Towards an Automatic Analysis of Web Service Security

Yannick Chevalier[1] and Denis Lugiez[2] and Michaël Rusinowitch[3] *

[1] IRIT, Team LiLac, Université de Toulouse, France. email: ychevali@irit.fr
[2] LIF, CNRS, Aix-Marseille Université, France. email: lugiez@lif.univ-mrs.fr
[3] LORIA-INRIA-Lorraine, France. email: rusi@loria.fr

Abstract. Web services send and receive messages in XML syntax with some parts hashed, encrypted or signed, according to the WS-Security standard. In this paper we introduce a model to formally describe the protocols that underly these services, their security properties and the rewriting attacks they might be subject to. Unlike other protocol models (in symbolic analysis) ours can handle non-deterministic receive/send actions and unordered sequence of XML nodes. Then to detect the attacks we have to consider the services as combining multiset operators and cryptographic ones and we have to solve specific satisfiability problems in the combined theory. By non-trivial extension of the combination techniques of [3] we obtain a decision procedure for insecurity of Web services with messages built using encryption, signature, and other cryptographic primitives. This combination technique allows one to decide insecurity in a modular way by reducing the associated constraint solving problems to problems in simpler theories.

Keywords: Security, Web services, verification, cryptographic protocols, combination of decision procedures, equational theories, rewriting.

1 Introduction

Web services promise to be a standard technology for Internet and enterprise networks. They require the ability to transmit securely messages in XML syntax using the SOAP protocol. Messages that travel over the networks can be observed and modified by intruders. Hence the protocol was extended by W3C for allowing one to sign and encrypt some parts of the contents. Nevertheless, as for classical protocols, cryptographic mechanisms are not sufficient for securing Web services. They can be subject to the same attacks (e.g. man-in-the middle) as classical cryptographic protocols, but the XML syntax and the specific way messages are processed (e.g. not examining the full content) gives the opportunity to mount a new class of attacks, such as XML rewriting attacks, as shown in [1].

Recently many decision procedures have been proposed for analysing cryptographic protocols. These procedures rely on automated deduction and constraint solving procedures extending semantic unification. Our objective in this work is

* this work has been supported by ACI-Jeunes Chercheurs Crypto, ACI-SI SATIN and ARA SSIA Cops

to investigate whether these results can be lifted to Web services. First it appears immediately that the flexible XML format requires one to consider message contents as sets of terms rather than terms: this introduces a first difficulty since no security decidability results exist yet for protocols using such a data structure. Second, since messages are only partially parsed by SOAP, the answer to a request might depend on the implementation: in other words we have to model a non-deterministic behaviour of the Web service protocols.

A role-based protocol model has been proposed in previous works [2]. This model admits two versions. In the first one, the *equational model*, operations (including decryption) are explicit and the basic operations are deterministic. We believe that this constraint is inherent to the model, as non-determinism in equational theories leads to inconsistency. The second model, the *pattern-matching model*, relies on patterns to filter incoming messages. It has been widely studied but proofs in this model are often complex and obscure, and there is no general combination result that would permit one to extend existing decision procedures to a new operation (the multiset operator, in our case).

Motivated by these facts, and also for the sake of deriving a uniform framework, we propose a protocol model that is more general than the two commonly used ones. It represents XML messages as multisets of terms and simulates non-deterministic behaviours. Moreover decidability and combination results (previous known only for the equational model [3]) have been lifted to this model. The proofs are involved and cannot be obtained from the ones in simpler models.

Related work. The Samoa Project project [1] offers a language to express SOAP-based security protocols which is compiled into the applied π calculus on which the resolution based system ProVerif is run to verify secrecy and authentication properties. Our approach is a complementary one since it provides decision procedures that are complete for finitely many sessions, even with an associative-commutative XML message constructor and other operators such as Dolev Yao encryption/decryption. A lot of work has been dedicated to security analysis of protocols modulo algebraic properties (see e.g. [4–6]). However no decidability results have been reported for associative or associative-commutative operators, supporting the combination with useful cryptographic primitives (such as XOR). To our knowledge, the ones we report here are the first of this kind, and we show how they apply to deciding security properties of XML services. Our results rely on extensive use of term rewriting and unification techniques [8].

Paper organization. First, we describe an example of XML-rewriting attacks in Sec. 2, then we recall the notions of term rewriting we use on our model in Sec. 3. Sec. 4 formally models service executions by the deductions that can be performed by intruders and honest agents. In particular we introduce *symbolic derivations* as parameterized deductions. They can be viewed as a variant of *strand spaces*, a fundamental concept in protocol analysis [9]. Security in this model reduces to a constraint satisfiability problem. We give an algorithm for solving it for the case of plain Web services in Sec. 5. Sec. 6 presents a combi-

nation scheme that applies to multiset and cryptographic operators: it provides us with a decidable analysis of XML protocols.

The long version [10] of this paper contains all missing lemmas and proofs.

2 An example of Web service and XML rewriting attack

Web services can be described as a set of receive/send action between clients and the provider, where the messages are XML terms following the SOAP format. Messages are constructed and parsed according to the security policy of the service which usually requires that some parts of the message are encrypted, signed using known algorithms like sha1, rsa,..., may contain timestamps, make reference to trusted third parties,.... A SOAP message is an envelope consisting of a mandatory body and of an optional header. The header contains the useful information about the message and usually has a security part that follows the WS-security specification defined by the OASIS organization. For instance, Alice may order a ticket for Bremen for herself to be charged to her account by sending send the following message to Bob, her correspondent in the travel agency:

```
<Envelope>
  <Header>
    <To> http://www.travel-for-free.com/~bob </>
    <Security>
       <timestamp>2007-04-16T09:56:43Z</>
       <signature>
         <signedInfo>
           <reference URI=#1>a string</>
           <signatureValue>another string </>
  </></></></></>
  <Body>
    <Order id=1>
       <beneficiary>Alice</>
       <account>Alice</>
       <trip>Bremen</>
</></></></>
```

In the header, the first string is a digest of the body, and the other string is the encryption of this digest using the private key of Alice. For efficiency purposes, only parts of the messages are encrypted or signed. Since a node labelled by a tag 'a' may have an arbitrary number of elements, we introduce a unary symbol $a(\_)$ for each tag 'a' and an associative-commutative multiset constructor. For instance, the body of the above message corresponds to the term $\mathit{Body}(\mathit{Order}(\mathit{beneficiary}(\mathit{Alice}) \cdot \mathit{account}(\mathit{Alice}) \cdot \mathit{trip}(\mathit{Bremen})))$ where $e_1 \cdot \ldots \cdot e_n$ denotes the multiset $\{e_1, \ldots, e_n\}$. Another possibility is to replace the multiset operator by an associative operator for sequences. In Web services, components of messages are selected by their tag and security policies usually refer to this name and don't consider the possibility of multiple occurrences of the same tag.

