一个安全标记公共框架的设计与实现*

梁洪亮+, 孙玉芳, 赵庆松, 张相锋, 孙 波

(中国科学院 软件研究所,北京 100080)

Design and Implementation of a Security Label Common Framework

LIANG Hong-Liang+, SUN Yu-Fang, ZHAO Qing-Song, ZHANG Xiang-Feng, SUN Bo

(Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China)

+ Corresponding author: Phn: 86-10-62544129 ext 17, Fax: 86-10-62645414, E-mail: hollace@sonata.iscas.ac.cn

http://www.sonata.iscas.ac.cn

Received 2002-01-31; Accepted 2002-04-11

Liang HL, Sun YF, Zhao QS, Zhang XF, Sun B. Design and implementation of a security label common framework. *Journal of Software*, 2003,14(3):547~552.

Abstract: Labels are the foundation for implementing multilevel systems and the prerequisite of enforcing mandatory access control in secure systems. How to define and enforce label functions which support multiple security policies is the focus here. A security label common framework (SLCF) based on static object label and dynamic subject label is put forward. SLCF introduces the notation of access history and provides a complete label funtions set. Based on SLCF, both multilevel confidential policy and multilevel integrity policy can be expressed and enforced. SLCF is implemented in a secure operating system based on Linux, the experimental results show that the system based on SLCF is flexible and practicable.

Key words: label framework; multilevel secure system; information flow control; confidentiality; integrity; secure operating system

摘 要: 标记是实现多级安全系统的基础,实施强制访问控制的前提.如何确定和实现标记功能并使其支持多种安全政策是研究的目的.提出了一个安全标记公共框架,该框架基于静态客体标记和动态主体标记,引入了访问历史的概念,并给出了一个完备的标记函数集合.基于此框架,既可以实施多等级保密性安全政策,又可以实施多等级完整性安全政策.该框架在一个基于Linux的安全操作系统中的实现结果表明,基于该框架的安全系统在保证安全性的同时,还具有相当的灵活性和实用性.

关键词: 标记;多级安全系统;信息流控制;保密性;完整性;安全操作系统

中图法分类号: TP309 **文献标识码**: A

在构建一个多等级安全(MLS)系统时,首先要对系统中的主体(如用户和进程)和客体(如文件、设备等资源)

* Supported by the National Natural Science Foundation of China under Grant No.60073022 (国家自然科学基金); the National High Technology Development 863 Program of China under Grant No.863-306-ZD12-14-2 (国家863高科技发展计划); the Knowledge Innovation Engineering Program of the Chinese Academy of Sciences under Grant No.KGCX1-09 (中国科学院知识创新工程)

第一作者简介: **梁洪亮**(1972年－),男,山东济南人,博士,助理研究员,主要研究领域为信息安全,系统软件.

进行标记[1],以指明它们的安全属性(如安全等级和类别),然后在此基础上实现强制访问控制政策[2].可以说,标记功能是实现多级安全系统的基础.但是在一个多等级安全系统中,标记究竟应该含有哪些内容?主客体的标记应该如何维护?这些问题至今还未能得到很好的解决.我们在分析和研究以往安全系统和安全标准以后,提出了一个安全标记公共框架,指明了标记功能应该包含的内容,并给出使用此标记框架实现多等级保密性安全政策和多等级完整性安全政策时标记的初始设置及其变化规则.需要指出的是,多等级安全自从在安全学术界提出以来一直是指多等级保密性,随着信息安全研究的发展,现在已经扩展到多等级保密性、完整性和可用性.本文为了保持与以往安全系统描述的一致性,仍保持为多等级保密性的含义.文中使用其他两个含义时将会显式标出.

