第9章

多级安全

大多数高保证工作都是在由愚蠢的机器人控制的动力装置和地狱机器领域完成的。随着信息处理技术对社会变得越来越重要,这些担忧蔓延到以前认为本质上无害的领域,例如操作系统。

– 博伯特伯爵

政府电话的密码似乎总是掉线,而且
我无法进入它
– 美国外交官和前中央情报局官员 KURT VOLKER,解释了为什么他用他的私人电话发短信

我简报;你
泄漏;他/她因
泄露机密信息而犯下刑事罪行。

– 英国公务员动词

9.1 简介

在接下来的几章中,我将通过案例研究来探讨安全策略的概念。安全策略是对我们要实现的目标的简洁描述;它是由对我们希望避免的不良结果的理解驱动的,反过来又推动了工程。在稍微充实这些想法之后,我将在本章的其余部分探索许多军事和情报系统中使用的多级安全 (MLS) 策略模型,这些系统在不同的分类级别保存信息(机密, Secret, Top Secret, ...),并且必须确保数据只能由其许可级别至少相同的委托人读取。此类政策也越来越多地被称为信息流控制 (IFC)。

它们很重要,原因有很多,即使您从未计划过
为政府承包商工作:

1. 从大约 1980 年到大约 2005 年,美国国防部花费了数十亿美元资助多级安全研究。所以这个模型非常详细,我们开始理解以极大的热情追求单一政策目标的二阶效应;

2. 用于实现它的强制访问控制 (MAC) 系统现已出现在所有主要操作系统中,如 Android、iOS 和 Windows,以保护核心组件免受恶意软件的篡改,如我在第 6 章中所述;

3. 虽然最初开发多级安全概念是为了支持军事系统的机密性,但现在许多商业系统都使用多级完整性策略。例如,安全关键系统使用多个安全完整性级别1。

诗人 Archilochus 有一句名言:狐狸知道很多小事,而刺猬知道一件大事。安全工程通常处于狐狸领地,但多级安全是刺猬方法的一个例子。

9.2 什么是安全策略模型?

在可能采用自上而下的安全工程方法的情况下,它通常采用威胁模型 - 安全策略 - 安全机制的形式。此过程中最关键且经常被忽视的部分是安全策略。

我们所说的安全策略是指一份清晰简洁地表达了保护机制要实现的目标的文档。它由我们对威胁的理解驱动,进而驱动我们的系统设计。它通常采用声明哪些用户可以访问哪些数据的形式。它在指定系统的保护要求和评估是否已满足方面起着相同的作用,系统规范对功能的作用和安全案例对安全性的作用。与规范一样,它的主要功能是沟通。

许多组织使用短语“安全策略”来表示一系列
乏味的语句,如图 9.1 所示:

1 请注意,该术语因不同的安全工程学科而异。
发电的安全完整性等级与 Biba 相似,而汽车安全完整性等级在 ISO 26262 中被设定为危险/风险指标,取决于故障导致事故的可能性,以及预期的严重性和可控性

Megacorp Inc 安全策略

1. 本政策经管理层批准。
2. 所有员工都应遵守本安全政策。
3. 数据应仅提供给“需要知道”的人。
4. 所有违反本政策的行为均应立即报告给安全部门。

图 9.1 – 典型的公司政策语言

这种语言很常见,但毫无用处 至少对安全工程师而言是这样。它回避了核心问题,即“谁决定‘需要知道’以及如何决定?”其次,它混合了不同级别的陈述(政策的组织批准在逻辑上不应成为政策本身的一部分)。第三,有一种机制,但它是隐含的而不是明确的:“sta should obey” 但这意味着他们实际上必须做什么?服从必须由系统强制执行,还是用户“以他们的荣誉”?第四,如何发现违规行为以及谁有具体责任报告违规行为?

仔细想想,这是政治语言。政治家的工作是解决社会中的紧张局势,这往往需要含糊不清的语言,不同的派别可以借此表达自己的意愿;公司高管经常在政治上运作,以平衡公司内部的不同派系2。

因为术语“安全策略”经常被滥用来表示将安全用于
政治,安全工程师开始使用更精确的术语。

安全策略模型是对系统必须具备的保护属性的简明陈述。它的要点通常可以写在一页或更少的纸上。它是系统的保护目标与整个社区或客户的最高管理层达成一致的文件。

它也可能是正式数学分析的基础。

安全目标是对特定实现提供的保护机制的更详细描述,以及它们如何与控制目标列表相关(一些但不是全部通常来自策略模型)。安全目标构成了产品测试和评估的基础。

保护配置文件类似于安全目标,但以独立于实现的方式表示,以实现跨产品和版本的可比较评估。

这可能涉及使用半正式语言,或至少使用合适的安全术语。保护配置文件是根据通用标准 [1396] 评估的产品的要求。 (我在第三部分讨论了通用标准;许多政府使用它们来相互承认国防信息系统的安全评估。)

当我不必如此精确时,我可能会使用短语“安全策略”来指代安全策略模型或安全目标。我永远不会用它来指代陈词滥调的集合。

2当规范变得政治化时,大项目往往在公司失败,他们
当由政府管理时,失败的频率更高 我将在第 3 部分中进一步讨论这些问题。

有时,我们面对一个全新的应用程序,必须从头开始设计安全策略模型。更常见的是,已经存在一个模型;我们只需选择合适的,并将其发展为安全目标即可。这两个步骤都不容易。在本书的这一部分,我提供了许多安全策略模型,在真实系统的上下文中描述它们,并检查安全目标可以用来满足它们的工程机制(和相关约束)。

9.3 多级安全策略

1940 年 3 月 22 日,罗斯福总统签署了第 8381 号行政命令,允许将某些类型的信息分类为受限、机密或机密 [978]。杜鲁门总统后来增加了更高级别的最高机密。这发展成为一种通用的文件敏感性保护标记方案,并在冷战期间也被北约政府采用。分类是标签,从未分类到机密、秘密和绝密(见图 9.2)。最初的想法是,泄露可能导致生命死亡的信息被标记为“机密”,而泄露可能导致许多生命死亡的信息被标记为“绝密”。政府雇员和承包商的许可取决于他们接受审查的谨慎程度;例如,在美国,“秘密”许可涉及检查 FBI 指纹文件,而“绝密”许可还涉及对之前五到十五年工作的背景调查以及面试和测谎仪测试 [548]。候选人必须披露他们近年来的所有性伴侣以及所有可能被用来勒索他们的材料,例如青少年吸毒或同性恋风 airs3。

访问控制策略很简单:只有当您的权限至少与文档的密级一样高时,您才能阅读该文档。因此,获得“最高机密”权限的官员可以阅读“机密”文件,但反之则不行。因此,信息只能向上流动,从机密到秘密再到绝密,但绝不能向下流动 除非授权人有意决定将其解密。

