http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (18): 4818-4822, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.4818.4822

Traces over Topological Spaces in Concurrent Security Protocols

Huanbao Wang
Department of Maths and Physics, Anhui Jianzhu University, Hefei, 230601, Anhui, China

Abstract: In this study we define a trace over their topological spaces in concurrent security protocols. For this reason, we specify a symbolic transition graph with a binary equivalence relation based on the CCS language with value-passing processes which is abbreviated to STGR, to describe concurrent communicating procedures in security protocols, namely a calculus of concurrent communicating processes and structure topological spaces with their message spaces among principals, where the regular participants or penetrators of the security protocol must be distinguished by the trace. A principal in a security protocol creates and transmit a message containing a new value, later receiving it back in a different cryptographic context. It can be concluded that some principal possessing the relevant key has received and transformed the message. The trace on STGR over topological spaces for value-passing processes based on CCS with infinite assignments being different from the original STG is used to follow the track of the regular participants of the security protocol, so it makes a model for proving safety properties of cryptographic protocols that run in an infinite concurrent way.

Key words: Trace, topological space, security protocol, concurrent computing

INTRODUCTION

We define a trace over their topological spaces in concurrent security protocols. Therefore the track of the regular participants of concurrent security protocols is followed by their traces. As we know, it is very difficult to describe the complex concurrent security protocol system in formal languages and to prove its safety properties such as secrecy, authentication and so on. One reason why security protocol formal analysis is hard is that the regular participants or penetrators of the security protocols have so many choices. In addition, security protocols, in many circumstances, run in concurrent ways. A principal in a security protocol creates and transmit a message containing a new value, later receiving it back in a different cryptographic context. It can be concluded that some principal possessing the relevant key has received and transformed the message (Li *et al*., 2009).

The inference can be obtained by the authentication test that is composed of three kinds of test cases (Guttman and Thayer, 2002). Suppose a principal in a security protocol sends a message containing a new value v, later receiving v back in a different context. The first one of three kinds of authentication test cases is an outgoing test, in which the new value v that is encrypted is transmitted and only a regular participant can extract it from that. The second one is an incoming test, in which the new value v encrypted is received back and only a regular participant can put it. The third one is an unsolicited test. These conclusions are expressed in the strand space formalism (Fabrega *et al*., 1999). We re-examine these ideas of authentication tests and define the trace over their topological spaces in concurrent security protocols.

In this study, we introduce a symbolic transition graph with a binary equivalence relation based on the CCS language with value-passing processes which is abbreviated to STGR, to describe concurrent communicating procedures in security protocols, namely a calculus of concurrent communicating processes. The topological spaces with their message spaces among principals are structured and the traces over them gained. So we can distinguish the penetrators from the regular participants of the security protocol by the traces. The conclusion of this kind provides a convenient context to prove concurrent security protocols correct.

OPERATIONAL SEMANTICS OF CCS

Let P, Q range over processes and A, B over alternations (sums), in which each alternate (or summand) of an alternation is a process guarded by an action μ of the form x or x^{-}, where x names a channel. The syntax in the CCS language is

$$P ::= A \mid \nu x P \mid P \mid P$$

where

$$A ::= 0 \mid \mu.P \mid A + A$$

And:

$$\mu ::= \bar{x} \mid x$$

The restriction v x p defines P as the scope of the name x; a name-occurrence in a process is free iff it is not scoped by v (Milner, 2009). The labeled transition system that is abbreviated to LTS over *Act* which is a given set of actions, is expressed for the semantics of the CCS language.

Suppose a set Q = {q_0, q_1, ...} of states, a state $q_0 \in$ Q called the start ones, a subset F of Q called the accepting states and a subset T of Q called the transition (Milner, 1999). A LTS over Act is a pair (Q, T) consisting of a set Q of states and a ternary relation T$\subset$ (Q×Act×Q) which is known as a transition relation. A transition $(q,\alpha,q') \in$ T is usually written $q\alpha \rightarrow q'$, where q and q′ are called as the source and the target of the transition respectively. We say P and Q are alpha-equivalent, written $P \equiv_\alpha Q$, if they differ only in a change of their restricted names.

An action may be an input one, of the form C ? X where c is from a set of channels, Chan, an output one, of the form c ! e, or a neutral one such as π. The set of free and bound variables of these actions is defined, where for example f v (c ! e) = f v (e) and b v (c ? x) = {x}. Suppose a set of expressions is Exp, ranged over by e which includes Var and V and a set of Boolean expressions, Bexp ranged over by b, where p ≠ b indicates that p (b) = true. If α ranges over a set Act of actions, then fv (α) = ϕ and b v (α) = ϕ.

