一个改进的可动态调节的机密性策略模型*

季庆光[1,2+], 卿斯汉[1,2], 贺也平[1,2]

[1](中国科学院 软件研究所,北京 100080)

[2](中国科学院 信息安全技术工程研究中心,北京 100080)

An Improved Dynamically Modified Confidentiality Policies Model

JI Qing-Guang[1,2 +], QING Si-Han[1,2], HE Ye-Ping[1,2]

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

[2](Engineering Research Center for Information Security Technology, The Chinese Academy of Sciences, Beijing 100080, China)

+ Corresponding author: Phn: +86-10-62645412 ext 8006, E-mail: qgji@ercist.iscas.ac.cn, http://www.iscas.ac.cn

Received 2003-05-19; Accepted 2003-11-11

Ji QG, Qing SH, He YP. An improved dynamically modified confidentiality policies model. *Journal of Software*, 2004,15(10):1547~1557.

http://www.jos.org.cn/1000-9825/15/1547.htm

Abstract: This paper presents a model which can support network security objects, improve the Amon ott's rules with small amount of operations and storages for practicality, enhance the flexibility available for system implementation by making the single level becoming level range , and control IPC objects effectively. For these purposes, the Amon ott's rules for dynamically modifying the current sensitivity level are extended to ones for sensitivity levels range, so Bell's work on making the single level becoming level range for network security can be combined with Amon ott's. Considering the cases in the practical system GEMSOS,DG/UX and prototype microkernel system Fluke, single level entity, multiple level entity and special access mode for progress, and the invariants and constraints corresponding to them are introduced. Based on Tmack's way, a sufficient mechanism for IPC objects is posed. In addition, some flaws in ABLP model are pointed out. A new confidentiality policy model with formal specification of invariants, constraints, variables, and constants has been presented with demonstrating reasonableness for some constraints, and it can be used for system design.

Key words: confidentiality policy; formal model; sensitivity level range; multiple level entity; IPC objects

摘 要: 试图提出一个模型,它能为有效处理网络安全对象提供支持.改进 Amon ott 的动态调整规则,使需要动态改变的量减少,从而使一个系统调用级的原子操作需要伴随的附加量的操作和存储减少,提高模型在系统中的实用性.通过把当前安全级变成敏感标签范围而增加模型在系统实现中的灵活性,能有效控制 IPC 对象.为此,

* Supported by the National Natural Science Foundation of China under Grant No.60083007 (国家自然科学基金); the National Grand Fundamental Research 973 Program of China under Grant No.G1999035810 (国家重点基础研究发展规划(973))

作者简介: 季庆光(1968－),男,博士,主要研究领域为安全系统形式化分析;卿斯汉(1939－),男,研究员,博士生导师,主要研究领域为信息安全理论与技术;贺也平(1962－),男,博士,研究员,主要研究领域为安全操作系统形式化模型,无线网络安全模型形式化分析,安全协议的形式化分析.

把Amon ott动态地调整当前敏感标签的规则推广为动态地调整敏感标签范围的规则,这把Bell为处理网络情况而把主体的当前安全级变成敏感标签范围的工作与 Amon ott 的工作有机地结合起来,同时,参考实际中使用的系统 GEMSOS 和 DG/UX 及安全原型微内核系统 Fluke,引入了单级实体、多级实体以及专用于进程的访问模式,并给出了它们应满足的不变量及限制性条件.另外,在参考原型系统 TMach 对 IPC 对象的某些处理方式的基础上,提出了使动态特征非常明显的 IPC 对象得到合理、有效管理的机制,同时还分析了 ABLP 实施方法中存在的一些不当之处.提出了一个新的机密性策略模型 DBLP 的模型不变量,限制性条件及变量类型和常量,并分析了一些限制性条件的合理性,它可以有效应用于系统设计.

关键词: 机密性策略;形式模型;安全级范围;多级实体;IPC 对象

中图法分类号: TP309 **文献标识码**: A

Amon Ott 在其硕士论文[1]中,引入规则“Auto-write”,“Auto-read”及“Auto-read-write”时,在某些前提条件下,它们可以动态地调整主体进程的当前敏感标签.文献[2]把类似于文献[1]中的规则称为 BLP 模型的 ABLP 实施方法,并把它用于开发安全操作系统原型 RS-Linux,同时指出 ABLP 实施方法的中心思想是,当在外层判断空间发现不满足常规判断条件的要求时,试图进入内层判断空间进行进一步判断,它允许一个主体(进程)在其生存期内有条件地动态调整当前敏感标签.的确,能有效调整进程的当前敏感标签将极大地增加系统的灵活性,这是无可置疑的.但是,这往往会使安全性受到极大的威胁,因为敏感标签的可调整性有可能导致机密性策略的失控.寻求灵活性与可控性之间矛盾的有效解决方案是困难的,人们对这个问题的研究已有相当长的历史.

历史上有不少系统把平稳性(tranquility)作为系统实现机密性策略的基本原理,例如,TMach[3],SeaView[4]和 Multics[5].为实现这个原理,Lunt 等人在开发 SeaView 系统时,要求主、客体相关的安全级函数是与状态无关的函数,这样,安全级函数就自动具有平稳性,为满足这个要求,同时考虑到实现的实用性,他们允许以系统外在方式改变客体的安全级,但改变必须经由一个可信进程来完成,而且这个可信主体的能力是不受限制的,可以说是一个完全可信的主体,它的所有行为都被认为是可信的,这与极小特权原理是相悖的.事实上,现实中对一个实体的信任只能在一定范围内.因此,一个可信实体能同时获取所有的特权是不合理的.为了限制可信主体的特权,Schell 等人在开发 GEMSOS 时,在文献[6]中提出了多级可信主体的概念,这个概念不仅拓展了 Bell-LaPadula 模型的可信主体的概念,而且预示了引入授权体系的必要性.GEMSOS 的多级可信主体实际上类似于 Lee 在文献[7]中引入的部分可信主体(partially trusted subject),它们的行为只有在指定的范围内是可信的.这样,可以把权利非常集中的可信主体变换成权利相对分散的多个部分可信主体,这不仅符合极小特权原理及 N 人原则的要求,而且还有利于管理授权的动态化,实现安全属性的合理动态配置及动态变更.正如 Clark 和 Wilson 在他们论文[8]中所指出的,引入部分可信主体更合乎现实的需要.因为在他们的完整性模型中,这些部分可信主体可以替代安全负责人(security officer)为用户提供完整性标签,这不同于 Biba 模型,它只承认一个可信进程——安全管理员.

