Gained and Excluded Classified Actions by Dynamic Security Policies *

Damas P. Gruska

Institute of Informatics, Comenius University
Mlynska dolina, 842 48 Bratislava, Slovakia
gruska@fmph.uniba.sk

Abstract. Dynamic security policies and formalisms for expressing information on private actions obtained by observing public ones are presented. Two sets of private actions are considered. The set of actions which execution is guaranteed according to observations and the set of actions which execution is excluded according to observations of public actions. Moreover, we consider also intruders which have limited memory capacity to record these sets during an attack as well as policies which change due to elapsing of time.

Key words: dynamic security policy, gained and excludes actions, information flow, security, non-interference

1 Introduction

Information flow based security properties assume an absence of any information flow between private and public systems activities. This means, that systems are considered to be secure if from observations of their public activities no information about private activities can be deduced. Such security properties could be quantified to avoid the unrealistic requirement of absolutely none information flow and hence to express an amount of possibly "leaked secrecy" in several ways. For example, Shannon's information theory was applied for simple imperative languages (see [CHM07,CMS09]) or for process algebras (see [Gru08]). Another possibility is to exploit probabilistic theory as it was used for process algebras in [Gru09] or by expressing subsets of private actions which occurrence (and not occurrence) can be deduced by an intruder who can observe public behaviour of the system [Gru11]. All above mentioned concepts are based on a static security policy, i.e. policy which is not changed during executions. This approach seems to be rather restrictive for applications where their security policies (classification, declassification etc.) change dynamically during runtime. Hence, there is a growing research and a number of papers devoted to dynamic security policies tailored for various formalizations and computational paradigms. For instance, in the case of imperative programs, security policy requires that values of classified variables could not be obtained by observing public ones, what can be formalizes by an equivalence relation on values of program's variables. In the case of a dynamic security policy, this relation can change during a computation (see, for example [DHS15]). In general, a

* Work supported by the grant VEGA 1/1333/12.

dynamic security property permits different information flows at different points during program/system's execution.

The aim of this paper is to formulate dynamic security policies for security concepts based on the sets of gained and excluded private actions [Gru11]. In this case, a dynamic security policy defines a set of private actions at a given state of execution. That means that the set of private actions is not fixed but it can change dynamically during system's execution. We study the resulting concepts and we show how they are related to traditional static security properties as well. Later, we consider also so called limited intruders (intruders with limited storage to record obtained information) who always try to be prepared for declassifications of private actions but have to take into account also new classification of non-public, so called invisible, actions. We will define also a special class of dynamic security policies which change only with elapsing of time.

The paper is organized as follows. In Section 2 we describe the timed process algebra TPA which will be used as a basic formalism. In Section 3 we formalize sets of gained and excludes private actions with respect to a static security policy. In Section 4 we define gained and excludes private actions with respect to dynamic security policies. The next section is devoted to limited intruders and time dynamic security policies. Section 6 contains discussion and plans for a future work.

2 Timed Process Algebra

In this section we define Timed Process Algebra, TPA for short. TPA is based on Milner's CCS (see [Mil89]) but the special time action t which expresses elapsing of (discrete) time is added (see also [Gru10]). The presented language is a slight simplification of the Timed Security Process Algebra (tSPA) introduced in [FGM00]. We omit the explicit idling operator ι used in tSPA and instead of this we allow implicit idling of processes. Hence processes can perform either "enforced idling" by performing t actions which are explicitly expressed in their descriptions or "voluntary idling". But in the both cases internal communications have priority to action t in the case of the parallel operator. Moreover we do not divide actions into private and public ones as it is in tSPA. TPA differs also from the tCryptoSPA (see [GM04]). TPA does not use value passing and strictly preserves *time determinacy* in case of choice operator $+$ what is not the case of tCryptoSPA.

To define the language TPA, we first assume a set of atomic action symbols A not containing symbols τ and t, and such that for every $a \in A$ there exists $\overline{a} \in A$ and $\overline{\overline{a}} = a$. We define $Act = A \cup \{\tau\}, Actt = Act \cup \{t\}$. We assume that $a, b, \ldots$ range over A, $u, v, \ldots$ range over Act, and $x, y \ldots$ range over $Actt$. Assume the signature $\Sigma = \bigcup_{n\in\{0,1,2\}} \Sigma_n$, where

$$\Sigma_0 = \{Nil\}$$
$$\Sigma_1 = \{x. \mid x \in A \cup \{t\}\} \cup \{[S] \mid S \text{ is a relabeling function}\}$$
$$\cup\{\backslash M \mid M \subseteq A\}$$
$$\Sigma_2 = \{|, +\}$$

with the agreement to write unary action operators in prefix form, the unary operators $[S], \backslash M$ in postfix form, and the rest of operators in infix form. Relabeling functions, $S : Actt \to Actt$ are such that $\overline{S(a)} = S(\bar{a})$ for $a \in A, S(\tau) = \tau$ and $S(t) = t$.

