Research Article

Task-Oriented Multilevel Cooperative Access Control Scheme for Environment with Virtualization and IoT

Jian Dong,^{1,2} Hui Zhu¹,¹ Chao Song,¹ Qiang Li,³ and Rui Xiao¹

¹State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710017, China

²Science and Technology on Communication Networks Laboratory, Shijiazhuang 050081, China

³Hangzhou Research Institute, NetEase Network Co., Ltd., Hangzhou 310000, China

Correspondence should be addressed to Hui Zhu; zhuhui@xidian.edu.cn

Received 9 March 2018; Revised 16 May 2018; Accepted 23 June 2018; Published 10 July 2018

Academic Editor: Kim-Kwang Raymond Choo

Copyright © 2018 Jian Dong et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the development of cloud computing technology and the proliferation of the Internet of Things (IoT) terminals, more and more scenes need the collaboration of virtual machines and IoT terminals to resolve. However, there are many severe challenges on the security of virtual machines and IoT terminals. Based on Bell-LaPadula Model (BLP), a task-oriented multilevel cooperative access control scheme virtualization and reality BLP, named VR-BLP, is proposed. Specifically, tasks are created for each user of the platform and tasks and users are divided into multiple levels to provide more granularities to limit access between virtual machines and IoT terminals. Moreover, with network isolation cooperating with process isolation and shared memory isolation mechanisms, VR-BLP is implemented to enhance the security isolations between tasks. Performance evaluations show that VR-BLP enhanced the security of environment with virtualization and IoT without causing significant performance penalty.

1. Introduction

Since it was created in 2006, cloud computing [1–3] has been growing more and more popular in today's society. With wide popularity and broad application, it is playing a more and more important role in the development of the society, not only making enterprises get more income while saving more costs, but also providing more convenient online services for ordinary consumers. As the basis of cloud computing, virtualization [4–6] technologies including system virtualization and network virtualization have been widely used on cloud platform, helping cloud computing to develop faster and faster. On the other hand, with more and more connected devices expected to be in use, the Internet of Things (IoT) [7] has become a critical focus area for many enterprises.

To address security challenges in cloud environment, many access control schemes are proposed, such as RBAC [8] and TBAC [9]. These schemes partially enhance the security of access control in cloud but do not apply to resource management for environment with virtualization and IoT. When integrating virtualization and IoT to build a bigger platform

which could allocate virtual machines and IoT terminals to users at the same time, enterprises and consumers could benefit more from virtualization and IoT [10]. For service provider enterprises, the hybrid platform could provide more kinds of combinations of services to clients, save the costs to build two kinds of platforms which are cloud computing platforms and IoT platforms, and achieve more efficient utility of the resources and the energies. For service buyer enterprises, the hybrid platform could provide more kinds of services meeting the needs of their own business, saving their money and energies to maintain two kinds of systems. For ordinary consumers, they could buy more flexible service according to their needs and budget. In short, it is of great use to integrate virtualization and IoT together.

Figure 1 shows a scene in which there exist multiple users and multiple tasks in a hybrid virtualization and IoT platform. In the hybrid platform, computing and storage resources form virtual resource pool through virtualization technologies, and IoT terminals form IoT terminal resource pool through network connections and network equipment form network equipment resource pool. Authorized users of the system could create a set of tasks and every task

FIGURE 1: Multiuser and multitask scenes over virtualization and IoT.

contains a set of resources from resources pool. However, the integration of virtualization and IoT results in security issues which could become a critical problem. Virtualization and IoT may incur new vulnerabilities to the hybrid virtualization and IoT platform [11]. Even though virtualization and IoT security have attracted considerable interest in recent years and several solutions have been proposed, the flourish of secure solutions for virtualization and IoT still faces many challenges in the balance between information sharing and privacy preservation [12]. Security issues on virtualization and IoT have been the vital barrier to the development of the integration of virtualization and IoT.

In order to construct a task-oriented secure isolation mechanism for environment with virtualization and IoT, a new multilevel cooperative access control scheme named VR-BLP is proposed in this paper. And our main contributions are threefold.

- (1) A task-oriented multilevel cooperative access control scheme for environment with virtualization and IoT, named VR-BLP, is proposed to enhance the security isolations between users and tasks.
- (2) Network isolation cooperates with process isolation and shared memory isolation to enhance security isolations between virtual machines and IoT terminals.
- (3) Performance evaluations show that VR-BLP is an efficient multilevel access control scheme for environment with virtualization and IoT.

The remainder of this paper is organized as follows. In Section 2, we survey the related works. In Section 3, we introduce the preliminaries of this paper. In Section 4, we present the architecture of our proposed VR-BLP scheme and provide the security proof of the VR-BLP scheme. Then, we present the implementation in Section 5, followed by evaluations in Section 6. Finally, we draw our conclusions in Section 7.

