Hades高可信架构中分区间信息流控制的研究

杨 霞[1]，古和亦[2]，汪 强[3]，桑 楠[1]，熊光泽[1]

(1. 电子科技大学计算机科学与工程学院 成都 611731；2. 中国人民武装警察部队警官学院 成都 610213
3. 许继昌南通信设备有限公司 河南 许昌 461000)

【摘要】为保障Hades高可信嵌入式操作系统体系架构自身的可信性，必须对其不同安全等级分区之间的信息交互进行严格的控制。该文分析了现有采用分区机制的可信系统的信息流控制方法的不足，提出了一种信息流控制模型，建立了信息流控制机制，并通过实验原型进一步验证了该信息流控制技术的正确性和有效性。

关 键 词 BLP安全模型； 高可信嵌入式操作系统体系架构； 信息流控制； 多级安全； 可信分离内核
中图分类号 TP302.8 文献标识码 A doi:10.3969/j.issn.1001-0548.2012.01.015

Inter-Partition Information Flow Control in High-Trusted Architecture Hades

YANG Xia[1], GU He-yi[2], WANG Qiang[3], SANG Nan[1], and XIONG Guang-ze[1]

(1. School of Computer Science and Engineering, University of Electronic Science and Technology of China Chengdu 611731；
2. Police Officer College, Chinese People's Armed Police Force Chengdu 610213;
3. XJ Changnan Communication Equipment Co. Ltd Xuchang Henan 461000)

Abstract To assure the trustworthiness of high-trusted architecture of embedded operating system that named Hades, the inter-partition information flow of Hades must be controlled strictly. By analyzing the limitation of existing methods, we present a new information flow control model and construct a information flow mechanism between partitions. The accuracy of these model and mechanism are verified through the experiment.

Key words BLP security model; high-trusted architecture of embedded operating system; information flow control; multi-level security; trusted separation kernel

随着计算机技术在嵌入式安全关键系统(embedded security/safety-critical systems，ESCS)中的广泛应用以及ESCS越来越复杂和庞大，系统的高可信问题逐渐突出，日益受到广泛关注。其中ESCS是指系统功能一旦失效将危及人的生命和财产的嵌入式系统，这类系统广泛存在于航空航天、国防、交通运输、核电能源和医疗卫生等诸多领域中[1]。据统计资料表明，随着软件技术在ESCS中的大量使用，软件的故障、失效以及安全泄密逐渐成为引发ESCS灾难性事故的主要根源[2]，因而ESCS高可信问题的重点在于其所使用的安全关键软件的防危性、安全性、实时性。另外，在一些复杂的航空、军事等安全关键系统中，存在多个安全等级(multi-level security，MLS)的应用，并且这些系统要求达到CC(common criteria)EAL4及以上的较高安全认证[3]。因此有关复杂ESCS的高可信保障机制已经成为开发商和用户关注的焦点，是学术界和商界亟待研究解决的热点课题。

文献[4]提出了一种基于MLS的高可信嵌入式操作系统体系架构(Hades)，该架构针对现有可信保障技术和方法的弊端，从嵌入式操作系统体系结构出发提高了基于MLS的ESCS的安全性、防危性、实时性。该架构采用“时空隔离”思想将CPU和内存资源根据不同的安全等级分离成多个相对独立的区域，通过这种分离可以将故障和不可信信息与其他可信区域隔离，从而提高整个系统的可信性。

Hades架构各分区子系统虽然相对独立，但分区之间不是孤立的，完全独立也是不现实的，因为它们需要信息和数据的共享和交互。为了保障整个系统自身的高可信能力，必须保证分区间信息和数据

收稿日期：2010－04－06；收回日期：2011－03－02
基金项目：国家863计划(2007AA01Z131)
作者简介：杨霞(1978－)，女，博士生，主要从事嵌入式系统可信计算方面的研究.

