Deciding recognizability under Dolev-Yao intruder model

Zhiwei Li and Weichao Wang

Department of SIS
UNC Charlotte, NC 28262, USA
email: {zli19,weichaowang}@uncc.edu

Abstract. The importance of reasoning about recognizability has recently been stressed in finding type flaw attacks, in which a protocol message may be forged from another message. However, the problem of deciding recognizability has never been fully exploited. To fill this gap, we present a terminating procedure to decide recognizability under the standard Dolev-Yao model. By incorporating the proposed procedure with Athena, a well-know security protocol verifier, our experiments succeed in finding potential type flaw attacks.

Keywords Security, Verification, Formal Methods

1 Introduction

Many security protocols are vulnerable to type flaw attacks, in which a protocol message may be subsequently forged from another message. Let us consider the Otway-Rees protocol [21]:

$$A \rightarrow B : M, A, B, \{N_A, M, A, B\}_{K_{AS}}$$
$$B \rightarrow S : M, A, B, \{N_A, M, A, B\}_{K_{AS}}, \{N_B, M, A, B\}_{K_{BS}}$$
$$S \rightarrow B : M, \{N_A, K_{AB}\}_{K_{AS}}, \{N_B, K_{AB}\}_{K_{BS}}$$
$$B \rightarrow A : M, \{N_A, K_{AB}\}_{K_{AS}}$$

After executing the first three messages, principal A is expecting a K_{AB}, which is a symmetric key shared between A and B, from the trusted third party S. The shared key K_{AB} is dynamically generated by S and A does not have any prior knowledge about the bit string. Therefore, any message of the form $M, \{N_A, t\}_{K_{AS}}$ would be accepted by A, as long as the bit string length of t equals to that of K_{AB}. Thus, an intruder can easily replay the message $\{N_A, M, A, B\}_{K_{AS}}$ to A and then A would use M, A, B as the secret if the length satisfies the requirement.

Li and Wang [12] show that reasoning about the principal's ability to verify messages would help find potential type flaw attacks. To formalize this, the notion of *recognizability* is proposed. That is, given a bit string, though the principal may not necessary know the message, he can verify whether or not it is the bit string representing the intended message. For example, in the above

Otway-Rees protocol, A can not "recognize" message K_{AB} and thus is vulnerable to type flaw attack. In general, the notion of "recognizability" offers a new perspective on the study of type flaw attacks. However, in [12] neither automatic procedure nor proof is given to decide recogonizability in security protocol.

In this paper, we address this problem under standard Dolev-Yao adversary model [9], which is described by a subterm equational theory [1]. The major contribution of this paper is a terminating procedure for deciding recognizability under Dolev-Yao model. Our construction is general enough to account for dynamic interactions between protocol participants. To the best of our knowledge, this is the first decision procedure for recogonizability in security protocols analysis.

Additional Related Work Various approaches have been proposed to defend against type flaw attacks. Heather et al. [11] propose a tagging scheme to prevent type-flaw attacks, in which tags are used to label each field of a message with its intended type. However, since tag information can potentially be confused with data [17], a tagged protocol may give rise to more intricate attacks. More importantly, the question of whether an existing protocol (without any change) is vulnerable to type-flaw attack is not answered.

Catherine Meadows [18] develops a formal model of types to characterize one's capability to verify messages. Without exploring the intuitive idea behind, the procedure of verifying the locality of types could be rather complicated. In [14, 13], Z specification language is employed to model ambiguous messages. The approach based on Z specification language cannot be directly applied to existing protocol analysis tools in a straight-forward way.

The remainder of this paper is organized as follows. Section 2 gives some background material on term rewriting systems and deducibility. In Section 3, we review the definition of recognizability, and in Section 4 we present a terminating procedure to decide recognizability under Dolev-Yao intruder model. We discuss several practical issues relevant to its use in analyzing security protocol in Section 5. Finally, Section 6 concludes the paper.

2 Preliminaries

Term Algebra We write $t_1 =_E t_2$ when equation $t_1 = t_2$ is a logical consequence of equational theory E. To avoid confusion, syntactic equality of two terms t_1 and t_2 will be denoted by $t_1 =_s t_2$. We use $fv(t)$ and $fv(T)$ to denote the sets of variables that occur in term t and term set T, respectively. A term is *ground* if $fv(t) = \emptyset$.

In this paper, we discriminate two types of function symbols, namely, *public* and *private* function symbols, set of which are denoted by $\mathcal{F}^+$ and $\mathcal{F}^-$, respectively. Public functions are used to describe operations that can be freely performed by a principal, such as encryption and decryption. We need to point out that decryption operation is conducted by calling a decryption algorithm even without proper decryption key and can be applied to any message. It can

be different from the decryption of a ciphertext. Private functions are used to constrain the relation between terms. For example, public key and private key are described by a private function in this paper.

We say s is a *subterm* of t, written $s \subseteq t$, if either $s =_s t$ or $t =_s f(t_1, \cdots, t_n)$ and s is a subterm of t_i for some i. We also write $s \subset t$ if $s \subseteq t$ and $s \neq_s t$. The *size* of a term t is defined as

$$\|t\| \stackrel{def}{=} \begin{cases} 1 & \text{if } t \text{ is a variable or constant symbol} \\ 1 + \sum_{i=1}^{n} \|t_i\| & \text{if } t = f(t_1, \cdots, t_n) \end{cases}$$

For convenience, we will use $ff(t)$ and $sub(t)$, respectively, to denote the outmost function symbol of t and the immediate subterm set of a term t[1]. A *context* C is a term with exactly a "hole" $\Box$. Then the term $C[t]$ is C except $\Box$ is replaced by t. Abusing notation slightly, we refer to $t[u \mapsto v]$ as t except that *every* occurrence of u in t is replaced by v.

