Revisiting Enforceable Security Policies

Naoyuki Nagatou
Department of Computer Science,
Graduate School of Science and Engineering,
Ritsumeikan University,
1-1-1 Noji-higashi, Kusatsu, Shiga,
525-8577, Japan.
Phone, Fax: +81-77-564-6210
nagatou@fse.cs.ritsumei.ac.jp

Abstract

We algebraically characterize a class of enforceable security policies by execution monitoring using a modal logic. We regard monitors as processes in Milner’s CCS and security policies as formulas in the modal logic. We show that a set of processes occurring in a monitor must be within the greatest fixed point for the formula, following Schneider’s definition on execution monitors. We also consider monitors that can derive some sequences from a single captured action sequence. To discuss such monitors, we introduce variables ranging over sets of processes in CCS. We then show that there is fixed points under the extension. This work may help us to understand such monitors to detect covert channels at run time and to analyze safety properties for multithreads, which need to examine multiple paths.

Keywords: execution monitoring, safety property, security policy.

1 Introduction

Execution monitors are mechanisms that capture a single sequence of actions such as states or system calls of a target, (for instance processes, programs and threads), and enforce a policy by terminating the target if the sequence violates a policy. Schneider [8] provides that the policies enforced by the mechanisms have a characteristic of Lamport’s safety properties. The monitors can be presented Büchi-like automata that recognize safety properties and are called security automata. Also, the class of these policies is said to be EM-enforceable, otherwise editing automata [2] and shallow history automata [4] are suggested. These monitors usually observe and examine a single sequence. However we may be able to develop monitors that deduce sequences from an observed one (Figure 1). Therefore, we consider monitors that seem to examine multiple sequences and characterize a class of enforceable security policies in these.

The abovementioned papers discuss several characterizations of enforceable security policy classes. We first discuss an algebraic characteristic, based on a modal logic. A monitor is an implementation of a security policy, and is abstracted by processes in Milner’s CCS [7]. A formula is a representation corresponding to a security policy. In this situation, we state that the set of subprocesses of the monitor process must be the greatest solution of an equation $Z \equiv \phi_Z$, where $\phi_Z$ is a formula contain variable $Z$. In addition, we introduce a symbol with special processes into CCS, to represent multiple sequences. The symbol is a representation of a set of processes and denotes a set of states of sequences deduced from one sequence. We then state that there is also the greatest fixed point of the function under extension. That is, there are monitors that can explore multiple sequences. These monitor may be able to

Figure 1. An aspect of a monitor that examines multiple sequences
be applied to run-time detection of covert channels and to run-time safety analysis of multithreads. We present states in the sequences, corresponding to the symbol. The structure of this paper is as follows. Related studies are summarized in Section 2. In Section 3, we explain several formal frameworks to prepare for algebraically characterizing a class of enforceable policies with EM. In Section 4, we derive a characteristic of enforceable security policies according to Schneider’s definition. In Section 5 and Section 6, we extend CCS and show that fixed solutions on modal formulas exist under the extension. Section 7 provides conclusion.

2 Related Works


Schneider [8] developed the possibility of execution monitors (e.g., reference monitors and Security Automata SFI Implementation [3]). The monitors work by capturing an action, which a program executes, and enforcing policies by terminating the program if the action violates a policy. The class of these mechanisms is called Execution Monitoring (EM). A policy for EM is specified by predicating on modal formulas exist under the extension. Section 7 provides conclusion ability to deduce multiple sequences from a single sequence, based on process algebra, while the above studies discuss monitors on a single sequence.

3 Relationships Between Monitor and Security Policy

We describe several notions (process and formula) to discuss the main results of this paper and relationships between these. Figure 2 illustrates the intuitive relationships.