通信过程是绝对可信赖的。Hades架构允许在分区间进行信息和数据的共享与交互，但仅允许受限访问，也就是说分区间信息和数据的通信必须在严格的控制之下进行，否则将会导致不可信的信息和数据在分区之间流动从而影响整个系统的可信性。另外，由于每个分区存在不同的安全等级，对不同安全等级分区间信息和数据的访问控制应更加严格，稍有疏忽将可能破坏高安全等级分区信息以及和数据的机密性和完整性，从而导致整个系统提供的服务是“不可信赖”的。由此可见，建立严格的信息流控制机制对于Hades架构至关重要。

本文重点研究了不同安全等级分区间信息流控制的方法和技术。首先对其他信息流控制方法，如MILS、sHype及VAX VMM等架构的分区间信息流控制技术及其不足进行分析，然后提出一种简单而严格的信息流控制模型，并基于该模型建立信息流控制机制，最后通过原型实验进一步验证分区间信息流控制方法的正确性和有效性。

1　传统的信息流控制技术及其不足

早期，有些采用分离思想的系统要求分区间绝对分离，绝对禁止分区间的信息和数据共享和交互，该种系统被称为纯数据分离系统，如IBM公司的PR/SM[5]系统。后来，文献[6]提出的分区间安全共享思想可解决分区间资源的共享问题。目前，流行的允许分区间信息和数据共享与通信的安全体系结构有MILS、sHype及VAX VMM等，虽然它们都允许分区之间信息和数据的交互，但各自的通信方式及系统对其控制过程都有所不同。

MILS架构[7]对分区间通信的管理和控制由中间件完成，而中间件技术的引用将使信息流控制机制变得比较庞大和复杂，增加了过多的系统开销。sHype[8]虽然提供了两种分区之间的数据同步、共享和通信机制，但没有对不同安全等级数据之间的交互与共享提供安全保障。VAX VMM[9]结构虽然为不同安全等级的数据建立了访问规则，采用独立可信分区或者安全子系统实现分区间共享文件的存储，其数据共享的思路非常好，但是在具体实现上将增加许多系统开销[10]。

2　多级安全信息流控制模型

Hades架构是基于多级安全关键嵌入式系统的高可信操作系统架构，该架构内设多个分区，每个分区的安全等级都根据其关键度的不同而指定。为满足应用的需要，Hades架构允许分区间可以进行信息和数据的共享与通信，为保障不同安全等级分区子系统之间的数据共享和通信过程不会对系统的可信性产生不良影响，必须对不同分区间的信息流进行严格的控制，以达到下面3个目标：

1) 绝对防止未经授权的用户非法访问系统敏感数据，即防止高级安全分区的信息和数据被泄密；

2) 绝对禁止高级安全分区的信息和数据被不可信分区恶意篡改和破坏；

3) 信息流控制必须尽可能的简单，不能因此增加许多代码以影响系统效率和认证开销。

为达到以上目的，Hades架构以简单的BLP安全模型为基础建立了信息流控制模型。

定义 1 主体由S_i唯一标识，是访问的发起者；客体由O_i标识，是被访问者。主体对客体的访问是信息流控制的目标。访问的方式由访问规则x描述，模型中定义了4种访问方式：

1) r(read)读，只是浏览而没有修改；

2) w(write)写，修改的同时浏览；

3) a(append)追加，只是修改而没有浏览；

4) e(execute)执行，既不是浏览也不是修改。

定义 2 访问集$A=(S_i,O_j,x)$，表示当前状态下主体S_i正在以x访问客体O_j，当前访问集合是所有当前访问操作请求的集合。

目前，Hades架构仅允许某个分区对其他分区发出读、写访问请求，因此信息流控制只考虑主体对客体的读、写访问，并就此制定分区间的访问规则。

定义 3 访问规则x规定主体不能读安全级别高于自己的，不能写安全级别低于自己的客体。

例如，子系统OS_1的安全等级为L_1，子系统OS_2的安全等级为L_2，假设安全等级$L_1<L_2$，则OS_1不能读OS_2子系统，并且OS_2不能写OS_1子系统，如图1所示。

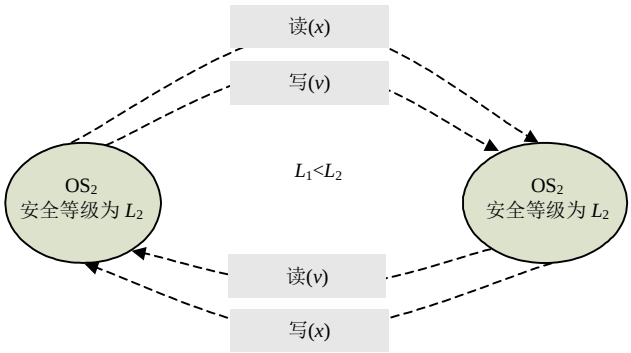

图1　信息流控制模型

BLP安全模型比较简单，仅增加较少的系统开

销就能建立严格的信息流控制策略，对Hades架构实现短小精炼的可信分离内核(trusted separation kernel，TSK)，并使其能够通过较高安全等级认证是有好处的。