图 9.2 – 多级安全

系统迅速变得更加复杂。文件分类的损坏标准从可能的军事后果扩大到

3 2015 年 6 月,约 2000 万美国人的清关审查数据被中国情报部门从人事管理办公室窃取。到那时,大约有 100 万美国人获得了绝密许可; OPM 数据还包括前雇员和求职者,以及他们的亲属和性伴侣。事后看来,以敏感工作收集所有公民的所有污垢可能不是一个好主意。

经济上的伤害甚至政治上的尴尬。既非机密也非公开的信息在美国被称为“受控非机密信息”(CUI),而英国则使用“Ocial”4。

还有一个代码字系统,可以进一步限制信息,尤其是 Secret 和更高级别的信息。例如,可能揭示情报来源或方法的信息 例如代理人的身份或解密能力 通常被归类为“绝密特种情报”或 TS/SCI,这意味着所谓的需要知道限制被施加同样,将一个或多个代码字附加到一个文件。一些代码字与特定的军事行动或情报来源有关,并且仅供一组指定用户使用。要阅读文档,用户必须拥有附加到它的所有代码字。分类标签加上一组代码字构成一个安全类别或(如果至少有一个代码字)一个隔间,它是一组具有相同访问控制策略的记录。

如今,划分通常使用自主访问控制机制来实现;我将在下一章讨论它。

还有描述符、注意事项和 IDO 标记。描述符是诸如“管理”、“预算”和“约会”之类的词:它们不调用任何特殊处理要求,因此我们可以处理标记为“机密 - 管理”的文件,就好像它只是标记为“机密”一样。警告是警告,例如“UK Eyes Only”或美国的“NOFORN”;他们确实创造了限制。还有国际防卫组织的标记,例如 NATO5。代码字、描述符、注意事项和 IDO 标记之间缺乏明显的差异有助于使系统变得混乱。更详细的解释可以在 [1562] 中找到。

9.3.1 安德森报告

在 1960 年代,当计算机开始广泛使用时,分类系统引起了严重的摩擦。当时在美国空军工作的 Paul Karger 描述了必须从机密系统注销 ,穿过院子到另一个小屋,向武装警卫出示通行证,然后进入并登录秘密系统– 一天十几次。人们很快意识到他们需要一种方法来在同一张桌子上处理不同级别的信息,但如何才能在不泄露秘密的情况下做到这一点呢?一旦修复了一个操作系统错误,就会发现其他一些漏洞。 NSA 聘请了一位著名的计算机科学家 Willis Ware 加入其科学顾问委员会,并于 1967 年将计算机安全问题的严重程度引起了社会和公众的关注 [1985]。人们一直担心即使是不熟练的用户也会

4 在采用 CUI 系统之前,美国对受控但未分类的数据有 50 多种不同的标记,包括仅供官方使用 (FOUO)、执法敏感 (LES)、专有 (PROPIN)、联邦税务信息 (FTI)、敏感但未分类 (SBU) 以及许多其他信息。一些机构在没有任何协调的情况下自行制作标签。当标记为机密的民用文件最终进入国家档案和记录管理局时,进一步的问题出现了,其中 CONFI DENTIAL 是国家安全分类。从这种标记动物园转变为单一的集中管理的政府范围系统已经花费了十多年的时间并且仍在进行中。英国有自己的冷战后简化故事。

5奇怪的是,在英国,“NATO Secret”不如“Secret”那么秘密,所以它是一种反将内容向下而不是向上移动的代码字。

发现漏洞并投机取巧;人们对恶意代码的威胁也有了敏锐且不断增长的认识。 （病毒直到 1980 年代才被发明;70 年代的关注点是特洛伊木马。）当人们发现五角大楼的全球军事指挥和控制系统 (WWMCCS) 容易受到特洛伊木马攻击时,引起了严重的恐慌;这具有将其使用限制在具有“绝密”权限的人的效果,这很不方便。

下一步是詹姆斯·安德森 (James Anderson) 于 1972 年为美国政府进行的一项研究,该研究得出的结论是,安全系统应该做好一两件事;并且这些保护属性应该通过足够简单的机制来验证并且很少改变 [51]。它引入了参考监视器的概念 操作系统的一个组件,它将调解访问控制决策,并且足够小以进行分析和测试,其完整性可以得到保证。用现代的说法,这些组件连同它们相关的操作程序构成了可信计算库 (TCB)。更正式地说,TCB 被定义为一组组件（硬件、软件、人员……）,其正确运行足以确保安全策略得到执行,或者更形象地说,其故障可能导致违反安全政策。 Anderson 报告的目标是使安全策略足够简单,以便 TCB 能够接受仔细的验证。

9.3.2 Bell-LaPadula 模型

得到广泛接受的多级安全策略模型是由 Dave Bell 和 Len LaPadula 于 1973 年提出的[210]。它的基本属性是信息不能向下流动。更正式地说,Bell-LaPadula (BLP) 模型强制执行两个属性：

- 简单安全属性：没有进程可以读取更高级别的数据。
  这也称为未读取 (NRU)；
- *-属性：任何进程都不能将数据写入较低级别。这也称为无减记 (NWD)。

*-属性是 Bell 和 LaPadula 的关键创新。它是由 WWMCCS 崩溃和对特洛伊木马攻击的更普遍的恐惧所驱动的。
未经授权的用户可能会编写一个特洛伊木马程序,并将其放置在被清除为“机密”的系统管理员可以执行它的地方;然后它可以将自己复制到系统的“秘密”部分,读取那里的数据并尝试以某种方式向其发出信号。敌方特工也很有可能在商业软件公司找到一份工作,并在产品中嵌入一些代码,以寻找要复制的秘密文件。如果它可以将它们写到它的创建者可以读取它们的地方,那么安全策略就会被违反。如果应用程序可以记录下来,信息也可能由于错误而泄露。

假定存在恶意代码和错误代码等漏洞。
还假设大多数员工粗心,有些不诚实;长期使用广泛的操作安全措施,特别是在国防领域

环境,以防止人们泄露纸质文件。因此,先前存在的文化假设安全策略的执行独立于用户操作； Bell-LaPadula 着手实施它不仅独立于用户的直接操作,而且独立于他们的间接操作（例如他们运行的程序采取的操作）。

因此,我们必须防止运行在“秘密”级别的程序写入“非机密”级别的文件。更一般地说,我们必须防止高级别的任何进程向低级别的任何对象发送信号。独立于用户操作执行安全策略的系统被描述为具有强制访问控制,这与 Unix 等系统中的自由访问控制相反,在 Unix 中,用户可以对其文件做出自己的访问决定。

Bell-LaPadula 模型使设计人员能够证明定理。给定简单安全属性（不读上)和星属性（不写下）,可以证明各种结果:特别是,如果您的起始状态是安全的,那么您的系统将保持安全。为了简单起见,从现在开始我们通常假设系统只有两个级别,高和低。