3.1 Monitor and Process

We suppose that a monitor is an implementation of a security policy. Let $P_r$ be a set of possible states, $Act$ a set of actions, and $→⊆ Pr × Act × Pr$ a transition relation, which for $(p, a, q) ∈ Pr × Act × Pr$, we write $p a → q$, a monitor $M$ is presented as a tuple $(Pr, Act, →)$. We then abstract $M$ with a process $E$ like CCS. Where, $Pr$ is a set of subprocesses of $E$, $Act$ is a set of actions captured by the monitor, and the transition is defined as semantics of operations in Figure 3.

Processes, and a set of subprocesses of the processes, are defined as follows:

$$E ::= \bar{a}.E | a.E | E+E | E | E | P | E \setminus K,$$

where $+$ denotes a summation of two processes, $∥$ denotes a composition of two processes, $\setminus K$ denotes a restriction (where $K ∈ Act$), and constant processes are ranged over by $P$. We express actions as $a, b, c, ⋯$, their co-actions as $\bar{a}, \bar{b}, \bar{c}, ⋯$, and a set of these as $Act$, a monitor can capture actions in.

The set of subprocesses in process $E$ is inductively
follows. We provide the semantics of formulas by defining induc-

tion on the structure a set \( \| \phi \|_V \) of processes that have \( \phi \)

relative to the valuation \( V \). Given a valuation \( V \), the meaning is shown in Figure 4, where \( P = \text{Sub}(E) \). We also

write \( E \models V \phi \) when \( E \in \| \phi \|_V \). The notation of the subset of processes \( P \) with property \( \phi \) is refined: \( \| \phi \|_P = \{ E \in P \mid E \models V \phi \} \). For each \( Z \), it is expected that \( V(Z) \) is a

subset of \( P \). Valuations may be revised by a set \( E \) of pro-
cesses. The useful notation is \( V[E[Z]] \), which represents the valuation similar to \( V \), except \( E \) is assigned to \( Z \). In all that follows \( V \) and \( E \) are omitted when they are clear.

3.2 Security Policy and Modal Formula

We suppose that a modal formula represents security policies as targets must have properties. Modal formulas are defined as follows: let \( I \) be a set of identifiers

\[
\phi ::= \text{tt} \mid \text{ff} \mid \phi_1 \lor \phi_2 \mid \phi_1 \land \phi_2 \mid [a] \phi \mid (a) \phi \mid Z,
\]

where \( a \in \text{Act} \) and \( Z \in I \). The syntactic meaning of \( Z \) is specified by assigning a formula to each identifier. The semantic meaning is provided by assigning a set of processes to each identifier. The assignment is defined by a valuation function \( V : I \rightarrow 2^{\text{Sub}(E)} \).

We provide the semantics of formulas by defining inductively

on the structure a set \( \| \phi \|_V \) of processes that have \( \phi \)

relative to the valuation \( V \). Given a valuation \( V \), the meaning is shown in Figure 4, where \( P = \text{Sub}(E) \). We also

write \( E \models V \phi \) when \( E \in \| \phi \|_V \). The notation of the subset of processes \( P \) with property \( \phi \) is refined: \( \| \phi \|_P = \{ E \in P \mid E \models V \phi \} \). For each \( Z \), it is expected that \( V(Z) \) is a

subset of \( P \). Valuations may be revised by a set \( E \) of pro-
cesses. The useful notation is \( V[E[Z]] \), which represents the valuation similar to \( V \), except \( E \) is assigned to \( Z \). In all that follows \( V \) and \( E \) are omitted when they are clear.

\[
\begin{align*}
\| \phi \lor \phi_2 \|_V &= \| \phi_1 \|_V \lor \| \phi_2 \|_V \\
\| \phi \land \phi_2 \|_V &= \| \phi_1 \|_V \land \| \phi_2 \|_V \\
\| [a] \phi \|_V &= \{ E \in P \mid V[E[Z]] \models \phi \} \\
\| (a) \phi \|_V &= \{ E \in P \mid \exists F : E \rightarrow F \land F \in \| \phi \|_V \} \\
\| Z \|_V &= V(Z) \\
\| \text{tt} \|_V &= P \\
\| \text{ff} \|_V &= \emptyset
\end{align*}
\]

Figure 4. Semantics of a formula

We can define a function on \( 2^P \), using \( \phi \). Formula \( \phi \)
determines the function \( f_V[\phi, Z] : 2^P \rightarrow 2^P \) with respect to

\( V \) and \( Z \). It, when applied to \( E \subseteq P \), is \( \| \phi \|_{V[E[Z]]} \). That is,

\( f_V[\phi, Z](E) = \| \phi \|_{V[E[Z]]} \).

