基于信息流源的访问控制研究

李伟楠 李翰超 石文昌
(中国人民大学数据工程与知识工程教育部重点实验室 北京 100872)
(中国人民大学信息学院 北京 100872)

摘 要 鉴于信息流对系统完整性的影响,探讨结合信息流实施访问控制的方法,提出一种基于信息流源的访问控制(ACSIF)模型,其中,信息流源指位于信息流出发点的实体。借助用户集合表示信息流源,利用信息流源描述完整性级别,根据集合包含关系定义完整性级别的支配关系,基于信息流构造访问控制规则。通过引入完整性约束主体和约束集合增强模型的适用性。模型的实现可有效地利用系统已有信息,以降低系统配置的复杂度,提高系统的实用性。

关键词 访问控制,信息流,信息流源,实用性
中图法分类号 TP309 **文献标识码** A

Research on Access Control Based on Source of Information Flow

LI Wei-nan LI Han-chao SHI Wen-chang
(Key Lab of Data Engineering and Knowledge Engineering of Ministry of Education, Renmin University of China, Beijing 100872, China)
(School of Information, Renmin University of China, Beijing 100872, China)

Abstract Due to the impact of information flow on system integrity, explored ways to enforce access control with consideration of information flow. Proposed a model for Access Control based on Source of Information Flow (ACSIF), in which, source of information flow refers to the entity that is a starting point of the flow. Source of information flow was expressed via set of users. Integrity level was described through sources of information flows. Dominance relationship of integrity levels was defined according to inclusion relationship of sets. Access control rules were constructed based on information flows. Integrity constrained subjects and constrained sets were introduced to improve adaptability of the model. Existing system information may be utilized effectively in implementing the model, hence, complexity of system configuration can be reduced, and system usability can be improved.

Keywords Access control, Information flow, Source of information flow, Usability

研究与设计可信计算平台,是当前信息安全领域的热点,它涉及到信息空间的方方面面。其中,系统完整性便是构建可信环境的关键因素之一。从上世纪70年代末G. H. Nibaldi建立可信计算基(TCB)[1]思想开始,几十年来,大量学者基于系统完整性度量展开研究,提出了各种解决方法,如基于hash值的静态完整性度量、基于信息流的完整性度量以及基于控制流的完整性度量等等。

文献[2]所描述的证明框架表明,在构建可信计算平台时,为了保证安全性和完整性,系统必须建立一个基础的自保护安全机制。毫无疑问,访问控制便是该安全机制的一个重点。最早的基于完整性的访问控制模型是1977年K. J. Biba提出的Biba模型[3]。虽然Biba模型的理论描述精确,但在现实系统中无法完全实现,无法达到完整性要求和系统的实用性要求之间的平衡。TE模型[4]提出了基于域和类型实施访问控制的思想;基于信息流的格模型[5]则从形式化定义的角度给出了系统中信息流的表示方法;Clark-Wilson模型(C-W)[6]提出了基于事务的完整性保护规则;CW-Lite模型[7]则是Clark-Wilson模型的轻量级版本,提高了模型的实用性。随着分布式环境的普及,为了满足多用户控制的要求,分布式访问控制模型应运而生。DLM模型[8]以满足分布式环境中信息机密性保护的要求为出发点,提出了一种基于集合标识机密性的方法,简化了控制策略的定义;DIFC[9]则提出一种基于不同的数据所有者实施分布式信息流控制的模型定义方法。

访问控制作为操作系统安全机制的重要组成部分,是基于信息流进行系统完整性度量的基本支撑。2006年Jaeger T.等人提出的PRIMA度量体系[10]是基于信息流的完整性度量工作的一项出色成果。PRIMA通过分析系统的安全策略获取信息流图,并利用CW-Lite模型的规则实现基于信息流的系统完整性度量。PRIMA以TE模型和SELinux[11]作为基础支撑,但基于TE模型的SELinux机制的配置非常复杂,很不实用。

到稿日期:2010-04-09 返修日期:2010-07-19 本文受国家863计划课题(2007AA01Z414),国家自然科学基金项目(60873213,60703103),北京市自然科学基金项目(4082018),上海市智能信息处理重点实验室开放课题(IIPL-09-006)资助。

李伟楠(1986—),女,硕士生,主要研究方向为信息安全、操作系统安全,E-mail:lwns1023@163.com;**李翰超**(1986—),男,硕士生,主要研究方向为信息安全、操作系统安全;**石文昌**(1964—),男,博士,教授,博士生导师,主要研究方向为信息安全、可信计算、系统软件,E-mail:wenchangshi@pku.org.cn(通信作者)。

本文以适度安全为原则，从实用性的角度出发，研究基于信息流源的访问控制（ACSIF，Access Control based on Source of Information Flow）问题（关于信息流源的定义请参见本文定义1），提出ACSIF模型并分析其现实意义。本文第1节通过给出核心概念的定义描述ACSIF模型的基本思想；第2节构造ACSIF模型的访问控制规则；第3节分析ACSIF模型的安全性、实用性和现实意义；第4节讨论与本文相关的工作；最后对全文进行总结。

