Verifying Security of Composed Interaction for Web Services

Xiaolie Ye
Beijing Institute of Technology /School of Computer Science and Technology, Beijing, China
Email: yexiaolie@hotmail.com

Lejian Liao
Beijing Institute of Technology /School of Computer Science and Technology, Beijing, China
Email: liaolj@bit.edu.cn

***Abstract*—SOAP-based complex interactions of multiple end points in Web Services mostly consist of sub-processes or sub-protocols, which are reused as modular and need to comply with corresponding standards and proposals. However, the consistency of local and global properties of interactions is important for practical applications with high security requirement. Therefore, a method is proposed to formally describe composed interactions with the definition of basic and composed interaction model for Web Services. Furthermore, the semantic of interactions, is presented as a path of transitions in Action-based Kripke Transition System, on which some properties, such as secrecy and authentication, are described and verified as formulae in Past Linear Temporal Logic. Then a scenario of composed interactions for Web Services is given and some formal properties corresponding to security are more effectively checked by our approach.**

- *Index Terms*—Web Services, security verification, model checking

I. INTRODUCTION

With the popularity of Internet and the application of Web services-based distributed computing model of SOC, more and more standards and proposals have been defined for dealing with different concerns. As a result, the complexity of applications requires various standards or specifications be used in measure of high flexibility and modularity for Web services. So, composed interacting procedures, which follow current standards and proposals, broadly exist in communications of end-points and need to be considered whether properties of the whole interactions are satisfied, not just that of sub-protocol or sub-procedure.

A.. Security Verificaiton for Web Services

Due to its growing importance, Web Services requires rigorous security. As a difference method against relying on a secure transport layer, a more suitable way of securing SOAP messages is using the Web services interactive protocol stack to realize the goal of 'End-to-End ' security or trust. However, Web service is based on the semi-structure of XML. In particular, for the diversity of XML and the combination of Web Services specifications, it is definitely necessary to verify its security. Meanwhile, the theoretical community has been very successful in the last decade in developing methods for analyzing cryptographic protocols. If a SOAP messaging is transferred into a protocol, the methodologies and tools, Intruding Model, and Model checking technologies could be applied in verifying the semantic properties of the interactions among participants as Web Services as well as the analysis of security protocol.

B. Research on the Security of Web Services

Karthikeyan Bhargavan and Cedric Fournet et al [1], propose a new specification language TulaFale for writing machine-checkable descriptions of SOAP-based security protocols and their properties. The TulaFale language is based on the pi calculus, plus XML syntax (to express SOAP messaging), logical predicates (to construct and filter SOAP messages), and correspondence assertions (to specify authentication goals of protocols). Then TulaFale is complied into the applied pi calculus, and then runs Blanchet'sresolution-based protocol verifier [2]. We will use it as a reference to be compared on some figures in an examination.

E. Kleiner and A.W. Roscoe [3] propose a method for mapping interacting messages to abstract symbols in the style of Dolev-Yao, and Casper notation. They show that this translation preserves flaws and attacks. Meanwhile, they provide a way for analyzing WS-Security protocols. And also they demonstrate how the approach can be used to prove some property and discover attacks upon an application oft WS-SecureConversation .

In [4], Michael Backes1et al, take a security analysis to a concrete scenario, WS-ReliableMessaging,, from the aspects of symbols and encryption and use the Automated Validation for Internet Security Protocol and Application (AVISPA) [5] tools and OFMC[6] to establish an abstract model from the interactive scripts in protocol specification language. But the scenario is too simple and only includes tow participants: a client and server. In this paper, a more complex scenario is proposed, in which a

doi:10.4304/jsw.4.10.1160-1168

client, a proxy and a server interoperate with each other and comply with the specifications corresponding.

With the development of the technologies involving Web services composition and modeling the composition of services, Barbara Carminati et al put efforts on security constraint-based Web services composition. The requirements come from the management and schedule of service procedures that should be prevented from threads to secrecy, integrity, privacy, availability and anonymous properties [7]. Moreover, Barbara Carminati proposes that the security policy and capability of Web services be defined by DAML-S and a non-center mode of policy management be established to implement the satisfaction to security constraint during composing Web services.

Beside mentioned above, Lalana Kagal et al present some ontology of policy language and a distributed solution for policy management to enhance the traditional identification and access control framework. As a result, the responsibility and Obligations of Web services are depicted in security policy to realize the dynamic and non-center management [8].

C. My Work

Comparing with these approaches, this paper proposes a mechanism to verify the security of composed interactions for web services. Specially, a scenario of composed interaction for Web Services has been presented in trust brokering mode that satisfies the demand of deriving keys from shared secret and keys per-message, which are specified in WS-Trust and WS-SecureConversation. Furthermore, the abstract model of the scenario is built on Action based Kripke Transition System and is verified by AVISPA. Finally, a comparison with a tool of TulaFale is presented at the end.

D. Structure of the Paper

The rest of this paper is organized as follows. Section II defines a conceptual interaction model for messaging in web services. Specially, with presenting a concrete scenario of composed interaction in trust broking model, the preparation of discussing the security of composed interactions for Web Services is completed. In Section III, the semantic of High Level Protocol Specification Language [9] (HLPSL for short.) is introduced as well as Action based Kripke Transition System (AKTS for short) is proposed to make the semantic of interactions more clear and extend the expression capability of temporal properties. In Section IV, we present the major description of a composed interaction in a variety of HLPSL and verify the security by composing basic roles. Section V lists the experiment result that shows some advantage comparing with another tool, TulaFale. Section VI is a conclusion and future plan.

II. INTERACTION MODEL FOR WEB SERVICES

A. Definitions of Basic and Composed Interaction Model

Define 1: A Basic and Composed Interaction Model (IM for short). A protocol or standard specification that depicts the behavior of messages exchanged is defined as an interaction model, denoted with $\Phi < * >$ shown in Fig. 1.

$\Phi < * >$ is an basic interaction model, if <> has no parameters;

$\Phi < * >$ is a composed interaction model, if $< * >$ has a parameter that is an instance φ of another interaction model;

Figure 1. Denotation of Basic and Composed Interaction Model

So, if an interaction model is parameterized in another one, the last one is a composed IM.