9.3.3 Bell-LaPadula 的标准批评

BLP 的引入引起了很大的兴奋:这是一种安全策略,可以按照国防机构的想法行事,直观清晰,但仍然允许人们证明定理。研究人员开始对其进行抨击并对其进行改进。

第一个大争议是关于 John McLean 的 System Z,他将其定义为一个 BLP 系统,增加了用户可以要求系统管理员暂时解密任何文件的功能,从 High 到 Low。这样,Low 用户可以在不破坏 BLP 假设的情况下读取任何 High 文件。 Dave Bell 反驳说 System Z 通过做一些他的模型不允许的事情来作弊（更改标签不是对状态的有效操作）,而 John McLean 的反驳是它没有明确告诉他:所以 BLP 规则是自己还不够。这个问题是通过引入一个宁静属性来解决的。强宁静表示安全标签在系统运行期间永远不会改变,而弱宁静表示标签永远不会以违反定义的安全策略的方式改变。

为什么淡定?在真实系统中,我们通常希望遵守最小特权原则并在未清除级别启动 进程,即使进程的所有者被清除为“最高机密”。如果他们随后访问机密电子邮件,他们的会话将自动升级为“机密”;通常,每次访问更高级别的数据时,进程都会升级（高水位线原则）。由于主题通常是内存管理子系统和文件句柄的抽象,而不是进程,这意味着当访问权限改变时状态改变,而不是数据实际改变时

动作。

实际含义是进程获取它读取的所有文件的安全标签,这些标签成为它写入的每个文件的默认标签集。因此,读取“Secret”和“Crypto”文件的进程此后将创建标记为“Secret Crypto”的文件。这将包括制作的临时副本

其它文件。如果它随后读取“Secret Nuclear”中的文件,那么它之后创建的所有文件都将被标记为“Secret Crypto Nuclear”,并且它将无法写入“Secret Crypto”中的任何临时文件。

这对应用程序的影响是多级安全的严重复杂性之一;大多数应用软件需要重写（或至少修改)才能在MLS 平台上运行。安全级别的实时变化意味着可以随时撤销对资源的访问,包括在交易过程中。由于吊销问题在现代操作系统中通常无法解决,至少在任何完整的形式中都无法解决,因此应用程序必须以某种方式应对。除非你付出一些努力和努力,否则你很容易发现所有东西都在最高的隔间里 或者系统分裂成数千个彼此根本不通信的小隔间。为了防止这种情况,标签现在通常在 MLS 机器之外使用,并使用自主访问控制机制来处理（我将在下一章中讨论）。

BLP 以及所有强制访问控制系统的另一个问题是分离用户和进程是容易的部分;困难的部分是何时需要一些受控的交互。大多数真实的应用程序需要某种可以破坏安全策略的可信主体;典型的例子是一个可信的文字处理器,它可以帮助情报分析员在将一份绝密文件编辑成机密时擦除它 [1270]。 BLP 未提及系统应如何保护此类应用程序。所以它成为可信计算库的一部分,但是不能使用仅基于 BLP 的模型来验证的一部分。

最后值得注意的是,即使使用高水位标记细化,BLP 仍然不处理主体或对象的创建或销毁（这是构建真正的 MLS 系统的难题之一）。

9.3.4 MLS 政策的演变

多级安全策略在实践和研究领域并行发展。

第一个多级安全策略是 1967–8 年为 ADEPT-50 编写的高水位标记版本 10,ADEPT-50 是为 IBM S/360 大型机 [2006] 开发的强制访问控制系统。这使用了级别、隔间和组的三元组,组是文件、用户、终端和作业。由于程序（而不是进程)是主题,因此它很容易受到特洛伊木马程序的攻击。尽管如此,它为 BLP 奠定了基础,也导致了当前的 IBM S/390 大型机硬件安全架构 [940]。

下一个重要步骤是 Multics。这始于 1965 年的麻省理工学院项目,后来发展成为霍尼韦尔的产品;它成为“可信系统”的模板和鼓舞人心的例子。 Paul Karger 和 Roger Schell 对其进行的评估具有巨大的影响力,并且是恶意软件可以隐藏在编译器中的想法的首次出现 [1019] – 并导致了 Ken Thompson 的著名论文“Reflections on Trusting Trust”十年后 [1883 年]。 Multics 有一个名为 SCOMP 的派生系统,我将在 9.4.1 节中讨论它。

从 1980 年代开始,大量研究资金涌入多级安全性,催生了许多替代方案。不干涉是 Joseph Goguen 和 Jose Meseguer 在 1982 年提出的[773]。在具有此属性的系统中,High 的操作不会影响 Low 可以看到的内容。不可演绎性的限制较少,由 David Sutherland 于 1986 [1847] 引入,用于对 LAN 等应用程序进行建模,LAN 上同时存在 Low 和 High 机器,而 High 机器加密其 LAN trac6。不可推论性被证明太弱了,因为没有什么可以阻止 Low 以 99% 的确定性对 High 输入进行推论。其他理论模型包括广义非干涉和限制性[1276];Harrison-Ruzzo-Ullman 模型解决了如何处理文件的创建和删除的问题,BLP 对此保持沉默 [868];分区模式工作站 (CMW) 策略尝试使用浮动标签对信息分类进行建模,如高水位标记策略 [2040, 807]。

在这波创新浪潮中,对现代系统影响最大的模型可能是类型执行（TE)模型,由 Earl Boebert 和 Dick Kain [271] 提出,后来由 Lee Badger 等人扩展为领域和类型执行（ DTE) [153]。这将主题分配给域,将对象分配给类型,矩阵定义了允许的域-域和域-类型交互。这在 SELinux 中使用,现在是 Android 的一个组件,它通过将主体和对象都放在类型中并具有允许的类型对矩阵 [1187] 来简化它。实际上,这是第二个访问控制矩阵;除了拥有用户 ID 和组 ID 之外,每个进程还有一个安全 ID (SID)。

Linux 安全模块框架提供了可插入的安全性,您可以在其中设置对 SID 进行操作的规则。

DTE 引入了一种配置语言 (DTEL),以及基于路径名的隐式文件类型;因此,给定子目录中的所有对象都可以声明为在给定域中。 DTE 比 BLP 更通用,因为它开始处理完整性和机密性问题。早期的用途之一是强制执行受信任的管道:其想法是将一组进程限制在管道中,以便每个进程只能与前一阶段和下一阶段对话。这可用于组装无法绕过的防护装置和防火墙,除非至少有两个阶段受到损害 [1430]。类型强制机制可以识别代码与数据,并且权限可以绑定到代码;因此,宁静问题可以在执行时处理,而不是在读取数据时处理。这可以使事情变得更容易处理。例如,它们用于 Sidewinder 防火墙。

TE/DTE 更大的灵活性和表现力的缺点是,由于状态爆炸,实施像 BLP 这样的策略并不总是直截了当;在编写安全策略时,您必须考虑不同类型之间所有可能的交互。可以使用其他机制来管理策略复杂性,例如运行原型一段时间以观察什么才算是正常行为;然后您可以打开 DTE 并阻止所有迄今为止未看到的信息流。但这并不能保证