The set of TPA terms over the signature Σ is defined by the following BNF notation:

$$P \ ::= \ X \ \mid \ op(P_1, P_2, \ldots P_n) \ \mid \ \mu X P$$

where $X \in Var$, Var is a set of process variables, $P, P_1, \ldots P_n$ are TPA terms, $\mu X -$ is the binding construct, $op \in \Sigma$.

The set of CCS terms consists of TPA terms without t action. We will use a usual definition of opened and closed terms where μX is the only binding operator. Closed terms which are t-guarded (each occurrence of X is within some subexpression $t.A$, i.e. between any two t actions only finitely many non timed actions can be performed) are called TPA processes. Note that Nil will be often omitted from processes descriptions and hence, for example, instead of $a.b.Nil$ we will write just $a.b$.

We give a structural operational semantics of terms by means of labeled transition systems. The set of terms represents a set of states, labels are actions from $Actt$. The transition relation $\to$ is a subset of TPA $\times$ $Actt$ $\times$ TPA. We write $P \xrightarrow{x} P'$ instead of $(P, x, P') \in \to$ and $P \not\xrightarrow{x}$ if there is no P' such that $P \xrightarrow{x} P'$. The meaning of the expression $P \xrightarrow{x} P'$ is that the term P can evolve to P' by performing action x, by $P \xrightarrow{x}$ we will denote that there exists a term P' such that $P \xrightarrow{x} P'$. We define the transition relation as the least relation satisfying the inference rules for CCS (see [Mil89]) plus the following inference rules:

$$\frac{}{Nil \xrightarrow{t} Nil} \ A1 \qquad \frac{}{u.P \xrightarrow{t} u.P} \ A2$$

$$\frac{P \xrightarrow{t} P', Q \xrightarrow{t} Q', P \mid Q \not\xrightarrow{\tau}}{P \mid Q \xrightarrow{t} P' \mid Q'} \ Pa1 \qquad \frac{P \xrightarrow{t} P', Q \xrightarrow{t} Q'}{P + Q \xrightarrow{t} P' + Q'} \ S$$

Here we mention the rules that are new with respect to CCS. Axioms $A1, A2$ allow arbitrary idling. Concurrent processes can idle only if there is no possibility of an internal communication ($Pa1$). A run of time is deterministic (S). In the definition of the labeled transition system we have used negative premises (see $Pa1$). In general this may lead to problems, for example with consistency of the defined system. We avoid these dangers by making derivations of τ independent of derivations of t. For an explanation and details see [Gro90]. Regarding behavioral relations we will work with the timed version of weak trace equivalence. Note that here we will use also a concept of observations which contain complete information which includes also τ actions and not

just actions from A and t action as it is in [FGM00]. For $s = x_1.x_2.\ldots.x_n, x_i \in Actt$ we write $P \xrightarrow{s}$ instead of $P \xrightarrow{x_1}\xrightarrow{x_2} \cdots \xrightarrow{x_n}$ and we say that s is a trace of P. The set of all traces of P will be denoted by $Tr(P)$.

We will write $P \stackrel{x}{\Rightarrow}_M P'$ for $M \subseteq A$ iff $P \xrightarrow{s_1}\xrightarrow{x}\xrightarrow{s_2} P'$ for $s_1, s_2 \in (M \cup \{\tau\})^\star$ and $P \stackrel{s}{\Rightarrow}_M$ instead of $P \stackrel{x_1}{\Rightarrow}_M \stackrel{x_2}{\Rightarrow}_M \cdots \stackrel{x_n}{\Rightarrow}_M$. Instead of $\Rightarrow_\emptyset$ we will write $\Rightarrow$ and instead of $\Rightarrow_{\{h\}}$ we will write $\Rightarrow_h$. By $Succ(P)$ we will denote the set of all successors P' of P i.e. processes for which $P \xrightarrow{s} P'$ for some $s \in Actt^*$. By $s|_B$ we will denote the sequence obtained from s by removing all actions not belonging to B. By $s' \sqsubseteq s$ we will denote that s' is a subsequence of s i.e. s' can be obtained by removing some actions from s. We conclude this section with definitions of M-bisimulation and M-trace equivalence.