B. WS-Trust and it's Interaction Model

WS-Trust [10] specifies a trust mode and a security token service framework in which applications establish a secure context to exchange SOAP messages on a mechanism how to deliver and proxy a security token, namely a claim to a resource, such as identity, key, name etc. The trust model is a procedure in which the SOAP messages from a requester should proof what they claim is trusted Moreover, WS-Trust also descriptively defines the issuance, canceling, updating, and negotiation of

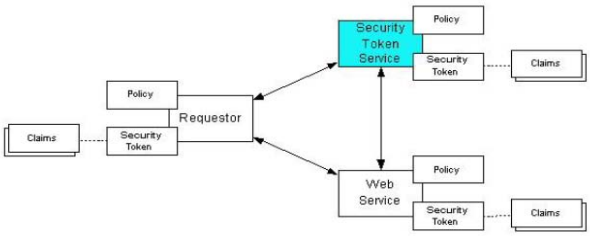

Figure 2. Security Token Service Framework

security token in a security token service framework shown as Fig. 2.

. But any mandatory and specific procedure or protocol does not exist in WS-Trust. So a formal definition of interacting profiles is proposed as shown in Fig. 3.

Define2: Interaction Model on WS-Trust.

$\Phi^{WS-Trust}(A, TM(P, TK), F^{STS})$

$-A$: It is a set of protocol participants except STS.

$-TM < P, TK >$: The trust mode, denoted with TM, specifies how to proxy and assess the trust relationship, in which,

$-P$:A set of policis denoted with P, defines how to proof claims corresponding to some tokens embedded in a SOAP message.

$-TK$: It is a set of security tokens, denoted with TK;

$-STS$: The security token service is called STS for short;

$-F^{STS}$: A frame of security token service, donoted with F^{STS}, defines a list of service interfaces involving issurance, cancel, update, negotiation etc.

Figure 3. Interaction Mode on WS-Trust

C. WS-SC and it's Interaction Model

Comparing with the specification of message authentication, such as WS-Security, which provides basic secure mechanism toward one-way messaging, the intention of WS-SecureConversation [11] (WS-SC for short) guarantees the security for a sequence of messages exchanged between participants and provides a mechanism of context oriented authentication, denoted with security context (SC for short) or session. Therefore, the performance and security of multiple interactions is improved for the mechanism. Moreover, a security context is shared among participants for the life of a communication session, which is indicated by a token that can be created by binding WS-Trust. The target of WS-SC is to establish a security context and amend it when need and specify how to derive shared key in multiple messages, even derive one key per message. As a reference to WS-Trust, a WS-SC interaction model is defined as shown in Fig. 4.

$\Phi^{WS-SC} < \Phi^{WS-Trust}\varphi_1, \Phi^{protocol}\varphi_2, ... > (\ \mathrm{P}(e, a, d\ ), DK, TK)$

$- < \Phi^{WS-Trust}\varphi_1 .. >$: For establishing a security context to protect an instance φ_2 of the interaction $\Phi^{protocol}$, Φ^{WS-SC} need to compose an instance of $\Phi^{WS-Trust}$, denoted with φ_1, in which a security token corresponding to the context be issured by *STS*;

$-\Phi^{protocol}\varphi_2$ is a behavior of generic interacting among participants.

$-\ \mathrm{P}(e, a, d)$: The major processes include establishing SC ('*e*' for short), amending SC ('*a*' for short), and deriving shared key('*d* ' for short);

$-$ *DK*: A set of drived keys created by derivation mechanism;

$-$ *TK* : A set of claims corresponding to SC;

Figure 4. Interaction Model on WS-SC

Define 3: Interaction Model on WS-SC.

D. A Scinario of Composed Interaction for Web services

We reference to the framework and pattern of WS-Trust and WS-SecureConversation and establish a scenario for discussing composed interactions, which are based on trust brokering model for Web Services. The scenario includes three participants: a client, a Web service and a (STS . For a convenient discussion, we assume the Web service shares secretes, authentication information and security tokens with STS by an internal mechanism, and the client, service and STS have been authenticated by an authority respectively with the certifications. As a fixed trusted root and a brokering service, the STS provides with security context tokens (SCTs for short) for the communication between the client and service. For accessing the Web service, the client requests a SCT from STS to protect a series of messages by the sequence of derived session keys. For the restriction of the paper's space, the amending, canceling and renewing security context mentioned in WS-SC are not discussed. Then the whole interactions include tow phases shown as the following Fig. 5. One is for the client to acquire a SCT issued by the STS involving the first step and the second step; another is for the client to interact with the Web service in a secure context, which is protected by the derivation of shared key referencing to the SCT and relates to step-3i and step-4i iteratively.

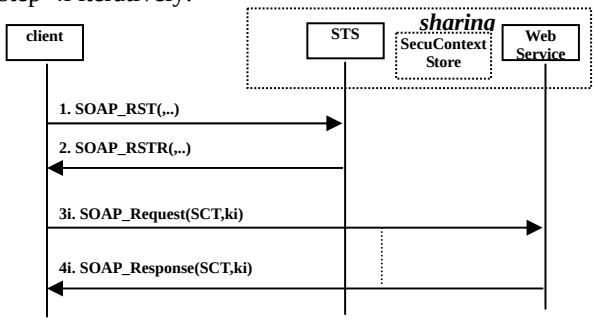

Figure 5. A Scenario of Interactions for Web Service.

So, we can give a formal description as shown in Fig. 6 for the issue about verifying the security of an interaction model for Web services. Moreover, we will discuses the semantic of composed interaction in an extensive Kripke structure and further to check the security of the scenario mentioned above.

$\Phi^P < \Phi^1, \Phi^2 ... > \models \varphi$, if Φ^P is a composed interaction for web services, φ is a temporal proposition about the properties of secrecy and authentication.

Figure 6. Issue on Verifying Security of IM.