6要做到这一点还需要很多其他的东西,例如用空值填充 High trac,这样 Low 用户就不能进行 trac 分析请参阅 [1632] 了解此类系统的早期示例。您可能还需要考虑高网络上的低流量,例如士兵打电话回家的设施。

您制定的政策是正确的。

1992 年,David Ferraiolo 和 Richard Kuhn 引入了基于角色的访问控制 (RBAC) 来管理策略复杂性。它使主要附加到角色而不是单个用户或机器的规则正式化 [678、679]。可以由给定角色的持有者执行的交易操作被指定,然后是授予角色成员资格的机制(包括授权)。多年来,角色或组一直是银行等组织在实践中用来管理访问控制的机制; RBAC 模型开始将其形式化。

它可用于提供更细粒度的控制,例如,通过授予“作为教授的罗斯”、“作为招生委员会成员的罗斯”和“罗斯正在阅读私人电子邮件”的不同访问权限。它的一个变体,基于方面的访问控制 (ABAC),添加了上下文,因此您可以区分“罗斯在他实验室的工作站”和“罗斯在地球上某个地方的电话上”。自 Windows 8 以来,Windows 都支持这两者。

SELinux 将其构建在 TE 之上,以便用户在登录时映射到角色,角色被授权用于域,域被授予类型权限。在这样的平台上,RBAC 可以有效地处理完整性问题和机密性问题,方法是在调用某些程序时允许修改角色成员资格。因此,例如,调用从网上下载的不受信任软件的进程可能会失去写入敏感系统文件所需的角色成员身份。我在 9.5.2 中更详细地讨论了 SELinux。

9.3.5 Biba 模型

将多级完整性模型纳入 Windows 7 后,人们重新对 Ken Biba [237] 于 1975 年设计的安全模型产生了兴趣,该模型仅处理完整性而忽略机密性。 Biba 的观察是机密性和完整性在某种意义上是双重概念 机密性是对谁可以阅读消息的约束,而完整性是对谁可以编写或更改消息的约束。

因此,您可以将 BLP 颠倒过来,将其回收到完整性策略中。

作为一个具体的应用,一个电子医疗设备,比如心电图,可能有两种独立的模式:校准和使用。校准数据必须防止损坏,因此普通用户应该能够读取但不能写入;当普通用户重置设备时,它将丢失其当前用户状态(即内存中的任何患者数据),但校准必须保持不变。

只有授权的技术人员才能重新进行校准。

为了对这样的系统建模,我们可以使用多级完整性策略,其规则是我们可以在更高级别读取数据(即,用户进程可以读取校准数据)并写入较低级别(即,校准进程可以写入用户进程中的缓冲区);但我们绝不能往下读或往上写,因为这两者都可能导致高完整性对象被低完整性(即可能不可靠的)数据污染。 Biba 模型通常根据低水位线原则制定,它是上面讨论的高水位线原则的对偶:对象的完整性是为其创建做出贡献的所有对象的最低级别。

这是第一个正式的诚信模型。数量惊人的真实系统按照 Biba 路线工作。例如,乘客信息

铁路系统可能会从信号系统获取信息,但不能影响它;电力公司的电力调度系统将能够看到安全系统的状态,但不会对其进行干扰。

安全关键系统社区在安全完整性级别方面进行讨论,这与安全机制失败的可能性以及它旨在提供的风险降低级别有关。

Windows,自版本 6 (Vista) 起,使用完整性级别标记文件对象,可以是低、中、高或系统,并实施默认策略 NoWriteUp。默认情况下,关键文件位于系统,其他对象位于中 - 除了处于低的浏览器。所以使用IE下载的东西可以读取Windows系统中的大部分文件,但不能写入它们。目标是限制恶意软件可能造成的损害。

如您所料,Biba 与 Bell LaPadula 存在相同的基本问题。没有无数例外,它不能很好地适应现实世界的操作。例如,一个真实的系统通常需要可以覆盖安全模型的可信主体,但 Biba 本身不能保护和限制它们,BLP 不能。例如,汽车的安全气囊在公共汽车上的重要性不如发动机,但当它展开时,您假设存在燃油起火的风险并关闭发动机 。还有其他 Biba 也无法表达的真实完整性目标,例如有保证的管道。对于 Windows,Microsoft 甚至放弃了 NoReadDown 限制,并且最终没有使用其完整性模型来保护基本系统免受用户侵害,因为这需要更频繁的用户确认。事实上,Type Enforcement 模型是由 Boebert 和 Kain 引入的,作为 Biba 的替代方案。不幸的是 Windows 没有包含 TE。

9.4 MLS 系统的历史例子

本书的第二版对 MLS 系统的历史进行了更全面的介绍;由于这些在很大程度上已经过时,并且 MLS 研究计划已经结束,因此我在这里提供一个较短的版本。

9.4.1 SCOMP

一个关键产品是安全通信处理器 (SCOMP),它是 1983 年推出的 Multics 的衍生产品 [710]。这是美国国防部认为它想要处理多级分类消息传递的不遗余力的实施。它已经正式验证了硬件和软件,并使用最小的内核来使事情变得简单。它的操作系统 STOP 使用 Multics 的环系统来维护多达 32 个独立的隔间,并允许它们之间适当的单向信息流。

SCOMP 用于军事邮件警卫等应用。这些防火墙允许邮件从低层传递到高层,但反之则不行 [538]。(通常,支持单向流的设备称为数据二极管。)SCOMP 的继任者 XTS-300 支持 C2G,即命令和控制卫士。这用于时间分段部队部署数据 (TPFDD) 系统,其

职能是计划美国部队调动和相关后勤。 SCOMP 最重要的贡献是作为橙皮书 [544]（美国可信计算机系统评估标准)的模型。这是安全计算机系统的第一套系统标准,于 1985 年推出,最终于 2000 年 12 月退役。橙皮书不仅在美国而且在盟国中都具有巨大的影响力;英国、德国和加拿大等国家在此基础上制定了自己的国家标准,直到这些国家标准最终被纳入通用标准 [1396]。

橙皮书允许在多个级别对系统进行评估,其中 A1 为最高级别,然后向下移动通过 B3、B2、B1 和 C2 到 C1。 SCOMP 是第一个被评为 A1 的系统。它在公开文献中也有广泛记载。作为第一个,并且相当公开,它为下一代军事系统设定了目标。

Unix 的 MLS 版本在 1980 年代后期开始出现,例如 AT&T 的 System V/MLS [47]。这增加了安全级别和标签,表明可以将 MLS 属性引入商业操作系统,而对系统内核的更改最少。到本书第二版（2007 年)时,Sun 的 Solaris 已成为高可靠性服务器系统和许多客户端的首选平台。分区模式工作站 (CMW) 是后者的一个例子,它允许同时查看和修改不同级别的数据,因此情报分析师可以在一个窗口中阅读“最高机密”数据,并在“机密”中编写报告 在另一个中,不会意外地向下复制和粘贴文本 [932]。有关工程,请参阅 [635、636]。