如何有效描述可信主体是重要的,正如 Rushby 在文献[9]中所指出的,Bell-LaPadula 模型中的这个可信主体使安全内核的安全性的概念变得复杂化,特别是模型中没有形式化可信进程这一概念,导致形式验证难以完成.在我们看来,对可信进程进行形式化描述首先是由 Bell 来完成的,Bell 在 1988 年为处理网络情况在文献[10]中把主体的当前安全级变成一个二元组(a-min,v-max),这一改进使对强制访问控制策略不敏感的进程和程序有更多的可操作的自由性.Bell 把从 a-min 到 v-max 的范围称为一个主体的写范围.Mayer 在文献[3]中指出,a-min 与 v-max 不等的主体可以被定义为可信主体;在某种程度上,这使得可信主体概念化.目前这一概念已被广泛使用,例如,Kargar[11]在开发卡操作系统安全内核时使用了它,文献[12]在开发 SELinux 时也使用了它.

经由外在于系统的可信主体实现安全级的变更是借助于管理配置实现动态调整的必由之路,McLean 在文献[13]中引入安全代数,它的一个架构就是一些模型的集合,这些模型除了能变动不同主体和客体的安全级的主体集合不同以外,它们是相同的.对能改变不同主体和客体的安全级的主体集合进行集合运算,这些模型就变成布尔代数.McLean 的工作表明,如果能改变不同主体和客体的安全级的主体集合就是所有的主体集合,那么系统可能是不安全的;如果能改变不同主体和客体的安全级的主体集合是空集合,那么系统虽然安全,但系统是

不实用的.为了实用,目前人们主要采用上面谈到的方法,加入可信主体.据我们所知,1997 年以前的文献,包括McLean 的,尽管 Biba 很早就已经在其低标模型(low-water mark model)中考虑完整性级别的动态变化,但对机密性策略,仍然没有考虑使用强制规则合理改变主体和客体的当前安全级的相关模型,之所以会出现这样的情况,我们认为不是不可能找到这样的规则,而是寻找它们是很难的.

尽管 Amon Ott 在其硕士论文[1]中首次表达了这些规则,但在公开的文献中没有报道过有关的分析和证明.在文献[2]中,重新叙述了这些可动态调整的规则,并进行了细致的证明,这是很有意义的工作.但是我们发现,文献[2]中的某些证明不严格,一些初始条件的设置是不恰当的,而且无论是文献[1]还是文献[2],都使用了可信主体的概念,而没有给出合理的定义,我们将在本文中详细加以分析.另外,我们发现可以建立一个框架,在这个框架下不使用当前级的概念,也可以有效实现 Ott 的动态调整规则,同时还可以借鉴 Bell 在 1988 年的工作,使该框架在推广到网络环境时,能有效处理多级安全操作系统与多级安全网络系统的衔接.本文将基于文献[14]对多级安全模型的思考及 TMach 对 IPC 对象的处理方式,并参考实际使用系统 GEMSOS 和 DG/UX 的设计经验,提出一个更实用的模型 DBLP.

本文第 1 节分析 ABLP 模型,指出该模型存在的一些有待改进的地方.第 2 节提出一个改进的机密性策略模型 DBLP,该模型把 Amon ott 动态地调整当前敏感标签的规则推广为动态地调整敏感标签范围的规则;引入单级客体、多级客体以及专用于 IPC 的访问模式,并给出它们应满足的不变量及限制性条件;提出使动态特征非常明显的 IPC 对象得到合理、有效管理的机制.第 3 节是与现有模型的比较,指出与 TMach 及文献[14]中提议的多级安全模型的不同.第 4 节总结全文.

1 对 ABLP 模型的分析

1.1 ABLP模型简介

在 ABLP 模型中,文献[2]把文献[1]的规则“Auto-write”,“Auto-read”及“Auto-read-write”中的第 3 种情况分离出来成为 3 条规则:规则 ABLP-1,规则 ABLP-2 和规则 ABLP-3.执行调整的规则变为:

假设 ABLP-1:不管是在内层判断空间中还是在外层判断空间,对请求进行授权都遵守以下规定:

1. 对授权 $rq(S_i,O_j,\underline{r})$时,执行规则 ABLP-1 中的 2.
2. 对授权 $rq(S_i,O_j,\underline{a})$时,执行规则 ABLP-2 中的 2.
3. 对授权 $rq(S_i,O_j,\underline{w})$时,执行规则 ABLP-3 中的 2.

同时,做出如下的约定:

约定 ABLP-1:设 $L_{RH}\in\underline{L}$, $L_{WL}\in\underline{L}$,用 L_{RH} 和 L_{WL} 表示与进程对应的两个判断参数,在进程的生存期使用,在进程被创建时赋初值, L_{RH} 的初值为系统的最小敏感标记值, L_{WL} 的初值为系统的最大敏感标记值.每个进程对应一个 L_{RH} 值及一个 L_{WL} 值.

1.2 ABLP模型中的一些问题

在 BLP 的 ABLP 实施方法中,存在着两方面的问题:一是模型本身存在的问题;另一方面是模型设计中存在的问题.下面我们对这些问题进行详细分析.

