TELKOMNIKA Indonesian Journal of Electrical Engineering
Vol.12, No.1, January 2014, pp. 747 ~ 752
DOI: http://dx.doi.org/10.11591/telkomnika.v12i1.3206

A Dynamic Non-interference Trust Chain Model Based on Security Process Algebra

Wang Xiaoxing*, Kong Xiangying, Chen Xuebing
Jiangsu Automation Research Institute
No.18, Road Shenghu, District Xinpu, Lianyungang, Jiangsu, 222006, China
*corresponding author, e-mail: hbgzwxx@126.com

Abstract

Trust Chain is the key technology of Trusted Computing. For lack of comprehensive theoretical model of Trust Chain, A dynamic Trust Chain Model is proposed based on Security Process Algebra and Non-interference. Then, give the formal description and proof of the model. Finally, Modeling Intel TXT according to the new model semantics and verify the security attributes of the model by automated verification tool.

Keywords: *Trusted Computing, Trust Chain, Security Process Algebra, Non-interference*

1. Introduction

Since the concept of Trusted Computing is put forward by Trusted Computing Group (Trusted Computing Group, TCG), it has become one of the hotspots in information security study [1]. Trust chain is the key technology of trusted computing, but early trust chain is a static trust measurement, and its basic idea is: a Trusted Platform Module (Trusted Platform Module, TPM) acts as Trusted Root, step by step to measure BIOS, MBR, OS Loader, OS and applications, forming a chain of trust [2]. However, this static trust chain can only do integrity measurement at setup of the platform, so large servers which are always on for many months or years, can be attacked after the setup measurement and may not detect the damage, and through continuous reboot to achieve integrity measurement is impossible. Intel's LaGrande plan [3] and AMD's Presidio project [4] by improving CPU architecture, increase SENTER and SKINIT privileged instructions, respectively, to support repeatedly integrity measurements of the platform. TPM 1.2 specification introduced Dynamic Root of Trusted Measurement (DRTM), which is a trusted hardware, namely CPU with the new privileged instructions. TPM after receiving the instruction resets the last eight new PCRs (PCR 16~23) of dynamic Platform Configuration Register (PCR), based on this mechanism it authenticates security guide block and builds a controllable and trusted running environment. The trusted running environment can create multiple isolated security domains, namely Locality mechanism, the guest operating systems or applications running in different security domains respectively.

Currently trusted chain technology has been applied in commercial applications, however, most studies are about the static trust chain and trust chain also lacks better theoretical models. IBM developed IMA (Integrity Measure Architecture) [5] on measuring executable files, dynamic shared libraries, kernel module and dynamic link libraries in system to ensure the Integrity of the system; Tian Li-ye et al. propose a trust chain scheme with TPCM which combined the advantages of linear and radial transfer [6]; For lack of commission depth control in distributed environment, Xianchen Guo et al. [7] show a role-based trust management model in multi-domain environment. Zhao Jia, et al. [8] mapped non-interference domain to the process and put forward trust chain model based on non-interference and [9] Si Limin, et al. proposed a application-level trust chain model and the formal definition of trust in running, Kong Xiangying, et al. [10] put forward a dynamic intransitive non-interference trust chain transfer model; For the randomness in the process of trust chain transfer in system operations, Liu Changping, et al. [11] in established random model of trust chain by random process algebra as formal description language, but there was no model instance given and the model was yet to be further simulated to verify; Fu Ning, et al. [12] combined Pi calculus and Q algebra to propose a type of process algebra QPi which can describe trust status of system and Xu Mingdi,

Received May 15, 2013; Revised July 14, 2013; Accepted August 25, 2013

et al. [13] used security process algebra (SPA) to formally model trust chain interface, but the model was only about TCG static trust chain analysis and proof and does not take into account of the dynamic trust chain transfer in TPM 1.2 specification.

Non-interference, the information flow theory, was put forward by Goguen and Meseguer [14]. Rushby [15] introduced domain and established non-interference model based on state machine, and studied transitive non-interference and intransitive non-interference of information, but did not describe the domain as an entity. Non-interference effectively solves covert channel analysis and becomes the hotspots in information flow model research. Since Process Algebra was introduced into information flow theory study, it has become the main research tool on information flow theory. Security Process Algebra (SPA) [16], one of the important methods in analysis of information flow security, can analyze information flow security properties in multilevel security system. In this paper, for TPM 1.2 specification of dynamic trusted measurement, based non-interference theory, abstract domain in non-interference to the Locality of security domains and propose dynamic SPA non-interference trust chain model (SPA - NI model) to effectively express the dynamic characteristics of trust transfer in system runtime, and formal description of trust chain model and system trust in runtime theorem are given. Finally through model Intel TXT (Trusted Execution Technology) [3] by the model semantic, the information flow security properties of the model are verified.