2. Related Work

The BLP [13, 14] model is the first access model in the sense of mathematical which was proposed by Bell and LaPadula in 1973. It is a state machine model based on a simulated military security strategy. With the BLP model becoming more and more popular, it was applied to a lot of different scenes to achieve more secure isolations of resources. Multiple level security (MLS) [15] has always been a focus of attention since the usage of computers in military and intelligence systems. With the development of cloud computing, scholars have done a lot of researches about the multiple level security of clouds.

A MUSHI system [16] toward multiple level security cloud with strong hardware level isolation was designed to provide hardware level isolation and protection to individual guest virtual machine (VM) execution. With MUSHI, a user could maintain confidentiality and integrity of his/her VM in a multicore environment even in the presence of malicious attacks from both within and outside the cloud infrastructure. A BLP-based multilevel security model [17] of workflow deployment over an architecture for federated clouds on the background of medical data security was proposed, which divided the medical service transactions into several states, and one state can be transformed to another by specific operations. By defining security operations, the model proves the system could keep secure. A BLP-based multilevel security model [18] for private cloud was proposed and was proved secure by mathematical method. The model used mandatory access control method to control user's operation and can guarantee that users cannot leak sensitive data after they read them. A Centralized Pervasive Computing Environment/Multilevel Security (CPCE/MLS) system [19] was designed to provide the security guarantee of the pervasive computing environment by introducing the server-storage terminals and implementing the multilevel security access control mechanism based on BLP model, process creation

supervision, and an auditing mechanism. A multilevel secure file sharing server [20] was designed to satisfy the requirements for certifiable, scalable, and multilevel cloud security and this paper showed how the secure file server can be used to create a high-assurance, MLS storage cloud. The file server was built on mature technology that was previously certified and deployed across domains and that supports high-performance, low-to-high, and high-to-low file sharing with verifiable security.

Though these works are very meaningful and valuable, they do not consider the security issues in a task-oriented environment with virtualization and IoT. To satisfy the requirements for environment with virtualization and IoT, a task-oriented cooperative access scheme VR-BLP is proposed and implemented with network isolation cooperating with process isolation and shared memory isolation mechanisms, to enhance the security isolations between tasks.

3. Preliminaries

In this section, we will introduce the knowledge related to the design and the implementation of our VR-BLP scheme.

3.1. BLP. While giving the definition of the subject, the object, the security level function, the state, the state transition rules, and so on, BLP model defines that a system is secure if and only if the system always satisfies the simple security property, the *-property, and the discretionary security property. Given a secure initial state, the state of the system retains security if every state transition satisfies the simple security property, the *-property, and the discretionary property. Through managing these state transitions, BLP prevents the leakage of confidential information in the process of information sharing. The subject of higher clearance level and larger category could access to the object with lower classification and smaller category with certain access attributes. The attributes include the read access, the write access, the append access, the execute access, and the control access. About the state transition rules, the BLP model designs a set of request elements that the subjects could make, including get, give, release, rescind, change, create, and delete. The BLP model develops a set of rules which are proof to satisfy the simple security property and the *-property. Therefore, a system which has implemented the BLP model is secure and could protect the privacy between information sharing.

3.2. LSM. LSM [21] is a framework in the Linux kernel that supports various computer security models and LSM has nothing to do with any separate security implementation. This framework is licensed under the GNU General Public License and it has been a part of the official Linux kernel since Linux 2.6. By providing a general purpose framework for security policy modules, LSM allows many different access control schemes to be implemented as loadable kernel modules and hence enables these security policies to develop independently. A quantity of existing access control implementations, including SELinux, Domain, and Type Enforcement (DTE) and Linux Intrusion Detection System (LIDS) has already been adapted to use the LSM framework.

3.3. SDN. SDN [22–24] uses stratification ideas to separate data and control. The control layer mainly includes a logic-centric and programmable controller so that it can grasp the global network information and it is convenient for operators and researchers to manage network configurations and the deployment of new protocols. There is a switch at the data layer which provides simple data forwarding to match data packets, so it can be quickly processed to meet the growing demands for traffic. The two layers use an open unified interface to interact. The controller sends unified standard rules to the switch through standard interfaces. The switch only needs to act according to these rules. Therefore, the SDN technology can effectively reduce the equipment load, help network operators control the infrastructure, and reduce overall operating costs. It has gradually become one of the most promising network technologies.

4. VR-BLP Scheme

In this section, we redefine the architecture, elements, security properties, and state transition rules of the BLP model for better applying to the circumstance of the environment with virtualization and IoT. And we propose a task-oriented multilevel access control scheme for environment with virtualization and IoT to construct a secure isolation between users and tasks.

4.1. Elements of VR-BLP Model. Before introducing the formal definitions of the task-oriented multilevel access control scheme for virtualization and IoT, we define the basic elements of the VR-BLP model as below.

Subject and Object. In VR-BLP model, one user is a subject and one task is an object. A user creates a set of tasks and each task creates a set of resources. Through mapping resources to tasks, users access resources by accessing tasks.

