在Linux安全模块框架授权钩放置的一致性分析

TRENT JAEGER

IBM TJ Watson研究中心的安东尼·爱德华Symbian公司和

小榄镇张

IBM TJ Watson研究中心

## 2.2 关系验证

图2示出的概念之间的关系。

1. 识别控制操作： 找到一组定义通过所有安全敏感操作访问的调解接口的操作。
2. 确定授权要求： 对于每一个控制操作，确定必须由LSM钩子被授权的授权要求（即政策）。­­

⑶验证完成授权：对于每个控制动作，验证正确的授权要求是由LSM钩授权。­

⑷检查钩放置净度：实施政策操作来控制操作应该从他们的授权，很容易识别

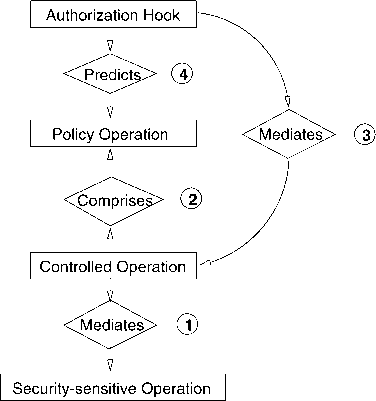


图2的授权概念之间的关系。验证问题是：（1）确定控制操作; （2）确定的授权的要求; （3）验证完整授权; 和（4）验证钩放置清晰度。­

挂钩。否则，即使给源琐碎的变化可能会使钩

无法操作。

其基本思想是，我们确定的控制操作及其授权要求，那么我们验证授权挂钩适当调解的控制操作。首先，我们需要确定在内核中代表控制操作。其次，由于控制操作是在比政策操作（即授权要求）一个较低的水平，我们需要每个控制操作的授权要求，可确定的方法。第三，我们需要比较到预期的授权要求进行的LSM钩子授权。这些任务是复杂的对内核的授权，所以很明显，自动化的支持是必需的。­­­­

控制操作来授权要求的映射不一定是静态的。例如，许多相同的操作都可以在开放阅读作为对写开放的文件的文件执行。因此，上下文也是在映射控制操作，以授权要求的决定性因素。我们的做法必须启用上下文的依赖，以进行有效的管理，从而控制运营和授权要求之间的预期关系可以进行测试。­

* 1. 相关工作

近日，静态分析来验证安全性已显示出希望。首先，现有的程序分析工具已被用于查找常见的安全错误，如缓冲区溢出和printf漏洞[球等。2003; Das等。2002; 拉罗歇尔和2001年埃文斯; Shankar等人。2001; Wagner等。2000。

其次，专门的工具已经被开发为寻找安全漏洞，如xgcc [恩格勒等。2000]，ITS4 /大鼠[Viega等。2000]，MOPS [陈和瓦格纳2002]，MAGIC [沙基等。2003]等[Ganapathy等。2003]。­

静态分析工具基于编程语言的形式属性，因此它们可用于完整的分析（即没有出现假阴性）。但是，静态程序验证的计算成本高昂，所以简化常常在分析模型方面。这些简化可导致更保守的分析（即，更多的假阳性）或某些特性（即，假阴性）的抽象。此外，静态分析工具可能需要的精力用于代码的注释，这是必要建立所需的分析模型显著量。­­­

专业分析工具专注于特定类型的ofbugs。恩格勒等。能够延长GCC的，叫xgcc， 做源分析，他们称其为 元编译[阿什克拉夫特和2002恩氏; 恩格勒等。2000; Hallem等。2002]。规则语言，称为金属，用来表示在高级语言进行必要的分析说明。而不是直接注释的代码中，金属规范定义的有限状态自动机，指导分析引擎。由于规则匹配多个语句，注释工作的量减少。各种软件漏洞，包括安全漏洞，已经被这个工具[阿什克拉夫特和2002恩格勒]研究发现。

大多数的专业分析工具缺乏完整性（即，可能会导致假阴性），但MOPS专门旨在为便于规范和分析的完整性[陈和2002年瓦格纳。使用MOPS，安全性被表达为有限状态自动机和方案表示为下推自动机。数据流是不是表示，因此混叠和价值的关系被忽略。然而，对于许多分析有用的错误仍然可以发现[陈等人。2004年]，而且往往可以显示很多数据流量关系不会通过其他手段[Zhang等人存在。2002]。­­­