1 ACSIF模型描述

1.1 符号约定

与访问控制行为相关的实体包括主体和客体。实体指系统中的对象，用 e 表示某个实体，用 E 表示所有实体的集合。

主体是访问操作的实施者，如用户执行某段代码产生的进程。用 s 表示某个主体，用 S 表示所有主体的集合。

客体是访问操作的实施对象，如文件、共享内存、进程或其他数据。用 o 表示某个客体，用 O 表示所有客体的集合。

用 u 表示系统中的某个用户，用 U 表示所有用户的集合。

模型目前主要考虑两类访问操作：读操作和写操作。用 $observe(s,o)$ 表示 s 读 o，用 $modify(s,o)$ 表示 s 写 o。

1.2 信息流和信息流源的定义

实体的完整性是指实体没有受到未授权用户修改，也没有受到授权用户以非授权的方式修改。实体的完整性很大程度上依赖于对实体所进行的操作，从信息流的角度看，即依赖于实体所接收信息的信息来源。ACSIF模型借助信息流源描述实体的完整性级别，能够有效地反映实体完整性所受到的影响。

下面，首先定义信息流和信息流源。

定义1（信息流和信息流源） 信息流指信息从一个实体到另一个实体的传递。处于信息流出发点位置的实体称为信息流源，处于信息流终点位置的实体称为信息流目的地。如果信息流源与信息流目的地之间没有其他实体，则称这样的信息流为直接信息流，其他情形的信息流称为间接信息流。当主体读或者写客体时，两者之间的信息流属于直接信息流。一个信息流相对于信息流源而言称为信息流出，相对于信息流目的地而言称为信息流入。

ACSIF模型基于信息流源为系统实体定义完整性级别。

1.3 实体完整性级别的描述

ACSIF模型根据信息流源定义实体完整性级别。信息流源的表示方式多种多样，我们选取用户集合来描述流入实体的信息的来源，并基于用户集合实施访问控制。

定义2（完整性级别） 实体 e 的完整性级别（$L_i(e)$）定义为按特定规则与 e 对应的用户的集合。实体可以是主体，也可以是客体。主体 s 的完整性级别指 s 的属主以及能够与 s 传递信息的其他主体和客体所对应的属主的集合。客体 o 的完整性级别指 o 的属主以及能够修改 o 的主体所对应的属主的集合。一般而言，集合中元素越多，完整性级别越低；集合中元素越少，完整性级别越高。

从信息流的角度来说，实体的完整性级别包括了流入该实体的信息的所有信息流源所对应的用户，即能够修改该实体的所有用户。它实际上代表了系统实体被感染的程度。

我们认为，所有能够修改系统实体的用户，都可能会破坏该实体的完整性，是实体完整性的潜在威胁。我们称对实体所作的修改为感染。对实体实施的修改越多，实体的完整性遭受的威胁就越大，这种威胁的程度称为感染程度。感染程度反映了修改实体的用户的多少，在包含关系成立的条件下，用户集合越大，系统实体完整性受威胁的程度越高，感染程度就越高，同时表示它的完整性级别越低。

定义3（阈值完整性级别和即时完整性级别） 模型为实体定义两类完整性级别：一类是实体 e 的阈值完整性级别（$L_u(e)$），它包含了 e 自身的属主以及能够向 e 传递信息的所有其他实体的属主，表示 e 可能被感染的最大程度；另一类是实体 e 的即时完整性级别（$L_{ic}(e)$），它包含了 e 自身的属主以及已经向 e 传递信息的所有其他实体的属主，表示 e 当前被感染的程度。即时完整性级别在读/写操作之后会动态更新，更新操作记为 $update(L_{ic}(e))$。

典型的一类主体是系统中的用户运行一段代码所生成的进程。以进程为例，我们根据进程属主为其设定完整性级别。进程属主以及能够向该进程传递信息的所有其他主体和客体的属主定义为该进程的阈值完整性级别。进程的即时完整性级别初始值定义为执行代码的进程属主或者继承自其父进程。在读操作执行后，根据完整性级别更新规则更新进程的即时完整性级别，以达到动态控制的目的。客体的阈值完整性级别包括创建客体的用户以及对客体进行读/写操作的主体的属主。客体的即时完整性级别为实际对客体执行过修改操作的所有主体的即时完整性级别的并集，初始值设置为客体属主或者继承自创建客体的主体。

例如，设主体 s_1 的阈值完整性级别为 $L_{iu}(s_1)=\{u_a,u_b,u_c\}$，客体 o_1 的阈值完整性级别为 $L_{iu}(o_1)=\{u_a,u_b\}$，表示在系统中主体 s_1 的属主只能接收用户 u_a，u_b 和 u_c 的信息，而客体 o_1 只能被用户 u_a 和 u_b 对应的主体修改。假设主体 s_1 的即时完整性级别为 $L_{ic}(s_1)=\{u_a,u_b\}$，则表示 s_1 当前已经接收了用户 u_a 和 u_b 的信息，主体 s_1 的初始即时完整性级别设置为启动该主体的用户，或者是直接继承创建它的实体的即时完整性级别，如 $L_{ic}(s_1)=\{u_a\}$，代表 u_a 执行了启动进程的操作。若客体 o_1 的即时完整性级别为 $L_{ic}(o_1)=\{u_a,u_b\}$，则表示客体 o_1 已经接收了用户 u_a 和 u_b 所拥有的主体的信息，客体 o_1 的初始即时完整性级别等于创建该客体的主体的即时完整性级别。