Security Functions Sets. F is a security function set and an arbitrary element of F contains three components $f_m(s)$, $f_c(s)$, and $f(o)$. $f_m(s)$ represents the highest clearance level of the subject s , $f_c(s)$ represents the current clearance level of the subject s , and $f(o)$ represents the classification of the object o .

Access Attribute Set. The access attribute set A contains three elements, $A = \{r, a, w\}$, where r stands for read-only, a stands for write-only, and w stands for read-write. A subject could only access an object with an access attribute included in the access attribute set A .

Access Matrix. The access matrix M contains a set of access attributes which record how a subject could access to an object. A subject could only access an object with an access attribute stored in the access matrix.

Current Access Set. $B = S \times O \times A$ represents the current set. An arbitrary element of B is written b . $b = (s_i, o_j, x)$ indicates the subject s_i has access to the object o_j in mode $x \subseteq m_{ij}$.

Current State of the System. $v = (b, M, f) \in V$ stands for an arbitrary state of the system in which $b = (s_i, o_j, x) \in B$, $x \subseteq m_{ij}$, $f = (fs, fo) \in F$.

Request Elements. The request elements include read-only, write-only, and read-write.

Requests. The requests $R = S \times RA \times O \times X$, where $X = A \cup F$. An arbitrary element of R is written R_k .

Decisions. The decisions include yes and no which stand for the decisions that rules make.

Relation. W stands for a relation that will be the union of partial functions which constitute the rules of operation of the system with respect to preservation of the simple security property and the *-property.

Rules. A rule is a function $\rho : R \times V \rightarrow D \times V$ that represents what the response and the state change are when a request and a state are inputted. It decides that what the system should react to the request of users and makes the system change according to the specific situation.

4.2. Security Properties

(1) *Simple Security Property.* The state $v = (b, M, f)$ satisfies the simple security property, if and only if for $\forall(s, o, x) \in b$

- (i) $x = a$
- (ii) $x = r \text{ or } x = w \implies f_m(s) \geq f(o)$

The simple security property means that, for $\forall(s, o, x) \in b$, when the subject s tries to access the object o with the access attribute a , the state $v = (b, M, f)$ satisfies the simple security property. When the subject s tries to access the object o with the access attribute r , the state $v = (b, M, f)$ satisfies the simple security property if the highest clearance level of the subject s is greater than or equal to the classification of the object o . When the subject s tries to access the object o with the access attribute w , the state $v = (b, M, f)$ satisfies the simple security property if the highest clearance level of the subject s is greater than or equal to the classification of the object o .

(2) **-Property.* The state $v = (b, M, f)$ satisfies *-property, if and only if $\forall(s, o, x) \in b, s \in S/S^*$, where S^* is trusted subjects

- (i) $x = r \implies f_c(s) \geq f(o)$
- (ii) $x = a \implies f_c(s) \leq f(o)$
- (iii) $x = w \implies f_c(s) = f(o)$

The *-property means that, for $\forall(s, o, x) \in b, s \in S/S^*$, where S^* is trusted subjects, when the subject s tries to access the object o with the access attribute r , the state $v = (b, M, f)$ satisfies the *-property if the current clearance level of the subject s is greater than or equal to the classification of the object o . When the subject s tries to access the object o with the access attribute a , the state $v = (b, M, f)$ satisfies the *-property if the current clearance level of the subject s is lesser or equal to the classification of the object o . When the subject s tries to access the object o with the access attribute w , the state $v = (b, M, f)$ satisfies the *-property if the current clearance level of the subject s is equal to the classification of the object o .

(3) *Discretionary Security Property.* The state $v = (b, M, f)$ satisfies discretionary property, if and only if $\forall(s_i, o_j, x) \in b, x \in m_{ij}$.

(4) *Fundamental Security Property.* The system is secure if and only if the initial state of the system is secure and every

state transition satisfies the simple security property, the *-property, and the discretionary property.

In our scheme, the subjects are user and the objects are task. Tasks are created by users and are assigned with different classifications. Users have different clearance levels given by the system as we designed. Our purpose is to make users with higher clearance levels able to access tasks with lower classifications. While tasks contain a set of resources, users with higher clearance levels could actually read data from resources with lower classifications, whether these resources are virtual machines or IoT terminals. Users with higher clearance levels could not write data to locations that belong to resources with lower classifications, whether these resources are virtual machines or IoT terminals. On the contrary, users with lower clearance levels could not read data from resources with higher classifications, whether these resources are virtual machines or IoT terminals. Users with lower clearance levels could write data to locations that belong to resources with higher classifications, whether these resources are virtual machines or IoT terminals.

4.3. *State Transition Rules.* A rule is a function $\rho : R \times V \rightarrow D \times V$. As shown in Figure 2, the interpretation of a rule is that, given a request and a state, a rule decides a response and a state change. R is a set of request; D is a set of decisions. The result of decisions is one of the sets of (yes, no) . Yes represents that the request is allowed to execute, and No represents that the request is denied to execute.

