is not safe in that context of verification. Moreover, it is impossible to find a safe interpretation function for theories that have the commutativity property of encryption. Indeed, this property puts in contradiction the safety conditions of an interpretation function: selecting elements beyond encryption is forbidden and the algebraic properties gives the same security level.

5.5. Example 2: The DEKAN Function

In this example, we define an $\mathcal{C}_0$-safe interpretation function where $\mathcal{C}_0$ is the verification context described in section 2. To that end, we follow steps described in section 5:

1. Define a n-polynomial selection function $\mathsf{S}_{\mathcal{T}}^{\overrightarrow{n}}$:
   - Choose a value for n: In this example, we choose $n = 1$.
   - Let $\mathcal{G}_1$ be the function that sets values to nodes as shown by Fig. 3. In this example, Intuitively, $\mathcal{G}_1$ allows to select the innermost keys that encrypt α in M together with the atomic components that are concatenated to α in M (called direct neighbors).

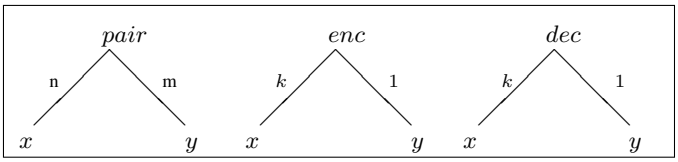

Figure 3. Safe Costs Assignment $\mathcal{T}_1$

   Where n, m , and k is as defined by Table 2 and where -1 is changed in this example to 0 in orer to select both neighbors and keys of encryption.

   we can easily verify that $\mathcal{G}_0$ satisfy the required conditions: direct encryption keys are always selected, the selection beyond encryption is forbidden, and the theory of the context of verification do not ass more information.
   - Define $\mathcal{T}_1$ as $\mathcal{T}_1(m) = \mathcal{G}_1(m_{\downarrow})$.
2. Define a $\mathcal{C}$-coherent homomorphism I from $((2^{\mathcal{A}})^1, \subseteq)$ to $(\mathcal{L}_{\overline{\mathcal{X}}}^1, \sqsupseteq_{\overline{\mathcal{X}}}^1)$.
   - Define mappings i_0 and i_1 from $(2^{\mathcal{A}}, \subseteq)$ to $(\mathcal{L}, \sqsupseteq)$ that are $\mathcal{C}$-coherent or simply choose $i_0 = \ulcorner \cdot \urcorner_0 \dagger [A \mapsto A, B \mapsto B, S \mapsto S, \ldots]$, where $\ulcorner \cdot \urcorner_0$ is defined in the context of verification $\mathcal{C}_0$. For i_1 we can choose $i_1 = \ulcorner \cdot \urcorner_0$.
   - Let extend this mapping with steps described previously to the mapping $\mathsf{I}_{\overline{\mathcal{X}}}^0$ that is from $((2^{\mathcal{A}})^0, \subseteq^0)$ to $((2_{\overline{\mathcal{X}}}^{\mathcal{T}})^1, \subseteq_{\overline{\mathcal{X}}}^1)$.
   - Define the mapping $\mathsf{I}(M_0) = (i_0(M_0), i_1(M_1))$.
   - Let extend this mapping with steps described previously to the mapping $\mathsf{I}_{\overline{\mathcal{X}}}$ that is from $((2^{\mathcal{A}})^1, \subseteq^1)$ to $((2_{\overline{\mathcal{X}}}^{\mathcal{T}})^1, \subseteq_{\overline{\mathcal{X}}}^1)$.

3. The interpretation function is now as follows:

$$\mathsf{F} = \mathsf{I}_{\overline{\mathcal{X}}} \circ \mathsf{S}_{\mathcal{T}_1}^{\overrightarrow{1}}$$

For instance:

$$\begin{aligned} \mathsf{F}(\alpha, \{\{\alpha, B, X\}_{k_{as}}\}_{k_{bs}}) &= \mathsf{I}_{\overline{\mathcal{X}}}(\{To_B, X\}, \{k_{as}\}) \\ &= \{B, \overline{X}\} \ + \ \{A, S\} * x^1 \end{aligned}$$

This function is called the DEKAN function (**D**irect **E**ncrypting **K**eys **a**nd **N**eighbors). This function is $\mathcal{C}_0$-safe. We believe that such function is powerful enough to allow us to appropriately analyze a large class of protocols with respect to the secrecy property. Also, The definition of DEKAN function could be easily extended when dealing with other syntaxes.

