Parallel Simulation of Dynamic Communication Processes on the Base of Probability Time Automata

Henryk Piech[1], Grzegorz Grodzki[1] and Aleksandra Ptak[2]
[1]*Institute of Computers and Information Sciences, Czestochowa University of Technology, Dabrowskiego 73, Czestochowa, Poland*
[2]*Department of Applied Informatics, Czestochowa University of Technology, Czestochowa, Poland*

Keywords: Parallel Conversion, Communication Security, Probability Time Automata.

Abstract: The proposition is connected with the research of the security or threatens referring to message decryption, user dishonesty, non-fresh nonce, uncontrolled information jurisdiction, etc. (security attributes), in network communication processes. Encrypted messages are usually sent in the form of protocol operations. Protocols may be mutually interleaving, creating the so called runs, and their operations can appear as mutual parallel processes. The investigation regards both particular security attributes and their compositions referring to more general factors, such as: concrete users, protocols, public keys, secrets, messages, etc. The abovementioned situation forms a conception about parallel strategy realized with the help of PTA and Petri net that includes the set of security tokens (attributes) in each node.

1 INTRODUCTION

The set of security communication attributes is presented in (Burrows et al., 1990). These elements are logically combined in the form of rules. The BAM and Hoare logic are used. The rules have a traditional form "if *conditions* then *conclusions*". Conditions are represented by protocol actions, whereas conclusions by attributes. Attribute corrections (modifications) are realized according to rules. It is necessary to regard the fact that the same attributes have a timed character, i.e. they lose their secure values in the activation time (Kwiatkowska et al., 2002). The proposed convention, in the second section, permits to change the activation time distance into probability values equivalent to a security level. The distributed form of investigation, according to chosen security factors, suggests using the parallel composition of the PTA node structure (Szpyrka and Szmuc, 2006). In general, calculation process organization is presented in the section devoted to the thread creation and the dynamic of their new designation (Tadeusiewicz, 2011). The acceleration estimators, according to several variants of the security analysis parallelization, are proposed in section 3. The selection and adaptation of the PTA and Petri net structure is presented in a general form in the last section.

2 DESCRIPTION OF SECURITY STATE WITH HELP OF PROBABILITY - TIME AUTOMATON NODES

The security state is assigned to communication run factors, such as: user, message, protocol, key, nonce, secret, etc. Generally, a communication run consists of interleaving protocols. Protocol is created as a sequence of operations built on the basis of users (sender, receiver, intruder), shared keys, nonces, secrets. The structure and operation (action) components are arguments exploited by rules (Burrows et al., 1990) to extend the set of security parameters. Same of such kind of parameters are included directly in protocol operations.

$$set\_sec1 = sp \subseteq OP,$$
$$set\_sec2 = r(sp \subseteq OP),$$
$$set\_sec = set\_sec1 \cup set\_sec2,$$

where:
set_sec1 - security parameters included in protocols operations,
set_sec2 - security parameters inferred from logic rules,
sp - security parameters,
$r = \{r_1, r_2, ..., r_k\}$ - the set of rules,

$OP = \{Op_1, Op_2, ..., Op_m\} = \{op_{1,1}, op_{1,2}, ...,$
$..., op_{m,l(m)}\}$ - the set of protocol operations,
set_sec - the full set of security parameters,
$Op(i)$ - set of operations in protocol i,
k - rule number,
m - protocols number,
$l(j)$ - protocol j operation number.

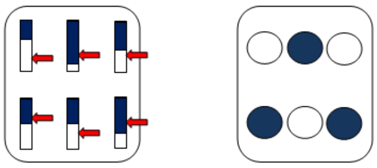

Figure 1: PTA node presentation - example of probability (left) and binary (right) form.

The task of a user consists in the selection of the set of security attributes, which will be the components of nodes of probability - time automaton (PTA). Components can be evaluated in two ways: as real and binary values. Real values express the probability of security parameters and binary values express the acceptable level of parameters. Therefore, it was decided to introduce a notation of tokens which will be adequate to node security components in the binary form.