R_k is an arbitrary element of R , D_m is an arbitrary element of D , and a rule ρ is security preserving if and only if $\forall(R_k, v) \in R \times V, \rho(R_k, v) = (D_m, v^*)$ and v is secure $\implies v^*$ is secure. Based on operations for environment with virtualization and IoT, we defined three state transition rules.

Rule 1. Read-only:

Given state $v = (b, M, f) \in V, s \in S \setminus S^*$, where S^* is trusted subjects, the handling process to request $R_k = (s_i, o_j, r)$ is as follows.

If $f_m(s_i) \geq f(o_j) \wedge r \in m_{ij}$, then set $f^* = f \cup \{\max(f_c(s_i), f(o_j)), b^* = b \cup \{(s_i, o_j, r)\}\}$

$$\rho 1(R_k, v) = (yes, (b^*, M, f^*))$$

Else

$$\rho 1(R_k, v) = (no, (b, M, f))$$

Rule 1 describes what the response is and what the system state is when the subject s_i requests to access the object o_j with access attribute r . When the request is submitted, the access control modules check if the request is allowed. First, the access control modules check the highest clearance level of the subject s_i and compare it with the classification of the object o_j . If the highest clearance level of the subject s_i is greater than or equal to the classification of the object o_j , then the access control modules check if the access attribute r satisfies the requirement $r \in m_{ij}$. Only when the two requirements are satisfied, the request is allowed to execute. Otherwise, the request is denied. If the request is allowed to execute, the access control modules calculate the next

FIGURE 2: Proposed VR-BLP scheme.

system state. As a result of Rule 1, the system state $v = (b, M, f)$ becomes $v^* = (b^*, M, f^*)$, where $f^* = f \cup \{\max(f_c(s_i), f(o_j))\}$, $b^* = b \cup \{(s_i, o_j, r)\}$. The new security function generated after Rule 1 executes is the greater one of the two security functions $f_c(s_i)$ and $f(o_j)$. The new security function sets f^* is the combinations of the previous security function sets f and the greater one of the two security functions $f_c(s_i)$ and $f(o_j)$. The new current access set b^* is the combinations of the previous current access sets b and the present current access set (s_i, o_j, r) . Finally, if Rule 1 is allowed to execute, the response is yes and the system state $v = (b, M, f)$ becomes $v^* = (b^*, M, f^*)$, where $f^* = f \cup \{\max(f_c(s_i), f(o_j))\}$, $b^* = b \cup \{(s_i, o_j, r)\}$.

Rule 2. Write-only:

Given state $v = (b, M, f) \in V$, $s \in S \setminus S^*$, where S^* is trusted subjects, the handling process to request $R_k = (s_i, o_j, a)$ is as follows.

If $f_m(s_i) \geq f(o_j) \wedge f_c(s_i) \leq f(o_j) \wedge a \in m_{ij}$, then set $b^* = b \cup \{(s_i, o_j, a)\}$

$$\rho_2(R_k, v) = (\text{yes}, (b^*, M, f))$$

Else

$$\rho_2(R_k, v) = (\text{no}, (b, M, f))$$

Rule 2 describes what the response is and what the system state is when the subject s_i requests to access the object o_j with access attribute a . When the request is submitted, the access control modules check if the request is allowed. First, the access control modules check the highest clearance level of the subject s_i and compare it with the classification of the object o_j . If the highest clearance level of the subject s_i is greater than or equal to the classification of the object o_j , then the access control modules check if the current clearance level of the subject s_i is lesser or equal to the classification of the object o_j . If the current clearance level of the subject s_i is lesser or equal to the classification of the object o_j , then the access control modules will check if the access attribute a satisfies

the requirement $a \in m_{ij}$. Only when the three requirements are satisfied, the request is allowed to execute. Otherwise, the request is denied. If the request is allowed to execute, the access control module will calculate the next system state. As a result of Rule 2, the system state $v = (b, M, f)$ becomes $v^* = (b^*, M, f)$, where $b^* = b \cup \{(s_i, o_j, a)\}$. The new current access sets b^* is the combinations of the previous current access sets b and the present current access set (s_i, o_j, a) . Finally, if Rule 2 is allowed to execute, the response will be yes and the system state $v = (b, M, f)$ becomes $v^* = (b^*, M, f)$, where $b^* = b \cup \{(s_i, o_j, a)\}$.

Rule 3. Read-Write:

Given state $v = (b, M, f) \in V$, $s \in S \setminus S^*$, where S^* is trusted subjects, the handling process to request $R_k = (s_i, o_j, r)$ is as follows.

