# 带时间特性的 BLP 模型及其在 Linux 上的设计与实现*)

谭 良[1,2] 周明天[2]

(电子科技大学计算机科学与工程学院 成都 610054)[1] (四川师范大学电子工程学院 成都 610066)[2]

**摘 要** BLP(Bell&LaPadula)模型是用来在计算机系统内实施多级安全政策和自主安全政策的一种访问控制模型。本文将系统时间看成一个基本的安全要素,提出了带时间特性的 BLP 模型(BLP with Time Character;记为 BLP_T),并将之形式化。BLP_T 解决了主客体安全标记随时间的变化问题,以及主体可自主地决定其他的哪些主体可以在何时访问他拥有的客体。最后,将 BLP_T 应用于 Linux,得到了一个原型。实验证明,该原型可以实现主客体安全标记时间特性、主体对客体的时间约束。BLP_T 使得 BLP 模型呈现出环境敏感性。

**关键词** 信息安全,访问控制,BLP,带时间特性的 BLP,时间约束

## Design and Implementation of BLP with Time Character on Linux

TAN Liang[1,2] ZHOU Ming-Tian[1]

(School of Comp. Sci. & Engn., Univ. of Electronic Sci. & Tech. of China, Chengdu 610054)[1]

(College of Electronic Engineering, Sichuan Normal University, Chengdu 610066)[2]

**Abstract** BLP(Bell&LaPadula) model is used to implement Multi-Level Security Policy (MLS) and Discretionary Access Control Policy(DAC) in computer system. In this paper, system time is look as a basic secure element, BLP_T (BLP with Time Character) is presented, and formalized. BLP_T resolves problems that secure labels of subject and object are changed along with time, and that the subject discretionarily decides that who, by when, can access his owns objects. Finally, BLP_T is implemented on Linux, and gain a demo. It is proved that this demo can realize time character of secure labels and time constraint between subject and object. BLP_T makes the BLP model take on environment sensitivity.

**Keywords** Information security, Access control, BLP, BLP with time character, Time constraint

## 1 引言

Bell&LaPadula(BLP)模型是由 Bell 和 LaPadula 于 1973 年提出并于 1976 年修定、整合和完善的安全模型[1],它是定义多级安全性(MLS)的基础。

在 BLP 公理的常规实施法中[2~4],进程(主体)的当前安全标记一旦确定之后,在进程的整个生存期内是不会改变的[5]。但无论采取什么方法,在给进程的当前安全标记赋值时,很难预测进程以后的资源访问行为,所赋的值极有可能无法满足合法的资源访问要求。在安全标记格模型[6,7]的基础上,文[8]、[9]提出了一种新的 MLS 实施策略 A-BLP(A new enforcement approach of BLP),文[10] 提出了 SLCF(Security Label Common Framework),允许系统根据进程活动的实际场景对当前安全标记进行合理的动态调整,可解决赋值不准确和合法访问受拒绝的问题,因此 BLP 模型具有历史敏感性,呈现出动态特征。

但是 A-BLP 和 SLCF 实施策略对主题安全标记的调整有局限性,因为这种方式依赖于主体的最大安全标记、主体当前安全标记、客体安全标记和主体的历史访问行为,而主体的最大安全标记、主体当前安全标记和客体安全标记在一些情况下可能会随着时间的变化而变化;并且 BLP 的自主访问控制也没有考虑时间特性,即主体可自主地决定其他的哪些主体可以在何时访问他拥有的客体。如:一个军事指挥员,在战争的不同阶段,其最大安全标记、当前安全标记应该有所不同,而其访问的客体的安全标记和访问许可也会发生变化,在前一个阶段还是绝密的战况信息,到了下一个阶段就成了公共信息。解决这一问题的办法是在某一时刻到来时由超级用户变更主体最大安全标记、当前安全标记和客体安全标记,或主体在某一时刻到来时变更客体的访问属性,但这种解决办法严重依赖于超级用户和主体的记忆。在一些主客体安全标记具有时间阶段性、客体对访问许可有严格时间要求的系统中(如军事情报、新闻等),这种解决方法是不适合的。