III. FUNDACTION OF VERIFYING SECURITY OF INTERACTIONS FOR WEB SERVICES

HLPSL is a formal language to specify the protocol and the security problems in AVISPA, which defines control flow, data structure, alternative adversary models and complicated security properties based on roles in a protocol. In this paper, basic and composed interactions are depicted by a variety of HLPSL

Interaction specifications in a variety of HLPSL are composed of roles that are parameterized with respect to a set of typed variables. And the roles can be categorized into tow type. Some roles, called basic role and denoted with $Role_B(\Psi_B)$, serve to describe behaviors of one single agent during a run of a protocol or sub-protocol; Other roles, called composed roles and denoted with $Role_P(\Psi_P)$, instantiate these basic roles to model a scenario the protocol designer intends to analyze with respect to some security goals. For example, an entire interacting run (potentially consisting of the execution of multiple sub-protocols), and a session of the protocol between multiple agents are possible scenarios of interest. Provided a set of roles describing a concrete scenario, the security goals are to be defined as safety temporal properties.

A. TLA based Formal Semantic of HLPSL

As shown in Fig. 7 and Fig. 8, the specification of basic and composed role is respectively represented by $Role_B(\Psi_B)$ and $Role_P(\Psi_P)$. In $Role_B(\Psi_B)$, the agent 'pl' is as a player and Ψ_B, Λ_B are respectively corresponds to

$Role_B(\Psi_B)$ played_by *pl* def=
{
local Λ_B
init $init_B$
transition
$lb_1 : event_1 =|> act_1$
$lb_2 : event_2 =|> act_2$
$lb_i : event_i =|> act_i$
}

Figure 7. Basic Role

$Role_P(\Psi_P)$ def=
{
local Λ_P
init $init_P$
compositon
$Role_B^1 \wedge Role_B^2 \wedge ... \wedge Role_B^n$
}

Figure 8. Composed Role

the set of the parameters, local variables. Moreover, the initial block $init_B$ defines the initial value of variables and knowledge. Then, the block 'transition' contains all transition entries $Trans_i$, such as: $l_i : event_i =|> act_i$. Specially, the composition $Role_P(\Psi_P)$ of several basic roles is defined to express an interoperable scenario. As mentioned in [9], the roles can be translated into Temporal Logic of Action (TLA for short) as following:

$$\mathrm{TLA}(\mathrm{System}^{HLPSL}) \hat{=} \mathrm{Init}(Role) \wedge \Box\, \mathrm{Next}(Role)$$

B. Formal Semantic of HLPSL in AKTS（Action based Kripke Transition System）

The semantic of HLPSL based on TLA has been presented generally as above. However, for more convince to verify common properties, a Past Linear Temporal Logical based on extensive Kripke Transition System is proposed with a mean of translating TLA of $Role_B(\Psi_B)$ and $Role_P(\Psi_P)$ into the components of Kripke transition system.

Definition 4: Action-based Kripke Transition System is an extensive Kripke structure $\Phi(\Sigma, s_0, ACT, R, AP, L)$, where:

- Σ is a set of states;
- s_0 is the initial state;
- ACT is a finite set of events/actions and in which an e_i is one element, $i \in 1...n$;
- $R : \Sigma \times 2^{ACT} \times \Sigma$ is the transition relation; if α ranging over 2^{ACT}, a transition can be written as $s_i \xrightarrow{\alpha} s_j$, or (s_i, α, s_j);
- $L : \Sigma \to 2^{AP}$, a labeling function from a state to a set of atomic propositions *AP*;
- *AP*: a finite set of atomic propositions.

In $Role_B(\Psi_B)$, we can use X_B to express the whole variables involving a set of the variables in parameter Ψ_B, a set of local variables Λ_B and a set of the fresh variables Υ_B, namely $X_B = \Psi_B \cup \Lambda_B$ and $\Upsilon_B \subseteq \Lambda_B$. The left side of transition, namely $LHS(Trans_i)$ consists of a set of predication formulae $\wedge_0^n pb_j, j \in 0...n$ with the equations of Boolean expressions ranging over several variables and a set of events $\wedge_0^n ev_j, j \in 0...n$, such as $rcv(...)$ and $start(...)$ etc.; The right side of transition, namely $RHS(Trans_i)$ consists of assignments to variables $\wedge_0^n pa_j, j \in 0...n$, goal predications $\wedge_0^n pg_j, j \in 0...n$, and actions $\wedge_0^n at_j, j \in 0...n$, such as $snd(...)$. Additionally, we can use X_P to express the whole set of variables in $Role_P(\Psi_P)$ and $X_P = \Psi_P \cup \Lambda_P$.

First of all, let the set of atomic propositions AP be constructed from $\wedge_0^n pb_j, j \in 0...n$ and $\wedge_0^n pg_j, j \in 0...n$ in each $Role_B(\Psi_B)$.

Definition 5: The set of Atomic Proposition (AP for short) of basic role in AKTS shown in Fig. 9.

$$AP_i^B = \bigcup_1^k \mu(\wedge pb_j) \cup \bigcup_1^k \mu(\wedge pa_j) \cup \bigcup_1^k \mu(\wedge pg_j)$$

$-\ \mu$: It is a function to get a set of atomic propositons from a boolean exp ression or goal predication, such as $\wedge\, pb_j$;

$-\ k$: It is the size of transion entry in a basic role.

Figure 9. Atomic Proposition in AKTS.

Then, the whole state set can be created from each side of transition entry trs_j just by defining a correspondence relationship.

Definition 6: The set of States of Basic Roles, respectively Σ_i^B in AKTS shown in Fig. 10.

$$\Sigma_i^B = \{s \mid s_0 \leftrightarrow Inits \vee s_i \leftrightarrow LHS(trs_j) \vee s_{j'} \leftrightarrow RHS(trs_j)\}$$

- $\leftrightarrow$: It is the corresponding relationship between a state and the side of a transition with the reference to the definition of states in ;
- *Inits*: In the section of *HLPLS*, each variable $v \in \mathrm{Var(Inits)}$ has been initialized and this can be as the initial state s_0 in AKTS;
- $LHS(trs_j), RHS(trs_j)$: In the section 'transition', each entry *trs* be as a pair of states s_j and $s_{j'}$;

Figure 10. States of Basic Role in AKTS.

