Application of RBAC Model in System Kernel

Guan Keqing*1, Li Hongxin1, Kong Xianli2
1School of Management Science and Engineering, Dongbei University of Finance and Economics
No. 217 Jianshan Street, Dalian City, China, 116025
2School of Mathematics and Quantitative Economics, Dongbei University of Finance and Economics
No. 217 Jianshan Street, Dalian City, China, 116025
*corresponding author, Ph: +8613898617806, e-mail:keqinguan@163.com

Abstract
In the process of development of some technologies about Ubiquitous computing, the application of embedded intelligent devices is booming. Meanwhile, information security will face more serious threats than before. To improve the security of information terminal’s operation system, this paper analyzed the threats to system’s information security which comes from the abnormal operation by processes, and applied RBAC model into the safety management mechanism of operation system’s kernel. We built an access control model of system’s process, and proposed an implement framework. And the methods of implementation of the model for operation systems were illustrated.

Keywords: access control, RBAC, operation system kernel, Information security

1. Introduction
With the development of some technologies about Ubiquitous computing, various small, embedded intelligent devices become ubiquitous service terminals for individuals. And the security of the information terminal’s operating system attracted widespread attentions. The application of mainstream operating systems such as Windows, Linux has been gradually extended from the areas of Personal Computer to the Pocket PCs, PDAs, smart phones and other mobile terminals. At the same time, with the popularity of the technologies about mobile electronic payment and online payment, PCs, smart phones and other mobile terminals’ operating systems will become important targets of Trojan and virus programs driven by economic interests. To improve the security of information terminal’s operation system, this paper proposed a novel access control model based on the idea of Role Based Access Control (RBAC) at the kernel level.

The idea of RBAC was proposed by Ferraiolo et al. of NIST in 1990s [1]. After that RBAC research institution has been setup by NIST to study the theory of RBAC and make it into application. In 1996 Sandhu et al. built a series of reference RBAC models called RBAC96 to perfect the idea of RBAC [2]. The important concepts such as sessions, constraints were introduced into the models. RBAC model can make a better balance between system’s security and privilege control’s flexibility than traditional access control models. And wider attentions are attracted to RBAC model at the level of theory and applications. In 1999 J. Linn discussed how to realize RBAC mechanism by using attribute certifications in distribute environment [3]. In 2005, M. Shehab et al. introduced RBAC model into multi-domain environment, proposed a distributed secure interoperability protocol to solve the problem of secure interoperability in multi-domain environment by using role mapping technology [4]. In 2007 Ma Yong and Qing Sihan proposed a RBAC Based E-commerce Solution for Anonymity and Accountability [5]. They encapsulate the trusted third party’s function rationally based on the idea of RBAC to realize the anonymity and accountability of the protocol.

This paper analyzed the key factor threaten to the information security in the level of system kernel, and built a system kernel based RBAC model. The implementation method was proposed, and discussed how to realize the model in varied kinds of operating system in the information terminals.
2. The security enhanced model of system kernel based on RBAC

2.1. The threaten to system’s information security from virus and Trojan

Computer viruses and Trojans always access the important data in the system by concealed operations. To analyze the realization mechanism, the concepts are defined as follows:

Definition 1 (System Process) : The process created in the system kernel, realizes the anticipated function by accessing specific resources.

Definition 2 (Trusted Object) : The set of system resource whose access permissions are assigned to the system process which can access it normally.

Definition 3 (Suspected Object) : For specific system process, it is the set of all system resource outside the trusted objects.

Based on the above definition, the characteristics of computer viruses and Trojans’ behaviour in the operating system can be described as following. It is illustrated in Figure 1.

1. In visual execution environment, the system processes access trusted object by normal operations, and produce normal visualized operation result. For example, when a user accesses the Internet through an information terminal, a web page is shown in the web browser normally.

2. In non-visual execution environment, the system processes access suspected object by abnormal operations, and any visualized operation result will not be produced. Therefore, the user or the system manager can not perceive it. For example, when a user is browsing the web pages, a Trojan program reads the user’s data files in the background and uploads it to the internet.

![Figure 1. Abnormal operation by system process](image1)

![Figure 2. Kernel-RBAC model](image2)

To guard against the information security threats from Trojans and viruses, we will build the access control mechanism to avoid the system processes’ abnormal operations to the suspected objects.