在本文中,我们将专注于 BLP 模型时间特性的研究,通过构建一般的、形式化的时间表达方法,在无时间特性的 BLP 模型基础之上,做时间特性的扩展,并将带时间特性的 BLP 模型应用于 Linux 操作系统中。

## 2 对安全标记和客体访问许可的时间特性分析

### 2.1 激活时间约束

规定主体安全标记或客体访问许可只能在某些时间范围内可以激活。这条约束是很自然的。对于主客体安全标记在某些时间范围内激活的情况,如上述的一个军事指挥员例子,对于客体访问许可只能在某些时间范围内激活的情况,如在一个组织中,有工作时间的限制,在非工作时间的范围内,将不允许某些访问许可的发生。

### 2.2 激活时间长度约束

规定主客体安全标记或客体访问许可每次只可以激活不超过一个固定长度的时间。当某个主客体安全标记或访问许可很重要、很关键时,如果它激活时间过长,就会有被人破坏而产生严重后果的可能。

*)国家 863 宽带 VPN 项目 863-104-03-01 课题资助;2003 年度四川省科技攻关项目 03GG007-007 资助。谭 良 博士,主要研究方向为信息安全、中间件;周明天 教授,博士生导师,主要研究方向为网络计算、信息安全、分布式并行处理。

### 2.3 定长时间内激活时间长度限制

规定主客体安全标记或客体访问许可在一定的时间长度内累计激活时间不超过一个规定的上限。

## 3 对 BLP 的时间特性扩展

### 3.1 时间的定义[11]

**定义 1**(时间点序列) $T=\{\cdots,t_i,t_{i+1},\cdots\}$,当 $t_i\in T,i\in Z$,定义集合里的元素是有序的,即 $t_i<t_j\Leftrightarrow i<j,i=t_j\Leftrightarrow i=j$,还定义 $t_i-t_j=i-j$。$t_i\in T$ 代表一个真实世界的时间点。但具体代表哪个时间点,则根据实际的实现而定。为讨论方便,定义一个函数映射 $t_i$ 到真实的时间:

$real\_time(t_i)$:规定 $real\_time(t_i)-real\_time(t_j)>0$,如果 $i>j$。

这个函数与系统具体实现有关。

如约定 $real\_time(t_i)-real\_time(t_j)=\Delta t,\Delta t$ 为常数,则更容易理解和处理。$real_time(t_0)$表示 $t_0$ 所代表的真实时间。当此值确定,若 $\Delta t$ 也确定,则所有 $real_time(t_i)$也都确定了。

**定义 2**(时间区间集合) 对于 $TR\subseteq(T\times T)$,有$(t_i,t_j)\in TR\Leftrightarrow t_i<t_j$,可以将 $TR$ 表示为$\{\cdots,tr_i,tr_{i+1},\cdots\}$,其中 $tr_i$ 表示某一时间区间。

### 3.2 BLP_T 的时间处理操作

$t_c\in T$,是一个全局的原子时间,表示当前时间。

$LTR\subseteq(L\times TR)$,当$(L(t_i,t_j))\in LTR$,则 $t_i$ 表示安全标记 $L$ 的激活时间,$t_j$ 表示安全标记 $L$ 的结束时间,$t_j=t_c+1$。

$ATR\subseteq(A\times TR)$,当$(x(t_i,t_j))\in ATR$,则 $t_i$ 表示访问许可 $x$ 的激活时间,$t_j$ 表示访问许可 $x$ 的结束时间,$t_j=t_c+1$。

**定义 3**(BLP_T 的全局函数)

$start:TR\to T:start((t_i,t_j))=t_i$

$end:TR\to T:end((t_i,t_j))=t_j$

$D\_active\_start\_time:A\to T:D\_active\_start\_time(x)=t_i$, $if(x,(t_i,t_j))\in ATR$

$D\_active\_end\_time:A\to T:D\_active\_end\_time(x)=t_j$, $if(x,(t_i,t_j))\in ATR$