A *substitution* is a finite tuple $[t_1/x_1, ..., t_n/x_n]$ mapping from variables x_i to terms t_i. The *domain* and *range* of a substitution σ are defined by $Dom(\sigma) \stackrel{def}{=} \{x | x\sigma \neq_s x\}$ and $Ran(\sigma) \stackrel{def}{=} \bigcup_{x \in Dom(\sigma)} \{x\sigma\}$, respectively. A substitution σ is *ground* if $fv(Ran(\sigma)) = \emptyset$. We write $\sigma = \theta$ if $Dom(\sigma) = Dom(\theta)$ and $x\sigma =_s x\theta$ for all x. We say that σ is more general than θ, notation $\sigma \preceq \theta$, if $\theta = \sigma\eta$ for some substitution η. We write $mgu(s, t)$ for the most general unifier of s and t.

Let X be a set of variables. Then, we define $[\sigma]_X$ as a substitution σ' such that $Dom(\sigma') = X$ and $x\sigma' = x\sigma$ for all $x \in X$; we also write $[\sigma = \theta]_X$ to indicate that $x\sigma =_s x\theta$ for all $x \in X$. We denote by ϕ a null substitution (i.e., $[]$).

Term Rewriting Systems As is commonplace, the reflexive transitive closure of a binary relation $\rightarrow$ is denoted by $\rightarrow^*$. A *term rewriting system* R consists of a set of rules, $l \rightarrow r$. A term rewriting system R defines a *term rewriting relation* $\rightarrow_R$ in a standard way: $C[l\sigma] \rightarrow_R C[r\sigma]$ where C is a context, $l \rightarrow r \in R$, and σ is a substitution. Relation $\leftrightarrow_R$ is defined by $s \leftrightarrow_R t$ iff $s \rightarrow_R t$ or $t \rightarrow_R s$.

We say that an element p is *reducible* for $\rightarrow$ if there is an element q such that $p \rightarrow q$ and *irreducible* otherwise. We write $p \rightarrow^! q$ if $p \rightarrow^* q$ and q is irreducible. If $s \rightarrow^!_R t$, then t is called an *R-normal form* of s. $\rightarrow_R$ is *terminating* if there exists no infinite derivation $t_0 \rightarrow_R t_1 \rightarrow_R \cdots$ and $\rightarrow_R$ is *confluent* if there is a term t such that $t_1 \rightarrow^*_R t$ and $t_2 \rightarrow^*_R t$ whenever $t_0 \rightarrow^*_R t_1$ and $t_0 \rightarrow^*_R t_2$. A term rewriting system R is *convergent* if $\rightarrow_R$ is terminating and confluent. Given an equational theory E, we define term rewriting system $R_E \stackrel{def}{=} \{l \rightarrow r | l = r \in E\}$. We will often use the term "convergent equational theory E" to mean that $\rightarrow_{R_E}$ is convergent. When $\rightarrow_{R_E}$ is convergent, $t_1 =_E t_2$ iff t_1 and t_2 have the same R_E-normal form[3, 8].

A substitution σ is *R_E-normal* if all terms in $Ran(\sigma)$ are R_E-normal. We write $\sigma_1 =_E \sigma_2$ to indicate that $Dom(\sigma_1) = Dom(\sigma_2)$ and $x\sigma_1 =_E x\sigma_2$ for

[1] We let $sub(t) = \{t\}$ and $ff(t) = \emptyset$ if $\|t\| = 1$.

all $x \in Dom(\sigma_1)$. In the rest of this paper, all substitutions are assumed to be R_E-normal.

Modeling Standard Adversaries The most straightforward way is to model knowledge in terms of message deducibility[9, 10, 15]. That is, given an equational system E and some messages T one might be able to compute another message t from T under equational theory E. Formally,

$\boxed{\vdash^{(n)}}$

$$\text{(R1)} \quad \frac{t \in T}{T \vdash^{(1)} t}$$

$$\text{(R2)} \quad \frac{T \vdash^{(n_1)} t_1 \cdots T \vdash^{(n_k)} t_k}{T \vdash^{(1+\max\limits_{1\le i\le k} n_i)} f(t_1, \cdots, t_k)} \; f \in \mathcal{F}^+$$

$\boxed{\vdash_E^{(n)}}$

$$\text{(R3)} \quad \frac{T \vdash^{(n)} s \quad s =_E t}{T \vdash_E^{(n+1)} t}$$

We say that t can be deduced from T, written $T \vdash t$, if $T \vdash^{(n)} t$ for some n. Likewise, t is deduced from T under E, notation $T \vdash_E t$, if $T \vdash_E^{(n)} t$ for some n. We write $T \vdash_E S$ to mean that $T \vdash_E s$ for every $s \in S$ and say that S and T are *equivalent* (under E), denoted as $S \equiv_E T$, if $S \vdash_E T$ and $T \vdash_E S$. As we can see, both $\vdash$ and $\vdash_E$ are closed under substitution. The following equational theory E_{dy} is used to model the standard Dolev-Yao intruder.

Public function symbols	$\mathtt{pair}, \mathtt{fst}, \mathtt{snd}, \mathtt{enc}, \mathtt{dec}$
Private function symbols	$\mathtt{kp}$
Equations E_{dy}	$\mathtt{fst}(\mathtt{pair}(x, y)) = x$
	$\mathtt{snd}(\mathtt{pair}(x, y)) = y$
	$\mathtt{dec}(\mathtt{enc}(x, y), \mathtt{kp}(y)) = x$
	$\mathtt{dec}(\mathtt{enc}(x, \mathtt{kp}(y)), y) = x$

Fig. 1. Equational Theory E_{dy} modeling the standard Dolev-Yao intruder.

We define $\mathcal{F}^\diamond = \{ff(l) | l = r \in E\}$ and say f is an *irregular* function symbol if $f \in \mathcal{F}^\diamond$. A term t is *regular* (or *semi-regular*) if t (or each strict subterm of t) contains no irregular function symbols. Similarly, a substitution σ is *regular* if $Ran(\sigma)$ contains no irregular function symbols. Clearly, if t is a regular term, then it is also an R_E-normal term; likewise, if σ is a regular substitution, then it is also an R_E-normal substitution. The notion of regularity gives an easier way to determine R_E-normality. The following definition sets the stage for our study of recognizability under Dolev-Yao adversaries.