2.2. Kernel-RBAC model

Considering Trojans and viruses can read user’s key data bypassing the protect mechanism of application program level, we build the security mechanism from the system kernel level to improve the security of the system. We introduced the idea of RBAC to build the access control model for system process. We called the model kernel-RBAC. It is shown in Figure 2.

2.2.1. Basic elements of the model

- System process: the processes which are managed by system kernel, including the processes created by system and the processes created by system users.

- Role: conceptualization of a group of permissions assigned to the system processes to finish specific tasks. Roles can be defined from two aspects. One is from the aspect of system management. For example, the roles can be defined as system processes or application program processes. The other is from the aspect of the user of the processes. For example, the roles can be defined as system administrators or normal users, etc.
- Permission: to allow the system processes of specific roles to do some specific operation to specific resources. Permissions are assigned to roles. System processes obtain permissions by activate roles.
- Operation: the access to data including reading, writing, creating, deleting, etc.
- Trusted object: the system resources which can be accessed by the system process of specific roles.
- Session: the interaction between user and system which manage the system process dynamically. When a system process is created, system will map the process to a session automatically, and activate the corresponding roles.

Roles are assigned to system process and permissions are assigned to roles. Both system process-role assignment and role-permission assignment can be many-to-many relationship. Thus one system process can act as multiple roles, and one role can be assigned to multiple system processes. Similarly, a single permission can be assigned to multiple roles and a single role can have multiple permissions. If a system user wants to access the resource of the system, he must establish a session to the system. And the system process is created in session to execute the operations.

2.2.2. access control rules

The access control rule is the key factor to data security. The system uses the access control rules to judge the validity of system process’s access to the data. And only the valid access is permitted.

Let $S_{op}$ is the set of system process’s trusted operation,

$$S_{op} = \{ \text{all permissions of the activated role of the system process in current session} \}$$

Let $\text{op}_{current}$ is the current operation of system process. If $\text{op}_{current} \in S_{op}$, then the operation is regard as legal operation. Otherwise, it is regard as illegal operation which would be prevented. For instance, we talk about the access control mechanism of data files. Let $\text{DIR}_{\text{admin}}$ is the set of files and routes for system administrator. And $\text{DIR}_{\text{user}}$ is the set of files and routes for normal users. Suppose a system process was created in the session of a normal user, i.e. the role of the system process is normal user; the attempt of the process to access the file in the routes belonging to $\text{DIR}_{\text{admin}}$ is regard as illegal operation. If the process attempts to access the file in the routes belonging to $\text{DIR}_{\text{user}}$, it is regard as legal operation.

3. The implement of Kernel-RBAC model
3.1. The implement scheme of Kernel-RBAC model

The implement of Kernel-RBAC model in operating system kernel is to build a Kernel-RBAC module in the original system kernel. The access request of the process is delivered by the Kernel-RBAC module to the API (application interface) of system kernel. The access control mechanism works during the delivery. The implement scheme is illustrated in Figure 3. The relative description is shown as follows:

1. Only processes are concerned; threads created in processes and information interaction between processes are not considered temporarily.
2. When a process is created, the key information of the process, such as name, id value, is recorded in the Context manage module.
3. When a process attempts to access the system resource (data information), the access request will be captured by the access control module in Kernel-RBAC. The AEF (Access-control Enforcement Function) module will send decision request to ADF (Access-control Decision Function) module. ADF module will obtain decision result through the information of access control context information in context manage module and sends the decision result to AEF module. The access control cache is updated when needed to improve efficiency.
4. AEF will decide whether the access request is allowed or not according to the access control decision result from ADF module. If it is allowed, the system kernel API will be called to finish the operation, otherwise it will return with an error code.

3.2. Data structure for context management module

The context management module illustrated in Figure 3 mainly involves the management information such as process information, role information, operation information,
resource information. The data structure of the information is illustrated in Figure 4 which is discussed in the following paragraphs:

1. Process information: the system process list is created during the system’s initialization to support the Kernel-RBAC model. Each data item in the list is a pointer to the address of the struct which is used to manage the process information. The information in the struct includes process name, process id, available roles, and activated roles.

2. Role information: the role list is created during the system’s initialization to manage the role information. Each pointer of data item in role list points to role management data struct. The information in the struct includes role’s id, role’s name, available permissions, etc.

3. Operation information: each pointer of data item in system operation list points to operation management data struct. The information in the struct includes operation ID, operation name, function address, etc.