Definition 1. *Let $(TPA, Actt, \rightarrow)$ be a labelled transition system (LTS). A relation $\Re \subseteq TPA \times TPA$ is called a* M-bisimulation *if it is symmetric and it satisfies the following condition: if $(P, Q) \in \Re$ and $P \xrightarrow{x} P', x \in Actt$ then there exists a process Q' such that $Q \stackrel{\widehat{x}}{\Rightarrow}_M Q'$ and $(P', Q') \in \Re$. Two processes P, Q are M-bisimilar, abbreviated $P \approx_M Q$, if there exists a M-bisimulation relating P and Q.*

Definition 2. *The set of M-traces of process P for $M, M \subseteq A \cup \{t\}$ is defined as $Tr_{tM}(P) = \{s \in (A \cup \{t\})^\star | \exists P'.P \stackrel{s}{\Rightarrow}_M P'\}$. Instead of $Tr_{t\emptyset}(P)$ we will write $Tr_t(P)$. Two processes P and Q are M-trace equivalent ($P \approx_{tM} Q$) iff $Tr_{tM}(P) = Tr_{tM}(Q)$.*

Note that for $M = \emptyset$ these two notions correspond to weak bisimulation and weak trace equivalence (see [Mil89]) and they will be denoted by $\approx$ and $\approx_t$, respectively.

3 Gained and Excluded Classified Actions

In this we recall (with slightly modified notations) concepts of gained and excluded classified actions (see [Gru11]). We suppose that all actions are divided into two groups, namely public (low level) actions L and private (high level) actions H i.e. $A = L \cup H, L \cap H = \emptyset$. Moreover, we suppose that $H \neq \emptyset$ and $L \neq \emptyset$ and that for every $h \in H, l \in L$ we have $\overline{h} \in H, \overline{l} \in L$. Hence the division is given by the set of high level actions. To denote sequences of public actions, i.e sequences consisting of actions from $L \cup \{t\}$ and sequences of private actions from H, we will use notation $\tilde{l}, \tilde{l'}, \ldots$ for sequences from $(L \cup \{t\})^\star$ (note that elapsing of time - i.e. t action is also a public action) and $\tilde{h}, \tilde{h'}, \ldots$ for sequences from $H^\star$, respectively. The set of actions could be divided to more than two subsets, what would correspond into more levels of classification. All the following concepts could be naturally extended to such setting.

First we define a set of private actions which occurrence can be learned by an intruder who see a process to perform a sequence of public actions $\tilde{l}$ (we will call such action as gained actions).

Definition 3. *Let $P \in TPA$ and $\tilde{l} \in Tr_{tH}(P)$. Then the occurrence of the set of private action which can be gained about P by public observing $\tilde{l}$ is defined as follows:*

$$g_H(P, \tilde{l}) = \{h | h \in H, P \stackrel{\tilde{l}}{\not\Rightarrow}_{H\setminus\{h\}}\}.$$

According to Definition 3 the set of private actions $g_H(P, \tilde{l})$ is the one which has to be performed by P if an intruder sees P to perform public actions $\tilde{l}$.

Example 1. Let $P = l_1.h.l_2.Nil + l_1.l_2.Nil$ and $P' = l_1.h.h'.l_2.Nil + l_1.h.l_2.Nil$. Let $\tilde{l} = l_1.l_2$ then we have $g(P, \tilde{l}) = \emptyset, g(P', \tilde{l}) = \{h\}$.

Definition 4. *Let $P \in TPA$. Then the occurrence of the set of private action which can be excluded by observing P performing public action $\tilde{l}$ (i.e. $\tilde{l} \in Tr_{tH}(P)$) is defined as follows:*

$$e_H(P, \tilde{l}) = \bigcap_{P \stackrel{\tilde{l}}{\Rightarrow} M} H \setminus M.$$

If we have that $e_H(P, \tilde{l}) = \emptyset$ that means that an intruder after observing $\tilde{l}$ cannot exclude occurrence of any private action.

There is no direct correlation between sets $g(P, \tilde{l})$ and $e(P, \tilde{l})$ since there are processes such that for one is the former set empty and the later nonempty and vice versa. If both of them are empty, that means, that an intruder can learn practically nothing on private actions by observing process P and seeing it to perform $\tilde{l}$. In some sense $g(P, \tilde{l})$ and $e(P, \tilde{l})$ are complementary as it is stated in the following proposition (see [Gru11]).

Proposition 1. *For every process P and every $\tilde{l}, \tilde{l} \in Tr_{tH}(P)$ it holds $g(P, \tilde{l}) \cap e(P, \tilde{l}) = \emptyset$ and $\emptyset \subseteq g(P, \tilde{l}) \cup e(P, \tilde{l}) \subseteq H$.*

Now we are prepared to formulate how much information can be gained by observing public activities $O, O \subseteq L^*$ of a process. The formal definition follows.