If we are sure that agents identifiers are not used in the analyzed protocole we can use only distance 0 to select both keys of encryption and the neighbors. Indeed, the reason of separating that distances of encryption and concatenation, is because we need to interpret in different way the agents identifiers according to theirs functions in the message. Indeed, when agent identifiers are used as a key, the security level of that agent identifiers are needed to be interpreted as public, and when an agent identifier is used as a neighbor, the security level is needed to be interpreted as agent names. Hence, if agents identifiers are not used as keys in a protocol, the DEKAN function could be used with distance 0 for the neighbors and keys of encryption. For instance:

$$\begin{aligned} \mathsf{F}_1(\alpha, \{\{\alpha, B, X\}_{k_{as}}\}_{k_{bs}}) &= \mathsf{I}_{\overline{\mathcal{X}}}(\{To_B, X\}, \{k_{as}\}) \\ &= \{A, B, S, \overline{X}\} \end{aligned}$$

For the sake of simplicity, this last function will be used as the DEKAN function in the following of this paper. In general, if we do not need to interpret the same messages in different ways, we can use the distance 0 for all distances. Else, we should split distances according to the interpretation of messages.

Now, let study in what theory the DEKAN function is safe. The condition that makes the theory involved is:

$$\forall m \rightarrow_{\mathcal{E}} m' \in \mathcal{W}_{\mathcal{E}} \cdot \mathcal{G}(m') \ \preceq \ \mathcal{G}(m)$$

This condition means intuitively that the selected elements in the left of an equation in the convergent rewriting system of $\mathcal{E}$, should be selected also in the right side of this equation. Hence, the DEKAN function will not be safe in the theory that has a convergent rewriting system that allows to select more elements in the right side of a rewriting rule than the left side. In fact, the selection of the DEKAN function concerns the encryption keys and the neighbors. More precisely, it concerns the encryption and concatenation function symbols. Hence, the algebraic properties that could affect the safety of the DEKAN function are the properties in which the encryption and concatenation are involved.

First, the DEKAN function is not safe for the properties that not make the DEK function safe (*i.e.*$enc(enc(x, y), z) = enc(enc(x, z), y)$, since the DEKAN function selects the encryption keys as the DEK function.

Another example that could make the DEKAN function not safe, the homomorphism property $enc(pair(x, y), z) = pair(enc(x, z), enc(y, z))$. In fact, the selection function of DEKAN function selects for x, both y and z in the right of the homomorphism equation, however it selects only z

in the left side of the equation. Therefore, if the convergent rewriting system obtained by orienting the theory, contains $enc(pair(x, y), z) \rightarrow pair(enc(x, z), enc(y, z))$, then selected elements in the right side is part of the left side, so in that case the DEKAN function is safe. However, if the convergent rewriting system obtained by orienting the theory with the homomorphism property, contains $pair(enc(x, z), enc(y, z))$
$\rightarrow enc(pair(x, y), z)$, then selected elements in the right side is not part of the left side. Therefore, the DEKAN function is not safe in that context. In general, the convergent rewriting system obtained by orienting the theory with the homomorphism property is the rewriting system that contains the first rule. Hence, the DEKAN function is in general safe under a theory that contains the homomorphism property.

In the same way, we can notice that the DEKAN function is safe under the equational theory that contains prefix property $\{m_1, \ldots, m_n\}_k = \{m_1, \ldots, m_i\}_k, \{m_i, \ldots, m_n\}_k$.

6. Analysis of Cryptographic Protocols under Different Equational Theories

This section presents the analysis of some famous cryptographic protocols as BULL, Woo and Lam, etc. Although almost of them are already discovered flawed, the choice of those protocols is motivated by the fact that we want to show that the approach is not only efficient to guaranty the secrecy property when some conditions are met, but also it can show the weaknesses of the analyzed protocol and gives an idea on how will be the flaw or on how to correct it. Indeed, the interpretation functions-based method is based on some conditions that are sufficient for the secrecy property. These conditions could be verified only on the description of the protocol. When these conditions are met, the protocol respects the secrecy property. Else, we can not deduce the non-security of the protocol because the method does not allow to infer the flaw. This fact could not be considered as a drawback of the approach because, in that case, the analysis shows us where the exact weaknesses of that protocol, and so it gives us an idea on how to correct them. This fact will be shown in different examples in this section.

