An Architectural Refinement Approach Based on Trusted Channel in MLS Environment

Wei Ma[1*], Xiaoyong Li[1], Congdong Lv[1] and Fei Li[2]

[1]*School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, P.R. China*

[2]*College of Computer Science and Technology, Zhejiang University, Hangzhou 310058, P.R. China*

Abstract: Architectural refinement is an important approach to save the development costs and speed up the design and development progress. With the traditional research usually focusing on the refinement of functions and components, the additional information flow which is unsupervised generated during the refinement process and the possible loss of consistency of security structure are not considered thoroughly. This paper proposes an architectural refinement approach based on trusted channel working in MLS (Multi-Level Security) environment. Applying characteristics of trusted channel to the refinement of the functions and components in security structure, this paper takes the security issues of additional information flow and consistency of security structure problems arising in the process of refinement into account, and uses TCB (trusted computing base) extension to illustrate architectural refinement to obtain a hierarchical TCB. This paper also makes formal description of this approach and rules that must be followed in the process of applying it. And in the end, the security of this approach is proved using the noninterference model.

Keywords: Security architecture, refinement, trusted channel, noninterference model.

1. INTRODUCTION

The notion of architectural refinement is proposed to resolve the substantial gap between abstract conception and specific implementation during the design of security architecture. Because the traditional multiple security architectures such as SLS (Single-Level Security), MLS (Multi-Level Security), MSLS (Multiple Single-Level Security) and MILS(Multiple Independent Levels of Security) [1] and security models such as BLP (Bell-LaPadula), Biba and lattice based information flow model [23-25] mainly focused on the general view and abstract conception of security system rather than specific implementation. Architectural refinement present substantial advantages such as detailed guidance, accelerated development and reduction of costs by specifying the detailed structure of some internal component of the security architecture, i.e. dividing one single component into several subcomponents or composing some independent components into one single component.

Researchers have been working on architectural refinement approach for a long time. Most of researches on architectural refinement mainly focused on functions, performance and reliability [3-7]. With security issues, there were researchers who worked on refinement at the level of coding language under the guidance of several secure modeling methods which are not compatible with complex systems at a more abstract level [27, 28]. Meanwhile, researchers have been working on architectural refinement such as refinement patterns and se mantics in security architecture from a macroscopic view [10, 11]. However, there are two key drawbacks in the current architecture refinement methods:

First, additional information flows between subcomponents without being well-controlled could be hidden trouble. Refinement would modify the originally internal information exchange inside one single component into external information flows between refined subcomponents. Such additional information flows are not restricted by the original information flow security policies, which may cause the disclosure of information.

Second, the consistency of security structure cannot be preserved, since the security properties would be hard to verify after the refinement. From the view of trusted computing, the security properties of refined architecture can be considered as the TCB (Trusted Computing Base) of the architecture. The refinement process would bring corresponding modification of the original TCB, resulting in the refined TCB structure might be hard to be verified.

In the field of TCB expansion and hierarchical TCB, trusted channel [14] has been proposed to build the confidentiality and integrity of the communication between TCB (Trusted Computing Base) subsets. Such mechanism could essentially improve the control of information flow between two components and inheritance of TCB structure which could become an insightful strategy to solve the fatal problems in architecture refinement.

In this paper, we apply the trusted channel into architectural refinement works to fix those two stubborn issues. With the first issue, this approach focuses on the cross-subcomponents information flows to ensure that the additional information flows would be verified during the refinement process to ensure the overall security policy would not be violated. And with the second issue, based on the conception of trusted channel, we intend to combine TCB expansion with architectural refinement together. This approach will focus on the consistency of TCB structure during the refinement process, so that the efficiency of verification for the refined TCB structure can be significantly improved.

This paper is consisted of five sections. Section 2 depicts background about the related work while section 3 briefly introduced the architectural refinement approach based on trusted channel, and the formal description is given in section 3, too. In section 4, we discuss the security of this approach with noninterference theory. And a simple discussion about the advantage of this approach is made in section 5. Finally, section 6 concludes this paper and proposes some perspectives.