The symbolic transition graph that is abbreviated to STG (Hennessy and Lin, 1995) is obtained from the CCS language using the standard approach of its structural operational semantics. A node n, i.e. an event of G graph of a strand space $(\langle s_i \rangle_{i\in I}, \neq)$ is labeled by a set of variables f v (n) and every directed branch is labeled by a guarded action in a STG, i.e. a directed graph. It is concluded that there is an expression:

$$fv(b) \cup fv(\alpha) \subseteq fv(m)$$

if a branch labeled by (b, α) goes from node m to n, i.e:

$$fv(n) \subseteq fv(m) \cup bv(\alpha)$$

if $m \overset{\alpha}{\mapsto} n$ i.e. $m \overset{true,\alpha}{\mapsto} n$ accordingly (Hennessy and Lin, 1996).

QUOTIENT SPACES OF CONCURRENT PROCESSES

The CCS language with value-passing processes is applied to describe concurrent communicating procedures in security protocols and to prove them correct. The STG as well as its related automaton, where the operational behaviors are bounded by the set Act of actions, is thought of as the operational semantics of the CCS language. If the domain of STG is finite, it is easy to check bisimulation equivalences between processes in the CCS language with some verification tools that have at their core algorithms (Lin, 1996).

But the domain of STG in concurrent communicating procedures in security protocols is usually infinite. The quotient space over topological spaces is achieved by partitioning off its domain according to the binary equivalence relation R $\in$ R over the infinite STG, where bisimulation equivalences between processes in concurrent security protocols are checked.

Suppose X is a set and let T be a family of subsets of X. Then T is called a topology on X if (1) Both the empty set and X are elements of T, (2) Any union of elements of T is an element of T, (3) Any intersection of finitely many elements of T is an element of T.

And if T is a topology on X, then the pair (X, T) is called a topological space. The notation X_τ may be used to denote a set X endowed with the particular topology T. The quotient space is defined as follows.

Definition 1: Suppose (X, T_x) is a topological space and let R $\in$ R be an equivalence relation on X. The quotient space Y =X/R is defined to be the set equivalence classes of elements of X:

$$Y = \{[x] : x \in X\} = \{\{v \in X : vRx\} : x \in X\}$$

Equipped with the topology where the open sets are defined to be those sets of equivalence classes those unions are open sets in X:

$$T_Y = \{U \subseteq Y : \cup U \in T_X\}$$

Equivalently, we can define them to be those sets with an open preimage under the quotient map q: X→X/R which sends a point in X to the equivalence class containing it. The expression is as follows:

$$T_Y = \{U \subseteq Y : q^{-1}(U) \in T_X\}$$

Symbolic bisimulation equivalences $\simeq^R$ over the quotient space of the infinite STG are, generally speaking, parameterized on the binary equivalence relation R $\in \mathbb{R}$. A process context C is, informally speaking, a process expression containing a pair of square brackets, represented by [ ]. Process contexts are formally given by the syntax:

$$C::=[]|\alpha.C+M|\upsilon aC|C|P|P|C$$

The notation C [Q] denotes the result of filling the square brackets in the context C by the process Q. The elementary contexts are such as α []+M,Vα [] and P| [], (Milner, 1999).

STGR: STG WITH A BINARY EQUIVALENCE RELATION R

A variant of symbolic transition graphs, i. e. a symbolic transition graph with a binary equivalence relation R which is abbreviated to STGR, is defined to an operational semantics of topological spaces of many value-passing processes in the CCS language with infinite assignments. A given binary relation ~ on a set A is said to be an equivalence relation if and only if it is reflexive, symmetric and transitive. Equivalently, for a, b and c in A:

- α~α (reflexivity)
- if α~b then b~c (symmetry)
- if α~b and b~c then b~c (transitivity)

A set A together with the relation~ is called a setoid. The equivalence class of *an* under ~, denoted [α], is defined as:

$$[a]=\{b\in A\,|\,a\sim b\}$$

Suppose (Q, T) is an LTS and let S be a binary relation over Q. Then S is called a strong simulation over (Q,T) if, whenever psq, if $p\xrightarrow{\alpha}p'$, then there exists $q'\in Q$ such that $q\xrightarrow{\alpha}q'$ and p′sq. A binary relation S over Q is said to be a strong bisimulation over the LTS (Q, T), if both S and its converse are simulations. We say that p and q are strongly bisimilar or strongly equivalent, written p~q if there exists a strong bisimulation S such that Psq (Lin, 1996).

