IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON TRUSTED COMPUTING

Received August 7, 2018, accepted September 9, 2018, date of publication October 9, 2018, date of current version October 29, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2870116

Active Defense Technology of Power Monitoring System With Adaptive Features

WEI LIU[1], YIYANG YAO[2], BAOHUA ZHAO[3], WEIYONG YANG[1], LONGYUN QI[1], AND XIAOLIANG LV[1]

[1]NARI Group Corporation (State Grid Electric Power Research Institute), Nanjing 210003, China
[2]State Grid Zhejiang Electric Power Co., Ltd., Hangzhou 310007, China
[3]Global Energy Interconnection Research Institute Co., Ltd., Beijing 100209, China

Corresponding author: Wei Liu (liuwei5@sgepri.sgcc.com.cn)

This work was supported by the 2017 Science and Technology Project of State Grid Corporation: Research on Key Techniques of Trusted Root and Trust Chain Construction in Power Cloud Platform under Grant 5211XT17001G.

ABSTRACT Internet of Things has found a lot of applications in power systems. However, with the increased the sensing, networking, and control capabilities, the security issues have become even more urgent at the same time. In this paper, a TMAC model based on trusted mandatory access control is proposed by studying the security situation of a power monitoring system. The model has self-learning characteristic and can realize the automatic escalation of the global security strategy based on intelligent agent, so as to build the safe immune ability and active defense system for the power monitoring. This paper introduces the key technology of the TMAC security model, formalizes some of the work, and finally tests its credibility and effectiveness for the typical application scenario of the power monitoring system. Through the study of this technology, the power monitoring system is further equipped with an immune ability against virus Trojan and hacker attacks, especially for application scenarios, such as substation, power plant, and master station.

INDEX TERMS Power monitoring system, security immunity, active defense, operating system ontology security.

I. INTRODUCTION

As an important basic industry for the people's livelihood, the power industry has always attached great importance to network security while paying attention to information construction, State Grid Corporation has put information security and power grid stability on the same important position. Leon Parnetta, the former U.S. defense secretary, once said "the next Pearl Harbor incident that we may experience is most likely to be a cyber-attack that may paralyze our power system, the grid, the security system, the financial system, and the government system." It can be said that the national power system has become the primary target of hostile forces attack, but also the first attack target of the national network battle.

It's been recognized that the Internet of Things has been another enabling technology for power systems. With the increased sensing, communication and control capabilities, the power system is also facing even more critical security issues. Typical examples include blackout in Ukraine in 2015, paralysis of US Eastern Internet Services in 2016, worldwide rappelling virus WannaCrypt in 2017. In general, computer network attack under the new situation has the following characteristics: 1) Wider, The development of the global energy Internet makes it possible to connect more power grids, so the impact of the attack is constantly expanding and the consequences are even much worse; 2) Spread faster, attacks evolved from the confrontation of nations to the participation of more non-governmental organizations. The huge benefits of the virus makes the spread of speed, and transmission cannot be controlled; 3) Attack means are more advanced; malicious code is hidden in a legitimate industrial control system upgrade software, and the virus updates quickly. In response to these endless loopholes and attacks, the traditional passive protection system appears difficult to parry.

In view of the increasingly serious security situation in the power system, this paper proposes active defense technology based on operating system security. This paper introduces power monitoring system security and active defense system based on TMAC, and some of the work carried out a formal description. The core of TMAC technology is the mandatory access control and trusted computing, including T, standing for trusted computing and MAC, standing for mandatory access control. Through the research of this

technology, we can achieve a safe immune against Trojan virus and hacker attacks, and reduce vulnerability 0Day risk, so as to construct the active defense system, but do not rely on continuous updates to maintain this ability. Different from other operating system security enhancement scheme, TMAC has some unique advantages, including active defense, self-learning, less business impact, global security strategy for automatic upgrades and other characteristics.

In this paper, the first part would describe the power monitoring system security situation; the second part will introduce the security model and the key technology of TMAC; the third one will give some formal axiom, which is an important basis for the formal verification of the results; the fourth part mainly shows the performance and effectiveness; finally we would give the conclusion of this paper.

II. SECURITY SITUATION OF POWER MONITORING SYSTEM

After years of hard work, the power monitoring system forms a three-dimensional structure of the security protection system from three aspects: security protection technology, emergency backup measures, and comprehensive security management. The three aspects support each other, integrate with each other, and dynamically associate to form a dynamic three-dimensional structure. However, the current protection system of power monitoring system mainly focuses on physical security and structural security. In terms of ontology security, especially key chips and security operating systems, there are just certain attempts. However, in terms of security immunity, there is no result of scale application.

From the view of power industrial control system technology development, power monitoring system control unit or monitoring device are more and more using common operating system, common hardware platform, and common development language, with the development of IT technology, followed by the introduction of more general safety risk. In the event of the similar Stuxnet virus and Ukraine incident, the safety of power monitoring system will be seriously affected. At present, every equipment manufacturer has insufficient consideration of system security, and the technical means of safety protection for power industrial control systems needs to be standardized. It is necessary to strengthen the research and application of safety enhancement technology for power monitoring devices.