Definition 1 (Regular Subterm Equational Theory). *E is a* subterm equational theory *if for every equation $l = r \in E$*

- *$r \subset l$; and*

– *all terms in* $sub(l) \cup \{r\}$ *are regular.*

It can be shown that E_{dy} is a convergent regular subterm equational theory. To reduce notational clutter, we will often use $\{s\}_t$ and $s \cdot t$ as shorthands for $\mathtt{enc}(s,t)$ and $\mathtt{pair}(s,t)$, respectively.

3 Recognizability

Definition 2 (Operational Equivalence). *Let T be a term set and σ_1 and σ_2 be two ground substitutions such that $fv(T) \subseteq Dom(\sigma_1) = Dom(\sigma_2)$. They are equivalent w.r.t. equational theory E and term set T, written $\sigma_1 \approx_{E,T} \sigma_2$, if for all terms u and v such that $T \vdash \{u, v\}$ we have $u\sigma_1 =_E v\sigma_1 \Leftrightarrow u\sigma_2 =_E v\sigma_2$.*

Example 1. The initial knowledge of Alice is represented by a ground term set $T_0 = \{N_A, K_B^-\}$. She wants to verify whether or not an incoming message is $\{N_A \cdot A\}_{K_B^+}$.

We use $T = T_0 \cup \{x\}$ to depict the new knowledge of Alice, in which x is a free variable employed to indicate the unknown incoming message. We also use a ground substitution $\sigma_0 = [\{N_A \cdot A\}_{K_B^+}/x]$ to describe the expected content of x, and substitution σ to indicate possible content of the incoming message. We often informally use *expected substitution* and *possible substitution* to refer to σ_0 and σ, respectively.

Now, we let $\sigma' = [\{N_A \cdot \{N_B\}_{K_A^+}\}_{K_B^+}/x]$, $u =_s \mathtt{lpair}(\mathtt{dec}(x, K_B^-))$, and $v =_s N_A$. Clearly, $T \vdash \{u, v\}$. In Dolev-Yao model, we have

$$u\sigma_0 =_s \mathtt{lpair}(\mathtt{dec}(\{N_A \cdot A\}_{K_B^+}, K_B^-)) \rightarrow_{R_{E_{dy}}} \mathtt{lpair}(N_A \cdot A) =_s N_A =_s v\sigma_0$$

$$u\sigma' =_s \mathtt{lpair}(\mathtt{dec}(\{N_A \cdot \{N_B\}_{K_A^+}\}_{K_B^+}, K_B^-)) \rightarrow_{R_{E_{dy}}} \mathtt{lpair}(N_A \cdot \{N_B\}_{K_A^+}) =_s N_A =_s v\sigma'$$

So, $u\sigma_0 =_{E_{dy}} v\sigma_0$ and $u\sigma' =_{E_{dy}} v\sigma'$. It can be shown that for any u and v such that $T \vdash \{u, v\}$ we have $u\sigma_0 =_{E_{dy}} v\sigma_0 \Leftrightarrow u\sigma' =_{E_{dy}} v\sigma'$. That is, $\sigma_0 \approx_{E_{dy},T} \sigma'$.

The concept of operational equivalence captures the fact that a principal cannot distinguish two messages by playing them with messages from the principal's knowledge. To establish recognizability, it is required to ensure that all possible substitutions are identical to the expected substitution.

Definition 3 (Recognizability [12]). *Let T be a ground term set, t be a ground term, and $\sigma_0 = [t/x]$. We say that t is* recognizable *by T under equational theory E, and write $T \rhd_E t$, if the following condition holds:*

$$\sigma \approx_{E,T\cup\{x\}} \sigma_0 \textit{ iff } \sigma =_E \sigma_0$$

In the above definition, term set T is restricted to be a ground term set. We stress that this is not a hurdle. Let us explain. First, in this paper the intended purpose of variables is to indicate that the bit string representation of that message or piece of message is not determined. Since t denotes the expected

message that x represents, t should be ground in a sense that the expected message should always be determined beforehand; Second, T is often used to model a principal's initial knowledge, that is all terms he explicitly knows before initiating a protocol, such as principal names and the keys that are known initially to the principal; Third, if we do need to deal with T containing variables, as we will see, this can be done by extending the domain of substitution σ_0 in a way that $T\sigma_0$ is guaranteed to be ground. This actually provides an easy way to verify multiple messages at a time. We postpone our further discussion until Section 5.

Example 2. Consider a ground term set $T_0 = \{K_B^+, \{N_A\}_{K_B^+}\}$. Let $\sigma_0 = [K_B^-/x]$, $u =_s \mathtt{enc}(\mathtt{dec}(\{N_A\}_{K_B^+}, x), K_B^+)$, and $v =_s \{N_A\}_{K_B^+}$.
Clearly, $T_0 \cup \{x\} \vdash \{u, v\}$. We see that

$$u\sigma_0 =_s \mathtt{enc}(\mathtt{dec}(\{N_A\}_{K_B^+}, K_B^-), K_B^+) =_{E_{dy}} \mathtt{enc}(N_A, K_B^+) =_s v =_s v\sigma_0$$

So, $u\sigma_0 =_{E_{dy}} v\sigma_0$. Assume that $\sigma_0' \approx_{E_{dy}, T_0 \cup \{x\}} \sigma_0$. Then,

$$\mathtt{enc}(\mathtt{dec}(\{N_A\}_{K_B^+}, x\sigma_0'), K_B^+) =_{E_{dy}} v\sigma_0' =_s \{N_A\}_{K_B^+}$$

Now, it is not hard to see that $\mathtt{dec}(\{N_A\}_{K_B^+}, x\sigma_0') =_{E_{dy}} N_A$ and thus $x\sigma_0' =_s K_B^-$. Note that $Dom(\sigma_0') = Dom(\sigma_0) = \{x\}$. Finally, we get $\sigma_0' = [K_B^-/x] = \sigma_0$. Similarly, if we let $\sigma_1 = [N_A/x]$, it can be shown that $\sigma_1 \approx_{E_{dy}, T_0 \cup \{x\}} \sigma_1'$ iff $\sigma_1' = \sigma_1$.