\( V \) is omitted when it is clear.

4 Derived Characteristics of Enforceable Policies

We analyze characteristics of EM-enforceable policies

under the modal logic in the previous section. Not all policies

can be enforced by EM. An EM monitor can catch only

one single finite sequence of actions but cannot use information

on actions a target may execute in the future to determine if a target violates policy. This provides the following definition:

**DEFINITION 1**

- monitors cannot get future information on the run, and
- monitors must discover violation in a finite sequence.

This definition provides that if a sequence violates a policy at some time, then any extensions of the sequence continue to violate after that time.

We consider a fail run "\( a_1 a_2 \cdots \in \text{Act}^{\omega} \)" as an example.

the aspect is illustrated in Figure 5. Let one of fail states be

\( E_i \), and \( E_i \not\models V[E_i[Z]] \). All later states must fail on property

\( \phi \) with respect to the set of processes \( E \). For each state \( E_j \)
such that \( j \geq i \), \( E_i \not\models V[E_i[Z]] \Rightarrow E_j \not\models V[E_j[Z]] \). That is,

\( E_i \not\models V[E_i[Z]] \Rightarrow E_j \not\models V[E_j[Z]] \). Moreover, \( E_i \not\models f[\phi, Z](E_i) \Rightarrow E_j \not\models f[\phi, Z](E_j) \). Using \( f[\phi, Z] \). We derive the following formal condition from the monitor condition.

**THEOREM 2**

If a policy is EM-enforceable then a greatest fixed point on a formula corresponding to the policy exists and then monitor process \( E_i \) itself must be in the greatest fixed point.
Given subprocesses $E$, valuation $V$, property $\phi$ and monitor process $E$. Let $E_i = f^i[\phi, Z](E)$, from the conditions of the monitors, for some $E_i \subseteq \text{Sub}(E)$,

$$f[\phi, Z](E) = E_1$$
$$f[\phi, Z](f[\phi, Z](E)) = E_2$$
$$\vdots$$
$$f[\phi, Z] \ldots (f[\phi, Z](E))) = E_i$$

$i$ times

$$f[\phi, Z] \ldots (f[\phi, Z](E))) = E_k$$

$k$ times

$$f[\phi, Z] \ldots (f[\phi, Z](E))) = E_j$$

$j$ times

$$(a)$$

$$E \xrightarrow{a_1} E_1$$
$$E \xrightarrow{a_2} E_2$$
$$\vdots$$
$$E \xrightarrow{a_k} E_1 \xrightarrow{a_2} \ldots \xrightarrow{a_k} E_k$$

$$\vdots$$

$$E \xrightarrow{a_k} E_1 \xrightarrow{a_2} \ldots \xrightarrow{a_k} E_k \xrightarrow{a_k} \ldots \xrightarrow{a_k} E_j$$

$$(b)$$

Figure 5. A fail transition on a run $"a_1a_2\ldots" \in \text{Act}^\omega$

Proof

Given subprocesses $E$, valuation $V$, property $\phi$ and monitor process $E$. Let $E_i = f^i[\phi, Z](E)$, from the conditions of the monitors, for some $E_i \subseteq \text{Sub}(E)$,

$$E \not\in E_i \Rightarrow$$
$$\forall j \geq i : E \not\in \nu f^{j-1}[\phi, Z](E) \phi \text{ or } E_j \not\in E,$$

where $i$ and $j \in \mathbb{N}$, the set of natural numbers. Then

$$E \not\in E_i \Rightarrow$$
$$\forall j \geq i : \exists k \leq j : \exists F \in f^{k-1}[\phi, Z](E).$$
$$F \not\in \nu f^{j-1}[\phi, Z](E) \phi \text{ or } E_j \not\in E.$$

