Object Oriented Secure Modeling using SELinux Trusted Operating System

Dr. Nitish Pathak*
1Bharati Vidyapeeth's Institute of Computers Applications and Management (BVICAM),
Guru Gobind Singh Indraprastha University (GGSIPU), New Delhi, India
Email: nitishforyou@gmail.com

Neelam Sharma
2 MAIT, Guru Gobind Singh Indraprastha University (GGSIPU), New Delhi, India

This research paper proposes the object oriented experimental setup for secure web application development and use of security performance flexibility model to keep high security in web applications. This model allows system administrators to skip or disable some unnecessary security checks in distributed trusted operating systems through which they can effectively balance their performance needs without compromising the security of the system. For example, system admin can tell that video on demand server is allowed to skip only security checks on reading files, while database server is allowed to skip only security checks on seeking files. Which operation is need to be skipped and which operation is not need to be skipped is very much subjective in nature, this will depend upon the user’s requirement and the particular application’s requirement. The selection of these operations and system calls for a particular application is the part of software requirement elicitation process. This UML 2.0 based research work proposes object-oriented class-based software development, source code generation in C++ and the integration of security engineering into a model-driven software development.

Keywords: SELinux, SPF, Forward engineering, DTOS, UML 2.0

Date of Submission: Jan 03, 2018                                             Date of Acceptance: Jan 23, 2018

I. INTRODUCTION

In the last decade, there has been vast growth in the field of networking, sharing of data worldwide. And then comes the most extensively used thing Internet have made cyber security a very crucial aspect of research and development. It's matter of concern for both the common users and researchers connected all over the world. Despite of lot of works undergoing we are still unable to get something that reliable and silver bullet that it may provide us with complete security for our systems. Being so advanced we still lack the basic potential to create such a system that is capable of stopping viruses and accessing our confidential data from our systems [1]. The security methods developed, researched till yet are implemented in the application layer of the computers which is making our systems more prone to data insecurity. These methods include encryption using a key i.e. cryptography, using firewalls, access control using authentication, and application layer access control. The most two burning domains are Cryptography and authentication techniques in which max research is being done. Although these are something very difficult to crack but no one knows the dynamic minds making some of probability of data insecurity [2]. To some extent using firewalls and application layer access control have helped us but they do have a drawback. These two techniques can help in stopping the attacks using viruses uploaded on internet but fails to protect from internal security issue thus finally making our system vulnerable [3].

The biggest threat to our application layer is viruses and Trojan Horses. Once these two enters in our system they have the potential to access and even modify each and every data present on the system. Now these days, to overcome the threats operating system application layer and the network entry points is used to implement the security measures. Although no preventive measures are used inside the kernel of Operating Systems. It is believed that security measures in kernel are much more effective than the application layer [4]. In fact, after lot of research such operating systems have been developed which have much more mechanisms inside the OS kernel providing us very good level of security thus securing our systems [5].

In reality, trusted operating systems are better choice for web applications to maintain the security concern, but this security will come at a cost. By using trusted systems, our web application will be more and more secure, but due to more security checks, the performance of the same system will disgrace in all respect [6].

The Security is not something expected not only by big organizations but also by common consumers so now concerns are being there on this and many vendors are trying hard to fix the issue. The companies which came up with some promising operating systems with security features are Argus-Systems Group, HP, and Sun Microsystems [7]. Open Source OS are also providing well secured kernel having excellent security features and commonly known as secure systems, operating systems. National Security Agency has released the most secured and that too open source operating system called as SELinux. Proper definitions of secure system vary from organization to organization [8]. These secure Systems are more complex for computer administrators to handle and manage. Such secure Systems require much more extra
effort and time to setup the desired security policy on the part of administrator [9]. The implementation of security policies, as per the requirement of user, is very complex in such systems. In this research paper, we are suggesting SELinux trusted operating systems for maintaining the security concern in web applications [10][11]. In the proposed design and development, we are using UML2.0 based conceptual modeling for the development of secure application. It makes the programming simpler, more effective and manageable [12]. This research work proposes SPF based secure software analysis, SPF based secure software design and SPF based secure software development. Far above the ground, quality of software design is necessary for the success of software [13].

