International Journal of Control and Automation
Vol.8, No.12 (2015), pp.187-194
http://dx.doi.org/10.14257/ijca.2015.8.12.17

Intransitive Noninterference in the Action-disordered System

Congdong Lv, Wei Ma and Xiaoyong Li

School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China
lvcongdonglv@163.com

Abstract

Actions in some systems are disordered. But IP-secure and TA-secure can't fit the disordered system. This paper proposes completely purge secure to deal with the problem. The function cpurge proposed can purge all actions which directly or indirectly interference the domain which can't be done by the function ipurge and TA.

Keywords: *Information Flow Security, Noninterference, IP-secure, TA-secure, purge*

1. Introduction

The classical theory of noninterference was introduced by Goguen and Meseguer [1] to deal with transitive policy. This theory is unable to deal with certain systems which require channel control and downgrading [2]. Haigh and Yong [3] proposed a variant of the classical definition for channel control applications to overcome this limitation. It was further promulgated by Rushby [4]. And the definition remains in use and continues to be the subject of research [5-9]. In [9], Meyden rewrites the function purge and ipurge and proposes ta, to and ito. These definitions state that the agent should not have more information than the maximal amount it is permitted to have.

Chong and Myers [15] have proposed a flexible language that attaches downgrading conditions to data items. Mantel and Sand [7] have proposed to introduce a programming annotation for downgrading. Bossi et al [16] develop a theory of downgrading grounded in simulation-based notions of unwinding. Sabelfeld and Sands [17] lay out some general principles and direction for research in this area. Here, we also use a simple downgrade model to explain the noninterference, as shown in Figure 1.

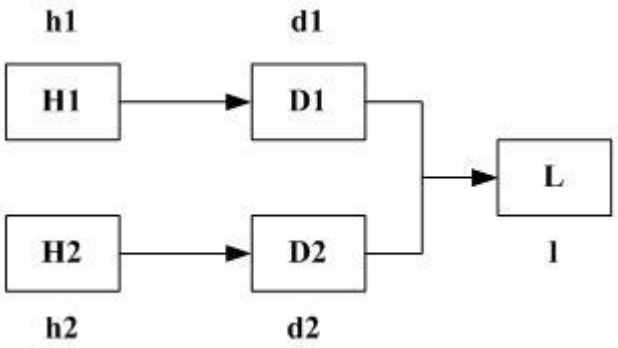

Figure 1. A Simple Downgrade Model

Example 1. Consider the intransitive policy $\rightarrow$ given by $H_1 \rightarrow D_1$, $H_2 \rightarrow D_2$, $D_1 \rightarrow L$ and $D_2 \rightarrow L$. Intuitively, H_1, H_2 are two high security domains. D_1, D_2 are two down

ISSN: 2005-4297 IJCA

graders. And L is an aggregator of downgraded information. Define the system M with actions $A = h_1, h_2, d_1, d_2, l$ of domains H_1, H_2, D_1, D_2, L respectively. Consider the sequences of actions $a_1 = h_1 h_2 d_1 d_2$ and $a_2 = h_2 h_1 d_1 d_2$. Case 1:

$purge_L(a_1) = purge_L(h_1 h_2 d_1 d_2) = d_1 d_2$;

$purge_L(a_2) = purge_L(h_2 h_1 d_1 d_2) = d_1 d_2$;

$purge_L(a_1) = purge_L(a_2)$, but $obs(s_0, a_1) \neq obs(s_0, a_2)$.

