2011 International Conference on Mechatronic Science, Electric Engineering and Computer
August 19-22, 2011, Jilin, China

Formal Analysis and Verification on a Secure Microkernel

Xiaokun Zhang, Xuying Zhao
Beijing Electronic Science and Technology Institute
NO.7 Fufeng Road, Fengtai District, Beijing, 100070, P.R.China
{sam, xyzhao}@besti.edu.cn

***Abstract*—CASMonitor, short for CAS Virtual Monitor, is a secure, high-assurance hypervisor prototype, which aims to level B3 or higher of TCSEC standard. This paper combines the B Method and Model Checking Technology to verify certain key problems of the CAS Monitor. To secure Os of B3 level, we use B language to construct formal models for the Security Policy Model and the Formal Top-Level Specification (FTLS). We then conduct Consistency verification on the ChineseWallPolicy model and the refined FTLS model and have found them to possess logical semantic.**

Keywords-CAS Monitor;Secure Operating System;Model Checking

I. INTRODUCTION *(HEADING 1)*

With the rapid development of computer technology and the growth of the Internet, cloud computing has now become an key area of concern. In a cloud computing environment, software services are provided over the network in a dynamic and scalable manner, often utilizing virtualized resources. Hypervisor, also called Virtual Machine Monitor (VMM), is a piece of software/hardware platform-virtualization software that not only allows multiple operating systems to run on a host computer concurrently, but also provides new features such as dynamic resources deployment, computation transferring etc. Hypervisor technology is able to support many operating systems on different hardware platforms, hence providing basic software technology in creating a cloud computing environment. CASMonitor [5], short for CAS Virtual Monitor, is a secure, high-assurance hypervisor prototype, which aims to level B3 or higher of TCSEC standard [2]. CASMonitor can be viewed as a Microkernel stemmed from XEN [4], including 55000 loc of C/ASM, and realizes many important mechanisms of XEN such as domain, event-channel, grant-table, Hypercall, Mandatory Access Control (MAC) etc.

The B Method [3] is a formal method to describe and design software systems in order to create the software system code. It stems from first-order logic and set theory, as well as pure mathematics elements such as statistics, mapping and substitution. It can be supported by firm mathematical theory and hence its specifications are concise, precise and without absurdities. This paper uses the B Method to describe the CASMonitor MAC model. In the MAC model, a Chinese-Wall control policy is equipped, which is used as labels interpreted to decide which domains can co-exist on the same system. Then we refine the MAC model to get Formal Top-Level Specification (FLTS), finally completing a correspondence analysis and conformance testing between FTLS and the Secure Policy Model.

This paper is organized as follows. In Section 2, we describe the MAC module together with its B model. In Section 3, we discuss the refined prototype system of FTLS based on the ChineseWallPolicy model. In Section 4, we test the correspondence and conformity between FTLS and the Secure Policy Model and analyze the results. In Section 5 we show the related work, and finally conclude this paper.

II. CHINESEWALLPOLICY MODEL

A. *Separation of Duty Model based on Chinese Wall Security Policy*

The Chinese Wall [1] Security Policy was earliest conceived by Brewer & Nash based on a realistic commercial business model. It divides all the information of a company into 3 levels of storage. The base level consists of single data elements, the next consists of Company Data (CD), and the top most level consists of information with regards to Conflicts of Interest (COI). Each company can only belong to one COI. Initially, an user may visit any CD within a COI without any limiting or enforcement factor. However, once he makes a decision, he cannot access any other elements of the same COI, it is as if a wall has been created around the CD. As shown in the figure below, CD1 is a subset of COI1, once a user accesses CD1, he cannot access any other information under COI1 such as CD2 (shaded region in Fig. 1).

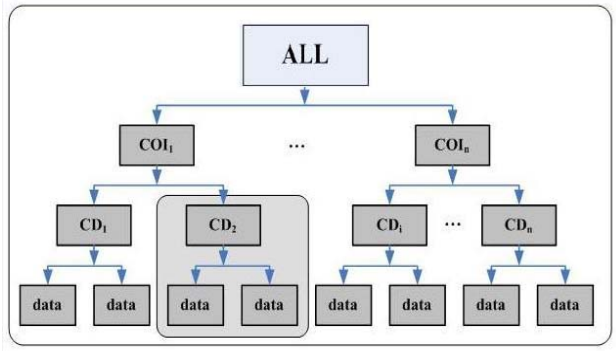

Figure 1. Chinese Wall Security Policy

The guest operating system we have defined to be working on a VMM can independently access data in a