2. ARCHITECTURAL REFINEMENT BACKGROUND

2.1. Architectural Refinement

Architectural refinement is important for efficiency, effectiveness and validity in the design of security architecture. Researchers [3-7] worked on multiple functional refinement methods in software and complex computer based architecture. Denford [5] pointed out that a good refinement method should be both functional and non-functional, and he proposed a method that focused on the nonfunctional requirements while still addressing the functional requirements throughout refinement. Meanwhile, it is considered that refinement should be specific technology based and domain based [3, 8]. Garlan [9] pointed out that the properties which would be preserved in the refinement process should be specified. Zhou Jie [12] worked on security policy refinement approach in MLS environment, and presented three patterns of refinement such as decomposition, aggregation and feedback. A policy refinement language used to describe refinement rules is given by Zhou Jie [26], too. van der Meyden [13] discussed different semantics in the refinement process of security architecture with noninterference model.

2.2. Trusted Channel

Because the security functions are independent and lack of mutual support with each other in one system, a TCB expand model, including TCB hierarchy and trusted support, was proposed [21, 22]. The notion of trusted channel based on TCB expand model was proposed in [14]. Trusted channel, which connects TCB subsets, refers to the logic pathway which ensures the confidentiality and integrity of information flow between TCB subsets, and it mainly solves the problem of the isolation issue in TCB expansion. Trusted channel is defined as a quaternion $\{PIF_{in}, PIF_{out}, T_{MP}, INF\}$ where PIF_{in} refers to the input port of the channel, and PIF_{out} is the output port of the channel, and T_{MP} refers to the channel monitor, and INF is the information transferred in channel. A series of Boolean Functions are defined to describe the functions such as establishment, connection, data encapsulation and transmission of trusted channel. Some important properties of trusted channel are closure, atomicity, authentication and dynamic, which ensure the security of trusted channel itself.

3. REFINE APPROACH BASED ON TRUSTED CHANNEL

3.1. A Simple Example of Architectural Refinement

The refinement process can be demonstrated in a schematic Fig (**1**). Suppose a MLS system S, $S = < C, \rightarrowtail, INF >$, where:

$C =< C_1, C_2, \dots, C_n >$ refers to the components in the system, and in MLS environment, components are supposed to be specific domains;

$\rightarrowtail$ refers to the information flow policy in the system;

INF refers to the content of information transfered in the

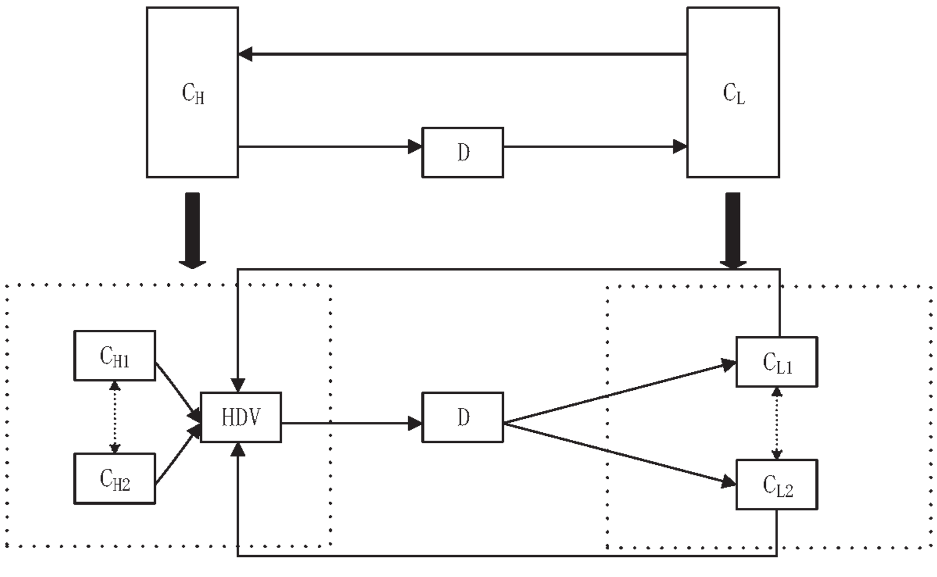

Fig. (1). An example of the refinement of a simple architecture.

system.

A simple example of refinement is shown as below. A system with multiple security levels $S_1 =< (C_L, C_H, D), \rightarrowtail, INF > C_L$ is an independent component with low security label and C_H is an independent component with high security label. The information flow policies allowed in the system is that the information is able to flow directly from component C_L with lower security level to component C_H with higher security label while the information flow from C_H to C_L must not be allowed before it is arbitrated and downgraded by the downgrader D. Obviously, the system design at this level is too abstract for specific implementation. Therefore, refinement based on this prototype system is necessary in practical implement $S_1' =< (C_{H1}, C_{H2}, HDV), (C_{L1}, C_{L2}), D, \rightarrowtail, INF >$, where C_H and C_L are refined into subcomponents C_{H1}, C_{H2} and C_{L1}, C_{L2}. Meanwhile, a downgrader D and a new component HDV (High-level Downgrader Verifier) which will verify the information flow between the downgrader and the higher components are added into the system.