Case 2:

$source(e, L) = \{L\}$;

$source(d_2, L) = \{D_2, L\}$;

$source(d_1 d_2, L) = \{D_1, D_2, L\}$;

Case 2.1:

$source(h_2 d_1 d_2, L) = \{H_2, D_1, D_2, L\}$;

$source(h_1 h_2 d_1 d_2, L) = \{H_1, H_2, D_1, D_2, L\}$;

$ipurge_L(a_1) = ipurge_L(h_1 h_2 d_1 d_2) = h_1 h_2 d_1 d_2$;

Case 2.2:

$source(h_1 d_1 d_2, L) = \{H_1, D_1, D_2, L\}$;

$source(h_2 h_1 d_1 d_2, L) = \{H_2, H_1, D_1, D_2, L\}$

$ipurge_L(a_2) = ipurge_L(h_2 h_1 d_1 d_2) = h_2 h_1 d_1 d_2$

So, $ipurge_L(a_1) \neq ipurge_L(a_2)$.

Case 3:

$ta_L(a_1) = ((e,(e,e,h_1),d_1),(e,e,h_2),d_2)$;

$ta_L(a_2) = ((e,(e,e,h_1),d_1),(e,e,h_2),d_2)$;

$ta_L(a_1) = ta_L(a_2)$;

$ta_L(a_1) = ta_L(a_2)$ but $obs(s_0, a_1) \neq obs(s_0, a_2)$.

From the example, we can see that purge, ipurge and TA are all can purge some actions. But if the actions are disordered, they don't know how to deal with the actions.

The aim in paper [11] is to provide the most accessible account they can of the methods they use and why their model, in its details, has turned out the way it has. Paper [10] shows how to automate the resulting timed noninterference check within the context of the recent extension of FDR [12] to analyze a discrete version of Timed CSP [13] and how an extended theory of digitization has the potential both to create more accurate specifications and to infer when processes are non-interfering in the more usual continuous-time semantics. A theory was developed for state-based noninterference in a setting where different security policies—we call them local policies—apply in different parts of a given system [25].

In this paper, we mainly deal with the non-interference in concurrent system. Actions in the system is completely disordered, the exits definition purge, ipurge, ta, to, ito can not deal with the problems. To overcome these limitations, we propose the definition cpurge. The definition can purge all the non-interference actions ignoring the order. Our contributions can be concluded as follow:

Propose function cpurge and CP-secure;

Analysis the relation between CP-secure, IP-secure and TA-secure;

Prove that when the policy is transitive, if CP-secure, then IP-secure and TA-secure;

Prove that there exists the shortest action set in the function cpurge;

Prove that n-th cpurge are equivalent to 1-th cpurge.

2. Background

We work here with the state-observed machine model used by Rushby, but similar results would be obtained for other models. This model consists of deterministic machines of the form $< S, s_0, A, step, obs, dom >$, where S is a set of states, $s_0 \in S$ is the initial state, A is a set of actions, $dom: A \to D$ associates each action to an element of the set D of security domains, $step: S \times A \to O$ is a deterministic transition function, and $obs: S \times D \to O$ maps states to an observation in some set O, for each security domain.

$source(e,m) = \{m\}$;

$$source(aa,m) = \begin{cases} source(a,m) \cup \{dom(a)\}; \exists n \in source(a,m), dom(a) \to n \\ source(a,m); otherwise \end{cases};$$

$ipurge_m(e) = e$;

$$ipurge_m(aa) = \begin{cases} a \times ipurge_m(a); dom(a) \in source(aa,m) \\ ipurge_m(a); otherwise \end{cases};$$

Definition 1. M is IP-secure with respect to a policy $\to$ if for all $m \in D$ and all sequences $a, a' \in A^*$ with $ipurge_m(a) = ipurge_m(a')$, we have $obs(s_0, a) = obs(s_0, a')$.

$ta_m(e) = e$;

$$ta_m(aa) = \begin{cases} (ta_m(a), ta_{dom(a)}(a), a); dom(a) \to m \\ ta_m(a); otherwise \end{cases};$$

Definition 2. M is TA-secure with respect to a policy $\to$ if for all $m \in D$ and all sequences $a, a' \in A^*$ with $ta_m(a) = ta_m(a')$, we have $obs(s_0, a) = obs(s_0, a')$.

Definition 3. n-th purge:

$purge_m^n(a) = purge_m(purge_m^{n-1}(a))$;

$purge_m^1(a) = purge_m(a)$.