computer system. Using DOMAIN_SET, each guest operating server is an element. To define permission to access data in his computer system, we use ssidrefs_Type to represent the set of access rights. Role and rights allocation is determined by a kind of algorithm in VMM and here we are primarily concerned about the isolation criteria of the security policy. The segregation of duties is so as to prevent any domain administrator from requiring two or more domain rights to accomplish a task, hence ensuring that each domain is highly isolated. If a task requires more than one right, such rights need to be allocated to two or more roles. These roles are conflicting roles as they complete one task. The conflict set contains all the sets of conflicting roles, cr is used to represent the conflicting roles of a task, and ConflictSets_Set is used to represent the conflict set.

We segregate the rights required to complete one task, e.g. the rights required for Task t1 can be represented as P1={p11,p12,p13,p14}, for Task t2 the rights can be represented as P2={p11,p22,p23}. Under secure conditions, rights have to be allocated to two or more roles and completing a single task lead to the conflict of roles set cr. If we could allocate P1={p11,p12,p13,p14} of Task t1 to r1 and r2, perm(r1)={p11,p12| p11,p12 ∈ P1} , perm(r2)={p13,p14| p13,p14 ∈ P1}, where r1 and r2 are conflicting roles, hence create the conflicting roles set cr of Task t1. As such, we have created a Duty Segregation Model based on Chinese Wall.

B. Modeling the Security Policy

The Security Policy in this section is thought of as a B Abstract Machine, mainly realizing the creation and deletion of domains under Chinese Wall control by API as following.

```
MACHINE ChineseWallPolicyModel
SETS
        DOMAIN_SET                                      =
{DOMAIN0,DOMAIN1,DOMAIN2,DOMAIN3};
        ssidrefs_Type = {aa, bb, cc, dd} ;
        conflictSets_Set ;
        accessed_Set
CONSTANTS
        maxTypes,
        maxSsidrefs,
        maxConflictsets
VARIABLES
        created_domain,
        ssidrefs,
        conflictAggregateSet,
        runningTypes,
        conflictSets,          /*conflict Sets*/
OPERATIONS
        CreateDomain(t_domain,t_ssidrefs)
        DeleteDomain(t_domain)
END
```

As in Fig. 2, the following B format model, the variable ssidrefs has been defined as the map of an existing domain according to its signal type. The variable conflictAggregateSet represents the types of conflicts that are in process and is defined as a subset of the signal type set. Initially defined as system default, the conflictAggregateSet does not make adjustments to any strategic methods. Hence the security policy is stable.

```
!(nn).((nn:conflictAggregateSet)=>
(!(mm).((mm:created_domain)=>
(!(xx).((xx:created_domain)=>
((ssidrefs(mm): nn &mm/=xx)=>(ssidrefs(xx)/:nn)))))))
```

Figure 2. Chinese Wall model

Before create a new domain, we must check the conflictAggregateSet and update the corresponding global data structure. In addition, after a domain is destroyed, a clean operation is needed.

III. FORMAL TOP-LEVEL SPECIFICATION(FTLS)

A. Refinement of the B Method

Refinement is a technology as well as a key task in software development. The goal of which is to develop the system specification (Abstract Model) into a more specific model. As compared to the original, the new model's tangibility is as follows:

- It can contain more details with respect to original system requirements and non-formal specification.
- It is closer to realizing a system as its mechanics are closer to the workings of an actual computer.

In the process of using B Method, refinement is strictly an abstract concept. Refined "accuracy", i.e. Semantic preservation, can verify the correctness of the final development results. We can use the security model (from Section 2) as the basis of our discussion and through continuation and refinement creates a more specific abstract machine systematic to FTLS. Refined CASMonitor layers are shown below:

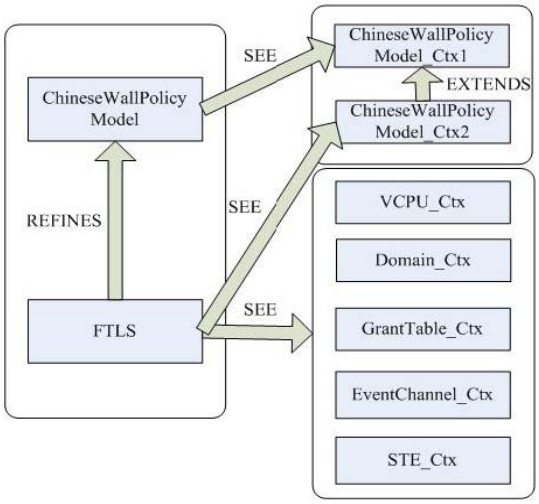

Figure 3. Refined CASMonitor layers

FTLS designs and actualizes many important mechanics of XEN such as, Domain, Event Channel, Grant Table, Hypercall, Mandatory Access Control etc. The following will explain each in detail.