In our example, the recipient Bob performs several security checks, including the verification of the signature, then looks for a part of the *Body* labelled *Order*,

orders a ticket for the requested trip and charges Alice's account. Then, he sends to Alice a SOAP message with a security header containing the digest of the original message and a body containing an accesscode for the trip encoded by the public key of the beneficiary (that Alice will transfer to the beneficiary). Like in the study of cryptographic protocols, we assume that cryptography is perfect, but we don't make any particular hypothesis on the implementation of parsing XML trees. An attacker Charlie may intercept the message and forge a new one that inserts a new *Order* item to the *Body*:

```
<Order id=2>
  <beneficiary>Charlie </>
  <account>Alice</>
  <trip> Hawaii</>
</>
```

before the previous *Order*. An implementation of the service may lead to a successful verification of the signature and to sending the accesscode to Charlie depending on how the XML term is parsed. If the first match of *Order* is chosen, then Charlie gets the accesscode, since the signature is correct because the reference to the initial *Order* is still used. Another implementation could parse the children of a node from right to left and select the correct *Order* subterm. Therefore the behavior of the protocol is non-deterministic which we model by rules that select a element *Order(_)* in a multiset. We shall model the service by deduction rules and equations that model all the operations that the participants can do and all algebraic relations that may hold between the operators A possible abstraction of the security protocol $\mathcal{P}$ that supports the service is:

$$A \to B : \mathrm{se}(h(\mathit{order}(x,y,z)), SK_A) \cdot \mathit{order}(x,y,z)$$
$$B \to A : \mathrm{se}(h(\mathit{order}(x,y,z)), PK_A) \cdot \mathrm{se}(\mathit{accesscode}(z), PK_x)$$

where h denotes a hashing function, $\mathit{order}(x,y,z)$ denotes the request for trip z for beneficiary x, with y the account to charge, h is a hash function, *accesscode(z)* is the code requested to get the ticket from automata, $\mathrm{se}(x,y)$ denotes the encryption of x using key y, PK_x is the public key of x, SK_x is the private key of x. The intruder deductive power is given by the classical Dolev-Yao rules (see [11, 12]) extended by a rule that allows us to select any element in a multiset. Some realistic implementations of the service (like the one in Section 4) are subject to the following attack (assuming that $C = I$ or C is compromised):

$$A \to I_B : \mathrm{se}(h(\mathit{order}(A,A,\mathit{Bremen})), SK_A) \cdot \mathit{order}(A,A,\mathit{Bremen})$$
$$I_A \to B : \mathrm{se}(h(\mathit{order}(A,A,\mathit{Bremen})), SK_A) \cdot \mathit{order}(C,A,\mathit{Hawaii}) \cdot \mathit{order}(A,A,\mathit{Bremen})$$
$$B \to I_A : \mathrm{se}(h(\mathit{order}(A,A,\mathit{Bremen})), PK_A) \cdot \mathrm{se}(\mathit{accesscode}(\mathit{Hawaii}), PK_C)$$

where I_A (resp. I_B) denotes a malicious agent I masquerading the honest participant A (resp. B), and the secret key SK_C is known by I.

3 Terms, subterms and ordered rewriting

Basic notions. We consider an infinite set of free constants C and an infinite set of variables $\mathcal{X}$. For all signatures $\mathcal{F}$ (*i.e.* a set of function symbols with arities),

we denote by $\mathrm{T}(\mathcal{F})$ (resp. $\mathrm{T}(\mathcal{F},\mathcal{X})$) the set of terms over $\mathcal{F}\cup\mathrm{C}$ (resp. $\mathcal{F}\cup\mathrm{C}\cup\mathcal{X}$). The former is called the set of ground terms over $\mathcal{F}$, while the later is simply called the set of terms over $\mathcal{F}$. Variables are denoted by x, y, terms are denoted by s, t, u, v, and finite sets of terms are written $E, F, ...$, and decorations thereof, respectively. For finite sets of terms, we abbreviate $E\cup F$ by E, F and the union $E\cup\{t\}$ by E, t.

Given a signature $\mathcal{F}$ a *constant* is either a free constant or a function symbol of arity 0 in $\mathcal{F}$. For a term t, $\mathrm{Var}(t)$ is the set of variables occurring in t and by $\mathrm{Cons}(t)$ is the set of constants occurring in t. An *atom* is either a variable or a constant and we denote by $\mathrm{Atoms}(t)$ the set $\mathrm{Var}(t)\cup\mathrm{Cons}(t)$. The application of a substitution σ to a term t (resp. a set of terms E) is denoted $t\sigma$ (resp. $E\sigma$). An *equational presentation* $\mathcal{H}=(\mathcal{F},A)$ is defined by a set A of equations $r=t$ with $r,t\in\mathrm{T}(\mathcal{F},\mathcal{X})$. For any equational presentation $\mathcal{H}$ the relation $=_{\mathcal{H}}$ denotes the equational theory generated by $(\mathcal{F},A)$ on $\mathrm{T}(\mathcal{F},\mathcal{X})$. An *$\mathcal{H}$-Unification system* $\mathcal{S}$ is a finite set of pairs of terms in $\mathrm{T}(\mathcal{F},\mathcal{X})$ denoted by $(u_i \stackrel{?}{=} v_i)_{i\in\{1,\ldots,n\}}$. The ground substitution σ satisfies $\mathcal{S}$, denoted by $\sigma\models\mathcal{S}$, iff for all $i\in\{1,\ldots,n\}$ $u_i\sigma =_{\mathcal{H}} v_i\sigma$. Syntactic subterms and positions are defined as usual (see [13, 14]), and the replacement of the subterm of t by s at position p is denoted by $t[p\leftarrow s]$, for a set of positions Π, $t[\Pi\leftarrow s]$ denotes the replacement of all subterms at position $p\in\Pi$ by s.

Congruences and ordered rewriting. Let $<$ be a simplification ordering on $\mathrm{T}(\mathcal{F})$ [4], total on $\mathrm{T}(\mathcal{F})$, such that (i) the minimal element for $<$ is a constant $\mathrm{c_{min}}\in\mathrm{C}$; (ii) each non-free constant is smaller than any non-constant ground term. $\mathrm{C_{spe}}$ denotes the set containing the constants in $\mathcal{F}$ and $\mathrm{c_{min}}$. Given a possibly infinite set of equations $\mathcal{O}$ on the signature $\mathrm{T}(\mathcal{F})$ the ordered rewriting relation $\rightarrow_{\mathcal{O}}$ is defined by $t\rightarrow_{\mathcal{O}} t'$ iff there exists a position p in t, an equation $l=r$ in $\mathcal{O}$, a substitution τ such that $t=t[p\leftarrow l\tau]$, $t'=t[p\leftarrow r\tau]$, and $l\tau>r\tau$.

It has been shown that by applying the *unfailing completion procedure* to a set of equations $\mathcal{H}$ yields a (possibly infinite) set of equations $\mathcal{O}$ such that: (i) the congruence relations $=_{\mathcal{O}}$ and $=_{\mathcal{H}}$ are equal on $\mathrm{T}(\mathcal{F})$, (ii) the ordered rewrite relation $\rightarrow_{\mathcal{O}}$ is convergent (*i.e.* terminating and confluent) on $\mathrm{T}(\mathcal{F})$. We say that $\mathcal{O}$ is an *o-completion* of $\mathcal{H}$. Since the rewrite system $\rightarrow_{\mathcal{O}}$ is convergent on ground terms, we can define $(t)\downarrow_{\mathcal{O}}$ as the unique normal form of the ground term t for $\rightarrow_{\mathcal{O}}$. A ground term t is in *normal form*, or *normalised*, if $t=(t)\downarrow_{\mathcal{O}}$. Given a ground substitution σ we denote by $(\sigma)\downarrow_{\mathcal{O}}$ the substitution with the same support such that $x(\sigma)\downarrow_{\mathcal{O}}=(x\sigma)\downarrow_{\mathcal{O}}$ for all variables x in the support of σ. A substitution σ is *normal* if $\sigma=(\sigma)\downarrow_{\mathcal{O}}$.