模型本身存在的问题:

- 模型中需要动态改变的量较多.一个系统调用级的原子操作要伴随多个附加量的操作和存储,这将严重影响模型在系统中的实用性.
- 模型中没有充分精确地界定可信主体以及可信主体在模型中的作用.如果按文献[3]中的定义,把 $L_{RH}\neq L_{WL}$ 的进程视为可信主体,显然不对;如果把能违背*-性质的进程视为可信主体,那么它们在该模型中的行为也是不确定的.
- 模型没有考虑这里特殊的实施方法可能导致的隐通道问题.

模型设计中存在的问题:

• 模型中定理的证明不严格.

定理 ABLP-1,ABLP-2,ABLP-3 的证明中不经严格分析和证明地认为:

如果 $b^* = b \cup (S_i, O_j, \underline{x})$,那么为证明 b 中元关于状态 (b^*, M, f^*, H) 仍满足 BLP 公理,只需证:对于 $\forall (S_i, O_k, \underline{y}) \in b$,若把 $rq(S_i, O_k, \underline{y})$ 视为 $rq(S_i, O_j, \underline{x})$ 的父请求,它关于状态 (b^*, M, f^*, H) 满足 BLP 公理.

事实上,这个命题不是显然的,因为 $b = \cup_{S_i} b(S_i, \underline{r}, \underline{a}, \underline{w})$,且 $b(S_i, \underline{r}, \underline{a}, \underline{w})$ 可以有不只一个元素,所以可以设 $(S_i, O_1, x_1) \in b(S_i, \underline{r}, \underline{a}, \underline{w})$, $(S_i, O_2, x_2) \in b(S_i, \underline{r}, \underline{a}, \underline{w})$,…, $(S_i, O_m, x_m) \in b(S_i, \underline{r}, \underline{a}, \underline{w})$ 且它们的请求顺序为 $rq(S_i, O_1, x_1) \prec rq(S_i, O_2, x_2) \prec \ldots \prec rq(S_i, O_m, x_m)$.为证明 b 中元关于状态 (b^*, M, f^*, H) 仍满足 BLP 公理,我们必须证明每一个 (S_i, O_j, x_j) 关于状态 (b^*, M, f^*, H) 都满足 BLP 公理.很明显,由上面的命题很难直接获得该命题.

• 模型中的约定存在问题:

约定 ABLP-1 认为,参数 L_{RH} 和 L_{WL} 在进程被创建时赋初值,L_{RH} 的初值为系统的最小敏感标记值,L_{WL} 的初值为系统的最大敏感标记值.一般地说,系统软件的敏感级应该比较低,因为所有的应用软件都要执行它.文献[15]就是把系统软件放置在病毒防护域,而病毒防护域是系统中敏感级最低的域.在这种情况下,如果按约定 ABLP-1 的设置,系统软件将在所有主动进程的操作范围内,这样,利用规则 ABLP-3,所有主动进程都能修改系统软件,这显然是存在问题的.

如果修改参数 L_{RH} 和 L_{WL} 的初值的设置,由于 ABLP 中并不限制当前级与这两个参数的关系,可能出现进程的当前级不在参数 L_{RH} 和 L_{WL} 的范围内,这将产生矛盾.

2 一个改进的机密性策略模型

在文献[12]中,描述 SELinux 配置的多级安全策略时,指出主体和客体都被分配在一个安全级范围(a range of levels),并指出存在安全级范围的迁移规则,该规则将通过创建新客体的主体的安全级范围及相关客体的安全级范围来确定被创建的新客体的安全级范围;也指出存在多实例化客体的安全级范围成员规则,该规则规定了多实例化客体成员的安全级范围.但这个传说中的多级安全模型无论从 SELinux 的源码还是公开文档资料中都是无法得到的.DG/UX 系统的多级安全策略及在文献[14]提议的多级安全模型中,也给主体和客体都配置上一个安全级范围,这说明从单个安全级变成安全级范围,由此而引入多级主体和多级客体是有其合理性的,它是应现实情况的需要作出的改进.文献[14]把主体对主体的操作也加入模型中,这有助于更好地描述 IPC,但是,文献[14]把主体的最大安全级弃之不用,是有悖于 Bell 仅把当前安全级变成一个二元组(a-min,v-max)的初衷的,能否直接用这个二元组替代最大安全级的作用需要进一步讨论.确切地讲,文献[14]提议的多级安全模型不是一个真正的模型,它只是一种安全策略的描述,因此把它精确地表达出来是很有意义的.另一方面,要建立一个模型,寻找合理、有效的描述方式是重要的;文献[16]中的描述方式是 BLP 模型描述方式的一个有效的改进,是值得借鉴的.下面描述的 DBLP 模型将充分考虑这里提到的各个方面.

2.1 DBLP模型变量类型

DBLP 模型变量类型及常量:

• 模型中的实体包括主体和客体,它们的型是

[SUBJECT,OBJECT];

• 模型中存在主体对主体,主体对客体的操作,也包含请求和判定的过程,它们的型设为

[OPERATION,REQUEST,DECISION];

• 模型中的主体和客体都有安全标签及描述它们的谓词,它们的型为

[LEVEL,BOOLEAN];

• 模型中考虑时序因素,它的型为

[TIME].

2.2 DBLP模型状态变量

状态变量的不同赋值是系统状态迁移的直接原因,因此,形式描述中必须清楚地刻画状态变量.

(1) 主体集 $S:\wp SUBJECT$.

客体集 $O:\wp OBJECT$,它由两部分组成:单级客体集 O_single 和多级客体集 O_multi,即 $O=O\_single\cup O\_multi$.单级客体和多级客体将在后面给出定义.

标签集 $level:\wp$ LEVEL.

访问模式集合 $A:\wp$ OPERATION.

(2) 安全标签上的控制关系 $-\succeq-:LEVEL\leftrightarrow LEVEL$.