II. TRUSTED OPERATING SYSTEM BASED SECURE WEB APPLICATIONS

The essential structural design of this operating system is shown in Fig. 1(a). Just for a reminder to the readers; architecture is just a concept although implementation can be done in a lot of ways. The architecture of traditional operating systems is given in Fig. 1(a). System call interface helps the application and middleware interface to communicate with the Operating System [14].

![Diagram](image)

Fig. 1: Structure of trusted operating systems and ordinary operating systems.

Fig. 1.(a), Illustrating thin or slim security layer of operating systems kernel security checks. Now in order to provide higher security, lots of security checks are there in kernel of Trusted Operating Systems. Fig. 1(b) demonstrates the additional security checks in the kernel. This will cause trusted operating systems to be slower than standard operating systems [15]. Fig 1(b) clearly depicts the thicker layer of kernel security checks. What all security measures are being taken in the kernel security check depends all on implementation and modeling. But the disadvantage of having extra security check is that whenever user tries to do any useful work it need to undergo all the checks thus deteriorating the system overall performance[16][17].

III. PROBLEM EXPLANATION AND OBJECT ORIENTED SOLUTION METHODOLOGY

Before moving into the problem, we are dealing, first we will be talking about the basic principles of Secure Operating Systems. As was mentioned previously, the term Trusted OS is interpreted differently and vary from one company to another software company. During system programming, company develops the system software according to the requirement of end users. But there are some important features in all Trusted Operating Systems. They are as follows Least Privilege, Mandatory Access Control (MAC), Discretionary Access Control (DAC) and auditing [18][21]. For example, admin of the system can disable all the read checks in web server because they are actually useless which finally increases throughput of the web server. Web server deals with sole public data and public information. Since majority of data is public on any web server, task of checking it during read from disk is something useless because this data is already readable by each and every user using internet [19]. The real task of security comes when it comes to writing access. For any web server integrity is the main issue rather than its confidentiality. As we had stated that there are many types of workloads that are continuously being checked by the security mechanisms of kernel in which many of them are very much useless or undesired in a Trusted Operating System. Primary concern of these workloads is quality as well as integrity of data rather than security of certain operations since the data they consist can be public [20]. This let us conclude that by disabling security measures of some parts of OS performance can be increased.

The essential structural design of distributed trusted operating systems (DTOS) and Flask is revealed in Figure 2. Unlike the conventional methods of adding more and more security layers at the kernel level, we are suggesting two supplementary or extra subsystems in this structural design. The responsibility of object manager in this model is to call the security server each time whenever a system user tries to access an object. Particular security server confirms the security pattern and informs the respective object manager if permission for requested operation is granted or denied. In this design approach we have to notice that the security server is not at all the component of the kernel. It is a different and separate part that can be called as per requirement, by the kernel. These different modules of the security server also can be altered or changed. This is the main reason that’s why DTOS as well...
The implementations of web application for maintain the security is very much subjective in nature. The security requirement for the same will depend upon the user’s needs. The object manager all the time calls the security server for checking the granted permissions. If security server grants permission for particular operations, then ok, operation or specific system call will be passed to kernel layer for execution. If permission is not granted, requested operations will not execute.

The security features and security policies can be altered, as per need, dynamically as the system is in execution phase. Security layer will execute the security checks according to new altered security policies. In order to boost performance in DTOS as well as in Flask, security policy caching was recommended as a means to strengthen performance. The mainly referenced security checks are stored in a software-implemented cache, located in the module of object manager. This can surely increase the performance of systems. By caching recent security policies in the object manager, few parts of the security check can be ignored. In this case, if security policies changed by authentic user, new security policies will be implemented with immediate effects. As stated all over this section, it seems the future of security in operating systems is dependent on two major factors. These two factors or themes are the development of TOS with very easier system administration work and system programming methods to apply a massive variety of security policies. At the time of implementation, these massive varieties of security policies will vary from one application to another.

IV. OBJECT ORIENTED SECURE MODELING FOR WEB APPLICATION

Object Oriented is an ever-growing discipline and since there lies, uncertainty for what really establishes sense of object-orientation; this encapsulates the nucleus factor and defines the vocabulary to be used in the remaining of theory. Object-Oriented Development is a set of practices and approaches for manufacturing software systems grounded on real world abstractions. Object-Oriented approach delivers a relatively smooth approach for progressing from analysis, to design, through to implementation. Object oriented development is presently the most accepted software development style. UML has now
become the language of choice intended for developers who wish to imagine and model the system underneath progress. UML is used in many customs for expressing the concepts such as software specification, website structure and business modeling. In the proposed design and development, we are using UML2.0 based conceptual modeling for the development of secure application.