$D\_active\_length:A\to T:D\_active\_length(x)=(D\_active\_start\_time(x)-D\_active\_end\_time(x))$

$L\_active\_start\_time:L\to T:L\_active\_start\_time(L)=t_i$, $if(L,(t_i,t_j))\in LTR$

$L\_active\_end\_time:L\to T:L\_active\_end\_time(L)=t_j$, $if(L,(t_i,t_j))\in LTR$

$L\_active\_length:L\to T:L\_active\_length(L)=(L\_active\_start\_time(L)-L\_active\_end\_time(L))$

$in\_range:TR\times T\to\{ture,false\}:in\_range(t,(t_i,t_j))=(t\geqslant t_i)\wedge(t\leqslant t_j)$

$not\_in\_range:TR\times T\to\{ture,false\}:not\_in\_range(t,(t_i,t_j))=(t<t_i)\vee(t>t_j)$

$in\_some\_range:T\times 2^{TR}\to\{true,false\}:in\_some\_range(t,2^{TR})=\exists tr_i\in 2^{TR},in\_range(t,tr_i)=true$

### 3.3 BLP_T 模型描述

在带时间特性的 BLP 模型中,系统的状态 $v\in V$ 由一个有序的五元组$(b,M,f,H,t_c)$所表示,其中 $b,H$ 的含义没发生变化;$M$ 中的元素 $M_{ij}\subseteq ATR$,表示在 $TR$ 主体 $S_i$ 对客体 $O_j$ 具有的访问权限;$f=(f_s,f_o,f_c)$,$f_s$、$f_o$、$f_c$ 求出的安全标记 $L\subseteq LTR$。

状态 $v=(b,M,f,H,t_c)$满足带时间特性的 BLP,可定义以下访问控制规则:

**公理 BLP_T-1**(ss-特性):在状态 $v=(b,M,f,H,t_c)$,若对任意主体 $S$,以下条件都成立,则状态 $v$ 满足简单安全特性(ss-特性):

(1)主客体安全标记只能在某些时间范围内可以激活,iff 对所有的

$$O\in b(S:r,w)\Rightarrow\begin{cases}①\ f_s(S)\in LTR\wedge\\ ②\ f_o(0)\in LTR\wedge\\ ③\ in\_range(t_c,(L\_active\_start\_time(f_s(S)),L\_active\_end\_time(f_s(S))))=ture\wedge\\ ④\ in\_range(t_c,(L\_active\_start\_time(f_0(O)),L\_active\_end\_time(f_o(O))))=ture\wedge\\ ⑤\ f_s(S)>f_o(O)\end{cases}$$

(2)主客体安全标记每次只可以激活不超过一个固定长度的时间 $T_{Length}$,$iff$ 对所有的

$$O\in b(S:r,w)\Rightarrow\begin{cases}①\ L\_active\_length(f_s(S))<T_{Length}\wedge\\ ②\ L\_active\_length(f_o(O))<T_{Length}\wedge\\ ③\ f_s(S)>f_o(O)\end{cases}$$

(3)主客体安全标记在一定的时间长度内的累计激活时间不超过一个规定的上限 $T_{Limit}$,iff 对所有的

$$O\in b(S:r,w)\Rightarrow\begin{cases}①\ \Sigma active\_length(f_s(S))<T_{Limit}\wedge\\ ②\ \Sigma active\_length(f_o(O))<T_{Limit}\wedge\\ ③\ f_s(S)>f_o(O)\end{cases}$$

**公理 BLP_T-2**(*-特性):在状态 $v=(b,M,f,H,t_c)$,若对 $S'$中的任意主体 $S$,以下条件都成立,则状态 $v$ 满足相对于 $S'$ 的 *-特性:

(1)主客体安全标记只能在某些时间范围内可以激活,iff 对所有的

$$O\in b(S:a)\Rightarrow\begin{cases}①\ f_c(S)\in LTR\wedge\\ ②\ f_o(O)\in LTR\wedge\\ ③\ in\_range(t_c,(L\_active\_start\_time(f_c(S)),L\_active\_end\_time(f_c(S))))=ture\wedge\\ ④\ in\_range(t_c,(L\_active\_start\_time(f_o(O)),L\_active\_end\_time(f_o(O))))=ture\wedge\\ ⑤\ f_s(S)>f_o(O)\end{cases}$$