按照用户会话期间(从登录进入系统到退出系统)主客体的标记是否可以变化,我们可以把以往实现的安全系统分为以下两类:(1) 主体和客体的标记都是固定不变的,如 MITRE Multics[3],IBM Secure Xenix[4]和 NRL MMS[5];(2) 主体和客体的标记都是动态变化的,如 AT&T IX[6].但是这两种方法都存在着局限性,我们在第 1 节中对此进行了详细阐述.第 2 节提出了一个安全标记公共框架(SLCF),探讨了标记功能应该包含的内容,并给出了实现多等级系统时主体标记的变化规则.该标记框架基于固定的客体标记和动态可变的主体当前标记.我们在第 3 节采用 SLCF(security label common framework),通过设置不同的标记初始值和构造不同的主体标记变化规则,证明了可以应用 SLCF 实现多等级系统完整性.在第 4 节中对该标记框架在 RS-Linux 的实现进行了简述,并描述了它所能满足和实现的国际信息安全评估准则中的功能要求.最后指出了标记框架下一步的发展方向,最后是小结.

1 问题的提出

在多等级安全系统中,实体(包括主体和客体)的标记包括两部分:等级分类和非等级类别.S 是主体集合(包括用户和进程等),O 是客体集合(包括文件、目录、设备等),A 是访问属性集合,G 是等级分类集合,C 是非等级类别集合,W 是会话集合,T 代表时间集合,P 表示幂集关系,H 为客体的层次(包含)关系,$H(p,o)$表示客体 p 是 o 的父客体.

当前访问集合 $B:=P(S\times O\times A)$.标记集合 $L:=\{(G,C)|G\in G\wedge C\in C\}$.

标记的支配函数≥:$(L_1=(G_1,C_1)\in L\wedge L_2=(G_2,C_2)\in L),L_1\geq L_2\rightarrow(G_1>=G_2\wedge C_1\supseteq C_2)$.

标记函数集合 $F:=\{(f_s,f_o,f_c)|f_s\in L^S\wedge f_o\wedge L^O\wedge f_c\in L^S\wedge(\forall S\in S(f_s(S)\geq f_c(S)))$.其中,$f_s$ 称为主体最大标记函数,f_o 称为客体标记函数,f_c 称为主体当前标记函数.并且,主体当前标记在任何时候都不会超过主体最大标记.

在文献[3]中,f_s 的值由安全管理员指定.在用户开始一个会话 w 时,用户可以选择一个不超过 f_s 的值作为 f_c,若不指定,缺省采用 f_s 作为 f_c.这样,在整个会话期内,f_c 的值一直是固定不变的,主体创建的所有非父客体(如文件)的标记都等于 f_c,并不低于该客体的父(包含)客体(如目录).主体创建的所有父客体的标记都不低于 f_c,并不低于该客体的父客体.设 t_s 为会话 w 的开始时间,t_e 为会话 w 的结束时间.

对于 $t_s<t_1<t_2<...<t_e$,下列式子成立:

$f_s\geq f_c(t_e)=...=f_c(t_2)=f_c(t_1)=f_c(t_s)$;

$f_o=f_c$,其中 o 为非父客体;

$f_o\geq f_p\geq f_c$,其中 o 为父客体,并且有 $H(p,o)$.

尽管这个标记方案简单明了,但是在实际使用过程中发现仍存在几个问题:(问题 1)实用性较差,因为一个用户或进程在一次会话期内可能需要访问不同级别的客体[5,7].(问题 2)这样会造成不恰当的客体标记,例如,一个秘密用户所写的午餐订单都会是秘密级别.(问题 3)如果用户创建一个升级客体(标记支配父客体的客体,例如,假设标记为 i 的用户要在标记为 j 的目录 d_j 下创建一个标记为 i 的目录 $d_i,i>j$),因为 $f_c\geq f_{dj}$,在 BLP 模型[8]下禁止向下写,则他必须退出系统,然后以标记为 j 的身份登录进入系统,在目录 d_j 下创建目录 d_i,然后再退出系统,重新以标记 i 登录,才可以使用目录 d_i 进行读写.可以看出,这种不便是固定标记方法所固有的缺点造成的.