In the above example, although neither N_A nor K_B^- is explicitly known (i.e., $T_0 \nvdash_{E_{dy}} \{N_A, K_B^-\}$), one can still recognize them, because for any $\sigma_0' \approx_{E, T_0 \cup \{x\}} \sigma_0$ and $\sigma_1' \approx_{E, T_0 \cup \{x\}} \sigma_1$ we get $\sigma_0' = \sigma_0$ and $\sigma_1' = \sigma_1$, respectively.

4 Deciding Recognizability

Our strategy of deciding recognizability is essentially a constraint solving procedure: Step 1 (operational equivalence) incorporates "constraints"[2] imposed by the intended message; a new substitution is obtained in Step 2 (recognizability) by solving those "constraints". Intuitively, "constraint" is the condition imposed on terms such that possible substitutions would be more restricted and thus a less general substitution is obtained. For example $\mathtt{lpair}(x\sigma) \rightarrow_{R_E} N_A$ is a "constraint", which holds only if $[N_A \cdot y/x] \preceq \sigma$.

This is reminiscent of the constraint-solving approach, first proposed by Millen and Shmatikov [19], used in security protocols analysis, in which verifying security properties can be reduced to solving symbolic constraints [5, 6, 20, 7].

[2] In further text we use "constraint"(with quotes) informally to mean a common sense fact of being restricted and constraint (without quotes) to mean either a type-I constraint or a type-II constraint, as in Definition 5.

The two key ingredients of our approach are the notions of constraint and reduction, which, as we will see, allow us to consider only a rather reduced term space. To formalize this, we will need the following definition.

Definition 4 (Markup Term Set). *A* markup term set*, notated as* $\boldsymbol{T}$*, is a triple* $\langle T, \eta, \sigma \rangle$*, where* η *is a substitution,* σ *is a ground substitution, and* $Dom(\eta) \subseteq fv(T)$*. Given an equational theory* E*, we call a markup term set* $\langle T, \eta, \sigma \rangle$ well-formed *if it obeys the the following conditions*

- *all terms in* T *are regular;*
- $T\eta\sigma$ *is a ground term set; and*
- *both* σ *and* η *are regular.*

Intuitively, σ is the expected substitution and η represents the partially solved variables. In well-designed protocols, messages should be regular. For example, protocol participants would not expect a messages like $\mathtt{enc}(A, \mathtt{dec}(B, C))$. The well-formed markup term sets precisely capture this fact.

4.1 Constraint

Definition 5 (Constraint). *Let* $\boldsymbol{T} = \langle T, \eta, \sigma \rangle$ *be a markup term set. Suppose that* $T\eta \vdash \{u, v\}$*. Then,*

- (u, v) *is a* type-I constraint *of* $\boldsymbol{T}$*, if both* u *and* v *are regular,* $u \in T\eta$*,* $u\sigma =_s v\sigma$*, and* $u \neq_s v$*;*
- (u, v) *is a* type-II constraint *of* $\boldsymbol{T}$*, if* u *is* R_E*-normal and semi-regular,* v *is regular,* $v \notin \mathcal{X}$*,* $u \neq_E v$*, and* $u\sigma \rightarrow_{R_E} v\sigma$*.*

Lemma 1. *Let* $\boldsymbol{T} = \langle T, \eta, \sigma \rangle$ *and* $\boldsymbol{T'} = \langle T, \eta, \sigma' \rangle$ *be two markup term sets. Suppose that* E *is a convergent regular subterm equational theory,* $\boldsymbol{T}$ *is well-formed, and* $\sigma \approx_{E,T\eta} \sigma'$*. If* (u, v) *is a type-I (or II) constraint of* $\boldsymbol{T}$*, then* (u, v) *is also a type-I (or II) constraint of* $\boldsymbol{T'}$*.*

A noticeable consequence of Lemma 1 is that, constraint property does not change with respect to operational equivalent substitutions and thus a new solver could be obtained, whenever one finds a constraint. More precisely, suppose that (u, v) is a type-I (or type-II) constraint of $\langle T, \eta_1, \sigma_1 \rangle$ and $\mu = mgu(u, v)$ (or the most general substitution satisfying $u\mu \rightarrow_{R_E} v\mu$). Then, it can be shown that $\mu \preceq \sigma_1'$ for any $\sigma_1' \approx_{E,T\eta} \sigma_1$. We therefore make the following definition.

Definition 6 (Update). *Let* $\boldsymbol{T} = \langle T, \eta_1, \sigma_1 \rangle$*. Suppose that* μ *is a substitution such that* $Dom(\mu) \subseteq fv(T\eta_1)$*. An* update *of* $\boldsymbol{T}$ *by* μ*, denoted by* $\boldsymbol{T} \downarrow_\mu$*, is a markup term set* $\langle T, \eta_2, \sigma_2 \rangle$ *such that* $\eta_2 = \eta_1\mu$*,* $\mu\sigma_2 = \sigma_1$*, and* $Dom(\sigma_2) = fv(T\eta_2)$*.*

4.2 Reduction

Definition 7 (Reduction). *Let $\boldsymbol{T} = \langle T, \eta, \sigma \rangle$ be a markup term set. Suppose that $T\eta \vdash u$. Then,*

- *(u, v) is a* type-I reduction *of $\boldsymbol{T}$, if $T\eta \vdash u$, $T\eta \nvdash v$, and $u =_s l\theta$ and $v =_s r\theta$ for some $l \to r \in R_E$ and substitution θ;*
- *(u, v) is a* type-II reduction *of $\boldsymbol{T}$, if u is R_E-normal, $T\eta \vdash u$, $T\eta\sigma \nvdash v$, and $u\sigma =_s l\theta$ and $v =_s r\theta$ for some $l \to r \in R_E$ and substitution θ.*

