SEHive:基于类型增强的 Hive 强制访问控制模型与实现

汤定一 韩伟力
(复旦大学软件学院 上海 200433)

摘 要 在借鉴 SELinux 模型的基础上,提出面向 Hive 的基于类型增强的强制访问控制模型 TE-MAC。该模型将主体与域关联,客体与类型关联,根据域与类型之间的访问规则控制访问,最大限度地减小用户可访问资源的范围,实现最小特权原则。同时引入组层次关系,便于结构化的权限管理。基于原型系统 SEHive 的实现,显示其对 Hive 中敏感数据强制访问控制的可行性。

关键词 大数据 Hadoop Hive 类型增强 强制访问控制

中图分类号 TP3 **文献标志码** A DOI:10.3969/j.issn.1000-386x.2022.07.043

SEHIVE: TYPE-ENHANCED MANDATORY ACCESS CONTROL MODEL AND ENFORCEMENT FOR HIVE

Tang Dingyi Han Weili
(*School of Software, Fudan University, Shanghai 200433, China*)

Abstract Referred to SELinux, this paper proposes a type-enhanced mandatory access control model named TE-MAC for Hive. The model associated subject with domain, object with type, and controlled access according to the access policies between domains and types. It minimized the range of accessible resources, and implemented the principle of least privilege. And group hierarchical relationships were introduced to facilitate structured authority management. Based on the implementation of the prototype system SEHive, it shows the feasibility of mandatory access control over sensitive data in Hive.

Keywords Big Data Hadoop Hive Type enhancement Mandatory access control

0 引 言

据 IDC 估计,到 2025 年每年将产生 163 ZB 的数据[1]。企业可以通过数据分析得到有价值的信息。例如,大型互联网公司收集海量的用户行为信息用于发现客户需求进而提供有针对性的营销;金融企业根据海量历史交易数据训练欺诈检测模型以控制风险;医疗机构收集健康数据用于生成更好的诊疗模型。

随着数据在数量、种类和产生速度方面的增长,需要面向大数据的数据分析工具。Apache Hive 作为基于 Apache Hadoop[2] 的数据仓库服务,提供与 SQL 查询语言类似的 HQL 查询语言,用于方便高效地对海量数据进行分析。Hive 构建在 Hadoop 2.x 核心组件(Hadoop 通用,Hadoop 分布式文件系统(HDFS),MapReduce 和 YARN)之上,为用户提供了挖掘数据价值的能力。

为保护存储的数据安全,当前 Hive 访问控制模型包括以下三种:基于存储的授权、基于 SQL 标准的授权和 Hive 默认授权。(1) 基于存储的授权通过 HDFS 提供一种自主访问控制模型,虽然能够保护元数据不被恶意用户破坏,但是没有提供细粒度的访问控制。(2) 基于 SQL 标准的授权提供了一种基于角色的访问控制模型。通过 SQL 语句,可以创建角色,给用户赋予角色和给角色赋予操作数据的权限。但该模型需要确保用户仅通过 hiveserver2 访问 Hive 服务,且必须限制用户执行非 SQL 语句。(3) Hive 默认授权的设计目的仅为防止用户误操作,而非防止恶意用户访问

收稿日期:2020-04-15。国家自然科学基金重点项目(U1836207);上海市创新行动计划项目(18511103600)。**汤定一**,硕士生,主研领域:大数据与信息安全。**韩伟力**,教授。

未被授权的数据。

SELinux 采用强制访问控制(MAC)解决自主访问控制安全性不足的缺陷。Linux 自带的自主访问控制(DAC)依据程序拥有者与文件的读、写和运行权限来决定是否允许访问。使用 DAC 存在两个主要问题:(1) root 具有最高权限,文件拥有者可以变更文件读写权限。如果恶意用户获取了 root 权限,那么他就能对所有数据进行恶意操作。(2) 文件拥有者如果错误地将文件配置为任何人可读写,那么非常容易遭到恶意用户的攻击。使用 MAC 可以解决这两个问题,通过针对特定程序与特定文件进行权限管理,使得以 root 身份运行的程序也不能任意访问文件,即使文件所有者错误地分配了权限仍可能确保安全,因为每个主体只能根据域与类型的对应关系访问客体。

大数据中通常包含大量个人隐私数据,例如私人金融账户和交易信息、健康诊疗报告信息等。在处理包含隐私信息的数据时,如果在没有额外保护的情况下将其存放在集群中,将会使所有用户都能访问它,从而造成隐私泄露的风险。因此,如果要对隐私数据进行分析,必须进行身份验证和访问控制。

现有开源工具,如 Apache Ranger[3],Apache Sentry[4],用于提供对包括 Hive 在内的 Hadoop 生态系统的细粒度访问。

Sentry 可为存储在 Apache Hadoop 集群上的数据和元数据提供基于角色的细粒度授权。它的特点是对于每个数据库或框架策略是独立的,且具有由独立的管理员维护的能力。它将权限分配给角色,再将角色分配给组,这样间接通过组将角色分配给成员用户。Ranger 提供基于访问策略的授权模型,由允许访问控制列表(Allow ACL)和拒绝访问控制列表(Deny ACL)来描述访问控制。它的特点是提供了一个可统一控制整个 Hadoop 集群安全性的管理控制台。它支持直接将权限赋予用户和组。