In object oriented class diagram, designer will identify the classes. These classes can be identified through software requirement specification (SRS). As normal practices, actors of use case diagram are considered as classes and the use cases are considered as member functions or methods of the classes. When we want to model the structure of a system or a web application, we can make use of object oriented class diagram. We develop the component based class diagram. In Store Stock Control Based Web Application, storing objects may be sales clerk, inventory, Credit card, Cheque, store manager, Payment, Person, marketing, stock manager, person, warehouse person, invoice, system, customer etc. (See Fig. 3.). The standard class diagram of Store Stock Control is as follows in Fig. 3.

![Class diagram for online store stock control web application.](image)

The class wise equivalent C++ code of this case study is as follows-

```cpp
class Person
{
    public:
        //##ModelId=553C976E01C4
        GetDetails();
        //##ModelId=553C97750038
        SetDetails();

    private:
        //##ModelId=553C971B02D3
        String Name;
        //##ModelId=553C97380226
        String Phone;
        //##ModelId=553C975B0001
        String Address;

    #include "Person.h"
};
```
V. RESULTS AND DISCUSSION OF SELINUX TRUSTED OPERATING SYSTEM

SELinux trusted operating system is better choice for the development of secure web applications. Table 1 and Table 2 shows the performance results that are appropriate to security performance flexibility (SPF). These tables showing the results with SPF, without SPF and showing the performance compression. Through SPF model of SELinux operating system, we can implement the high security concerns for web applications.

<table>
<thead>
<tr>
<th>File System Tests</th>
<th>SELinux without SPF</th>
<th>SELinux with System-SPF model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Disk Reads (K) per second</td>
<td>94167</td>
<td>93135</td>
</tr>
<tr>
<td>Random Disk Writes (K) per second</td>
<td>79188</td>
<td>79508</td>
</tr>
<tr>
<td>Sequential Disk Reads (K) per second</td>
<td>335527</td>
<td>325591</td>
</tr>
<tr>
<td>Sequential Disk Writes (K) per second</td>
<td>149616</td>
<td>153174</td>
</tr>
<tr>
<td>Disk Copies (K) per second</td>
<td>102252</td>
<td>102744</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>File System Tests</th>
<th>SELinux Without SPF</th>
<th>SELinux with System-SPF model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Disk Reads (K) per second</td>
<td>94167</td>
<td>99762</td>
</tr>
<tr>
<td>Random Disk Writes (K) per second</td>
<td>79188</td>
<td>84768</td>
</tr>
<tr>
<td>Sequential Disk Reads (K) per second</td>
<td>335527</td>
<td>363571</td>
</tr>
<tr>
<td>Sequential Disk Writes (K) per second</td>
<td>149616</td>
<td>159727</td>
</tr>
<tr>
<td>Disk Copies (K) per second</td>
<td>102252</td>
<td>110315</td>
</tr>
</tbody>
</table>

TABLE 1: SECURITY CHECKS EXECUTED IN SELINUX TRUSTED OPERATING SYSTEM.

TABLE 2: SECURITY CHECKS SKIPPED IN SELINUX TRUSTED OPERATING SYSTEM.

With the help of above software development process, developers can identify software Metrics like no. of data members, no. of data members per super class, no. of data members per sub class, member functions, the length of the program, Volume, vocabulary of a program, average number of live variables, Count of executable statements, member functions per class, data structure metrics, and information flow etc. These software Metrics can be identified through Hallstead software science measures and data structure metrics. With the help of above mentioned approach, software project planning will become easier to developers.
VI. CONCLUSION
This research paper presents a SPF based approach for web applications and the integration of security engineering into a model-driven software development. This research work showcase the effectiveness of UML 2.0 based object oriented modeling with primary focus of security through the system level SPF in web applications. This model allowed system administrators to skip or disable some unnecessary security checks in distributed trusted operating systems through which they can effectively balance their performance needs without compromising the security of the system. In this paper, we described experiential forward engineering, source code structuring and restructuring of secure software system.

REFERENCES