The reason why we distinguish between constraint and reduction is that a constraint imposes immediate restriction on possible substitutions, whereas a reduction does not. Further, we show that the type-II reduction counterpart of Lemma 1 does not generally hold and thus no immediate restriction can be obtained. To wit, we let $T = \{\{N_A\}_{K_B^+}, x\}$, $\sigma = [K_B^-/x]$, $\sigma' = [N_B/x]$, and $u =_s \mathtt{dec}(\{N_A\}_{K_B^+}, x)$. Then, it can be shown that $\sigma \approx_{E_{dy},T} \sigma'$ and, further, (u, N_A) is a type-II-reduction of $\langle T, \phi, \sigma \rangle$. However, (u, N_A) is not a type-II-reduction of $\langle T, \phi, \sigma' \rangle$, because

$$\mathtt{dec}(\{N_A\}_{K_B^+}, x)\sigma' =_s \mathtt{dec}(\{N_A\}_{K_B^+}, N_B) \not\to_{R_{E_{dy}}} N_A$$

Though reductions do not give any direct impact on possible substitutions, they may introduce new constraints afterwards, thanks to the transformation lemma.

Lemma 2 (Transformation Lemma).

(i). Suppose that $T \vdash_E t$. Then, $\sigma_1 \approx_{E,T} \sigma_2$ iff $\sigma_1 \approx_{E,T\cup\{t\}} \sigma_2$;

(ii). Suppose that $T \vdash s$ and x never occurs in T. Let $s\sigma_1 \to^!_{R_E} w_1$ and $s\sigma_2 \to^!_{R_E} w_2$. Then, $\sigma_1 \approx_{E,T} \sigma_2$ iff $\sigma_1' \approx_{E,T\cup\{x\}} \sigma_2'$, where $\sigma_1' = \sigma_1 \cup [w_1/x]$ and $\sigma_2' = \sigma_2 \cup [w_2/x]$.

Proof. (i). The "if" part is trivial. We now prove the "only if" part, that is, to show that $u\sigma_1 =_E v\sigma_1 \Leftrightarrow u\sigma_2 =_E v\sigma_2$ for all terms u and v such that $T \cup \{t\} \vdash \{u, v\}$. We only need to prove one direction, due to the symmetry. Note that $T \cup \{t\} \equiv_E T$. Then, there exists two terms u' and v' such that $T \vdash \{u', v'\}$, $u' =_E u$, and $v' =_E v$. Clearly, $u\sigma_1 =_E u'\sigma_1$ and $v\sigma_1 =_E v'\sigma_1$. So, $u\sigma_1 =_E v\sigma_1$ implies $u'\sigma_1 =_E v'\sigma_1$. By operational equivalence, we have $u'\sigma_2 =_E v'\sigma_2$ and thus $u\sigma_2 =_E v\sigma_2$. Likewise, it can be shown that $u\sigma_2 =_E v\sigma_2 \Leftrightarrow u\sigma_1 =_E v\sigma_1$. Hence, $\sigma_1 \approx_{E,T\cup\{t\}} \sigma_2$.

(ii). ("If " part) Let $T \vdash \{u, v\}$. Clearly, $T \cup \{x\} \vdash \{u, v\}$. Since $\sigma_1' \approx_{E,T\cup\{x\}} \sigma_2'$ by assumption, we have $u\sigma_1' =_E v\sigma_1' \Leftrightarrow u\sigma_2' =_E v\sigma_2'$. Note that $T \vdash \{u, v\}$ and x does not occur in T. Obviously, $x \notin fv(u)$ and $x \notin fv(v)$. So, $u\sigma_1' =_s u\sigma_1$, $v\sigma_1' =_s v\sigma_1$, $u\sigma_2' =_s u\sigma_2$, and $v\sigma_2' =_s v\sigma_2$. So, $u\sigma_1 =_E v\sigma_1 \Leftrightarrow u\sigma_2 =_E v\sigma_2$. Hence, $\sigma_1 \approx_{E,T} \sigma_2$. ("Only if" part) It suffices to show that for all terms u and v such that $T \cup \{x\} \vdash \{u, v\}$ we have $u\sigma_1' =_E v\sigma_1' \Leftrightarrow u\sigma_2' =_E v\sigma_2'$. Let $u' =_s u[x \mapsto s]$ and $v' =_s v[x \mapsto s]$. Since x never occurs in T and $T \vdash s$ by

assumption, one can easily show that $T \vdash \{u', v'\}$. Note that $\sigma_1' = \sigma_1 \cup [w_1/x]$ and σ_1' is ground. We see that $u'\sigma_1 =_s u\sigma_1[x \mapsto s\sigma_1] =_E u\sigma_1[x \mapsto w_1] =_s u\sigma_1'$. Thus, $u\sigma_1' =_E u'\sigma_1$. Similarly, $v\sigma_1' =_E v'\sigma_1$, $u\sigma_2' =_E u'\sigma_2$, and $v\sigma_2' =_E v'\sigma_2$. On the other hand, since $\sigma_1 \approx_{E,T} \sigma_2$ and $T \vdash \{u', v'\}$, by operational equivalence we get $u'\sigma_1 =_E v'\sigma_1 \Leftrightarrow u'\sigma_2 =_E v'\sigma_2$. Finally, we conclude that $u\sigma_1' =_E v\sigma_1' \Leftrightarrow u\sigma_2' =_E v\sigma_2'$. □

As the above lemma suggests, if (u, v) is a type-I reduction of $\langle T, \eta, \sigma\rangle$, then $\sigma_1 \approx_{E,T\eta} \sigma_2$ iff $\sigma_1 \approx_{E,T\eta\cup\{v\}} \sigma_2$. So, we can change $\langle T, \eta, \sigma\rangle$ to $\langle T \cup \{v'\}, \eta, \sigma\rangle$ where $v'\eta =_s v$, without losing or adding any condition(s) for operational equivalence; this is analogous to the transformation made by update in the previous subsection.

4.3 Our Construction

The last missing building block is the following definition, which formalizes our previous discussion about how a constraint or a reduction enables a useful transformation.