If $f_m(s_i) \geq f(o_j) \wedge f_c(s_i) \leq f(o_j) \wedge w \in m_{ij}$, then set $f^* = f \cup \{f(o_j)\}$, $b^* = b \cup \{(s_i, o_j, w)\}$

$$\rho_3(R_k, v) = (\text{yes}, (b^*, M, f^*))$$

Else

$$\rho_3(R_k, v) = (\text{no}, (b, M, f))$$

Rule 3 describes what the response is and what the system state is when the subject s_i requests to access the object o_j with access attribute a . When the request is submitted, the access control modules check if the request is allowed. First, the access control modules check the highest clearance level of the subject s_i and compare it with the classification of the object o_j . If the highest clearance level of the subject s_i is greater than or equal to the classification of the object o_j , then the access control modules check if the current clearance level of the subject s_i is lesser or equal to the classification of the object o_j . If the current clearance level of the subject s_i is lesser or equal to the classification of the object o_j , then the access control modules will check if the access attribute w satisfies the requirement $w \in m_{ij}$. Only when the three requirements are satisfied, the request is allowed to execute. Otherwise, the

request is denied. If the request is allowed to execute, the access control modules calculate the next system state. As a result of Rule 3, the system state $\nu = (b, M, f)$ becomes $\nu^* = (b^*, M, f^*)$, where $f^* = f \cup \{f(o_j)\}$, $b^* = b \cup \{(s_i, o_j, w)\}$. The new security function generated after Rule 1 executes is the security function $f(o_j)$. The new security function sets f^* is the combinations of the previous security function sets f and the security function $f(o_j)$. The new current access sets b^* is the combinations of the previous current access sets b and the present current access set (s_i, o_j, r) . Finally, if Rule 3 is allowed to execute, the response is yes and the system state $\nu = (b, M, f)$ becomes $\nu^* = (b^*, M, f^*)$, where $f^* = f \cup \{f(o_j)\}$, $b^* = b \cup \{(s_i, o_j, w)\}$.

4.4. Security Proof of the Model

Proof (Rule 1 keeps secure). Let the initial state $\nu = (b, M, f) \in V$ be secure. For the request $R_k = (s_i, o_j, r) \in R$, after executing Rule 1, we get a new state ν^* ; then $\nu^* = \nu$ or $\nu^* = (b^*, M, f^*)$. If $\nu^* = \nu$, then ν^* is secure since ν is secure, else if $\nu^* = (b^*, M, f^*)$: $\nu^* - \nu = ((s_i, o_j, r), M, f^*)$, where $f^* = f \cup \{\max(f_c(s_i), f(o_j))\}$, and $f_m(s_i) \geq f(o_j)$, so $\nu^* - \nu$ satisfies the simple security property. Since ν satisfies the simple security property, ν^* satisfies the simple security property. Since $f^* - f = \max(f_c(s_i), f(o_j)) \geq f(o_j)$, then $\nu^* - \nu$ satisfies the star-property. From the above, we prove that Rule 1 keeps secure. $\square$

Proof (Rule 2 keeps secure). Let the initial state $\nu = (b, M, f) \in V$ be secure. For the request $R_k = (s_i, o_j, r) \in R$, after executing Rule 2, we get a new state ν^* , then $\nu^* = \nu$ or $\nu^* = (b^*, M, f)$. If $\nu^* = \nu$, then ν^* is secure since ν is secure, else if $\nu^* = (b^*, M, f)$: $\nu^* - \nu = ((s_i, o_j, r), M, f)$, where $f_m(s_i) \geq f(o_j)$, so $\nu^* - \nu$ satisfies the simple security property. Since ν satisfies the simple security property, ν^* satisfies the simple security property. Since $f_c(s_i) \leq f(o_j)$, then $\nu^* - \nu$ satisfies the star-property. Since ν satisfies the star-property, ν^* satisfies the star-property. From the above, we proof that Rule 2 keeps secure. $\square$

Proof (Rule 3 keeps secure). Let the initial state $\nu = (b, M, f) \in V$ be secure. For the request $R_k = (s_i, o_j, r) \in R$, after executing Rule 1, we get a new state ν^* ; then $\nu^* = \nu$ or $\nu^* = (b^*, M, f^*)$. If $\nu^* = \nu$, then ν^* is secure since ν is secure, else if $\nu^* = (b^*, M, f^*)$: $\nu^* - \nu = ((s_i, o_j, r), M, f^*)$, where $f^* = f \cup \{f(o_j)\}$, and $f_m(s_i) \geq f(o_j)$, so $\nu^* - \nu$ satisfies the simple security property. Since ν satisfies the simple security property, ν^* satisfies the simple security property. Since $f^* - f = f(o_j)$, then $\nu^* - \nu$ satisfies the star-property. From the above, we prove that Rule 3 keeps secure. $\square$

5. Implementation of VR-BLP

This section will mainly introduce the application scenarios of VR-BLP scheme and the specific design architecture of our scheme.

5.1. Application Scenarios. As shown in Figure 3, users are divided into different groups with different clearance levels and each user creates a set of tasks allocated with classifications the same as the user's clearance level. The policy making module guides security label module how to add different security label to different tasks according to the tasks' classifications and it notices the access control module to execute state transition rules to isolate resources.