4 Modelling service execution

4.1 A model for secure Web services

Let us first briefly review the situation we want to model. Some simple Web services are akin to functions in a library. Their execution is triggered by the

[4] by definition $<$ satisfies for all $s,t,u\in\mathrm{T}(\mathcal{F})$ $s<t[s]$ and $s<u$ implies $t[s]<t[u]$

reception of a request from a client, to which they immediately respond. These services are advertised by a WSDL specification that defines among other things the contents of the input and output messages. It is also possible to specify some cryptographic protection for integrity, confidentiality and authenticity of a request by means of a published security policy. This policy will constrain acceptable requests by mandating that some parts have to be signed, encrypted or integrity protected. These parts are expressed either by identifiers or by XPath expressions, and we say that these services are *protected*. Finally, some WS standards, *e.g.* BPEL, WS-SecureConversation, and others, permit to express sequences of simple service invocations, which we call *workflows*. We call totally ordered sequences *workflow executions*, as they also correspond to traces of service workflows.

The analysis of Web services is thus very similar to the one of cryptographic protocols. A protected service workflow is a composition of *roles* (client, service,...), and a workflow execution is akin to a protocol execution. *Agents* impersonate the roles in a workflow execution.

This similarity shall not, however, hide the differences at the level of messages. In the case of cryptographic protocols, the patterns of admissible messages are fixed, and known by the intruder, whereas security policies just express constraints on the presence of some particular subterms at some position in a message. Moreover, and since services are in most cases automatically generated, the verification of a message is likely to be independent of the construction of a response. Finally, some implementations may return only one node in a hedge when several ones correspond to an XPath expression, thereby allowing for XML injection attacks. The work described in this paper focuses on security policies expressing paths of subterms from the root (envelope) of the message, leaving general XPath constraints to future work.

Example 1. A client A invokes a service B by sending $\mathrm{se}(h(order(x,y,z)), SK_A) \cdot order(x,y,z)$, where x, y and z are instantiated parameters of the client. Then it receives a response r which is parsed to check that it contains the nodes $\mathrm{se}(z, PK_A)$ and $\mathrm{se}(u,v)$ at its root, with $v = PK_x$ and $z = h(order(x,y,z))$.

The intruder and the insecurity problem. In the Dolev-Yao model [11], attacks on protocols are modelled by the addition of a malicious participant, called the intruder, that controls the network. It can intercept, block and/or redirect all messages sent by honest agents. It can also masquerade its identity and take part in the protocol under the identity of an honest participant. Its control of the communication network is modelled by assuming that all messages sent by honest agents are sent directly to the intruder and that all messages received by the honest agents are always sent by the intruder. Besides the control on the net, the intruder has specific rules to deduce new values and compute messages. From the intruder's point of view a finite execution of a protocol is therefore the interleaving of a finite sequence of messages it has to send and a finite sequence of messages it receives (and add to its knowledge). Therefore the intruder is simply an additional role that runs concurrently with the honest participants.

The protocol is insecure if some secret knowledge is revealed during the execution of the protocol, which is modelled by adding a last step that reveals the secret. The insecurity problem amounts to finding a sequence of actions of the intruder such that its composition with the extended protocol is executable.

4.2 Deduction systems

We give a formal model for roles and the execution of roles (including the intruder). Messages are ground terms and deduction rules are rewrite rules on sets of messages representing the knowledge of an agent. Each role derives new messages from a given (finite) set of messages by using deduction rules. Furthermore, these derivations are considered *modulo* the equational congruence $=_{\mathcal{H}}$ generated by the equational axioms satisfied by the function symbols of the signature $\mathcal{F}$. Let $\mathcal{O}$ be an o-completion of $\mathcal{H}$. We write $(t)\!\downarrow$ as a short-hand for $(t)\!\downarrow_{\mathcal{O}}$.

Definition 1 *A* deduction rule *is a rule* $d : t_1, \ldots, t_n \to t$ *with* $t_1, \ldots, t_n, t$ *terms s.t.* $\mathrm{Const}((t\sigma)\!\downarrow) \subseteq \cup_{i=1}^{n} \mathrm{Const}((t_i\sigma)\!\downarrow) \cup \mathrm{C_{spe}}$ *for any ground substitution* σ*.*

This condition is very similar to the *origination* condition for well-definedness in [16]. When the equational theory is regular (i.e. the same variables occur on each side of axioms) then this condition holds iff $\mathrm{Var}(t) \subseteq \mathrm{Var}(t_1, \ldots, t_n)$.

Example 2. $x, y \to \langle x, y \rangle$ is a deduction rule when assuming that the set of equational axioms is empty. This rules allows an agent (who can be the intruder) to construct a new pair from two values already known.

Deduction rules are the basic ingredients for deduction systems.

Definition 2 *A* deduction system $\mathcal{D}$ *is a triple* $\langle \mathcal{F}, \mathcal{R}, \mathcal{H} \rangle$ *where* $\mathcal{F}$ *is a signature,* $\mathcal{R}$ *is a set of deduction rules and* $\mathcal{H}$ *is a set of equations between terms in* $\mathrm{T}(\mathcal{F}, \mathcal{X})$*. For each deduction rule* $d : t_1, \ldots, t_n \to t \in \mathcal{R}$*, the set* $\mathrm{GI}(d)$ *denotes the set of ground instances of the rule* d modulo $\mathcal{H}$*:*

$$\mathrm{GI}(d) = \{l \to r \mid \exists \sigma,\ \textit{ground substitution on } \mathcal{F},\ l =_{\mathcal{H}} t_1\sigma, \ldots, t_n\sigma \textit{ and } r =_{\mathcal{H}} t\sigma\}$$

where $=_{\mathcal{H}}$ *is extended to sets of terms in the natural way. The set of rules* $\mathrm{GI}_{\mathcal{D}}$ *is defined as the union of the sets* $\mathrm{GI}(d)$ *for all* $d \in \mathcal{R}$*.*

Example 3. Let $\mathcal{F} = \{\langle \_, \_ \rangle, \mathrm{se}(\_, \_), \cdot, a(\_)\}$, $\mathcal{R}_{\mathcal{D}} = \{x, y \to \langle x, y \rangle ; x, y \to \mathrm{se}(x, y); \mathrm{se}(x, y), y \to x; \langle x_1, x_2 \rangle \to x_i; x, y \to x \cdot y; x \cdot y \to x; a(x) \to x; x \to a(x)\}$, $\mathcal{H} = \{x \cdot y = y \cdot x, x \cdot (y \cdot z) = (x \cdot y) \cdot z\}$. The rules associated with a are extended to any n-ary operator added to the signature representing an XML node. Then the deduction system $\mathcal{D} = \langle \mathcal{F}, \mathcal{R}_{\mathcal{D}}, \mathcal{H} \rangle$ describes the classical Dolev-Yao model with the addition of an associative-commutative operation $\cdot$ and of the rules that allow us to build multisets with $\cdot$ or to extract parts of a multiset.