Elements in frames mean selected security attributes. Arrows appoint security threshold levels of attributes. If component probability, represented by the white bar, is greater than security threshold then security token is acceptable, which is represented by the white ring in the right part of the figure. The example structure of the security node is presented in Fig. 2. Security attribute types infer from protocol logic (BAN, Hoare or PCL) formalisms (Burrows et al., 1990) which is partly described, by character of communication dealings, in the following way:
$A \leftrightarrow^K B$ - users A, B communicate via shared key K,
$\rightarrow^K A$ - user A has K as its public key,
$A \Leftrightarrow^Y B$ - users A and B share Y as a secret,
$\{X\}_K$ - the message X encrypted by key K,
$\{X\}_K^A$ - the message X encrypted by key K by user A,
$< X >_Y$ - the message X with a secret Y attached,

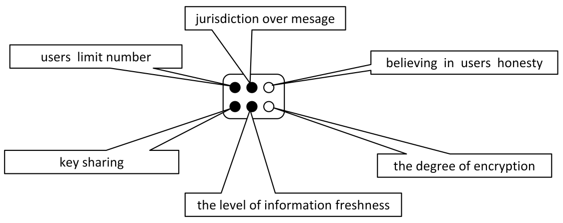

Figure 2: The structure of security node consisting of six attributes - example.

$A| \equiv X$ - user A believes the message X,
$A \triangleright X$ - user A sees the message X,
$A \triangleleft X$ - user A once send the message X,
$A| \Rightarrow X$ - user A has jurisdiction over X,
$\#(X)$ - the message is fresh.

At this point, we can to cite (Burrows et al., 1990) several rules from the logic of authentication protocols (BAN):
1. Authentication rule - Type I :
if $(A| \equiv ((A \leftrightarrow^K B),\ A \triangleright X_K)$ then $(A| \equiv (B \triangleleft X)$.
The rules can be interpreted as follows: if A and B shared key K and A sees message, then A believes that this message are from B.
2. Nonce rule
if $(A| \equiv \#(X),\ A| \equiv (B \triangleleft X))$ then $A| \equiv (B| \equiv X)$.
The rules can be interpreted as follows : if A believes that X is "current" and that B said X, then A believes that B believes X.
3. Jurisdiction rule:
if $(A| \equiv (B| \Rightarrow X),\ A| \equiv (B| \equiv X)$ then $A| \equiv X$
The rule can be interpreted as follows: if A believes that B has jurisdiction over X and A believes that B confirms X then A believes X.
4. Vision rule - Type I :
if $(A| \equiv (A \leftrightarrow^K B),\ A \triangleright \{X\}_K^C, C \neq A)$ then $A \triangleright X$.
The rule can be interpreted as follows: if A and B shared the key K and A sees the message X, encrypted by the shared symmetric key, and the encryption was done by another A user then A sees X.
5. Freshness rule :
if $\#(X)$ then $\#(X,Y)$.

The rule can be interpreted as follows: if X is fresh then $X \wedge Y$ is also fresh.
Due to the determined character of attribute number la and their binary form, the number of security state (level) is strictly defined and equal to 2^{la}. The created node, in the investigation process (accompanied to communication run realization), saves its structure but changes its values of attributes (and consequently the security state). Security level can only decrease. In the proposed parallel system for selected security elements (the so called main security factors) different, independently converted, security nodes are created. Generally, this situation is presented as in fig. 3.

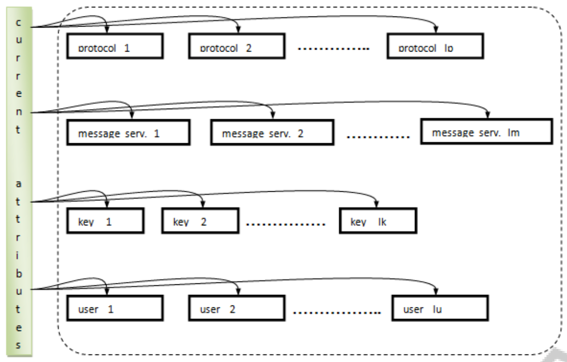

Figure 3: Security modules (security node structures for different main factors) are built on the basis of chosen attribute sets. Each module (bold frame) consists of a different set of attributes.

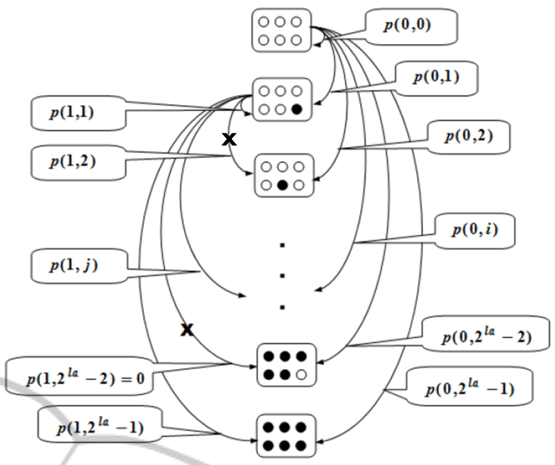

Figure 4: The scheme of probabilistic - time automaton for communication security investigation, where $p(i,j)$ - the probability of state changing: from state i to j; $j \geq i$.