Definition 7: The labeling of Basic Roles, namely L_i^B in AKTS shown in Fig. 11:

$L_i^B = \{< s, P) > |$ if s $\leftrightarrow LHS(trs_i) \vee RHS(trs_i)$, then
$P = \mu_{LHS}(trs_i) \vee P = \mu_{RHS}(trs_i)\}$

- P: It is a set of atomic propositions satisfied in the state s;

- $\mu_{LHS}(trs_i) : P = \mu(\wedge pb_j)$;

- $\mu_{RHS}(trs_i) : P = \mu(\wedge pb_j) \cup \mu(\wedge pa_j) \cup \mu(\wedge pg_j) - \Delta$, in which $\Delta = \{ pb_j \mid v \in Var(pb_j)$ and $v \in Var(pa_j)\}$;

Figure 11. Labeling of Basic Roles.

Definition 8: The set of ACT for $\bigcup Role_B(\Psi_B)$ in AKTS shown in Fig. 12:

$ACT^B = \{act \mid act = (\wedge_0^m et_k \mid\!> \wedge_0^n an_l)_i$ in $trs_i\}$ (aa)
$ACT^P = \bigcup ACT^B$

Figure 12. Set of ACT in AKTS.

Definition 9: Direct Transition between Two States

$s_i \xrightarrow{\circ} s_j \quad s_i, s_j \in \Sigma$ iff $L(s_i) \rightarrow L(s_j)$

Definition 10: The transition relationship R_i^B for $Role_i^B(\Psi_B)$ in AKTS is defined as shown in Fig. 13:

In $Role_i^B(\Psi_B)$, each transition trs_i can be divided into tow states s_i, s_j and a transition relationship $s_i \xrightarrow{\alpha_i} s_j$ be added into R_i^B. Or if a direction transition exists such as $s_j \xrightarrow{\circ} s_k$, in which s_j is the original state, s_j in $s_i \xrightarrow{\alpha_i} s_j$ is substituted with the destination s_k. In an action at_i labeling a transition, when receiving a message, the role will analyze it with pairing and encrypting rules in $ana(m)$ and the result will update current knowledge kw_i of $Role_i^B(\Psi_B)$. The syntax of Var is as a function to get the set of variables in proposition. Then each at_i can perform the assignment to some variables and create unique value for fresh variables in $\Upsilon(trs_i, B)$ so as to make another message by $comp(m, kw_i^{'})$ and send it.

Definition 11: A composed ATKS, denoted with Φ^P shown in Fig. 14.

$\Phi^P = (\Sigma^P = \bigcup \Sigma_i^B, s_0{}^P = V(s_0^{\Phi_i^B}), ACT, R, AP, L)$
$- \Sigma^P = \bigcup \Sigma_i^B$;
$- \ s_0^P = V(s_0^{\Phi_i^B}), V$ *is a sign of vector*;
$- ACT^P = \bigcup ACT^B$;
$- R^P = \bigcup R_i^B$;
$- AP^P = \bigcup_1^n AP_i^B$;
$- L^P = \bigcup_1^n L_i^B$;

Figure 14. Composed Role in ATKS.

$R_i^B = \{ s_i \xrightarrow{\alpha_i} s_j \mid s_i \leftrightarrow LHS(trs_i) \wedge (s_j \leftrightarrow RHS(trs_i)$
$\vee (\exists s_k \leftrightarrow LHS(trs_i) \wedge s_j \xrightarrow{\circ} s_k \rightarrow s_j = s_k), s_i, s_j \in \Sigma\}$
$- \alpha_i : (ev_i \mid\!> at_i)$ *in* trs_i;
$- ev_i : start \vee (rcv(m) \rightarrow add(kw_i, m)$
$\wedge ana(m, kw_i))$;
$- ana(m)$: If $\exists m1, m2 \in Msg$, $m = pair(m1, m2)$, then $ana(m1) \wedge ana(m2)$; And if $\exists m1, k1 \in Msg$, $m = acrypt(k1, m1) \wedge comp(inv(k1), kw_i)$, then ana(m1);
$- comp(m, kw)$: if $\exists m \in kw$, *then* m; Or if $\exists m1, m2 \in Msg$, *then* $m = pair(m1, m2) \wedge comp(m1, kw)$ $\wedge comp(m2, kw)$;
Or if $\exists m1, k1 \in Msg, m = acrypt(k1, m1) \wedge comp(k1, kw)$ $\wedge comp(m1, kw)$;
$- at_i : \text{mod}(Var(\wedge pa_k) \cup \Upsilon(trs_i, B)) \rightarrow comp(m, kw_i^{'})$ $\wedge snd(m)$;
$- \Upsilon(trs_i, B) : \{\beta \mid \exists \beta_1 = new() \wedge \beta_2 = new()$ *occurs in* $\wedge pa, \beta_1 \neq \beta_2 \wedge \beta_1 \notin Used_{<B,trs_{i-1}>} \wedge \beta_2 \notin Used_{<B,trs_{i-1}>} \wedge$ $\beta_1 \notin Used^{\text{int ruder}} \wedge \beta_2 \notin Used^{\text{int ruder}}\}$;
$- Used_{<B,trs_i>} : Used_{<B,trs_{i-1}>} \cup \Upsilon(trs_i, B)$;
$- Used^{\text{int ruder}}$: *the set of used fress value in* int *ruder*;

Figure 13. Transition Relationship of Basic Role in AKTS

Definition 12 (Path): Let Φ be an ATKS and let $s \in \Sigma$, denoted with σ, shown in Fig. 15.

– A path denoted by σ is a sequence of $s_0 \xrightarrow{\alpha} s_1$, $s_1 \xrightarrow{\alpha} s_2, ...,$ with $s_i \xrightarrow{\alpha} s_{i+1} \in R$ for all $i \geq 0$;
– $\sigma(i) = s_i$ is to define the i^{th} state in such a path σ;
– $|\sigma|$ is the length of a path σ;

Figure 15. Definition of Path.

Definition 13: Past Linear Temporal Logical (PLTL for short) from AKTS, shown in Fig. 16.