Each deduction rule $l \to r$ in $\mathrm{GI}_{\mathcal{D}}$ defines an deduction relation $\to_{l \to r}$ between finite sets of terms: given two finite sets of terms E and F we define $E \to_{l \to r} F$

if and only if $l \subseteq E$ and $F = E, r$. We denote $\rightarrow_{\mathcal{D}}$ the union of the relations $\rightarrow_{l \rightarrow r}$ for all $l \rightarrow r$ such that $l \rightarrow r \in \mathrm{GI}(d)$ for some $d \in \mathcal{R}$, and by $\rightarrow_{\mathcal{D}}^{*}$ the transitive closure of $\rightarrow_{\mathcal{D}}$. We simply denote by $\rightarrow$ the relation $\rightarrow_{\mathcal{D}}$ when there is no ambiguity about the deduction system $\mathcal{D}$.

Definition 3 *A* derivation *D of length $n \geq 0$, is a sequence $E_0 \rightarrow_{\mathcal{D}} E_1 \rightarrow_{\mathcal{D}} \cdots \rightarrow_{\mathcal{D}} E_n$ where $E_0, \ldots E_n$ are finite set of ground terms such that $E_i = E_{i-1}, t_i$ for every $i \in \{1, \ldots, n\}$. The term t_n is called the* goal *of the derivation.*

Example 4. The previous deduction system has a derivation:

$$\begin{aligned}\{K_a, \langle \mathrm{se}(s, K_a), a\rangle, a, b\} &\rightarrow_{\mathcal{D}} \{K_a, \langle \mathrm{se}(s, K_a), a\rangle, \mathrm{se}(s, K_a), a, b\} \\ &\rightarrow_{\mathcal{D}} \{K_a, \langle \mathrm{se}(s, K_a), a\rangle, \mathrm{se}(s, K_a), s, a, b\}\end{aligned}$$

that shows the discovering of a secret s by an intruder that knows the encryption key K_a, identity of agents and intercepts a message $\langle \mathrm{se}(s, K_a), a\rangle$. The ground deduction rules employed are $\langle \mathrm{se}(s, K_a), a\rangle \rightarrow \mathrm{se}(s, K_a)$ and $\mathrm{se}(s, K_a), K_a \rightarrow s$

$\mathrm{Der}_{\mathcal{D}}(E) = \{t \mid \exists F \text{ s.t. } E \rightarrow_{\mathcal{D}}^{*} F \text{ and } t \in F\}$ is the set of terms derivable from E is. We write $\mathrm{Der}(E)$ instead of $\mathrm{Der}_{\mathcal{D}}(E)$ when there is no ambiguity on $\mathcal{D}$.

4.3 Symbolic derivation

Given a deduction system $\langle \mathcal{F}, \mathcal{R}, \mathcal{E}\rangle$, a role applies rules in $\mathcal{R}$ to construct the response of step i and tests equalities to check the well-formedness of a message. Hence the activity of a role can be expressed by a fixed symbolic derivation:

Definition 4 *(Symbolic Derivation) A symbolic derivation for deduction system $\langle \mathcal{F}, \mathcal{R}, \mathcal{E}\rangle$ is a tuple $(\mathcal{V}, \mathcal{S}, \mathcal{K}, \mathrm{In}, \mathrm{Out})$ where $\mathcal{V}$ is a finite sequence of variables, $(x_i)_{i \in Ind}$, indexed by a linearly ordered set $(Ind, <)$, $\mathcal{K}$ is a set of ground terms (the initial knowledge)* In, Out *are disjoint subsets of Ind and $\mathcal{S}$ is a set of equations such that for all $x_i \in \mathcal{V}$ one of the following holds:*

- $i \in \mathrm{In}$
- *There exists a ground term $t \in \mathcal{K}$ and an equation $x_i \stackrel{?}{=} t$ in $\mathcal{S}$;*
- *There exists a rule $l_1, \ldots, l_m \rightarrow r \in \mathcal{R}$ such that $\mathcal{S}$ contains the equations $x_i \stackrel{?}{=} r$ and $x_{\alpha_j} \stackrel{?}{=} l_j$ for $j \in \{1, \ldots, m\}$ with $\alpha_j < i$.*

A symbolic derivation is closed *if $\mathrm{In} = \mathrm{Out} = \emptyset$, in which case it may be simply denoted by $(\mathcal{V}, \mathcal{S}, \mathcal{K})$. A substitution σ* satisfies *a closed symbolic derivation if $\sigma \models_{\mathcal{E}} \mathcal{S}$.*

To improve readability of the examples, we replace every index i in a set of indices I by the variable x_i associated to this index, and we do not model the equations $x_i \stackrel{?}{=} t$ for $t \in \mathcal{K}$, writing t directly when x_i is needed, nonetheless keeping track of these variables in a set $\mathcal{V}_{\mathcal{K}}$. We will also employ _ to denote a variable appearing only once.

Example 5. The client A of $\mathcal{P}$ can be described by the deduction system $\mathcal{D}$ of Ex. 3. Starting from the initial knowledge $\mathcal{K}_A = \{a, b, Bremen, SK_a, PK_a\}$, the client applies the following ground instances of deduction rules. First it builds $x_1^A = order(a, a, Bremen)$ from $a, Bremen \in \mathcal{K}_A$ (ground instance of the rule $x, y, z \to order(x, y, z)$), then builds $x_2^A = h(x_1^A)$ from x_1^A and $x_3^A = \mathrm{se}(x_2^A, SK_a)$. Finally it computes and sends $x_4^A = x_3^A \cdot x_1^A$, and waits for the response x_5^A. From x_5^A it extracts two components x_6^A and x_7^A, decrypts x_6^A with SK_a and checks that the result is equal to x_2^A and finally decrypts x_7^A to obtain the accesscode. A symbolic derivation for this role is $(\mathcal{V}_{\mathcal{K}_A} \cup (x_i^A)_{1 \le i \le 9}, \mathcal{S}, \mathcal{K}_A, \{x_5^A\}, \{x_4^A\})$ with:

$$\mathcal{S} = \begin{cases} x_1^A \stackrel{?}{=} order(a, a, Bremen) & x_2^A \stackrel{?}{=} h(x_1^A) & x_3^A \stackrel{?}{=} \mathrm{se}(x_2^A, SK_a) \\ x_4^A \stackrel{?}{=} x_3^A \cdot x_1^A & x_5^A \stackrel{?}{=} x_6^A \cdot \_ & x_5^A \stackrel{?}{=} x_7^A \cdot \_ \\ x_6^A \stackrel{?}{=} \mathrm{se}(x_8^A, PK_a) & x_7^A \stackrel{?}{=} \mathrm{se}(x_9^A, PK_a) & x_8^A \stackrel{?}{=} x_2^A \end{cases}$$

