Dynamic trust measurement model based on random number simulation

Yu Songsen , Zhang Jiantao , Zheng Yuechi , He Zhicong , Lin Mingfeng , Cao Zhibin

South China Normal University , Software College , Foshan , Guangdong ,China

342022764@qq.com,12774472231@qq.com, 1769826756@qq.com, 172227846@qq.com, caozhibin199677@163.com

Key words：trusted computing, dynamic trust measurement model, data security checksum, attestation of integrity

Abstract: The study of dynamic trust measurement is a hot spot in the current field of trusted computing. In this paper, aiming at the shortcomings of the chain in the trusted measurement model, we try to dynamically measure the data security of the current entity with the extension of the data check on the trusted computing platform module and then to determine the credible degree of the entity. As for the experimental method, we bring in the random simulation method, combining with the theoretical calculation of the classical probability to get the credibility of entities or executable files; by skipping the procedure of matching the standard integrity measurements with the actual integrity measurements, we improve the efficiency of the measurement to a certain extent and keep the behavior of the entity in a trusted state, and furthermore, to ensure the applications over the operating system with various entities of trust and security.

Introduction

Information security problems has been one of the hottest and core problems in the computer industry especially in this day when mobile terminal has been all over the world. As a consequence, information security problems need more attention. Nowadays, traditional firewalls and antivirus can't deal with all kinds of security problems thoroughly. In other words, these traditional methods have not been able to solve the security problems fundamentally. Therefore, putting forward a new model which can deal with such security problems and putting it in the practice is imminent.

The thesis[2] says the simplification of the software and hardware architecture led to a security risk that resources can be used arbitrarily, the code can be tempered with at will, even executable file can be even modified and malicious programs are easily to be implanted. In order to solve this problem fundamentally, we must work from the resource and the underlying structure like chip, hardware and operating system. In other word, we must try to ensure the stability and security of the upper application and network based on setting up defenses at source of invasion.

Thus, the concept of trusted computing has emerged, which is used to deal with the problem that information security incident occurs frequently in the information age of prosperity unexampled. The international trusted computing group TCG gives a definition of "trust" as follow: An entity is credible when its behavior always achieve the desired goal as expected. The technology of trusted computing bases on a trusted computing platform supported by the hardware security module which is widely used in computer and communication systems to improve the integral safety. The foundation of trusted computing is the trust from the source of the system, the requirement of it is trustworthy system architecture, behavior, allocate resource and user's stored data.It works from chips, mainboard, BIOS and operating system, builds a structural system of trusted computing which uses TPM as trusted computing platform and measures, and then stores and reports the integrity of the platform, data and behavior by the way of integrity measurement.

Early TCG's standard put forward a chain model to deal with the model of trusted computing. **The thesis[1]** says the chain model is divided into two subchain models, which are two totally different trusted computing model. The static subchain model is to deal with the process which is from the underlying hardware platform to BIOS then to operating system and this model is quiet perfect and has been used widely in PC platform and mobile terminal. But the other process, beyond operating system such as application layer and network can't use the static subchain model because the data and the behavior of software is more variational. Therefore, such process need a way of dynamic trusted computing to deal with the information security problems of application layer. However, TCG standards didn't give a strict standards in the aspects of dynamic trusted measurement. So the research of dynamic trusted measurement is the key point in the field of trusted computing. In this article, we will introduce another theory to improve the dynamic trust measurement model.

Trusted Computing Platform

the basic function module

In **thesis[2]**,the author mentioned that the basic security functions of Trusted Computing Platform, one of the core of Trusted Computing, which includes integrity measurement and reporting of the platform, trusted platform identity and data security protection, etc.

Platform integrity, refers to the platform components to make use of password system integrity measurement, and report to the external entity platform. In fact, the method is to collect the integrity of the platform components measurement values, and compare to the benchmark measure of value given vendor or a third party, that is the expected value, when both of the components and programs operating results are meet the expected value, we can say that the integrity of the component has not been destroyed. Building trust from the root to the various components of a trust relationship by integrity measurement mechanisms is in order to protect the integrity of the platform. In general, the integrity mechanism is divided into three steps: integrity measurement, integrity storage and integrity report, whose basic principle is: a platform may be allowed to enter any state, including unnecessary or unsafe condition, but platform can't whether the entry or exit for concealing and modify this state. In fact, an independent process should be able to evaluate the integrity of the state and make the right response.