3 PROBABILITY - TIME AUTOMATA AS COMMUNICATION SECURITY INVESTIGATION MODEL

It is proposed to use probability - time automata (PTA) and converted to them colored Petri nets as main tools for the investigation communication security according to selected main factors, such as: protocols, users, keys, messages, etc. The nodes presented in fig.1 will be the fundamental part of PTA (and Petri nets). Let us introduce the definition of the security state which will correspond to the automaton node.

Definition 1. A tuple (At, Th, Tk, na),
where:
At - security attribute set,
Th - the vector of the low level of feasible attribute values (thresholds),
Tk - security tokens,
na - the number of attributes, is the communication security state described as follows:
1. $At = \{at_1, at_2, ..., at_n\} \in [0,1]^n$ - the vector of attribute activation probabilities,
2. $Th = \{th_1, th_2, ..., th_n\} \in [0,1]^n$ - the vector of a threshold attribute activation (acceptation),
3. $Tk = tk_1, tk_2, ..., tk_n \in \{0,1\}^n$ - the binary vector of an attribute activation:
if $at_i \geq th_i$ then $tk_i = 1$ otherwise $tk_i = 0$.
Global structure of this automata is presented in fig.4. To regard the time parameter which is an intrinsic characteristic according to the security aspect, the following definition is proposed:

Definition 2. A probabilistic timed automaton PTA is a tuple of the form
$(L, l', X, \sum, inv, p)$, where:

- L is a finite set of locations,
- $l' \in L$ is the initial location,
- X is a finite set of clocks (for each attribute),
- $\sum$ is a finite set of possible steps, where $\sum_c \in \sum$ are declared as being current possible,
- the function $inv : L \rightarrow CC(X)$ is the invariant condition,
- the finite set $p \subseteq L \times CC(X) \times \sum \times Dist(2^X \times L)$ is the probabilistic edge relation.

A time state of a probabilistic timed automaton is a pair (l, v) where $l \in L$ and $v \in T^X$ are such that $v \in inv(l)$. Informally, the behavior of a probabilistic timed automaton can be understood as follows. The model starts in the initial location l' with all clocks set to 0, that is, in the state $(l', 0)$. In this, and any other state (l, v), there is a nondeterministic choice of either (1) making a discrete transition or (2) letting time pass. In case (1), a discrete transition can be made according to any probabilistic edge $(l, g, \sigma, p^*) \in p$ with source location l which is enabled; that is, the zone g is satisfied by the current clock valuation v. Then the probability of moving to the location l" and resetting all of the clocks in X to 0 is given by $p^*(X, l")$. In case (2), the option of letting time pass is available only if the invariant condition $inv(l)$ is satisfied while time elapses and there does not exist an enabled probabilistic edge with a current step. Note that a timed automaton (Alur and Dill, 1994), (Beauquier, 2003) is a probabilistic timed automaton for which every probabilistic edge $(l, g, \sum, p^*)$ is such that $p^* = \mu(X, l")$ (the point distribution assigning probability 1 to $(X, l")$) for some $(X, l") \in 2^X \times L$.

Additionally, the recognition of the number of users and their character (honesty, intruder) (Burrows et al., 1990) is wanted. These considerations are based on time influences on chosen security attributes; strictly on their level (value). Time influences may refer only to specific attributes, such as: keys, nonces, secrets. The investigation consists in finding the typical actions, being directly connected with attributes, in protocol operations and defining the way of their influences on the attribute value. So, for each message the time state will be defined as a set of pair(l_i, v_j), where i- the number of a node (operation in a run), j- the code of attribute. The location l_i will be referred to an automaton node which is equivalent a number of the realized protocol operation (strict operation in communication run). On this stage the communication run is defined.

Definition 3. The time sequence of protocols operations whose elements e_i are represented by actions set $A_i = \{a_1, a_2, ..., a_{lac(i)}\}_i$ correcting node attributes $At_i = \{at_1, at_2, ..., at_{la}\}_i$ (where i - the number of operation in communication run, j- the number of security attribute, $lac(i)$- the number of actions, la - the number of security attributes), which is equivalent automaton steps (nodes) and can belong to different protocols and messages is named the communication run:

$$oi(k,s) \rightarrow At_i(k,m), \tag{1}$$

where:
$o(k,s)$ - s-th operation in k-th protocol,
k - protocol number,
s - operation number in protocol,
m - message code.