在另一项努力，我们使用一个程序分析工具，CQual [Foster等。1999]，在一个方法来找到LSM钩放置错误静态[Zhang等人。2002]。使用GCC分析自动化CQual注释，我们就可以执行验证所有的控制操作由至少一个LSM钩介导的CQual分析。在一般情况下，我们还需要验证时，其所需授权钩子已经查了控制，操作只运行。CQual提供了可用于定义预期的授权类型点阵，虽然它是复杂的概念得到它的权利。此外，在控制运营和授权要求之间的关系的背景下，相关性是超出了CQual可以处理。­­­

Java静态分析工具，叫 JaB.A[Koved等。2002年]，已被用来收集对Java控制操作的实际授权。对于我们而言，这种方法有两个缺点：（1）它不分析Linux内核的C代码和（2）它没有提供有关所做的授权是否正确指导。关于第一点，我们实际上定义从C到雅巴分析概念[Zhang等人翻译。]，并建立了一个原型实现。关于第二点，JABA确实提供了一种上下文敏感控制流图，并且可以利用任何分析上下文敏感的数据流图。因此，我们将研究在第5生成分析日志使用这些图形。

由于使用这些方法的复杂性，我们发现，运行时数据采集帮助我们在得到准确的数据，很快，这样我们就可以探讨可能的分析选项。从检查收集到的数据，我们已经开发出我们在本文中，使我们能够确定是否适当的授权挂钩用于控制操作检查描述一致性分析方法。最终，方法是独立的，我们是否做在运行时或通过代码的静态分析收集到的数据一致性分析。在本文中，我们将考察数据采集的两种手段。­­

另一个相关的问题是系统的认证。从历史上看，黄皮书[NCSC 1985]用于在安全操作系统的施工指导，但现在正由通用标准[1998 ITSEC]取代。然而，认证工作是临时性的和费力的，并没有普遍提高通常使用的操作系统的安全性是成功的。古特曼认为在他的论文[古特曼2000]该认证的方法，包括正式的验证工具，是注定要失败的，除非他们表示在源代码级的概念。古特曼还主张的静态组合和运行时分析。我们使用不同于认证，在某种意义上说，它检查特定的错误，而不是提供一个自上而下保证整个系统满足其需求的方法。一个有趣的研究问题是，这样的检查有足够的广度和深度，是否可以提供一个信心堪比认证。与认证，源代码的发展这样的信心可以保持。­­­

1. 解决方案说明

关键的洞察力，我们在LSM框架运行时分析杠杆的LSM授权钩放置在很大程度上是正确的，使得不一致相对于常态的情况下很可能是一个指示错误的。例如，它会被认为是不寻常的，如果一个特定的控制操作对同一系统调用的不同运行不同的授权要求。虽然这种认识并不保证我们发现所有LSM钩放置错误（见第6节），它可以让我们确定了一些错误，并一直担任该工具开发宝贵的指导。

在所有下面的讨论中，我们采用了以下假设。首先，我们利用在控制操作对象在一个类型安全的方式在内核中进行处理的假设。这并不能否定任何发现错误的，但也可能有其他错误也是如此。其次，我们假设访问授权的数据类型的对象定义一组控制操作（即调解界面）。这些数据类型是对应于系统调用的概念（例如，文件，索引节点，插座，skbuffs，IPC消息队列，等等）的那些。到内核​​数据的访问旨在经过这些数据结构。虽然我们还没有明确证实这一点，我们已经做了其他地方提出了更详细的分析[爱德华兹等人。2001]。­­

* 1. 授权一致性

我们首先定义一个控制操作和一组授权要求之间的一致性。­

定义 1 （授权一致性）。 一个控制操作和一组授权需求（被授权即政策操作）之间的关系是 ­­利弊，3S ^ ^恩 如果每当执行受控的操作授权钩相关联的那些授权要求被调用。

我们发现，这种形式的一致性不是绝对需要的。在不同的系统调用，其具有不同的授权的情况下，可能会出现一个控制操作的执行。显然，在这种情况下满足授权要求会有所不同。­

因此，有必要能够限定其中一致性预期上下文。在一般情况下，上下文可以是任意的，但我们的经验是，三种类型的上下文关系：⑴ 系统调用;⑵系统调用与特定的输入（例如，标志）; 和（3）一组特定的控制操作。在第一种情况下，所述授权是在大的系统调用（例如，的fcntl）。这些授权适用于系统调用所有的控制操作。在第二种情况下，授权取决于一些参数到系统调用，通常是一个标志（例如，开放读）。因此，一些系统调用来下一个背景和一些以另一个。在第三种情况下，该组控制操作的外观，独立于它们出现的系统调用的需要特定的授权（例如，与访问相关联的该组所有者字段的操作）。在这种情况下，一致性忽略了系统调用的信息。

