COMPUSOFT, An international journal of advanced computer technology, 4 (4), April-2015 (Volume-IV, Issue-IV)

ISSN:2320-0790

An International Journal of Advanced Computer Technology

AGENT BASED PUBLIC AUDIT FOR SECURED CLOUD STORAGE

Vinod Kumar Pal, Rahul Roy, G.MICHAEL

Asst. Prof.
Bharath University Chennai-600073

Abstract: Operating system kernels typically enforce lowest restrictions on the applications permissible to execute leading to the power of malicious programs to abuse system resources. Malware running as standalone processes will freely execute enjoying the privileges provided to the user account running the method. Main stream software package kernels lack a robust and reliable mechanism for distinctive the running processes and binding them to the corresponding possible applications. Method authentication is completely different from method identification. Our supervisor call instruction observances are often integrated with existing obligatory access management systems to enforce application-level access rights. We tend to address the identification downside by proposing a unique secure application identification model within which user-level applications square measure needed to gift identification proofs at run time to be genuine to the kernel. Our supervisor call instruction observance is often integrated with existing obligatory access management systems to enforce application-level access rights.

Keyword:- cloud computing, kernel,

Introduction

Operating system kernels usually enforce lowest restrictions on the applications allowable to execute leading to the power of malicious programs to abuse system resources. Malware running as standalone processes will freely execute enjoying the privileges provided to the user account running the method. Main stream OS kernels lack a robust and reliable mechanism for characteristic the running processes and binding them to the corresponding viable applications. Method authentication is completely different from method identification. Our supervisor call instruction observations are often integrated with existing obligatory access management systems to enforce application-level access rights. We tend to address the identification drawback by proposing a completely unique secure application identification model during which user-level applications square measure needed to gift identification proofs at run time to be genuine to the kernel. Our supervisor call instruction observation are often integrated with existing obligatory access management systems to enforce application-level access rights

LITERATURE SURVEY

The security of A2 relies on the confidentiality of the application credentials. Thus, we analyze our security guarantees by discussing the confidentiality of the application credentials and the integrity of A2 components. Unforgeability of credentials. Forging existing secret credentials (that appear on the credential list) by attackers is computationally hard, as long as a strong pseudorandom number generator is used to generate the credential. Besides existential forgery, a malware process using a self-generated arbitrary value as its credential cannot successfully pass the authentication because that self generated credential is not registered with the kernel and does not appear on the credential list. Confidentiality of code capsules and credential list. To protect the secret credential from being revealed to other applications, A2 restricts the read access to applications' binaries, namely code capsules (where the application's copy of credential is stored). Malware may attempt to steal a credential from application's or A2 components' memory at runtime, which is prevented by the standard process memory isolation mechanism of the system. Similarly, A2 restricts the access to the credential list (owned by registrar) by other processes, thus ensuring its confidentiality. More specifically, only the registrar and the verifier have the direct access, and the Authenticator can (indirectly) query the list.

NETWORK SECURITY

We take into account the subsequent problems associated with strong network routing in an exceedingly extremely dynamic and dynamical traffic environment: What network routing ought to a web Service supplier use thus on (i) accommodate users exigent "good" service whereas being unpredictable within the traffic that they\'d wish to send to completely different destinations, (ii) minimize the number of "over provisioning" that must be tired the network so as to form "best effort networking better" while not resorting to classy traffic prediction and management mechanisms, (iii) operate the network with efficiency with principally static routing configurations and while not dynamic routing changes to avoid congestion owing to forceful changes in traffic flows between a network's ingress and egress routers.

PROCESS

This module is mainly based on database. We can add the process to database for backup, and also remove the process from the database. List of the process in the database also can show by this module. This module is made for database purpose.

TIME SETTINGS

Here we can set the limit for number of process running per the day. By this, we provide the process that running many times per the day. Also can set the number of (limit the) process that running at same time. By this, protect the run time error. From this module we can set the time limit for process running.

SYSTEM PROCESS

In this module, we are controlling the system. set the time for when we want to turnoff, logoff, restart the system. Also to set the time for hibernate the system. by this we can restart the system automatically. Here we select any one of the above options and set the time for that.

PROCESS CONTROL

In this module we are set the time for running the process. In this we can select the time for instance, after, or set some time. Simultaneously we are set the time for closing the process.