From the perspective of power monitoring system operation, it is difficult for the current power industrial control system to update patch; we need to introduce active defense. In the case of existing defects, it can still be immune to attack. As the industrial control environment is relatively fixed; the implementation of mandatory access control and trusted computing technology for secure operating system is relatively easy, and also it can achieve good results. There are a large number of embedded devices in the monitoring device, because of different shapes, different architectures, high performance requirements; it is difficult to implement high level security operating system, and its security protection is relatively blank; it is necessary to carry out the reinforcement research to adapt to the security situation of industrial control equipment. It is of great practical significance to reduce the maintenance cost of protection strategies and to realize the immunity of unknown attacks.

From the perspective of power safety supervision, the most important means of attackers is to obtain the authority through breaking the system restrictions, and then further operate the industrial control business. However, using the operating system itself to detect related threats is difficult, so we need to further strengthen the equipment ontology security and safety supervision platform, providing basic support for the construction of intelligent active defense system as well as the overall situation awareness. This paper studies the ontology security model with self-learning ability; realizes the self-learning of behavior model of system call and remote access; and automatically generates mandatory access control security policy, which will further improve the availability of active defense system. The advanced technology and application of ontology security advanced behavior analysis are studied. The automatic upgrade of security policy will provide strong technical support for global situation awareness.

From the perspective of the application of new technology, power monitoring platform would be part of the production control into the cloud platform. The monitoring platform itself is also the use of virtualization technology and container technology. There are virtual escape risks. Basic research on cloud platform will improve the overall security monitoring level, and can effectively block the security risks such as virtual machine and container escaping.

In the operating system ontology security integrity, the present studies can be summarized as follows:

A. TRUSTED COMPUTING AND SOFTWARE ASSERTION

National Development and Reform Commission Order No. 14 has emphasized ''establishing a safe immune mechanism based on trusted computing technology.'' Therefore, the TCM-based trusted computing technology system has been widely used. In trusted computing, the software assertion [1]–[5] is a technique used by remote entities to verify the authenticity of code running on a particular machine. Trusted computing and software assertion play an important role to build a monitoring system with the active defense features; however, it also has its disadvantages, which cannot solve the program's own vulnerability.

B. THE CORE FILE INTEGRITY CHECKING

Because malicious software execution often leads to changes in the core file system, file integrity detection [6] is the core operating system ontology security means, while these methods can easily be bypassed.

C. ROOTKIT DETECTION

Rootkit has the highest authority of the operating system, so it has great harmfulness, and the existing detection methods [7], [8] find insecurity by verifying the invariance of core

operating system data and the consistency of API calls, but are generally easily evaded

D. VIRTUALIZATION MONITORING

HUKO [9], [10] as the representative, virtualization monitoring also has a more distinctive features. These methods mainly use the CPU privilege level which is higher than ring0 to achieve real-time monitoring of the operating system. These programs can well meet the ontology security monitoring requirements, and can achieve good semantic acquisition and less performance loss, but it needs hardware support.

E. CONTROL FLOW INTEGRITY MONITORING

Integrity monitoring control flow CFI is always an important research direction of the ontology security. The main idea is based on compiler support, insert CFI detection code in the core jump code, and ensure correctness of flow. At present, there is already a lot of work in this area [11]–[13].

F. CO-PROCESSOR INTEGRITY MONITORING

The co-processor integrity monitoring based on integrity monitoring coprocessor is mainly through regular co-processor polling to verify the integrity of the system's core data [14], [15]; this method is run independently, but it has the problem of bypassing.

G. COVERT CHANNEL ANALYSIS

Covert channel refers to the violation of information channel access control policy in the mandatory access control, including the time covert channel and storage covert channel. In addition to the traditional Tsai semantic information flow analysis method, there are scholars using statistical method [16].

H. OS FORMAL VERIFICATION

Formal verification is based on established formal specifications, analysis and verification of the relevant characteristics of the system to determine whether the system meets the desired characteristics. Formal verification does not completely ensure that the performance of the system is correct, but it can maximize the understanding and analysis of the system, and try to find errors such as inconsistency, ambiguity, incompleteness and so on. There have been many achievements in formal verification. In 1992, the information display subsystem of the London Air Traffic Management System adopted a formal method in the demand phase to construct a VDM model [17]. The air traffic anti-collision system developed by the University of California Safety Critical Systems Research Group uses Statecharts-based demand state machine language RSML [18], and also formalized modeling in the requirements phase. Vassev and Hinchey [19] used the autonomous system specification language ASSL in the literature to establish a formal model for NASA tasks and generated functional prototypes. The French NLAAS-CNRS and EdF laboratories use a low-error-tolerant formalization technique in the real-time operating system of the France 4 nuclear power plant to effectively protect safety-critical systems such as nuclear power control [20]. SEL4 [21] is a formalized microkernel operating system that ensures that important and non-critical parts of the system operate independently. At present, the main application of SEL4 is still in the field of drones. The anti-hacking operating system CertiKOS [22], [23] developed by the Flint project team led by Shao *et al.* uses modular layered verification methods supports concurrency, multi-threaded on multiple CPU cores at the same time, and integrated refinement verification to ensure high credibility of the operating system kernel.