Definition 5. *Let $P \in TPA$. By $g_H^O(P)$ we will denote the set of private actions which occurrence by P can be gained (detected) by an intruder observing sequences of public actions from O as*

$$g_H^O(P) = \bigcup_{\tilde{l} \in O} g_H(P, \tilde{l}).$$

We say that no private information can be gained by observing P by O if $g_H^O(P) = \emptyset$.

Now we can define a set of private actions which occurrence could be excluded by the set of observations O.

Definition 6. *Let $P \in TPA$. Then the occurrence of the set of private action which executions could be excluded by the set of observations $O, O \subseteq Tr_{tH}(P)$ is defined as follows:*

$$e_H^O(P) = \bigcup_{\tilde{l} \in O} e_H(P, \tilde{l}).$$

For a given process P the size of sets $g_H^O(P)$, $e_H^O(P)$ with respect to the size of H and O, give us another quantification of security. For example, small $|O|$ and big $|g_H^O(P)|$ and/or $|e_H^O(P)|$ indicate a rather low level of security.

4 Dynamic Security Policies

Division of actions to public and private ones is based on fixed (static) security policy which is not changed during system computation. This approach seems to be rather restrictive for applications where their security policies (classification, declassification etc.) change dynamically during runtime. In the presented framework by a policy we mean some set M of actions which are supposed to be private at the given points during program/system's execution. By dynamic security policy D we mean a partial mapping which assigns to every process P and sequence of actions some policy i.e. subset of actions. Moreover we require that D is uniquely defined with respect to weak bisimulation.

Definition 7. *By dynamic security policy we mean partial mapping D, $D : TPA \times Actt^* \to 2^{Actt}$ such that $D(P, s) = D(P', s)$ whenever $P \approx P'$.*

Hence by $D(P, s)$ we denote the set of actions which are private after the execution of s by P if $P \xrightarrow{s}$ otherwise $D(P, s)$ is not defined. Now we can define the set of gained actions with respect to dynamic security policy D. In this case an intruder gains private actions given by a security policy valid at the moment and from this set removes actions which are declassified. To do so we divide every execution trace to (maximal) intervals during which a policy is not changed. The formal definition follows.

Definition 8. *Let $P \in TPA$. Let $s \in Tr_t(P)$ and $s = s_1 \ldots s_k$ such that $s_i = x^i_1 \ldots . x^i_{n_i}$ for $i = 1, \ldots, k$. such that $P_i \xrightarrow{s_i} P_{i+1}$ where $P = P_1$ i.e. $P_i \xrightarrow{x^i_1} P^1_i$, $P^1_i \xrightarrow{x^i_2} P^2_i$, ..., $P^{n_i-1}_i \xrightarrow{x^i_{n_i}} P_{i+1}$ such that $D(P_i, \epsilon) = H_i$ and also $D(P_i, x^i_1 \ldots . x^i_j|_{Actt \setminus H_i}) = H_i$ for every j, $1 \leq j < n_i$. Moreover, we suppose that $H_i \neq H_{i+1}$. Let $\tilde{l} = \tilde{l_1} \ldots , \tilde{l_k}$ where $\tilde{l_i} = s_i|_{Actt \setminus H_i}$. Then we define*

$$g_D(P, \tilde{l_1}) = g_{H_1}(P, \tilde{l_1})$$
$$g_D(P, \tilde{l_1} \ldots . \tilde{l}_{n+1}) = (g_D(P, \tilde{l_1} \ldots . \tilde{l}_n) \cap H_{i+1}) \cup g_{H_{i+1}}(P_i, \tilde{l}_{n+1}).$$

Definition 9. *Let $P \in TPA$. By $g^O_D(P)$ we will denote the set of private actions which occurrence by P can be gained (detected) by an intruder observing sequences of public actions from O under dynamic security policy D as*

$$g^O_D(P) = \bigcup_{\tilde{l} \in O} g_D(P, \tilde{l}).$$

We say that no private information can be gained by observing P by O if $g^O_D(P) = \emptyset$.

We can define partial ordering between dynamic security policies.

Definition 10. *Let D, D' are two dynamic security policies. We say that D is stronger than D' (denoted by $D' \preceq D$) if for every P and s it holds $D'(P, s) \subseteq D(P, s)$.*

Both the ordering of dynamic security properties as well as ordering (by inclusion) of sets of observations influence resulting sets of gained action as it is stated by the following proposition.