It can be noticed that $e_i = \{o(k,s) | k = 1,2,...,lp, s = 1,2,...,ls(k)\}$, where lp - the number of protocols, $ls(k)$ - the number of operations in k protocol, and $\forall_k \forall e_{j>i}(s_{j,k} > s_{i,k})$, i.e. for given k protocol operations should save a given order. For different protocols $k \neq h \forall e_{j>i}((s_{j,k} > s_{i,h}) \vee (s_{j,k} < s_{i,h}) \vee (s_{j,k} = s_{i,h}))$. Each operation consist of actions $a_r^{\alpha(r,s)}$, where $\alpha(r,s) \in \{0,1\}$, r - the code of action. It can be said that $\alpha(r,s)$ refers to the present (activation) concrete action in s-th operation: if $\alpha(r,s) = 0$ action a_r is not present in s-th operation, otherwise r-th action is activated in this operation.

$$o(k,s) = \{a_{r(1)}^{\alpha(r(1),s)} a_{r(2)}^{\alpha(r(2),s)}, ..., a_{r(la(s))}^{\alpha(r(1),s)}\}, \tag{2}$$

where:
$r(1), r(2), ..., r(la(s))$ - the sequence of action codes in s-th operation,
$la(s)$ - the number of actions in s-th operation.

Generally, a number of actions cannot be ordered in different operations because only after the operation is finished the values of state security attributes will be modified. Therefore, it is proposed to create a stable list of actions which will be activated for a particular operation. Transition to new automaton node is realized by correction node attributes with the help of the action set of the current ended operation.

4 PARALLEL PROCESS OF CORRECTION COMMUNICATION SECURITY ATTRIBUTES

The corrections of attributes can be realized simultaneously. The new recognized action is used with the help of communication logic rules to activate the set of attributes. At this moment adequate processor units start to correct the clock and value of attributes. After the correction, the medicated attribute is sent to different processors analyzing communication security level in accordance with particular main factors. The estimation of acceleration inferring from parallelization can be defined as $acc = la * (1 + lmf)/2$, where lmf - the number of main factors. The number of main factors is the sum of number of selected protocols, messages, keys, users, nonce's (see fig.3). The main security factor(s) is (are) declared for the current action. Action usually influences one or several attributes. Analyzing security situation in network several (their number is e.g. equal lac) processors can serve set of communication actions. Therefore, the acceleration parameter will be estimated as interval in following way:

$$accn = [la * (1 + lmf)/2; lac * la * (1 + lmf)/2]. \tag{3}$$

The upper bound of acceleration (fig.11) is achieved when the sets of attributes, evoked by actions, are mutually independent: $set_at(i) \cup set_at(j) = \oslash$, i, j - the number of actions (Tudruj and Masko, 2005).

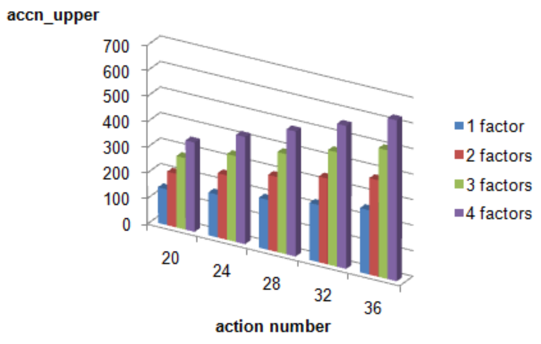

Figure 5: The upper bound of acceleration $accn$ in parallel security checking variant.

The stages of this algorithm are as follows:

1. action input,
2. the recognition of the attribute corrected by the action,
3. the recognition of the type of correction,
4. correction realization*,
5. go to the 3-rd point until the last attribute,
6. the recognition of the main factor activated by the action,
7. token structure creation for the main factor**,
8. security state estimation for the main factor**,
9. go to the 6-th point until the last factor,
10. auxiliary analysis (threaten state prognosis creation, the distribution of probabilities of transitions to the next stages)**,
11. go to the 1-st point until the last action***.

There are three stages of parallelization: *- simultaneous corrections of attributes, **- simultaneous main security factor analysis, ***- the simultaneous serving of actions.
The type of action influences is practically regarded by two forms of algorithm attribute corrections:
$mc = \{0,1\}$ - correction by multiplication by a given updating coefficient *MCC* in case logic and heuristic rules influence, $mc = 1$ - the activation this form of attribute correction, $mc = 0$ - the rejection this form of correction.
$ec = \{0,1\}$ - correction by exchanging to current level (represented by current coefficient value of *ECC*) in case of lifetime or users(intruders) influences.
Therefore, it is possible to use simultaneously two form of correction for single attribute. So, if $ec = 1$ then attribute value does not have to be increased:

$$at_{t=k+1}(i) \overset{mc=0,\ ec=0}{\rightarrow} at_{t=k}(i),$$

$$at_{t=k+1}(i) \overset{mc=1,\ ec=0}{\rightarrow} at_{t=k}(i) * MCC,$$

$$at_{t=k+1}(i) \overset{mc=0,\ ec=1}{\rightarrow} ECC,$$

$$at_{t=k+1}(i) \overset{mc=1,\ ec=1}{\rightarrow} \min\{at_{t=k}(i) * MCC, ECC\}.$$