PROCESS AUTHENTICATION

Process authentication is completely different and freelance from method identification and needs stronger properties, for instance, un forge ability and ant replay. In distinction, identification may be thanks to describe a principal. Method IDs and method names are identifiers for processes in an OS setting. Typically, these method identifiers are generated by the system once examining the possible file names and installation ways of processes. This examination of possible file names and installation ways is that the simplest kind of method authentication.

PROBLEM DEFINITION

We implement and evaluate a prototype of our monitoring architecture in Linux as device drivers with no modification of the kernel. The extension provides a language for the specification of application-level access rights. Process authentication based on the installation path is weak. Without secure process authentication, malware may impersonate legitimate applications and abuse system resources, thus violating system assurance. We have demonstrated its feasibility by presenting our architecture, implementation, and evaluation of a prototype Linux system supporting process authentication. We explained how process authentication can isolate malicious processes and, thus, prevent them from abusing and accessing system resources. The authentication model of A2 is highly portable and can be made compatible with legacy applications without any customization.

Our evaluation results indicate that the overhead of performing process authentication at the system call level is acceptable.

EXISTING SYSTEM

Existing obligatory access management systems to enforce application-level access rights. We have a tendency to implement and appraise a epitome of our watching design in UNIX system as device drivers with no modification of the kernel. The extension provides a language for the specification of application-level access rights. Method authentication supported the installation path is weak. While not secure method authentication, malware could impersonate legitimate applications and abuse system resources, so violating system assurance

DISADVANTAGE

- ➔ A problem is how to protect the secrecy of application credentials that is stored by the application.
- ➔ The authentications protocol requires additional operation while processing, so we avoid the modification and customize the existing application.

PROPOSED SYSTEM

In this project, we described the existing MAC-based approaches to application authorization alone are not sufficient for defeating malwares. The kernel must have secure mechanisms for authenticating and identifying processes, beyond the simple and easy-to-forget process ID or process name based identification. Thus the critical problem in detecting malicious activities in the user process is able to ability to strongly identify processes at runtime and bind them to correct application identities. One purpose is to protect the secret application credential from being revealed to unauthorized processes through the file system. The other purpose is to bind a credential with the executable file, which is later used to verify the identities of the running processes by the kernel.

ADVANTAGE

- ➔ Our security goal is to ensure the system assurance.
- ➔ Goals of these pieces of work significantly differ, none of the existing solutions provides a satisfying solution for the application authentication problem as A2 does.

SYSTEM STUDY

FEASIBILITY STUDY:

The feasibleness of the project is analyzed during this section and business proposal is place forth with a really general set up for the project and a few value estimates. Throughout system analysis the feasibleness study of the planned system is to be distributed. this is often to make sure that the planned system isn\'t a burden to the corporate. For feasibleness analysis, some understanding of the key necessities for the system is important.

- ➔ Feasibility may be a sensible extent to that a project is performed with success. to gauge feasibleness, a feasibleness study is performed, that determines whether or not the answer thought-about to accomplish the wants is sensible and feasible within the code or not. 3 key issues concerned within the feasibleness analysis area unit

Technical feasibleness

Operational feasibleness

Economic feasibleness

TECHNICAL FEASIBILITY:

This study is administered to envision the technical practicability, that is, the technical needs of the system. Take into account the financial factors additionally. Since it would happen that developing a specific system is also technically attainable however it\'s going to need immense investments and edges is also less. For evaluating this, economic practicability of the planned system is administered. Any system developed should not have a high demand on the obtainable technical resources. This can result in high demands on the obtainable technical resources. This can result in high demands being placed on the consumer. The developed system should have a modest demand, as solely borderline or null changes area unit needed for implementing this technique.

In technical practicability the subsequent problems area unit taken into thought. Once the technical practicability is established, it\'s necessary to

OPERATIONAL PRACTICABILITY:

The projected system commonly solves the issues and takes blessings of the opportunities known throughout scope definition, it satisfies necessities the wants the necessities} known within the requirements analysis section of system development. Since the applied math figures are keep in an exceedingly bound format within the laptop, it reduces the manual work and enhances the quality of presentation additionally.

Operational practicability assesses the extent to that the specified code performs a series of steps to unravel business issues and user needs. This practicability relies on human resource and involves visualizing whether or not or not the code can operate once it\'s developed, and be operated once it\'s put in. This measures however well your company are going to be ready to solve issues and profit of opportunities that are given throughout the course of the project.