Proposition 2. *For every process P, sets of obsevations O, O' and dynamic policies D, D' such that $O \subseteq O'$ and $D' \preceq D$ it holds $g_D^O(P) \subseteq g_D^{O'}(P)$ and $g_D^O(P) \subseteq g_{D'}^O(P)$, respectively.*

Proof. Sketch. The first part of the proof follows directly from Definition 9. The second part follows from Definitions 8 and 10.

Now we can show how the property "no private information can be gained by observing P" by dynamic security policy is related to persistent variant of absence-of-information-flow property, so called Strong Nondeterministic Non-Interference (SNNI, for short). We recall its definition (see [FGM00]). Process P has SNNI property (we will write $P \in SNNI$) if $P \setminus H$ behaves like P for which all high level actions are hidden for an observer. To express this hiding we introduce hiding operator $P/M, M \subseteq A$, for which it holds if $P \xrightarrow{a} P'$ then $P/M \xrightarrow{a} P'/M$ whenever $a \notin M \cup \bar{M}$ and $P/M \xrightarrow{\tau} P'/M$ whenever $a \in M \cup \bar{M}$. Moreover, process has persistent SNNI property (denoted by PSNNI) if also all its successors have SNNI property. Formal definition of SNNI and PSNNI follows.

Definition 11. *Let $P \in TPA$. Then $P \in SNNI_H$ iff $P \setminus H \approx_t P/H$ and $P \in PSNNI_H$ iff $P' \in SNNI_H$ for every $P', P' \in Succ(P)$.*

The persistent variant of SNNI property is stronger than SNNI itself as it is expressed by the next proposition.

Proposition 3. $PSNNI_H \subset SNNI_H$.

Proof. Clearly $PSNNI_H \subseteq SNNI_H$. Let $P = (l.l.Nil + h.e.(h.l.Nil + l.Nil))$. Then it is easy to check that$P \notin PSNNI_H$ but $P \in SNNI_H$.

Now we are ready to formulate relationship between PSNNI property and the set of gained actions under a constant dynamic security policy.

Proposition 4. *If $P \in PSNNI_H$ then $g_{D_H}^O(P) = \emptyset$ for constant dynamic policy D_H which assigns the set H to every process and sequence of actions.*

Proof. The main idea. Let $P \in PSNNI_H$ and suppose that $g_{D_H}^O(P) \neq \emptyset$. Hence there exists $P', P' \in Succ(P)$ and such that and subsequence o' of some observation o from O, $o \in Tr_{tH_i}(P')$ and $h, h \in H$ and such that $P' \overset{o'}{\not\Rightarrow}_{H \setminus \{h\}}$. But then there exists sequence s which contains o' and h such that $s \in Tr_t(P'/H)$ but $s \notin Tr_t(P' \setminus H)$ i.e. $P' \setminus H \not\approx_t P/H$ i.e. $P' \notin SNNI_H$ and hence $P \notin PSNNI_H$.

The inverse of the previous proposition does not hold as it shows the following example.

Example 2. Let $P = \sum_{1 \leq i \leq n} h_i.l.Nil$ and $H = \{h_1, \ldots, h_n\}$. Then $g^O_{D_H}(P) = \emptyset$ but P has not PSNNI property since $P \setminus H \not\approx_t P/H$. Indeed $P \setminus H$ cannot perform the sequence of action $\tau.l$ while P/H can perform it and an intruder seeing l can deduce that a private action was performed.

Corollary. Let $D \preceq D_H$ and $P \in PSNNI_H$ then $g^O_D(P) = \emptyset$.

Proof. The proof follows from Proposition 2 and 4.

Now we will define sets of excluded private action first for a single observation and later for a set of observations.

Definition 12. *Let $P \in TPA$. Let $s \in Tr_t(P)$ and $s = s_1 \ldots . s_k$ such that $s_i = x^i_1 \ldots . x^i_{n_i}$ for $i = 1, \ldots, k$. such that $P_i \stackrel{s_i}{\rightarrow} P_{i+1}$ where $P = P_1$ i.e. $P_i \stackrel{x^i_1}{\rightarrow} P^1_i$, $P^1_i \stackrel{x^i_2}{\rightarrow} P^2_i$, ..., $P^{n_i-1}_i \stackrel{x^i_{n_i}}{\rightarrow} P_{i+1}$ such that $D(P_i, \epsilon) = H_i$ and also $D(P_i, x^i_1 \ldots . x^i_j|_{Actt \setminus H_i}) = H_i$ for every j, $1 \leq j < n_i$. Moreover, we suppose that $H_i \neq H_{i+1}$. Let $\tilde{l} = \tilde{l_1} \ldots , \tilde{l_k}$ where $\tilde{l_i} = s_i|_{Actt \setminus H_i}$. Then we define*

$$e_D(P, \tilde{l_1}) = e_{H_1}(P, \tilde{l_1})$$
$$e_D(P, \tilde{l_1} \ldots . \tilde{l}_{n+1}) = (e_D(P, \tilde{l_1} \ldots . \tilde{l}_n) \cap H_{i+1}) \cup e_{H_{i+1}}(P_i, \tilde{l}_{n+1}).$$