The experiments have approved that heuristic rules that influence in specific cases (for example in multi usage of the same nonce) are more effective when correction is realized in the following way:

$$at_{t=k+1}(i) \overset{mc=1,\ ec=0}{\rightarrow} at_{t=k}(i) * (1 - MCC),$$

or

$$at_{t=k+1}(i) \overset{mc=1,\ ec=0}{\rightarrow} at_{t=k}(i) * (1 - at_{t=k}(i)).$$

The actual value of ECC, in case of lifetime type of influence, will be counted by formula:

$$ECC = 1 - et^{t_j - lt_i}, \qquad (4)$$

where:
t_i - the time of attribute activation,
lt_i - the attribute lifetime.
In reality, the time activity is transformed into probability attribute value, in accordance with the given attribute lifetime (fig.6).
The actual value of *ECC* in case of additional users (intruders) type of influence will be counted by formula:

$$ECC = if\ (nus < nht)\ then\ ECC = 1, \qquad (5)$$

else

$$ECC = e^{nht - nus},$$

where:
nus- the number of users (in the environment of main security factor),
nht - the number of honest users.
In reality, the time activity is transformed into probability attribute value, according with the given number of honest users (fig.7).

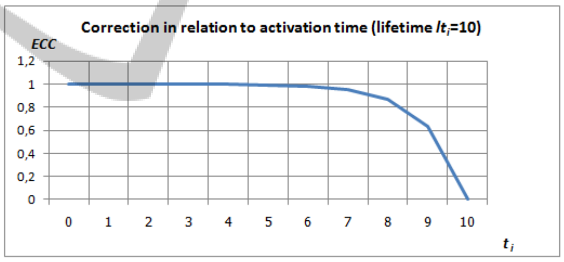

Figure 6: The action time influence on attribute.

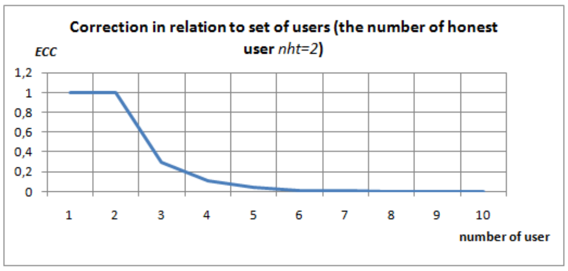

Figure 7: The action users influence on attribute.

5 CONCLUSIONS

The security investigation is acquiring an increasing importance with the growing network communication. Therefore, the problem of dynamic security estimation is increasingly grasping the interest of the data mining community (Szpyrka and Szmuc, 2006). The parallel approach guarantees not only the possibility to accelerate the reaction on impending threatens but also permits to treat chosen main security factors independently and simultaneously provide security attribute corrections as the result of the effect of influence on the same protocol actions (Tudruj and Masko, 2005).

REFERENCES

Alur, R. and Dill, D. L. (1994). A theory of timed automata. *Theoretical Computer Science*, 126:183–235.

Beauquier, D. (2003). On probabilistic timed automata. *Theor. Comput. Sci.*, 292(1):65–84.

Burrows, M., Abadi, M., and Needham, R. (1990). A logic of authentication. *ACM Trans. Comput. Syst.*, 8(1):18–36.

Kwiatkowska, M., Norman, G., Segala, R., and Sproston, J. (2002). Automatic verification of real-time systems with discrete probability distributions. *Theoretical Computer Science*, 282:101–150.

Szpyrka, M. and Szmuc, T. (2006). Verification of automatic train protection systems with rtcp-nets. In *Proceedings of the 25th International Conference on Computer Safety, Reliability, and Security*, SAFECOMP'06, pages 344–357, Berlin, Heidelberg. Springer-Verlag.

Tadeusiewicz, R. (2011). Introduction to inteligent systems, chapter no 1. In *The Industrial Electronic Handbook, Wilamowski B.M., Irvin J.D.(Eds.)*, pages 1–12. CRC Press Boca Raton.

Tudruj, M. and Masko, L. (2005). Towards massively parallel numerical computations based on dynamic smp clusters with communication on the fly. In *Parallel and Distributed Computing, 2005. ISPDC 2005. The 4th International Symposium on*, pages 155–162.