B. Domain

The data structure of a domain records the information crucial to a domain in the core. This information is used in changing and managing domains. Of these, DOMAINS is the set of all domains, DOMAIN_ID is the overall indicator of the domain, CodeSpace represents coding space of all the domains, DataSpace represents all data-

containing space of all the domains and StackSpace represents the Stack space of all the domains.

Domain statistics describe all the information a core contains, excluding domain codes and user mode content. The precise B code is as follows:

```
MACHINE  DomainCtx
SETS
      DOMAINS;
      DOMAIN_ID ;
      CodeSpace ;
      DataSpace ;
      StackSpace
CONSTANTS
      domain0,
      domain0_id0,
      codespace0,
      dataspace0,
      stackspace0
END
```

C. Scheduling

Scheduling is used to schedule each domain in a just and reasonable manner. As the domain operates on a virtual CPU, and each domain is highly segregated, hence we only need to schedule the respective virtual CPUs in order to schedule the domains. The function of scheduling is actualized in the abstract machine model Schedule.

doSchedule makes use of any scheduling necessary. As the scheduling in the original CASMonitor is not our research target, we used the simplest Time Slice rotation method of scheduling i.e. RoundRobin.

```
MACHINE   Schedule
SEES
      VCUPCtx,DomainCtx,KernelInormation
EXTENDS
      VCPU,Domain
OPERATIONS
      vcpu<--selectNext = PRE squen /= <>
       THEN vcpu:=first(squen) || changeSquen
                          END;
      doSchedule( vcpu )=
      PRE   vcpu: created_vcpu
       THEN
              IF vcpu_slice(vcpu)>0
              THEN decSlice(vcpu)
         ELSE setFullSlice(vcpu)
       END
  END
END
```

The logic behind the RoundRobin scheduling is that in determining which VCPU to operate, operating time is decreased for every definite time slice (fullSlices amount of time slices) allocated by each VCPU. Hence if each VCPU still has time slices, it should be decreased by 1. If there are no time slices (time slice = 0), then it should be given a complete times slice (fullSlices) and put up. A new VCPU with time slice not equal 0 should be selected and put into schedule.

D. Grant Table

The capabilities of the grant table are accomplished via Hypercall. When Hypercall occurs, it will search for the function in the Hypercall list and apply the function. Hypercall's capabilities are in abstract machine model GrantTable, of which the Hypercall function is doGrantTableOp. Creating a grant table is done by the function setupGrantTable. As such operations are completed in the kernel mode, it cannot directly operate user data but must instead convert the user data application's grant table address into a physical address, then create a grant table. After creating a grant table, all grants should be null, and must be authorized by the domain before usage.

E. Event Channel

The event channel is a notice mechanism, similar to the signal in the UNIX system. Each domain has multiple event channel ports which are used to receive events. Events could be created by other domains or the CASMonitor. A domain can actively check for events or it can put in place an event function such that the CASMonitor will use this function when an event takes place.

The capabilities of the event channel are also completed using Hypercall. When Hypercall occurs, it will search for the function in the Hypercall list and apply the function. The grant table's capabilities exist in the abstract machine model EventChannel, of which the Hypercall function is doEventChannelOp. doEventChannelOp will apply the corresponding Hypercall function according to the order numbers of the parameters.

F. Mandatory Access Control

Mandatory Access Control is realized by refining security policy model. As of now CASMonitor mainly actualizes the access control strategies of Chinese Wall Policy. This is mainly used in decision controls before virtual CPU creation.

VCPU create and abort operations are mainly actualized using Chinese Wall Policy. The aim of this policy is to actualize conflicting VCPUs that may not operate on the same platform.