ECONOMIC FEASIBILITY:

This study is dispensed to visualize the economic impact that the system can wear the organization. the number of fund that the corporate will pour into the analysis and development of the system is restricted. The expenditures should be even. so the developed system moreover among the budget and this was achieved as a result of most of the technologies used are freely on the market. Solely the bespoke merchandise had to be purchased.

In economic practicability, price profit analysis is completed within which expected prices and edges are evaluated. Economic analysis is employed for evaluating the effectiveness of the projected system. The developed system is economical in comparison to the present system job done manual system. Therefore the projected system is therefore quick that coming up with is created simply. The Economic practicability is analyzed victimization following studies,

 Cost primarily based Study: within which Development prices and disbursement are alright managed then the advantages derived out of the system.

 Time primarily based Study: The analysis of the time needed to realize a comeback on investments is additionally among the limit.

4. SYSTEM SPECIFICATION

HARDWARE SPECIFICATION:

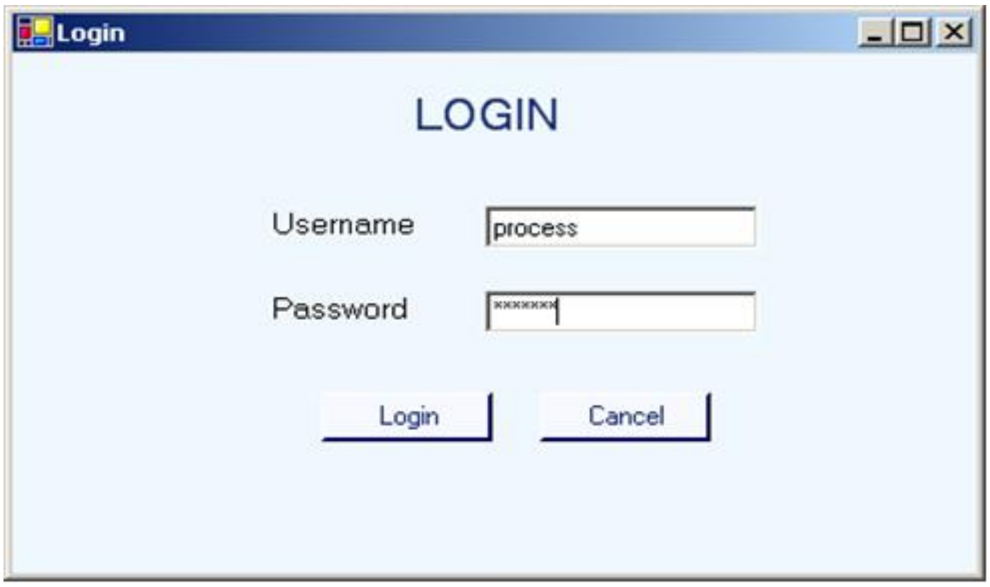
PROCESSOR	: Intel(R) Pentium(R) Dual-Core Processing
RAM	: 1GB RAM
HARD DISK	: 20 GB