9.4.2 数据二极管

人们很快意识到,随着除了邮件之外还开发了更复杂的网络服务,简单的邮件守卫和密码箱限制太多。第一代 MLS 机制对于实时服务来说是低效的。

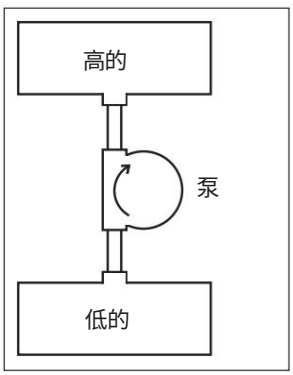

图 9.3：- NRL 泵

因此,美国海军研究实验室 (NRL) 开发了泵 一种单向数据传输设备（数据二极管）,以允许安全的单向信息流（图 9.3。主要问题是,从低到高发送数据时很容易,需要确保传输可靠性意味着确认

边缘消息必须从高位发送回低位。 Pump 使用多种机制限制可能的反向泄漏带宽,例如缓冲和随机确认时间 [1012、1013、1014]。这种方法的吸引力在于,可以通过使用数据二极管连接不同安全级别的独立系统来构建 MLS 系统。由于这些系统不处理多于一个级别的数据 一种称为系统高层的架构 它们可以由廉价的商用现成 (COTS) 组件构建。

您无需担心在内部应用 MLS,只需保护它们免受外部攻击,无论是物理攻击还是基于网络的攻击。随着硬件成本的下降,这已成为首选,世界上的军事基地现在到处都是 KVM 开关(让人们在低系统和高系统之间切换键盘、视频显示器和鼠标)和数据二极管(以链接低和高网络)。 [1015] 中讲述了泵的故事。

早期的应用是物流。一些信号情报设备是“最高机密”,而喷气燃料和鞋带之类的东西不是;但即使是这样简单的商品也可能成为“秘密”,因为它们的数量或移动可能会泄露有关战术意图的信息。管理所有这些所需的系统可能很难构建; MLS 在美国和英国的物流项目最终以代价高昂的灾难告终。在英国,皇家空军的后勤信息技术系统 (LITS) 是一个耗时 10 年(1989-99 年)、耗资 5 亿英镑的项目,旨在为皇家空军的 80 个基地 [1386] 提供单一的商店管理系统。

它被设计为在两个级别上运行:喷气燃料和靴子抛光剂的“限制”,以及核弹等特殊商店的“秘密”。它最初被实现为两个独立的数据库系统,通过泵连接以强制执行 MLS 属性。该项目成为一个典型的故事,即需求的缓慢变化导致成本不断上升。其中一项变化是随着冷战的结束,分类规则有所放宽。结果,人们发现几乎所有“秘密”信息现在都是静态的(例如,空投核弹的操作手册现在保存在战略储备中,而不是在空军基地)。为了省钱,“秘密”信息现在保存在 CD 上并锁在保险箱中。

MLS 的另一个主要应用是窃听。调查的目标不应该知道他们正在被窃听,所以第三方必须保持沉默当电话公司开始实施窃听作为无声电话会议时,电话会议的费用必须由窃听者承担,而不是由目标承担.现代要求是多层次的:不同层次的多个机构可能想要监视一个目标,并相互监视,警察监视毒贩,反腐败部门监视警察,等等。消除隐蔽通道比看起来更难;有关 2000 年代中期的调查,请参阅 [1707];纯粹的 MLS 安全策略是不够的,因为嫌疑人可能会尝试破解或混淆窃听设备,因此需要抵制在线篡改。在一个臭名昭著的案例中,在雅典奥运会期间,在希腊总理及其高级同事的手机上发现了窃听器;移动电话公司开关设备中的合法拦截设施被未经授权的软件滥用,并在窃听者的修改导致某些文本消息无法发送时被发现 [1550]。

这家电话公司被罚款 7600 万欧元(近 1 亿美元)。现在使用现代 VOIP 系统管理窃听的最简单方法可能只是将所有内容写入磁盘并在以后提取您需要的内容。

也有许多军用嵌入式系统。在潜艇中,速度、反应堆输出和 RPM 都是最高机密,因为这三项测量的历史记录将揭示潜艇的性能 这是为数不多的甚至美国和英国也不共享的信息之一。当船舶在港口时,仪器需要不是绝密的,因此工程变得更加复杂,因为这会使维护复杂化。至于空战,一些美国雷达不会显示性能机密的美国飞机的速度,除非操作员有适当的许可。当你读到有关 F-16 飞行员看到一个速度异常快的 UFO 的故事时,他们的雷达上的速度没有任何意义,你可以把两个和两个放在一起。当强大的参与者试图将 MAC 策略融入物联网基础设施时,看看会产生什么样的其他副作用,以及它们会产生什么样的迷信,这将是一件很有趣的事情。

9.5 MAC:从 MLS 到 IFC 和完整性

在本书的第一版中,我注意到使用强制访问控制来防止篡改和提供实时性能保证的趋势 [1313, 1018],并大胆地说“也许多级系统的真正未来不在于机密性,但诚信。”政府机构已经了解到 MAC 是阻止恶意软件所需要的。到第二版时,多级完整性已经进入了 Windows 的大众市场,Windows 基本上使用了 Biba 模型。

9.5.1 视窗

在 Windows 中,所有进程和所有安全对象(包括目录、文件和注册表项)都可能具有完整性级别标签。文件对象默认标记为“中”,而 Internet Explorer(以及使用它下载的所有内容)标记为“低”。因此,需要用户操作来升级下载的内容,然后才能修改现有文件。也可以使用 Windows 实施粗略的 BLP 策略,因为您还可以设置“NoReadUp”和“NoEx ecuteUp”策略。这些不是默认安装的;微软担心恶意软件会在系统中自行安装然后隐藏起来。将浏览器保持为“低”会使安装更加困难,并且允许所有进程(甚至是低进程)检查系统的其余部分会使隐藏更加困难。但是这种针对 MAC 的仅完整性方法确实意味着以低速运行的恶意软件可以窃取您的所有数据;所以有些用户可能会关心为敏感目录设置“NoReadUp”。

Joanna Rutkowska 在 [1634] 中讨论了所有这些;她还描述了一些有趣的基于虚拟化的潜在攻击。

9.5.2 SELinux

SELinux 的情况与 Windows 有点相似,因为强制访问控制机制的直接目标也是限制妥协的影响。 SELinux [1187] 由 NSA 基于 Flask 安全架构 [1811] 实现,它将策略与执行机制分开;安全上下文包含所有关联的安全属性