之所以选择用户集合表示实体完整性级别，是因为实体对应的创建者和启动者是访问操作的初始驱动者，也是系统中所有信息的初始传递者。在确定实体的阈值完整性级别时，实际上是规定了能够访问该实体的所有授权用户，因此所有不在该集合中的非授权用户都不能访问该实体。从信息流的角度来说，实体的阈值完整性级别标识了能够流入该实体的信息的所有信息流源的属主，即实体的最低完整性级别。而即时完整性级别则为授权用户的访问行为定义了动态约束条件，即所有的授权访问是否能够实施，将随着实体的即时完整性级别的改变而进行动态判定。从信息流的角度来说，实体的即时完整性级别标识了实体已经接收的信息的信息流源的属主，即主体的实际完整性级别。

如图1所示，ACSIF模型利用信息流源为主体和客体确定阈值完整性级别 L_u 和即时完整性级别 L_{ic}，图中使用菱形表示实体的完整性级别。模型按照访问控制策略的判定结果执行读/写操作后，动态更新主体和客体的即时完整性级别。

图中直线箭头表示读/写操作，曲线箭头表示完整性级别更新操作。此外，为了支持系统中的合法应用，我们定义了一类特殊的完整性约束主体，并为约束主体添加约束集合作为其附加属性。完整性约束主体需要严格定义并被验证，并非所有主体都能够声明为约束主体，图中使用虚线菱形框表示完整性约束主体所特有的约束集合（关于完整性约束主体和约束集合的定义参见定义 5）。

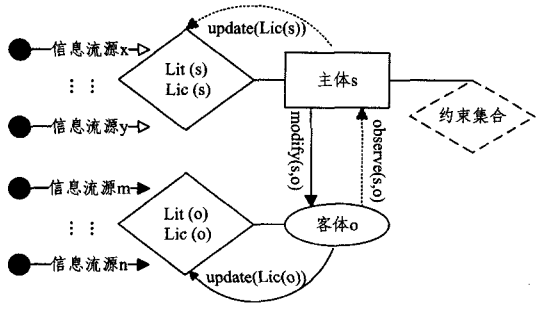

图 1　ACSIF 模型的基本思想

2　访问控制策略

2.1　访问控制规则

模型规定，信息流只能从高完整性级别的实体流向低完整性级别的实体。

为了描述访问控制规则，模型定义完整性级别的支配关系。

定义 4（支配关系）　设两实体 e_1 和 e_2 的完整性级别分别为 $L_i(e_1)$ 和 $L_i(e_2)$，定义支配关系（$\sqsupseteq$）为 $L_i(e_1)\sqsupseteq L_i(e_2)$ 当且仅当 $L_i(e_1)\subseteq L_i(e_2)$。

若实体 e_1 的完整性级别能够支配实体 e_2 的完整性级别，则表示实体 e_1 的完整性级别比实体 e_2 高，那么信息可以从支配实体 e_1 流向被支配实体 e_2。

规则 1（读操作规则）　设 s 为主体，o 为客体，$observe(s,o)$ 为读操作，则 $observe(s,o)$ 的条件是 $L_{ic}(o)\sqsupseteq L_{it}(s)$，其中 $L_{it}(s)\subseteq U, L_{ic}(o)\subseteq U$。

规则 1 控制系统中的读操作，主体 s 读客体 o，信息流的流向是从 o 到 s，主体的完整性可能会被破坏。这里，读操作关注的重点是作为信息流源的客体当前已经被多少用户所修改，即客体被感染的程度。同时判定当前感染程度是否能够被主体容忍，即主体的阈值完整性级别 $L_{it}(s)$ 是否被客体的即时完整性级别 $L_{ic}(o)$ 支配。若是，则说明客体 o 对主体 s 来说是一个合法的信息流源，主体可以接收客体流入的信息，允许读操作进行；否则，拒绝读操作。

规则 2（写操作规则）　设 s 为主体，o 为客体，$modify(s,o)$ 为写操作，则 $modify(s,o)$ 的条件为 $L_{ic}(s)\sqsupseteq L_{it}(o)$；其中，$L_{ic}(s)\subseteq U, L_{it}(o)\subseteq U$。

规则 2 控制系统中的写操作，主体 s 写客体 o，信息流的流向是从 s 到 o，客体的完整性可能会被破坏。这里，模型关注的重点则是作为信息流源的主体当前已经接收了多少用户的信息，即主体被感染的程度。同时判定当前感染程度是否能够被客体容忍，即客体的阈值完整性级别 $L_{it}(o)$ 是否被主体即时完整性级别 $L_{ic}(s)$ 支配。若是，则说明主体 s 对客体 o 来说是一个合法的信息流源，主体可以修改当前客体，允许写操作进行；否则，拒绝写操作。

2.2　即时完整性级别更新规则

为了实现动态控制，使得系统实体的完整性级别在访问行为结束后能够自动更新，模型分别定义了两类完整性级别的更新操作。阈值完整性级别 $L_{it}(e)$ 是一个静态完整性级别，根据策略要求在初始状态确定，在系统运行过程中不能被随意修改。而即时完整性级别 $L_{ic}(e)$ 是一个动态完整性级别，反映的是实体随着读写操作行为的进行其自身完整性级别的变化，它记录了流入实体的所有信息的信息流源。