The contraposition is

$$\exists j \geq i : \forall k \leq j : \forall F \in f^{k-1}[\phi, Z](E) :$$
$$F \not\models \nu f^{j-1}[\phi, Z](E) \phi \text{ and } E_j \not\in E$$

$$\Rightarrow E \not\in E_i,$$

from the definition of $f[\phi, Z]$.

$$\exists j \geq i : \forall k \leq j :$$
$$\forall F \in f^{k-1}[\phi, Z](E). F \in f^{k}[\phi, Z](E) \text{ and } E_j \not\in E$$

$$\Rightarrow E \not\in E_i,$$

and then,

$$\exists j \geq i : \forall k \leq j.$$  
$$f^{k-1}[\phi, Z](E) \subseteq f^{k}[\phi, Z](E) \text{ and } E_j \not\in E$$

$$\Rightarrow E \not\in E_i.$$

For all index $i$, the left side of (1) gives

$$E \subseteq f[\phi, Z](E) \subseteq \ldots \subseteq f^j[\phi, Z](E) \ldots .$$

That is, $f^j[\phi, Z]$ is continuous. Hence, the greatest fixed point is

$$\bigcup_n f^n[\phi, Z](E).$$

From above, we conclude that a monitor as an implementation of policy must be in the greatest fixed point of $f[\phi, Z]$.

EXAMPLE 3

As an example we consider a simple policy $Z \equiv (a)tt \land [a]Z$, where equation $\phi \equiv \psi$ means that both have the same semantics. Let $E$ be a process shown in Figure 6.

$$f[(a)tt \land [a]Z, Z](\{a.E\}) = \{a.E\}$$
$$f^2[(a)tt \land [a]Z, Z](\{a.E\}) = \{a.E\}$$

$$\vdots$$

Thus, the greatest solution of the equation is $\{a.E\}$. The initial process which is always capable of performing $a$-transition is in $\nu Z.(a)tt \land [a]Z$. This aspect is shown in Figure 7.
In general, we cannot guess the form of targets. However, if we can assume that both monitors and targets are image-finite, then the monitors can precisely enforce a policy as per Theorem 2.2 in [5]. Given two processes $E$ and $F$ as a start state, $E \sim F$, and $E$ and $F$ having the same properties is equivalence, where $\sim$ indicates observational equivalence. Hence, the monitor guarantees that targets have the same properties represented by a policy and that the monitor conservatively enforces the policy if they do not. Conservative enforcement means that if a target violates a policy at some state, then the target does not have the property. But we have no idea to have the property in states before the state. Clearly whenever $E$ cannot synchronize with $F$, then $E \not\sim F$, and $F$ does not have the property that $E$ has.

5 Extension of CCS

We discuss monitors that explore multiple sequences. Such monitors need to hold several possible states in itself. To represent this, we extend CCS to introduce a special variable $X$, ranging over sets of processes and distinguishable from variables for recursive definitions in CCS, and then show that $f[\phi, Z]$ is monotonic under the extension. $X$ is a set of processes in CCS; how this set is chosen determines the information deduced by a monitor.

We define a transition of $X$ as

$$E_1 \xrightarrow{\alpha} E_1' E_2 \xrightarrow{\alpha} E_2' \ldots E_n = E_n' \quad \text{Variable} X \xrightarrow{\alpha} X'$$

where $E_i \in X$. For example, let $X$ be $\{E_1, E_2, E_3\}$. A transition of $X$ appears as follows

$$E_1 \xrightarrow{\alpha} E_1' E_2 \xrightarrow{\alpha} E_2' E_3 = E_3$$

$$X \xrightarrow{\alpha} X'$$

Here, we introduce an axiom:

$$E = E \quad \text{NoTransition}.$$  

This is explicitly distinguished from deadlocks such as STOP, $0(zero)$ and NIL; it means to allow transitions remaining particular states. Some monitors may need to hold previous states. The no transition rule indicates that the monitor holds the previous state to compare a current state with a previous state.