III. THE SECURITY MODEL BASED ON TMAC

In the TMAC security model, we implement an integrated security protection model combining trusted computing and mandatory access control. Through trusted computing technology, we can build a white list for executive programs, and eliminate ''alien'' viruses, etc. Mandatory access control can construct a white list of program behavior, to solve the program's own shortcomings and vulnerabilities, so as to meet the protection requirements, and provide technical support for the construction of active defense system. Trusted computing and mandatory access control together constitute the key technologies of active defense in power monitoring system. Based on this premise, the learning strategy can guarantee the completeness of strategy and self-learning refers to the use of a time window to run the business application. TMAC use this time window to automatically learn business activities and generate security policy. Self-learning reduces human intervention, so most of the security strategies are obtained through self-learning, and security operations staff does not need to know the professional knowledge of information security; it is easy to use. If we just want to protect a specific application, then other application will not be affected, so the impact on other businesses can be ignored. In addition, attacks often have comprehensive coverage, if single-node attack information can be timely shared with other ontology nodes, it will help to build a global defense system. TMAC's automatic upgrade of security policies implements this function.

No rules no standards, the role of TMAC is essentially to protect the business by applying the rules The normal business application logic is allowed by ''rules'', and its security model is simple and effective, so it will not interfere with normal business activities, can achieve the business impact minimization.

A. TRUSTED COMPUTING WITH AUTOMATIC IMMUNITY

The core idea of trusted computing is to calculate security protection at the same time of computation, including trusted boot, integrity measurement and trusted remote attestation. Trusted computing has changed the traditional protection mode of passive response such as ''blocking and killing.'' The core idea is to carry out security protection at the same time of calculation, so that the calculation result is always the same as expected. The calculation process can be controlled

and controlled without interference. It is a new calculation mode with both operation and protection coexistence and active immunity. In August 2014, the National Development and Reform Commission issued (2014) Order No. 14 "Safety Protection Regulations for Power Monitoring Systems" and "Overall Program for Safety Protection of Power Monitoring Systems" and other supporting technical documents. The overall solution requires a system with control functions in the production control area. The trusted computing technology is used to realize the security and credibility of the computing environment and the network environment, to establish immunity against malicious code, and to cope with high-level complex network attacks. The TMAC model mainly implements the following trusted functions:

1) TRUSTED BOOT

Trusted boot based on the IMA framework of Linux, we won't go into details here.

2) TRUSTED METRICS OF EXECUTABLE PROGRAMS

Trusted metrics for executable programs includes static integrity metrics and dynamic integrity metrics. Static measure refers to calculating the hash value of the object to be measured. Dynamic integrity measurement is an important calculation system of the measured object, including such as the code segment, data segment and read-only field core hash value; TMAC check the operation status to ensure the system running state being trusted.

3) CRITICAL DATA INTEGRITY MEASUREMENT

The critical data refers to the user configuration class, account class and other related information in the operating system. We achieve active detection based on the 'inotify' mechanism in Linux.

4) TRUSTED COMMUNICATION

Trusted label based on IPsec to realize trusted authentication and secure communication.

TMAC uses red black tree algorithm to store the trusted security policy, so it has the complexity of O (logN) algorithm, and has less impact on the startup of the system. TMAC can quickly find policies and make decisions at execution time

B. MANDATORY ACCESS CONTROL MODEL WITH SELF-LEARNING FEATURE

Based on the LSM framework, TMAC implements the function of mandatory access control and restricts the access of various applications to files, kernel permissions, networks, and system resources through MAC policies, and achieve its security function as a loadable kernel module form. TMAC mandatory access control uses three tuple security policies $<$ subject, object, permission $>$. The policies are simple and clear, and have strong semantic expression ability. It supports self-learning mode and compatibility mode. TMAC can configure the security policy automatically according to the

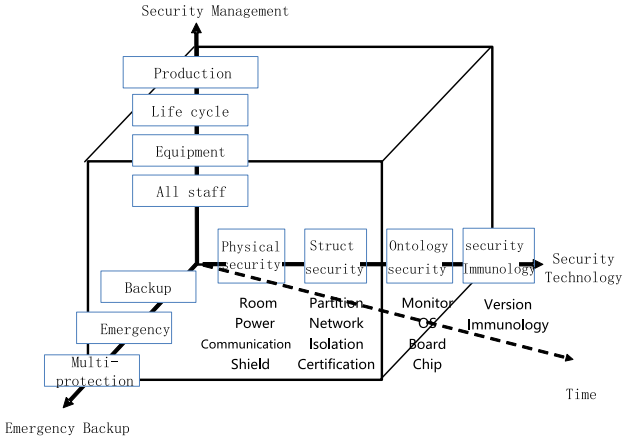


FIGURE 1. Power monitoring system protection system.

```
/root/apache-tomcat-6.0.41/bin/startup.sh {
  #include <abstractions/apache2-common>
  #include <abstractions/base>
  #include <abstractions/bash>
  #include <abstractions/consoles>

  /etc/centos-release r,
  /proc/*/net/if_inet6 r,
  /proc/*/net/ipv6_route r,
  /root/apache-tomcat-6.0.41/** rw,
  /root/apache-tomcat-6.0.41/bin/catalina.sh rix,
  /sys/devices/system/cpu/ r,
  /tmp/hsperfdata_root/ r,
  /tmp/hsperfdata_root/* rw,
  /usr/bin/bash ix,
  /usr/bin/dirname rix,
  /usr/bin/touch rix,
  /usr/bin/tty rix,
  /usr/bin/uname rix,
  /usr/bin/which rix,
  /usr/lib/jvm/java-1.7.0-openjdk-1.7.0.75-2.5.4.2.el7_0.x86_64/jre-abrt/bin/java rix,
  /usr/lib/jvm/java-1.7.0-openjdk-1.7.0.75-2.5.4.2.el7_0.x86_64/jre/bin/java rix,
  /usr/share/javazi/** r,
```