约定 1（最小上界运算符）　假设 $L_x(e_1)$ 和 $L_y(e_2)$ 是实体 e_1 和 e_2 的完整性级别，用 $H(L_x(e_1),L_y(e_2))$ 表示 $L_x(e_1)$ 和 $L_y(e_2)$ 两个集合的最小上界，即 $H(L_x(e_1),L_y(e_2))=L_x(e_1)\cup L_y(e_2)$。

规则 3（主体即时完整性级别更新规则）　若 $observe(s,o)$，则在读操作结束后，对主体 s 的即时完整性级别执行更新操作 $update(L_{ic}(s))$，更新后有 $L_{ic}(s)=H(L_{ic}(o),L_{ic}(s))$，其中 $L_{ic}(s)\subseteq U, L_{ic}(o)\subseteq U$。

规则 4（客体即时完整性级别更新规则）　若 $modify(s,o)$，则在写操作结束后，对客体 o 的即时完整性级别执行更新操作 $update(L_{ic}(o))$，更新后有 $L_{ic}(o)=H(L_{ic}(o),L_{ic}(s))$，其中 $L_{ic}(s)\subseteq U, L_{ic}(o)\subseteq U$。

以上两个完整性级别的更新规则，主要是为了保证在运行的过程中能够准确描述流入实体的信息流源，刻画系统中的信息流动，反映当前系统中实体的实际完整性。信息流入是造成实体完整性可能被破坏的因素，信息流入对实体的影响是持续的，静态的完整性表示方法无法反映这种持续性。为此，我们定义 $L_{ic}(e)$ 的更新规则，记录流入实体的信息对实体完整性的影响，进而限制实体下一步的信息流出。当主体 s 对客体 o 执行写操作时，信息流从 s 流向 o，则 o 的完整性可能会被 s 破坏，我们利用 s 的即时完整性级别来更新 o 的即时完整性级别；而当主体 s 读客体 o 时，信息流从 o 流向 s，此时 s 的完整性可能会被 o 破坏，我们利用 o 的即时完整性级别来更新 s 的即时完整性级别。

2.3　完整性约束主体和约束集合

按照上两节定义的规则，系统中的读写操作行为严格按照完整性保护的要求进行授权判定。但是，约束规则过于严格，将使 ACSIF 模型的实用性受到限制。比如，设有网络用户 u_{net}，从网络下载的数据对象 o_{net} 的完整性级别 $L_{it}(o_{net})$ 定义为 $\{u_{net},u_1,u_2\}$，$L_{ic}(o_{net})$ 为 $\{u_{net}\}$。但实际系统在与网络交互之前，任何一个主体 s 的阈值完整性级别 $L_{it}(s)$ 都不会包含用户 u_{net}，即 $L_{ic}(o_{net})$ 不可能支配 $L_{it}(s)$，这样会造成凡是网络数据都不可读的结果，这在实际的系统使用中是不合理的。

为了接收来自网络的信息，假设事先定义一些可以接收网络数据的主体 s_{net}，完整性级别 $L_{it}(s_{net})$ 定义为 $\{u_{net},u_1,u_2,u_3\}$。主体 s_{net} 在接收了网络数据后即时完整性级别 $L_{ic}(s_{net})$ 中便会包含 u_{net}，即 $u_{net}\in L_{ic}(s_{net})$。此时，根据模型访问控制规则的定义，进程 s_{net} 仍然无法与系统其他进程或者文件进行交互，这会造成一些合法的网络信息无法在系统中传递，导致系统不能被正常使用。

为了解决这一问题，提高模型的实用性，本文定义完整性约束主体和约束集合，用来限制一些信息流源对实体完整性造成的影响，从而支持合理的网络信息传递请求。

定义 5（完整性约束主体和约束集合）　完整性约束主体（s_i）是一类特殊的主体，每个这样的主体有一个确定的用户集合（$set(s_i)$）与之对应。该主体在接受客体 o 的信息流入时，对其即时完整性级别按以下方式执行更新操作 $update$

$(L_{ic}(s_i))$;$L_{ic}(s_i)=H((L_{ic}(o)-set(s_i)),L_{ic}(s_i))$。其中用户集合($set(s_i)$)称为约束集合,它所包含的用户在一定条件下不会对系统实体的完整性造成威胁。完整性约束主体和约束集合在系统初始状态时确定。

完整性约束主体实际上是一类具有提升自身完整性级别的特权主体。定义了完整性约束主体后,需要为完整性约束主体定义额外的完整性级别更新规则。

规则 5(完整性约束主体即时完整性级别更新规则) 设 s_i 为完整性约束主体,o 为客体,$observe(s_i,o)$为读操作,则 $observe(s_i,o)$的条件是 $L_{ic}(o)\supseteq L_{it}(s_i)$。读操作结束后,对 s_i 的即时完整性级别执行更新操作 $update(L_{ic}(s_i))$,有 $L_{ic}(s_i)=H((L_{ic}(o)-set(s_i)),L_{ic}(s_i))$,其中 $L_{it}(s_i)\subseteq U$,$L_{ic}(o)\subseteq U$,$L_{ic}(s_i)\subseteq U$。