A symbolic transition graph is a directed graph in which every node n is labeled by a set of variables fv (n) and every directed branch is labeled by a guarded action such that if a directed branch labeled by b, α goes from node m to n which we write as $m\overset{b,\alpha}{\mapsto}n$, then:

- $fv(b)\cup fv(\alpha)\subseteq fv(m)$

And:

- $fv(n)\subseteq bv(\alpha)\cup fv(m)$

And the transition branch $m\overset{E_A,\alpha}{\mapsto}n$ is denoted by $m\overset{\alpha}{\mapsto}n$ (Hennessy and Lin, 1995). The STGR is achieved by partitioning domains of value-passing processes with Boolean expressions which are a class of binary equivalence relation expressions. It is applied to an operational semantics of topological spaces of many value-passing processes based on CCS with infinite assignments in concurrent security protocols (Wang *et al*., 2006).

Definition 2: A symbolic transition graph with a binary equivalence relation R which is abbreviated to STGR, over a (A, T_A) of topological spaces of many value-passing processes based on CCS with infinite assignments, is a directed graph in which every node n is labeled by a set of variables fv (n) and every branch is labeled by a guarded action such that if a branch labeled by (R, α) goes from node m to n which we write as $m\overset{R,\alpha}{\mapsto}n$ then:

- $fv(R)\cup fv(\alpha)\subseteq fv(m)$

And:

- $fv(n)\subseteq bv(\alpha)\cup fv(m)$

And the transition branch $m\overset{E_A,\alpha}{\mapsto}n$ is denoted by, $m\overset{\alpha}{\mapsto}n$ where E_A is the total relation over a (A, T_A).

It is concluded that this form of operational semantics with a STG or a STGR must necessarily be given for open terms which may contain free variables. For example, even if c ? X.t is a closed term its residual after the symbolic action c ? X, namely t, will in general contain free occurrences of x. So the nodes of a STGR have their open terms, while a set A of (A, T_A) that is of topological spaces of many value-passing processes based on CCS with infinite assignments is a closed term. Thus a late operational semantics of a STGR is achieved as follows. Let Lact be a set of late actions and Lact is defined to be:

Lact:

$$=\{c?x\,|\,c\in Chan, x\in Var\}\cup$$

$$\{c!v\,|\,c\in Chan, v\in V\}\cup NAct$$

Proposition 1: A late operational semantics of topological spaces of many value-passing processes based on CCS with infinite assignments for a STGR has the form as follows:

- $m\overset{R,a}{\mapsto}n,\ a\in NAct$

Implying:

$$m_\sigma \xrightarrow{R\sigma,\ a} n_\sigma$$

- $m \xmapsto{R,\ c!e} n$

Implying:

$$m_\sigma \xrightarrow{R\sigma,\ c!e} n_\sigma$$

- $m \xmapsto{R,\ c?x} n$

Implying:

$$m_\sigma \xrightarrow{R\sigma,\ c?z} n_{\sigma[x\mapsto z]}\ z = new(fv(m_\sigma))$$

Let SyAct, ranged over by α represent the set of symbolic actions, be:

$$SyAct = \{c?x, c!e \mid c \in Chan\} \cup NAct$$

where, NAct is some set of neutral actions in a STGR. And let m, 6 be a node and a substitution, respectively. Also if t is a term of the term m_6 we use t [x ↦z] to denote the term, $m_{\sigma[x\mapsto z]}$

As far as is late or early actions concerned two symbolic variants of bisimulation equivalence, i.e. a late and early version, denote the form $\simeq_E^R$ and $\simeq_L^R$ between open terms respectively (Hennessy and Lin, 1995).

TRACE OVER TOPOLOGICAL SPACES

A branch labeled by (R, α) goes from node m to n which denotes $m \xmapsto{R,\alpha} n$, where R is a binary equivalence relation and the Greek letter α ranges over a set Act of actions, in an above-mentioned STGR over a topological space (A, T_A) of many value-passing processes based on CCS with infinite assignments. The action α is expressed as c ? X.t, where the residual after the symbolic action c ? X, namely t, will in general contain free occurrences of x, so the term of the nodes of a STGR is open one. We define a trace of action sequences over their topological spaces in concurrent security protocols to distinguish between their regular participants and penetrators in the light of the fact that a principal in a security protocol creates and transmit a message containing a new value, later receiving it back in a different cryptographic context (Crazzolara and Winskel, 2001).