Definition 8 (Markup Term Set Rewriting). *Let $\boldsymbol{T} = \langle T, \eta, \sigma\rangle$ be a markup term set. We define a binary relation $\rightarrow_E$ on markup term sets as follows:*

- *If (u, v) is a type-I constraint of $\boldsymbol{T}$, then $\boldsymbol{T} \rightarrow_E \boldsymbol{T}\downarrow_\mu$, where $\mu = mgu(u, v)$;*
- *If (u, v) is a type-II constraint of $\boldsymbol{T}$, then $\boldsymbol{T} \rightarrow_E \boldsymbol{T}\downarrow_\mu$, where μ is the most general substitution satisfying $u\mu \rightarrow_{R_E} v\mu$;*
- *If (u, v) is a type-I reduction of $\boldsymbol{T}$, then $\boldsymbol{T} \rightarrow_E \langle T \cup v', \eta, \sigma\rangle$, where v' a term satisfying $v'\eta =_s v$*[3].
- *If (u, v) is a type-II reduction of $\boldsymbol{T}$, then $\boldsymbol{T} \rightarrow_E \langle T \cup \{z\}, \eta, \sigma \cup [v/z]\rangle$, where z is a fresh variable.*

Intuitively, the markup term set rewriting is a recognizing process; every time a markup term set is rewritten, either a constraint or a reduction is found. By collecting all the constraints and reductions, we get all information needed to recognize any proven recognizable term.

The first feature of markup term set rewriting we obtain is that well-formedness property of markup term sets is invariant under transformations in both forward and backward directions.

Lemma 3 (Well-formedness Preserving). *Let E be a regular subterm equational theory. Suppose that $\boldsymbol{T_0} \rightarrow_E^{*(n)} \boldsymbol{T_n}$. Then, $\boldsymbol{T_0}$ is well-formed iff $\boldsymbol{T_n}$ is well-formed.*

Another feature of this transformation is its naturality in the sense that such a transformation will not impose or relax any restrictions on operational equivalence. The following lemma states this formally.

[3] This can be done by replacing every $x\eta \subseteq v$ with x.

Lemma 4 (Naturality). *Let E be a convergent regular subterm equational theory and $\boldsymbol{T} = \langle T, \phi, \sigma\rangle$ be a well-formed markup term set. Suppose that $\boldsymbol{T} \rightarrow_E^{*(n)} \boldsymbol{T_n} = \langle T_n, \eta_n, \sigma_n\rangle$. Then, $\sigma \approx_{E,T} \sigma'$ iff $\sigma' = [\eta_n\sigma'_n]_{Dom(\sigma)}$ for some σ'_n such that $\sigma'_n \approx_{E,T_n\eta_n} \sigma_n$.*

4.4 Algorithms

We now present the algorithm for markup term set rewriting under standard Dolev-Yao intruder model, that is, given a well-formed markup term set $\boldsymbol{T}$ as input, it returns a well-formed markup term set $\boldsymbol{T'}$ such that $\boldsymbol{T} \rightarrow^!_{E_{dy}} \boldsymbol{T'}$. Then, in light of the correctness theorem (Theorem 1), one can decide the recognizability accordingly.

Not surprisingly, with the proven salient features in Section 4.3, markup term set rewriting enables us to find recognizable terms.

Theorem 1 (Correctness). *Let T be a regular and ground term set, and $\sigma = [t/x]$ be a ground substitution. If $\boldsymbol{Solve}(\langle T\cup\{x\}, \phi, \sigma\rangle) = \langle T_n, \eta_n, \sigma_n\rangle$ and $x\eta_n =_s t$, then $T \rhd_{E_{dy}} t$.*

Proof. Let σ' be an arbitrary substitution satisfying $\sigma' \approx_{E_{dy},T\cup\{x\}} \sigma$. Then, we can apply Lemma 4 and obtain that $\sigma' = [\eta_n\sigma'_n]_{Dom(\sigma)}$ for some σ'_n such that $\sigma'_n \approx_{E,T_n} \sigma_n$. Then, $x\sigma' =_s x\eta_n\sigma'_n =_s t\sigma'_n =_s t$. Moreover, since $\sigma' \approx_{E_{dy},T\cup\{x\}} \sigma$, we get $Dom(\sigma') = Dom(\sigma) = \{x\}$. Thus, $\sigma' = [t/x] = \sigma$. Now, it is not hard to see that $\sigma' \approx_{E_{dy},T\cup\{x\}} [t/x]$ if and only if $\sigma' = [t/x]$. By the definition of verifiability, we have $T \rhd_{E_{dy}} t$. □

Theorem 2 (Termination). *Suppose that T is a ground term set and $\sigma = [t/x]$ is a ground substitution. Then, algorithm $\boldsymbol{Solve}(\langle T \cup \{x\}, \phi, \sigma\rangle)$ is terminating.*

Proof. (Sketch) Let $\boldsymbol{T_0} = \langle T \cup \{x\}, \phi, \sigma\rangle$ and $\boldsymbol{T_0} \rightarrow^!_{E_{dy}} \boldsymbol{T'} = \langle T', \eta', \sigma'\rangle$. Then, $\boldsymbol{T'} = \mathbf{Solve}(\boldsymbol{T_0})$. We assume, without loss of generality, that

$$\begin{aligned} \boldsymbol{T_0} &\rightarrow^*_{E_{dy}} \boldsymbol{T_i} = \langle T_i, \eta_i, \sigma_i\rangle \\ &\rightarrow_{E_{dy}} \boldsymbol{T_{i+1}} = \langle T_{i+1}, \eta_{i+1}, \sigma_{i+1}\rangle \rightarrow^!_{E_{dy}} \boldsymbol{T'} \end{aligned} \tag{1}$$

Using Definition 6 and 8, we observe that $T_i\eta_i\sigma_i \subseteq T_{i+1}\eta_{i+1}\sigma_{i+1}$. Moreover, since E_{dy} is a regular subterm equational theory, a case-by-case analysis shows that each $t \in (T_{i+1}\eta_{i+1}\sigma_{i+1})\backslash(T_i\eta_i\sigma_i)$ occurs in $T_i\eta_i\sigma_i$. Thus, one can easily see that $T \cup \{t\} \subseteq T'\eta'\sigma'$ and each $t \in (T'\eta'\sigma')\backslash(T \cup \{t\})$ occurs in $T \cup \{t\}$. Note that the number of terms occurring in term set $T \cup \{t\}$ is bounded by $\|t\| - 1 + \sum_{u\in T}(\|u\| - 1)$. Consequently, $T'\eta'\sigma'$ is a finite term set.