5.2. System Modules. The system modules consist of three parts: the policy making module, the security label module, and the access control module.

5.2.1. Policy Module. Users are assigned with different clearance levels according to the needs of the design, and tasks are assigned with classifications which are equal to the clearance level of the user who create those tasks. Since tasks contain a set of resources, resources are mapping to tasks resulting in that users achieve the multilevel security access control of resources. Therefore, the security label module adds different security labels to tasks and the access control modules isolate resources through network isolation, memory isolation, and process isolation. The policy making module makes policies and send these policies to the security module and the access control module to guide them to isolate resource meeting our expectations.

5.2.2. Security Label Module. The security label module receives security policies from the policy making module and attach security labels to tasks and resources. For tasks, they are attached with security labels according to their classifications. For resources, they are attached with the same security labels as the task which creates them. In this way, resources could be isolated from others by access control module through their security labels easily.

5.2.3. Access Control Module. The access control module receives policies from the policy making module and achieves the isolation of resources through process isolation, shared memory isolation, and network isolation. For process isolation and shared memory isolation, they are implemented by a security module named KMAC that we designed based on LSM. Through attaching tasks' security labels to virtual machines and virtual machines' disk images, KMAC does not allow one virtual machine request to access a disk whose security label is different from the virtual machine's security label. Through attaching tasks' security labels to virtual machines and virtual machines' shared memory, KMAC does not allow one virtual machine request to access a shared memory whose security label is different from the virtual machine's security label. For network isolation, we use SDN and traditional network technologies to isolate resources. When a user requests to access a task with access attribute read-only, write-only, or read-write, the request is analyzed by the security switch and the ACL rules in the security switch decides if the request is legal. According to decisions that the security switch makes, the request is allowed or denied to execute.

FIGURE 3: Application framework of VR-BLP.

To strengthen the security isolation of virtual machines, a KMAC module based on LSM is designed. For security reason, KMAC module must be compiled into the Linux kernel as a LSM module. KMAC strengthen the security isolation of the process and the shared memory between virtual machines. As shown in Figure 4(a), a virtual machine is a QEMU process in the host machines. To strengthen the security isolation between QEMU processes, the disk images of each process are isolated. When a QEMU process starts, KMAC module allocates the same unique security label to the QEMU process and its disk image. After that, the QEMU process cannot access other processes with different security labels. As shown in Figure 4(b), mandatory access control of shared memory between virtual machines is implemented in the KMAC module. By using `ivshmem`, a virtual PCI device is added to a virtual machine to create a piece of shared memory. By using `inode_create` and `file_mmap` function of the LSM module, we could control virtual machines' access to shared memory. When a QEMU process starts, KMAC module allocates the same unique security label to the QEMU process and its shared memory. After that, the QEMU process cannot access other processes' shared memory with different security labels.

As shown in Figure 4(c), by using SDN, virtual machines of task10 are attached to virtual network N10 with a VLAN id 10 on the OpenStack platform. Virtual machines of task20 are attached to virtual network N20 with a VLAN id 20. For physical resources, VLAN10 and VLAN20 are created

on the physical layer 3 switch S. Physical resources of task10 are attached to VLAN10 and physical resources of task20 are attached to VLAN20. Resources in VLAN10 could only communicate with those resources in VLAN10 and resources in VLAN20 could only communicate with those resources in VLAN20. Therefore, Resources in task10 could only communicate with those resources in task10 and resources in task20 could only communicate with those resources in task20. The security switch receives policies from the policy making module and make corresponding ACL rules to control users' request to tasks. According to the state transition rules in VR-BLP scheme, the security switch gets users' request and makes decisions.

6. Evaluation of VR-BLP

6.1. Environments of Evaluation. As shown in Figure 5, two workstations which have 32 cores, 32GB RAM, and dual network interface card serve as a controller node and a compute node of the OpenStack platform. Four PCs with wireless network adapters connected by wireless routers are used to simulate IoT terminals. A switch connects the wireless router with two workstations.

6.2. Device Configuration for Evaluation. In Figure 6(a), four tasks of one user are created and task1, task2, task3, and task4 are assigned with security levels 1, 2, 3, and 4 respectively, while security level $4 > 3 > 2 > 1$. Each task contains two

FIGURE 4: Resources isolation.

FIGURE 5: Topology of test environment.

KVM virtual machines and a PC and these resources are all allocated with a unique IP address. Resources of task1 are in the VLAN101 network. Resources of task2 are in the VLAN102 network. Resources of task3 are in the VLAN103 network. Resources of task4 are in the VLAN104 network. In Figure 6(b), four users are assigned with security levels 1, 2, 3, and 4, respectively. User1, user2, user3, and user4 all create

a task1 with the same security level within one user group. Each task1 contains two KVM virtual machines and a PC. These resources are all allocated with a unique IP address. Resources of user1 are in VLAN105 network. Resources of task2 are in VLAN201 network. Resources of task3 are in VLAN301 network and resources of task4 are in VLAN401 network.