However, this form of refinement brings new problems. Before the refinement of C_H and C_L, there are such information flows which exist as the form of interior information exchanged only inside a component of the architecture, which would not bring any uncertain influence; but in the architecture after the refinement, this type of information flows, which are not restricted by the security policies, should be considered. Additionally, after the refinement, security characteristics of a component itself may also get changed, which might lead to the loss of consistency of security structure and the difficulty of verification of security properties. Therefore, information flows within subcomponents that generated after a component is refined and new characteristic of subcomponents should be taken into account. More specifically, there are two cases can be derived from the refinement process discussed above. First, if the refined subcomponents such as C_{H1} and C_{H2} are with same security label, there is no security sensitive information flow between them. Such well controlled case has beyond the scope of our discussion. On the other hand, if there is security sensitive information exchanged between refined subcomponents such as C_{H1} and C_{H2}, the newly generated inter-subcomponent information flow would be under surveillance. And our approach mainly focuses on this case.

3.2. Description of the Architectural Refinement Approach

Trusted channel is introduced to resolve the issues. In the process of refinement, through the establishment of trusted channels between subcomponents, we can effectively guarantee the confidentiality and integrity of information transmitted between subcomponents; at the same time, with the characteristics of trusted channel such as such as closure, atomicity, authentication and dynamics, the consistency of TCB structure would be reserved in the process of refinement.

We developed appropriate adjustment to our previous formal description to specifically solve the insecure issue after refinement. As shown in Fig (**2**), a system refined with trusted channel can be described as:

$S_r =< (C_{H1}, C_{H2}, HDV), (C_{L1}, C_{L2}), D, \rightarrowtail, INF, T >$, where T is defined as $T =< TC, P_{in}, P_{out} >$, in which TC refers to the trust channel between two refined components. P_{in} and P_{out} indicate the communication ports of the trusted channel.

In Fig (**2**), in the left part of the figure, we consider that there is security sensitive information exchanged between C_{H1} and C_{H2} so there is a trusted channel between them. Either subcomponent has a pair of ports P_{in} and P_{out} as the I/O ports of the trusted channel. Such trust channel structure was only build between high level components in our demonstration example, since we consider low level has less security sensitivity.

In [14], a series of functions which describe the process such as negotiation and establishment of trusted channels are given, and it is unnecessary to go into details here; but in order to meet the characteristics of architecture refine-

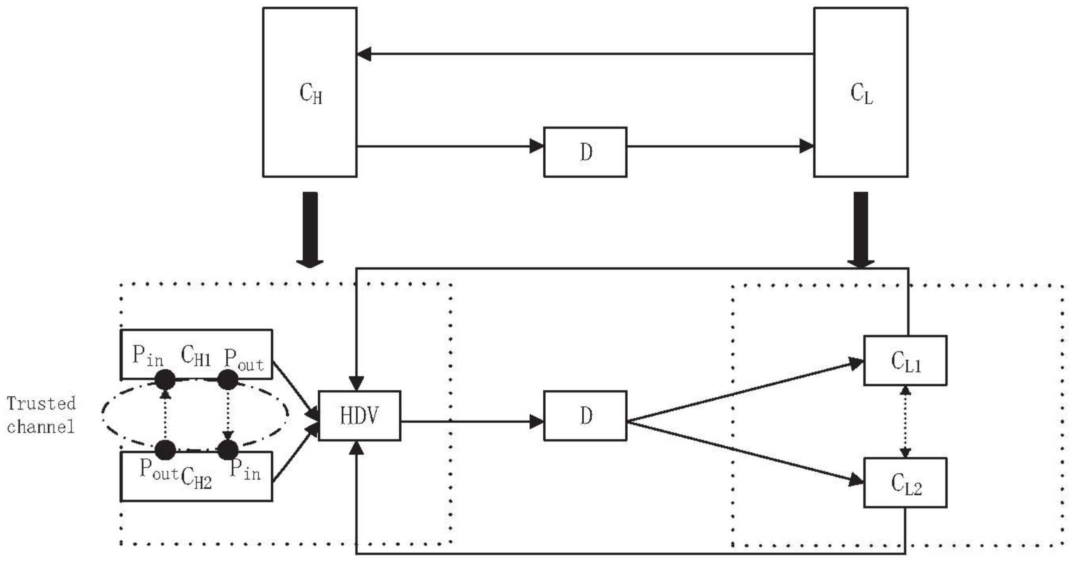

Fig. (2). Refined component using trusted channel.

ment process, we need to add some other definitions of functions:

Definition 1. Connection Check Function

f_{che}: $SubC \times SubC \rightarrow \{true,\ false\}$,

this function is used to check the connection between two subcomponents.