2. The Dynamic Trust Chain Model SPA-NI

First of all, we give the definition of trust: trust is not only the integrity of the process metrics, but no leaking of high-level security information in process execution. The trust chain transfers by component instance as a unit and process is the basic unit of system resource allocation. In this paper, transfer the entities in trust transfer process into processes and the conversions of process operations and status are characterized by security process algebra and the interactions between processes are constrained by non-interference theory. The formal definition of SPA-NI model is given as follow.

2.1. SPA-NI Model Semantic

Definition 1 A SPA-NI system M is a five-tuple

$$M = \{P, A, S, OP, \rightarrow\} \quad (1)$$

Where,

P is the set of entities, namely system resource allocation units, which is descried as $\{p, q, \cdots\}$;

A is the set of actions. It is divided into three types: receive actions $I = \{a, b, c \cdots\}$, send actions $O = \{\overline{a}, \overline{b}, \overline{c} \cdots\}$ and internal action τ (internal computing or updating internal state values). $Act = I \cup O$ is the visible action sets and is expressed by a, b etc. Act_H is high security level action and Act_L is low security level action.

S is the set of states, which is expressed as $\{s_0, s_1, s_2, ..., s_n\}$. s_0 is the initial state, namely the state after PCRs reset when CPU send privileged instructions, which is absolutely trusted.

OP is the set of operations. It is expressed as $OP = \{., +, ||, |||, \setminus, /\}$. They denote prefix operator, selection operator, parallelism operator, alternating operator, limit operator and hidden operator respectively.

$\rightarrow \subseteq P \times A \times S$ is the set of state transfer, which describes the state change after execution of system actions. $s' = \rightarrow (p, a, s)$ denotes system state changes from s to s' after process p executes action a.

Definition 2 The grammar of process set P is defined as follow:

$$P = 0 \mid \alpha.p \mid p_1 + p_2 \mid p_1 \|_S \ p_2 \mid p \setminus A \mid p / A \mid v^t(a) \quad (2)$$

Where,

0 is the empty process, which never does any actions; $a.p$ expresses the system reaches process p after executes action a; $p_1 + p_2$ denotes performing actions of p_1 or p_2; $p_1 \|_S \ p_2$ means p_1 and p_2 are executed in parallel and synchronize the actions in set S; $p \setminus A$ denotes that p can execute the actions a if $\alpha \notin A$; p / A expresses all the actions in set A should be converted into internal actions; $v^t(a)$ denotes the execution time of action a.

Definition 3 The trace of a process is an action sequence α after the process executed, which is expressed as a three-tuple

$$Tr(p) = (\alpha, Me_p, t) \quad (3)$$

Where, $\alpha \in A, p \in P, Me_p$ is the boolean value of integrity. $Me_p = true$ denotes the trace of process p passing the integrity measurement, while, $Me_p = false$ denotes not passing the integrity measurement. t is the runtime of action sequence. The state of system changes from s_i to s_j after the process passes the integrity measurement and executes the action sequence, namely $s_j = \rightarrow (p, \alpha, s_i)$. s_j is called system reachable state.

Definition 4 $\forall p, q \in P$, call p and q are equivalent, iff $Tr(p) = Tr(q)$, and mark as $p \approx_{Tr} q$.

Definition 5 A process p is non-interference if for any subsets A_1 and A_2 of action sets, $check(A_1, A_2) = true$.

The function $check : A \times A \rightarrow \{true, false\}$ is non-interference checking function, which is defined as:

$$check(A_1, A_2) = \begin{cases} true, & if\ (p \setminus A_1) / A_2 = p / A_2 \\ false, & otherwise \end{cases} \quad (4)$$

For any two action subsets A_1 and A_2, if function check is true, then A_1 is non-interference to A_2, that is action subset A_1 could not deduce the executions on A_2.

2.2. Trust Chain Transfer Theorem

Definition 6 For $p \in P, \vee p_H \in P_H, S \subseteq Act_H$, *p* is dynamic trusted, iff

$$p \setminus Act_H \approx_{Tr} ((p \ \|_S \ p_H) / S) \setminus Act_H \quad (5)$$

$p \setminus Act_H$ is the view of low security level, and $((p \|_S \ p_H) / S) \setminus Act_H$ is low security level user's view of process *p* when *p* executes in parallel with any high security level process p_H. Formula (5) shows the two views are equivalent, that is to say low security level users could not deduce the high security level information by its view, so that protect high level information, therefore, we call the process *p* is dynamic trusted.