Definition 4. Function to get the length of the action sequence:

$len: A^* \to Z_n$;

$len(A^*) = m; m \in Z_n$.

Definition 5. The action sequence a and the action sequence a' are equivalent on domain $m \in D$ for function $purge$ if $ipurge_m(a) = ipurge_m(a')$.

Definition 6. $[a]_X$ is a equivalence class if it satisfied the following condition:

$\forall a \in A^*, [a]_X = \{b \mid b \in A^* \wedge X(a) = X(b)\}$, where X can be $purge$, $ipurge$, ta, to and ito.

3. Our Proposed

Definition 7. Functions $cpurge$ and $csource$ are defined as follow:

Suppose $a = a_1 \times \cdots \times a_n$, $A = \{a_1, \cdots, a_n\}$, $A_1 = A$, $A_2 = \{\}$;

$D = \{dom(a_1), \cdots, dom(a_n), m\}$, $D_2 = D_1 = \{m\}$;

When the parameter is an action sequence, the process is as follow:

$flag = 0$;

process-1:

$while(a \hat{I} a)$

$while(n \hat{I} D_1)$

$if(dom(a) \circledR n)$

$\{ flag = 1; D_2 = D_2 \grave{E} dom(a); A_2 = A_2 \grave{E} \{a\}; \}$

$if(flag = 1)$

$\{ D_1 = D_2; A_1 = A - A_2; flag = 0; goto$ process-1;$\}$

$cpurge_m(a) = A_2; csource_m(a) = D_1.$

The parameter can also be an action set A, that is $cpurge_m(A)$. The alter definition is as follow:

$flag = 0;$

process-2:

$while(a \hat{I} A_1)$

$while(n \hat{I} D_1)$

$if(dom(a) \circledR n)$

$\{ flag = 1; D_2 = D_2 \grave{E} dom(a); A_2 = A_2 \grave{E} \{a\}; \}$

$if(flag = 1)$

$\{ D_1 = D_2; A_1 = A - A_2; flag = 0; goto$ process-1;$\}$

$cpurge_m(A) = A_2; csource_m(A) = D_1.$

Definition 8. M is CP-secure with respect to a policy $\circledR$ if for all $m \hat{I} D$ and all sequences $a, a' \hat{I} A^*$ with $cpurge_m(a) = cpurge_m(a')$, we have $obs_m(s_0, a) = obs_m(s_0, a')$.

Theorem 1. For function $cpurge$ and any action sequence $a = a_1 \times\times\times a_n$, if $cpurge_m(a) = \{a_1, \times\times\times, a_n\}$, then a is the unique smallest action set for $[a]_{cpurge}$.

Proof. Suppose $a = a_1 \times\times\times a_n$.

Assume exist b', $cpurge_m(b') = cpurge_m(a)$ and $len(b') < len(a)$, then

$\$a_i \hat{I} \{a_1, \times\times\times, a_n\}, dom(a_i) \ddot{I} csource(a_{i+1} \times\times\times a_n)$.

But $cpurge_m(a) = a_1 \times\times\times a_n$, $\forall a_i \hat{I} \{a_1, \times\times\times, a_n\}$, $dom(a_i) \hat{I} csource(a_{i+1} \times\times\times a_n)$.

This is inconsistent with launch of the assumptions.

So not exist b', $cpurge_m(b') = cpurge_m(a) = \{a_1, \times\times\times, a_n\}$ and $len(b') < len(a)$.

That is if $a = a_1 \times\times\times a_n$ and $cpurge_m(a) = \{a_1, \times\times\times, a_n\}$, then a is the unique smallest action set for $[a]_{cpurge}$.

Theorem 2. n-th $cpurge$ is equivalent to 1-th $cpurge$, that is:

$cpurge_m^n(a) = cpurge_m^1(a) = cpurge_m(a)$.

Proof. Suppose $cpurge_m(a) = a_1 \times\times\times a_n$.