3 信息流控制机制

分区间所有对信息和数据的访问请求，首先由用户模式中的可信服务分区[4]和可信监控器监控，并且所有的访问请求最终受TSK授权和控制，只有得到了TSK的授权，该访问请求才能被允许执行。将发出访问请求的分区称为主体，被访问的目标分区作为客体，主体对客体的所有访问请求均在严格控制之下，并且分区间所有的信息流都必须经受TSK的授权。

为满足分区间访问控制的需要，本文建立了一种仅允许授权信息和数据在分区间交互的信息流控制机制(information fow control，IFC)。IFC覆盖了Hades架构的每个层次，由可信服务分区、可信监控器、访问控制模块(access control module，ACM)和BLP模型4部分组成，如图2所示。可信服务分区是一个专门为系统提供可信服务的分区，可信监控器位于用户模式的“客户”OS中，而ACM和BLP位于TSK中。为确保系统本身的可信性，IFC中各模块必须为可信的，如可信服务分区和可信监控器等模块自身的可信性必须通过可信验证器的验证。

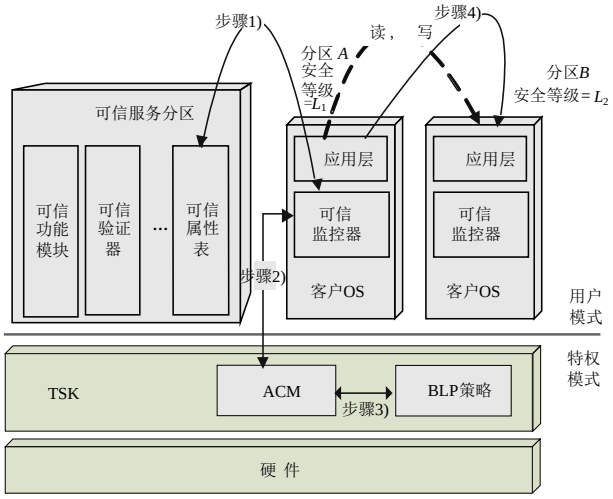

图2 信息流控制机制

下面分别介绍IFC机制中各模块的工作内容：

3.1 可信服务分区

可信服务分区作为用户模式中最重要的模块，包含对可信验证器、分区可信属性表、可信功能模块的管理，为Hades架构提供用户级的可信服务。

1) 可信验证器：用于检验分区和其它系统模块的可信度，并根据检验结果为被检验的分区和系统模块打上“可信”和“不可信”标签。

2) 分区可信属性表：用于存放分区的安全等级、可信度等可信相关信息，作为分区的“可信标识”与分区一一对应。

3) 可信功能模块：安全、防危、可靠等各分区的公共可信服务功能模块。

3.2 可信监控器

IFC机制在每个分区子系统中安置可信监控器模块，用于控制和管理分区之间的所有信息流，负责分析分区间通信的内容，并判定该通信是否符合系统可信策略的要求。可信监控器必须是能够通过可信验证器检验的可信模块，即说必须确保可信监控器自身是可信赖的。

3.3 ACM和BLP

不但要防止非授权信息和数据向其他分区的流动和向TSK中增加过多的开销。因此，Hades架构在TSK中仅定义简单而严格的可信机制，还要绝对禁止低安全等级分区读取高安全等级分区的数据和高安全等级分区向低安全等级分区写入数据的情况发生，并设计ACM和BLP访问控制策略用于分区间的信息流控制，除此之外，TSK中没有任何可信相关措施。

Hades架构正是通过IFC机制中各模块之间的协调工作实现对信息流的严格控制的。下面将结合实例详细描述该控制过程。

假设有分区A和分区B，安全等级分别为L_1、L_2。由于向分区中写数据和从分区中读数据的信息流控制过程类似，只是控制策略不同，本文仅以某个分区写信息和数据到其他分区的操作为例，说明分区间信息流的控制过程。分区间信息流控制过程分为4个步骤，如图2所示。