$\sigma(i) \mid\!= \perp$ holds *always* ;
$\sigma(i) \mid\!= p$, iff $p \in L(\sigma(i))$;
$\sigma(i) \mid\!= \neg\varphi$ iff $\sigma(i) \mid\!\neq \varphi$;
$\sigma(i) \mid\!= \varphi_1 \wedge \varphi_2$ iff $\sigma(i) \mid\!= \varphi_1$ and $\sigma(i) \mid\!= \varphi_2$;
$\sigma(i) \mid\!= \varphi_1 \vee \varphi_2$ iff $\sigma(i) \mid\!= \varphi_1$ or $\sigma(i) \mid\!= \varphi_2$;
$\sigma \mid\!= Y\varphi$ iff $|\sigma| > 0$ and $\sigma(|\sigma| - 1) \mid\!= \varphi$;
$\sigma \mid\!= H\ \varphi$ iff $\exists j, 0 < j < |\sigma|$ and $\sigma(j) \mid\!= \varphi$;
$\sigma \mid\!= O\ \varphi$ iff $\exists i, 0 < i < |\sigma|\ \sigma(i) \mid\!= \varphi$
and $\exists j, i < j < |\sigma|$ and $\sigma(j) \mid\!= \perp$;
$\sigma \mid\!= \varphi_1 S\ \varphi_2$ iff $\exists i, 0 < i < |\sigma|\ \sigma(i) \mid\!= \varphi_2$
and $\exists j, i < j < |\sigma|$ and $\sigma(j) \mid\!= \varphi_1$;

Figure 16. Past Linear Temporal Logical.

Let σ be a path of actions in the system Φ and a Past Linear Temporal Logical (PLTL for short) shown in Fig.16, is inductively defined from AKTS, in which p is a atomic proposition and φ is a generic proposition formula. Then φ holds in $\sigma(i)$, denoted with

$$\sigma(i) \models \varphi \ ;$$

Past temporal operators, $Y, H, O,$ and S respectively are 'yesterday', 'historically', 'once', and 'since'. Moreover, a temporal formula φ is valid in a path σ , denoted with $\sigma \models \varphi$ iff $\sigma(0) \models \varphi$; a temporal formula φ is universally valid in Φ , written $\Phi \models \Box \varphi$, if

goal
% Weak authentication.
$[]\left(\text{request}(...) - >< - > \text{witness}(...)\right)$
end goal

$\forall \sigma$ in Φ $\sigma \models \varphi$. So for the following goal formula in HLPSL, the corresponding formulae in PTLT are those shown in Fig. 17:

Weak authentication:
$\Phi \models \Box(\text{request}(..) \rightarrow O\,\text{witness}(...))$
Secrecy:
$\Phi \models \Box \text{Secrete}(..)$

Figure 17. PTLTL of goal

IV. VERFYING SECURITY OF A SCENARIO OF COMPOSED INTERACTIONS

*, concatenation, such as Z*Y*X;
=>, pase a segment into elements by type;
{x}_k, message x encrypted by a key k , which can be symmetric or asymmetric key;
{x}_inv(k), message x encrypted by a private key k;
Hash(x), hashing message x;
cons(x, set_{type}), add an element x into the set_{type};
in(x, set_{type}), determine whether x in set_{type};
Rcv(...), receiving events;
Snd(...), sending events;
/\ , condition conjunction;
not(..), negative condition;
' , next state of a variable , such as Var';
ST(val, st_type), group the value val into a security token with the type st_type;
Ref(Id_x): reference to an identifier;
new(), creating a random value as None or key material;

Figure 18. Syntax in a variety of HLPSL

A. Syntax of HPSPL

HLPSL is exactly appropriate to describe the composed interacting behaviors on TLA. And for convenience, a variety of HLPSL is shown in Fig. 18.

B. Abstract description of Composed IM in HLPSL

The abstract description of the scenario mentioned before as a classic composed IM is specified in a variety of HLPSL with the symbols defined in Fig. 19.

The client is a composed role in the scenario of composed interactions, in which the basic role of STS_Client follows the specification of WS-Trust to request a security context token as shown in Fig. 20, another basic role of SC_Client continuously accesses a service and be protected from deriving a shared key per messaging,, specified in WS-SC with a security context token as shown in Fig. 21. Moreover, the composed role of the tow basic is also presented in Fig. 21.

C, S, T : Respectively, Client, Web Services, and STS;
Pwd: Password of a user ;
$\text{Cert}_{\text{agent}}$: Certification of agent, agent $\in \{\text{C, S, T}\}$;
Set_{sct}: Set of Security Context Token;
Id_{sc}: Identifier of a security context;
Id_{req}, Id_{res}: Identifers of request and response messages;
Id_{r} ,Id_{s}: Identifers of message received and sent;
RT_{rst}, RT_{rstr}: Request Type of RST and RSTR;
Act_{sct}, Act_{req}: Action Type of SCT and Request;
T_{ct}, T_{exp}: Respectively, created time and expire time;
SSK_{i}, SEK_{i}: Respectively, derivation signature and encryption key of the i^{th} generation in a sending direct;
RSK_{i}, REK_{i}: Respectively, derivation signature and encryption key of the i^{th} generation in a receiving direct;
PS, HS: Hash functions, PSha1;
K_{n} : the temple key, n=0,1,2,3...
EP_{x}: the entropy of secrete material for session keys, $\text{x} \in \{\text{C, S}\}$.

Figure 19. Statements of Variables in IM

The service of SC_Server is a basic role and the interacting behaviors are defined as shown in Fig. 22, in which different keys are derived for signing and encrypting messages from a shared secrete by the algorithm PSHA1. Additionally, in order to keep the keys fresh, a mechanism of subsequent derivation is presented in Fig. 22.

The services of STS_Server as a special web services is a basic role and the behaviors of interactions are shown in Fig. 23., which depicts how STS authenticates the requester and the target service and responds a security context token to STS_Client.