Flask 中的主体或客体,其中一个属性包括 Type Enforcement 类型属性。安全标识符是安全上下文的句柄,由安全服务器映射。这是制定政策决策的地方,并驻留在内核中以提高性能 [819]。从Linux 2.6开始就成为主流。服务器为内核的其余部分提供安全 API,安全模型隐藏在其后。服务器内部实现了通用的约束引擎,可以表达RBAC、TE、MLS。在 2000 年代中期的典型 Linux 发行版中,它被用来分隔各种服务,因此接管您的 Web 服务器的攻击者不会因此也获得您的 DNS 服务器。正如第 6 章所述,Android 对它的采用使其成为世界上最流行的操作系统的一部分。

9.5.3 嵌入式系统

有许多实地系统实现了 Biba 模型的某些变体。
除了我已经提到的医疗设备和铁路信号应用之外,还有公用事业。例如,在电力公司中,通常有一个安全系统层次结构,这些系统在最高安全完整性级别上完全独立运行;这些对电力调度等操作系统可见,但不受其影响;零售级计量系统可以被计费系统观察到,但不受计费系统的影响。电力调度系统中的零售电表和变电站级电表都将信息馈送到欺诈检测中,最后是执行信息系统,它可以观察一切,而不会对操作产生直接影响。

在汽车中,大多数品牌的动力总成和机舱都有单独的 CAN 总线,因为您不希望收音机上的恶意应用程序能够操作您的刹车(尽管在 2010 年,安全研究人员发现这种分离是完全不够的[1085])。

还值得记住的是,简单的完整性控制只是阻止恶意软件接管机器 它们不会阻止它感染低隔间并将其用作跳板以传播到其他地方,或向其他机器发出指令。

总而言之,从早期多级系统中吸取的许多经验教训都适用于许多更广泛的应用程序。许多故障模式也是如此,我现在将对其进行讨论。

9.6 出了什么问题

工程师从失败的系统中学到的东西比从成功的系统中学到的更多,而 MLS 系统在这方面是一位有效的老师。花费数十亿美元构建系统以遵循具有高水平保证的简单策略已经阐明了信息流控制的许多二阶和三阶后果。我将从更具理论性的部分开始,一直到业务和工程结束。

9.6.1 可组合性

考虑一个接受两个“高”输入H1和H2 的简单设备;复用它们;通过使用一次一密（即随机生成器)对它们进行异或来加密它们;在H3上输出 pad 的另一个副本;并输出密文,该密文使用提供完美保密性的密码系统加密,被认为是低密文（输出 L）,如图 9.4 所示。

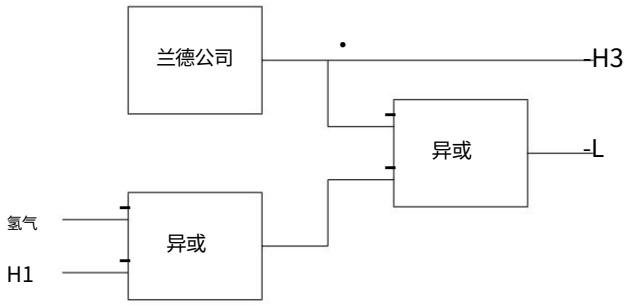

图 9.4 – 具有反馈的安全系统的不安全组合

孤立地,该设备被证明是安全的。但是,如果允许反馈,则H3的输出可以反馈到H2,结果是高输入H1现在出现在低输出 L 上。时序不一致也会破坏两个安全系统的组合（由Daryl McCullough [1260]）。

一般来说,组合问题 如何将两个或多个安全组件组合成一个安全系统 是困难的,即使在关于理想组件[1430]的证明结果的相对整洁的水平上也是如此。 （简单的信息流不会组合;非干扰或不可演绎性也不会。)大多数低级问题都是在引入某种反馈时出现的;没有它,可以在许多正式模型下实现组合 [1277]。然而,在现实生活中,反馈无处不在,界面问题、功能交互等问题会使安全属性的组合变得更加困难。例如,一个系统生成数据的速度可能足以对另一个系统执行拒绝服务攻击。安全组件的组合常常因更高级别的不兼容性而受挫。组件可能是根据两种不同的安全策略设计的,或者是根据不一致的要求设计的。

9.6.2 级联问题

级联问题给出了组合问题的一个例子（图 9.5）。在橙皮书引入了一系列评估级别之后,这导致了关于系统可以运行的级别数量的跨度限制规则 [548]。例如,评估为 B3 的系统通常允许

以非机密、机密和机密或机密、机密和绝密级别处理信息;没有系统允许同时处理未分类和绝密数据 [548]。

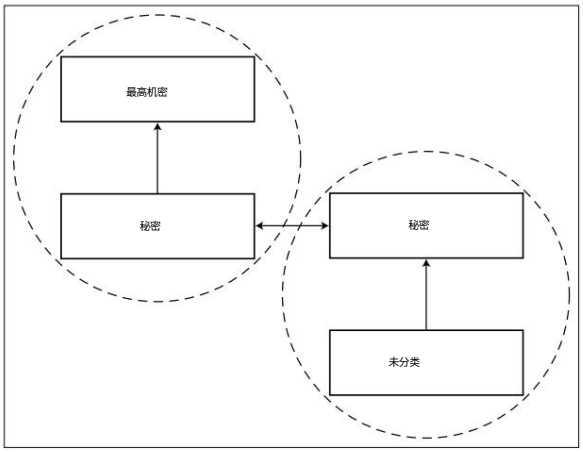

图 9.5： 级联问题

如图所示,很容易将两个 B3 系统连接在一起,从而破坏此策略。第一个系统将未分类和秘密连接在一起,其秘密级别与第二个系统通信 第二个系统也处理绝密信息 [923]。这违反了跨度限制。

9.6.3 隐蔽通道

对多级系统施加跨度限制的原因之一来自一个著名且被广泛研究的问题:隐蔽通道。 Lampson 于 1973 年首次指出 [1125],隐蔽通道是一种机制,它不是为通信而设计的,但仍然可以被滥用以允许信息从高到低进行通信。

当高级进程可以通过影响某些共享资源向低级进程发出信号时,就会出现典型的隐蔽通道。在现代多核 CPU 中,它可以增加它在时间t时使用的 CPU 内核的时钟频率,以表示绝密文件中的第 i 位是 1,并让它缩减以表示该位是 a 0。

这给出了每秒数十位的隐蔽信道容量 [35]。自 2018 年以来,CPU 设计人员一直在为一系列利用 CPU 微架构的封面频道而苦苦挣扎;它们的名称如 Meltdown、Spectre 和 Foreshadow,不仅为 High 提供了向 Low 发出信号的方法,还为 Low 提供了绕过访问控制和读取 High 内存的方法。我将在边信道一章中详细讨论这些。

开发人员能够在常规操作系统中始终如一地进行机密性保护的最佳做法是将其限制在每秒 1 位左右。 （这是 DoD 的目标 [545],进行系统分析的技术可以在 Kemmerer [1036] 中找到。)在我们希望防止大型 TS/SCI 文件的环境中,每秒一位可能是可以容忍的 例如卫星