$cpurge_m^1(a) = cpurge_m(a) = \{a_1, \times\times\times, a_n\};$

$cpurge_m^2(a) = cpurge_m^1(cpurge_m(a)) = cpurge_m(\{a_1, \times\times\times, a_n\}) = \{a_1, \times\times\times, a_n\};$

Assume $cpurge_m^n(a) = \{a_1, \times\times\times, a_n\}$,

$cpurge_m^{n+1}(a) = cpurge_m^1(cpurge_m^n(a)) = cpurge_m^1(\{a_1, \times\times\times, a_n\}) = \{a_1, \times\times\times, a_n\};$

So $cpurge_m^n(a) = cpurge_m^1(a) = cpurge_m(a)$.

Theorem 3. If ® is transitive, then
M is P-secure with respect to ® if M is CP-secure;
M is IP-secure with respect to ® if M is CP-secure;

M is TA-secure with respect to ® if M is CP-secure.

Proof. Suppose M is P-secure and we have $purge(a_1) = purge(a_2)$. The smallest action sequence of a_1 and a_2 is $a = \{a_1, \times\times, a_n\}$.

Then $dom(a_1) \mathbin{®} m, \ldots, dom(a_n) \mathbin{®} m$.

Because $dom(a_i) \mathbin{®} m, i £ n$

So a_i Î $cpurge_m(a_1)$ and a_i Î $cpurge_m(a_2)$.

Assume exist b, b Î $cpurge_m(a_1)$ but it is not an action in a.

From the definition of $cpurge$, $dom(b) \mathbin{®} dom(a_i) \mathbin{®} \times\times \mathbin{®} m$,

Because ® is transitive, so $dom(b) \mathbin{®} m$.

So $dom(b)$ Î $\{dom(a_1), \ldots, dom(a_n)\}$,

So not exist b, b Î $cpurge_m(a_1)$ but it is not an action in a.

$cpurge_m(a_1) = \{a_1, \times\times, a_n\}$, and $cpurge_m(a_2) = \{a_1, \times\times, a_n\}$.

So if $purge(a_1) = purge(a_2)$, then $cpurge(a_1) = cpurge(a_2)$, then $obs_m(s_0, a_1) = obs_m(s_0, a_2)$.

So we have M is P-secure with respect to ® if M is CP-secure.

Suppose M is IP-secure and we have $ipurge(a_1) = ipurge(a_2)$. The smallest action sequence of a_1 and a_2 is $a = \{a_1, \times\times, a_n\}$.

Then $dom(a_n) \mathbin{®} m$;

$dom(a_{n-1}) \mathbin{®} dom(a_n)$ Ú $dom(a_{n-1}) \mathbin{®} m$;

…

$dom(a_1) \mathbin{®} dom(a_2)$ Ú … Ú $dom(a_1) \mathbin{®} m$;

Because ® is transitive, so $dom(a_1) \mathbin{®} m, \ldots, dom(a_{n-1}) \mathbin{®} m, dom(a_n) \mathbin{®} m$.

Because $dom(a_i) \mathbin{®} m, i £ n$, so a_i Î $cpurge_m(a_1)$ and a_i Î $cpurge_m(a_2)$.

Assume exist b, b Î $cpurge_m(a_1)$ but it is not an action in a.

From the definition of $cpurge$, $dom(b) \mathbin{®} dom(a_i) \mathbin{®} \times\times \mathbin{®} m$,

Because ® is transitive, so $dom(b) \mathbin{®} m$.

So $dom(b)$ Î $\{dom(a_1), \ldots, dom(a_n)\}$,

So not exist b, b Î $cpurge_m(a_1)$ but it is not an action in a.

$cpurge_m(a_1) = \{a_1, \times\times, a_n\}$, and $cpurge_m(a_2) = \{a_1, \times\times, a_n\}$.