在 AT&T IX 系统中,设计者们注意到了不应该把一个多等级安全系统简单地看成是一个静态的主体/客体模型,而应该看做是一个动态的信息流模型.基于此,Mcilroy 等人提出了与 Secure Xenix 不同的标记方案.他们

认为,主体和客体的标记都有一个上界阈值,在一次会话期间,主体和客体的标记是随着时间而变化的,并且不会超过它们的上界阈值.f 表示标记函数是随时间变化的函数,U 是随时间变化的上界标记函数.对于一次从源 x 流向目标 y 的信息流,以下规则成立:$f_y(t)\geq f_x(t), U_y(t)\geq f_y(t), U_x(t)\geq f_x(t)$.

这种标记方法使得主体在一次会话期间可以访问不同级别的客体,也不存在升级客体的创建问题.但是实际结果证明,这种方法严重影响了系统的性能.因为在客体标记不变的情况下,对客体访问时只需要在打开客体时进行标记检查,而在允许客体标记动态变化的情况下,对客体的每一种访问请求都需要进行标记检查.这在具有大量文件的系统中对系统性能的影响非常大.而且在 IX 实现这种方法时,必须要依赖一个特权服务器和私有通信路径(问题 4).另外,允许客体标记可以变化会造成标记不当(问题 5).例如,一个上级在下级递交的某份报告上签署意见(如“同意”或“反对”)后,下级就无法打开这份报告了.因为此时报告的标记已经升高了.

可以看出,在实施多等级安全系统时,采用全静态的标记方法和全动态的标记方法都存在一些局限.下一节我们将描述一个新的标记框架,可以很好地解决这些问题.

2 一个安全标记公共框架(SLCF)

在一次会话过程中,主体是代表用户执行动作的进程,进程在系统中是变化的,依照用户或系统的要求执行各种操作,而文件等客体是静态的和被动的.另外,进程是有“生命”的,每个进程都有自己的生存周期,按照所完成的不同任务,“生命”或长或短.但是无论长短,系统都要保证进程不会执行不符合安全需求的信息流.基于此,我们提出了一个新的标记框架,其中主体的当前标记是动态变化的,客体的标记是固定的.并且随着信息的流入(如读操作)和流出(如写操作)而“记忆”(存储)主体的标记变化.

新的标记框架同样包括主体、客体、标记、访问集合等概念,除标记函数集合以外,其他概念与第 1 节所述相同,此处不再重复.新的标记函数集合除了$\{f_s, f_o, f_c\}$以外,

(1) 为主体增加了 4 个标记函数$(f_{il}, f_{ih}, f_{ol}, f_{oh})$,它们分别表示在一个进程的生命周期内,流入信息的最低标记、流入信息的最高标记、流出信息的最低标记、流出信息的最高标记.

(2) 为客体增加了一个标记函数 f_d,f_d 把客体标记 f_o 映射为易于理解的信息(一般是一组字符串),用于客体信息的输出(如显示或发布信息).

下面我们描述如何使用 SLCF 实现多等级保密性安全政策.首先,我们给出初始状态时标记的初始值,$f_{il}=f_{ih}$=LOW(系统的最小标记值),$f_{ol}=f_{oh}$=HIGH(系统的最大标记值).接下来我们用与 C 语言相似的语法来描述在多等级保密性安全政策下,信息流动时所需的安全判定条件以及主体标记的变化规则.为了方便描述,假设信息源的标记值表示为 f_o,$T(o,s)$=true 表示允许信息从 o 流入 s,$T(o,s)$=false 表示禁止信息从 o 流入 s.