As for the trusted platform identity, it is the only identity identification of the platform, and provides the external entity with the platform identity certificate and the application identification service. This function issues every platform a unique identity certificate, to identify the identity of the platform by using the password mechanism, so the verifier can verify the identity of the platform through the third party.

Finally, the data security protection is to provide protection of sensitive data to system platform, in which the sensitive data include the sensitive data of the platform itself and the user's sensitive data. This function realizes the safe storage of sensitive information, and adds the platform configuration information, and accomplishes the binding of the platform state. In addition, this function can provide the service interface for the user data protection.

In short, trusted computing platform is a support system which is used to construct the three trusted computing functions in the computer system, and constructs the architecture, in which the cryptography is as the basis, trusted platform control module is as a trusted root, trusted motherboard is as a platform, credible foundation support software is as the core, and the trusted network connection is as the link of the architecture.

The Expansion of the Functional Modules

Trusted computing platform in addition to the above three basic function modules, we also try to introduce another function module for the trusted computing platform and name the module as data-check module from the perspective of theoretical analysis. In this way, the trusted computing platform function module has been further expanded. Because data check has the extremely widespread application in the field of computer, introducing the data-check function into dynamic trust measurement model can greatly improve the efficiency of the dynamic trusted measurement, ensure the stringency of the dynamic trusted measurement, eliminate the unsafe data in the process of software running and reserve the safe and effective data which is actually useful when applications are running.

The introduction of data-check module should be based on the data classification, The **thesis[4]** mentioned. From a macro point of view, the data in the computer system can be divided into the secure data and insecure data. The insecure data can be further subdivided into suspicious data and malicious data. These two kinds of insecure data are the "eyesore" of the dynamic trust measurement, they need to be treated timely to ensure the security of the current application’s operation. Based on this classification, we make a general division of the data-check module, and add the data check area to the traditional trusted computing platform, and then we can get the trusted computing platform, as shown in the following figure:

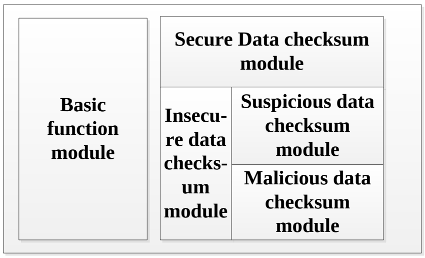

Basic function module
Secure Data checksum module
Insecure data checksum module
Suspicious data checksum module
Malicious data checksum module

Fig.1 Extension of Trusted Computing Platform

Trusted computing platform which has added data-check module ,will have extended function which is checking the data of the entity currently running. After checking, based on data classification, the data will be stored in different check area for subsequent processing (the processing of specific rules will be explained in given below).

Random number simulation

random number generator $RAND()$

Random number generator $RAND()$ can be used to simulate the circulating data of produce within the application. For a given entity, the data is completely in accordance with data classification standards that we just mentioned, which includes security data, suspicious data and malicious data. Here we give two important properties of the random number generator, the two properties will point to the simulated and the corresponding relationship between random number generator and the application data, or rather, random number generator the data come from the entity.

The nature of the $RAND()$:

(1) When the $RAND()$ is not used for simulating the circulation data within the entity, the data, it generates, doesn't make any sense;

(2) When the $RAND()$ is used for simulating the circulation data within the entity, the data resulting from the $RAND()$ will be with the categories, namely to simulate the circulation of security data, suspicious data and malicious data within the entity.

Random Number Simulation

With the nature of the random number generator, we can give the concept of random simulation in the applications of dynamic trusted measurement. Random simulation is to make the random number generator $RAND()$ associated with a given application and because of the association, these data have been classified as security data, suspicious data and malicious data, which implements the simulation of data.