To compose two symbolic derivations we identify some input variables of one derivation with some output variables of the other and vice-versa. This connection should be compatible with the variable orderings inherited from each component, as detailed in the following definition:

Definition 5 *Let $\mathcal{C}_1 = (\mathcal{V}_1, \mathcal{S}_1, \mathcal{K}_1, \mathrm{In}_1, \mathrm{Out}_1)$, $\mathcal{C}_2 = (\mathcal{V}_2, \mathcal{S}_2, \mathcal{K}_2, \mathrm{In}_2, \mathrm{Out}_2)$ be two symbolic derivations, with disjoint sets of variables and index sets $(Ind_1, <_1)$ and $(Ind_2, <_2)$ respectively. Let I_1, I_2, O_1, O_2 be subsets of $\mathrm{In}_1, \mathrm{In}_2, \mathrm{Out}_1, \mathrm{Out}_2$ respectively. and assume that there is an order-preserving bijection ϕ from $I_1 \cup I_2$ to $O_1 \cup O_2$ such that $\phi(I_1) = O_2$ and $\phi(I_2) = O_1$. A composition of two symbolic derivations along the sets I_1, O_1, I_2, O_2 is a symbolic derivation*

$$\mathcal{C} = (\mathcal{V}, \phi(\mathcal{S}_1 \cup \mathcal{S}_2), \mathcal{K}_1 \cup \mathcal{K}_2, (\mathrm{In}_1 \cup \mathrm{In}_2) \setminus (I_1 \cup I_2), (\mathrm{Out}_1 \cup \mathrm{Out}_2) \setminus (O_1 \cup O_2))$$

where ϕ is extended to a substitution on terms by setting $\phi(x_i) = x_j$ if $\phi(i) = j$, $\mathcal{V}$ is a sequence of variables indexed by $Ind = (Ind_1 \setminus I_1) \cup (Ind_2 \setminus I_2)$, ordered by a linear extension of the transitive closure of the relation:

$$<_1 \cup <_2 \cup \{(u, v) \mid v = \phi(w) \text{ and } u <_1 w \text{ or } u <_2 w\}$$

and such that the variable of index i in $\mathcal{V}$ is equal to the variable of index i in $\mathcal{V}_1$ if $i \in Ind_1$, and to the variable of index i in $\mathcal{V}_2$ if $i \in Ind_2$.

A composition of two symbolic derivations is also a symbolic derivation and we can compose an arbitrary number of symbolic derivations in the same way.

Example 6. Let us consider the symbolic derivation of the previous example, and let us assume that the variables are ordered as $y_1^A, x_1^A, y_1^B, y_2^B, x_2^A, y_2^A$. The normal execution of the protocol $\mathcal{P}$ corresponds to a composition of the two symbolic derivations of the previous example along $I_1 = \{x_2^A\}, O_1 = \{y_1^A\}, I_2 = \{x_1^B\}, O_2 = \{y_1^B\}$, *i.e.* where x_2^A (resp.x_1^B) is replaced by y_1^B (resp. y_1^A).

4.4 The formal statement of protocol insecurity

As mentioned in Section 4.1, protocol insecurity can be reduced to the executability of the protocol extended by a step revealing a secret. Since we are interested to decide insecurity for combined deduction systems we will define a slightly more general problem that is also more modular:

Ordered Satisfiability

Input: a symbolic derivation $\mathcal{C}_h$ for $\langle \mathcal{F}, \mathcal{R}, \mathcal{E} \rangle$ (protocol), a set of terms $\mathcal{K}_i$ (intruder knowledge), X the set of all variables, C the set of free constants occurring in $\mathcal{C}_h$ and a linear ordering $\prec$ on $X \cup C$.

Output: SAT iff there exists a symbolic derivation $\mathcal{C}_i = (\mathcal{V}_i, \mathcal{S}_i, \mathcal{K}_i, \text{IN}_i, \text{OUT}_i)$ for $\langle \mathcal{F}, \mathcal{R}, \mathcal{E} \rangle$, a closed composition $\mathcal{C}_a$ of $\mathcal{C}_i$ and $\mathcal{C}_h$, and a substitution σ such that σ satisfies $\mathcal{C}_a$ and: $\forall c \in C, x \prec c$ implies $c \notin \text{Const}(x\sigma)$

4.5 Comparison with pattern-matching and equational models

The pattern-matching model. In this model pattern-matching is used to extract the components of incoming messages, independently of the feasibility of the operation by the agent. For instance, if a role A must execute the receive/send sequence $\text{se}(x, y) \rightarrow \text{se}(x, K)$ and if the actual message is $\text{se}(N_B, K)$, then pattern-matching yields the relation $x = N_b$. This allows us to get rid of the algebraic properties of explicit destructors (like decryption or projection) and provides the *syntactic Dolev-Yao model*. But when the algebraic properties of other operations like the exclusive or $\oplus$ are added to the model, this may lead to meaningless protocols (e.g. not *well-defined*): a step $x \oplus y \rightarrow x$ is unrealistic, but pattern matching succeeds. In our setting symbolic derivations are well-defined and can be translated into a well-defined protocol. Therefore decidability results on well-defined protocols for some signature transfer immediately to symbolic derivations on the same signature.

The equational model. The original Dolev-Yao model [11] used explicit constructors (*e.g.* for encryption) and destructors (*e.g.* for decryption) and the corresponding equational theory. To retrieve components of a message, pattern-matching is replaced by an explicit computation using destructors. For instance, to retrieve x in $m = \text{se}(x, K)$ yields the equation $x = \text{sd}(m, K)$. The executability of the protocol is guaranteed by the satisfiability of the system of equations modulo the equational theory enriched by axioms stating that a destructor is the inverse of a constructor. This approach is sound for classical cryptographic protocols but this no longer the case for non-deterministic security protocols arising in Web services. If f_a extracts the component $a(\_)$ in a multiset m, we get that $f_a(m) = a(x_1)$ and $f_a(m) = a(x_2)$ for $m = a(x_1) \cdot a(x_2)$, hence we get $a(x_1) = a(x_2)$ which leads to some inconsistencies (all terms headed by a are equal modulo the equational theory.)

5 Plain Web services

In a first abstraction, Web services are protocols that exchange multisets of nodes. Hence we will consider first a signature that is reduced to a multiset operator, a constant (the empty multiset) and unary operators. We call *Plain Web services* protocols defined on this signature. We shall present now a procedure to decide ordered satisfiability for plain Web services. We note first that the ordered satisfiability for a deduction system in which one can enclose data in a node a (rule $x \to a(x)$) or expose the data contained in the node a (rule $a(x) \to x$) is easily decidable (by [20] and the fact that the deduction system is local and the equational theory is empty). Then for applying the combination result of next section, it remains to decide ordered satisfiability problems for the deduction system associated to the multiset operator that is employed in the construction and analysis of XML messages. We believe that the proof can be adapted for sets (with union and subset extraction). A related result for words (*i.e.* when the ordering of nodes matters) can be found in [17].