FIGURE 2. Mandatory access control policy of TMAC.

violation log, and define some policy as mandatory, and the other policies alert only. Specific policy is as shown in the following picture:

Based on the LSM framework, TMAC re-implements its security functions in the form of loadable kernel modules without significant loss in security or additional system overhead. TMAC's mandatory access control has self-learning capability, which provides a 'complaining' mode, in which TMAC does not block any business behavior, only records access to all objects by the monitored process, and generates the following logs. These objects include files, ports, devices, and memory objects.

The self-learning model of TMAC generates the security policy automatically based on the generated log. It includes two parts: (1) Self learning of deterministic events; (2) Recommendation of non-deterministic strategies.

1) SELF LEARNING OF DETERMINISTIC EVENTS

For this kind of event, the main program will not only access the corresponding object, but also call other executable programs to access the resources in most cases. The last resource access sequence forms a tree structure, as shown in figure 3. The initial body is the root node of the tree. The leaf node represents the real object to be visited. In the TMAC model,

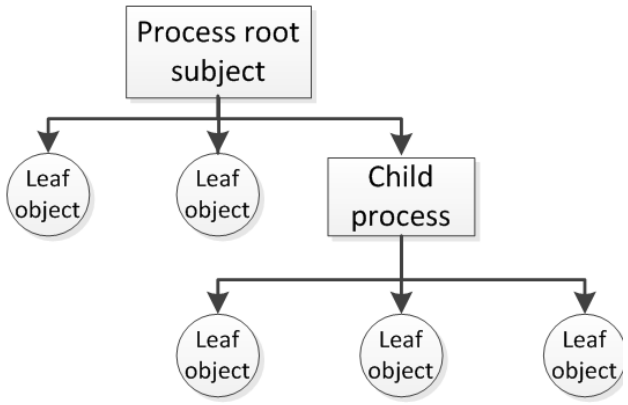


FIGURE 3. Tree access sequence of subject.

the principal body represented by the root node is the only principal subject of the whole behavior sequence, and all the leaf nodes are the objects they access. All access objects are explicit, and we call them deterministic access events. In the view of these access sequences, we complete the tree traversal algorithm with depth first to complete the traversal of the whole tree, and convert it into <subject, object, permission > security policy, where the subject is the root node, the object is all leaf nodes.

2) NON-DETERMINISTIC POLICIES RECOMMENDATION

When the behavior sequence for business systems has occurred; the tree structure is deterministic, so generation of three tuple < subject, object, permission > is deterministic. But a business application behavior sequence of one or more may not represent the entire business behavior, because the behavior sequence business may often be a big tree and the behavior of a few times may be just one subset. If a policy is defined directly based on deterministic events, then other sequences that could not be covered will be rejected or false positives, which will seriously affect the user experience.

To solve this problem, in the TMAC model, we performed lexical analysis of the three tuple, in which we merge non deterministic strategies, in order to try to cover the main tree behavior sequence in a certain range. For example, there are policies like <a,/usr/web0/a/1.html,r>, <a,/usr/web0/a/2.html,r> and <a,/usr/web0/b/1.html,r> in the behavior sequence. Then we would merge into <a, /usr/web0/a/∗.html, r> and <a, /usr/web0/b/1.html, r>, and even directly merge into <a, /usr/web0/∗/∗.html, r>.

Here, the deterministic learning strategies accurately describe the behavior sequence of business application, but it may be incomplete, so we focus more on the basis of the introduction of non-deterministic learning to portrait business behavior. Although the expanded part of the scope of the loss of a certain degree of security, but we think that this does not affect the safety of the whole situation, so it can be a very good way to solve the problem of incomplete business model.

C. AUTOMATIC UPGRADE OF SECURITY POLICY

The automatic upgrade of TMAC is mainly used to solve the problem of threat intelligence sharing, as well as the automatic negotiation upgrade of security policy.

TMAC exists in each operating system node in the form of agent. It has the following characteristics: 1) Distribution: it has neither global control nor global storage in physical and logical terms. 2) Connectivity: each agent is interconnected by network. 3) Collaboration: each system in the system can cooperate with each other to realize the sharing of threat intelligence and corresponding security policy. 4) Fault tolerance: using redundant communication, calculation and knowledge judgment to improve the reliability of work. Base on the behavior sequence from business behavior record, each TMAC node realizes the sharing of authority three tuples, and accept the recommendation strategy from other agent based on self-learning. According to the principle of majority compliance, each node makes its own decisions and up-dates its own protection policies to achieve self-escalation of the global protection situation.