Definition 2. Security sensitive Check Function

f_{ssc}: $INF \rightarrow \{true,\ false\}$,

this function is used to check the information exchanged between two subcomponents security sensitive or not.

Definition 3. Channel Correlation Function

f_{cf}: $SubC \times SubC \rightarrow INF$,

this function is used to define the information exchanged between two subcomponents.

Definition 4. Component Content Function

f_{cf}: $SubC \times SubC \rightarrow INF$,

this function is used to define the information exchanged between two subcomponents.

Definition 5. Trusted Channel Content Function

f_{tf}: $T \rightarrow INF$,

this function is used to define the information flow within a trusted channel.

Definition 6. Security Label Obtain Function

f_{label}: $\{C,\ T\} \rightarrow L$,

this function is used to obtain the security label of an entity such as component, subcomponent and channel. L refers to a collection of security labels.

Definition 7. Port Affirm Function

f_{aff}: $P_{in} \times P_{out} \rightarrow T$,

this function is used to confirm the input port and output port of a trusted channel.

Definition 8. Security sensitive Check Function

f_{PG}: $SubC \rightarrow SubC \times P$,

this function is used to generate a port in a subcomponent.

Definition 9. Component Decompose Function

f_{decmp}: $C \rightarrow SubC \times SubC$,

this function is used to refine one single component into two independent subcomponents. The refinement of component can be regarded as the recursive call of this function.

Definition 10. Port-Component Relation Function

f_{PCR}: $P \times C \rightarrow \{true,\ false\}$,

this function is used to check the relationship between a component and a port (input or output).

On the basis of these definitions, we can formally describe the architectural refinement approach based on trusted channel as follows:

$$\begin{aligned}
&\forall C \xrightarrow{refine} (C) = (SubC_1, SubC_2) \\
&\Rightarrow f_{PG}\left(SubC_1\right) = \left(SubC_1, P_{in1}\right) \wedge f_{PG}\left(SubC_2\right) \\
&= \left(SubC_2, P_{out2}\right) \\
&\Rightarrow f_{aff}\left(P_{in1}, P_{out2}\right) = T_1 \\
&\Rightarrow f_{PCR}\left(P_{in1}, SubC_1\right) \\
&= true \wedge f_{PCR}\left(P_{out2}, SubC_2\right) \\
&= true \wedge f_{cor}\left(SubC_1, SubC_2, T_1\right) = true
\end{aligned}$$

Moreover, according to the specificity of architectural refinement, the following three rules must be obeyed during the process of refining components and establishing trusted channels.

Rule 1: Rule of necessity.

$$\begin{aligned}
&\forall i, j, \exists INF, f_{ssc}\left(INF\right) \\
&= true \wedge f_{che}\left(SubC_i, SubC_j\right) = INF \\
&\Rightarrow \exists\ k, f_{est}\left(SubC_i, SubC_j, T_k\right) = true;
\end{aligned}$$

Rule 1 shows that any security sensitive connection between two subcomponents must be connected via a trusted channel. This rule defines the necessity of trusted channel which ensures that no unverified security sensitive information flow between subcomponents during the process of refinement;

Rule 2: Rule of uniqueness.

$$\begin{aligned}
&\exists k, f_{est}\left(SubC_i, SubC_j, T_k\right) = true \\
&\Rightarrow \exists\ !k, f_{est}\left(SubC_i, SubC_j, T_k\right) = true;
\end{aligned}$$

Rule 2 shows that if there is trusted channel between two subcomponents, this trusted channel would be unique. Based on the atomicity of trusted channel, this rule defines the uniqueness of trusted channel in the refined architecture, and ensures that there are no redundant trusted channels between subcomponents, which make the information flows between subcomponents more likely to be arbitrated and controlled.

Rule 3: Rule of uniformity.

$$\begin{aligned}
&\exists\ k, f_{est}\left(SubC_i, SubC_j, T_k\right) = true \\
&\Rightarrow \forall\ INF \in f_{che}\ \left(SubC_i, SubC_j\right),\ INF \in f_{tf}\left(\ T_k\right)
\end{aligned}$$

Rule 3 shows that if a trusted channel exists between two subcomponents, all of the information should flow via this trusted channel no matter security sensitive or not. With rule 3, it is able to evade the risk of the existence of cover channel.