As mentioned earlier, under the condition of excluding data outside, the research methods of dynamic trusted measurement, more specifically, by extending the data-check module on the trusted computing platform, dynamic measuring the current data's security, and then determining the credible degree of the entity. In order to put the dynamic condition and random reflect more comprehensively, we need to set the seed of the random number generator for the current time, which can reflect that it occurs randomly and changeably when the condition that entity is not credible happens. At this moment, the random number generator can be expressed as $RAND(Seed(Current))$.

Dynamic trusted measurement

A sequence of microscopic entities' behavior based on dynamic moments

The concept of the sequence of microscopic entities' behavior is derived from the software behavior trace, The **thesis[3]** mentioned. Its essence is cutting up the time in limited time for a given process (application or entity according to the length of the running time). So, a given process will be divided into the child process under each dynamic moment, or the sequence of microscopic entities' behavior based on dynamic moments is a child process group. Here we give the definition and nature of the sequence of microscopic entities' behavior based on dynamic moments:

(1)Definition: for any given period of time $T = [t_1, t_i]$ and specific entity A, the sequence of microscopic entities' behavior based on dynamic moments can be expressed as $Seq(A) = A_1A_2A_3...A_i$, A_i is a specific behavior of A in one moment. Among then, $i \geq 1$.

(2)、Property:

(2.1) Entity A is trusted if and only if $Seq(A)$ is trusted, that is, $A_1, A_2...A_i$ are all trusted.

(2.2) For any entities A and B, respectively, we have the formulas that $Seq(A) = A_1A_2A_3...A_i$ and $Seq(B) = B_1B_2B_3...B_{i+k}$, if A_i=B_i, wherein $k \geq i \geq 1$, then it can be judged that $Seq(A)$ is the child process sequence of $Seq(B)$, and, we cannot inferred that $Seq(B)$ is trusted through the trusted $Seq(A)$, however, if $Seq(B)$ is trusted, it is sure that $Seq(A)$ is also trusted.

Preliminary description of Data Classification and the relationship between data and $\boldsymbol{RAND()}$.

When we put forward the concept of trusted computing platforms, we have already mentioned data can be divided into three categories, they are safe data, suspicious data and malicious data. The suspicious data is an intermediate type of data and it has its own unique meaning, we define that the suspicious data is one kind of data which is similar with the safe data but also with some modification. The meaning of data with modification is that the current data might seem normal but they are likely to be modified when it is being used, and it is difficult for us to predict the result of the modification, and it needs further monitoring and judging in the monitoring area. According to the intermediate role which is called suspicious data, we first define them into two categories, they are suspicious data and non-suspicious data. There is no doubt that non-suspicious data include the safe data and the malicious data. As for the suspicious data, we use $a[modify(a)]$ to represent suspicious data, it means that a is modified accidentally, and we also use $a[\emptyset]$ to represent non-suspicious data, and it means that a is impossible to be modified, then the preliminary description of data classification is completed.

Combined with the random number generator and the random number simulation mentioned earlier, since data a flow within an entity, it must be associated with a random number generator, more specifically, data a is the return value of $RAND()$,though the value of a is random and uncertain. However, But no matter how the data a is random, it is always in line with our preliminary classification criteria, data a is the one of the suspicious data and non-suspicious data. The relationship between data a and $RAND()$ can be represented by the following formula:

$$a = a[\emptyset]||a[modify(a)] = RAND(Seed(Current)) \quad (1)$$

data security mapping and refined data classification

Data security map is used to determine the safety of data which is flowing into a dynamic time action $A_i(i \geq 1)$ of the microscopic entity action sequence which is called $Seq(A)$, we suppose that a_i is used into the A_i, the data of an action changes randomly, so we get the formula:

$$a_i = a_i[\emptyset]||a_i[modify(a_i)] = RAND(Seed(Current)) \quad (2)$$

System policy makers to develop trusted policies need to further develop the range of data security for one or more executable files in the system. The range is not fixed, it depends on the specific application, Here, we assume the range of data security is that $Safety = [s1，s2]$.