Controlling VCPU create operation is a refinement of the policy, of which

$$!(nn).((nn : conflictAggregateSetr) =>$$

$$(!(mm).((mm : created\_domainr)) =>$$

$$(ssidrefsr(mm)/ : nn \text{ or } t\_ssidrefs / : nn)))$$

Criteria are used to decide if Chinese Wall Policy allows the creation of a VCPU; this criteria is used to execute the decision-making.

IV. Experiment

Correspondence Analysis method is derived from the Event-B tool that performs refinement testing and traces refinement testing on two refined levels (ChineseWallPolicy model & FTLS). The entire analytical process is divided into parallel multi-level simulation and conformance testing.

A. Multiple levels of refinement

In order to check the simulation relation between the abstract levels and the specific levels, we use Event-B to carry out parallel multi-level simulation. The model is first initialized and can be divided into two stages: the initialization of a constant and the initialization of a variable. After initiation is complete, we can then proceed

with the simulation of the model and observe apparent state changes and state transitions. Finally, using model checking, we apply conformance testing to verify the validity of the model.

To simulate the strict correspondence between FTLS & ChineseWallPolicy model, we simultaneously initiate both models and conduct parallel multi-level simulation to verify correspondence between events. Parallel multi-level simulation can also utilize multi-level animation to intuitively check the validity of specifications and the synchronous operation of the two layers.

Fig. 4, 5 shows the multi-level animation details:

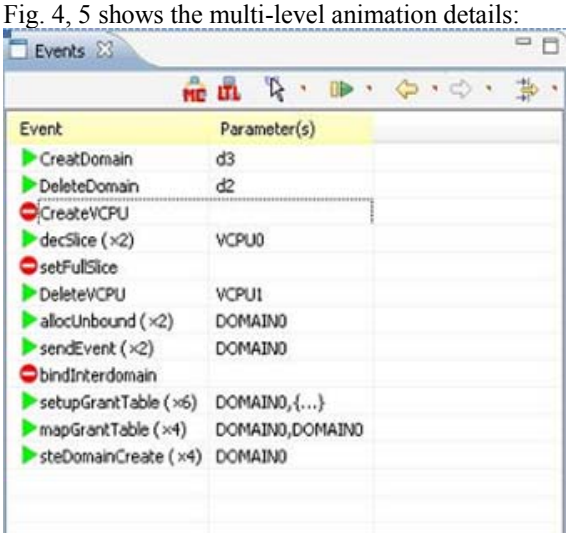

Figure 4. Events in multi-level animation

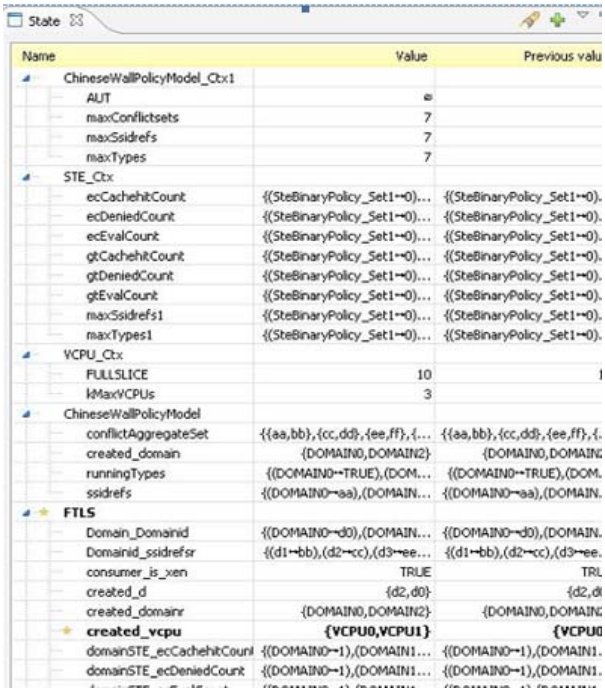

Figure 5. States in multi-level animation

B. *Correspondence analysis*

In the correspondence analysis of secure policy and FLTS, requested operations also possess a corresponding consistency. Of which, the initialization in FLTS is inherited from the initialization in Chinese Wall Policy Model. Events such as CreatDomain and DeleteDomain in FTLS are refinements of CreatDomain and DeleteDomain in Chinese Wall Policy model. In refining, invariants are used to describe the correspondence between abstract and specific variables. By simplifying parameters of events to reduce uncertainty, Witnesses could be used to simplify correspondence between descriptive parameters and to ensure consistency between the model and FTLS. Several other events are new additions to FLTS and they are more specific and detailed, such as VCPU, Event-Channel, Grant-table etc.