An alphabet Σ is a union of all sets of variables fv (n) for all transitions such as $m \xmapsto{R,\alpha} n$ in a STGR, where there exists:

$$fv(n) \subseteq bv(\alpha) \cup fv(m)$$

And Σ^* is the set of all strings written in the alphabet Σ. An independency relation I then induces a binary relation ~ on Σ^*, where:

$$uv(u, v \in \Sigma^*)$$

if and only if there exist x , y ∈ Σ^* and a pair:

$$<a,b> \in I(a,b \in \Sigma)$$

Such that:

$$u = xaby$$

And:

$$v = xbay$$

Here, the reflexive, symmetric and transitive closure of the above-mentioned binary relation ~ on Σ^* is considered as the trace.

Definition 3: The trace is defined as the reflexive, symmetric and transitive closure of a binary relation ~ on Σ^* over a topological space (A, T_A) of many value-passing processes based on CCS with infinite assignments in a STGR and is denoted by $\equiv_D$, where the dependency relation D is induced by the independency relation I. The trace monoid, commonly denoted as M (D), is defined as the quotient monoid:

$$\mathbb{M}(D) = \Sigma^*/\equiv_D$$

Note that the transitive closure simply implies that u ≡ v if and only if there exists a sequence of strings:

$$(\omega_0, \omega_1, \cdots, \omega_n)$$

Such that:

$$u \sim \omega_0\ v \sim \omega_n$$

And:

$$\omega_i \sim \omega_{i+1}\ (0 \le i < n)$$

And the dependency relation D induced by the independency relation I is an equivalence relation on Σ^*. Clearly, different dependencies will give different equivalence relations. The homomorphism:

$$\phi_D : \Sigma^* \to \mathbb{M}(D)$$

Is commonly referred to as the natural homomorphism or canonical homomorphism. The trace is achieved by the above-mentioned definition 3 in a STGR of concurrent security protocols.

CONCLUSION

In this study, we define the trace over topological spaces for value-passing processes based on CCS with infinite assignments in a STGR of concurrent security protocols. Here, the STGR denotes a symbolic transition graph with a binary equivalence relation $R \in \mathbb{R}$ which is a variant of symbolic transition graphs, i.e. STG. To be dissimilar to the original STG, the STGR and its symbolic operational semantics rely on quotient spaces of value-passing processes which are commonly induced by a binary equivalence relation over topological spaces.

Our STGR contributes to modeling for concurrent security protocols by partitioning domains of their messages in order to reduce computational complexities to prove their safety properties. Furthermore, the regular participants or penetrators of the security protocol must be distinguished by the trace defined over topological spaces in concurrent security protocols, as we know, a principal in a security protocol creates and transmit a message containing a new value, later receiving it back in a different cryptographic context. The further work is needed for the trace construction algorithm of STGRs and the verification algorithm of safety properties of security protocols.

ACKNOWLEDGMENTS

This research is supported by the Anhui Provincial Natural Science Foundation of China under grant No. 090412057 and the Provincial Key Projects of Excellent Youth Talents Fund of the Institutions of Higher Education in Anhui Province of China under grant No. 2011SQRL 113ZD.

REFERENCES

Crazzolara, F. and G. Winskel, 2001. Events in security protocols. Proceedings of the 8th ACM Conference on Computer and Communications Security, November 5-8, 2001, Philadelphia, pp: 96-105.

Fabrega, F.J.T., J.C. Herzog and J.D. Guttman, 1999. Strand spaces: Proving security protocols correct. J. Comput. Security, 7: 191-230.

Guttman, J.D. and F.J. Thayer, 2002. Authentication tests and the structure of bundles. Theoret. Comput. Sci., 283: 333-380.

Hennessy, M. and H. Lin, 1995. Symbolic bisimulations. Theoret. Comput. Sci. 138: 353-389.

Hennessy, M. and H. Lin, 1996. Proof systems for message-passing process algebras. Formal Aspects Comput., 8: 397-407.

Li, Y., J.G. Jiang and H.B. Wang, 2009. Formal analysis of non-repudiation protocol by spi. J. Communi., 30: 94-98.

Lin, H., 1996. Symbolic Transition Graph with Assignment. In: CONCUR'96: Concurrency Theory: 7th International Conference Pisa, Italy, August 26-29, 1996 Proceedings, Montanari, U. and V. Sassone (Eds.). Springer, Berlin, Heidelberg, pp: 50-65.

Milner, R., 1999. Communicating and Mobile Systems: The p-Calculus. Cambridge University Press, University of Cambridge.

Milner, R., 2009. The Space and Motion of Communicating Agents. Cambridge University Press, Cambridge.

Wang, H., Y. Zhang and Y. Li, 2006. A diagram of strand spaces for security protocols. J. Comput. Res. Develop., 43: 2062-2068.