In addition, we define an equivalence relation $\sim$ between two processes $X$ and $X'$ as

$$X \sim X' \Longleftrightarrow \forall E \in X : \exists E' \in X' : E \sim E' \text{ and } \forall E \in X' : \exists E' \in X : E \sim E'$$

Further, $\text{Sub}(X)$ is defined as

$$\text{Sub}(X) = \bigcup_{E \in X} \text{Sub}(E).$$

We define a relation between $X$ and a formula in the modal logic as

$$X \models \phi \Longleftrightarrow \forall E \in X : E \models \phi,$$

and

$$||\phi||_P^{E[Z]} = \bigcup_{E_i \in X} \{ E_i \models_{V[Z]} \phi \} \cup \{ E \in P | E \models_{V[Z]} \phi \}.$$  

If $\forall E \in X : E \in ||\phi||_P^{E[Z]}$ then we write $X \in ||\phi||_P^{E[Z]}$.

6 Greatest Fixed Points in Extended CCS

We state that greatest solution of the equations exists in extended CCS. From the fact, it is concluded that a monitor deducing multiple paths must be at least formed in extended CCS.

**Theorem 4**

Given formula $\phi$, fixed points of $F[\phi, Z]$ exists in extended CCS.

**Proof**

We state that $f[\phi, Z]$ is monotonic on extended CCS. That is, let $E, F \subseteq P$,

$$\text{if } E \subseteq F \text{ then } f[\phi, Z](E) \subseteq f[\phi, Z](F).$$

Also, it is sufficient to discuss on $X$ in a subset of processes.

We first consider the case that $\phi$ does not contain $Z$. If $\phi$ does not contain $Z$, then $f[\phi, Z]$ is constant. Thus, $f[\phi, Z](E) = f[\phi, Z](F)$, and then

$$f[\phi, Z](E) \subseteq f[\phi, Z](F)$$

whenever $E \subseteq F$.

Next, we describe the case when $\phi$ contains $Z$. This is divided into several subcases by a structure on $\phi$. When $\phi = Z$

$$X \in f[\phi, Z](E)$$

iff $X \in ||\phi||_P^{E[Z]}$

iff $\bigcup_{E_i \in X} \{ E_i \models_{V[Z]} \phi \} \cup \{ E \in P | E \models_{V[Z]} \phi \}$. 

- 231 -
If we assume that $E \subseteq F$ then
\[
X \in \bigcup_{E_i \in X} \{E_i | E_i \models_{V[E/F]} \phi \} \cup \{ E \in P | E \models_{V[E/F]} \phi \}.
\]
Hence, $f(\phi, Z)(E) \subseteq f(\phi, Z)(F)$ whenever $E \subseteq F$.

In the case that $\phi = [a] \psi$, let $X \in f([a] \phi, Z)(E)$.

\[
X \in f([a] \psi, Z)(E)
\]
iff \[
X \in \| [a] \psi \|_{V[E/Z]} \]
iff \[
X \in \bigcup_{E_i \in X} \{ E_i | E_i \models_{V[E/Z]} [a] \psi \}
\]
iff \[
X \in \bigcup_{E_i \in X} \{ E_i | \forall F_i : E_i \models_{V[E/Z]} F_i \}
\]
From above, for any formula $\phi$, $f(\phi, Z)(E) \subseteq f(\phi, Z)(F)$.

We conclude that $f(\phi, Z)$ is also monotonic under extended CCS with $X$. Then, from Tarski’s theorem [11] it has the greatest fixed point.

Although the following examples have not yet been formally shown we guess that our framework can discuss those monitors.