访问许可函数 $M:O\to ACL$ 是一个部分函数,这里 $ACL\subseteq S\times A$ 且可能的操作有:

- $\underline{r}$ – 一个主体能以读访问一个客体,它能看该客体的内容;
- $\underline{a}$ – 一个主体能以盲写访问一个客体,它能改变该客体的内容,但不能看该客体的内容;
- $\underline{w}$ – 一个主体能以写访问一个客体,它能看而且能改变该客体的内容;
- $\underline{signal}$ – 一个主体能以发信号方式访问一个主体,它能向另一个主体发信号;
- $\underline{cinnect}$ – 一个主体能以连接方式访问一个主体,它能与另一个主体建立连接;
- $\underline{e}$ – 一个主体能以执行方式访问一个客体,它能执行该客体.
- $\underline{control}$ – 一个主体能以控制方式访问一个客体,它能修改该客体的访问控制列表.

(3) 使可用谓词 $\alpha_O:O\to$ BOOLEAN ,$\alpha_S:\to$ BOOLEAN,它们分别表示一个客体,一个主体处于可用态.一个客体或一个主体处于可用态,是指该客体或主体已被创建;一个客体或一个主体处于不可用态,是指该客体或主体已被删除或没有被创建.下文中提到此概念时都按此处的解释理解,以后不再说明.

标名谓词 $named(o):O\to$ BOOLEAN,它表示客体 o 不是 IPC 对象.

指示谓词 $single(o):O\to$ BOOLEAN,它表示客体 o 在某操作下是单级对象.指示谓词 $\mathrm{multi}(o):O\to$ BOOLEAN,它表示客体 o 在某操作下是多级对象.

(4) 激活与终止函数 $\vartheta_S:S\to\wp S$,表示把一个不可用主体映射到能创建该主体的主体集合. $\vartheta_O:S\to\wp S$,表示把一个不可用客体映射到能创建该客体的主体集合. $\kappa_O:S\to\wp S$,表示把一个处于可用态的客体映射到能删除该客体的主体集合. $\kappa_S:S\to\wp S$,表示把一个处于可用态的主体映射到能删除该主体的主体集合.

(5) 当前访问集 $b\subseteq S\times O\times A\cup S\times S\times A$,它表明了当前可访问的情况.

(6) 层次结构 H,它与文献[5]中的层次结构相同.

(7) 敏感级函数 $f=(f_s,a\_\min_s,v\_\max_s,L\_\min_o,L\_\max_o)$,要求它们满足:

- $f_s:S\to$ level,而 $v\_\max_s$ 和 $a\_\min_s$ 将在下面递归地给出定义.
- $L\_\max_o:O\to$ level, $L\_\min_o:O\to$ level

把 $[a\_\min_s,v\_\max_s]$ 称为主体的安全标签范围,记为 $ran(s)$;把 $[L\_\min_o,L\_\max_o]$ 称为客体的安全标签范围,记为 $ran(o)$,它表明该客体所包含数据的最小安全级及最大安全级.

(8) 请求集 R,它表明了系统请求进行的各种操作.

(9) 判断集 $D=\{\underline{yes},\underline{no},\underline{error},?\}$.

定义 2.1. 如果一个客体 o 满足 $L\_\min_o(o)=L\_\max_o(o)$ 或 $L\_\min_o(o)\neq L\_\max_o(o)$,但主体每次对客体 o 访问仅由该客体的安全标签范围内的一个安全级确定,这样的客体 o 被称为具有单级特性,我们也简称此时的客体为单级客体.如果一个客体 o 满足 $L\_\min_o(o)\neq L\_\max_o(o)$,且主体每次对客体 o 访问由该客体的整个安全标签范围确定,此时安全标签范围不仅表明客体所包含数据的最小安全级及最大安全级,而且还表明对客体的某些完整性限制,这样的客体 o 被称具有多级特性,简称为多级客体.一个主体 s 称为可信主体,如果它的安全级无论从何种状态出发,在没有出现特权操作的策略执行过程中都满足如下条件:

$$f_s^*(s)=f_s(s),v\_\max_s^*(s)=v\_\max_s(s),a\_\min_s^*(s)=a\_\min_s(s).$$

由定义不难看出,客体的单级特性和多级特性是与对该客体的具体操作紧密联系的,我们提出单级客体和多级客体的概念是想区分客体对写的敏感性,单级客体更多关心的是客体的机密性,而并不关心向该单级客体

进行写访问的主体的可信任程度.与此相反,多级客体把写访问留给那些可信程度较高的可信主体.也就是说,单级客体是可信主体和一般主体共同访问的对象,而多级客体仅是可信主体访问的对象.举例来说,文件系统、某些可信设备(trusted device)(按文献[15]的定义,可信设备是这样的设备,它不要求强制访问控制的检查,或它要求强制访问控制的检查,但必须由核以外的一些 TCB 分量来负责实施)就是多级客体,对它们的操作必须由可信主体来完成.例如,安装和拆卸文件系统.当一个主体试图进入文件系统对其中的文件进行操作时,它必须先访问文件系统,此时,文件系统作为一个被访问对象并不关心访问主体的可信任程度.在这种情况下,文件系统应被视为单级客体.

2.3 DBLP计算模型

本节我们将抽象地定义 DBLP 的计算模型,并给出规则的一般性形式定义.

定义 2.2. 一个系统 Σ 由以下几部分组成: $(V, R\times D, \tau, v_0)$,其中

- V 是状态空间;
- $R\times D$ 是在当前状态下,为下一个状态的发生所作出的请求输入和判断输出;
- τ 是系统的状态转移函数 $\tau: R\times D\times V\to V$,它与输入的请求和输出的判断有关;
- $v_0\in V$ 是初始状态.

下面我们按文献[16]的方式定义安全系统.

定义 2.3. 设 V_Σ是一个状态空间,C 是 V_Σ的子集,CT 是 $V_\Sigma\times V_\Sigma$的子集,而且 $\Sigma=(V_\Sigma, R\times D, \tau_\Sigma, \sigma_0)$ 是一个系统,那么,一个状态 $v\in V_\Sigma$ 被称为是关于 C 的安全状态,如果 $v\in C$;一个请求和判断 (r,d) 被称为关于 C 和 CT 是安全的,如果 $((v,r,d),v^*)\in\tau_\Sigma$,则有:

- 如果 $v\in C$,则 $v^*\in C$;
- $(v,v^*)\in CT$.