对于 $O\in b(S:r)$,$O\in b(S:w)$,只需将⑤分别改为 $f_c(S)<f_o(O)$,$f_c(S)=f_o(O)$。

(2)主客体安全标记每次只可以激活不超过一个固定长度的时间 $T_{Length}$,iff 对所有的

$$O\in b(S:a)\Rightarrow\begin{cases}①\ L\_active\_length(f_c(S))<T_{Length}\wedge\\ ②\ L\_active\_length(f_o(O))<T_{Length}\wedge\\ ③\ f_s(S)>f_o(O)\end{cases}$$

对于 $O\in b(S:r)$,$O\in b(S:w)$,只需将③分别改为 $f_c(S)<f_o(O)$,$f_c(S)=f_o(O)$。

(3)主客体安全标记在一定的时间长度内的累计激活时间不超过一个规定的上限 $T_{Length}$,iff 对所有的

$$O\in b(S:a)\Rightarrow\begin{cases}①\ \Sigma active\_length(f_c(S))<T_{Length}\wedge\\ ②\ \Sigma active\_length(f_o(O))<T_{Length}\wedge\\ ③\ f_c(S)>f_o(O)\end{cases}$$

对于 $O\in b(S:r)$,$O\in b(S:w)$,只需将③分别改为 $f_c(S)<f_o(O)$,$f_c(S)=f_o(O)$。

**公理 BLP_T-3**(ds-特性):在状态 $v=(b,M,f,H,t_c)$,若对任意主体 $S_i$和任意客体 $O_j$,以下条件都成立,则状态 $v$ 满足自主安全特性(ds-特性):

(1)客体的访问许可只能在某些时间范围内可以激活，iff对所有的

$(S_i, O_j, x) \in b \Rightarrow x \in M_{ij} \wedge in\_range(t_c, (active\_start\_time(x), active\_start\_time(x))) = ture$

(2)客体的访问许可每次只可以激活不超过一个固定长度的时间 $T_{Length}$

$(S_i, O_j, x) \in b \Rightarrow x \in M_{ij} \wedge active\_leng(x) < T_{Length}$

(3)客体的访问许可在一定的时间长度内的累计激活时间不超过一个规定的上限 $T_{Limit}$

$(S_i, O_j, x) \in b \Rightarrow x \in M_{ij} \wedge \Sigma active\_length(x) < T_{Limit}$

## 4 BLP_T 模型在 Linux 中的设计与实现

目前 Linux 操作系统采用的是一种基于权限位的自主访问控制机制，这种基于权限位的 DAC 机制过于简单，在某些情况下(如用户临时向指定用户开放对所属文件的某些操作权限)无法满足用户的需求。BLP 模型不仅可以实现细粒度的自主访问控制，而且可以防止敏感信息的泄漏，解决信息的保密性问题。因此，该模型在安全操作系统中得到广泛的应用。虽然在应用不断深入的过程中人们发现它存在某些方面的不足，但是通过适当改进，它不断被赋予新的生命力。本文基于安全访问框架模式，在 Linux 中引入 BLP_T，并将实现了 BLP_T 的 Linux 称为 BLP_T_ Linux。

### 4.1 系统当前实际时间的安全考虑

普通的 Linux 内核要管理系统的运行时间(uptime)和当前墙上时间(wall time)，即当前实际时间。内核中大量的活动(其中一些活动是周期性的，还有一些是非周期性的)由 uptime 时间驱动(time driven)。对于 wall time，内核必须借助实时时钟硬件的帮助才行。实时时钟(real time clock)是用来持久存放系统时间的设备，它与 CMOS 集成在一起，并通过主板电池供电，所以即便在关闭计算机系统之后，实时时钟仍能继续工作。系统启动时，内核读取实时时钟，将所读的时间存放在变量 xtime 中作为墙上时间(wall time)。xtime 保存着从 1970 年 1 月 1 日 0：00 到当前时刻所经历的秒数。