(1) 在信息从客体 o 流入主体 s 时(如 s 读取 o),判定条件及主体标记的变化规则(规则 1)如下:

IF ($f_c\geq f_o$) THEN $T(o,s)$=true;

ELSE IF ($f_s\geq f_o$) && ($f_{ol}\geq f_o$) THEN {f_c=Max(f_c,f_o), f_{ih}=Max(f_{ih},f_o), $T(o,s)$=*true*};

ELSE $T(o,s)$=false

(2) 在信息从主体 s 流入客体 o 时(如 s 写入 o),判定条件及主体标记的变化规则(规则 2)如下:

IF ($f_o\geq f_c$) THEN $T(s,o)$=true;

ELSE IF ($f_o\geq f_{ih}$) THEN {f_c=Min(f_c,f_o),f_{ol}=Min(f_{ol},f_o),$T(s,o)$=true};

ELSE $T(s,o)$=false

(3) 当信息在主体 s 和客体 o 之间双向流动时(如 s 既读 o 又写 o),判定条件及主体标记的变化规则(规则 3)如下:

IF (f_c= =f_o) THEN {$T(o,s)$=true, $T(s,o)$=true};

ELSE IF ($f_s\geq f_o$) && ($f_{ol}\geq f_o$) && ($f_o\geq f_{ih}$) THEN {f_c=f_o,f_{ih}=Max(f_{ih},f_o),f_{ol}=Min(f_{ol},f_o),$T(o,s)$=true,$T(s,o)$=true}

ELSE {$T(o,s)$=false,$T(s,o)$=false}

从上面的变化规则可以看出,信息的流动取决于主体的 6 个标记和客体的标记之间的支配关系,主体标记的变化依赖于它的访问历史(即信息的流向和访问过的客体).并且随着信息的流入,主体的当前标记和主体的

流入信息的最大标记是单调递增的;随着信息的流出,主体的当前标记和主体的流出信息的最小标记是单调递减的.

由于主体的当前标记可以在系统最小标记到主体最大标记的范围内变化,所以主体在同一会话期间可以访问不同级别的客体.更重要的是,主体的流入信息的最大标记(f_{ih})指定了信息能够合法地流出主体的客体标记的下界,主体的流出信息的最小标记(f_{ol})指定了信息能够合法地流入主体的客体标记的上界.只要信息源客体的标记支配主体的流出信息的最小标记(f_{ol}),就是非法的流入信息流;只要主体的流入信息的最大标记(f_{ih})支配信息宿客体的标记,就是非法的流出信息流;只要禁止这两种非法信息流,我们就可以确保不使进程产生向下的信息流,因此就实现了多等级保密性安全政策.

下面我们用一个例子来说明,如图 1 所示,图中有一个进程和 3 个文件,它们的初始标记值见表 1.为了描述简洁起见,我们在表中用一个整型值表示标记值.

因为 f_c(process2)≥f_o(file1),f_{ol}(process2)≥f_o(file1)≥f_{ih}(process2),故允许如图 1 所示的⑤(process2,file1,*r*),⑥(process2,file1,*w*);可见问题 2 在本标记框架中得到了解决.因为 f_o(file3)≥f_c(process2),f_{ol}(process2)≥f_o(file3)≥f_{ih}(process2),故允许如图 1 所示的④(process2,file3,*w*);可见问题 3 在本标记框架中得到了解决.

假设在 t_1 时刻,进程 process2 启动了一个从 file2 到 file3 的信息流,也就是先执行如图 1 所示的①(process2,file2,*r*),然后执行如图 1 所示的④(process2,file3,*w*),根据上面的标记变化规则,变化后的标记值见表 2.可见,一个进程可以在其生命周期内访问不同级别的客体(问题 1 在本标记框架中得到了解决),并且保证了信息的保密性.因为在 t_1 时刻后,process2 只可以读取标记低于 f_{ol}=3 的客体,只可以修改标记高于 f_{ih}=2 的客体.并且无论何时都禁止如图 1 所示的③(process2,file3,*r*).