如果完整性约束主体 s_i 接收来自实体 e 的信息,它将不会被约束集合 $set(s_i)$中的用户所感染。例如,假设 s_{i1} 为完整性约束主体,e_1 为实体,且 $L_{it}(s_{i1})=\{u_{net},u_1,u_2\}$,$L_{ic}(s_{i1})=\{u_1,u_2\}$,$L_{ic}(e_1)=\{u_{net}\}$,$set(s_{i1})=\{u_{net}\}$。根据规则 5,完整性约束主体 s_{i1} 读取实体 e_1 的信息后,其即时完整性级别更新为 $update(L_{ic}(s_{i1}))=H((L_{ic}(e_1)-set(s_{i1})),L_{ic}(s_{i1}))$,即有 $u_{net}\notin update(L_{ic}(s_{i1}))$。在实际系统中,完整性约束主体可以定义为一些网络进程,如 openSSH,它不会因为接受合法的网络输入而被感染,而它的输出信息的感染程度也不会因为接收了网络输入而增大。

由于完整性约束主体可以消除从一些特定的信息流源流入的信息对该主体的完整性的影响,我们必须谨慎定义,以防止完整性约束主体对系统完整性造成危害。一般来说,完整性约束主体的设置必须经过严格验证,Clark-Wilson[6]采用的形式化验证方法就是一个例子。但是,形式化验证难度很大,在实际系统中难以实现。ACSIF 模型在系统实现时可以考虑借助诸如 Patos[12]的进程完整性度量方法来验证完整性约束主体。

3 模型分析

3.1 安全性分析

ACSIF 模型选择使用信息流源描述实体的完整性级别,主要是为了能够更加简便地描述系统中的信息流动,为基于信息流的完整性度量提供一个实用的访问控制支撑模型。

模型使用用户集合表示流入实体的信息所对应的信息流源,进而反映系统实体的完整性级别。阈值完整性级别定义了所有拥有访问该实体权限的授权用户,而即时完整性级别则对授权用户的访问方式进行动态约束。从信息流的角度出发,模型只允许信息从高完整性的实体流向低完整性的实体,从而保证信息不会被非法篡改。

ACSIF 模型中定义的读/写访问规则能够保证系统从初始状态起,只接收从安全策略中定义的合法信息流源流入的信息。初始状态时,主体的初始即时完整性级别继承自创建或执行当前主体的其他主体的即时完整性级别,而客体的初始即时完整性级别则与创建该客体的主体的即时完整性级别相同。

在系统运行过程中,读操作规则保证流入主体 s 的信息来自于合法的信息流源。当 s 读客体 o 时,形成从 o 到 s 的信息流。主体 s 的阈值完整性级别 $L_{it}(s)$表示流入 s 的信息所对应的信息源的属主必须包含在 $L_{it}(s)$中。此时,若客体 o 希望向 s 传递信息,该读操作必须满足条件:o 是 s 的一个合法信息流源,即 o 的属主必须包含在 $L_{it}(s)$中。同时,我们需要考虑到信息流的连续传递性。假设实体 e_1 向 e_2 传递信息后,e_2 又向 e_3 传递信息,那么可以形成从 e_1 经过 e_2 传递给 e_3 的间接信息流。这时,除了 e_2 之外,e_1 也是 e_3 的一个间接信息流源,即 e_1 可以影响 e_2 并通过 e_2 间接影响 e_3 的完整性。为了解决这一问题,我们必须考虑间接信息流源因素。庆幸的是,依据即时完整性级别更新规则,e_3 的间接信息流源的属主都会包含在 e_3 的直接信息流源的即时完整性级别中。据此,我们重新修改读操作,需要满足的条件为客体 o 的即时完整性级别 $L_{ic}(o)$是 s 的阈值完整性级别 $L_{it}(s)$的子集。这正是 ACSIF 模型读操作规则所定义的判定条件。若 $L_{ic}(o)$中存在一个用户 u_x,且 u_x 不属于 $L_{it}(s)$,则根据读规则,s 无法读 o,即非法的信息流源都无法传递信息给 s,从而保证 s 的完整性不被破坏。

写操作规则保证流入客体 o 的信息都来自于合法的信息流源。当主体 s 写客体 o 时,形成从主体 s 流入客体 o 的信息流,客体 o 的阈值完整性级别 $L_{it}(o)$表示所有流入 o 的信息的信息流源的属主都必须包含在 $L_{it}(o)$中。此时,若 s 希望向 o 传递信息,该写操作必须满足条件:s 是 o 的一个合法信息流源,即 s 的属主必须包含在 $L_{it}(o)$中。与读操作相同,需要考虑到信息流的连续传递性,加入间接信息流源对写操作进行控制。我们重新修改写操作,需要满足的条件为主体 s 的即时完整性级别 $L_{ic}(s)$是客体 o 的阈值完整性级别 $L_{it}(o)$的子集。而这正是 ACSIF 模型写操作规则所定义的判定条件。若 $L_{ic}(s)$中存在用户 u_y 且 u_y 不属于 $L_{it}(o)$,则根据写规则,s 无法写 o,即非法的信息流源无法篡改 o,从而保证客体 o 的完整性。

ACSIF 模型利用即时完整性级别的更新保留信息流源对实体完整性的影响。它根据读操作和写操作动态修改实体的即时完整性级别,而变化后的即时完整性级别又反过来影响实体下一步读操作和写操作的访问判定结果,从而更加精确地描述系统的当前状态。