The refinement of correspondence between events of the Chinese Wall Policy Model and FTLS are shown as follows:

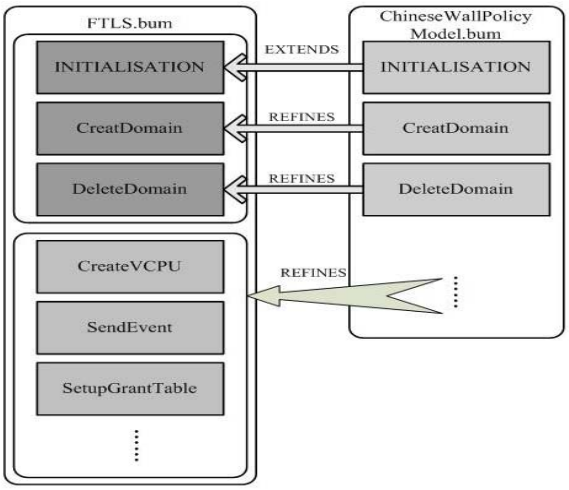

Figure 6. Correspondence analysis

C. *Consistency Verification*

Through simultaneous parallel multi-level simulation, we use model checking technology to conduct consistency tests on the security model. We can clearly explain that the state transitions within the security model strictly satisfy the invariant limits present. Hence the testing results show that the security model is reliable and accurate.

We have gone through a series of parallel multi-level simulation and directly observed the effectiveness of the statutes. Through correspondence analysis, It is strictly semantics that the rigorous analysis and consistency of the corresponding validation logic with ChineseWallPolicy and FTLS. Finally, we conduct comprehensive consistency testing on FTLS and the results are shown in Fig. 7.

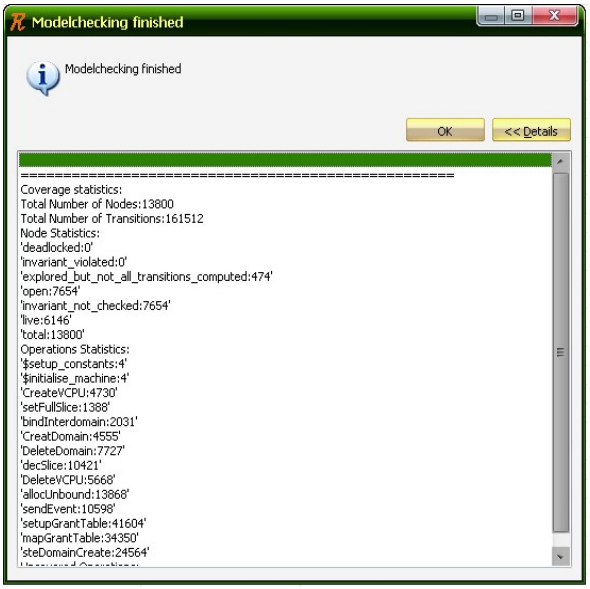

Figure 7. FTLS consistency verfication

V. CONCLUSION

In this paper, we use B method to verify the design of CASMonitor. We firstly give a Chinese Wall model, and secondly provide a corresponding Formal Top-Level Specification. As a secure and high-assurance platform, classical part of an operation system is needed. Moreover, we introduce the Mandatory Access Control module into our model, which is important to a high-assurance system.

Our experiment also shows an effective method to development of micro kernel. Further research work will focus on more complex and accuracy high-assurance model. In addition, it is reasonable to support our operating system development in future.

REFERENCES

[1] Brewer D, Nash M. The Chinese Wall Security Policy[C]//Proc. of the IEEE Symposium on Security and Privacy. IEEE Computer Society Press, 1989: 206-214.

[2] Department of Defense. Trusted Computer System Evaluation Criteria(TCSEC). DoD 5200.28-STD.

[3] Steve Schneider, Palgrave. *The B-Method: An Introduction*, Cornerstones of Computing series, 2001.

[4] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. L. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the art of virtualization. In *ACM Symposium on Operating Systems Principles*, pages 164–177, 2003.

[5] B. Z. Wu. Casmonitor: A os monitor based on hypervisor framework. Master's thesis, Graduate University of Chinese Acadmy of Sciences. Beijing, China, 2010.