1) 当分区A向分区B发送数据时，首先由分区A的可信监控器从分区可信属性表中获取分区A和B的安全等级和可信度等属性，然后根据分区A和B的可信属性分别进行如下的处理：

(1) 分区A、B均为“可信”分区，意味着分区A无论发送什么数据到B都不会给B带来安全隐患，但由于A和B可能存在不同的安全等级，必须根据BLP安全模型判断是否允许该操作的执行。假设L_1=L_2，允许A发送数据到B，并且由A的可信监控器将数据从A发送到B; 如果$L_1 \neq L_2$，A的可信监控器将请求TSK判断是否允许该操作的执行。

(2) 分区A或者B为“不可信”分区，表示绝对

禁止不可信分区的信息和数据扩散到其他分区，并防止不可信分区获得可信分区的信息和数据，因此必须禁止分区*A*对于*B*的所有操作请求，*A*的可信监控器立即返回，操作结束。

2) 当*A*为可信分区并且$L_1 \neq L_2$时，*A*的可信监控器向TSK发出是否允许该操作请求的决议。

3) TSK的ACM根据BLP安全策略对访问请求做出仲裁，针对下面两种情况分别做出决策：

(1) $L_1<L_2$即*A*的安全等级低于*B*分区，根据BLP安全策略允许数据从*A*发送到*B*，因此ACM模块做出“允许”访问的决策，并将该授权反馈给*A*分区的可信监控器。

(2) $L_1>L_2$即*A*的安全等级高于*B*分区，此时，根据BLP安全策略绝对禁止数据从*A*流向*B*，因此ACM做出“禁止”访问的决策，并将该决策返回给*A*分区的可信监控器。

4) *A*的可信监控器根据TSK返回的仲裁结果进行最后的处理，如果结果为“允许”，则将数据发送给*B*，并返回“成功”的标识给应用程序，结束操作；否则，立即返回“禁止”标识给应用程序，结束操作。

至此，Hades架构的信息流控制过程全部结束。Hades架构的IFC机制具有过程简单、控制严格、开销较小等优点。

4　原型实验

为了验证本文所提出的信息流控制模型和IFC机制的有效性，本文建立了一种实验原型，并在下述实验平台上测试运行。

4.1　实验硬件平台介绍

为了对分区机制和时空隔离思想提供硬件支持，必须采用能够支持内存管理单元(memory management unit，MMU)的CPU。因此，本文实验选择基于ARM920T内核的ARM2440开发板作为实验的硬件平台，其具体配置如表1所示。

表1　实验硬件环境配置

CPU	内存/MB	外存	显示设备	电源/V
32位ARM920T 内核400 MHz	256	64 MB Flash 2 G的SD卡	3.5寸TFT 真彩液晶屏	5

4.2　实验软件平台介绍

本文跟据实际需要对具有自主知识产权的国产嵌入式实时操作系统CRTOS内核进行修改，并将修改后的CRTOS内核作为Hades架构的可信分离内核作为实验的软件平台。

为了满足实验的需要，首先修改CRTOS内核使其支持MMU，并创建5个应用分区。为了给分区创建独立而完整的硬件环境，使其成为一个可以安装“客户”OS的分区子系统，在每个分区中安装Xen虚拟机，通过建立虚拟的硬件环境，使“客户”OS认为自己在独享计算机资源。实验中，为每个分区都指定不同的安全等级和可信度，信息由分区可信属性表统一管理，如表2所示。分区安全等级的数字越大，表明安全等级越高。

表2　分区可信属性表

分区	安全等级(SL)	可信度
P_1	1	可信
P_2	2	可信
P_3	3	不可信
P_4	3	可信
P_5	4	可信

在5个分区子系统中分别安装嵌入式Linux、Win CE、Vxworks和RT-Linux内核，实验软件平台体系结构如图3所示。

4.3　实验内容

为实验需要，在每个分区设立标识、安全等级、可信度等属性。每个分区都有一个名称和ID号作为分区标识，并且二者一一对应，不允许重复。分区安全等级可以是0～255的无符号整数，可信度只有“可信”和“不可信”两种情况。下面定义分区可信属性的数据结构：

```
Struct Partition_security_struct {
string partition_name; /*分区名称*/
integer partition_id; /*分区ID*/
undefined int security_level; /*安全等级*/
Boolean partition_trusted; /*可信度*/
}
```