定义2（执行上下文）。 一个 执行上下文定义了一组执行路径。执行上下文可通过（1）的系统调用（所有执行）来定义; （2）系统调用特别参数值（或值的范围）; 和（3）一组受控操作（包括它们所有的路径）的。其他情况下的定义是可能的。­

我们的解决方案必须支持我们预期一致的授权环境的描述。通常，在静态分析上下文敏感的数据流是指不同的输入之间进行区分，以相同的功能。在这种情况下，上下文灵敏度是窄得多（仅取决于系统调用），或者可以忽略剩余的上下文的大部件（受控的操作集）。这类分析需要的静态分析器显著量注释的，并且可能取决于变量静态分析器（例如，用户提供的标志）的理解之外。例如，JABA完全忽略基本类型的值，但显然这些可以影响分析。­­­­­­

* 1. 授权一致性水平

一个执行环境通常由许多控制操作的，所以它是有帮助的聚合控制操作是在以同样的方式是一致的。

表一，授权一致性水平：名称和授权影响

|  |  |
| --- | --- |
| 水平 | 授权 |
| 系统调用 | 在系统调用的所有控制操作 |
| 系统调用输入 | 在同一个系统调用相同的输入所有控制操作 |
| 数据类型 | 相同的数据类型的对象，所有控制操作 |
| 宾语 | 同一对象上的所有控制操作 |
| 会员 | 上相同的数据类型的所有控制操作，访问同一部件，用相同的操作 |
| 功能 | 在相同功能的所有成员一样控制操作 |
| Intrafunction | 同样的控制操作实例 |
| 路径 | 相同的执行路径相同的控制操作实例 |

例如，如果所有的上下文控制操作具有相同的授权，那么我们就可以在大的，而不是单独的操作视图相对于上下文一致性。­

我们发现，我们可以描述每个受控运行时，它是在特定情况下由一组我们称之为离散值的执行被称为授权钩之间的一致性­一致性水平。 此外，一致性水平形成总阶如下。

定义 3 （一致性级别总订单）。 如果两个不同的控制操作权限一致为同一值oflevel我，那么他们的授权级别的任何值一致 ­Ĵ 哪里 I>Ĵ 在里面 一致性水平全序 （见表一）。

如果两个不同的控制操作都在同一个对象上执行的，但他们有一致的授权，那么该成员和访问这些操作的值不影响一致性。例如，如果一个特定的对象上的所有控制操作具有相同的授权要求，那么也不要紧成员访问是什么。表1列出了离散的一致性水平。我们统称这一组级别作为授权一致性水平。­ 这些级别包括受控OP- erationJs执行的各个方面，包括（在函数系统调用，系统调用输入，功能，位置，路径控制操作）下它被执行的情况下，将其在（数据类型执行的对象和对象），并执行操作（构件和访问）。

一致性水平骨料控制操作成 稠度等级 所有的控制操作具有相同的权限挂钩称为鉴于目前的位置。

定义 4 （一致性类）。 两个不同的控制操作属于同 ­稠度等级 对于执行上下文，如果他们有相同的权限挂钩称为每次他们在这方面的执行。

* 1. 授权一致性影响

控制操作可以通过授权一致性的分类划分控制操作分为两类：⑴ 众所周知异常

和（2）稠度类，它们的授权要求需要验证。在第一种情况下，我们考虑一些授权的一致性水平是非法的。我们定义下面这些情况不变。在第二种情况下，我们必须确定在执行上下文每个受控操作的最大一致性水平是否指示上可接受的授权要求或没有。例如，如果一个受控操作属于一组在物体一致性电平控制操作，这表明所有的对象上的控制操作具有相同的授权检查。它是那么的手动任务，以确定这是否是正确的。然而，一致性的聚集体的数量表示一个分区的受控操作成具有相同授权的最大尺寸的类。­­­­­

* + 1. 异常。 授权于表I中的双线下方的水平的一致性， intrafunction 和 路径，一直被认为是反常的。这些类型的敏感性意味着一个函数（intrafunction）内执行路径（路径）或位置确定特定受控操作的相同部件上的授权要求。­

下面不变正式表达了我们的路径不一致不变。

定义5（路径不一致不变）。

VC1，C2式C， E1，E2ËË（C1 = C2）八（E1 = E2） - ^（C1， E1） = E（C2， E2）（1）

这不变的状态，同样的控制操作（C1 = C2）在同一事件中运行（E1 = E2通过系统调用和它的输入端定义）必须具有相同的授权要求（由函数定义 [R）。也就是说，事件内执行路径可以在不影响受控operationJs授权要求。­