照片 从 TS/SCI 用户泄露给“秘密”用户。但是,它可能对高价值加密密钥构成致命威胁。这是军事和银行学说在专用硬件中进行加密的原因之一。

据我所知,带宽最高的隐蔽信道出现在大型预警雷达系统中,其中 High(雷达处理器)控制着数百个天线元件,这些天线元件用经过调制的高速脉冲序列照射 Low(目标)带有伪随机噪声,使干扰更加困难。在这种情况下,雷达代码必须是可信的,因为隐蔽信道带宽是每秒许多兆位。

9.6.4 来自恶意软件的威胁

1983 年弗雷德·科恩 (Fred Cohen) 撰写了第一篇关于计算机病毒的论文,并使用病毒轻松渗透多层安全系统,令国防计算机界震惊。以前被认为是多级安全的系统 [450]。自 1960 年代以来,人们一直在考虑恶意软件,并采取各种措施来缓解它,但他们的重点一直放在特洛伊木马上。

可以通过多种方式使用恶意代码来破坏访问控制。如果参考监视器(或其他 TCB 组件)可能被破坏,则恶意软件可以将整个系统交付给攻击者,例如通过发布未经授权的许可。出于这个原因,稍微宽松的规则适用于所谓的封闭安全环境,这些环境被定义为“系统应用程序得到充分保护,免受恶意逻辑插入”[548],这反过来又激励供应商篡改-使用 TPM 等技术对 TCB 进行验证。但即使 TCB 保持完好无损,恶意软件仍可将自身从低位复制到高位(BLP 无法阻止)并使用隐蔽通道向下发送信息。

9.6.5 多实例化

另一个困扰研究界的问题是多实例化。
假设我们的 High 用户创建了一个名为 agents 的文件,而我们的 Low 用户现在尝试做同样的事情。如果 MLS操作系统禁止他,就会泄露信息 即在High有一个名为agents的文件。但如果允许的话,它现在将有两个同名的文件。

通常我们可以通过命名约定来解决问题,例如为低级和高级用户提供不同的目录 。但是这个问题对于数据库来说仍然是一个难题 [1649]。假设一个 High 用户将分类货物分配给一艘船。

系统不会将此信息透露给 Low 用户,他们可能会认为这艘船是空的,并试图为其分配其他货物,甚至改变其目的地。

美国和英国的做法在这方面存在分歧。美国青睐的解决方案是 High 用户在分配真正的 High 货物的同时分配一个 Low 封面故事。因此,底层数据将类似于图 9.6。

等级	货物	目的地
秘密	导弹	伊朗
受限制的 -		-
未分类发动机备件	塞浦路斯	

图 9.6 – 美国如何处理机密数据

在英国,理论更简单 系统会自动回复“已分类”给试图查看或更改 High 记录的 Low 用户。两个可用视图如图 9.7 所示。

等级	货物	目的地
秘密	导弹	伊朗
受限制的	分类 分类	
未分类 –		-

图 9.7 – 英国如何处理机密数据

这使得系统工程更简单。它还可以防止掩盖故事仍然可能出现的错误和隐蔽渠道（例如,Low 用户试图为塞浦路斯添加一个弹药容器）。缺点是每个人都倾向于需要最高的可用权限才能完成工作。

（在实践中,为了不宣传秘密任务的存在,掩盖故事仍然被使用。）

9.6.6 MLS 的实际问题

多级安全系统非常昂贵,而且难以构建和部署。成本和混乱的来源有很多。

1. 它们是小批量制造的,通常采用高标准的物理稳健性,使用由军事采购官僚机构驱动的详细文档、测试和其他质量控制措施。

2. MLS 系统具有特殊的管理工具和程序。训练有素的 Unix 管理员不能未经大量进一步培训就进行 MLS 安装;如此多的 MLS 系统是在没有使用其功能的情况下安装的。

3. 很多应用程序需要重写或者至少要大大修改才能运行
   在 MLS 操作系统下 [1629]。

4. 因为流程在看到新标签时会自动升级,所以它们使用的文件也必须如此。新文件默认为属于任何可能输入的最高标签。所有这一切的结果是事物被过度分类的长期趋势。当系统组件累积他们看到的所有标签时,会出现一个特殊的问题,导致标签爆炸

他们在那里获得了这样一个集合,以至于任何一个委托人都无法再访问它们。因此,它们被置于受信任的计算基础中,最终包含操作系统的相当大的一部分(加上实用程序、窗口系统软件、中间件,如数据库软件)。这种“TCB 膨胀”不断推高评估成本并降低保证。

5. 数据分类可能会变得复杂:

- 在冲突前夕,食品等“无害”商店的位置可能会暴露战术意图,因此可能会突然升级;
- 分类并不总是单调的。分类为“机密”的设备可能很容易包含分类为“机密”的组件,另一方面,很难授予“机密”访问权限以访问“绝密”数据库中的机密信息;
- 信息可能需要降级。情报分析员可能需要拍摄一张在 TS/SCI 分类的卫星照片,并将其粘贴到“机密”的战地指挥官评估中。如果信息被病毒秘密隐藏在图像中,这可能涉及特殊过滤器、图像有损压缩等。一种选择是“打印和传真”机制,将文档转换为位图,并将其记录下来以供追溯。
- 我们可能需要担心攻击者可用的信息量。例如,我们可能很乐意解密任何一张卫星照片,但解密整个集合将揭示我们的监视能力和情报优先事项的历史。 (我将在 11.2 节中更详细地讨论这个聚合问题。)
- 类似地,对非机密数据执行的非机密程序的输出也可能是机密的,例如,如果应用于在线论坛的标准数据挖掘技术列出了恐怖嫌疑人名单。

6. 尽管 MLS 系统可以防止不需要的事情(例如信息泄露),但它们也可以防止需要的事情(例如构建一个搜索引擎以跨机构的所有绝密分区数据运行)。

因此,即使在军事环境中,其好处也值得怀疑。 9/11 之后,许多规则都放宽了,最高机密之上的访问控制通常是酌情决定的,以允许信息共享。当然,这样做的代价是斯诺登泄密事件。

7. 最后,执着的政府保密是一种长期负担。已故参议员丹尼尔·莫伊尼汉 (Daniel Moynihan) 对其真正目的及其在美国外交和军事事务中的巨大成本进行了批判性研究 [1346]。例如,杜鲁门总统从未被告知 Venona 解密,因为该材料被认为是“军队财产”。正如他所说:“部门和机构囤积信息,政府变成了一种市场。

秘密成为组织资产,永远不会共享,除非以换取另一个组织的资产。”