Definition 13. *Let $P \in TPA$. By $g^D(P)$ we will denote the set of private actions which occurrence by P can be excluded by an intruder observing sequences of public actions from O under dynamic security policy D as*

$$e^O_D(P) = \bigcup_{\tilde{l} \in O} e_D(P, \tilde{l}).$$

We say that no private information can be excluded by observing P by O if $g^O_D(P) = \emptyset$.

For excluded action we can formulate the similar property which holds for gained actions. Also the proof is similar.

Proposition 5. *For every P' and D, D' such that $D' \preceq D$ it holds $e^O_D(P) \subseteq e^O_{D'}(P)$.*

There is no direct correlation between sets $g_D(P, \tilde{l})$ and $e_D(P, \tilde{l})$ since there are processes such that the former set is empty and the later one is nonempty and vice versa. If the both of them are empty, that means, that an intruder can learn practically nothing on private actions by observing process P to perform $\tilde{l}$ under the dynamic security policy D. In some sense $g_D(P, \tilde{l})$ and $e_D(P, \tilde{l})$ are complementary as it is stated in the following proposition.

Proposition 6. *For every process P it holds $g_D(P, \tilde{l}) \cap e_D(P, \tilde{l}) = \emptyset$ and $\emptyset \subseteq g_D(P, \tilde{l}) \cup e_D(P, \tilde{l}) \subseteq \bigcup_{s, s \sqsubseteq \tilde{l}} D(P, s)$.*

Proof. Let $h \in g_D(P, \tilde{l})$. We now that every execution of sequence of visible action $\tilde{l}$ has to contain h for some $P' \in Succ(P)$ i.e. if $P' \stackrel{\tilde{l}}{\Rightarrow}_M$ then $h \in H_i$. That means $h \notin H_i \setminus M$ i.e. $h \notin e_D(P, \tilde{l})$. As regards the second part of the proposition let us consider process $P = l.Nil + h.l.Nil$. We have $g(P, l) = e(P, l) = \emptyset$. If we consider $H = \{h\}$ then we see that $g_{D_H}(P, \tilde{l}) \cup_{D_H} (P, \tilde{l}) = H$ i.e. $\subseteq$ cannot be replaced by $\subset$ in general.

We could further quantify levels of security by relating size of $g^O_D(P), e^O_D(P)$ to the size of O and D as it was suggested at the end of the previous section.

5 Limited Intruders and Variants of Dynamic Security Policies

In this section we will assume intruders which are aware of dynamicity of security policy and try to learn as much as can be done with limited amount of memory where obtained information can be recorded. That means, (s)he tries to record also invisible but declassified actions for the case that they become classified in the future. First, let us consider three types of actions. Low level (public) actions, which are always visible, and the rest of the actions is called invisible (I). Moreover, the invisible actions could contain private (high level) actions, i.e. $A = L \cup I, L \cap I = \emptyset$ and $H \subseteq I$. Now we will consider dynamic security policies for which $D(P, s) \subseteq I$. That means that neither the set of visible nor invisible actions are changed by D. Only the set of high level actions can be changed but they are always invisible. We assume that $|I| \geq n$. We suppose intruders who try to learn information about all classified actions but also about declassified actions which are not visible at the moment for the case that they will become classified in the future. Moreover, we assume that s(he) has only limited storage to record obtain information and hence in the case that the amount of this information is bigger then memory capacity, then some part of this already obtained information has to be forgotten. To model this we define a mapping which assigns to given set of classified actions M, capacity of storage n and given set N a set of subsets of N, with size n (or less if the capacity of storage is not reached) preferably containing actions from M. The formal definition follows.

Definition 14. $\mathcal{F}^n_M(N) = \{N' |$ *where* $N' = N$ *if* $|N| \leq n$ *otherwise if* $|N \cap M| \geq n$ *then* $N' \subseteq N \cap M$ *such that* $|N'| = n$ *and if* $|N \cap M| < n$ *then* $N' \subseteq N$ *such that* $N \cup M \subseteq N'$ *and* $|N'| = n\}$.