The three rules, along with the formal description, compose the complete architectural refinement approach. The formal description defines the process of this approach and the rules define the restrictions when the approach is adopted.

4. SECURITY ANALYSIS

In this section we need to discuss the security of refinement approach based on trusted channel. At the beginning of this section, we will introduce noninterference model briefly. After that, the security issue will be discussed with two aspects: one is the trusted channel security and the other is information flow security.

4.1. Noninterference Model

Noninterference model, which was introduced in 1982 for the first time [16, 17], after being modified and improved by series of scholars [18, 19], was completed by van der Meyden [20] with intransitive noninterference TA-security. In this paper, we primarily use the symbols and definitions of TA-security which is introduced as following.

A system M is defined as a deterministic machine $M = < S,\ S_0,\ A,\ \text{step},\ \text{obs},\ \text{dom} >$, where:

S is a set of system states and $S_0 \in S$ is the initial state;

A is a set of system actions;

D is a set of domains in the system;

O is a set of observations which can be observed by the domains;

$\text{step}: S \times A \rightarrow S$ associates each action to an element of the set D of security domains;

$\text{obs}: S \times D \rightarrow O$ is a deterministic transition function;

$\text{dom}: A \rightarrow D$ maps states to an observation in some set O;

$a \in A$ means that a is an action, and $\alpha \in A^*$ means α is a sequence of actions;

$\rightarrowtail \subseteq D \times D$ refers to the interference policy between domains, and it can be understood as information flows between two domains;

$\not\rightarrowtail$ refers that there is no interference between domains.

As the ϵ stands for an empty action sequence, the function ta_u is defined as following:

1. if $\text{dom}(a) \not\rightarrowtail u$, then $ta_u(\alpha a) = ta_u(\alpha)$;
2. if $\text{dom}(a) \rightarrowtail u$, then $ta_u(\alpha a) = (ta_u(\alpha), ta_{\text{dom}(a)}(\alpha), a)$.

Definition 11.

A system M is TA-secure with respect to a policy $\rightarrowtail$ if for all domains u and all $\alpha, \alpha' \in A^*$ such that $ta_u(\alpha) = ta_u(\alpha')$, we have $obs_u(S_0 \cdot \alpha) = obs_u(S_0 \cdot \alpha')$.

Based on the above definition 11, a theorem called *weak unwinding* theorem [20] is proposed by van der Meyden as follows:

Theorem 1.

A weak unwinding for a system M with respect to a policy $\rightarrowtail$ is an indexed collection of equivalence relation$\sim_u$s, $u \in D$ on the states of M satisfying the following conditions:

1. *OC(Output Consistenty):*

   If $s \sim_u t$ then $obs_u(s) = obs_u(t)$
2. *WSC(Weak step Consistency):*

   If $s \sim_u t$ and $s \sim_{dom(a)} t$, then $s \cdot a \sim_{dom(a)} t \cdot a$.
3. *LR(Left Respect):*

   If $dom(a) \not\rightarrowtail u$ then $s \sim_{dom(a)} s \cdot a$.

Suppose that there exists a weak unwinding for M with respect to $\rightarrowtail$ Then M is TA-secure with respect to $\rightarrowtail$.

4.2. Trusted Channel Security

With a given MLS system, we need to make two hypothesizes as follows:

Hypothesis 1.

Every component (domain) in the MLS is with root of trust.

Hypothesis 2.

A complete chain of trust exists in every component (domain).

According to the formal description of refined system S_r', P_{in} undertakes the responsibility of decrypting and P_{out} undertakes the responsibility of encrypting, so it is obviously that the closure of trusted channel is not violated; with hypothesis 2, there is an intact chain of trust in a component, so the trusted channel including trusted channel monitor and ports of the trusted channel can be considered trusted, which is intuitively that the trusted channel is able to be authenticated; rule 2 indicates the atomicity of trusted channel; and because the channel connection is established during the process of refinement, so we can take for that the dynamic of trusted channel is also ensured. Meanwhile, with hypothesis 1 and the definition of trusted expand, the refinement process with trusted channel is a process of expansion from the TCB initial core, which refers to the root of trust within a component, to the higher level of TCB subset. So the security of trusted channel can be assured in this approach.

4.3. Information Flow Security

Because we assume that the security of the original security architecture can be trusted, we only need to prove the security of the architecture after refinement, i.e., information flow security between the sub components.