最近 MLS 条令削弱作战效能的例子包括在阿富汗战争中使用未加密的无人机通信（因为武装部队担心如果他们让 NSA 官僚机构参与进来，无人机将无法使用），以及使用众所周知，冠状病毒危机期间英国政府内阁会议使用的 Zoom 视频会议系统不安全（政府的加密视频会议终端属于机密，因此部长们不得将其带回家）。这让我想起一位愤怒的英国将军的一句俏皮话：“侏罗纪公园和国防部有什么区别？一个是充满恐龙的主题公园，另一个是电影！”

不乏内部战略批评。美国分类系统 Mitre 的 JASON 程序 2004 年的一份报告得出结论，它不再适合目的 [978]。有很多有趣的原因，包括生产者和消费者社区的风险/收益计算大相径庭；分类开始由分销渠道而不是实际风险主导。相对容易受到攻击导致政府系统过于保守和规避风险。它注意到许多不正当的结果；例如，伊拉克的“捕食者”图像是未分类的，并且在一段时间内以明文形式传输，因为陆军担心加密会使 NSA 官僚机构参与密钥管理并阻碍作战。

Mitre 建议为特定目的设置灵活的隔间，特别是在将易腐烂的信息传送到战术隔间时；智能地使用权利管理和虚拟化等技术；并且用专注于交易风险的系统取代对清算个人的终生信任。

不管怎样，自本书第二版以来最大的变化之一是国防部关于 MLS 的庞大研究计划已经消失，MLS 设备不再在政府系统市场上非常积极地推广，并且系统在十年内一直保持相当稳定.大多数政府系统现在都在运行系统高级 也就是说，完全在 Ocial、Secret 或 Top Secret。上一节中讨论的困难，加上硬件成本的下降和虚拟化的到来，削弱了在同一台机器上拥有不同级别的动机。因此，部署的 MLS 系统往往是不同级别之间的防火墙或邮件防护，并且通常用一个新的首字母缩略词 MILS（多个独立的安全级别）来指代。真正的分离是在网络层面，在非机密网络之间，秘密互联网协议路由器网络（SIPRNet）使用加密背后的基本标准设备处理秘密数据，以及处理绝密材料和联合全球情报通信系统（JWICS）他们的系统保存在安全分区信息设施 (SCIF) 中 屏蔽房间以防止电子窃听，我将在后面的侧信道章节中讨论。

偶尔会有可怕的变通办法，例如“向下浏览”系统，它可以让处于高位的人查看处于低位的网站；他们可以单击按钮和链接进行导航，但不能输入任何文本。这种丑陋的黑客行为显然有可能被滥用；充其量它们可以帮助诚实的人避免粗心的错误。

9.7 总结

强制访问控制最初是为军事应用而开发的,它仍然用于专门的防火墙（守卫和数据二极管）。然而,如今 MAC 机制的主要用途是在 Android、iOS 和 Windows 等平台上,它们可以保护操作系统本身免受恶意软件的侵害。

自 20 世纪 70 年代中期以来,MAC 机制一直是计算机安全研究的主要课题,并且在尝试将它们用于军事多级安全时吸取的教训构成了许多用于安全评估的方案的基础。从业者了解他们的长处和局限性很重要,这样你就可以在适当的时候借鉴研究文献,避免在不适当的时候被拖入过度设计。

有很多问题需要我们做“狐狸”而不是“刺猬”来解决。通过尝试将所有安全问题都视为刺猬问题,MLS 通常会导致不适当的安全目标、策略和机制。

研究问题

在 NSA 推出 SELinux 后,Earl Boebert 于 2001 年勾画出一个长期挑战,即使强制访问控制机制适应安全关键系统（参见本章开头的引述和 [270]）。作为构建高度可靠的专用设备的工具,可以限制错误和故障的后果,类型强制和基于角色的访问控制等机制在安全领域之外应该很有用。我们会看到它们广泛应用于物联网吗?我们已经在汽车和配电等应用中提到过 Biba 类型的机制； SELinux、Windows 和 Android 等产品中的 MAC 机制是否能让设计人员锁定信息流并减少意外交互的可能性?

美国国家安全局继续资助 MLS 的研究,现在在国际金融公司的标签下,尽管水平低于过去。在现代智能手机中正确地做到这一点很困难;有关此类工作的示例,请参阅 Adwait Nadkarni 及其同事的 Weir 系统 [1372]。除了现代操作系统具有更大的内在复杂性之外,手机还有大量的侧通道,而且它们的应用程序通常只在与云服务通信时有用,而真正繁重的工作必须在这里完成。分别提供“低端”和“高端”手机的商业产品包括三星的 Knox 等产品。

一组单独的研究问题围绕着实际的军事行动,现实与政策相去甚远。参与近期冲突的所有武装部队,包括美国和英国在伊拉克和阿富汗的部队,都存在个人手机的安全问题,在某些情况下,叛乱分子追踪他们的家人回家并威胁骚扰他们。皇家海军曾在 2009 年试图禁止使用手机,但离开的水手太多了。通过 Instagram 跟踪船只很容易;一艘军舰由数百名年龄在 18-24 岁之间的年轻男女组成,除了在社交媒体上发布快照外,别无他法。纪律往往集中在直接的操作威胁上,例如当看到一名水手在快速处理地雷时:问题是在地雷附近使用无线电的风险!不同的海军有

尝试了不同的东西:挪威人拥有自己的水手专用网络,而美国正在尝试具有 MLS 功能的手机。但北约的演习表明,一支海军要破解另一支海军的导航系统非常容易。甚至以色列人也遇到过他们的士兵在约旦河西岸和戈兰高地使用手机的问题。

进一步阅读

英国政府信息分类系统的非机密手册,以及不同级别所需的物理、逻辑和其他保护机制,自 2013 年以来一直公开,最新文件(在撰写本文时)已于 2018 年 11 月在政府安全网页 [802] 上发布。关于 Walker 间谍网的报告详细描述了一次惊人的失败,并揭示了运行一个系统的绝对复杂性,在该系统中,任何时候可能有 300 万人获得许可,每年处理 100 万份申请 [876].查普曼 (Chapman) [407] 是关于滥用分类程序以掩盖公共部门浪费、欺诈和管理不善的经典之作。

在技术方面,Dieter Gollmann 的计算机安全 [779] 等教科书介绍了 MLS 系统,而许多关于实际 MLS 系统的已发表论文可以在两个会议的会议记录中找到:学术会议是 IEEE安全与隐私研讨会(在行业中被称为“奥克兰”,因为它曾经在这里举行),而 NSA 供应商社区的非机密 bash 是计算机安全应用程序会议 (ACSAC),其会议记录(如奥克兰会议)由IEEE。 Fred Cohen 使用病毒破坏 MLS 系统的实验在他的书“A Short Course on Computer Viruses”[450] 中有所描述。该领域的许多经典早期论文都可以在 NIST 档案 [1395] 中找到;直到 1999 年,NIST 一直在举办关于多级安全性的系列会议。最后,史蒂夫·利普纳 (Steve Lipner) [1171] 撰写了橙皮书的历史;这也讲述了美国空军早期参与的故事以及从 WWMCCS 等系统中学到的东西。