Mapping $\mathcal{F}^n_M$ could be naturally extended for sets of sets. Let $T \subseteq 2^I$, then $\mathcal{F}^n_M(T) = \bigcup_{N \in T} \mathcal{F}^n_M(N)$. Now we reformulate Definitions 3, 8 and 9 for limited intruders.

Definition 15. *Let* $P \in TPA$ *and* $\tilde{l} \in Tr_{tI}(P)$. *Then the occurrence of the set of private action which can be gained about* P *by public observing* $\tilde{l}$ *is defined as follows:*

$$g_H(P, \tilde{l}, n) = \mathcal{F}^n_H(\{h | h \in I, P \stackrel{\tilde{l}}{\not\Rightarrow}_{I \setminus \{h\}}\}).$$

Definition 16. *Let $P \in TPA$. Let $s \in Tr_t(P))$ and $s = s_1 \dots s_k$ such that $s_i = x^i_1 \dots . x^i_{n_i}$ for $i = 1, \dots, k$. such that $P_i \stackrel{s_i}{\rightarrow} P_{i+1}$ where $P = P_1$ i.e. $P_i \stackrel{x^i_1}{\rightarrow} P^1_i$, $P^1_i \stackrel{x^i_2}{\rightarrow} P^2_i$, ..., $P^{n_i-1}_i \stackrel{x^i_{n_i}}{\rightarrow} P_{i+1}$ such that $D(P_i, \epsilon) = H_i$ and also $D(P_i, x^i_1 \dots . x^i_j|_{Actt \setminus H_i}) = H_i$ for every j, $1 \leq j < n_i$. Moreover, we suppose that $H_i \neq H_{i+1}$. Let $\tilde{l} = \tilde{l_1} \dots , \tilde{l_k}$ where $\tilde{l_i} = s_i|_{Actt \setminus H_i}$. Then we define*

$$g_D(P, \tilde{l_1}, n) = g_{H_1}(P, \tilde{l_1})$$
$$g_D(P, \tilde{l_1} \dots . \tilde{l}_{n+1}, n) = \mathcal{F}^n_{H_{i+1}}((g_D(P, \tilde{l_1} \dots . \tilde{l}_n) \cap H_{i+1}) \cup g_{H_{i+1}}(P_i, \tilde{l}_{n+1})).$$

Definition 17. *Let $P \in TPA$. By $g^O_D(P)$ we will denote the set of private actions which occurrence by P can be gained (detected) by an intruder observing sequences of public actions from O under dynamic security policy D as*

$$g^O_D(P, n) = \bigcup_{\tilde{l} \in O} g_D(P, \tilde{l}, n).$$

We say that no private information can be gained by observing P by O if $g^O_D(P) = \emptyset$.

Note that in the previous definition we do not apply $\mathcal{F}^n_M(H)$ since the whole concept is based on one time attacks. If $M \in g^O_D(P, n)$ than (there is a possibility that) an intruder can gain actions from M. Hence $E = \bigcup_{M \in g^O_D(P,n)} M$ is the set of possibly gained actions. Clearly, with a bigger memory an intruder can learn more as it is stated by the following proposition.

Proposition 7. *Let $n \leq m$. Then for every M, $M \in g^O_D(P, n)$ there exists M', $M' \in g^O_D(P, m)$ such that $M \subseteq M'$.*

Proof. The main idea. It is clear from Definition 14 that mapping $\mathcal{F}^n_M$ is monotonic with respect to parameter n.

If n sufficiently large limited and unlimited intruders can learn the same as it is stated by the following proposition.

Proposition 8. *Let $|\bigcup_s D(P, s)| \leq n$ for every P and s. Then $g^O_D(P, n) = g^O_D(P)$.*

Proof. The main idea. From Definition 14 we see that $\mathcal{F}^n_M(H)$ is identical function if $|\bigcup_s D(P, s)| \leq n$

For $|\bigcup_s D(P, s)| > n$ we cannot say, in general, whether $g^O_D(P, n)$ is equal to $g^O_D(P)$ or not. We cannot say this even if $|D(P, s)| > n$ for some s. We could define also sets of excluded actions for limited intruders in the similar way as it is done for gained actions. But instead of that we concentrate on special cases of dynamic security policies. First, we define finite dynamic security policies and then time dynamic security policies.

Definition 18. *We say that dynamic security policy is finite iff there exists set* $F = \{H_1, \ldots, H_m\}$ *such that for every* P *and* s *we have* $D(P, s) \in F$.