Once we have the range of data security, data classification can be refined. And then we are able to get the classification which is mentioned at the time when we put forward the theory of the Trusted Computing Platform extension. It is sure that $a[\emptyset]$^$(a[\emptyset] \in [s1, s2])$ is secure, but $a[\emptyset]$^$(a[\emptyset] \notin [s1, s2])$ is malicious. non-suspicious data $a[\emptyset]$ is mentioned previously when we do the initial classification, so $a[\emptyset] = a[\emptyset]$^$(a[\emptyset] \in [s1, s2]) + a[\emptyset]$^$(a[\emptyset] \notin [s1, s2])$,and then non-suspicious data have a more detailed classification, they are secure data and malicious data. $a[modify(a)]$ is still used to represent the suspicious data.

After finishing the data classification, we can introduce the definition and expression of the data security map. Data security map is a special map, its data defined domain is the data collection based on the current sub-behavior of the action sequence which represents the entity under a dynamic time, and the map's range is the data security certification collection, that is $ST = \{safe，suspicious，malicious\}$. Data security certification collection and data classification has a one-for-one relationship, data security map can mathematically expressed as:

$$f(a) = \begin{cases} safe & if\ a\ is\ \ a[\emptyset]\hat{}(a[\emptyset] \in [s1, s2]) \\ suspicious & if\ \ a\ \ is\ \ a[modify(a)] \\ malicious\ if\ a\ is & a[\emptyset]\hat{}(a[\emptyset] \notin [s1, s2]) \end{cases} \tag{3}$$

In addition, data security map need to be associated with the monitoring area, The purpose of establishing this association is to handle the suspicious data properly. After the mapping, the data which is considered as suspicious data needs to be transferred to the monitoring area to monitoring their dynamic change. if data a or data set a is overwritten but after data security mapping, they are determined to be safe, then a can out of the monitoring area and transferred into the entity. If the data is determined to be malicious, then the circulating of the data will be revoked. Malicious data itself will die in the monitoring area or move to the malicious data checksum module. This association is usually used in other dynamic trust measurement model in **thesis[4]**,and it is called the rules of consequence-change judgement.

The probability of generating malicious data and the dynamic trust measurement rule

In reality, it is random and uncertain that when the malicious data will attack the executable file, so this feature matches the way of random number simulation, generally speaking, we can use some mathematical probability module to describe how many times the malicious data attack the executable file, for example, the Poisson distribution. Here, we combine the most intuitionistic module---the classical probability module with the theories as well as rules we define before to predict the probability of when the malicious data create.

From the regulation of data classification, data security mapping function as well as the rules of consequence-change judgement, malicious data is consist of originally malicious data and the malicious data which is transferred from the suspicious data after the monitoring of the monitoring area. When an application is connected to a specific random number generator, we have the following expression:

$$malicious\ data = RAND(Seed(Current))|_{malicious+(suspicious \rightarrow malicious)} \tag{4}$$

Setting the start times of data security mapping to n, with a counter $count()$ to note down the times of generating the malicious data, combined with the classical probability module's theory and the total number of sample place's basic event is $N(S) = n$, the total number of basic events that are contained in the event of a malicious data is :

$$count(malicious\ data)=count(RAND(Seed(Current))|_{malicious+(suspicious \rightarrow malicious)}) \tag{5}$$

Bernoulli's law of large numbers points that if n is the times of Bernoulli experiment A and the probability of experiment A in each test is P, then for any positive number ε, we have:

$$\lim_{n\to\infty} P(|\frac{\mu_n}{n} - p| < \varepsilon) = 1 \tag{6}$$

Thus when n is a big number, if we use the frequency to represent probability, we know that the probability of how many times the malicious data attack the executable file is:

$$P=P(malicious\ data)= count(malicious\ data)/n \tag{7}$$

At this point, we have to let the framer of the system strategy write down the expected fault tolerance probability of each executable file($P0$) into the executable file list(Software function list FL mentioned in **thesis[5]**). In traditional way, we would compare the integrity measurement to the integrity standard digest value, on this basis, we assume there is no other influence except the data, use the probability comparison to get the executable file's reliability, or we can call it the reliability of the application, and then we put forward the following dynamic trust measurement rule, set the current running entity to A, then:

$$G(A) = \begin{cases} trust & if\ \ P \leq P_0 \\ malicious & if\ \ P > P_0 \end{cases} \tag{8}$$

Because probability P is given after the executable file running for many times, it will be write into the executable file list, so, when any malicious file is pulsed-on accidentally, by comparing P with P_0 , also P >P_0,we can get rid of the complicated calculation process of traditional dynamic trust measurement model and improve the measurement efficiency.