IV. THE FORM DESCRIPTION OF SECURITY MODEL

This section describes the overall framework for formal design and verification. It is generally believed that the framework of feasible formal design and verification is roughly as shown below. The top level refers to the functional requirements, the basic working principle, the security attributes and strategies that should be satisfied, which represents the original needs of the designer. Demand is only the concept and idea expressed in the designer's mind or natural language. It is not a formal logical expression, and cannot be proved by reasoning. The intermediate layer includes specification of system, description of security attributes and formal model of system, which is a formal logic description. The bottom layer includes the hardware and software implementation of the system as well as the internal and external environmental factors of the system operation, which is the true implementation layer. This paper gives high-level formal description for logical expressions. The four axioms lead to illustrate the completeness of TMAC. On the one hand, trusted computing eliminates dissidents; on the other hand, mandatory access control addresses the threats posed by its own security vulnerabilities.

The TMAC model is a finite state machine model; subject represents entities such as processes that can initiate access, and the objects are defined as data files. The access patterns of subject to objects are divided into r w a c e (read only, read write, write only, execute and control), etc. C refers to the ability to authorize or revoke the access rights of other subjects to the object. The security policy of TMAC model consists of four parts: discretionary security policy, mandatory security policy, sensitive data security policy and Trust policy. The Trust policy is mainly used to mark the trusted subject.

A. MODEL ELEMENTS

Trusted subject is the subject that is in the white list, and the subject itself has not been tampered with. The subject under mandatory access control is the subject that exists in the white list of the trusted process and has the corresponding security policy. All subjects, trusted subjects, and subjects under mandatory access control are defined as follows:

$$\begin{aligned} S &= \{S_1, S_2, \ldots, Sn\} \\ S_T &= \{x | x \in Trust,\ Hash(x) = PTrust(x)\} \\ S_M &= \{x | x \in Mac, P(x) \in PMac\} \end{aligned}$$

Trust represents the white list of the trusted process, Mac represents the white list of the mandatory access control process, PMac represents the set of mandatory access control policies, and P(x) is the process policy query function.

The object, the protected object and the access attribute are defined as follows:

$$\begin{aligned} O &= \{O_1, O_2, \ldots, Om\} \\ Op &= \{x | x \in O, x \in DenyProfile\} \\ A &= \{r, w, e, a\} \end{aligned}$$

Among them, the protected object refers to the object defined by the object related mandatory access control policy. Unless the subject related mandatory access control has the definition of the relevant permissions, all processes associated with the object execute the policy.

We use the three tuple (B, M, f) to represent the system state:

$$b \in (S \times O \times A)$$

Represents the subject, object and security attribute in a specific state.

The discretionary access matrix is represented as follows:

$$M = \{M_1, M_2, \ldots, Mn\}$$

$f \in F$, on behalf of access function, identified as f = (FS, fo, FC), where FS refers to the main security level function; FC refers to the subject of the current security level function, and fo means the security level function object.

B. FOUR SECURITY AXIOMS OF TMAC MODEL

In order to explain what kind of state is a safe state, and what kind of system is a secure system, we introduce a set of security axioms. We believe that security is built under the following four axioms:

1) DISCRETIONARY SECURITY

State v = (B, M, f) satisfies discretionary security, if and only if all

$$(S_i, O_j, x) \in b \Rightarrow x \in M_{ij} \tag{1}$$

2) MANDATORY SECURITY

The simple security of the state v = (B, M, f) is defined as:

$$S \in S \Rightarrow [(O \in b(S : x, x)) \Rightarrow (f_s(S) \not\succ f_0(0))] \tag{2}$$

Among them, $\not\succ$ indicates that the latter item is dominated by the former one, and the function b(S:x_1, x_2,..., x_n) returns the collection of objects that the subject S has access to x_i to b. Mandatory security defines the requirements that the behavior of the system must satisfy the mandatory access control policy under the TMAC model. That is, the access authorization of the subject S to the object O.

3) SENSITIVE DATA SECURITY

The state v = (B, M, f) satisfies the sensitive data security if and only if

$$O \in O_p \Rightarrow \begin{cases} O \in b(S : x) \Rightarrow (f_o(O) \triangleright f_c(S)) \\ \forall S \in (S - S_M) \Rightarrow O \notin b(S : x) \end{cases}$$

If an object is defined as sensitive data, then it will only be subject to the control policy of the controlled subject, and others will be rejected.

4) TRUST SECURITY

The state v = (B, M, f) satisfies the trusted security if and only if

$$O_p \in Trust \Rightarrow \begin{cases} Op \in St \Rightarrow Hash(Op) = PTrust(Op) \\ 0p \notin St \Rightarrow Hash(Op) \neq PTrust(Op) \end{cases}$$

It ensures that executable objects whose hash has not changed can be executed; otherwise they will be rejected, thus eliminating the alien like virus and Trojan and achieving self-immunity.

These theorems constrain that any behavior of the system cannot violate these four theorems under the TMAC model.

V. EXPERIMENT

This chapter mainly introduces the experimental situation of TMAC. In order to evaluate the effectiveness and performance impact of TMAC, we performed three experiments in the environment shown in Table 1, where performance evaluation experiments were used to test the performance impact of TMAC, web protection effectiveness testing and virtualized escape protection were used to show the effectiveness of TMAC. The results show that TMAC does have its effect against web attacks and virtual machine escaping, with little performance impact. We conducted performance testing and effectiveness testing in the experimental environment shown in Table 1.