普通的 Linux 把系统当前实际时间看成是一个外部条件，与安全政策的实现无关，超级用户有权修改系统的时间。在 BLP_T_Linux 系统中，把系统当前时间作为基本的安全要素，映射为 BLP_T 的 $t_{currenttime}$，并将与时间相关的全局函数作为一个模块集成到可信计算机(TCB)中。

### 4.2 主客体安全标记

在 BLP_T_Linux，唯一的主体是进程，它可以在系统初始化时创建或用户登录时创建，或在系统运行过程中被其它进程创建；客体包括文件、目录、特别文件、共享内存、消息、信号量、流、管道以及某些进程等。在下面的描述中，以文件客体为例。

主客体的安全标记的表示形式为

$\{L_{t1}, L_{t2}, \cdots, L_{tn}\}$

其中 $L_{ti} = (L, (t_i, t_j)) \in LTR$。

修改主体进程的数据结构，增加存放主体安全标记的结构体[12,13]；修改文件 inode 节点结构，其中的内存索引节点结构和磁盘节点结构均应做相应修改，增加文件安全标记的结构体。

在系统中，主客体的安全标记则通过如下方法赋值：

·用户的安全标记是在系统管理员创建用户时设置，在用户安全文档中为每个用户建立一项，表明安全级范围、时间段，并说明用户的缺省安全级和时间段。进程的安全级在登录后创建进程的时候确定，进程的安全级在其代表用户的安全级范围之内。

·用户创建的客体的安全标记可由创建该客体的进程来设定，且客体的安全级必须大于或等于其父目录的安全级。父目录是多级目录的问题，请参见文[12]、[13]，这里不做详细讨论。

### 4.3 访问控制列表(ACL)的扩展

普通的 Linux 操作系统是通过访问控制列表(ACL)来实现自主访问控制政策的。所谓访问控制列表，就是与系统中客体相联系的用来指定哪些用户和组可以以何种模式访问该客体的控制列表。在下面的描述中，以文件客体为例，对 BLP_T_Linux 系统中的访问控制列表(ACL_T)加以详细的说明。

在 BLP_T_Linux 系统中，每一个文件只能有一个与之相联的 ACL_T 列表，而一个 ACL_T 可以有多个项，每一个项由一个四元组(*Type*, *ID*, *Perm*, *Tconstraint*)组成，分别描述如下：

·*Type*：用户或组的标志。用来表明此 ACL_T 项是指定用户还是指定组的访问控制机制；

·*ID*：用户或组的 ID。用来表明此 ACL_T 项所指定的用户或组的 ID 号；

·*Perm*：存取权限。用来表明此 ACL_T 项所指定的用户或组对此文件的访问权限。

·*Tconstraint*：时间约束。用来表明此 ACL_T 项所指定的用户或组对此文件的访问权限起作用的时间要求。

用户和组标志包括以下 6 类：(1)ACL-USER-OBJ_T：代表客体属主；(2)ACL-USER_T：表示除客体属主以外的其他单个用户；(3) ACL-GROUP-OBJ_T：代表属主所在的用户组；(4) ACL-GROUP_T：代表其他用户组；(5) ACL-OTHER_T：代表除了上述 ACL 项中出现的用户或用户组之外的其他用户；(6) ACL-MASK_T：掩码，表示除了 ACL-USER-OBJ_T 和 ACL_OTHER_T 类型外的所有 ACL 项所允许的最大权限。此外，指定目录客体的缺省 ACL_T 属性，如果某个新创建的客体没有被明确设置 ACL_T 访问权限，则它也可以继承父目录的缺省 ACL_T 作为自己的访问 ACL_T 权限，对于目录客体，同时继承父目录的默认 ACL_T 作为自己的默认 ACL_T 权限。