So if $ipurge(a_1) = ipurge(a_2)$, then $cpurge(a_1) = cpurge(a_2)$, then $obs_m(s_0, a_1) = obs_m(s_0, a_2)$.

So we have M is IP-secure with respect to ® if M is CP-secure.

Suppose M is TA-secure and we have $ta(a_1) = ta(a_2)$. The smallest action sequence of a_1 and a_2 is $a = \{a_1, \times\times, a_n\}$.

Then $dom(a_n) \mathbin{®} m$;

$dom(a_{n-1}) \mathbin{®} dom(a_n)$ Ú $dom(a_{n-1}) \mathbin{®} m$;

...

$dom(a_1) \mathrel{®} dom(a_2) \cup \ldots \cup dom(a_1) \mathrel{®} m$;

Because $®$ is transitive, so $dom(a_1) \mathrel{®} m, \ldots, dom(a_{n-1}) \mathrel{®} m, dom(a_n) \mathrel{®} m$.

Because $dom(a_i) \mathrel{®} m, i \le n$, so $a_i \in cpurge_m(a_1)$ and $a_i \in cpurge_m(a_2)$.

Assume exist b, $b \in cpurge_m(a_1)$ but it is not an action in a.

From the definition of $cpurge$, $dom(b) \mathrel{®} dom(a_i) \mathrel{®} \ldots \mathrel{®} m$,

Because $®$ is transitive, so $dom(b) \mathrel{®} m$.

So $dom(b) \in \{dom(a_1), \ldots, dom(a_n)\}$,

So not exist b, $b \in cpurge_m(a_1)$ but it is not an action in a.

$cpurge_m(a_1) = \{a_1, \ldots, a_n\}$, and $cpurge_m(a_2) = \{a_1, \ldots, a_n\}$.

So if $ta(a_1) = ta(a_2)$, then $cpurge(a_1) = cpurge(a_2)$, then $obs_m(s_0, a_1) = obs_m(s_0, a_2)$.

So we have M is TA-secure with respect to $®$ if M is CP-secure.

Proposition 1 is to explain the relation between IP-secure and CP-secure in intransitive.

Proposition 1. (1) If M is IP-secure with respect to policy $®$ cannot have M is CP-secure with respect to $®$.

(2) If M is CP-secure with respect to policy $®$ cannot have M is IP-secure with respect to $®$.

Here we give two examples to explain the proposition.

Example 1. Suppose $dom(a_1) \mathrel{®} m, dom(a_2) \mathrel{®} dom(a_1)$;

Then $ipurge_m(a_1 a_2) = ipurge_m(a_1) = a_1$;

But $cpurge_m(a_1 a_2) = \{a_1, a_2\}, cpurge_m(a_1) = a_1$, $cpurge_m(a_1 a_2) \neq cpurge_m(a_1)$.

Example 2. Suppose $dom(a_1) \mathrel{®} m, dom(a_2) \mathrel{®} dom(a_1), dom(a_3) \mathrel{®} dom(a_1)$;

Then $cpurge_m(a_1 a_3 a_2) = cpurge_m(a_2 a_3 a_1) = \{a_1, a_2, a_3\}$;

But $ipurge_m(a_1 a_3 a_2) = a_1, ipurge_m(a_2 a_3 a_1) = a_2 a_3 a_1$, $ipurge(a_1 a_3 a_2)_m \neq ipurge_m(a_2 a_3 a_1)$.

Proposition 2 is to explain relation between TA-secure and CP-secure in intransitive.

Proposition 2. (1) If M is TA-secure with respect to policy $®$ cannot have M is CP-secure with respect to $®$.

(2) If M is CP-secure with respect to policy $®$ cannot have M is TA-secure with respect to $®$.

Here we also give two examples to explain the proposition.

Example 3. Suppose $dom(a_1) \mathrel{®} m, dom(a_2) \mathrel{®} dom(a_1)$;

Then $ta_m(a_1 a_2) = ta_m(a_1) = (e, e, a_1)$;

But $cpurge_m(a_1 a_2) = \{a_1, a_2\}, cpurge_m(a_1) = a_1$, $cpurge_m(a_1 a_2) \neq cpurge_m(a_1)$.