A. PERFORMANCE EVALUATION

AB is a suite of benchmark tests for HTTP Server that comes with Apache, which can simultaneously simulate multiple concurrent requests. In order to evaluate the impact of TMAC on the performance of the system and the service, we calculate the response time of the server by constantly adjusting

TABLE 1. The experiment environment.

Context	Version
CPU	Intel i7-2620 2.7GHz
Memory	16GB
OS	centos 7.1(Linux version 3.10.0-229)
Web server	Apache 2.2
Virtual simulator	qemu 1.5.3
Container	docker 1.11.0

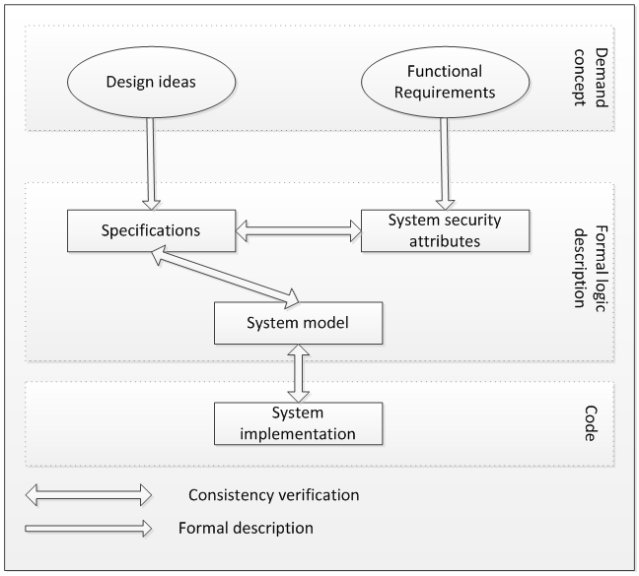


FIGURE 4. Formal proof framework.

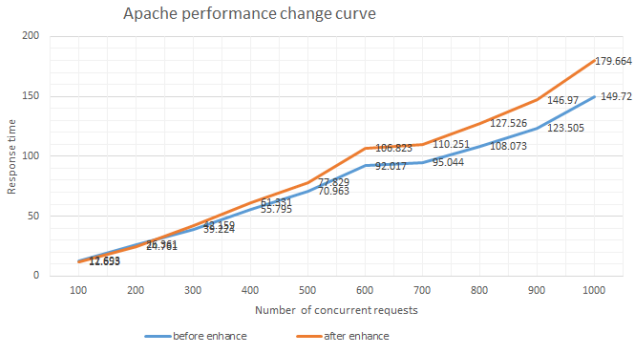


FIGURE 5. Performance comparison.

the number of concurrent requests. AB test results show that TMAC may introduce a corresponding delay of less than 5%, which means that the performance loss of TMAC is within acceptable range. Of course, the specific performance impact index is also related to the actual business applications.

To further evaluate the performance of TMAC, we alse selected several Linux benchmarks, including UnixBench

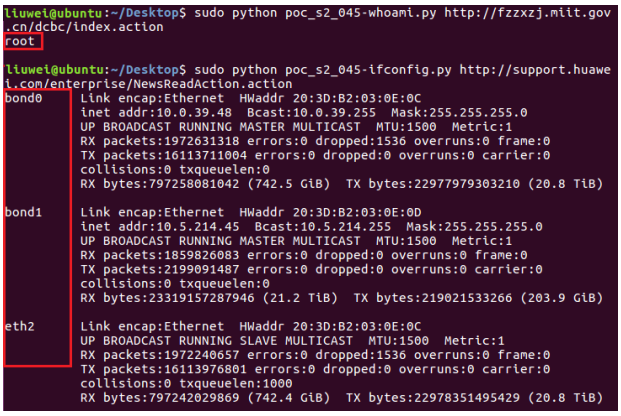


FIGURE 6. S2-045 vulnerability.

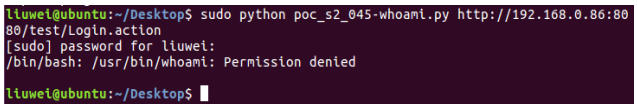


FIGURE 7. Web vulnerability blocked.

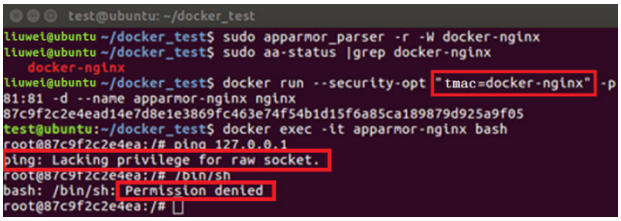


FIGURE 8. Container escape protection.

and other real world applications, to illustrate the performance of TMAC by comparing it with original operating system. The various software configuration information of the monitored system is shown in Table 2, where the 'decompressed file' operation is for CPU computing, the 'compilation' is for IO operations, and the UnixBench is a comprehensive test. Finally, I got a similar conclusion to the ab test; the performance loss does not exceed 5%.