时间约束包括以下 4 类：(1)NULL_TIME：表示无时间约束，此时 ACL_T 就转化成 ACL；(2)ACTIVE_TIME：激活时间约束；(3)ACTIVE_TIME_LENGTH：激活时间长度约束；(3)CONSTRAINT_TIME_LENGTH：定长时间内激活时间长度限制。

我们基于 Linux 实现了一个 BLP_T_Linux 原型。在该原型中，我们将时间模块加入 Linux 内核，修改和增加相关数据结构，将文件和目录的 ACL 扩展为 ACL_T。鉴于篇幅限制，我们略去详细的实现过程。实验证明，该原型可以实现主客体安全标记时间特性、主体对客体的时间约束。

## 5 性能测算影响

我们通过实验方法测算 BLP_T_Linux 的安全机制产生的性能负面影响。硬件实验平台为普通微机，CPU 为 Pentium R(4)CPU 2.66GMHz，内存 512M，硬盘 80G，Linux 内核版本为 2.4.20，实验时各种安全机制处于工作状态。

带时间特性的安全判定模块通过截获服务请求的方式进行安全控制。服务请求来自系统调用。open 系统调用是最典型、使用最频繁的与安全有关的系统调用，作者以该调用为代表设计测算实验方案。

设计一个 time-open-read-check 程序，它的主要功能是试图用 open 系统调用打开一个指定的文件(文件大小约 1M)并将之读入内存，测出打开文件的时间开销和读入文件的时间开销。打开文件前，open 首先发出访问文件的请求，请求得到允许后才能打开文件。在打开文件操作成功后，它把相应文件的内容读入内存中。time-open-read-check 程序记录打开文件的时间开销和读入文件的时间开销，它还能以指定的次数循环执行以上操作。程序结束前，输出所记录的时间开销。如果文件打开成功，还要关闭该文件。

open 系统调用打开文件的过程中需要对文件路径名进行遍历。历经路径名的每个分量时，都要判断主体对相应分量所对应的目录客体的访问权限和时间限制。实际上，该系统调用所提交的一个服务请求可能引起多个访问许可判断，这由路径名长度确定。实验时，我们对多达 16 级分量的多种子目录深度的路径进行测试，在原 Linux 和 BLP_T_Linux 环境中分别进行同样内容的实验。在运行 time-open-read-check 程序时，我们先关闭读入文件功能，只实现打开文件功能，以 5 万次作为循环次数，连续运行 8 次，以算术平均作为对一个请求处理一次的时间开销。实验数据如表 1 所示。然后打开读入文件功能，以 5000 次作为循环次数，连续运行 8 次，实验数据如表 2 所示。

表 1　处理服务请求的时间开销

| 文件路径名深度/层数 | 传统 Linux 时间开销/μs | BLP_T_Linux 时间开销/μs | BLP_T_Linux 增加的时间开销/% |
|---|---|---|---|
| 2 | 2.5 | 61.2 | 2348 |
| 4 | 2.7 | 86.8 | 3115 |
| 6 | 3.1 | 127.2 | 4003 |
| 8 | 3.8 | 167.6 | 4311 |
| 10 | 4.1 | 223.2 | 5344 |
| 12 | 4.4 | 285.6 | 6391 |
| 14 | 4.9 | 361.6 | 7280 |
| 16 | 5.6 | 464.4 | 8193 |

表 2　资源访问的时间开销

| 文件路径名深度/层数 | 传统 Linux 时间开销/μs | BLP_T_Linux 时间开销/μs | BLP_T_Linux 增加的时间开销/% |
|---|---|---|---|
| 2 | 10120 | 12121 | 20 |
| 4 | 10368 | 12158 | 17 |
| 6 | 10425 | 12173 | 17 |
| 8 | 10497 | 12192 | 16 |
| 10 | 10519 | 12250 | 16 |
| 12 | 10493 | 12351 | 18 |
| 14 | 10240 | 12405 | 21 |
| 16 | 10118 | 12429 | 23 |