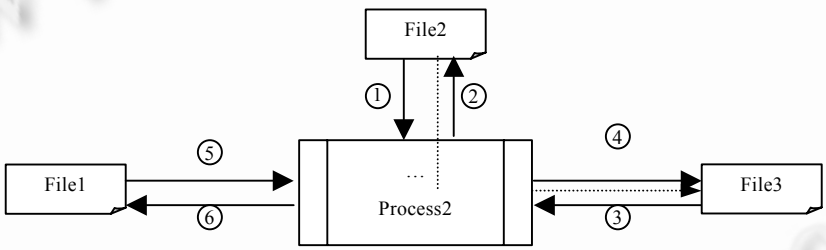

Fig.1 The dynamic changing of subject's labels
图 1 主体标记动态变化示意图

Table 1 Initial values of labels
表 1 初始标记值

Name of entity	Lable	Value of lable
Process2	f_s	2
Process2	f_c	2
Process2	f_{il}	LOW
Process2	f_{ih}	LOW
Process2	f_{ol}	HIGH
Process2	f_{oh}	HIGH
File1	f_o	1
File2	f_o	2
File3	f_o	3

Table 2 Label values after file2→process2→file3
表 2 file2→process2→file3 后的标记值

Name of entity	Lable	Value of lable
Process2	f_s	2
Process2	f_c	2
Process2	f_{il}	LOW
Process2	f_{ih}	2
Process2	f_{ol}	3
Process2	f_{oh}	HIGH
File1	f_o	1
File2	f_o	2
File3	f_o	3

再如,假设进程 process2 是一个执行中的特洛伊木马程序,它在 t_1 时刻启动的是一个从 file2 到 file1 的信息流(即读取文件 file2 的内容并写到文件 file1 中),根据上面的标记变化规则,在如图 1 所示的①(process2,file2,*r*)以后,因为 f_c(process2)≥f_o(file1),f_{ih}(process2)≥f_o(file1),故禁止如图 1 所示的⑥(process2,file1,*w*).

可见,本标记框架可以有效地防止意图泄密的特洛伊木马程序.由于本框架主体的标记变化只依赖于主体的访问历史和固定的客体标记,因此不需要额外的辅助机制(如问题 4 中的特权服务器),也不像 AT&T IX 那样对系统性能造成很大影响.另外,由于采用了静态的客体标记,自然也不会出现前面描述的问题 5.可见,采用本标记框架实施多等级保密性安全政策,很好地解决了以往多等级保密系统中存在的问题.

3 应用 SLCF 实现多等级完整性

由于 SLCF 具有较强的抽象能力,我们可以将其应用于广义的多等级安全系统.下面我们采用 SLCF 中的标记方法,修改标记初始状态和主体的标记变化规则,以实现多等级完整性安全政策.与保密性安全政策禁止向下的信息流相反,完整性安全政策禁止系统中发生向上的信息流.因此设置初始状态下 $f_{il}=f_{ih}$=HIGH(系统的最大标记值),$f_{ol}=f_{oh}$=LOW(系统的最小标记值).$T(s,o)$及 f_o 的含义同上.下面给出在多等级完整性安全政策下,信息流动时所需的安全判定条件以及主体标记的变化规则.

(4) 在信息从客体 o 流入主体 s 时(如 s 读取 o),判定条件及主体标记的变化规则(规则 4)如下:

IF ($f_o \geq f_c$) THEN $T(o,s)$=true;

ELSE IF ($f_o \geq f_{oh}$) THEN {$f_c=\mathrm{Min}(f_c,f_o)$,$f_{il}=\mathrm{Min}(f_{il},f_o)$, $T(o,s)$=true};

ELSE $T(o,s)$=false

(5) 在信息从主体 s 流入客体 o 时(如 s 写入 o),判定条件及主体标记的变化规则(规则 5)如下:

IF ($f_c \geq f_o$) THEN $T(s,o)$=true

ELSE IF ($f_s \geq f_o$) && ($f_{il} \geq f_o$) THEN {$f_c=\mathrm{Max}(f_c,f_o)$, $f_{oh}=\mathrm{Max}(f_{oh},f_o)$, $T(s,o)$=true};