**EXAMPLE 5**

We describe run-time multipath analysis of safety properties, due to Sen et al [9, 10]. Let two threads be $T_1, T_2$, each performs "e1 e3", "e2 e4". Their monitor observes actions of both and deduces multiple paths from one observed sequence. Respectively corresponding $a$, $b$, $c$, $d$ to $e1$, $e2$, $e3$, $e4$ in the example in [9], three sequences "abcd", "acbd" and "abcde" are deduced from one, for example "abcd", under the following order conditions (see Figure 8), where to perform action $a$ before $b$ writes $a < b$. The order of execution of actions is supposed as follows:

\[
a < b,
\]
\[
a < c,
\]
\[
b < d.
\]

Also, these three sequences preserving the orders succeed, otherwise reach bad states. Let the policy in the article be $z = (\langle abcd \rangle tt \land [abcd] Z) \lor (\langle acbd \rangle tt \land [acbd] Z) \lor (\langle abde \rangle tt \land [abcde] Z), \langle abcd \rangle$ abbreviates $\langle a \rangle \langle b \rangle \langle c \rangle \langle d \rangle$. A solution of the equation is shown Figure 9. Their monitor looks as observing "abcde" by queuing actions although really observes sequence "abcd". Let correspond a set of monitor states to $X$, the aspect of transitions of the monitor is shown in Figure 10. So, we distinguish a sequence that changes states in the monitor and a actually observed sequence.

\[1\text{In this case, all deduced sequences succeed.}\]
Figure 9. The diagram of the monitor process

\[ X = \{a.b.c.d.E, \ a.b.d.c.E, \ a.c.b.d.E\} \]

We consider a fail case in which the policy is \(\{abcd\}tt \lor \{abcd\}Z\). "abcd" and "abcd" are not recognized and the target that performs "abcd" is terminated by that monitor although "abcd" realizes the formula. We think that it is correct since their monitor supposes that all deduced sequences must have the property.

EXAMPLE 6

We describe run-time covert channel analysis (CCA). Given a set \(traces(S)\) of possible sequence of system \(S\) and a set of low-level actions \(L \subseteq Act\). According to [12] a set of sequences LLES, call low-level equivalence set, is defined as \(LLES(\alpha, S) = \{\beta \in traces(S) | \beta = (\alpha[L])\}\). And a security property is defined as follows:

\[ \forall \alpha \in traces(S) : Q(LLES(\alpha, X)) \]

where \(|L|\) means to preserve actions included in \(L\) and \(Q\) as a predicate that define a set of successful sequences. We regard LLES(\(\alpha, S\)) as a sequence of \(X_1 X_2 \cdots X_n\) and \(Q\) as a predicate on a set of processes:

\[ traces(S) \iff \phi, \]

\[ LLES(\alpha, S) \iff X_1 \cdots X_n \]

where \(n\) is the length of \(\alpha\), \(\alpha \iff\) an observed sequence, \(Q \iff\) the same processes exists in \(X\).

\(traces(S)\) is corresponded to \(\phi\) since both present the set of possible sequences of the system. Then let security properties be that

\[ if \ P \models \phi \ then \ P \setminus H \ such \ that \ P \setminus H \models \phi \ exists. \]

The existence is represented by existence of \(E'\) in \(X\) such that \(E \sim E'\). For example, let an observed sequence "abcd" increasing from "abc" and \(\phi = (\langle abcd \rangle tt \lor \langle acbd \rangle) \land \langle abdc \rangle tt\) of which a solution is the process shown in Figure 11. Given \(L = \{a, b, d\}, H = \{c\}\) and the monitor as Figure 11 the aspect of transitions of the monitor is shown in Figure 12. When the monitor observes "abcd" and reaches the state \(X_4, E\) that is translated with "abcd" and "abcd" exists in this.