为了验证IFC机制是否有效，实验中设计的5个不同安全等级的分区相互发出读、写操作请求，发出请求的分区作为主体分区，目标分区作为客体分区，如图4所示。为了实验方便，在每个分区中创建读、写两个任务负责发出访问请求。

为了使分区间所有的信息流必须在CRTOS 内核的监控下，CRTOS中ACM模块的核心算法如下：

```
Access_Control_Module(
  Partition_security_struct subject, 主体安全属性,
  Partition_security_struct object, 客体安全属性,
  char   requested_mode, 访问操作请求
```

boolean allowed) 输出的访问仲裁结果

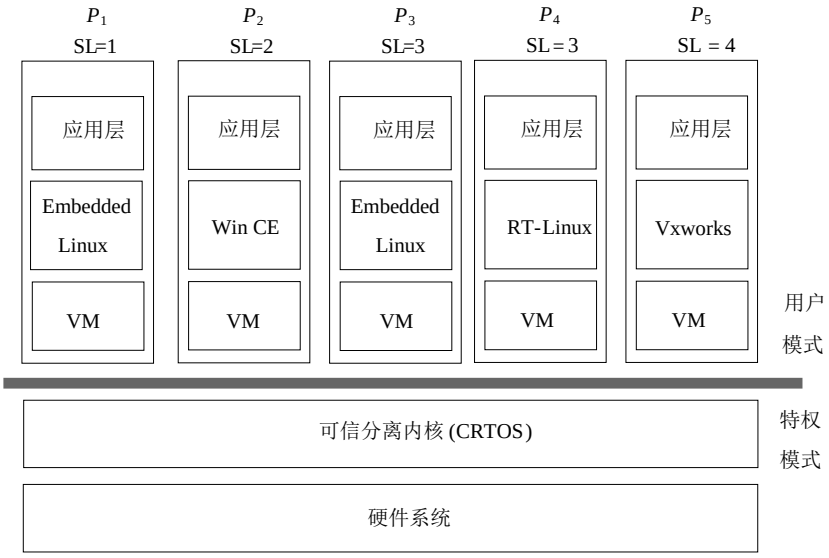

图3 实验平台体系结构

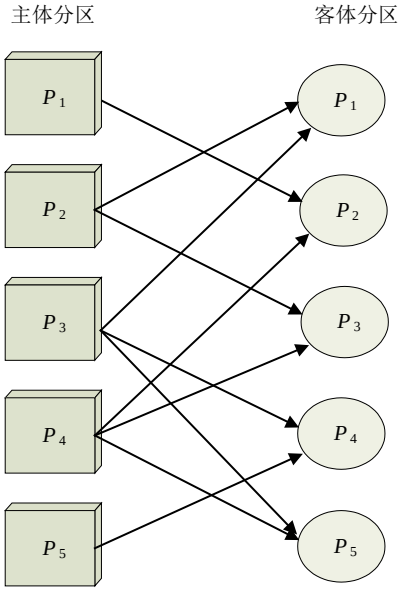

图4 分区间的访问请求

if((subject.partition_trusted!==true)or(object.partition_trusted!==true))//如果主体分区或者客体分区为不可信分区

allowed=false; //对于不可信分区，不管为主体还是客体，所有操作请求都被禁止

return;

if(requested_mode==w) //主体对客体的操作请求为读(write)

if(subject.security_level<=object.security_level) //主体安全等级小于等于客体安全等级

allowed=true; //该操作请求(write)被允许

else allowed=false; //否则请求被禁止

if (requested_mode==r) //主体对客体的操作请求为读(read)

if(subject.security_level>=object.security_level) //主体安全等级大于等于客体安全等级

allowed=true; //该操作请求(read)被允许

else allowed=false; //否则请求被禁止

return allowed;

将上面设计的实验内容和算法在实验平台上运行，可以得到由CRTOS内核对分区间的访问请求所做出的仲裁结果，如表3所示。实验结果显示没有“绕道”而行的分区间信息流，即实验中分区间所有的信息流都在IFC的严格控制之中。

表3 各分区之间相互访问的实验结果

主体	客体	操作请求	仲裁结果
P_1	P_2	读、写	允许写，禁止读
P_2	P_1	读、写	允许读，禁止写
	P_3	读、写	允许写，禁止读
P_3	P_1	读、写	禁止读、写
	P_4	读、写	禁止读、写
	P_5	读、写	禁止读、写
P_4	P_2	读、写	允许读，禁止写
	P_3	读、写	禁止读、写
	P_5	读、写	允许写，禁止读
P_5	P_4	读、写	允许读，禁止写