Σ被称为关于 C 和 CT 的安全系统,如果它满足如下条件:

- σ_0 关于 C 是安全的;
- $(r,d)\in R\times D$ 关于 C 和 CT 是安全的.

一般,我们把 C 称为系统的不变量,CT 称为系统的限制性条件,而 $v=(b,\alpha,f,H,M)$ 称为系统的状态,这里, $\alpha=(\alpha_o,\alpha_s,named)$.

定义 2.4. 一个规则是函数 $\rho: R\times V\to D\times V$,如果 $\rho(r,v)=(d,v^*)$,那么 ρ 称为关于 C 和 CT 是安全的,如果 $v\in C\Rightarrow v^*\in C$,而且 $(v,v^*)\in CT$.

2.4 DBLP的C和CT

C_1 可用态属性:如果一个主体能访问一个客体(或主体),则该客体(或被访问主体)和主体都必须处于可用状态.即 $\forall o\in O$, $\forall s,s_1\in S$,如果 $(s,o,x)\vee(s,s_1,x)\in b$,那么 $\alpha_o(o)\vee\alpha_s(s_1),\alpha_s(s)$.

C_2 简单安全性:如果一个主体能以读或写访问一个客体,则该主体的最大安全级必须控制客体最大敏感级.即 $\forall o\in O$, $\forall s\in S$,如果 $(s,o,\underline{r},\underline{a})\in b$,那么 $f_s(s)\succeq L\_\max_o(o)$.

C_3 *–安全性:

如果一个主体能以发信号或连接访问另一个主体,则两个主体的标签范围满足以下关系:

- 如果 s_1 向 s_2 发信号,即 $(s_1,s_2,\underline{signal})\in b$,则 $v\_\max_s(s_2)\succeq v\_\max_s(s_1)$;
- 如果 s_1 向 s_2 发出连接请求,即 $(s_1,s_2,\underline{connect})\in b$,则 $ran(s_1)\subseteq ran(s_2)$.

若 s 是可信主体(用 S_T 代表所有可信主体所构成的集合),o 是多级对象,则

- 若 $(s,\underline{r})\in M(o)$,则 $L\_\max_o(o)\preceq v\_\max_s(s)$;
- 若 $(s,\underline{a})\in M(o)$,则 $L\_\min_o(o)\succeq a\_\min_s(s)$;
- 若 $(s,\underline{w})\in M(o)$,则 $L\_\max_o(o)\preceq v\_\max_s(s)\wedge L\_\min_o(o)\succeq a\_\min_s(s)$.

若 s 是主体,o 是单级对象,则

- 若 $(s,\underline{r})\in M(o)$,则 $L\_\min_o(o)\preceq v\_\max_s(s)$;
- 若 $(s,\underline{a})\in M(o)$,则 $L\_\max_o(o)\succeq a\_\min_s(s)$;
- 若 $(s,\underline{w})\in M(o)$,则 $L\_\min_o(o)\preceq v\_\max_s(s)\wedge L\_\max_o(o)\succeq a\_\min_s(s)$.

C_4 兼容属性:客体在分级结构中的序关系应与客体在安全标签中的序关系兼容.即

如果 $o_2\in H(o_1)$,则 $\alpha_o(o_1)$ 而且 $\alpha_o(o_2)$,同时 $L\_\max_o(o_1)\preceq L\_\min_o(o_2)$.

C_5 敏感级控制属性:敏感级函数 $f=(f_s,a\_\min_s,v\_\max_s,L\_\min_o,L\_\max_o)$,在任何状态下满足:

- 对任意的 $s\in S$, $f_s(s)\succeq v\_\max_s(s)\succeq a\_\min_s(s)$
- 对任意的 $o\in O$, $L\_\max_o(o)\succeq L\_\min_o(o)$

上面我们定义了系统的不变量,接下来我们定义系统的限制性条件.

CT_1 安全标签动态变化原则:

(1) 若 $(s,\underline{r})\in M(o)$,且 *single*(*o*),则 $L\_\min_o(o)\preceq a\_\min_s(s)$,或者 $a\_\min_s(s)\preceq L\_\min_o(o)\preceq v\_\max_s(s)$,在第 1 种情况下:

$$b^*=b\cup(s,o,\underline{r}),$$
$$L\_\max_o^*(o)=L\_\max_o(o),$$
$$L\_\min_o^*(o)=L\_\min_o(o),$$
$$f_s^*(s)=f_s(s),$$
$$v\_\max_s^*(s)=v\_\max_s(s),$$
$$a\_\min_s^*(s)=a\_\min_s(s).$$

在第 2 种情况下:

$$b^*=b\cup(s,o,\underline{r}),$$
$$L\_\max_o^*(o)=L\_\max_o(o),$$
$$L\_\min_o^*(o)=L\_\min_o(o),$$
$$f_s^*(s)=f_s(s),$$
$$v\_\max_s^*(s)=v\_\max_s(s),$$
$$a\_\min_s^*(s)=L\_\min_o(o).$$

(2) 若 $(s,\underline{r})\in M(o)$,且 *multi*$(o)$,则 $L\_mas_o(o)\preceq v\_mas_s(s)$,在这种情况下:

$$b^*=b\cup(s,o,\underline{r}),$$
$$L\_\max_o^*(o)=L\_\max_o(o),$$
$$L\_\min_o^*(o)=L\_\min_o(o),$$
$$f_s^*(s)=f_s(s),$$
$$v\_\max_s^*(s)=v\_\max_s(s),$$
$$a\_\min_s^*(s)=a\_\min_s(s).$$

(3) 若 $(s,\underline{a})\in M(o)$,且 *single*(*o*),则 $L\_mas_o(o)\succeq a\_mas_s(s)$,或者 $a\_\min_s(s)\preceq L\_mas_o(o)\preceq v\_\max_s(s)$ 在第 1 种情况下:

$$b^*=b\cup(s,o,\underline{a}),$$
$$L\_\max_o^*(o)=L\_\max_o(o),$$
$$L\_\min_o^*(o)=L\_\min_o(o),$$
$$f_s^*(s)=f_s(s),$$
$$v\_\max_s^*(s)=v\_\max_s(s),$$
$$a\_\min_s^*(s)=a\_\min_s(s).$$