Definition 7 The state of system is trusted after execution of process *p*, iff
(1) process p is trusted;
(2) the trace $Tr(p)$ of *p* has passed integrity measurement, that is $Me_p = true$.

As the process and its trace are both trusted, so its execution is trusted, and then the system reaches a trusted state.

Definition 8 A system has the property of trust transfer, iff
(1) the initial system state is trusted;
(2) the state transfer $\rightarrow$ is trusted;

$\forall p \in P$, if p meets the non-interference property, that is to say for any two action subsets A_1 and A_2, $check(A_1, A_2) = true$. When $A_1 \subseteq Act_H$ and $A_2 \subseteq Act_L$, the high security actions is non-interference to low security actions, thus, the low security users could not deduce the high security actions only through the low security actions' execution, and then the system has the property of trust transfer.

Definition 9 A system M is a dynamic trusted system, iff
(1) the initial state of M is trusted;
(2) all the system reachable states are trusted.

When the system setup with a trusted state, if all the state conversions and reachable states are trusted, the system meets the dynamic trusted conditions. Now, we give system dynamic runtime trust theorem.

Theorem 1 A system M based on SPA-NI is trusted, iff
(1) M starts with DRTM;
(2) $\forall p \in P$ in system M meets trust check, namely $check(Act_H, Act_L) = true$.
(3) $\forall p \in P$ in system M, its trace $Tr(p)$ passes the integrity measurement $Me_p = true$.

Proof:

To prove the theorem, we just need prove the two conditions in Definition 9.

According to the definition of DRTM, the initial state s_0 is the state after CPU sends the privileged instruction to TPM and TPM resets the PCRs. For the reset action is atomic, so the initial state s_0 is absolutely trusted, and then the condition (1) is met.

For condition (2), as $\forall p \in P$ in system M meets trust check and its trace $Tr(p)$ passes the integrity measurement, then state transfer is trusted. We prove the reachable states trusted by induction of the length *l* of trace $Tr(p)$.

1) when the length *l*=0, the system state is initial state s_0, and it is trusted;
2) suppose the length *l*=*n*, the system state *s* is trusted, then after the action *a* of process $p, s' \Rightarrow (p, \alpha \circ a, s_0)$ is trusted derivated by Definition 3, 8, 9.

3. Semantic Realization and Validation of Trust Chain Model SPA-NI

In paper [13], they used SPA to analyze the security attributes of static trust chain. We use their methods for reference to verify the security attributes of dynamic trust chain model above. In this section, we use the model semantic to describe Intel TXT. The process flow of Intel TXT is as follow: CPU sends GETSEC [SENTER] instruction to TPM, and TPM resets its last eight PCRs; CPU measures Authenticated Code Module (AC Module) and start AC Module; AC Module reads Launch Control Policy (LPC) Strategy by LPC Engine and judge whether Measured Launched Environment (MLE) meets the integrity measurement and then load MLE; MLE measures the applications and runs them in its own independent isolated domain. All the measurements should be extended into the measurement log.

We define DRTM system as a tree process entities (TXT, TPM and MLE) composition. TXT is defined as TCPU (trusted CPU with SENTER privileged instruction) and ACM composition. The operations of DRTM system include ret PCR (ret_PCR), measure code (m_),

extend PCR (m_ExtendTPM), update measurement log (updateLog) and execute code (e_). The TXT semantic is as follow:

$$DRTM \triangleq (TXT \,||\, TPM \,||\, MLE) \setminus Bind$$

$$TXT \triangleq TCPU \,||\, ACM$$

$$TCPU \triangleq e\_SENTER.\ \overline{ret\_PCR}.\ m\_ACM.\ \overline{m\_ExtendTPM}.\ \overline{e\_ACM}.TCPU$$

$$ACM \triangleq e\_ACM.\ m\_MLE.\ \overline{load\_TPMLPC}.\ \overline{e\_MLE}.\ ACM$$

$$TPM \triangleq ret\_PCR.\ w\_PCR.\ \overline{updateLog}.\ TPM + m\_ExtendTPM.\ w\_PCR.\ \overline{updateLog}\ .TPM + load\_TPMLPC.\ \overline{updateLog}\ .TPM$$

$$MLE \triangleq e\_MLE.\ \overline{m\_Application}.\ \overline{m\_ExtendTPM}.\ e\_Application.\ \underline{0}$$