The architecture figure of the dynamic trust measurement model

Now we provide the architecture figure of this article, it is the organic combination of the article as well as the theories we predict before, and readers can make a comparison.

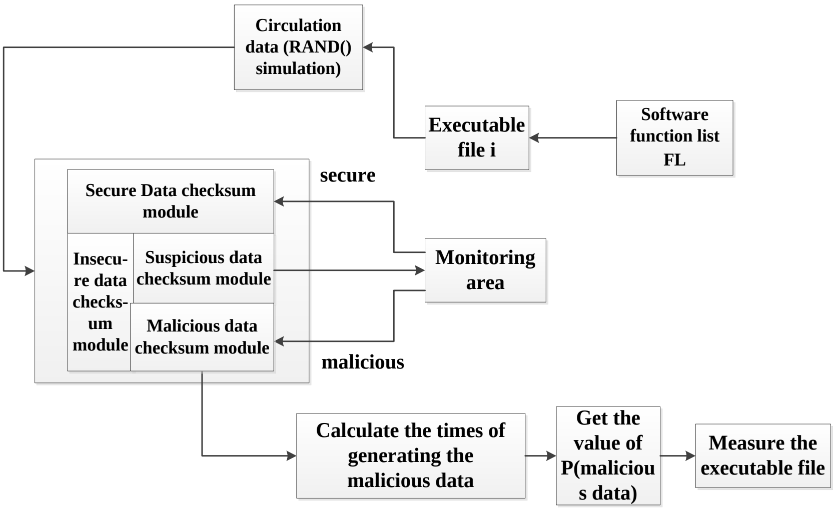
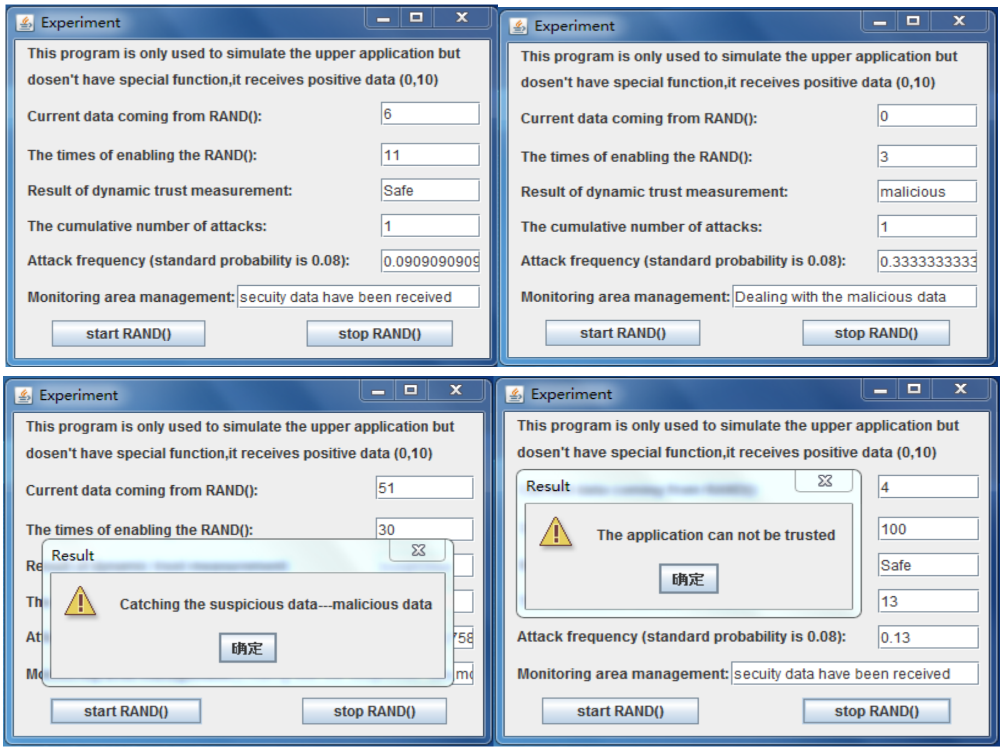