Definition 19. *By time dynamic security policy we dynamic security policy such that for every* P *and* s, s' *such that* $s|_{\{t\}} = s'|_{\{t\}}$ *it holds* $D(P, s) = D(P, s')$.

In the case of finite dynamic security policy D we could design process P_D which could "compute" the both sets of excluded and private actions for finite state processes in the following sense $h.g \in Tr_t(C[P]|P_D)$ iff $h \in g_D^O(P, n)$ and $h.e \in Tr_t(C[P]|P_D)$ iff $h \in e_D^O(P, n)$ where $C[]$ is an appropriate process context (see [Gru10]). In this case we obtain decidability of sets $g_D^O(P)$ and $e_D^O(P)$ (and their derivatives) for finite state processes which are undecidable in general.

Time dynamic security policies are useful for formalization of systems which allow partial classification/declassification of actions within some time windows. Study of these two specific dynamic security policies we leave for the further research.

6 Conclusions

We have presented several security concepts based on an information flow and dynamic security policies. They express which set of private actions was performed (gained sets) or which set of private actions could be excluded by an intruder observing systems public actions (excluded sets), taking into account dynamic security policies which can change sets of high level actions during runtime. The concepts offer a finer security notion with respect to traditional ones which usually only ensure that an intruder cannot learn that some private action was performed (for example, persistent variant of SNNI). Moreover, the sets of excluded and gained actions can be used for reduction of a space of possible private actions and if the reduction is significant then it really threatens systems security.

Concepts of gained and excluded private actions are complementary. Roughly speaking, only systems for which both the sets - gained and excluded private actions are empty could be considered fully secure with respect to a given dynamic security policy D and set of observations O. But since this is a very rare situation we have suggested how to numerically express corresponding level of security by relating size of sets of gained or excluded actions to the set of all appropriate actions and the size of O. That means, if the resulting measure is small enough the system can still be considered secure with respect to some given requirements. Later we have introduced the concept of limited intruders, i.e. intruders who have limited storage to record information obtained by observing public system activities. Such intruders could try to record also declassified but invisible action for the case that during execution they could become classified. We had to resolve the case when not all information on invisible actions could be recorded due to the lack of memory space which has an intruder at disposal.

In the future we plan, besides already mentioned research, to investigate also additional covert channels which could be exploited by an intruder. Particularly interesting are termination and divergence channels. They can be exploited by an intruder who can learn that the system is still working but does not react (for example, by power consumption). It might happen, for example, that the system is completely secure if an intruder cannot see termination (or divergence) and vice versa.

References

[CHM07] Clark D., S. Hunt and P. Malacaria: A Static Analysis for Quantifying the Information Flow in a Simple Imperative Programming Language. The Journal of Computer Security, 15(3). 2007.

[CMS09] Clarkson, M.R., A.C. Myers, F.B. Schneider: Quantifying Information Flow with Beliefs. Journal of Computer Security, to appear, 2009.

[DHS15] van Delft B., S. Hunt, D. Sands: Very Static Enforcement of Dynamic Policies. In Principles of Security and Trust, LMCS 9036, 2015.

[FGM00] Focardi, R., R. Gorrieri, and F. Martinelli: Information flow analysis in a discrete-time process algebra. Proc. 13^{th} Computer Security Foundation Workshop, IEEE Computer Society Press, 2000.

[GM04] Gorrieri R. and F. Martinelli: A simple framework for real-time cryptographic protocol analysis with compositional proof rules. Science of Computer Programming archive Volume 50, Issue 1-3, 2004.

[GM82] Goguen J.A. and J. Meseguer: Security Policies and Security Models. Proc. of IEEE Symposium on Security and Privacy, 1982.

[Gro90] Groote, J. F.: Transition Systems Specification with Negative Premises. Baeten, J.C.M. and Klop, J.W. (eds.), *CONCUR'90*, Springer Verlag, Berlin, LNCS 458, 1990.

[Gru11] Gruska D.P.: Gained and Excluded Private Actions by Process Observations. Fundamenta Informaticae, Vol. 109, No. 3, 2011.

[Gru10] Gruska D.P.: Process Algebra Contexts and Security Properties. Fundamenta Informaticae, vol. 102, Number 1, 2010.

[Gru09] Gruska D.P.: Quantifying Security for Timed Process Algebras, Fundamenta Informaticae, vol. 93, Numbers 1-3, 2009.

[Gru08] Gruska D.P.: Probabilistic Information Flow Security. Fundamenta Informaticae, vol. 85, Numbers 1-4, 2008.

[Mil89] Milner, R.: *Communication and concurrency.* Prentice-Hall International, New York, 1989.