TABLE 2. Software configuration information.

Name	Version
UnixBench	3.0-a9
Kernel build	2.6.34.12
Bzip2	1.0.4

B. EFFECTIVENESS TEST OF WEB PROTECTION

In March 2017, Struts2 0day vulnerability appeared (number S2-045) and was rated as high risk. The vulnerability has a wide range of impact, and can directly access the system root permissions, so the harm is huge. As shown below, we could easily access to a fund management system and ROOT through the POC, a website hardware card infor-

mation, in which the first instruction 'whom' executed by S2-045 returned root permission, and second instructions 'ifconfig' executed by S2-045 returned the hardware card information.

S2-045 is typical command execution vulnerability. TMAC controls the system call of the WEB middleware to the system commands through the mandatory access control policy of the operating system layer (obtained by self-learning). We can put an end to S2-045 at the OS level. It can fundamentally solve the security problems of this kind of application components. As shown in the figure below, the simple TMAC security policy effectively blocked the implementation of S2-045 vulnerability.

C. EFFECTIVENESS TEST OF VIRTUALIZATION ESCAPE PROTECTION

TMAC can be applied in the basic security protection of cloud platform, and build security protection in the host and virtual machine operating system, like Dockers, KVM and other levels; it can effectively prevent the virtual machineand container from escaping and other security risks.

In the TMAC model, if the component libvirtd daemon loads the configuration file, then each QEMU virtual machine creates a configuration file for the virtual machine when it is started. This configuration file is named using the UUID of the QEMU virtual machine and contains only the access rules required for its runtime, such as access to disk, PID file and log file, and so on. Before the QEMU virtual machine starts, the component libvirtd daemon will switch to this unique configuration file to prevent the QEMU process from accessing another QEMU process or accessing any file resource in the host. Because of the UUID binding, it ensures that each virtual machine or container uses different mandatory access control and is isolated from each other. As shown in the diagram below, the TMAC container can regulate the program behavior of the Dockers container, and the Nginx container can only carry web services, but cannot execute *ping* or other shell commands, which can effectively prevent the container escaping.

VI. CONCLUSION

In order to adapt to the ever-changing and increasingly serious security situation, this paper introduces the current security situation of power monitoring and control system, and introduces the key technologies of ontology security and active protection of power monitoring and control system based on TMAC model, including forced immunity and self-learning access control technology, and automatic escalation of security policy and global situation awareness. At the end of the paper, we give some formal description of axiom and show the effectiveness of experimental results.

Although TMAC can build operating system layer intelligent active defense system, the application of TMAC in the product is still insufficient; the research of global auto upgrade is in the initial stage. Research on self-learning and upgrading the global security policies will be to further carry out.

We are grateful for the support and assistance of 2017 Science and Technology Project of State Grid Corporation: Research on Key Techniques of Trusted Root and Trust Chain Construction in Power Cloud Platform. NO.5211XT17001G. Also, we thanks for editor's hard work and reviewers' valuable comments to our paper, which are very helpful for us to improve the quality of the paper.

REFERENCES

[1] H. Zhou, M. Ruan, C. Zhu, V. C. M. Leung, S. Xu, and C.-M. Huang, "A time-ordered aggregation model-based centrality metric for mobile social networks," *IEEE Access*, vol. 6, pp. 25588–25599, 2018.

[2] H. Zhang, Y. Qi, H. Zhou, J. Zhang, and J. Sun, "Testing and defending methods against DoS attack in state estimation," *Asian J. Control*, vol. 19, no. 4, pp. 1295–1305, 2017.

[3] J. I. Xiang-Min *et al.*, "Multiple third-party remote attestation mechanism based on credibility weights policy," *J. Shandong Univ.*, pp. 67–74, Jul. 2015.

[4] L. Yang, J. Ma, Q. Jiangm, and W. Lou, "A delegation based cross trusted domain direct anonymous attestation scheme," *Comput. Netw.*, vol. 81, pp. 245–257, Apr. 2015.

[5] G. Cabodi, P. Camurati, C. Loiacono, F. Savarese, and D. Vendraminetto, "Formal verification of embedded systems for remote attestation," in *Proc. WSEAS Trans. Comput.*, 2015, pp. 760–769.

[6] G. H. Kim and E. H. Spafford, "The design and implementation of tripwire: A file system integrity checker," in *Proc. 2nd ACM Conf. Comput. Commun. Secur. (CCS)*, Fairfax, VA, USA, 1994, pp. 18–29.

[7] N. Swamy, M. Hicks, G. Morrisett, D. Grossman, and T. Jim, "Safe manual memory management in cyclone," *Sci. Comput. Program.*, vol. 62, no. 2, pp. 122–144, Oct. 2006.

[8] B. Cogswell and M. Russinovich. (2007). *Microsoft Rootkit Revealer*. [Online]. Available: http://www.microsoft.com/technet/sysinternals/utilities/RootkitRevealer.mspx

[9] A. Srivastava and J. T. Giffin, "Efficient monitoring of untrusted kernel-mode execution," in *Proc. NDSS*, San Diego, CA, USA, Feb. 2011, pp. 463–472.