We consider the signature $\mathcal{F} = \{\cdot, 1\}$ with the following equational theory:

$$\mathcal{E} = \{x \cdot (y \cdot z) = (x \cdot y) \cdot z \ (A), x \cdot y = y \cdot x \ (C), x \cdot 1 = x \ (U)\}$$

(usually denoted ACU). Our decidability result is independent of the presence or absence of the (U) axiom. The deduction rules are:

$$\mathcal{R} = \{x \cdot y \to x, \quad x, y \to x \cdot y, \quad \to 1\}$$

This theory is regular and the set of variables of the right-hand side is included in the set of variables of the left-hand side for each rule, hence the rules are indeed deduction rules. We state now some some basic facts on $\mathcal{M} = \langle \mathcal{F}, \mathcal{R}, \mathcal{E} \rangle$.

Proposition 1 *Let E and t be respectively a set of terms and a term in normal form modulo $\mathcal{E}$. We have: (i)* $\mathrm{Const}(E) \subseteq \mathrm{Der}_{\mathcal{D}}(E)$*, (ii)* $E \subseteq \mathrm{Der}_{\mathcal{D}}(\mathrm{Const}(E))$*; (iii)* $\mathrm{Der}_{\mathcal{D}}(E) = \mathrm{Der}_{\mathcal{D}}(\mathrm{Const}(E))$*; (iv)* $t \in \mathrm{Der}_{\mathcal{D}}(E)$ *iff* $\mathrm{Const}(t) \subseteq \mathrm{Const}(E)$.
Proof. Let $c \in \mathrm{Const}(E)$ be a constant, and $e \in E$ be a term such that $c \in \mathrm{Const}(e)$. We have therefore $e = e' \cdot c$. Thus in one step we have $E \to E, c$. Since c is arbitrary, we have $\mathrm{Const}(E) \subseteq \mathrm{Der}_{\mathcal{D}}(E)$. We easily see that we also have $E \subseteq \mathrm{Der}_{\mathcal{D}}(\mathrm{Const}(E))$ and (i) follows. From this, (ii) follows by double inclusion. For (iii) we have $t \in \mathrm{Der}_{\mathcal{D}}(E)$ is equivalent to $\mathrm{Der}_{\mathcal{D}}(E) = \mathrm{Der}_{\mathcal{D}}(E, t)$, and thus by (ii) it is equivalent to $\mathrm{Der}_{\mathcal{D}}(\mathrm{Const}(E)) = \mathrm{Der}_{\mathcal{D}}(\mathrm{Const}(E, t))$. By contradiction assume there exists $c \in \mathrm{Const}(t) \setminus \mathrm{Const}(E)$. Given the constraint on constants in deduction rules we have $c \notin \mathrm{Der}_{\mathcal{D}}(\mathrm{Const}(E))$ whereas c trivially is in $\mathrm{Der}_{\mathcal{D}}(\mathrm{Const}(E, t))$, which contradicts the equality, and thus $t \in \mathrm{Der}_{\mathcal{D}}(E)$ □

The decision procedure is given by Algorithm 1. The completeness of this algorithm is easy and the correctness derives from Proposition 1.
Complexity. Step one and checking satisfiability of linear equation systems over positive integers are in NP. Step three can be performed in PTIME.

Theorem 1 *Ordered satisfiability of multiset deduction systems is in NP.*

Similar results have been obtained for pure AC-symbols under some restrictions in [18] but they do not cover our case.

Algorithm 1 Ordered satisfiability for multiset deduction systems

Input: - $\mathcal{C} = (\mathcal{V}, \mathcal{S}, \mathcal{K}, \text{IN}, \text{OUT})$;
- A finite set of ground terms $K_{\mathcal{D}}$;
- An ordering $\prec$ on $\text{Var}(\mathcal{C}) \cup \text{Const}(\mathcal{C})$.

1: For each variable x in $\mathcal{V}$ guess the set of constants $P_x = \{a_1, \ldots, a_k\}$ that may occur in the solution, where $a_i \prec x$ for $i = 1, \ldots, k$.
2: Check the satisfiability of $\mathcal{S}$ by reduction to satisfiability of a set of linear equations over $\mathbb{N}$ by setting $x = \Sigma_{c \in P_x}(\lambda_{x,c} + 1)c$ for each variable $x \in \mathcal{V}$.
3: Check for each variable $x \in \text{IN}$ that $P_x \subseteq \text{Const}(\mathcal{K}_{\mathcal{D}}) \cup \bigcup_{\substack{x' \in \text{OUT} \\ x' <_{\mathcal{V}} x}} P_{x'}$
4: If both checks are successful return SAT else FAIL

6 Protected Web services

Protected Web services are Web services that contain cryptographic operators. Many results have been obtained for proving security of cryptographic protocols with various equational theories (xor, ...), and we want to be able to use these results in our framework. [5]. In this last section we present a combination algorithm that allows us to reuse previous results, *either from the equational model or the pattern-matching model*. In particular it provides the first modularity result covering the pattern-matching model for cryptographic protocols.

From now on, we shall assume that $\mathcal{F}$ is the disjoint union of two signatures $\mathcal{F}_1$ and $\mathcal{F}_2$, and we assume that $\mathcal{E}_1$ (resp. $\mathcal{E}_2$) is a consistent equational theory on $\mathcal{F}_1$ (resp. $\mathcal{F}_2$). We recall that C is an infinite set of free constants, that $\mathcal{X}$ is an infinite set of variables and that $\text{T}(\mathcal{F}_1, \mathcal{X})$ (resp. $\text{T}(\mathcal{F}_2, \mathcal{X})$) denotes the set of terms on $\mathcal{F}_1 \cup \text{C}$ (resp. $\mathcal{F}_2 \cup \text{C}$). The ordering $<$ is a simplification ordering on $\text{T}(\mathcal{F}, \mathcal{X})$ total on $\text{T}(\mathcal{F})$ and the constant $\text{c}_{\min}$ is the minimal element for $<$.

A term t in $\text{T}(\mathcal{F}_1, \mathcal{X})$ (resp. in $\text{T}(\mathcal{F}_2, \mathcal{X})$) is called a *pure* 1-term (resp. a pure 2-term). The term s is *alien* to the term u if the top operators of s and u are not both in $\mathcal{F}_1$ or both in $\mathcal{F}_2$.

Definition 6 *Let t be a term in $\text{T}(\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{X})$. The set of its* factors *is denoted* $\text{Factors}(t)$ *and is the set of maximal syntactic strict subterms of t that are either alien to t or atoms. The set of its* subterm values *is denoted by* $\text{Sub}(t)$ *and is defined recursively by* $\text{Sub}(t) = \{t\} \cup \bigcup_{u \in \text{Factors}(t)} \text{Sub}(u)$.

For a set of terms E, the set $\text{Sub}(E)$ is the union of the subterm values of the elements of E. A set of equations $\mathcal{S}$ is called *homogeneous* iff it contains only pure equations. We denote by $t\delta_s$ the term obtained by replacing all *subterm* occurrences of s in t by $\text{c}_{\min}$.