Also, This section shows us how the verification process is easy and could be made automatic. Indeed, the verification of the secrecy property as presented in theorem 4.3, consists simply on checking whether the protocol is an increasing one with using a safe interpretation function. To that end, we use the interpretation functions given as example in section 5.4.

6.1. Who and Lam Protocol

To the lake of the space we omitted the analysis of of Who and Lam protocol, but it could be found in [8, 9, 10, 11]. This analysis shows the efficiency of the approach to show weaknesses of a protocol, and also to correct them.

6.2. Shamir Three Pass Protocol and Commutativity of Encryption

This example shows how the approach could help to find weaknesses that concerns the cryptosystem (encryption).

Let's consider the Shamir Three Pass protocol cited in [4] and described by Table 3. This protocol allows two agents to exchange a secret message without sharing any initial secret.

$$\begin{array}{ll}\text{Message 1}: & A \rightarrow B : \{M\}_{k_a} \\ \text{Message 2}: & B \rightarrow A : \{\{M\}_{k_a}\}_{k_b} \\ \text{Message 3}: & A \rightarrow B : \{M\}_{k_b}\end{array}$$

Table 3. Shamir Three Pass Protocol

Let's extend the equational theory $\mathcal{E}_0$ with the commutativity of encryption:

$$\{\{m\}_x\}_y = \{\{m\}_y\}_x$$

As we have discussed in section 5.4, and section 5.5, the DEK and DEKAN are not safe under equation theory that contains the commutativity of encryption property. In fact, all interpretation functions that are constructed as the guideline, given in this paper, could not be safe with that property since that property makes in contradiction the safety conditions (Selecting elements beyond encryption is forbidden and the algebraic properties gives the same security level). In fact, the Shamir Three Pass protocol has a flaw, cited in [4], that is due the commutativity property. This fact conducts us to deduce for the secrecy property, the following principal:

P1- A good encryption operator should not have the commutativity property.

6.3. Bull Protocol and $\oplus$ Properties

Let's consider the Bull Protocol cited in [3] and described by Table 4. This protocol aims to share a fresh session key between a number of agents and a server.

$$\begin{array}{lllll}
\text{Message 1}: & A & \rightarrow & B: & h([A,\ B,\ N_a],\ k_{as}),\ [A,\ B,\ N_a]\ (= X_a) \\
\text{Message 2}: & B & \rightarrow & C: & h([B,\ C,\ N_b,\ X_a],\ k_{bs}),\ [B,\ C,\ N_b,\ X_a]\ (= X_b) \\
\text{Message 3}: & C & \rightarrow & S: & h([C,\ S,\ N_c,\ X_b],\ k_{cs}),\ [C,\ S,\ N_c,\ X_b]\ (= X_c) \\
\text{Message 4}: & S & \rightarrow & C: & A,\ B,\ k_{ab} \oplus h(N_a, k_{as}),\ \{A,\ B,\ N_a\}_{k_{ab}}, \\
 & & & & B,\ A,\ k_{ab} \oplus h(N_b, k_{bs}),\ \{A,\ B,\ N_b\}_{k_{ab}}, \\
 & & & & B,\ C,\ k_{bc} \oplus h(N_b, k_{bs}),\ \{A,\ B,\ N_b\}_{k_{bc}}, \\
 & & & & C,\ B,\ k_{bc} \oplus h(N_c, k_{cs}),\ \{A,\ B,\ N_c\}_{k_{bc}}, \\
\text{Message 5}: & C & \rightarrow & B: & A,\ B,\ k_{ab} \oplus h(N_a, k_{as}),\ \{A,\ B,\ N_a\}_{k_{ab}}, \\
 & & & & B,\ A,\ k_{ab} \oplus h(N_b, k_{bs}),\ \{A,\ B,\ N_b\}_{k_{ab}}, \\
 & & & & B,\ C,\ k_{bc} \oplus h(N_b, k_{bs}),\ \{A,\ B,\ N_b\}_{k_{bc}}, \\
\text{Message 6}: & B & \rightarrow & S: & A,\ B,\ k_{ab} \oplus h(N_a, k_{as}),\ \{A,\ B,\ N_a\}_{k_{ab}},
\end{array}$$