Tasks ◌		IP Address ◌	VLAN ID ◌
Task1 ◌	Kvm10 ◌	10.1.1.13 ◌	101 ◌
	Kvm11 ◌	10.1.1.4 ◌	
	PC1 ◌	10.1.1.14 ◌	
Task2 ◌	Kvm20 ◌	10.1.2.29 ◌	102 ◌
	Kvm21 ◌	10.1.2.17 ◌	
	PC2 ◌	10.1.2.61 ◌	
Task3 ◌	Kvm30 ◌	10.1.3.18 ◌	103 ◌
	Kvm31 ◌	10.1.3.23 ◌	
	PC3 ◌	10.1.3.34 ◌	
Task4 ◌	Kvm40 ◌	10.1.4.51 ◌	104 ◌
	Kvm41 ◌	10.1.4.67 ◌	
	PC4 ◌	10.1.4.34 ◌	

Users ◌			IP Address ◌	VLAN ID
User1 ◌	Task1 ◌	Kvm10 ◌	10.1.1.12 ◌	105 ◌
		Kvm11 ◌	10.1.1.3 ◌	
		PC1 ◌	10.1.1.10 ◌	
User2 ◌	Task1 ◌	Kvm20 ◌	10.2.1.9 ◌	201 ◌
		Kvm21 ◌	10.2.1.14 ◌	
		PC2 ◌	10.2.1.6 ◌	
User3 ◌	Task1 ◌	Kvm30 ◌	10.3.1.8 ◌	301 ◌
		Kvm31 ◌	10.3.1.15 ◌	
		PC3 ◌	10.3.1.11 ◌	
User4 ◌	Task1 ◌	Kvm40 ◌	10.4.1.5 ◌	401 ◌
		Kvm41 ◌	10.4.1.19 ◌	
		PC4 ◌	10.4.1.16 ◌	

(a) Policies of access control between different tasks

(b) Policies of access control between different users

FIGURE 6: Policies of test environment.

6.3. Result Analysis. In this section, the above isolation experiments are conducted. The check mark in Tables 1 and 2 means that one virtual machine or PC could access another virtual machine or PC. As shown in Table 1, resources of task1 could access resources of task1. Resources of task2 could access resources of task1 and task2. Resources of task3 could access resources of task1, task2, and task3. Resources of task4 could access resources of task1, task2, task3, and task4. Therefore, the experimental results show that the VR-BLP scheme achieves multilevel security isolation between different tasks.

As shown in Table 2, resources of user1 could access resources of user2. Resources of user2 could access to resources of user1 and user2. Resources of user3 could access resources of user1, user2, and user3. Resources of user4 could access resources of user1, user2, user3, and user4. Therefore, the experimental results show that the VR-BLP scheme achieves multilevel security isolation between different users.

At last, as shown in Figure 7, we test the impacts of VR-BLP on the performance of the host and the guest in the case of single CPU and double CPU, respectively, using Unix-Bench [25]. As we can see from the results, our scheme has little impact on the performance of the host and the guest with an overhead of no more than 2%.

7. Conclusion

In this paper, a task-oriented multilevel cooperative access control scheme based on BLP, named VR-BLP, has been proposed. Specifically, the access control between virtual machines and IoT terminals achieves fined-grained isolation by dividing tasks and users into multiple levels. Moreover, VR-BLP enhances the security isolations between tasks through the cooperation of network isolation, process isolation, and shared memory isolation. Performance evaluations show that VR-BLP enhanced the security of environment with virtualization and IoT with only a small performance loss.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

TABLE 1: Access control between different tasks.

Tasks		Access to Tasks			
		Task1	Task2	Task3	Task4
Task1	Kvm10	✓			
	Docker11	✓			
	PC12	✓			
Task2	Kvm20	✓	✓		
	Docker21	✓	✓		
	PC22	✓	✓		
Task3	Kvm30	✓	✓	✓	
	Docker31	✓	✓	✓	
	PC32	✓	✓	✓	
Task4	Kvm40	✓	✓	✓	✓
	Docker41	✓	✓	✓	✓
	PC42	✓	✓	✓	✓

TABLE 2: Access control between different users.

Users		Access to Users			
		User1	User2	User3	User4
User1	Task1	Kvm10	✓		
		Docker11	✓		
		PC12	✓		
User2	Task1	Kvm20	✓	✓	
		Docker21	✓	✓	
		PC22	✓	✓	
User3	Task1	Kvm30	✓	✓	✓
		Docker31	✓	✓	✓
		PC32	✓	✓	✓
User4	Task1	Kvm40	✓	✓	✓
		Docker41	✓	✓	✓
		PC42	✓	✓	✓

Conflicts of Interest

The authors declare that they have no conflicts of interest.

FIGURE 7: Performance comparison of VR-BLP.