ACSIF 模型为了提高实用性,定义了完整性约束主体和约束集合。读写访问控制规则过于严格,使得模型不能支持一些合法应用,最常见的例子就是网络应用。完整性约束主体通过定义一个约束集合来提升自身即时完整性级别,在一定程度上可以支持一些必要的应用。比如,模型定义一部分与网络交互的主体如 ftpd 等为完整性约束主体,则该主体接收从网络进程流入的网络信息后,其完整性级别不会降低,仍能够与系统内其他实体交互。但完整性约束主体在一定程度上降低了模型的安全性能,完整性约束主体若接收了有害信息,则可能会影响系统其他实体的完整性。Clark-Wilson 模型[6]利用形式化验证的方式解决安全隐患,但该方法在实际系统中很难实现。ACSIF 模型利用 Patos[12]中提出的进程完整性度量方法实现约束主体的验证,但该度量方法重点考虑的是进程的静态数据段,而对于进程的动态信息缺乏全面的验证,在安全性上要比 Clark-Wilson 模型低。

ACSIF 模型可以实现交互过程的隔离。举例来说,假设存在用户 u_a,u_b 和 u_c,u_a 和 u_b 可以交互,u_b 和 u_c 可以交互,但 u_a 和 u_c 不可交互,在传统系统中,u_b 可能会将 u_a 的信息泄露给 u_c。ACSIF 模型可以解决这一问题。当 u_a 向 u_b 传递信息时,该信息保存在一个属主为 u_b 的客体 o_{ab} 中,o_{ab} 接收信

息后，其即时完整性级别更新为 $L_{ic}(o_{ab})=\{u_a,u_b\}$。假设此时用户 u_c 启动的主体 s_c 要读取该信息，按照访问控制规则，需要满足条件 $L_{ic}(o_{ab})\supseteq L_{it}(s_c)$。而又已知 u_a 和 u_c 不可交互，即 $u_a\notin L_{it}(s_c)$，则有 $L_{ic}(o_{ab})\subseteq L_{it}(s_c)$不可能成立。因此，$s_c$ 无法读取 o_{ab}，即用户 u_c 无法读取 u_a 的信息，用户 u_a 与 u_b 的信息交互过程与用户 u_b 与 u_c 的信息交互过程是相互隔离的。

3.2 实用性考虑

ACSIF 模型的策略实施相对 TE 模型[4]要简单得多。

首先，模型中定义的完整性级别是由用户集合来表示的，并且实体的即时完整性级别在初始时设定为主体和客体各自的属主或者是继承其父辈实体的完整性级别。因此，在模型实施时可以借用系统中已有的自主访问控制信息，如访问控制列表(ACL)中的用户集合，获取实体的初始即时完整性级别。而对于阈值完整性级别，其中一部分信息也可以从访问控制列表中获取。假设文件的属主和用户组包含用户 u_a，u_b 和 u_c，则其阈值完整性级别同样包含这 3 个用户。阈值完整性级别在系统初始状态时确定。在系统实现时，系统开发人员确定应用系统中的主体能够与哪些实体进行信息交互；若开发人员特别定义，阈值完整性级别可以设置为极大值，即所有用户集合 U。特殊地，如 *init* 进程的完整性级别设置为 $L_{ic}(init)=\emptyset$，$L_{it}(init)=U$，并且该完整性级别在执行过程中不可改变。而对于完整性约束主体，则需要在设定阈值完整性级别和即时完整性级别之外，设定额外的约束集合，以保证一些与网络交互的进程能够执行正常的操作。

其次，ACSIF 模型基于信息流为实体的即时完整性级别定义了更新规则。实体的初始即时完整性级别可以利用访问控制列表的信息获取。而在运行过程中，随着读写访问操作的实施，即时完整性级别可以动态地进行修改，以降低安全策略制定的复杂度，并实时地反映系统信息的流动情况。

可以看出，ACSIF 模型实现了强制访问控制，在策略实施过程中可以借助系统已有的信息，并满足完整性级别更新的动态实时性要求，其实用性相比 TE 模型有较大提高。

3.3 模型对基于信息流的完整性度量的支撑

ACSIF 模型能够清晰地描述系统信息流。模型利用用户集合表示信息流源，并根据信息流的要求定义访问控制规则，刻画各个用户之间信息流动的轨迹。通过分析模型的访问控制策略，可以获取系统的合法信息流图；而通过获取当前系统中实体的即时完整性级别，也可以刻画出系统当前的实际信息流图。利用这两个信息流图，我们可以获得度量目标所对应的信息流源，从而确定需要度量的对象集合，减少不必要的度量操作。与 TE 模型相比，ACSIF 模型能够为基于信息流的完整性度量提供更加直接和有针对性的支撑，并且大大简化了安全策略的配置。

另外，ACSIF 模型使用用户集合反映实体的完整性级别，该完整性级别可以实现用户之间的隔离。这种隔离功能为保护 TBOS[13]的度量机制提供了一种新的思路。我们可以将度量机制作为一个独立的交互模块，度量机制中与度量相关的主体和客体的完整性级别所包含的用户是一些特别定义的用户，而这些用户与外界其他的对象之间不能随意传递信息。例如，度量主体和被度量客体均被度量管理员(Measure-admin)所拥有，大部分度量实体的初始即时完整性级别 $L_{ic}(e)$和阈值完整性级别 $L_{it}(e)$可以设置为 $L_{ic}(e)=L_{it}(e)=\{Measure\text{-}admin\}$，这样可以保证度量机制内部的度量主体与被度量客体不被外界对象破坏，从而实现安全交互。少数特别定义的与外界交互的度量主体，可以单独设置 $L_{ic}(e)$和 $L_{it}(e)$。