Example 4. Suppose $dom(a_1) \mathrel{®} m, dom(a_2) \mathrel{®} dom(a_1), dom(a_3) \mathrel{®} dom(a_1)$;

Then $cpurge_m(a_1 a_3 a_2) = cpurge_m(a_2 a_3 a_1) = \{a_1, a_2, a_3\}$;

But $ta_m(a_1 a_3 a_2) = (e, e, a_1), ta_m(a_2 a_3 a_1) = (e, ((e, e, a_2), e, a_3), a_1)$,

$ta(a_1 a_3 a_2)_m \neq ta_m(a_2 a_3 a_1)$.

4. Related Work

The notion of noninterference was first proposed by Goguen and Meseguer [1]. At first, part of works was motivated by multi-level secure system. Some others dealt with

deterministic systems. Then, nondeterministic systems [18-22] were focused. Intransitive policies [4, 8, 23] were also another part. Paper [9] is work on intransitive policies in the deterministic case. They have done a great work. They proposed TA-secure, TO-secure and ITO-secure. We focus in this paper on intransitive policies in the concurrent system in which the actions are disordered. The complexity of intransitive noninterference was well discussed in [24] and [25]. Now we will give a table to compare the definition of information flow security for transitive and intransitive policies.

As shown in Table 1, a is an n-length action sequence. The column of Time for a is the time cost for a function to deal with the sequence. The column of Space for a is the space cost for a function to deal with the sequence. The column of Time for a finite system is the time cost for a function to check the finite system is the function secure or not.

Table 1. Comparison of Functions

	transitive/ intransitive	action order	time for an action sequence	space for an action sequence
$purge$ [1]	transitive	ordered	O(n)	S(1)
$ipurge$ [3]	intransitive	ordered	O(n)-O(n2)	S(1)-S(n2)
ta [9]	intransitive	partly ordered	O(n)-O(2n)	S(1)-S(2n)
to [9]	intransitive	ordered	O(n)-O(n3)	S(1)-S(n2)
$cpurge$ (Ours)	intransitive	completely disordered	O(n)-O(n2)	S(1)

5. Conclusions

Actions in P-secure and IP-secure are ordered. Actions in TA-secure are partly disordered, but not completely. They cannot deal with the completely disordered actions. CP-secure proposed in this paper can purge all actions which interference with the domain even when actions are disordered. In the future, more work should do on the disordered actions. The properties of the function $cpurge$ are also discussed in this paper. We also research on the relation between $ipurge$ and $cpurge$. So is ta and $cpurge$. We also compare our work with the other related works.

Acknowledgments

Project supported by the Higher Specialized Research Fund for the Doctoral Program ordered Ministry of Education of China (No. 20120009110007) and 2012 MOR Scientific Research and Development Program ordered by Ministry of Railways of China (Now is China Railway Company) (No. 2012X010-B).

References

[1] J. A. Goguen and J. Mesegure, "Security policies and security models. In Proc.1982 Symposium on Security and Privacy", Oakland, CA., **(1982),** pp. 11-20.
[2] J. M. Rushby, "Design and verification of secure systems.ACM SIGOPS Operating Systems Review", vol. 15, no. 5, **(1981),** pp. 12-21.
[3] J. T. Haigh and W.D. Young, "Extending the Noninterference Version of MLS for SAT", IEEE Trans. Software Eng., vol. 13, no. 2, **(1987),** pp.141-150.