Lemma 1.

Given two components C_1 and C_2, there are interference relationship, i.e. $C_1 \rightarrowtail C_2$ or $C_2 \rightarrowtail C_1$, iff $\exists\, i, f_{est}(C_1, C_2, T_i) = \text{true}$.

Proof:

There are two conditions should be considered.

1. If $C_1 \rightarrowtail C_2$ or $C_2 \rightarrowtail C_1$, then intuitively, there are corresponding information flow between C_1 and C_2, which means that C_1 and C_2 are connected by a channel, i.e. $f_{est}(C_1, C_2, T_k) = \text{true}$. And according to rule 1, we have that $\exists\, i, f_{est}(C_1, C_2, T_i) = \text{true}$.
2. If $\exists\, i, f_{est}(C_1, C_2, T_i) = \text{true}$, then we have:

$$f_{est}\left(C_1, C_2, T_i\right) = true$$
$$\Rightarrow \exists\ P_{in}, P_{out}, f_{aff}\left(P_{in}, P_{out}\right) = T_i$$
$$\Rightarrow \left(f_{PCR}\left(C_1, P_{in}\right) = true \wedge f_{PCR}\left(C_2, P_{out}\right) = true\right)$$
$$\vee (f_{PCR}\left(C_1, P_{out}\right)$$
$$= true \wedge f_{PCR}\left(C_2, P_{in}\right) = true)$$
$$\Rightarrow C_1 \rightarrowtail C_2 \vee C_2 \rightarrowtail C_1$$

Lemma 1 is proved.

Theorem 2.

Architecture generated by the refine approach based on trusted channel satisfies TA-security.

Proof:

According to theorem 1, we can prove the architecture is TA-security when it satisfies the three conditions proposed by theorem 1.

Firstly, suppose that the refined architecture is $S = < C, \rightarrowtail >$, where C is the set of subcomponents and $\rightarrowtail$ is the information flow policy in the architecture. Obviously,

$$s \sim_{dom(a)} t \wedge \left(obs_c\left(step\left(s,a\right)\right) \neq obs_c\left(s\right)\right)$$

which means that this architecture satisfies OC.

Secondly, we suppose that $s \sim_c t$ and $s \sim_{dom(a)} t$, and it is clear that if action a changes the output state, the changed result is only related to the system state that can be observed by $dom(a)$, which can be explained as follows:

$$s \sim_{dom(a)} t \wedge \left(obs_c\left(step\left(s,a\right)\right) \neq obs_c\left(s\right)\right)$$
$$\vee \left(obs_c\left(step\left(t,a\right)\right) \neq obs_c\left(t\right)\right)$$
$$\Rightarrow obs_c\left(step\left(s,a\right)\right) = obs_c\left(step\left(t,a\right)\right)$$

And now there are three conditions should be considered:

if $obs_c\left(step(s,a)\right) \neq obs_c\left(s\right)$, we have $obs_c(step(s,a)) = obs_c(step(t,a))$;

if $obs_c\left(step(t,a)\right) \neq obs_c\left(t\right)$, same as above;

if $obs_c\left(step(s,a)\right) = obs_c\left(s\right) \vee obs_c\left(step(t,a)\right) = obs_c\left(t\right)$, as $s \sim_c t$, so $obs_c\left(s\right) = obs_c(t)$

$$\Rightarrow obs_c\left(step(s,a)\right) = obs_c\left(step(t,a)\right)$$
$$\Rightarrow s \cdot a \sim_c t \cdot a$$

So WSC is proved.

And at last, we need to prove LR. According to lemma 1, as $dom(a) \not\rightarrowtail C$, it is known that $\forall\ i, f_{est}(dom(a), C, T_i) = false$, which means that there's no trusted channel between $dom(a)$ and C. And as Rule 1 says, there is no connection between $dom(a)$ and C. So simply we have $obs_C(s) = obs_C(step(s,a))$, and according to the definition of $\sim_C$, it can be transferred into $s \sim_C step(s,a)$. So this architecture satisfies LR.

Theorem 2 illustrates that the information flow security of refinement approach based on trusted channel can be guaranteed.

5. DISCUSSION

There are two advantages of architectural refinement approach based on trusted channel, verified extra information flow and consistency of TCB structure.