文献[5]概括性地将 Hadoop 原有授权模型、Apache Ranger 和 Sentry 使用的模型总结为 Hadoop 生态系统访问控制模型 HeAC。该模型直接或间接通过组和角色向用户分配不同的权限。在 HeAC 的基础上提出了将属性加入 RBAC 模型以实现更细粒度的权限管理[6]。该扩展将属性引入用户、组、服务和数据对象。同时,该模型也引入了组层次结构,根据组上定义的偏序关系,高级组从低级组继承角色。文献[7]也对基础 RBAC 模型进行了扩展,在 HDFS 文件层次结构中抽象出了数据组的概念,通过定义不同的数据组和其上的继承关系,用于简化多源异构数据的管理。这些工作和文献[8-9]均为基于 RBAC 模型的扩展。

文献[10]将 ABAC 模型与原生 Hadoop 相结合,提出了一个纯 ABAC 访问控制模型。通过主体、对象、环境或上下文来进行访问控制决策,满足多个用户在不同时间、位置和条件下以不同粒度访问数据的安全需求。文献[11-12]也将 ABAC 模型及其扩展应用到 Hadoop 平台。

综上所述,现有工作通常都是在 RBAC 模型和 ABAC 模型的基础上进行扩展。然而,在金融和医疗等对于隐私数据的保护有更高安全性要求的行业中,现有工作没有给出较好的解决方案。因此,我们提出了 Hive 上基于类型增强的强制访问控制模型(TE-MAC)。该模型机制与 SELinux[13] 采用的类型增强访问控制机制类似,解决了 Hive 自带的 DAC 模型安全性不足的问题,最大限度地减小用户可访问资源的范围,实现了最小特权原则。TE-MAC 将访问主体与域关联,数据对象与类型关联,授权规则由策略文件中域与类型的允许访问规则组成。同时,引入了用户组层次关系,使得数据组拥有的域可以通过偏序关系继承,便于结构化管理权限。

1 SEHive 访问控制模型:TE-MAC

本文提出了在 Hive 上的类型增强访问控制模型 TE-MAC。此模型将域分配给用户和组,将类型分配给对象,通过域和类型之间的规则进行访问控制。其概念模型如图 1 所示。

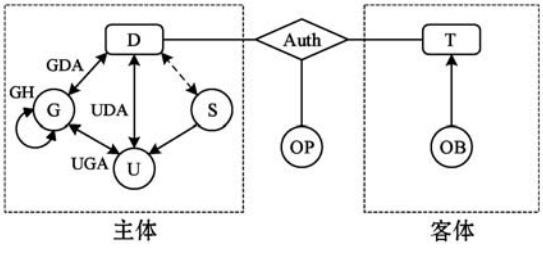

图 1 TE-MAC 模型对象间关系图

TE-MAC 的基本组件包括:用户(U)、组(G)、会话(S)、域(D)、类型(T)、数据对象(OB)和对象上的操作(OP)。用户是使用 Beeline 客户端通过连接到 hiveserver2 进行交互以访问 Hive 服务和数据对象的人员。组是系统中具有相似组织职权的用户的集合。域和类型对应着访问的主体和客体,域与用户和组关联,类型与数据对象关联。在 TE-MAC 模型中,制定域到类型允许操作的规则,以实现类型增强的强制访问控制。

一个用户可以属于由 *directUG* 函数定义的多个组。根据 *directUT*,用户可以拥有多个域。同理,根据 *directGT*,组也可以拥有多个域。*OBJECT-PRMS* 是数

据对象权限的集合,它是数据对象和操作的叉积的集合。

从用户到其拥有的权限可由如下方法得到。首先,用户通过直接拥有的域或间接通过所属组拥有的域得到了拥有的域的集合。其次,通过域与类型之间的允许访问规则得到了域拥有的 *OBJECT-PRMS* 的集合。最后,通过用户通过域和类型,间接得到了拥有的 *OBJECT-PRMS* 的集合。这与基于存储的 Hive 授权不同,其 *OBJECT-PRMS* 直接被分配给用户和组。也与基于 RBAC 的 Hive 授权模型不同,其中通过授予或撤销用户角色来给用户分配或删除权限。这反映了 TE-MAC 模型的优势,可以更灵活地修改用户拥有的域、数据对象对应的类型,同时支持域和类型之间访问规则的修改,这对于灵活多变的数据组织形式更为方便。