同样，我们定义intrafunction不一致不变。

定义6（Intrafunction不一致不变）。

VC1，C2式C， E1，E2ËË， （F（d）= F（C2））八

（中号（C1） = M（C2））八（E1 = E2） - K（C1，E1） = E（C2，E2）（2）

在这种情况下，在同一个函数两个受控操作（通过函数计算 F），并且使相同的成员访问（通过函数计算 中号 ）必须具有相同的授权要求 [R。

* + 1. 授权一致性类。对于其他情况下，我们不能很容易地识别为错误。相反，我们分区控制业务并入其授权的一致性类，并确定它们的授权要求是否正确。­

授权稠度等级计算如下。起价为最高（系统调用）每个一致性水平，我们分区控制操作进入一致性类，所有控制操作具有一致性水平相同的值，然后我们测试类是否也有相同的权限。如果没有，那么我们尝试基于两个层面，并再次测试下一个较低的水平和分区。这种方法反复进行，直到我们已经交给的各项控制操作一致性类。­­

分类是通过一致性水平定义。系统调用级别，所有系统调用的控制操作都在一个班。为系统调用输入，相同的系统调用的，并用相同类型的输入所有受控操作是聚合（见3.4节）。对于数据类型级别，受控操作由系统调用，输入，并且该操作的对象的数据类型进行分类。因此，在分析的每一步中创建依次更细的分区。

甲分类成功（即，为x一致的，其中x是水平），如果它是在其中所有在该类受控操作具有相同的授权的第一级。需要注意的是其他类，在具有相同授权凝聚而形成的最大规模的类相同的一致性。一旦类被创建它是一个手动过程来验证每一类的授权是正确的。对于文件系统，类的数量是足够小，手工验证是可行的。­

作为一个例子，考虑读系统调用。文件操作是数据类型一致的，因为在文件对象的所有控制操作被授权读取。手动验证涉及到检查读取文件的权限就足够了。由于读的授权也适用于文件的inode，我们纪念文件的inode作为授权读为好。然而，分类后，一个索引节点的控制操作未被授权。这是一个不同的对象，所以索引节点的操作可以是目标一致的。这是该文件的目录inode的操作，以确定是否一个信号应该在这个目录中读取的结果被发送。[[1]](#footnote-1)

除此之外，当授权完全缺失，用于识别错误最常见的方式是找到两个分类（即两个积累了与不同的授权）执行的重要的共同操作。这种情况多发生在那里的fcntl两个不同的类别（根据不同的系统调用输入）在相同的操作F\_­­〇wner字段（参见4.2.4节）。

* 1. 必要的数据采集

通过登录系统调用进入/退出/参数，函数入口/出口，控制操作（即，对象，数据类型，部件，和操作），和授权，我们收集所有的一致性水平必要的值。所有的信息可以很容易地记录，但有意义的对象标识符和系统调用输入的变化的识别需要进一步的分析。

在执行期间，对象通过函数指针引用，但这不是必需的对象的足够的识别。例如，一个inode具有在授权使用持久标识符（即，设备，节点号码）。因此，对于每个数据类型，我们定义了用于计算特定的方法其

表二。日志记录类型

|  |  |
| --- | --- |
| 记录类型 | 数据 |
| 控制操作授权函数入口函数退出 | 上下文ID控制操作ID OID上下文ID授权ID OID上下文ID指令地址上下文ID |

对象标识符。这些标识符用于确定对象上的所有操作和授权。

在系统调用的情况下，我们假定在同一个变量使用的对象具有相同的授权要求。为了模拟这种情况，我们使用第一控制操作在其中一个对象显示为一个标识符。如果两个对象在相同的受控操作必须将它们分配给同一个可变第一访问（因为变量将在两个受控操作相同）。然而，不同的执行路径可能导致在一个不同的受控操作中使用第一相同的变量之中。然而，与相同的授权要求类聚合将合并这些情况，所以这个假设已经被证明有效。

该系统调用的参数改变几乎每一个电话，但只有少数的论点真正影响的授权（例如，存取标志上打开）。因此，我们收集到的参数，但是只能使用我们发现影响授权要求做到分区的参数。只有很少的系统调用我们已经讨论基于他们的输入参数，如打开，读写控制，和fcntl有不同的权限。因为不同的授权是根据不同的输入所使用的，这些系统调用都比较复杂，因此，更容易出错。­­

1. 其实，这个对象也应该读LSM钩授权，所以我们把它添加到组通过此挂钩授权的对象。 [↑](#footnote-ref-1)