Acknowledgments

This work is supported by the National Key Research and Development Program of China (2016YFB0800804), National Natural Science Foundation of China (61672411 and U1401251), Research Foundations for Science and Technology on Communication Networks Laboratory (no. KX172600023), and China 111 Project (no. B16037).

References

- [1] D. Kapil, P. Tyagi, S. Kumar, and V. P. Tamta, "Cloud Computing: Overview and Research Issues," in *Proceedings of the 2017 International Conference on Green Informatics (ICGI)*, pp. 71–76, Fuzhou, China, August 2017.
- [2] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: Architecture, applications, and approaches," *Wireless Communications and Mobile Computing*, vol. 13, no. 18, pp. 1587–1611, 2013.
- [3] E. Aguiar, Y. Zhang, and M. Blanton, "An overview of issues and recent developments in cloud computing and storage security," *High Performance Cloud Auditing and Applications*, pp. 3–33, 2014.
- [4] R. Uhlig, G. Neiger, D. Rodgers et al., "Intel virtualization technology," *The Computer Journal*, vol. 38, no. 5, pp. 48–56, 2005.
- [5] P. Barham, B. Dragovic, K. Fraser et al., "Xen and the art of virtualization," in *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles*, pp. 164–177, Bolton Landing, New York, NY, USA, 2003.
- [6] N. Jain and S. Choudhary, "Overview of virtualization in cloud computing," in *Proceedings of the 2016 Symposium on Colossal Data Analysis and Networking. CDAN 2016*, March 2016.
- [7] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): a vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.
- [8] H.-C. Chen, M. A. Violetta, and C.-Y. Yang, "Contract RBAC in cloud computing," *The Journal of Supercomputing*, vol. 66, no. 2, pp. 1111–1131, 2013.
- [9] R. K. Thomas and R. S. Sandhu, "Task-based authorization controls (TBAC): A Family of models for active and enterprise-oriented authorization management," in *Proceedings of the IFIP WG11.3 Workshop on Database Security*, pp. 166–181, 1998.
- [10] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: a survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.
- [11] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.

- [12] K. Hashizume, D. G. Rosado, E. Fernández-Medina, and E. B. Fernandez, "An analysis of security issues for cloud computing," *Journal of Internet Services & Applications*, vol. 4, no. 1, pp. 1–13, 2013.
- [13] D.-E. Bell and L.-J. LaPadula, *Secure Computer Systems: A Mathematical Model. Volume II*, MITRE Corporation, Bedford, MA, USA, 1973.
- [14] D.-E. Bell and L.-J. LaPadula, *Secure computer systems: Mathematical foundations*, MITRE Corporation, Bedford, MA, USA, 1973.
- [15] D. McCullough, "Specifications for multi-level security and a hook-up," in *Proceedings of the IEEE Symposium on Security and Privacy*, p. 161, 1987.
- [16] N. Zhang, M. Li, W. Lou, and Y. T. Hou, "MUSHI: toward multiple level security cloud with strong hardware level isolation," in *Proceedings of the 2012 IEEE Military Communications Conference, MILCOM 2012*, pp. 1–6, Orlando, FL, USA, November 2012.
- [17] L. Freitas and P. Watson, "Formalizing workflows partitioning over federated clouds: multi-level security and costs," *International Journal of Computer Mathematics*, vol. 91, no. 5, pp. 881–906, 2014.
- [18] X. Haiwei, Z. Yunliang, G. Zhien, and D. Yiqi, "A multilevel security model for private cloud," *Chinese Journal of Electronics*, vol. 23, no. 2, pp. 232–235, 2014.
- [19] Z. Tan, D. Liu, X. Zhuo, Y. Dai, and L. T. Yang, "Implementation and performance analysis of multilevel security system in pervasive computing environment," *The Journal of Supercomputing*, vol. 66, no. 3, pp. 1243–1259, 2013.
- [20] M. R. Heckman, R. R. Schell, and E. E. Reed, "A multi-level secure file sharing server and its application to a multi-level secure cloud," in *Proceedings of the 34th Annual IEEE Military Communications Conference, MILCOM 2015*, pp. 1224–1229, Tampa, FL, USA, October 2015.
- [21] C. Wright, C. Cowan, J. Morris, S. Smalley, and G. Kroah-Hartman, "Linux security modules: General security support for the Linux Kernel," in *Proceedings of the Foundations of Intrusion Tolerant Systems, OASIS 2003*, pp. 213–226.
- [22] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: a comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.
- [23] R. Jain and S. Paul, "Network virtualization and software defined networking for cloud computing: a survey," *IEEE Communications Magazine*, vol. 51, no. 11, pp. 24–31, 2013.
- [24] A. Gember, R. Grandl, J. Khalid, and A. Akella, "Design and implementation of a framework for software-defined middlebox networking," in *Proceedings of the ACM SIGCOMM Computer Communication Review*, vol. 43, pp. 467–468, China, August 2013.
- [25] "UnixBench," <https://github.com/kdlucas/byte-unixbench>.