We use CoPS [17] to verify the security properties of model, namely the formula (5). The formula (5) exhibits Bisimulation-based Non-deducibility on Composition (BDNC) property, however, BNDC is difficult to verify, so we validate Persistent BNDC (P_BNDC), Processing Persistent BNDC (PP_BNDC) and Strong BNDC (SBNDC), which are included by BDNC. In order to validate the semantic above, we should give the restrict set Bind and high security level set Act_H :

$$Bind = \{m\_ExtendTPM,\ load\_TPMLPC,\ ret\_PCR,\ a\_ACM\_ACPI, w\_PCR,\ m\_Application, e\_ACM,\ e\_MLE\}$$

$$Act_H = \{m\_ExtendTPM,\ \overline{m\_ExtendTPM},\ load\_TPMLPC, \overline{load\_TPMLPC}, ret\_PCR,\ \overline{ret\_PCR},\ w\_PCR,\ \overline{updateLog}\}$$

The restrict set Bind synchronize the actions between TXT and TPM and high security level actions set Act_H contains all the actions (measure, extend, append log) of TPM. The validation results by CoPS are showed in table 1. As we can see from table 1, the semantic model all meets the three security properties, so it also meets BDNC property. The model shows better non-interference attributes and it protects high security actions efficiently.

Table 1. The validation results of semantic model

Security Property	Whether meet
P_BNDC	Yes
PP_BNDC	Yes
SBNDC	Yes

4. Conclusion

In this article, we formally descript and define dynamic trust chain transfer model based on SPA and non-interference theory. The model abstracts the entities of system into process entities, and uses SPA to describe the state conventions and non-interference to limit the rules in trust chain transfer. We give the trust transitive property and system runtime trust theorem. By modeling Intel TXT, the model shows better security attributes and effectively solve the problem that repeat measurement in runtime lack theory model. The application dynamic trusted measurement realization is future research.

References

[1] Zhou Haiqing, Jiang Weizhong, Jin Hai. The Principle and Application of Trusted Computing Technology. Beijing: Science Press, 2011: 23-30.
[2] TCG. TCG Specification Architecture Overview [EB/OL]. [2010-04-20]. http://www.trustedcomputinggroup.org/groups/tpm.
[3] Intel LaGrande Technology Preliminary Architecture Specification, September, 2006 http://www.intel.com

[4] Jeff Teo. Features and Benefits of Trusted Computing. 67-71.
[5] Refiner Sailer, Xiaolan Zhang, Trent Jaeger. *IBM Research Report, Design and Implementation of a TCG-based Integrity Measurement Architecture.* 13th Usenix Security Symposium, San Diego, California. August 2004.
[6] Tian Li-Ye, Shen Chang-Xiang. Productive information system oriented trust chain scheme. *Telkomnika*. 2012; 10(5): 1093-1100.
[7] Guo Xianchen, Zheng Jun, Zhang Qikun, Liu Hongchang. Role-based trust management model in multi-domain environment. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2013; 11(1): 417-424.
[8] Zhao Jia, Shen Changxiang, Liu Jiqiang, Han Zhen. A Noninterference-Based Trusted Chain Model. *Journal of Computer Research and Development*. 2008; 45(6): 76-79.
[9] Si Limin, Cai Mian, Chen Yinjing, Guo Ying. Research of a Trust Chain Transfer Model. *Computer Science*. 2011; 38(9): 79-81.
[10] Kong Xiangying, 1Zhuang Yi. Research on Trust Chain Transfer Model Based on Dynamic Intransitive Noninterference. *JCIT: Journal of Convergence Information Technology*. 2012; 7(21): 157-163.
[11] Liu Changping, Fan Mingyu, Wang Guangwei. Modeling trust chain with stochastic process algebra. *Application Research of Computers*. 2010; 27(12): 4650-4653.
[12] Fu Ning, Zhou Xingshe, Zhan Tao. QPi: A Calculus to Enforce Trustworthiness Requirements. *Journal of Computer Research and Development*. 2011; 48(11): 2120-2130.
[13] Xu Mingdi, Zhang Huanguo, Zhao Heng, et al. Security Analysis on Trust Chain of Trusted Computing Platform. *Chinese Journal of Computers*. 2010; 33(7): 1165-1176.
[14] Goguen JA, Meseguer J. *Security policies and security models*. Proceedings of the 1982 IEEE Symposium on Security and Privacy. California: IEEE Computer Society Press, 1982: 11-20.
[15] Rushby J. *Noninterference, Transitivity, and Channel-Control Security Policies*. SL-92-02, Menlo Park: Stanford Research Institute, 1992.
[16] Focardi R, Gorrieri R. A classification of security properties for process algebras. *Journal of Computer Security*, 1995; 3(1): 5-33.
[17] Piazza C, Pivato E, *Rossi S. CoPS-checker of persistent security*. Proceedings of the Tools and Algorithms for the Construction and Analysis of Systems. Barcelona, Spain. LNCS 2988. Springer, 2004: 144-152.