在第 2 种情况下:

$$b^* = b \cup (s, o, \underline{a}),$$
$$L\_\max_o^*(o) = L\_\max_o(o),$$
$$L\_\min_o^*(o) = L\_\min_o(o),$$
$$f_s^*(s) = f_s(s),$$
$$v\_\max_s^*(s) = L\_\max_o(o),$$
$$a\_\min_s^*(s) = a\_\min_s(s).$$

(4) 若 $(s, \underline{a}) \in M(o)$,且 $multi(o)$,则 $L\_\min_o(o) \succeq v\_\min_s(s)$,在这种情况下:

$$b^* = b \cup (s, o, \underline{r}),$$
$$L\_\max_o^*(o) = L\_\max_o(o),$$
$$L\_\min_o^*(o) = L\_\min_o(o),$$
$$f_s^*(s) = f_s(s),$$
$$v\_\max_s^*(s) = v\_\max_s(s),$$
$$a\_\min_s^*(s) = a\_\min_s(s).$$

(5) 若 $(s, \underline{w}) \in M(o)$,且 $single(o)$,则

状态变迁既满足 $(s, \underline{r}) \in M(o)$,$single(o)$的情况,又满足 $(s, \underline{a}) \in M(o)$,$single(o)$的情况.

(6) 若 $(s, \underline{w}) \in M(o)$,且 $multi(o)$,则

状态变迁既满足 $(s, \underline{r}) \in M(o)$, $multi(o)$ 的情况,又满足 $(s, \underline{a}) \in M(o)$, $multi(o)$ 的情况.

我们把可信主体定义为可信操作范围有限且操作后 $ran(s)$ 不变的主体,这与以往所有文献中定义的可信主体是不相同的.值得注意的是,可信主体的 $ran(s)$ 是可以经由外在于策略的执行进行管理调整的.

在文献[17]中引入了多级信息结构 *container*,按照 Thomas[18]对文件系统的处理方法,系统中多级信息结构是很多的.但是文献[17]中仍然把 *container* 处理为单级客体,赋予它一种特殊的安全属性 CCR,同时规定 *container* 的安全级至少要等于 *container* 所包含实体的最大安全级,这种处理方法显然不自然.相比之下,我们引入多级客体,之后把单级客体视为退化的多级客体的处理方法是比较合理、自然的.

文献[14]中没有考虑兼容性问题,我们引入的兼容属性好象不是很自然,但这是为配合 $v\_\max_s(s)$, $a_\min_s(s)$的动态变化而提出的.

CT_2 自主访问控制:

如果一个主体被认可访问某客体,该主体必须获得访问这个客体的必要的自主许可权.即

$$\forall s \in S, o \in O, x \in A, (s, o, x) \in b^* - b \wedge named(o) \Rightarrow (s, x) \in M(o).$$

CT_3 主体的创建与删除:

只有 TCB 或标签范围包含待创建主体标签范围的主体能创建一个主体,只有 TCB 或包含标签范围是待删除的主体标签范围的主体能删除一个主体.即

$$\forall s \in S, \neg\alpha_s(s) \wedge \alpha_s^*(s) \Rightarrow \exists s_1 \in \vartheta_s(s) \text{ s.t. } (s_1 = TCB \vee ran(s) \subseteq ran(s_1)),$$
$$\forall s \in S, \alpha_s(s) \wedge \neg\alpha_s^*(s) \Rightarrow \exists s_1 \in \kappa_s(s) \text{ s.t } (s_1 = TCB \vee ran(s) \subseteq ran(s_1)).$$

CT_4 客体的创建与删除:

如果一个客体处于不可用态(或可用态),且是被标名的,那么能创建(删除)该客体的主体必须满足以下条件之一:(1) 该客体不是某分级结构的根对象,且该主体能以写方式访问该客体的父对象;(2) 该客体是某分级结构的根对象,且该主体被授权许可提供这个根对象的访问许可权.即

$$\forall o \in O, named(o) \wedge [(\neg\alpha_o(o) \wedge \alpha_o^*(o)) \vee (\alpha_o(o) \wedge \neg\alpha_o^*(o))] \Rightarrow \exists s_1 \in \vartheta_o(o) \wedge \exists o_1 \in O,$$
$$\text{s.t. } (o \in H(o_1) \wedge ran(o_1) \subseteq ran(s_1)) \vee (\forall o_1 \in O, o \notin H(o_1) \wedge give(s_1, o, v)).$$

我们用 $give(s_1, o, v)$ 表示 s_1 获得了对 o 的访问授权.

如果一个客体处于不可用态,且是没被标名的,那么能创建该客体的主体必须是 TCB.即

$$\forall o \in O, \neg named(o) \wedge \neg \alpha_o(o) \wedge \alpha_o^*(o) \Rightarrow \vartheta_o(o) = TCB.$$

如果一个客体处于可用态,且是没被标名的,那么能删除该客体的主体必须满足以下条件之一:(1) 该主体是 TCB;(2) 该客体的标签范围与该主体的标签范围相同.即

$$\forall o \in O, \neg named(o) \wedge \alpha_o(o) \wedge \alpha_o^*(o) \Rightarrow \vartheta_o(o) = TCB \vee \exists s_1 \in \vartheta_o(o) \wedge ran(o) = ran(s_1).$$

CT_5 主体安全属性修改限制:

如果一个主体被认可可以修改它的标签范围,该主体必须具备下列条件:(1) 该主体是可信主体,或该主体修改完标签范围后,与它的当前访问集不发生冲突,也就是说,修改后的状态仍然满足*–性质;(2) 标签范围的最大值不超过该主体的最大安全级.