Role STS_Client (...) by Req
% Request a new security context token if no token or
% expired for a security context ;
State=0 ∧ Rcv(Start) ∧
not(in(Id_T^{SC}'∗Req∗S∗T∗SSec'∗T_T^{CT}'∗T_T^{Exp}', Set_{Req}^{SC}))
=|> State':=1∧...∧ K_{Req}':=new() ∧ EP_{Req}':=new() ∧
Snd((Msc_{Req}^{SCT}':=Id_{Req}^{MRST}'∗ACT_{SCT}∗T)∗
(ST_{Req}^{CT}:=ST($Cert_{Req}$, CT))∗
(Enc_{Req}^{K}':={K_{Req}'}_($Cert_T$))∗
(Enc_{Req}^{U}':={C∗Pwd}_K_{Req}')∗
(Enc_{Req}^{S}':={{Hash(Msc_{Req}^{SCT}'∗ST_{Req}^{CT}∗Enc_{Req}^{K}'∗
Enc_{Req}^{U}'∗(Enc_{Req}^{B}')
}_inv($Cert_{Req}$)}_K_{Req}')∗
(Enc_{Req}^{B}':={RT_{RST}∗S∗EP_{Req}'}_K_{Req}')
) ∧ $witness_{sts}$(.Req..) ∧ $secret_{sts}$(.Req..)
% Recieve materials to establish a new security context
% token;
State=1∧
Rcv((Msc_T^{SCT}'=> Id_T^{MRSTR}'∗Id_{Req}^{MRST}'∗ACT_{SCT}∗T)∗
(ST_T^{CT}' => $Cert_T$ ')∗(Enc_T^{K}'=> {K_T'}_($Cert_{Req}$'))∗
(Enc_T^{S}'=>{{Hash(Msc_T'∗ST_T^{CT}∗Enc_T^{K}'∗Enc_T^{A}'∗
Enc_T^{B}')}
inv($Cert_T$')}K_T')∗(Enc_T^{A}'=>{K_{Req}'}_K_T')∗
(Enc_T^{B}'=>{RT_{RSTR}∗Id_T^{SC}'∗S∗EP_S'∗T_T^{CT}'
∗T_T^{Exp}'}_K_T'))
=|>
SSec':=EP_{Req}'∗EP_S' ∧
Set_{Req}^{SC}' :=cons(Id_T^{SC}'∗Req∗S∗T∗SSec'∗T_T^{CT}'∗
T_T^{Exp}',Set_{Req}^{SC}') ∧ $request_{sts}$(.Req..)
End Role.

Figure 20. Basic Role STS_Client

Role SC_Client(...) by C
Rcv(start) ∧ State = 3i ∧in(Id_T^{SC}'∗C∗S∗T∗
SSec'∗T_T^{CT}'∗T_T^{Exp}',Set_C^{SC}') =|> State' :=4i ∧
N_C^{Dv}' ∧ Id_C^{MRE}' :=new() ∧ $Dats_C$':=new() ∧
% deriving the encryption and signature keys by
% by PSha1 specified in WS-SC.
DvK_C^{S}':= Off(PSha1(SSec'∗N_C^{Dv}'∗C∗S),off_sig,) ∧
DvK_C^{E}':= Off(PSha1(SSec'∗N_C^{Dv}'∗C∗S),off_enc,) ∧
Snd((Msc_C^{Req}':=Id_C^{MRE}'∗ACT_{req}∗S∗C)∗
(ST_C^{SC}':=ST(Id_T^{SC}', SC))∗
(ST_C^{DK}':=ST(Ref(Id_T^{SC}'))∗N_C^{Dv}', DK)∗
(Enc_C^{S}':={{Hash(Msc_C^{Req}'∗ST_C^{SC}'∗ST_C^{DK}'∗
Enc_C^{K}'∗Enc_C^{D}')}_DvK_C^{S}'}_DvK_C^{E}')∗
(Enc_C^{D}':={Dat_C^{Req}}_DvK_C^{E}')) ∧
$witness_{C-3}$(...) ∧ $secret_{C-3}$(...)
State = 4 ∧ in(Id_T^{SC}'∗Requor'∗S∗T∗SSec'
∗T_{CT}'∗T_{Exp}', Set_S^{SC}) ∧
Rcv((Msc_S^{Res}'=>Id_S^{MRO}'∗Id_C^{MRE}'∗ACT_{Res}∗C∗S)∗
(ST_S^{SC}'=>Id_T^{SC}')∗(ST_S^{DK}'=>ST(Ref(Id_T^{SC}'))
∗N_S^{Dv}', DK))∗(Enc_S^{S}'=>{{Hash(Msc_S^{Res}'
∗ST_S^{SC}∗ST_S^{DK}'∗Enc_S^{D}')}_ (DvK_S^{S}':= Off
(PSha1(SSec'∗N_S^{Dv}'∗S∗C),off_sig, ...))}_
(DvK_S^{E}':= Off(PSha1(SSec'∗N_S^{Dv}'∗S∗C),
off_enc, ...)))∗Enc_S^{D}') =|> State' :=3i ∧
$Dats_S^{Res}$' := {Enc_S^{D}'}_DvK_S^{E}' ∧ $request_{C-4}$(...)
End Role

Role ComposedClient (...) by C
STS_Client(C,..)$|_{Req=C}$ ∧SC_Client(C,..);
End Role

Figure 21. Basic Role of SC_Client and Composed Role of Client

Role SC_Server(..) by S

$Rcv((Msc_C^{Req}'=>Id_C^{MRST}'*ACT_{Req}*S*Requor')*$
$(ST_C^{SC}'=>Id_T^{SC})*(ST_C^{DK}'=>Ref(Id_T^{SC}')*N_{Requor}^{Dv}')*$
$(Enc_C^{S}'=>\{\{Hash(Msc_C^{Req}'*ST_C^{SC}'*ST_C^{DK}*Enc_C^{D}')\}$
$\_DvK_C^{S}':=Off(PSha1(SSec'*N_{Requor}^{Dv}'*Requor'*S),$
$off\_sig,...)\}$
$\_DvK_C^{E}':=Off(PSha1(SSec'*N_{Requor}^{Dv}'*Requor'*S),$
$off\_enc,...))*Enc_C^{D}')/\backslash Requor'=C/\backslash$
$in(Id_T^{SC}'*Requor'*S*T*SSec'*T_{CT}'*T_{Exp}', Set_S^{SC})$
$=|>$
$Dats_C^{Req}':=\{Enc_C^{D}'\}\_DvK_S^{E}'/\backslash Id_S^{MRO}'/\backslash Dats_S^{Res}'$
$/\backslash N_S^{Dv}':=new()/\backslash$
$DvK_S^{S}':=Off(PSha1(SSec'*N_S^{Dv}'*S*Requor'),$
$off\_sig,...)/\backslash$
$DvK_S^{E}':=Off(PSha1(SSec'*N_S^{Dv}'*S*Requor'),$
$off\_enc,...)/\backslash$
$Snd(\ (Msc_S^{Res}':=Id_S^{MRO}'*Id_C^{MRE}'*Act_{Res}*$
$Requor'*S)*(ST_C^{SC}':=ST(Id_T^{SC}', SC))*$
$(ST_S^{DK}':=ST(Ref(Id_T^{SC}'))*N_S^{Dv}', DK)*$
$(Enc_S^{S}':=\{\{Hash(Msc_S^{Res}'*ST_C^{SC}'*ST_S^{DK}'*$
$Enc_S^{D}')\}\_DvK_S^{S}'\}\_DvK_S^{E}')*$
$(Enc_S^{D}':=\{Dats_S^{Res}\}\_DvK_S^{E}')$
$)/\backslash witness_S(...)/\backslash request_S(...)/\backslash secret_S(...)$