Lemma 1. *Let $l \to r$ be a rule in $\text{GI}(d)$ with $d : t_1, \ldots, t_n \to t$ and $s \notin \text{C}_{\text{spe}}$ alien to some t_i or t. Then $(l\delta_s)\downarrow \to (r\delta_s)\downarrow$ is also a rule in $\text{GI}(d)$.*

[5] The freshness of nonces will be handled as for protocols and not discussed here.

PROOF. Let σ be a ground normal substitution such that $(t_1\sigma, \ldots, t_n\sigma)\downarrow = l$ and $(t\sigma)\downarrow = r$. Let us prove that for $u \in \{t_1, \ldots, t_n, t\}$ one has $((u\sigma)\downarrow\delta_s)\downarrow = (u(\sigma\delta_s))\downarrow$. This suffices to find a ground instance $(l\delta_s)\downarrow \to (r\delta_s)\downarrow$ in $\mathrm{GI}(d)$. If u is a variable then σ in normal form implies $(u\sigma)\downarrow = u\sigma$ and thus $((u\sigma)\delta_s)\downarrow = ((u\sigma)\downarrow\delta_s)\downarrow$. Otherwise, and since the rule is pure, the assumption implies that s is alien to u. Since σ is a normal substitution, and since for $u \in \{t_1, \ldots, t_n, t\}$ the term u is pure, the factors of $u\sigma$ are in normal form.

These facts and Lemma 10 from [10] imply that for all $u \in \{t_1, \ldots, t_n, t\}$ one has $((u\sigma)\downarrow\delta_s)\downarrow = ((u\sigma)\delta_s)\downarrow$. Since s is alien to the term u we also have $(u\sigma)\delta_s = u(\sigma\delta_s)$. By applying a bottom-up normalisation we have $u(\sigma\delta_s) =_E u(\sigma\delta_s)\downarrow$. Putting together these equalities we obtain a substitution $\sigma' = (\sigma\delta_s)\downarrow$ such that the rule d instantiated by σ' is $(l\delta_s)\downarrow \to (r\delta_s)\downarrow$. □

Let $\langle \mathcal{F}_1, S_1, \mathcal{E}_1 \rangle$ and $\langle \mathcal{F}_2, S_2, \mathcal{E}_2 \rangle$ be two deduction systems on disjoint signatures. Our combination algorithm relies on the following lemma stating that a satisfiable closed symbolic derivation for the union of $\langle \mathcal{F}_1, S_1, \mathcal{E}_1 \rangle$ and $\langle \mathcal{F}_2, S_2, \mathcal{E}_2 \rangle$ can be split into satisfiable symbolic derivations on $\mathcal{F}_1$ and $\mathcal{F}_2$ respectively. The abstraction of a term t in a signature $\mathcal{F}_i$, denoted $\mathrm{Abs}_i(t)$, consists in replacing the maximal subterms of t with root in $\mathcal{F}_j, j \neq i$ by new constants.

The abstraction $\mathrm{Abs}_i(\sigma)$ of a substitution σ in $\mathcal{F}_i$ is the substitution such that for all x in the support of σ, $x\mathrm{Abs}_i(\sigma)$ is the abstraction of $x\sigma$ in $\mathcal{F}_i$. In the statement of next lemma, $\mathrm{Sig}(t)$ designates the signature to which the top symbol of the term t belongs. From a system of equations $\mathcal{S}$, one notes that one can derive, by introducing new variables, an equisatisfiable homogeneous set of equations $\mathcal{S}_1 \cup \mathcal{S}_2$ such that terms in $\mathcal{S}_i$ are pure $\mathcal{F}_i$-terms for $i = 1, 2$.

The proof idea is to "abstract" a solution σ for a symbolic derivation $\mathcal{C}$ in deduction system $\langle \mathcal{F}_1 \cup \mathcal{F}_2, S_1 \cup S_2, \mathcal{E}_1 \cup \mathcal{E}_2 \rangle$ and to apply iteratively Lemma 1 in order to show for $i = 1, 2$ that $\mathrm{Abs}_i(\sigma)$ is a solution of the symbolic derivation in the deduction system $\langle \mathcal{F}_i, S_i, \mathcal{E}_i \rangle$.

Lemma 2. *Let $\mathcal{C} = (\mathcal{V}, \mathcal{S}, \mathcal{K})$ be a closed derivation satisfied by a normal substitution σ for the deduction system $\langle \mathcal{F}_1 \cup \mathcal{F}_2, S_1 \cup S_2, \mathcal{E}_1 \cup \mathcal{E}_2 \rangle$. For $i = 1, 2$ let $\mathcal{S}_i$ denote the purification of $\mathcal{S}$* [6] *and:*

$$\mathcal{K}_i = \{\mathrm{Abs}_i(t) \in \mathrm{Sub}(\mathcal{K}) \mid \mathrm{Sig}(t) = i \text{ and } \exists x \in \mathcal{V},\ x\sigma = t\} \cup \{x \in \mathcal{V} \mid \mathrm{Sig}(x\sigma) \neq i\}$$

Then two closed symbolic derivations $\mathcal{C}_i = (\mathcal{V}, \mathcal{S}_i, \mathcal{K}_i)$ (for $i = 1, 2$) can be computed such that they are satisfied by $\mathrm{Abs}_i(\sigma)$ for intruder $\langle \mathcal{F}_i, S_i, \mathcal{E}_i \rangle$ respectively.

Algorithm 2 reduces satisfiability of a $\mathcal{D}$-symbolic derivation $\mathcal{C}$ to satisfiability of $\mathcal{D}_1$ and $\mathcal{D}_2$, symbolic derivations augmented with ordering constraints on the variables of $\mathcal{C}$ and the ordering constraints between variables and constants of $\mathcal{C}$. A *partial* symbolic derivation denotes a symbolic derivation for which variables in $\mathcal{V}$ whose index is not in IN may have no associated equation in $\mathcal{S}$. This leads to the following theorem:

[6] as in unification

Theorem 2 *If the ordered satisfiability problem for closed symbolic derivation is decidable for two deduction systems $\langle \mathcal{F}_1, S_1, \mathcal{E}_1 \rangle$ and $\langle \mathcal{F}_2, S_2, \mathcal{E}_2 \rangle$ with disjoint signatures $\mathcal{F}_1$ and $\mathcal{F}_2$ then the ordered satisfiability problem for closed symbolic derivation is decidable for the deduction system $\langle \mathcal{F}_1 \cup \mathcal{F}_2, S_1 \cup S_2, \mathcal{E}_1 \cup \mathcal{E}_2 \rangle$.*

Algorithm 2 Combination Algorithm $\mathbf{Solve}_{\mathcal{D}}(\mathcal{C}, \prec)$

Input: $\mathcal{C} = (\mathcal{V}, \mathcal{S}, \mathcal{K}, \text{IN}, \text{OUT})$ where $\mathcal{S}$ homogeneous;
A finite set of ground terms $K_{\mathcal{D}}$;
A linear ordering $\prec$ on $\text{Var}(\mathcal{C}) \cup \text{Const}(\mathcal{C})$.