ELSE $T(s,o)$=false

(6) 当信息在主体 s 和客体 o 之间双向流动时(如 s 既读 o 又写 o),判定条件及主体标记的变化规则(规则 6)如下:

IF ($f_c = = f_o$) THEN {$T(o,s)$=true, $T(s,o)$=true}

ELSE IF ($f_s \geq f_o$) && ($f_{il} \geq f_o$) && ($f_o \geq f_{oh}$) THEN {$f_c=f_o$,$f_{oh}=\mathrm{Max}(f_{oh},f_o)$,$f_{il}=\mathrm{Min}(f_{il},f_o)$,$T(o,s)$=true,$T(s,o)$=true}

ELSE {$T(o,s)$=false,$T(s,o)$=false}

从上面的变化规则可以看出,信息的流动同样取决于主体的 6 个标记和客体的标记之间的支配关系,主体标记的变化也依赖于它的访问历史(即信息的流向和访问过的客体).但是,与保密性安全政策相反,随着信息的流入,主体的当前标记和主体的流入信息的最小标记是单调递减的;随着信息的流出,主体的当前标记和主体的流出信息的最大标记是单调递增的.

在 SLCF 中,主体的流入信息的最小标记(f_{il})指定了信息能够合法流出主体的客体标记的上界,主体的流出信息的最大标记(f_{oh})指定了信息能够合法流入主体的客体标记的下界.因此,只要信息源客体的标记不能支配主体的流出信息的最大标记(f_{oh}),就是非法的流入信息流;只要主体的流入信息的最小标记(f_{il})不能支配信息宿客体的标记,就是非法的流出信息流;只要禁止这两种非法信息流,我们就可以确保不使进程产生向上的信息流,因此就实现了多等级完整性安全政策.

4 讨 论

1976 年,BELL 等人提出的 BLP 模型[8]和 1977 年 BIBA 提出的 BIBA 模型[9]一直以来被认为是开发多级安全系统的奠基石,但是随着基于这两个模型(尤其是 BLP 模型)的安全系统的不断开发,人们发现严格实施此模型的系统的实用性很差,而带有可信主体扩展的系统又过于放宽了对安全性的要求[5,7].基于本文提出的 SLCF,我们在所研制的安全操作系统(RS-Linux)*中,实现了多等级保密性和多等级完整性安全政策[11].实际运行情况表明,系统在保持较强的安全性的同时,具有很好的灵活性和实用性.另外,我们在文献[10]中对使用该标记框架实现多等级保密性安全政策进行了严格的数学证明.

由于标记功能是实现多等级安全系统的基础,1983 年美国国防部的 TCSEC(即“桔皮书”)[2]和 1999 年 ISO/IEC 的国际信息安全通用评估准则 ISO/IEC 15408(即“CC 标准”)[12]都把它作为一项安全功能需求明确指出.我们基于 SLCF 框架,在 RS-Linux 中实现了 CC 标准中所有与标记相关的安全需求.其中包括:非标记用户数

* RS-Linux 于 2001 年 6 月通过了国家公安部信息安全检测中心的评测,达到计算机信息系统安全保护等级划分准则第三级的要求.于 2002 年 1 月通过了中国科学院组织的“基于国际/国内标准的安全操作系统”项目的鉴定和验收.

据导出(FDP_ETC.1)、标记用户数据导出(FDP_ETC.2)、强制访问控制政策(FDP_IFC.1)、强制访问控制功能(FDP_IFF.2)、非标记用户数据导入(FDP_ITC.1)和标记用户数据导入(FDP_ITC.2).