Circulation data (RAND() simulation)
Executable file i
Software function list FL
secure
Secure Data checksum module
Insecu-re data checks-um module
Suspicious data checksum module
Malicious data checksum module
Monitoring area
malicious
Calculate the times of generating the malicious data
Get the value of P(malicious data)
Measure the executable file

Fig.2 The architecture figure of the dynamic trust measurement model

Simulation experiment

We use a simple simulation experiment to explain the model that has been put forward earlier, specially, for a particular application of the upper layer, we can develop more specific and complex policy of dynamic trust measurement. Suppose an application (Here we don't provide its function) receive only positive numbers 0 to 10(not include 0 and 10) and the number that less than 0 is malicious data and the possible tampering behavior is modifying the integer numbers and changing them into character or string. This behavior will be determined as a special suspicious behavior, Whereby the received data will be identified as suspicious data (usually larger than 9), and then the monitoring area will further deal with the suspicious data. What's more, we assume that the standard probability of dynamic trust measurement is 0.08, since the probability of generating the malicious data is the result of a large number of experiments, here we explain the model just through 100 experiments in a dynamic time and according the contents from 3.1 to 3.4, we are able to get the following results of the experiment:



This program is only used to simulate the upper application but dosen't have special function,it receives positive data (0,10)

Current data coming from RAND(): 6
The times of enabling the RAND(): 11
Result of dynamic trust measurement: Safe
The cumulative number of attacks: 1
Attack frequency (standard probability is 0.08): 0.0909090909
Monitoring area management: secuity data have been received

This program is only used to simulate the upper application but dosen't have special function,it receives positive data (0,10)

Current data coming from RAND(): 0
The times of enabling the RAND(): 3
Result of dynamic trust measurement: malicious
The cumulative number of attacks: 1
Attack frequency (standard probability is 0.08): 0.3333333333
Monitoring area management: Dealing with the malicious data

This program is only used to simulate the upper application but dosen't have special function,it receives positive data (0,10)

Current data coming from RAND(): 51
The times of enabling the RAND(): 30
Re
Th
At 758
Mo mo

Catching the suspicious data---malicious data

This program is only used to simulate the upper application but dosen't have special function,it receives positive data (0,10)

4
100
Safe
13
Attack frequency (standard probability is 0.08): 0.13
Monitoring area management: secuity data have been received

The application can not be trusted

Fig.3 The result of the simulation experiment

Conclusions

Our article is based on the basic theory of trusted computing, and we try to improve the shortcomings of TCG specification and improve the model by using the random number generator to get the data security determination method. What's more, we also use the theoretical calculation of classical probability to get the reliability judgement of entity or executable file. Instead of comparing the integrity measurement to the integrity standard digest value, we improve the work efficiency to some extent, and ensure the trust comes from the entity, so that we can make sure any running program on our operating system is safe and credible.

Acknowledgments

This work is supported in part by Guangdong Province high-tech science and technology project(2013B010406008),Guangdong modern information service Development Project(GDEID2012IS052),Guangzhou Application of basic research project(2014J4100111),Foshan science and technology project(2014AG100162),Shenzhen Application of basic research project(JCYJ20130401095559824) and The National College Students Innovation Training Project.

Reference

[1] Na Xu, Wei Wei TPM security chip-based design and implementation [J] Computer Applications and Research, 2006 (8): 117-119

[2] Li Ruihua trusted computing platform integrity measurement and physical measurement behavior research [D] Beijing: Beijing University of Technology, 2010: 1-73

[3] Johnson's Lu, Cai Mian, Li Chen credible measure of software-based dynamic behavior [J] Wuhan University: Natural Science Edition, 2010,56 (2): 133-137

[4] Yang Bei, WU Zhen-qiang, Fu Xiang Ping integrity measurement model based on the dynamic Trusted Computing [J]. Computer Engineering, 2012,38 (2): 78-81

[5] Cao Jihong, Lixie Hua, Xu Ming-song, Fan Qing Research and Design of a complete chain of trust trusted mobile terminal model [J]. Computer Engineering and Design, 2012, 33 (3): 911-915