1: Choose a partial symbolic derivation $\mathcal{C}_{\mathcal{D}} = (\mathcal{V}_{\mathcal{D}}, \mathcal{S}_{\mathcal{D}}, \mathcal{K}_{\mathcal{D}}, \text{IN}_{\mathcal{D}}, \text{OUT}_{\mathcal{D}})$ such that:

- $|\mathcal{V}_{\mathcal{D}}| \leq |\text{Sub}(\mathcal{C})| + |\text{IN}| + |\text{OUT}| + |\mathcal{K}_{\mathcal{D}}|$
- $|\text{IN}_{\mathcal{D}}| = |\text{OUT}|$ and $|\text{OUT}_{\mathcal{D}}| = |\text{IN}|$
- A composition $\mathcal{C} \circ \mathcal{C}_{\mathcal{D}}(\mathcal{K}_{\mathcal{D}})$ is defined
- $\mathcal{S}_{\mathcal{D}}$ is homogeneous, contains equations $x \stackrel{?}{=} t$ with $t \in \text{Sub}(\mathcal{C}) \cup \mathcal{V}_{\mathcal{D}}$ and $x \in X$, a set of new variables with $|X| \leq |\text{Sub}(\mathcal{C}) \cup \mathcal{V}_{\mathcal{D}}|$, and defines an equivalence relation on $\text{Sub}(\mathcal{C}) \cup \mathcal{V}_{\mathcal{D}}$

2: Choose a linear ordering $\prec_X$ on variables in $X \cup \text{Const}(\mathcal{C}) \cup \text{Var}(\mathcal{C})$ extending $\prec$. Let X_1 and X_2 be two disjoint subsets of X
3: Form the closed partial symbolic derivation $\mathcal{C}' = \mathcal{C} \circ \mathcal{C}_{\mathcal{D}}$, and purify it into two pure symbolic derivations $\mathcal{C}_1$ and $\mathcal{C}_2$, where variables of X_1 (resp. X_2) are considered as constants in $\mathcal{C}_2$ (resp. $\mathcal{C}_1$).
4: If $\mathbf{Solve}_{\mathcal{D}_1}(\mathcal{C}_1, \prec_X)$ and $\mathbf{Solve}_{\mathcal{D}_2}(\mathcal{C}_2, \prec_X)$ return SAT else FAIL

The proof of completeness of Alg. 2 is intricate, and relies on a locality result for deductions and on a pumping lemma for solutions. Its correctness relies on the ordering $\prec_X$ to construct a solution σ from two partial solutions σ_1 and σ_2.

7 Conclusion

Web service XML messages are often vulnerable to rewriting attacks since the associated message format and processing model are quite tolerant to inclusion of new elements. We have developed a verification procedure that accounts for this and moreover can be extended with most existing protocol analysis procedures for algebraic operators. The combination algorithm applies in particular to encryption/decryption operators [19], list with associative concatenation [17], XOR [20], abelian groups [16]. Our framework allows us to describe specific implementation of XML/XPath library and we aim at extending this work to take the various security standards for Web services into account. We also plan to implement it in the AVISPA platform [21].

References

1. Bhargavan, K., Fournet, C., Gordon, A.D., Pucella, R.: Tulafale: A security tool for web services. In: Formal Methods for Components and Objects. Volume 3188 of Lecture Notes in Computer Science., Springer (2003) 197–222

2. Cervesato, I., Durgin, N.A., Lincoln, P., Mitchell, J.C., Scedrov, A.: A meta-notation for protocol analysis. In: CSFW. (1999) 55–69
3. Chevalier, Y., Rusinowitch, M.: Combining intruder theories. In: Proc. of ICALP. Volume 3580 of Lecture Notes in Computer Science., Springer (2005) 639–651
4. Basin, D.A., Mödersheim, S., Viganò, L.: Algebraic intruder deductions. In Sutcliffe, G., Voronkov, A., eds.: LPAR. Volume 3835 of Lecture Notes in Computer Science., Springer (2005) 549–564
5. Comon-Lundh, H., Delaune, S.: The finite variant property: How to get rid of some algebraic properties. In: Proceedings of RTA'05). Lecture Notes in Computer Science, Nara, Japan, Springer (2005)
6. Goubault-Larrecq, J., Roger, M., Verma, K.N.: Abstraction and resolution modulo ac: How to verify diffie-hellman-like protocols automatically. J. Log. Algebr. Program. **64**(2) (2005) 219–251
7. Schmidt-Schauß, M.: Unification in a combination of arbitrary disjoint equational theories. J. Symb. Comput. **8**(1/2) (1989) 51–99
8. Baader, F., Schulz, K.U.: Unification in the union of disjoint equational theories. combining decision procedures. J. Symb. Comput. **21**(2) (1996) 211–243
9. Thayer, F.J., Herzog, J.C., Guttman, J.D.: Strand spaces: Proving security protocols correct. Journal of Computer Security **7**(1) (1999)
10. Chevalier, Y., Lugiez, D., Rusinowitch, M.: Towards an Automatic Analysis of Web Service Security. Technical report, INRIA (2007) `http://www.inria.fr/rrrt/liste-2007.html`.
11. Dolev, D., Yao, A.: On the Security of Public-Key Protocols. IEEE Transactions on Information Theory **2**(29) (1983)
12. Weidenbach, C.: Towards an automatic analysis of security protocols in first-order logic. In: 16th International Conference on Automated Deduction. Volume 1632 of Lecture Notes in Computer Science., Springer (1999) 314–328
13. Dershowitz, N., Jouannaud, J.P.: Rewrite systems. In: Handbook of Theoretical Computer Science, Volume B. Elsevier (1990) 243–320
14. Baader, F., Nipkow, T.: Term Rewriting and All That. Cambridge University Press (1998)
15. Bonacina, M.P., Hsiang, J.: Towards a foundation of completion procedures as semidecision procedures. Theor. Comput. Sci. **146**(1&2) (1995) 199–242
16. Millen, J., Shmatikov, V.: Symbolic protocol analysis with an abelian group operator or Diffie-Hellman exponentiation. Journal of Computer Security (2005)
17. Chevalier, Y., Kourjieh, M.: A symbolic intruder model for hash-collision attacks. In: 11th Annual Asian Computing Science Conference. Lecture Notes in Computer Science, Springer (2006) `ftp://ftp.irit.fr/IRIT/LILAC/main.pdf`.
18. Bursuc, S., Comon-Lundh, H., Delaune, S.: Associative-commutative deducibility constraints. In Thomas, W., Weil, P., eds.: Proceedings of the 24th Annual Symposium on Theoretical Aspects of Computer Science (STACS'07). Volume 4393 of Lecture Notes in Computer Science., Aachen, Germany, Springer (2007) 634–645
19. Rusinowitch, M., Turuani, M.: Protocol insecurity with finite number of sessions is NP-complete. In: Proc.14th IEEE Computer Security Foundations Workshop, Cape Breton, Nova Scotia (2001)
20. Chevalier, Y., Kuesters, R., Rusinowitch, M., Turuani, M.: An NP Decision Procedure for Protocol Insecurity with XOR. In: Proceedings of the Logic In Computer Science Conference, LICS'03. (2003)
21. Armando, A., et al.: The AVISPA tool for the automated validation of internet security protocols and applications. In Etessami, K., Rajamani, S.K., eds.: CAV. Volume 3576 of Lecture Notes in Computer Science., Springer (2005) 281–285