CT_6 客体安全属性修改限制:

如果一个主体 s_j 被认可把 o_j 的标签范围修改为 $[L_{\min}, L_{\max}]$,那么下列条件必须被满足:

- s_j 是可信主体,而且它的最大可视标签控制了 o_j 的最大安全标签.或者它的最大可视标签控制了 $L_{\max}, L_{\max}$ 控制了 o_j 的最大安全标签.
- 对于任何一个当前能以读、写方式访问 o_j 的主体 s,它的最大可视标签控制了 $L_{\max}$.
- o_j 的标签范围修改为 $[L_{\min}, L_{\max}]$ 之后,所得状态仍然满足*–性质.
- o_j 的标签范围修改为 $[L_{\min}, L_{\max}]$ 之后,在分级结构中的兼容性保持不变.
- 主体 s_j 被授权修改 o_j 的标签范围.

如果一个主体 s_j 被认可修改 o_j 的访问控制列表 ACL,那么 s_j 能以控制方式访问 o_j.

2.5 实现IPC保护的方法

Linux 的 IPC 类型主要包括:Pipe,FIFO,System V message queue,System V semaphore,System V shared memory,TCP socket,UDP socket 及 Unix domain socket.正如 Stevens 在其名著[19]所描述的,对应于这些类型的 IPC 对象的生命周期或者是进程级的(process-persistent),或者是内核级的(kernel-persistent),它们与主体进程的生命周期紧密相关,因此对它们的保护应该把它们与主体进程结合起来考虑,不宜采用与生命周期较长的文件系统级(filesystem-persistent)的对象相同的方法.

为了处理 IPC,在 TMach 系统中引入了一种特殊的抽象对象——联结点(connection point),它是用户在创建替代自己的主动进程的同时,为主动进程生成的.任何需要访问该进程创建的 IPC 对象的进程必须而且仅需获得访问该进程联结点的特权.这样一来,为了删除 IPC 对象必须先删除对应的联结点,而为了删除联结点必须先删除对应的主动进程.在 TMach 系统中提出的这一方法不适合于本模型,因为进程的标签在本模型中是可以动态变化的,从而在不同的条件下,同一进程将生成不同的联结点,这样,同一进程就有可能对应多个联结点.另外,文献[16]中把联结点定位为单级对象,没有考虑可信主体与非可信主体的通信问题.在文献[20]中,为处理这个问题,提出了多级 IPC 对象(例如:多级命名管道)的概念,它们只能由具有某些特权的可信主体生成,利用这个概念能有效地解决这样的应用问题:一个可信的主动进程为多个安全级不同的客户进程提供服务.但是,文献[20]中把这些多级 IPC 对象完全视为与文件系统级的对象一样的对象,从而使它们与拥有它们的主动进程分离开.我们认为,这样做如果处理得好,将花费内核资源去实现一些不必要的操作;如果处理得不好,将使这些 IPC 对象游离于主动进程的控制范围之外.

上面我们讨论了处理 IPC 对象与拥有它的主动进程的关系,这是问题的一方面,另一方面,处理 IPC 时必须充分考虑隐通道问题,因为处理 IPC 时将面临大量的共享资源.目前,许多系统(如:B2 级的 Xenix version 3 及 B3 级的 TMach 等)为了避开隐通道的困扰,采取回避不同级别进程的通信的方法;正如文献[21]中所指出的,这将使系统难以建立上面提到的多级应用.鉴于这些认识,在本模型中我们将使主动进程与 IPC 对象一体化,同时解决多级应用和隐通道问题.

2.5.1 本模型实现 IPC 保护的方法

在本模型中,我们类似于文献[14]直接考虑进程对进程的操作,而不是把它们转化为主体对客体的操作.在

进程与进程之间的操作主要有两类:协调进程调度的操作——发信号;实现进程间通信的操作——连接.IPC 对象是为完成这些操作而由相应进程生成,因此我们可以采用控制 IPC 对象所隶属的进程,从而控制 IPC 对象,这就是本模型实现 IPC 保护的基本出发点.我们的规则如下:

规则 1. 如果 IPC 对象所隶属的进程为 $proc_1$,那么进程 $proc_2$ 能读访问该 IPC 对象的条件是: $proc_1$ 能向 $proc_2$ 发信号,即 $v\_\max_s(proc_2) \geq v\_\max_s(proc_1)$.

规则 2. 如果 IPC 对象所隶属的进程为 $proc_2$,那么进程 $proc_1$ 能通过该 IPC 对象发消息的条件是: $proc_1$ 可请求连接 Westboro 进程 $proc_2$,即 $ran(proc_1) \subseteq ran(proc_2)$.

这两条规则蕴涵了如下的事实:

• client 进程被许可对 server 进程的操作是单向的,也就是说,两个进程的地位不能对调,因为条件不是对称的.这与直觉是相符的.

• 规则允许的模式只有:client 进程是非可信进程,server 进程是非可信进程;client 进程是非可信进程,server 进程是可信进程;client 进程是可信进程,server 进程是可信进程.它不允许非可信进程为可信进程提供服务.

• 规则能有效实现文献[20]中的多级 IPC 对象的思想,因为可信进程对应的就是多级 IPC 对象.

• 正如文献[20]所指出的,由于进程间通信是通过 IPC 对象发送信息,它不是把消息较长时间地存放在 IPC 对象中,因此不会出现信息泄露.具体处理的方法可参考文献[21],另外我们认为也可以使用类似于文献[15]中的多级设备的处理方法.