To avoid any confusion, we assume the markup term set $\boldsymbol{T_i}$ as the input for Algorithm 1 in the following discussion.

Let us first analyze line (1) to (14) of Algorithm 1, which cope with reductions (either type-I or II). Each reduction (either type-I or II) would produce a new term, that is $v'\eta\sigma$ or w, in $T'\eta'\sigma'$, because both $v'\eta\sigma$ and w are ground terms,

Algorithm 1 **Solve($\boldsymbol{T}$)**

Input: a well-formed markup term set $\boldsymbol{T} = \langle T, \eta, \sigma \rangle$
Output: an updated markup term set

/* *type-I reduction* */
if $\exists u, v.\ u \in T\eta, T\eta \not\vdash v$ and $\mathtt{fst}$(or $\mathtt{snd}$)$(u) \rightarrow_{R_E} v$ **then**
  v' is obtained by replacing every $x\eta$ in v with x, where $x \in Dom(\eta)$.
  return Solve$(\langle T \cup v', \eta, \sigma \rangle)$
if $\exists u, v, s.\ u \in T\eta, T\eta \not\vdash v, T\eta \vdash s$ and $\mathtt{dec}(u, s) \rightarrow_{R_E} v$ **then**
  v' is obtained by replacing every $x\eta$ in v with x, where $x \in Dom(\eta)$.
  return Solve$(\langle T \cup v', \eta, \sigma \rangle)$

/* *type-II reduction* */
if $\exists u, w.\ u \in \mathcal{X} \cap T\eta$, $\mathtt{fst}$(or $\mathtt{snd}$)$(u\sigma) \rightarrow_{R_E} w$
and there does not exist a term v such that $T\eta \vdash v$ and $v\sigma =_s w$ **then**
  let z be a fresh variable
  $\boldsymbol{T} \leftarrow \langle T \cup z, \eta, \sigma \cup [w/z] \rangle$
  return Solve$(\boldsymbol{T})$
if $\exists u, w, s.\ u \in T\eta, T\eta \vdash s$ and $\mathtt{dec}(u, s)\sigma \rightarrow_{R_E} w$
and there does not exist a term v such that $T\eta \vdash v$ and $v\sigma =_s w$ **then**
  let z be a new variable that never occurs in $T\eta$
  $\boldsymbol{T} \leftarrow \langle T \cup z, \eta, \sigma \cup [w/z] \rangle$
  return Solve$(\boldsymbol{T})$

/* *type-I constraint* */
if $\exists u, v.\ u \in T\eta, T\eta \vdash v$, v is regular, $u \neq_s v, u\sigma =_s v\sigma$ **then**
  $\boldsymbol{T} \leftarrow \boldsymbol{T} \downarrow_\mu$ where $\mu = mgu(u, v)$
  return Solve$(\boldsymbol{T})$

/* *type-II constraint* */
if $\exists u, v.\ u \in \mathcal{X} \cap T\eta, T\eta \vdash v$, v is regular, $v \notin \mathcal{X}$, and $\mathtt{fst}$(or $\mathtt{snd}$)$(u\sigma) \rightarrow_{R_E} v\sigma$ **then**
  $\boldsymbol{T} \leftarrow \boldsymbol{T} \downarrow_\mu$ where μ is the most general substitution
  satisfying $\mathtt{fst}$(or $\mathtt{snd}$)$(u\mu) \rightarrow_{R_E} v\mu$
  return Solve$(\boldsymbol{T})$
if $\exists u, v, s.\ u \in T\eta, T\eta \vdash v$, v is regular, $v \notin \mathcal{X}$, $T\eta \vdash s$, and $\mathtt{dec}(u, s)\sigma \rightarrow_{R_E} v\sigma$ **then**
  $\boldsymbol{T} \leftarrow \boldsymbol{T} \downarrow_\mu$ where μ is the most general substitution satisfying
    $\mathtt{dec}(u, s)\mu \rightarrow_{R_E} v\mu$
  return Solve$(\boldsymbol{T})$
return $\boldsymbol{T}$

which are not subject to change in markup term set rewriting. As a result, the number of reductions explored by the algorithm is also bounded.

Now, we turn to line (15) to (23) of Algorithm 1, which cope with constraints (either type-I or II). It's important to note that to build a constraint, say (u, v), there must exist a term $u_0 \in T_i\eta_i$. Then, $u_0\sigma_i \in T_i\eta_i\sigma_i \subseteq T'\sigma'\eta'$. Since $u_0\sigma_i$ is ground and subject to no change, there is exactly one $u_0\sigma_i \in T'\sigma'\eta'$. Though not unique, such terms u'_0 that satisfy $u'_0\sigma'_i =_s u_0\sigma_i$ is finite, simply because $T'_i\eta'_i$ is a finite term set. The number of constraints explored by the algorithm is therefore bounded.

Finally, we conclude that **Solve**$(\langle T \cup \{x\}, \phi, \sigma\rangle)$ is terminating. □

We do not address computational complexity here due to the lack of space, and also due to the fact that efficiency is not a major concern in deciding recognizability. However, we claim without proof that, the problem of deciding recognizability under standard Dolev-Yao model can be solved in polynomial time.

5 Discussion

In this section, we first address the last two missing important issues: (i). How to deal with cases in which (prior) knowledge is not ground? (ii). How to account for dynamic interactions between protocol participants? Then, we present our experimental results.