Table 4. Bull Protocol

Let's extend the equational theory $\mathcal{E}_0$ with the following properties:

$$\begin{array}{rcll}
x \oplus (y \oplus z) & = & (x \oplus y) \oplus z & \text{(Associativity)} \\
x \oplus y & = & y \oplus x & \text{(Commutitavity)} \\
x \oplus 0 & = & x & \text{(Netral Element)} \\
x \oplus x & = & 0 & \text{(Nilpotence)}
\end{array}$$

In Bull protocol the $\oplus$ is used as an encryption operator. The most important property that makes the protocol working is the associativity property. However, the associativity property of $\oplus$ makes non safe the DEK, the DEKAN, and all others functions that are computed as the guideline of this

paper. Indeed, the associativity property of $\oplus$, puts in contradiction the safety conditions that are: selecting elements beyond encryption is forbidden and algebraic properties gives the same security level. So, in that case, we could not use the DEK and DEKAN function as a safe interpretation functions to verify the secrecy property of the Bull protocol unless we consider the $\oplus$ operator as a non-encryption operator. In that case, we can notice that the secrecy k_{ab} is sent in clear in the Bull protocol and that is not secure. In fact, the protocol has a flaw [15] and it is based on the associativity property of $\oplus$ (the same that makes the protocols working). This fact conduct us to deduce for the secrecy property, the following principal:

P2- A good encryption operator should not have the associativity property.

6.4. Needham-Schroeder Lowe Protocol and Homomorphism

This example shows the impact of the homomorphism property on the security of a protocol. Let's consider the Needham-Schroeder Lowe protocol cited in [12] and described by Table 5. This protocol aims to authenticate mutually two agents to each others.

$$\begin{array}{lll} 1. & A \rightarrow B & : \{A, N_a\}_{k_b} \\ 2. & B \rightarrow A & : \{\langle N_a,\ N_b\rangle,\ B\}_{k_a} \\ 3. & A \rightarrow B & : \{N_b\}_{k_b} \end{array}$$

Table 5. Needham Schroeder Lowe Protocol

By using the DEKAN function, we can prove that the Needham-Schroeder Lowe protocol respects the secrecy property under the equational theory $\mathcal{E}_0$.

Now let suppose that the equational theory $\mathcal{E}_0$ contains the homomorphism property:

$$\{m_1, m_2\}_k = \{m_1\}_k.\{m_1\}_k$$

The convergent rewriting system corresponding to the theory of $\mathcal{E}_0$ is the rewriting system that contains the following rule:

$$\{m_1, m_2\}_k \ \rightarrow\ \{m_1\}_k, \{m_1\}_k$$

With this rule, the DEKAN function will be always equal to the DEK function. Indeed, the selection function of the DEKAN function does not choose elements on the message m but on $m_{\downarrow}$, so the selection function choose elements on messages that have the form $\{\alpha_1\}_k, \ldots, \{\alpha_n\}_k$, where α_i are atomic messages. Hence, the neighbors will never be chosen, and so the DEKAN function becomes like the DEK function. Therefore with the homomorphism property, we will not be able to ensure the secrecy property of the last protocol while we was able without this property as shown previously. This due to the fact that the homomorphism property makes the selection of of neighbors impossible, and so to use neighbors to get the security level of a message. Indeed, with the homomorphism property, the intruder could change easily the neighbors of a message. As an example, let's take the step 2 in the last protocol. In this step, agent B sends the message $\{\langle N_a,\ N_b\rangle,\ B\}_{k_a}$ to A. If the intruder intercepts this message, he can compute the message $\{\langle N_a,\ N_b\rangle\}_{k_a}$ and the message $\{I\}_{k_a}$. After that, he can produce the message $\{\langle N_a,\ N_b\rangle\}_{k_a}, \{I\}_{k_a}$ that is equal to the message $\{\langle N_a,\ N_b\rangle,\ I\}_{k_a}$ according to the homomorphism property. Hence, he can mislead the agent A that the nounce N_b is coming from him and so he can get it. So, with the homomorphism property the intruder can perform an attack which he was unable to perform without this property. So, to correct that weakness, we should find a means (different from the concatenation), to say to

$$\begin{array}{lll} 1. & A \rightarrow B & : \{\{N_a\}_{k_a^{-1}}\}_{k_b} \\ 2. & B \rightarrow A & : \{\langle\{N_a,\ N_b\rangle\}_{k_b^{-1}}\}_{k_a} \\ 3. & A \rightarrow B & : \{N_b\}_{k_b} \end{array}$$