SOFTWARE SPECIFICATION

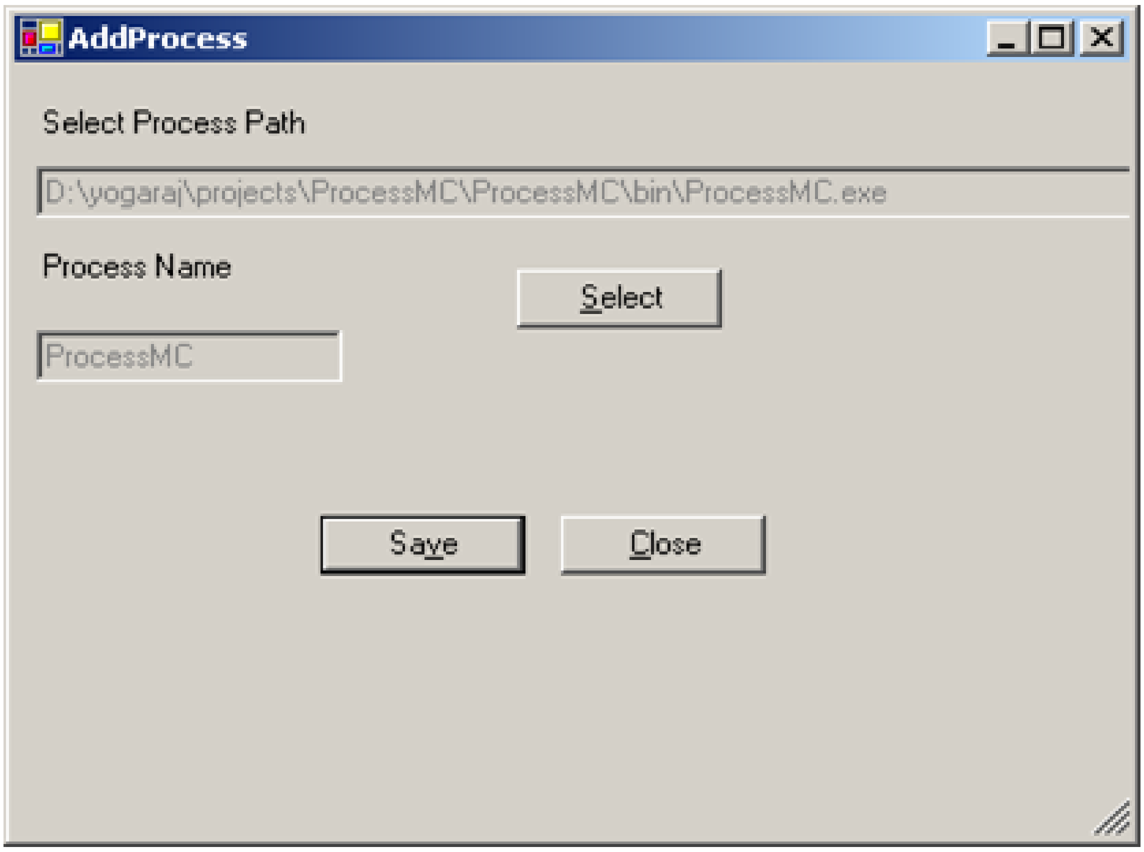
OPERATING SYSTEM	: Windows XP, Windows2007 (32Bit (Original)
ENVIRONMENT	: Visual Studio .NET 2005 or 2008 or 2010
.NET FRAMEWORK	: Version 2.0 or Version 3.0 or Version 4.0
LANGUAGE	: C#.NET
BACK END	: MS-SQL-Server 2000