从表3可看出，实验结果符合预期，由此说明

Hades架构的IFC机制对于分区间的访问控制是有效的。

4.4 系统开销测试

为了使CRTOS内核能够通过较高安全等级认证，不允许向其中增加过多的开销。IFC机制要求分区间所有的信息流必须得到CRTOS内核的授权，因此在CRTOS中仅添加了ACM模块，将增加内核的开销，如果开销过大，将会影响CRTOS通过安全认证。于是，IFC机制对CRTOS的系统开销将是性能评测的重点。

本文在表1所示的硬件平台上，对CRTOS的信息流控制进行了系统开销测试。测试结果显示IFC机制仅向CRTOS内核增加了0.1 KB的开销，在允许范围内。由此说明，CRTOS不会因为IFC机制而加大安全认证的难度。

5 总　结

Hades 架构允许不同安全等级分区之间进行信息和数据的通信，但为了保证各分区信息和数据的机密性和完整性，应严格防止敏感数据的非法流出、被篡改和破坏，并禁止“脏”数据向其他可信分区的流入。为达到此目的，对分区之间的信息流必须进行严格的控制和管理。本文首先对其他安全系统的分区间信息共享和通信机制进行了分析和对比，针对不足建立基于 BLP 安全模型的访问控制模型和严格的信息流控制机制，设计并实现了实验原型。通过实验说明了 IFC 机制对不同安全等级分区间的信息流能够达到绝对严格的控制，并且仅向可信分离内核中增加非常少量的代码，不会因此影响可信分离内核通过较高安全级别的认证。因此，本文所提出的 IFC 模型和机制对保障 Hades 高可信嵌入式操作系统架构自身的可信性具有重要的意义和作用。

参 考 文 献

[1] KNIGHT J C. Safety critical systems: challenges and directions[C]//Tthe 24th International Conference on Software Engineering. Orlando: Julian Krugman, 2002: 547-550.

[2] 杨仕平, 桑楠, 熊光泽. 安全关键软件的防危性测评技术研究[J].计算机学报, 2004, 27(4): 442-450.
YANG Shi-ping, SANG Nan, XIONG Guang-ze. Research on safety testing and evaluation technology of safety critical software[J]. Chinese Journal of Computers, 2004, 27(4): 442-450.

[3] THOMAS P A, ARGER J K. Multi-level security requirements for hypervisors[C]//Processing of IEEEE ACSAC. Tucson: IEEE Press, 2004: 129-134.

[4] YANG Xia, ZHAO Xiang-yu, LEI Jian, et al. A trusted architecture for ESCS with MLS[C]//The Fifth International Conference on Embedded Software and Systems (ICESS 2008). Chengdu: IEEE Press, 2008: 44-49.

[5] IBM Inc. Certification report for processor resource/system manager (PR/SM) for the IBM eServer zSeries 900[J/OL][2003-12-18]. http://www.commoncciteriaportal.org/files/epfiles/0178b.pdf.

[6] MADNICK S E, DONOVAN J J. Application and analysis of the virtual machine approach to information system security[C]// Proceedings of the ACM SIDGARCHSIDGOPS Workshop on Virtual Computer Systems. New York : ACM, 1973: 210-224.

[7] KEVIN G W. Safety kernel enforcement of software safety policies[D]. Virginia: University of Virginia, 1995.

[8] SAILER T J R, VALDEZ E, CACERES R, et al. Building a mac-based security architecture for the Xen open-source hypervisor[C]//Proceedings of the 21st Annual Computer Security Applications Conference. Washington: IEEE Press, 2005: 140-148.

[9] KARGER P A Z, BONIN M E, MASO D W N, et al. A retrospective on the VAX VMM security kernel[J]. IEEE Transactions on Software Engineering, 1991, 17(11): 1147-1165.

[10] 毛韡锋. 四级安全操作系统SECOS的研究与实现[D]. 浙江: 浙江大学, 2006.
MAO Hua-feng. Research on forth-level secure operating systems SECOS[D]. Zhe jiang: Zhejiang University, 2006.

编　辑　蒋　晓