Table 6. Modified Needham Schroeder Lowe Protocol

A that N_b is from B. For example, the signature could be used to insure the origine of the message. Hence, we could modify the Needham Schroeder Lowe protocol as follows:

By using the DEKAN function, this new version of Needham Schroeder Lowe protocol could be proved secure with respect to the secrecy property even with the homomorphism property. Indeed, the signature should not be considered as an encryption operator, and costs of the signature operator could be the same as costs of pair operator described by Table 3. Moreover, the encryption with a public key should be considered as an encryption operator, and the costs could be the same as costs of encryption operator enc described by Table 3. With this extension of the DEKAN, the protocol could be easily proved secure with respect to the secrecy property.

7. Conclusion

This paper shows the efficiency and flexibility of the interpretation function-based method through the analysis of some famous protocols. In fact by using the notion of a context of verification and a safe interpretation functions, the interpretation function-based method gives general and sufficient conditions allowing to guarantee the secrecy property of cryptographic protocols. This result is true under any context of verification (even using equational theories). This fact gives a certain flexility to the interpretation function-based method and makes it more general to prove the secrecy property of a large class of protocols. Moreover, the approach does not only help a lot to discover protocols weaknesses, but also helps to correct them and so to transform a non-secure protocols to a secure ones.

As future works, we want to analysis more cryptographic protocols under more algebraic properties (more context of verification). Also, we aim to extend the approach to other security properties as authentication, integrity, etc.

References

[1] Martín Abadi and Véronique Cortier. Deciding knowledge in security protocols under equational theories. *Theor. Comput. Sci.*, 367(1):2–32, 2006.

[2] M. Boreale and D. Gorla. Process calculi and the verification of security properties. *Journal of Telecommunication and Information Technology— Special Issue on Cryptographic Protocol Verification*, (4/02):28–40, 2002.

[3] J. A. Bull and D. J. Otway. A nested mutual authentication protocol. *SIGOPS Oper. Syst. Rev.*, 33(4):42–47, 1999.

[4] J. Clark and J. Jacob. A survey of authentication protocol literature. Technical report, 1997.

[5] H. Comon and V. Shmatikov. Is it possible to decide whether a cryptographic protocol is secure or not. *Journal of Telecommunications and Information Technolog,*, 2002.

[6] H. Comon-Lundh and V. Cortier. New decidability results for fragments of first-order logic and application to cryptographic protocols. In *RTA*, pages 148–164, 2003.

[7] N. Dershowitz and J.P. Jouannaud. Rewrite systems. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 243–320. 1990.

[8] H. Houmani and M. Mejri. Practical and universal interpretation functions for secrecy. In *International Conference on Security and Cryptography: Secrypt*, Barcelona, Spain, August 2007.

[9] H. Houmani and M. Mejri. Secrecy by interpretation functions. *Journal of Knowledge-Based Systems*, 20(7):617–635, 2007.

[10] H. Houmani and M. Mejri. Secrecy by interpretation functions. *International Journal of Computers*, 2008.

[11] H. Houmani and M. Mejri. Sufficient conditions for secrecy under equational theories. In *The 2nd International Conference on Information Security and Assurance*, Busan, Korea, April 2008. IEEE CS.

[12] G. Lowe. An attack on the needham-schroeder public-key authentication protocol. *Inf. Process. Lett.*, 56(3):131–133, 1995.

[13] C. Meadows. What makes a cryptographic protocol secure? the evolution of requirements specification in formal cryptographic protocol analysis. In *Proceedings of ESOP 03*. Springer-Verlag, April 2003.

[14] L. C. Paulson. Mechanized proofs for a recursive authentication protocol. In *10th Computer Security Foundations Workshop*, pages 84–95. IEEE Computer Society Press, 1997.

[15] P. Y. A. Ryan and S. A. Schneider. An attack on a recursive authentication protocol: A cautionary tale. *Inf. Process. Lett.*, 65(1):7–10, 1998.

[16] A. Sabelfeld and A. Myers. Language-based information-flow security, 2003.

[17] T. Y. C. Woo and S. S. Lam. A Lesson on Authentication Protocol Design. *Operating Systems Review*, pages 24–37, 1994.