根据表 1 的结果，相对于原来的 Linux 系统而言，BLP_T_Linux 处理服务请求所增加的时间开销与路径名深度的关系可描述为图 1 的形式。该图显示，随着路径名深度的增大，处理服务请求所增加的时间开销的比例急剧上升。这是显然的，因为原 Linux 只有简单的访问控制检查，而 BLP_T_Linux 增加了强制访问控制等一系列访问控制检查。

根据表 2 的结果，处理服务请求的绝对时间开销不是很大，表 1 中 BLP_T_Linux 的开销在几十微秒至一千微秒之内，它们在整个系统活动的开销中占的比例很小。服务请求处理开销的这种增长不可能导致系统性能以同样的趋势下降。根据实验结果，BLP_T_Linux 引起系统性能的下降大致在 15%～25%的范围内。

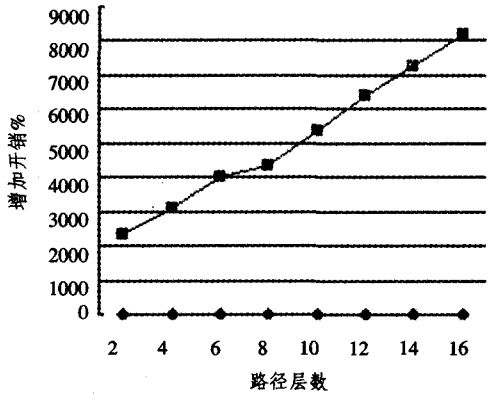


图 1　服务请求处理的开销增长比例

**总结**　BLP_T 在 BLP 上做了时间特性的扩展。BLP_T 引入的时间特性主要体现在主客体安全标记随时间的变化和主体对客体访问许可的时间要求上。我们将系统时间看成一个基本的安全要素，形式化地描述了 BLP_T。最后，我们将 BLP_T 安全政策应用到 Linux 操作系统，得到了一个原型。实验证明，该原型可以实现主客体安全标记时间特性、主体对客体的时间约束。与原系统相比，性能下降 15%～25%。

## 参 考 文 献


1　Bell DE, Lapadula LJ. Secure computer systems: unified exposition and multics interpretation [J]. MITRE Corp, 1976, MTR-2997-3130

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

3　George F, Meade G. Department of Defense Trusted Computer System Evaluation Criteria [J]. Department of Defense Computer Security Center, MD 20755, 1983. 382～443

4　Gligor VD, Burch EL, Chanderskaranetal CS. On the design and the implementation of secure Xenix work stations [C]. In: Proc. of the 1986 IEEE Symposium on Security and Privacy. Oakland, California. IEEE Computer Society Press, 1986. 102～117

5　Gligor VD, Chandersekaran CS, Chapman RS, et al. Design and implementation of secure Xenix [J]. IEEE Transactions on Software Engineering, 1987, SE-13(2): 208～221

6　Sandlu R S. Lattice-Based Access Control Model [J]. IEEE, 1993

7　谭良，罗讯，周明天. 动态多级安全系统安全标记的格模型[J]. 电子科技大学学报，2004,33(4):442～445

8　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[J]. 计算机研究与发展，2001,38(11):1366～1372

9　石文昌，孙玉芳. 多级安全政策的历史敏感性[J]. 软件学报，2003, 14(01):91～96

10　梁洪亮，孙玉芳，赵庆松. 一个安全标记公共框架的设计与实现[J]. 软件学报，2003,14(3):547～552

11　董光宇，卿斯汉，刘克龙. 带时间特性的角色授权约束[J]. 软件学报，2003,13(8):1521～1527

12　刘文清，卿斯汉，刘海峰. 一种用于操作系统安全内核的多级分层文件系统的研究与实现[J]. 电子学报，2002,30(5):763～765

13　刘克龙. 安全 Linux 操作系统及安全 Web 系统的形式化建模与实现[D]. 北京：中国科学院软件所，2001