4 相关工作

从 20 世纪 70 年代开始，研究人员提出了大量的访问控制模型。这些模型分别针对某些实际的问题，从不同的角度定义系统访问控制的规则，在一定程度上满足了系统的安全要求。如表 1 所列，我们从模型设计的复杂度、对度量的支撑度、模型的安全程度以及策略实施的实用性上对不同的访问控制模型进行了对比。

TE 模型[4,20]划分系统中的主体和客体的域和类型，并以它们作为访问行为实施的基本单位。然而，对系统所有的主客体划分域和类型的操作过于复杂，TE 相应的 SELinux 机制[11]也缺乏易用性。Prima 系统[10]基于该模型实施度量，其安全策略的配置为系统管理员带来了较大负担。ACSIF 模型利用系统已有的自主访问控制信息，简化了访问控制策略的配置，虽然在安全性上不如 TE，但在模型设计和策略实现上提高了实用性。

ACSIF 模型使用用户集合作为完整性级别的表征。该思想的一个来源是 DLM 模型[8]，它利用系统对象的读者列表来表示实体的敏感程度。但 DLM 只是记录了对象在读写过程中读者集合的变化，而且该模型的侧重点在于实现分布式环境下多用户对数据共同控制的思想。ACSIF 模型更强调首先在单机系统中实现强制访问控制，在系统初始状态便设定了一个阈值完整性级别约束系统中的信息流。这点上，ACSIF 模型要比 DLM 更严格，但它不能解决分布式环境下的问题。IFEDAC 模型[14]同样使用用户集合来表示完整性级别，但不同的是，ACSIF 模型是一个强制访问控制模型，其完整性级别在系统初始状态时确定，而 IFEDAC 则是一个自主访问控制模型，访问行为最终仍是由属主控制。ACSIF 面向信息流，设计的目标是为系统完整性度量提供支撑，它重新定义了信息流，并将用户集合作为信息流源的一种特殊实现方法。

UCON 模型[15]是一个使用控制模型，它利用授权、义务和条件 3 部分共同约束系统中的访问行为，并着重强调对访问请求需要进行持续判定。尽管 UCON 模型框架几乎可以涵盖以前的所有访问控制模型，但在解决特定的诸如数字版权、网格安全[16]等问题时，仍需要详细定义其 3 个组件的构成。而在描述完整性方面，UCON 模型强调对象的使用权限不会被滥用。ACSIF 模型继承了 UCON 模型的框架，在模型设计时，利用用户集合实施授权判定，提高实用性；借助完整性级别的更新规则，实现动态控制。模型主要的目标是依赖信息流定义读写规则，从而清晰地描述系统实体之间的信息流动，为完整性度量提供支撑。

Histar 系统[17]和 Asbestos 系统[18]主要利用了分布式信息流控制(DIFC)[9]的思想。它们通过安全属性的设置实现分区控制，将每一次信息交互作为一个独立的通信过程，通过对单个过程中访问操作的控制来解决系统的泄密问题。虽然两个系统在实现时并不是针对完整性，但 ACSIF 模型仍可以借助其分区控制的思想。Histar 每个信息交互过程都有其对应的标识，在该标识之下设定完整性级别进行访问行为控制，其他不具备该标识的实体都无法获取该交互过程中的信息。同时，这两个系统都是基于信息流实现访问控制，即安全属性

的判定不仅仅体现在初始的安全策略定义中，还体现在之后安全属性的变化引起的信息流的变化上。Histar 修改了系统内核的对象定义，着重在于减少可信代码的数量，利用尽可能小的可信部分实现系统控制，其信息流定义的粒度较细。而 ACSIF 模型在系统实现上的目标在于提高实用性，其实现要求尽可能少地改动原有系统，同时尽可能多地借助系统已有信息。

UMIP[19] 系统是一项以实用性为出发点的完整性保护工作，但它主要是为了降低网络威胁对系统完整性所造成的影响。UMIP 实现的访问控制模型中只定义了两个完整性级别，即高完整性级别和低完整性级别，无法准确反映系统实体的感染程度。它定义了 3 类主要的网络接口进程作为特权主体，利用这 3 类接口接收低完整性级别的网络信息，并对该信息进行标记。但 UMIP 没有考虑如何去验证这些网络信息的可信性。

CW-Lite 模型[7,20] 抛弃了 Clark-Wilson 模型中对良构事务的形式化验证的要求，定义了一种轻量级的信息流控制模型。它提出的过滤主体的思想对于提高系统实用性以及在安全性和实用性之间获得平衡有重要作用。ACSIF 模型为了支持系统中的合法应用，通过添加约束集合的方式定义了一类特殊的完整性约束主体。模型修改即时完整性级别更新规则，提升完整性约束主体的完整性级别，进而允许一些从低完整性实体流向高完整性实体的信息流，并利用诸如 Patos[12] 的进程完整性度量方法对该主体实施可信性验证，保证其接收的低完整性级别的信息不会破坏系统安全。ACSIF 模型在一定程度上实现了过滤思想，具有较强的可操作性。关联模型分析如表 1 所列。