5.1. Verified Extra Information Flow

There are four kinds of information flows should be concerned in the architecture:

- Information flows from low security level component to high security level component. Apparently, this kind of information flows is free and no verification is needed. After the refinement process, component HDV will verify the income information and distribute it to corresponding subcomponents C_{H1} and C_{H2}.
- Information flows from high security level component to low security level component. First, downgrader D takes charge of arbitrating that whether the information allowed to flow to low security level component or not. If the information flow is permitted, then D lowers the security level of the information and give the permission to this information flow.
- Information flows between refined subcomponents but not security sensitive. Because that there is no security issue involved, this sort of information flows is free to flow too.
- Information flows between refined subcomponents and which are security sensitive. Trusted channel is used to verify this kind of information flows. Due to that the information flowing in trusted channel is encrypted so that the integrity of information is assured. Also, the content of information flows will be used to determine the establishment of trusted channel in the process of negotiation by trusted channel ports. So when a trusted channel is established between two subcomponents, it is indicated that the information within it is verified and under surveillance.

5.2. Consistent of TCB Structure

TCB expand is a bottom-up process. It starts with a TCB initial core and splits the entire TCB into hierarchical TCB subsets based on security policies. There are two features of TCB expand: 1) TCB initial core and 2) TCB subsets which are connected with trusted channels. And similarly, architectural refinement is mean to divide the components in the original architecture into multiple subcomponents based on the information flow relationship. There are also two features with architectural refinement using trusted channel: 1) one single component divided into several subcomponents and 2) subcomponents with security functions connected with trusted channels.

The significant difference between TCB expand and architectural refinement is the TCB initial core. However, with a trusted architecture, it is able to be considered that there are a trusted core with the architecture. Due to the trust of the raw architecture with the process of refinement, it is appropriate to regard architectural refinement with trusted channel as TCB expand. Hence, it is convincible to illustrate the characteristic of architectural refinement using trusted channel with TCB expand. As shown in [22], the consistency of TCB structure would be reserved in TCB expand, so the architectural refinement with trusted channel would not break the consistency of TCB structure either and would not increase the difficulty of verifying the refined architecture.

6. CONCLUSION

In conclusion, we have analyzed two issues brought by architectural refinement in MLS environment: unsupervised information during the process of refinement and the loss of consistency of security structure in the refined architecture. We have studied the reason of the issues and figured out that trusted channels would be a solution for this problem. With trusted channel, we have made the following contributions: we proposed an architectural refinement approach based on trusted channel in MLS environment. And with this approach, two subcomponents, which are refined from one single component, should be connected with trusted channel if the subcomponents are with different security label. And meanwhile, we use TCB expand process to illustrate the refinement process for the two processes are similar, so that the refinement with trusted channel would follow the rules of TCB expand and generate hierarchical TCB structure in the refined architecture.

To verify the correctness and security of our approach, we use the unwinding theory of noninterference model to prove both the trusted channel security and the information flow security of the refined architecture with this approach. With the theoretical proof, it could be considered that this approach is applicable in actual realistic environment. In our future work, we will study the specific application of the approach and the performance, etc. The CC (Common Criteria) [29] evaluation would be an application scene of our approach which satisfies the conditions of higher level such as EAL4 in CC.

CONFLICT OF INTERESTS

The authors declare that there is no conflict of interests regarding the publication of this paper.

ACKNOWLEDGMENTS

This work was supported by Research Fund for the Doctoral Program of Higher Education of China (RFDP20120009110007), Program for Innovative Research Team in University of Ministry of Education of China (IRT201206) and Program for Science and Technology Research and Development of Ministry of Railway of China (2012X010-B).

REFERENCES

[1] A. F. Jim, W. H. Harrison, O. Paul, and T. Carol, "The MILS architecture for high-assurance embedded systems", *International Journal of Embedded Systems*, *vol.* 2, no. 3-4, pp.239-247. 2007.

[2] Barbosa and M. Antonio, "A refinement calculus for software components and architectures", in ESEC/FSE-13 Proceedings of the 10th European software engineering conference held jointly with 13th, 2005, pp. 377-380.

[3] K. S. Barber, T. Graser, and J. Holt, "Enabling iterative software architecture derivation using early non-functional property evaluation", in Proceedings of 17th IEEE International Conference on Automated Software Engineering, 2002, pp. 172-182.

[4] A. Lawrence, B. A. Nixon, and E. Yu, "An approach to building quality into software architecture", in Proceedings of the 1995 conference of the Centre for Advanced Studies on Collaborative research IBM Press, 1995.

[5] M. Denford, J. Leaney, and O'Neill T, "Non-functional refinement of computer based systems architecture", in proceedings of 11th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems, 2004, pp. 168-177.