7 Conclusion and Future Work

We have discussed an algebraic characteristic of enforceable policies by applying execution monitoring. We regard a monitor as a process like CCS, and a policy as a formula in the modal logic. We then show that, given \(\phi\) as a policy, if its policy is enforceable, the monitor process \(E\) is in the greatest fixed point of a function \(f[\phi, Z]\) on a set of processes. Also, we extend CCS using variables ranging over sets of processes to present a monitor that can compute other action sequences from a observed finite sequence. Under the extension, we state that run-time analysis for covert channels is a predicate that define a set of successful sequences. We regard LLES(\(\alpha, S\)) as a sequence of \(X_1 X_2 \cdots X_n\) and \(Q\) as a predicate on a set of processes:

\[ traces(S) \iff \phi, \]

\[ LLES(\alpha, S) \iff X_1 \cdots X_n \]

where \(n\) is the length of \(\alpha\), \(\alpha \iff\) an observed sequence, \(Q \iff\) the same processes exists in \(X\).

\(traces(S)\) is corresponded to \(\phi\) since both present the set of possible sequences of the system. Then let security properties be that

\[ if \ P \models \phi \ then \ P \setminus H \ such \ that \ P \setminus H \models \phi \ exists. \]

The existence is represented by existence of \(E'\) in \(X\) such that \(E \sim E'\). For example, let an observed sequence "abcd" increasing from "abc" and \(\phi = (\langle abcd \rangle tt \lor \langle acbd \rangle) \land \langle abdc \rangle tt\) of which a solution is the process shown in Figure 11. Given \(L = \{a, b, d\}, H = \{c\}\) and the monitor as Figure 11 the aspect of transitions of the monitor is shown in Figure 12. When the monitor observes "abcd" and reaches the state \(X_4, E\) that is translated with "abcd" and "abcd" exists in this.

7 Conclusion and Future Work

We have discussed an algebraic characteristic of enforceable policies by applying execution monitoring. We regard a monitor as a process like CCS, and a policy as a formula in the modal logic. We then show that, given \(\phi\) as a policy, if its policy is enforceable, the monitor process \(E\) is in the greatest fixed point of a function \(f[\phi, Z]\) on a set of processes. Also, we extend CCS using variables ranging over sets of processes to present a monitor that can compute other action sequences from a observed finite sequence. Under the extension, we state that run-time analysis for covert channels is a predicate that define a set of successful sequences. We regard LLES(\(\alpha, S\)) as a sequence of \(X_1 X_2 \cdots X_n\) and \(Q\) as a predicate on a set of processes:

\[ traces(S) \iff \phi, \]

\[ LLES(\alpha, S) \iff X_1 \cdots X_n \]

where \(n\) is the length of \(\alpha\), \(\alpha \iff\) an observed sequence, \(Q \iff\) the same processes exists in \(X\).

\(traces(S)\) is corresponded to \(\phi\) since both present the set of possible sequences of the system. Then let security properties be that

\[ if \ P \models \phi \ then \ P \setminus H \ such \ that \ P \setminus H \models \phi \ exists. \]

The existence is represented by existence of \(E'\) in \(X\) such that \(E \sim E'\). For example, let an observed sequence "abcd" increasing from "abc" and \(\phi = (\langle abcd \rangle tt \lor \langle acbd \rangle) \land \langle abdc \rangle tt\) of which a solution is the process shown in Figure 11. Given \(L = \{a, b, d\}, H = \{c\}\) and the monitor as Figure 11 the aspect of transitions of the monitor is shown in Figure 12. When the monitor observes "abcd" and reaches the state \(X_4, E\) that is translated with "abcd" and "abcd" exists in this.
Figure 12. An aspect of run-time covert channel analysis

and run-time analysis for multithreads may be available. In run-time safety analysis, to examine multiple paths is necessary because the order of observing actions cannot be assumed, and in run-time covert channel analysis because of low level equivalence set. As future work, we try to formalize those cases in our framework. Also, the modal logic in Section 3.2 may need to express relation on processes in $X$ since the comparability between processes in $X$ corresponding to $Q$ is out of this.

References