4. Resource information: each pointer of data item in system resource list points to resource management data struct. The information in the struct includes resource ID, resource type, and resource address, address range, etc.

![Diagram of Kernel-RBAC model](image1)

**Figure 3.** The implement scheme of Kernel-RBAC model

![Diagram of data structure](image2)

**Figure 4.** Data structure for context management module

### 3.3. Performance influence test of the model

Based on the ideas above, we build the test platform upon Ubuntu 10.04 with the kernel version is Linux 2.6.32. The hardware environment is the PC with 1.73G Hz's CPU frequency and 2GB of memory.
The tests are repeated ten times in four cases aiming at 1K bytes of data to compare the average time cost as illustration in Table 1.

The column R0 is the time cost of reading the data directly and printing it in the terminal when Kernel-RBAC access control function is disabled. The column R1 is the time cost of reading the data by kernel API function call transferred by Kernel-RBAC when the access control function is enable. The column W0 and column W1 represent the time cost of writing the data with the access control function is disabled and enabled respectively.

According to the results, the Kernel-RBAC model’s influence to system’s performance is within the acceptable range.

<table>
<thead>
<tr>
<th>NO.</th>
<th>R0</th>
<th>R1</th>
<th>W0</th>
<th>W1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>136</td>
<td>136</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>135</td>
<td>140</td>
<td>49</td>
<td>49</td>
</tr>
<tr>
<td>3</td>
<td>141</td>
<td>136</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>136</td>
<td>139</td>
<td>50</td>
<td>51</td>
</tr>
<tr>
<td>5</td>
<td>134</td>
<td>137</td>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>6</td>
<td>142</td>
<td>140</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>7</td>
<td>136</td>
<td>135</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>8</td>
<td>140</td>
<td>141</td>
<td>49</td>
<td>51</td>
</tr>
<tr>
<td>9</td>
<td>135</td>
<td>137</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>10</td>
<td>137</td>
<td>136</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>average</td>
<td>137.2</td>
<td>137.7</td>
<td>49.7</td>
<td>50.1</td>
</tr>
</tbody>
</table>

3.4. The implementation of the model in different systems

Different operating systems are implemented in different mechanism. Different implement method for different operating system’s features should be considered to inject the Kernel-RBAC model into the operating system’s kernel.

At present the operating systems of Personal Computer and embedded mobile devices can be divided into two main categories, they are Open source operating systems such as linux and non-open source operating systems such as Windows. We will introduce respectively the implement methods of Kernel-RBAC in the two categories of operating systems.

3.4.1. The implementation based on kernel’s source code

For open source operation systems such as linux, we can modify the system kernel’s source code to implement the Kernel-RBAC model.

First, we build a Kernel Information Monitor (KIM) module to manage the information for access control context dynamically. The KIM module will be start up during the initialization of the system. Second, we build a System Call Controller (SCC) module to realize the access control function. The key function of SCC is to control the system call. The linux’s system kernel manage the system call by the special interrupt 0x80 and the array system_call_table[] to index the system functions [6]. system_call_table[] should be modified to make the system call manage functions in SCC instead the original system functions. SCC will decide whether make a function call according to the access control’s decision result.

3.4.2. The implementation based on system service hooking

The method of modifying the source code can’t be used to non-open source operation systems. Take Microsoft Windows for instance, the KIM module and the SCC module need to be realized. The method to realize KIM is similar to which proposed above, the SCC module should be implemented based on the System Service Dispatch Table (SSDT) provided by windows [7]. When the system call is executed in windows, the system service dispatch function will search the SSDT, and call the specific system service function according to the index value and function’s address provided by the SSDT. When the SCC module is built in the system, the SSDT should be modified and reloaded to ensure the SCC can capture the system call.

Above all, no matter what type of operation systems are regarded as target system to implement Kernel-RBAC model, the key factor is capture and control of system calls.
4. Conclusion

In this paper, we have built an access control model which we called Kernel-RBAC to strengthen the access control mechanism for system's process in the level of system kernel in information terminals. The model's implement scheme is proposed, and we have talked about how to inject the model into different operating systems. The application of the model can support not only the personal computers, but also the embedded mobile devices.

Therefore, in the situation of the rising popularity of embedded intelligent devices and other technology about ubiquitous computing, the Kernel-RBAC model can be regard as a method to improve the security of information terminal's operating system so that more reliable information security can be provided to individuals and enterprises.

References