[4] J. Rushby, "Noninterference, transitivity, and channel-control security policies", SRI International, Computer Science Laboratory, **(1992)**.
[5] V. D. R. Meyden, "Architectural refinement and notions of intransitive noninterference", Formal Aspects of Computing, vol. 24, no. 4-6, **(2012)**, pp. 769-792.
[6] N. B. Hadj-Alouane, S. Lafrance and F. Lin, "On the verification of intransitive noninterference in mulitlevel security[J]", Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, vol. 35, no. 5, **(2005),** pp. 948-958.
[7] H. Mantel and D. Sands, "Controlled declassification based on intransitive on interference",Programming Languages and Systems. Springer Berlin Heidelberg, **(2004)**, pp. 129-145.
[8] V. D. Oheimb, "Information flow control revisited: Noninfluence= noninterference+ nonleakage", Computer Security–ESORICS 2004, Springer Berlin Heidelberg,**(2004), pp.** 225-243.
[9] V. D. R. Meyden, "What, indeed, is intransitive non interference?",Computer Security-ESORICS 2007.,Springer Berlin Heidelberg, **(2007), pp.** 235-250.
[10] A. W. Roscoe and J. Huang, "Checking noninterference in Timed CSP. Formal Aspects of Computing", vol. 25, no. 1, **(2013),** pp. 3-35.
[11] C. Morgan, "Compositional noninterference from first principles. Formal Aspects of Computing", vol. 24, no. 1, **(2012),** pp. 3-26.
[12] A.W. Roscoe, "Model-checking CSP, A classical mind: essays in honour of CAR Hoare", **(1994),** pp. 353-378.
[13] J. Davies and S. Schneider, "A brief history of Timed CSP", Theoretical Computer Science, vol. 138, no. 2, **(1995),** pp. 243-271.
[14] S. Eggert, H. Schnoor and T. Wilke, "Noninterference with local policies.Mathematical Foundations of Computer Science 2013", Springer Berlin Heidelberg, **(2013),** pp. 337-348.
[15] S. Chong and A. C. Myers, "Security policies for downgrading", Proceedings of the 11th ACM conference on Computer and communications security, **(2004),** pp.198-209.
[16] A. Bossi, C. Piazza and S. Rossi, "Modelling downgrading in information flow security", Computer Security Foundations Workshop, 2004. Proceedings. 17th IEEE. IEEE, **(2004),** pp. 187-201.
[17] A. Sabelfeld and D. Sands, "Dimensions and principles of declassification.Computer Security Foundations", 2005, CSFW-18 2005. 18th IEEE Workshop. IEEE, **(2005),** pp. 255-269.
[18] D. Sutherland, "A model of information", Proc. 9th National Computer Security Conference, **(1986),** pp.175-183.
[19] J. T. Wittbold and D. M. Johnson, "Information Flow in Nondeterministic Systems.IEEE Symposium on Security and Privacy", **(1990),** pp. 144-161.
[20] D. McCullough, "Noninterference and the compensability of security properties.Security and Privacy", 1988, Proceedings., 1988 IEEE Symposium on. IEEE, **(1988), pp.** 177-186.
[21] R. Focardi and R. Gorrieri, "Classification of security properties.Foundations of Security Analysis and Design", Springer Berlin Heidelberg, **(2001),** pp. 331-396.
[22] P. Y. A. Ryan, "Mathematical models of computer security.Foundations of Security Analysis and Design", Springer Berlin Heidelberg, **(2001),** pp. 1-62.
[23] A. W. Roscoe and M. H. Goldsmith, "What is intransitive noninterference?", In IEEE Computer Security Foundations Workshop, **(1999),** pp. 228-238.
[24] S. Eggert, V. D. R. Meyden and H. Schnoor, "The complexity of intransitive noninterference[C", Security and Privacy (SP), 2011 IEEE Symposium on. IEEE, **(2011), pp.** 196-211.
[25] S. Eggert, V. D. R. Meyden and H. Schnoor, "Complexity and Unwinding for Intransitive Noninterference", arXiv preprint arXiv, **(2013),** pp. 1308-1204.

Author

Congdong Lv, he was born in 1987. He got his Bachelor Degree from Nanjing Normal University. Now, he is a PhD candidate in Beijing Jiaotong University. His research interests include information security, cloud security and formal method.