[6] F. Xavier and P. Botella, "Putting non-functional requirements into software architecture", in proceedings of International Workshop on IEEE Software Specification and Design, 1998, pp, 60.

[7] R. S. Nelson, G. R. R. Justo, and P. R. F. Cunha, "A framework for building non-functional software architectures", in Proceedings of the 2001 ACM symposium on Applied computing ACM, 2001.

[8] V. Ambriola and A. Kmiecik, "Architectural transformations", in Proceedings of the 14th international conference on Software engineering and knowledge engineering ACM, 2002, pp. 275-278.

[9] G. David, "Style-Based Refinement for Software Architecture", in Proceedings of the Second International Software Architecture Workshop (ISAW-2), 1996, pp. 72-75.

[10] A.Sabelfeld and A. C. Myers, "Language-Based Information-Flow Security", IEEE Journal on Selected Areas in Communications, vol. 21, no.1, 2003, pp. 5-19.

[11] C. M. Andrew and L. Zheng, "End-to-End Availability Policies and Noninterference", in Computer Security Foundations Workshop of IEEE Computer Society, 2005, pp. 272-286.

[12] J. Zhou and J. Alves-Foss, "Architecture-based refinements for secure computer systems design", in Proceedings of the 2006 International Conference on Privacy, Security and Trust: Bridge the Gap between PST Technologies and Business Services ACM, 2006.

[13] R. V. D. Meyden, "Architectural refinement and notions of intransitive noninterference", *Formal Aspects of Computing*, vol. 24, no. 4-6, pp. 769-792, 2012.

[14] Y. Li, G. Li, and C. X. Shen, "Research of trusted channel model", *Computer Engineering and Applications*, vol. 47, no. 26, pp.70-73, 2011.

[15] Z. S. Ye, S. Smith, and D. Anthony, "Trusted Paths for Browsers", In Proceedings of the 11th USENIX Security Symposium, 2002, pp. 263-279.

[16] J. A. Goguen and J. Meseguer, "Security policies and security models", *IEEE Symposium on Security and privacy*, vol. 12, 1982.

[17] J. A. Goguen and J. Meseguer, "Inference control and unwinding", in Proceedings of the Symposium on Security & Privacy IEEE Computer Society, 1984.

[18] J. T. Haigh and W. D. Young, "Extending the Noninterference Version of MLS for SAT", *Software Engineering IEEE Transactions*, vol. 13, no. 2, pp. 141-150, 1987.

[19] J. Rushby, "Noninterference, transitivity and channel-control security policies", *SRI International, Computer Science Laboratory,* 1992.

[20] R. V. D. Meyden, "What, Indeed, Is Intransitive Noninterference", Lecture Notes in Computer Science, vol. 4734, pp. 235-250, 2007.

[21] J. H. Liao, Y. Zhao ,and C. X. Shen, "Channel-Based TCB Extension Model", *Journal of Beijing University of Technology*, vol. 36, no. 5, pp. 592-596, 2010.

[22] Y. Li, F. Wang, J. Hu, and C. X. Shen, "Research of trusted expand model of TCB", *Computer Engineering and Applications*, vol. 46, no. 13, pp. 1-3, 2010.

[23] D. E. Bell and L. J. Lapadula, "*Secure Computer Systems: Mathematical Foundations*", Secure Computer Systems Mathematical Foundations, 1973.

[24] K. J. Biba, "*Integrity Considerations for Secure Computer Systems*", Electronic Systems Division Air Force Systems Command, 1977.

[25] D. E. Denning and P. J. Denning, "Certification Of Programs For Secure Information Flow", *Communications of the Acm*, vol. 20, no. 7, pp. 504-513, 1977.

[26] J. Zhou and J. A. Foss, "Security policy refinement and enforcement for the design of multi-level secure systems", *Journal of Computer Security*, vol. 16, no. 2, pp. 107-131, 2008.

[27] Y. Deng, J. Wang, J. J. P. Tsai, and K. Beznosov, "An approach for modeling and analysis of security system architectures", *IEEE Transactions on Knowledge & Data Engineering*, vol. 15, no. 5, pp. 1099-1119, 2003.

[28] M. Moriconi, "Secure software architectures", in proceedings of IEEE Symposium on Security and Privacy Proceedings, 1997.

[29] ISO/IEC 15408 Standard. Common Criteria for Information Technology Security Evaluation Version 3.1 Revision 4, 2014. http://www.commoncriteriaportal.org/cc/

Received: September 16, 2014 Revised: December 23, 2014 Accepted: December 31, 2014