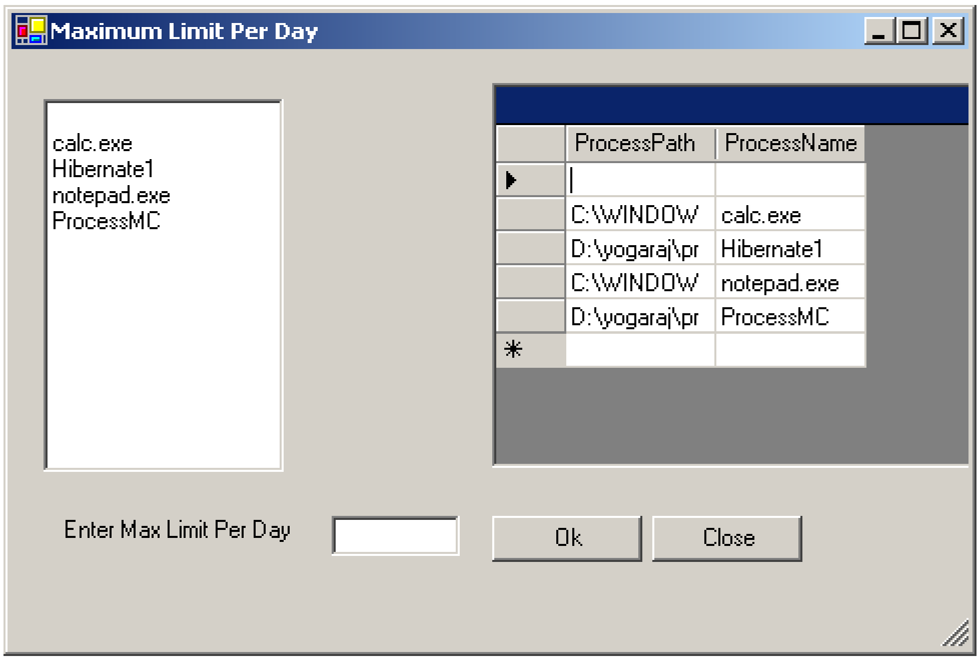
<u>Screen short</u>

Login page:-

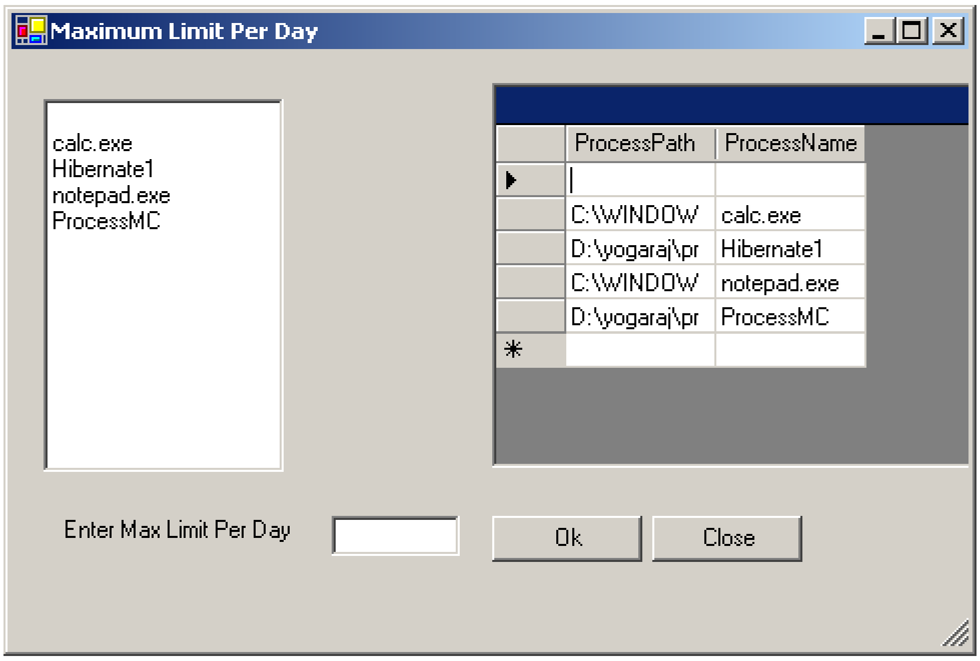


Set path:-





	ProcessPath	ProcessName
▶	|	
	C:\WINDOW	calc.exe
	D:\yogaraj\pr	Hibernate1
	C:\WINDOW	notepad.exe
	D:\yogaraj\pr	ProcessMC
*		



calc.exe
Hibernate1
notepad.exe
ProcessMC

	ProcessPath	ProcessName
▶	|	
	C:\WINDOW	calc.exe
	D:\yogaraj\pr	Hibernate1
	C:\WINDOW	notepad.exe
	D:\yogaraj\pr	ProcessMC
*		

REFERENCES

[1] H.M.J. Almohri, D. Yao, and D. Kafura, “Identifying Native Applications with High Assurance,” Proc. ACM Conf. Data and Application Security and Privacy (CODASPY ’12), Feb. 2012.

[2] P. Loscocco and S. Smalley, “Integrating Flexible Support for Security Policies into the Linux Operating System,” Proc. USENIX Ann. Technical Conf., 2001.

[3] “grsecurity,” http://www.grsecurity.net/, 2013.

[4] Z.M.H. Chen and N. Li, “Analyzing and Comparing the Protection Quality of Security Enhanced Operating Systems,” Proc. 16th Ann. Network and Distributed System Security Symp., 2009.

[5] C. Wright, C. Cowan, S. Smalley, J. Morris, and G. Kroah- Hartman, “Linux Security Module Framework,” Proc. 11th Ottawa Linux Symp., 2002.

[6] K. Xu, H. Xiong, D. Stefan, C. Wu, and D. Yao, “Data-Provenance Verification for Secure Hosts,” IEEE Trans. Dependable and Secure Computing, vol. 9, no. 2, pp. 173-183, Mar./Apr. 2012.

[7] W. Dai, T.P. Parker, H. Jin, and S. Xu, “Enhancing Data Trustworthiness via Assured Digital Signing,” IEEE Trans. Dependable and Secure Computing, vol. 9, no. 6, pp. 838-851, Nov./Dec. 2012.

[8] G. Xu, C. Borcea, and L. Iftode, “Satem: Trusted Service Code Execution across Transactions,” Proc. IEEE 25th Symp. Reliable Distributed Systems (SRDS ’06), pp. 321-336, 2006.

[9] A.M. Fiskiran and R.B. Lee, “Runtime Execution Monitoring (REM) to Detect and Prevent Malicious Code Execution,” Proc. IEEE Int’l Conf. Computer Design: VLSI in Computers and Processors (ICCD ’04), pp. 452-457, 2004.

[10] T. Jaeger and R. Sandhu, Operating System Security. Morgan & Claypool, 2008.