End Role

Figure 22. Mode of Role Services S

C. Security of the composed interaction for Web services

As one aspect for the security of interaction description, the secrete indicated by the syntax secrete(...) in HLPSL, is defined by two stages, which are respectively shown as the following:

- Building a security context

These described in the Fig. 25 are all the secrete terms shared by client and STS.

$Sig_C:=\{Hash(Msc_C^{SCT}'*ST_C^{CT}*Enc_C^{K}'*Enc_C^{U}'*(Enc_C^{B}')\}$
$\_inv(Cert_C)$
$secret_{C\text{-}0}(K_C'*Sig_C*EP_C'\},...,\{C,T\});$
$Sig_T:=\{Hash(Msc_T^{SCT}'*ST_T^{CT}'*Enc_T^{K}'*Enc_T^{A}'*Enc_T^{B}')\}$
$\_inv(Cert_T)$
$secret_T(K_T'*K_C'*Sig_T*EP_S'*T_T^{CT}'*T_T^{Exp}'$
$,..,\{T,Requer'\}),$
$Requer'=C;$

Figure 25. Secrete Terms in 1st Stage

- Messaging in a security context

Role STS_Server(...) by T

$Rcv((Msc_C^{SCT}':=Id_C^{MRST}'*Act_{SCT}*T)*$
$(ST_C^{CT}=>ST(Cert_C,CT))*(Enc_C^{K}'=>\{K_C'\}\_Cert_T)*$
$(Enc_C^{S}'=>\{\{Hash(Msc_C^{SCT}'*Cert_{Requer'}'*Enc_C^{K}'*Enc_C^{U}'$
$*Enc_C^{B}')\}\_inv(Cert_C')\}\_K_C')*(Enc_C^{U}'=>\{Requer'*$
$Pwd'\}\_K_C')*(Enc_C^{B}'=>\{RT_{Rst}*TarSvr'*EP_C'\}\_K_C'))$
$/\backslash Requer'=C/\backslash in(Requer'*Pwd'*Cert_{Requer'}'*TarSvr',$
$Set_S^{Auth})/\backslash not(in(Id_T^{SC}'*C*S*T*SSec'*T_T^{CT}'*T_T^{Exp}',$
$Set_T^{SC})\ =|>$
$request_T(...)/\backslash Id_T^{MRSTR}'/\backslash Id_T^{SC}'/\backslash,T_T^{CT}'/\backslash T_T^{Exp}'/\backslash EP_S'/\backslash$
$K_T':=new()/\backslash$
$Snd(\ (Msc_T^{SCT}':=Id_T^{MRSTR}'*Id_C^{MRST}'*Act_{SCT}*T)*$
$(ST_T^{CT}':=ST(Cert_T,CT))*(Enc_T^{K}':=\{K_T'\}\_Cert_C)$
$(Enc_T^{A}':=\{K_C'\}\_K_T')*(Enc_T^{S}':=\{\{Hash(Msc_T^{SCT}'$
$*ST_T^{CT}'*Enc_T^{K}'*Enc_T^{A}'*Enc_T^{B}')\}\_inv(Cert_T)\}\_K_T')*$
$(Enc_T^{B}':=\{RT_{rstr}*Id_T^{SC}'*Requer'*EP_S'*T_{CT}'*$
$T_{Exp}'\}\_K_T'))/\backslash SSec':=EP_C'*EP_S'/\backslash Set_{SCT}':=cons(Id_T^{SC}'$
$*Requer'*TarSvr'*T*\ SSec'*T_T^{CT}'*T_T^{Exp}', Set_T^{SC})$
$/\backslash\ witness_T(...)/\backslash secret_T(...)$

End Role

Figure 23. Mode of Role STS T

These described in the Fig. 26 are all common secrete terms with the client C and the service S.

As another aspect of the protocol, AVISPA checks the correspondence of witness (...) and request (…) to validate the authentication, which is defined by two

$Sig_C:=\{Hash(Msc_C^{Req}'*ST_C^{SC}'*ST_C^{DK}'*Enc_C^{K}'*Enc_C^{D}')$
$\}\_DvK_C^{S}'$
$secret_{C\text{-}3}(Sig_C*Dat_C^{Req}'*DvK_C^{S}'*DvK_C^{E}',...,\{C,S\})$
$Sig_S:=\{Hash(Msc_S^{Req}'*ST_S^{SC}'*ST_S^{DK}'*Enc_S^{K}'*Enc_S^{D}')$
$\}\_DvK_S^{S}'$
$secret_S(Sig_S*Dats_S^{Res}'*DvK_S^{S}'*DvK_S^{E}',...,$
$\{Requor',S\}),Requer'=C;$

Figure 26 Secrete Terms in 2nd Stage

$witness_{C\text{-}0}(C,T,...,Msc_C^{SCT}'*ST_C^{CT}*Enc_C^{K}'*Enc_C^{U}'*Enc_C^{S}'$
$*Enc_C^{B}')...$
$request_T(T,Requer',...,Msc_C^{SCT}'*ST_C^{CT}*Enc_C^{K}'*Enc_C^{U}'*$
$Enc_C^{S}'*Enc_C^{B}')$
$witness_T(T,Requer',...,Msc_T^{SCT}'*ST_T^{CT}*Enc_T^{K}'*Enc_T^{A}'$
$*Enc_T^{B}')...$
$request_{C\text{-}1}(C,T,...,Msc_T^{SCT}'*ST_T^{CT}*Enc_T^{K}'*Enc_T^{A}'*Enc_T^{B}'),$
$Requer'=C$

Figure 27. Authentication Terms in 1st Stage

stages, which are respectively shown as the following:

- Building a security context

Those described in Fig. 27 are authenticated in AVISPA.