3 相关工作

对于 TMach 的安全模型,在早期的文献[3]中引入了多级主体,而且有类似于本模型的不变量,但是那里没有引入主体对主体的操作,只是抽象地描述了主体对主体的激活公理和终止公理.另外,它不仅没有提出对 IPC 对象的处理方法,而且没有多级客体.在文献[16]中多级主体被弃掉,但是提出了有效的实现 IPC 对象安全保护的方法,而且也有多级客体(它们在组织级的安全策略的描述中提到,但在数学模型中并没有相应地表达出来),文献[16]中的模型比较接近 BLP 模型.TMach 的安全模型不能实现安全级的非管理的动态调节,但本模型能实现,而且实现方式与文献[1]不同.在文献[14]中提到的多级安全模型虽然引入了多级主体、多级客体和主体对主体的操作模式:发信号和连接,但是没有明确提出处理 IPC 对象的方法,也没有处理安全级与文件系统的分级结构的兼容性,而且主体的安全级范围是替代它的当前安全级还是替代最大安全级,这也是不清楚的.同时,该模型中也没有动态调节安全级范围的规则.因此,它与本模型有质的不同.但是文献[14]中指出了引入多级主体的必要性,即在可信计算机系统网络中,由于系统所处环境的因素,不同系统可能授权处理不同级别的数据.在安全系统上实现 COTS 应用的实用方面的限制可能导致要求对该应用必须有一个受限制的可信范围,也就是说,存在一个范围,在该范围内该应用必须是可信的,这些都深深地影响了我们构造本模型的思路.

4 结束语

本文在分析现有多级安全模型及安全系统可能的应用环境的基础上提出了一个新的支持安全级动态变更的多级安全模型 DBLP.它与 Ott 的模型存在的相似之处是:支持安全级动态变更,但变化规则不同.在已有多级主体和多级客体的基础上,提出了具有多级属性的操作和具有单级属性的操作,推广了文献[3,14]中的不变量,这也是文献[6]中有关规则的深化.我们认为,本模型较好地实现了文献[10]中的思想与安全级动态变更的思想的结合,这不仅精简了文献[1]中所用的模型,使 Bell 模型更自然,而且这将有助于推广到处理网络的情况.本模型处理 IPC 对象的方法不仅达到了 TMach 的安全模型中获得的效果,而且在舍弃联结点对象的基础上,把文献[14]中的思想融入本模型中,借助于对进程的控制实现了对 IPC 对象的控制.本文还分析了文献[2]中 ABLP 模型的一些不足,但我们没有给出有关的改进.在以后的工作中,我们将通过改进 ABLP 模型的分析方法来给出本文的动态变更规则满足 BLP 的规则的证明.我们将另文描述本模型的规则,并给出有关的证明.另外,我们将研究如何用无干扰理论分析本文提出的动态调节规则可能导致的隐通道问题.

致谢 感谢审稿人提出的许多好的修改意见.

References:

[1] Ott A. Regel-Basierte zugriffskontrolle nach dem Generalized framework for access controlansatz am beispiel Linux. Diplomarbeit Universitat Hamburg, 1997.

[2] Shi WC. Research on and enforcement of methods of secure operating systems development. [Ph.D. Thesis] Beijing: Institute of Software, The Chinese Academy of Sciences, 2001 (in Chinese with English abstract).

[3] Mayer FL. An interpretation of refined Bell-LaPadula model for the TMach kernel. In: Proc. of the 4th Aerospace Computer Security Applications Conf. IEEE Computer Society Press, 1988. 368~378.

[4] Lunt T, Denning D, Schell R, Heckman M, Shockley W. The SeaView security model. IEEE Trans. on Software Engineering, 1990,16(6):593~607.

[5] Bell DE, La Padula LJ. Secure computer system: Unified exposition and multics interpretation. Mitre Report, MTR-2997 Rev. 1, 1976.

[6] Schell RR, Tao TF, Heckman M. Designing the GEMSOS security kernel for security and performance. In: Proc. of the 8th National Computer Security Conf. 1985. 108~119.

[7] Lee TMP. Using mandatory integrity to enforce commercial security. In: Proc. of the IEEE Symp. on Security and Privacy. IEEE Computer Society Press, 1988. 140~146.

[8] Clark DD, Wilson DR. A comparison of commercial and military security policies. In: Proc. of the 1987 IEEE Symp. on Research in Security and Privacy. IEEE Computer Society Press, 1987. 184~238.

[9] Rushbyc J. Design and verification of secure systems. ACM Operating Sysstem Review, 1981,15(5):12~21.

[10] Bell DE. Security policy modeling for the next-generation packet switch. In: Proc. of the IEEE Symp. on Security and Privacy. IEEE Computer Society Press, 1988. 212~216.

[11] Kargar PA, Austel V, Toll D. A new mandatory security policy combining secrecy and integrity. IBM Research Report, RC21717, 2000.

[12] Loscocco PA, Smalley SD. Integrating flexible support for security policies into the Linux operating system. Technical Report, NAI Labs, 2001.

[13] McLean J. The algebra of security. In: Proc. of the IEEE 1988 Symp. on Research in Security and Privacy. IEEE Computer Society Press, 1988. 2~7.

[14] Secure Computing Corporation. Assurance in the Fluke microkernel: Formal top-level specification. CDRL A004. Technical Report, Secure Computing Corporation, 1999.

[15] Data General. Managing security on DG/UX system, manual 093-701138-o4. Westboro: Data General Corporation, MA01580, 1996.

[16] Trusted Information System, Inc,. Trusted mach mathematical model. Technical Report, TIS tmach EDOC-0017-96B, Trusted Information System, Inc, 1996.

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

[18] Thomas T. A mandatory access control mechanism for the UNIX file system. In: Proc. of the 4th Aerospace Computer Security Applications Conf. IEEE Computer Society Press, 1988. 173~177.

[19] Stevens WR. UNIX Network Programming. Volume 2: Interprocess Communications. Prentice-Hall, Inc., 1999.

[20] Sutton S, Hinrichs S, Inskeep T. MISSI B-level windows NT feasibility study. Final Report, MISSI, MISSI Contract MDA904-95-C-4088, 1996.

[21] Parenty TJ. The incorporation of multi-level IPC into UNIX. In: Proc. of the IEEE Symp. on Security and Privacy. IEEE Computer Society Press, 1989. 94~99.

附中文参考文献:

[2] 石文昌.安全操作系统开发方法的研究与实施[博士学位论文].北京:中国科学院软件研究所,2001.