[10] X. Xiong, D. Tian, P. Liu, and A. Perrig, "Practical protection of kernel integrity for commodity os from untrusted extensions," in *Proc. NDSS*, San Diego, CA, USA, Feb. 2011, pp. 114–130..

[11] E. Shi, A. Perrig, and L. Van Doorn, "BIND: A fine-grained attestation service for secure distributed systems," in *Proc. IEEE Symp. Secur. Privacy*, May 2005, pp. 154–168.

[12] R. Sinha, S. Rajamani, S. Seshia, and K. Vaswani, "Moat: Verifying confidentiality of enclave programs," in *Proc. CCS*, 2015, pp. 1169–1184.

[13] I. Evans *et al.*, "Control Jujutsu: On the weaknesses of fine-grained control flow integrity," in *Proc. CCS*, 2015, pp. 901–913.

[14] N. L. Petroni, Jr., T. Fraser, J. Molina, and W. A. Arbaugh, "Copilot—A coprocessor-based kernel runtime integrity monitor," in *Proc. 13th USENIX Secur. Symp.*, San Diego, CA, USA, 2004, pp. 179–194.

[15] N. L. Petroni, Jr., T. Fraser, A. Walters, and W. A. Arbaugh, "An architecture for specification-based detection of semantic integrity violations in kernel dynamic data," in *Proc. 15th USENIX Secur. Symp.*, Vancouver, BC, Canada, Jul. 2006, pp. 289–304.

[16] Q. Xiao, M. K. Reiter, and Y. Zhang, "Mitigating storage side channels using statistical privacy mechanisms," in *Proc. CCS*, 2015, pp. 1582–1594.

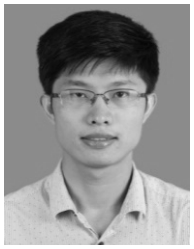
[17] A. Hall, "Using formal methods to develop an ATC information system," *IEEE Softw.*, vol. 13, no. 2, pp. 66–76, Mar. 1996.

[18] *Introduction to TCAS II, Version 7.1 [Z]. Version 1.2*, U.S. Dept. Transp., Federal Aviation Admin., Washington, DC, USA, 2014.

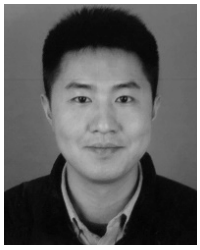
[19] E. Vassev and M. Hinchey, "Developing experimental models for NASA missions with ASSL," in *Proc. Workshop Formal Methods Aerosp. (FMA) EPTCS*, 2010, pp. 88–94.

[20] L. Andriantsiferana, J.-P. Courtiat, R. C. De Oliveira, and L. Picci, "An experiment in using RT-LOTOS for the formal specification and verification of a distributed scheduling algorithm in a nuclear power plant monitoring system," in *Formal Description Techniques and Protocol Specification, Testing and Verification*. T. Mizuno, N. Shiratori, T. Higashino, and A. Togashi, Eds. Boston, MA, USA: Springer, 1997, pp. 433–448.

[21] G. Klein *et al.*, "seL4: Formal verification of an OS kernel," in *Proc. ACM SIGOPS 22nd Symp. Oper. Syst. Princ.*, 2009, pp. 207–220.
[22] G. Ronghui *et al.*, "CertiKOS: An extensible architecture for building certified concurrent OS kernels," in *Proc. ACM 12th USENIX Symp. Oper. Syst. Design Implement. (OSDI)*, 2016, pp. 653–670.
[23] F. Xu, M. Fu, X. Zhang, H. Zhang, Z. Li, and X. Feng, "A practical verification framework for preemptive OS kernels," in *Proc. Int. Conf. Comput. Aided Verification*. Cham, Switzerland: Springer, 2016, pp. 59–79.



WEI LIU was born in Daping, Ganzhou, China, in 1986. He received the M.S. degree from Nanjing University, Nanjing, China, in 2012. He is currently the Director of the Security OS Research Office, NARI Group Corporation (State Grid Electric Power Research Institute), where he is mainly responsible for the underlying security research of the operating system, covering cloud security, mobile security, and industrial security to support the development of the entire department. His research interests include power system information security and operating system.



YIYANG YAO received the B.Eng. degree in software engineering and the master's degree in computer science from Northwestern Polytechnic University, Xi'an, China, in 2007 and 2010, respectively. Since 2010, he has been a member of the Information and Telecommunications Branch, State Grid Zhejiang Electric Power Co., Ltd. His research interests include network security and image processing.

BAOHUA ZHAO received the M.S. degree from the Beijing University of Technology in 2016. His research interests include trusted computing, information security, and computer technology.

WEIYONG YANG is currently pursuing the Ph.D. degree with Nanjing University. He has long been involved in power information security and has published many papers and patents. He is also with NARI Group Corporation (State Grid Electric Power Research Institute), Nanjing, China.

LONGYUN QI graduated from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2002. He is currently with NARI Group Corporation (State Grid Electric Power Research Institute), Nanjing. His main research direction is power system information security.

XIAOLIANG LV received the B.S. degree from the Ocean University of China, Qingdao, China, in 2007. He is currently with NARI Group Corporation (State Grid Electric Power Research Institute), Nanjing, China. His research interests include mobile information security and industrial security.