实现结果充分表明了 SLCF 不仅提供了全面和灵活的表达能力,而且具备易于实现的特点.为了方便用户使用和管理安全标记功能,我们实现了基于命令行和图形的维护工具,用户可以很方便地设置和修改主体和客体的安全标记.另外,我们修改了一些常用系统命令和应用程序,如 ls,who,id,tar 等,用户可以使用这些工具方便地查看用户和文件的安全标记属性,并可以选择以含安全标记的方式导入和导出文件.

5 结 语

标记功能是实现多级安全系统的基础,是实施强制访问控制的前提.本文分析了国际上前期几个安全系统的标记实施情况,并指出了它们所存在的缺陷.针对发现的问题,我们提出了一个安全标记公共框架(SLCF).该框架基于静态客体标记和动态主体标记,引入了访问历史的概念,并给出了一个完备的标记函数集合.文中证明,基于此框架既可以实施多等级保密性安全政策,也可以实施多等级完整性安全政策.另外,我们在基于 Linux 的安全操作系统(RS-Linux)的研制项目中实现了该框架.实现结果表明,基于该框架,可以完全实现 CC 标准中所有与标记相关的安全功能需求,而且该系统在保证安全性的同时,还具有相当的灵活性和实用性.

多级可用性安全系统是抵御 DOS 攻击的一个较好的方法,如何利用和完善 SLCF 框架,实现多级可用性安全系统,将是我们进一步探讨和研究的内容.

References:

[1] Branstad D. Data categorization and labeling (executive summary). In: Proceedings of the 13th National Computer Security Conference. Washington: NIST Press, 1990. 32~33.

[2] Department of Defense Computer Security Center, Department of Defense Trusted Computer System Evaluation Criteria. Fort George G. Meade, MD 20755, 1983.

[3] Bell DE, Lapadula LJ. Secure computer systems: unified exposition and multics interpretation. MTR-2997, MITRE Corp., 1976.

[4] Gligor VD, Burch EL, Chandersekaran CS, Chapman RS, Dotterer LJ, Hecht MS, Jiang WD, Luckenbaugh GL, Vasudevan N. On the design and the implementation of secure Xenix workstation. In: IEEE Symposium on Security and Privacy. IEEE Computer Society, 1987. 102~117.

[5] Landwehr CE, Heitmeyer CL, Mclean J. A security model for military message systems. ACM Transactions on Computer Systems, 1984,9(3):198~222.

[6] Mcilroy MD, Reeds JA. Multilevel security in the UNIX tradition. Software Practice and Experience, 1992,22(8):673~694.

[7] Lin TY. Bell and Lapadula axioms: a “new” paradigm for an “old” model. In: Proceedings of the 1992~1993 ACM SIGSAC on New Security Paradigms Workshop. Little Compton: ACM Press, 1993. 82~93.

[8] Bell DE, LaPadula LJ. Secure computer systems: a mathematical model. Technical Report, ESD-TR-73-278, Bedford, MA: MITRE Corp., 1973.

[9] Biba KJ. Integrity considerations for secure computer systems. Technical Report, ESD-TR-76-372, Bedford, MA: USAF Electronic Systems Division, Hanscom Air Force Base, 1977.

[10] Shi WC. Sun YF, Liang HL. An adaptable labeling enforcement approach and its correctness for the classical Blp security axioms. Computer Research and Development, 2001,38(11):1366~1372 (in Chinese with English Abstract).

[11] Liang HL, Sun YF. Enforcing integrity protection in operating system. In: Proceedings of the 2001 International Conference on Computer Networks and Mobile Computing. CA: IEEE Computer Society, 2001. 435~440.

[12] Joint Technical Committee 1. Evaluation criteria for IT security——part 2: security functional requirements. ISO/IEC 15408-2:1999(E). The International Organization for Standardization and the International Electrotechnical Commission, 1999.

附中文参考文献:

[10] 石文昌,孙玉芳,梁洪亮.经典 BLP 安全公理的一种适应性标记实施方法及其正确性.计算机研究与发展,2001,38(11):1366~1372.