Although we have assumed that T is a ground term set in the definition of recognizability, our construction has been kept in a fairly general way. In view of both Lemma 4 and Theorem 1, we only need $\boldsymbol{T} = \langle T, \phi, \sigma\rangle$ to be a well-formed markup term set. In other words, we just need $T\sigma$ is ground term set and contains only regular terms. This is reasonable, because all well-designed protocols should allow for regular terms. As we do not need T to be ground, the only requirement is that $T\sigma$ is ground. Thus, our construction can be applied to recognize multiple messages by extending the domain of σ to $fv(T)$.

To account for dynamic interactions, starting from a ground initial knowledge, one can simply use a fresh variable to signify an incoming message and the expected substitution is specified according to the protocol specification. In this way, we get a new markup term set $\boldsymbol{T'} = \langle T \cup X, \phi, \sigma\rangle$, in which X is a set of variables representing incoming messages and σ is the expected substitution. This is analog to the role played by security protocol compilation [4], in which how messages are interpreted and processed by an agent is examined.

Experimental Results To evaluate the proposed approach, we have implemented the algorithm and incorporated it with Athena [22], a well-know security protocol verifier, with C++. We use it to verify a set of well studied protocols that are vulnerable to type flaw attacks.

The experiments show that, with the security protocols properly labeled, all type flaw attacks are found. Table 1 lists the number of states that are generated and examined.

Protocol	Number of states explored	
	when the first attack is detected	the full state space search
WooLam	7	13
Neumann Stubblebine	97	117
Otway Rees	2	28

Table 1. Experimental results.

6 Conclusion

In this paper, we provide a terminating procedure to decide recognizability under Dolev-Yao adversaries. Our construction is general enough to deal with multiple messages. To recap the power of recognizability, we apply it to security protocols verification. By exploring potentially ambiguous messages in a protocol, it provides a more effective way to identify vulnerabilities to type flaw attack(s).

The general idea of verifying information is also useful in finding guessing attacks [16]. In [2] Baudet uses the notion of static equivalence to characterize off-line guessing attacks and an efficient procedure is given for finding such attacks in a bounded number of sessions. Since our approach provides a way to explore ambiguous messages by checking recognizability, we plan to extend the procedure to the study of off-line guessing attacks and non-trace security properties. We also plan to investigate the relationship between our approach and the applied pi calculus, as the results obtained in static/observational equivalence may be reused for studying recognizability under a larger class of equational theories.

References

1. Martín Abadi and Véronique Cortier. Deciding knowledge in security protocols under equational theories. *Theor. Comput. Sci.*, 367(1):2–32, 2006.
2. Mathieu Baudet. Deciding security of protocols against off-line guessing attacks. In *CCS '05*, pages 16–25, New York, NY, USA, 2005. ACM.
3. Garrett Birkhoff. On the structure of abstract algebras. *Mathematical Proceedings of the Cambridge Philosophical Society*, 31(04):433–454, 1935.
4. Yannick Chevalier and Michaël Rusinowitch. Compiling and securing cryptographic protocols. *Inf. Process. Lett.*, 110(3):116–122, 2010.
5. H. Comon-Lundh and V. Shmatikov. Intruder deductions, constraint solving and insecurity decision in presence of exclusive or. In *LICS '03*, pages 271–280, June 2003.
6. Hubert Comon-Lundh, Véronique Cortier, and Eugen Zălinescu. Deciding security properties for cryptographic protocols. application to key cycles. *ACM Trans. Comput. Logic*, 11(2):1–42, 2010.
7. Stéphanie Delaune and Florent Jacquemard. A decision procedure for the verification of security protocols with explicit destructors. In *CCS '04*, pages 278–287, New York, NY, USA, 2004. ACM.
8. Nachum Dershowitz and David A. Plaisted. Rewriting. In *Handbook of Automated Reasoning*, pages 535–610. 2001.
9. D. Dolev and A. Yao. On the security of public key protocols. *Information Theory, IEEE Transactions on*, 29(2):198–208, Mar 1983.

10. L. Gong, R. Needham, and R. Yahalom. Reasoning about belief in cryptographic protocols. In *Proc. of IEEE Symposium on Security and Privacy*, pages 238–248, 1990.
11. James Heather, Gavin Lowe, and Steve Schneider. How to prevent type flaw attacks on security protocols. *J. Comput. Secur.*, 11(2):217–244, 2003.
12. Zhiwei Li and Weichao Wang. Rethinking about type-flaw attacks. In *Global Telecommunications Conference, 2010. GLOBECOM 2010. IEEE*, to appear.
13. Benjamin Long, Colin Fidge, and David Carrington. Cross-layer verification of type flaw attacks on security protocols. In *Thirtieth Australasian Computer Science Conference*, volume 62, pages 171–180, 2007.
14. Benjamin W. Long. Formal verification of type flaw attacks in security protocols. In *Proceedings of the Tenth Asia-Pacific Software Engineering Conference Software Engineering Conf*, page 415, 2003.
15. G. Lowe. Breaking and fixing the needham-schroeder public-key protocol using fdr. In *TACAs '96*, pages 147–166, 1996.
16. Gavin Lowe. Analysing protocols subject to guessing attacks. *J. Comput. Secur.*, 12(1):83–97, 2004.
17. Catherine Meadows. Identifying potential type confusion in authenticated messages. In *In Proceedings of Foundations of Computer Security 02*, pages 75–84, 2002.
18. Catherine Meadows. A procedure for verifying security against type confusion attacks. In *CSFW 03*, pages 62–72. IEEE Computer Society Press, 2003.
19. Jonathan Millen and Vitaly Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *CCS '01*, pages 166–175, 2001.
20. Jonathan Millen and Vitaly Shmatikov. Symbolic protocol analysis with an abelian group operator or diffie–hellman exponentiation. *J. Comput. Secur.*, 13(4):695–695, 2005.
21. Dave Otway and Owen Rees. Efficient and timely mutual authentication. *SIGOPS Oper. Syst. Rev.*, 21(1):8–10, 1987.
22. Dawn Xiaodong Song, Sergey Berezin, and Adrian Perrig. Athena: a novel approach to efficient automatic security protocol analysis. *J. Comput. Secur.*, 9(1-2):47–74, 2001.