- Messaging in a security context

Those described in Fig. 28 are authenticated in AVISPA.

$witness_{C\text{-}0}(C, T, ..., Msc_C^{SCT'} * ST_C^{CT} * Enc_C^{K'} * Enc_C^{U'} * Enc_C^{S'} * Enc_C^{B'})...$

$request_T(T,Requer',...,Msc_C^{SCT'} * ST_C^{CT} * Enc_C^{K'} * Enc_C^{U'} * Enc_C^{S'} * Enc_C^{B'})$

$witness_T(T, Requer', ..., Msc_T^{SCT'} * ST_T^{CT} * Enc_T^{K'} * Enc_T^{A'} * Enc_T^{B'})...$

$request_{C\text{-}1}(C,T,...,Msc_T^{SCT'} * ST_T^{CT} * Enc_T^{K'} * Enc_T^{A'} * Enc_T^{B'}), Requer'=C$

Figure 28. Authentication Terms in 1st Stage

V EXPERIMENT RESULTS

Only the back-end CL_AtSe[12] successfully verifies the secrecy and authentication of the abstract model specified by HLPSL for the composed interactions in AVISPA. Additionally, Comparing with another analyzer TulaFale, which is combined with Blanchet' ProVerif based on Pi calculus, HLPSL and CL_AtSe are better in performance and scales of the validation of security protocols with the result shown as the following Tab I.

TABLE I. COMPARISON OF TOW APPROACHES

Tools	Model Category	Back-ends	Scale (lines)	Performance (seconds)
HLPSL	***TLA***	***CL-AtSe***	***<320***	***24/100 iterations***
TulaFale	***Pi calculus***	***ProVerif***	***>500***	***>25***

VI CONCLUSION AND FUTURE WORK

Complying with the specifications WS-Trust and WS-SC, a scenario of interactions for Web Services is presented to specially refine the brokering model with the mechanisms of deriving keys and per-message keys. Furthermore, AKTS is proposed to enhance the semantic of temporal logic to check the validity of more properties, specially secrecy and authentication. Moreover, the definitions are how to build AKTS from HLPSL. In future, we hope that the study on AKTS will be done about the inter-operation scenario with more complicated exchanges and more combined properties.

ACKNOWLEDGMENT

The work is funded by Grant 60873237 from the National Natural Science Foundation of China, and by Grand 4092037 from Beijing Municipal Natural Science Foundation and partially supported by Beijing Key Discipline Program

REFERENCES

[1] K. Bhargavan, C. Fournet, A. Gordon, and R. Pucella. TulaFale: A Security Tool for Web Services. In Proc. 2nd FMCO, LNCS 3188, pp. 197–222. Springer, 2004

[2] B. Blanchet. An Efficient Cryptographic Protocol Verifier based on Prolog Rules. In *14th IEEE Computer Security Foundations Workshop (CSFW-14)*, pages 82–96. IEEE Computer Society, 2001.

[3] E. Kleiner and A.W. Roscoe On the Relationship between Web Services Security and Traditional Protocols[J].Electronic Notes in Theoretical Computer Science, 2006, 155. Elsevier: pages 583-603.

[4] Michae Backes, Sebastian Mdersheim, Birgit Pfitzmann and Luca Viganò. Symbolic and Cryptographic Analysis of the Secure WS-ReliableMessaging Scenario. Lecture Notes in Computer Science.428-445.2006.

[5] L. Viganò. Automated Security Protocol Analysis with the AVISPA Tool[J].Proceedings of the XXI Mathematical Foundations of Programming Semantics (MFPS'05). Electronic Notes in Theoretical Computer Science, 2006, 155. Elsevier: 61-86.

[6] David Basin, Sebastian Mdersheim and Luca Viganò. OFMC: A Symbolic Model Checker for Security Protocols. International Journal of Information Security. Volume 4, Number 3 / 2005. Springer Berlin / Heidelberg: pages 181-208, 2005.

[7] B. Carminati, E. Ferrari and P.C.K. Hung. Security conscious Web Service Composition. In Proc. of the IEEE International Conference on Web Services. IEEE Computer Society Washington, DC, USA: pages 489~496, 2006. Computer Society Washington, DC, USA: pages 489~496, 2006.

[8] L. Kagal, T. Finin and A. Joshi. A Policy based Approach to Security for the Semantic Web. The SemanticWeb - ISWC 2003. Lecture Notes in Computer Science, Volume 2870/2003. Springer Berlin / Heidelberg:pages 402-418, 2003.

[9] Y. Chevalier, L. Compagna, J. Cuellar, P. Hankes Drielsma, J. Mantovani,S. M¨odersheim, and L. Vigneron. A High Level Protocol Specification Language for Industrial Security-Sensitive Protocols. In Proc. Workshop on Specification and Automated Processing of Security Requirements (SAPS'04), pp. 193–205. Austrian Computer Society, 2004.

[10] C. Kaler, A. Nadalin, et al. Web Services Trust Language (WS-Trust) Version 1.1, May 2004. At http://msdn.microsoft.com/ws/2004/04/ ws-trust/.

[11] C. Kaler, A. Nadalin, et al. Web Services Secure Conversation Language (WS-SecureConversation) Version 1.1, May 2004. At http://msdn.microsoft.com/ws/2004/04/ ws-secure-conversation/.

[12] Mathieu Turuani1, The CL-Atse Protocol Analyser[J]. Lecture Notes in Computer Science, Volume 4098/2006. Springer Berlin / Heidelberg:pages 277-286, 2006

Xiaolie Ye PHD student in School of Computer Science and Technology, Beijing Institute of Technology (BIT for short). He obtained Master degree in Software Engineering in 2004 from Institute of Computing technology, Tsinghua University. Current research fields include distributed Artificial Intelligence, Intelligent Web, and Information Security.

Lejian Liao Professor in School of Computer Science and Technology, Beijing Institute of Technology (BIT for short). He obtained PhD degree in Computer Sciences in 1994 from Institute of Computing technology, Chinese Academy of Sciences. Current research fields include distributed Artificial Intelligence and Intelligent Web.