表 1　关联模型分析

★和☆代表程度，★＝2☆

模型	复杂度	度量支撑度	安全度	实用性
Biba	★★★	★	★★★	☆
TE	★★☆	★★	★★☆	★
DLM	★☆	★	★★	★★
C-W	★★☆	★	★★☆	☆
CW-Lite	★★	★★	★★	★★
DIFC	★★	★	★★	★★★
UCON	★★	★	★★☆	★☆
ACSIF	★☆	★★☆	★★	★★★

结束语　本文针对外来信息对系统实体的潜在感染能力，结合信息流对系统完整性的影响，从信息流的视角，研究了访问控制问题，提出了一种利用信息流源追踪和标识系统实体完整性级别的访问控制模型（ACSIF 模型），并分析了模型的安全性、实用性及其在系统完整性度量中的作用。本文工作的意义主要体现在如下两个方面：一是提出了一个融合了信息流特征的、实用性较强的访问控制模型，二是为基于信息流的系统完整性度量提供了一种基本的访问控制支撑途径。我们在 Linux2.6.26 内核版本中进行了原型系统实验。对“读/写”系统调用的测试显示，一般情况下，ACSIF 模型的实现对系统性能的影响在 5%左右，实现模型所带来的系统开销是可以接受的。本文主要讨论了针对“读/写”操作的访问控制问题，下一步的工作将进一步充实和完善模型的访问规则，研究“执行”等操作对实体完整性的影响，并进行全面的原型系统开发和测试实验；同时，在用户集合的基础上，通过加入用户的考核评价等级，提高完整性级别的准确度，以便能够更真实地反映实体的可信性。

参 考 文 献

[1] Singaravelu L, Pu C, Hartig H, et al. Reducing TCB Complexity for Security-sensitive Applications: Three Case Studies[C]// Proceedings of the First ACM SIGOPS/EuroSys European Conference on Computer Systems (Eurosys 2006). Belgium, Apr 2006: 18-21

[2] Guttman J, Herzog A, Millen J, et al. Attestation: Evidence and Trust[R]. MTR080072. Mar. 2008

[3] Biba K J. Integrity considerations for secure computer systems [R]. MTR-3153. April 1977

[4] Hallyn S, Kearns P. Domain and Type Enforcement for Linux [C]// Proceedings of the 4th Annual Linux Showcase and Conference. Oct. 2000

[5] Denning D. A lattice model of secure information flow[J]. Communications of the ACM, 1976, 19(5): 236-243

[6] Clark D D, Wilson D R. A comparison of commercial and military computer security policies[C]// Proceedings of the 1987 IEEE Symposium on Security and Privacy. May 1987

[7] Shankar U, Jaeger T, Sailer R. Toward Automated Information-flow Integrity Verification for Security-critical Applications[C]// Proceedings of the 13th Annual Network and Distributed Systems. 2006

[8] Myers A C, Liskov B. Protecting privacy using the decentralized label model[C]// ACM Transactions on Software Engineering and Methodology (TOSEM). 2000

[9] Myers A C, Liskov B. A Decentralized Model for Infromation Flow Control[C]// Proceedings of the 16th ACM Symposium on Operating Systems Principles. 1997

[10] Jaeger T, Sailer R, Shankar U. PRIMA: Policy Reduced Integrity Measurement Architecture[C]// Proceedings of the Eleventh ACM Symposium on Access Control Models and Technologies. Lake Tahoe, California, USA, June 2006

[11] National Security Agency. Security-Enhanced Linux (SELinux) [EB/OL]. http://www.nsa.gov/selinux

[12] Li Xiao, Shi Wenchang, Liang Zhaohui, et al. Operating System Mechanisms for TPM-based Lifetime Measurement of Process Integrity[C]// Proceedings of the IEEE 6th International Conference on Mobile Adhoc and Sensor Systems (MASS). 2009

[13] 石文昌. 操作系统信任基的设计研究[J]. 武汉大学学报：信息科学版, 2010, 35(5)

[14] Li Ning-hui, Mao Zi-qing, Chen Hong, et al. Trojan Horse Resistant Discretionary Access Control [C] // SACMAT '09. Stresa, Italy, June 2009

[15] Park J, Sandhu R. The $UCON_{ABC}$ usage control model[J]. ACM Transactions on Information and System Security (TISSEC), 2004, 7(1): 128-174

[16] 崔永泉，洪帆，龙涛，等. 基于使用控制和上下文的动态网格访问控制模型研究[J]. 计算机科学, 2008, 35(2): 37-41

[17] Zeldovich N, Wickizer N B, Kohler E, et al. Making Information Flow Explicit in HiStar[C]// The 7th Symp on Operating System Design and Implementation (OSDI). Seattle, WA, 2006

[18] Vandebogart S, Efstathopoulos P, Kohler E, et al. Labels and event processes in the Asbestos operating system [J]. ACM Transactions on Computer Systems (TOCS), 2007, 25(4): 11

[19] Li Ninghui, Mao Ziqing, Chen Hong. Usable Mandatory Integrity Protection for Operating Systems[C]// IEEE Symposium on Security and Privacy. 2007

[20] 石文昌，梁朝晖. 信息系统安全概论[M]. 北京：电子工业出版社, 2009: 334-337