Flexible Security Policy Architecture for Trusted Operating Systems

Dr. Nitish Pathak
Assistant Professor, Bharati Vidyapeeth's Institute of Computer Applications and Management (BVICAM), Guru Gobind Singh Indraprastha University (GGSIPU), New Delhi, India

Abstract- This research paper propose the security policy architecture for Trusted Operating Systems use of security performance flexibility model to balance performance and to keep high security for trusted operating systems. This model allows system administrators to skip or disable some unnecessary security checks in trusted operating systems which they can effectively balance their performance needs without compromising the security of the system. 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 for a particular application is the part of software requirement elicitation process. This paper discusses the Design aspect of Security in Trusted Operating Systems.

Keywords- Mandatory Access Control, SPF, Hewlett-Packard's Virtual Vault, DTOS, security policy

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. Its 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 includes 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 [4]. It is believed that security measures in kernel are much more effective than the application layer.

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]. A shocking stuff to know is that the technologies developed for OS kernel are not new and were developed very long before around in 1970-80. The cost and the complexity led to its failure of reaching the wholesome consumers and are widely used only in organizations which need high security and can’t compromise with it [6]. In reality, trusted operating systems are better choice for web applications to maintain the security concern, but this security will come at a cost.

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 [9]. National Security Agency has released the most secured and that too open source operating system called as SE Linux. Proper definitions of secure system vary from organization to organization.
These secure Systems are more complex for computer administrators to handle and manage [8]. Such secure Systems require much more extra effort and time to setup the desired security policy on the part of administrator. 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 [9].

II. DESIGN ASPECT OF SECURITY IN TRUSTED OPERATING SYSTEMS

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 [10].

In this section, we will be discussing about some history of Trusted Operating Systems. We have been researching for better security methods since 1984s. We were not sure about the appropriate requirements what operating system must have. In this regard, the trusted System judgment Criteria was defined by US Department of Defense in 1983. After these basic guidelines were set and operating system were programmed and evaluated accordingly. Ratings or strength is known to be of 4 classes A, B, C and D respectively and A with max security and D the least one. Class D has the least security measures build into OS [11][12].

Most common examples of class B operating systems are Hewlett-Packard’s Virtual Vault, SELinux, Argus System’s Pitbull and Trusted Solaris. Sun Microsystems developed trusted solaris as much more secure system to maintain the security aspects in applications. Class A operating systems are one which are widely used in military domains and provides tight security features necessary to secure confidential information. Commercial vendors do not provide such operating system in the market for common use [13].

The best examples around us of DAC are UNIX and LINUX OS. Every file this OS is provided with set bits that tell us read, write and execute permissions. Each user has the rights to change the permissions of each file.

Administrators have rights to grant permissions to even themselves, another user and anyone in the group. MAC can be most excellent explained as security mechanism that is not in the hand of the user rather it is controlled totally by the operating system and the guidelines set by the system administrator [18]. Thus, MAC controls the flow of data much better than DAC. The reason for better security can be best explained using their implementation and modeling. MAC is the anchor, providing us the desired security in trusted operating systems [14].

Security labels are used to implement diverse MAC code which ultimately confines and deals with many of the security issues. Despite of weakness in DAC, still it is irreplaceable and both DAC and MAC are implemented together in a system allowing some extent of security discretion for computer users. MAC main task is to limit how the user will modify or change the security permissions residing in the system. In addition to this, permissions with the system administrators are also taken into consideration by MAC security guidelines [15].

It is very difficult to decide the minimum permissions one operating system must provide to any user or process. However, giving access more than required generally leads to security threats. There are many techniques least privilege can be implemented in our operating system.

The permissions in computer system tell us that what all permissions are given to the process and what all actions are allowed to take during executions. In the computer system, each and every process holds list of permissions and capabilities [16]. These capabilities store the list of actions or permissions which a process will be allowed for the period of its lifetime span. With the above security technique, we can make our system well secure using least privilege.

Anyways a system with a user with all permission’s can be very dangerous. It makes the attacker do whatever he wishes to do because he has access to the whole system and even the root account. Least privilege along with DAC and MAC helps lot in making the system more secure.

Another significant aspect of security is Auditing in Trusted Operating Systems. It actually records all the security operations and transactions in a system. Logging information actually helps us in getting the source of attack or does the system can expect any other attack in future. Most systems available in the market are programmed in such a way to store basic logging information but fail to do the high grained logging feature which is mainly required in system with high security. Systems currently available cannot store data about the user that which file he is accessing which is very credential data in retrieving security holes in our programs [17].
III. SECURITY IN TRUSTED OPERATING SYSTEMS

The essential structural design of this operating system is shown in Figure 1. 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 Figure 1(a). System call interface helps the application and middleware interface to communicate with the Operating System.

![Diagram](https://via.placeholder.com/150)

(a) Ordinary operating systems (b) Proposed structure of trusted operating systems

**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.

Figure 1(b) demonstrates the additional security checks in the kernel. This will cause trusted operating systems to be slower than standard operating systems. Figure 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. 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.

IV. FLEXIBLE SECURITY POLICY MODEL FOR TRUSTED OPERATING SYSTEMS

There is a negligible amount of research that can actually be undertaken to SPF model. Still, the SPF frame is related to innovative ideas from three distinct areas of research. The first area of research is related to security. The other two research areas are flexibility of operating system and Quality of Service in systems. A lot of quality research work has been researched for quite long time in Operating system flexibility.

There are two major dimensions in operating system security that should be pointed out. The ideas are:

1) *Making the system administration work easier to maintain web security.*

As it has been revealed previously, system administration work of secure TOS is certainly far additional difficult as compare to that of conventional operating systems. The system administration work should be easier to maintain web security.

2) *Adding security features to facilitate additional security policies to be imposed in web applications.*

In conventional Systems, security and data protection policies are quite inadequate and incomplete to those that can actually fit the security model implemented in the system’s kernel. The structure of these OS is little bit different from traditional OS, so, the programming efforts and implementations of such systems are also different. Very good system programming is required on part of software developers. Without good system programming skills, the implementation of such systems for web applications will not be possible.

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. This let us conclude that by disabling security measures of some parts of OS performance can be increased. This research paper proposes a concept of SPF, in order to gain improved performance and speed for particular system workloads for Trusted Operating Systems.

Although the latest research work explained previously implement so many security policies at kernel level as well as application layer level to protect the systems and web application, but after this these efforts towards security are still relatively incomplete, imperfect and limited. The implementation of security policy at kernel level is still very limited. In order to suggest a much needed security, operating system structural design must be developed to grasp variable security policies.
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 as Flask both are built and implemented upon kernels.

Almost security policies can be implemented if we consider security server as a separate part or separate module.

These separate modules can be easily modified as per web applications requirement. This is not hard and fast that security need always will be unchanging or statically placed within the kernel. These security needs and implementation will vary from one real life application to another.

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.

![Fig. 2: Flexible security policy architecture for trusted operating systems](Image)
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.

V. CONCLUSION

This research paper presents a SPF based approach for SELinux Trusted operating system. Flexible security policy architecture for Trusted Operating Systems. This model allowed system administrators to skip or disable some unnecessary security checks in trusted operating systems through which they can effectively balance their performance needs without compromising the security of the system. 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.

REFERENCES