在 TE-MAC 模型中,域可以直接分配给用户,也可以分配给组,相比现有的 RBAC 模型只支持给组分配角色更灵活。组层次结构(TH)被引入系统,G 上定义了偏序关系,写为 $\geqslant_G$。组的继承关系从低到高。比如,$g_1 \geqslant_g g_2$ 表示 g_1 组继承 g_2 组拥有的域。在这个例子中,称 g_1 是资深组,g_2 是初级组。在组层次结构(GH)中,组的有效域是直接分配给该组的域与其所有初级组的有效域的并集。该定义是递归的,其中最初级的组具有相同的直接分配的域和有效域。用户的有效域是用户拥有的有效域和用户所属组拥有的有效域的并集。

用户创建的会话(Session)具有用户的全部权限。会话需要具有多种权限,以访问所需的 Hadoop 以及 Hive 中的服务和对象,但会话对于服务如 Yarn 管理器的访问权限不在 TE-MAC 模型的讨论范围。TE-MAC 模型的主要优点是解决了 Hive 自带的 DAC 模型安全性不足的问题,最大限度地减小用户可访问资源的范围,实现了最小特权原则。此外,它引入了组层次结构的概念,该概念使得组可以继承拥有的域并减轻安全管理员的负担。

1.1　基础对象和对象间关系

(1) 用户(U)表示具有合法访问 Hive 服务的用户。组(G)表示 Hadoop 或轻量级目录访问协议(LDAP)中的用户组。会话 S 表示通过 Beeline 客户端访问 hiveserver2 的连接,会话与用户之间是多对一关系,即每个用户可以发起多个会话,而每个会话对应一个用户。

(2) 数据对象(OB)表示 Hive 中的数据库、表、列等对象。

(3) 用户操作(OP)表示 Hive 中对数据对象的操作,包括增、删、查询等。

(4) 域(D)可以直接分配给用户和组,再间接通过用户和组分配给会话。会话与域之间是多对多关系,即一个会话可以拥有多个域,一个域也可以被多个会话拥有。类型(T)与数据对象是一对多关系,类型可以对应多个数据对象,每个数据对象唯一对应一个类型。

(5) 用户与组的关系 $directUG: U \rightarrow 2^G$,等价地,$UGA \subseteq U \times G$。

(6) 用户与域的关系 $directUD: U \rightarrow 2^D$,等价地,$UDA \subseteq U \times D$。

(7) 组与域的关系 $directGD: G \rightarrow 2^D$,等价地,$GDA \subseteq G \times D$。

(8) 用户组的层次关系 $GH \in G \times G$,组的偏序关系使用 $\geqslant_g$ 表示。令 g_i 和 g_j 为两个组,如果 $g_i \geqslant_g g_j$ 成立,则 g_i 继承 g_j 的权限。

(9) 数据对象与类型之间的关系 $directOBT: OB \rightarrow T$,表示每个数据对象与一个类型对应。

1.2　用户和组的有效域

(1) 组的有效域代表组拥有的域的集合,由组直接拥有与间接通过偏序关系继承的域构成。定义为:

$$effectiveGD(g_i) = directGD(g_i) \cup \left(\bigcup_{\forall g \in \{g_j \mid g_i \geqslant_g g_j\}} effectiveGD(g) \right)$$

(2) 用户的有效域代表用户拥有的域的集合,由用户直接拥有与间接通过组拥有的域构成。定义为:

$$effectiveUD(u) = directUD(u) \cup \left(\bigcup_{\forall g \in \{directUG(u)\}} effectiveGD(g) \right)$$

1.3　会话的有效域和权限

(1) 用户与会话的关系 $userSession: S \rightarrow U$。

(2) 会话的有效域 $effectiveD: S \rightarrow 2^D$,等价地:$effectiveD(s) \subseteq effectiveUD(userSession(s))$。

(3) 权限判断规则:$Authorization_{(d,t,ops)}(s:S, ob:OB, op:OP) = (d \in effectiveD(s)) \wedge (directOBT(ob) = t) \wedge (op \in ops)$。

策略文件中定义的规则为 $(d:D, t:T, ops:2^{OP})$ 三元组,表示允许域以给定的操作方式访问类型,其中 *ops* 表示操作的集合。权限判断过程分为两步:首先得到会话拥有的域集合,以及数据对象的类型。其次,在规则中进行匹配,是否存在这样一条规则,使得域 d 属于 $effectiveD(S)$,且类型 t 与数据对象的类型 $directOBT(ob)$ 一致,且该请求的操作 *op* 在给定的操作方式集合 *ops* 当中。

2　TE-MAC 模型的实现

在开源安全工具 Apache Sentry 的基础上进行扩

展,实现了本文提出的 TE-MAC 模型。Sentry 的主要组件是策略引擎,策略提供器和插件。策略引擎用于根据策略控制所有对服务和数据对象的访问。策略提供器用于解析和存储策略,支持基于文件和基于数据库的策略提供器。插件是绑定到 Hive 服务的组件。当产生访问数据对象的请求时,插件将会调用策略引擎,该引擎将根据策略提供器中定义的策略对请求进行评估然后做出判断。如何基于 Sentry 实现强制访问控制机制面临三个主要挑战:1)如何使得用户在访问时通过 TE-MAC 机制,而非 Sentry 原有 RBAC 策略,这需要系统化的实现。2)策略如何存储和维护,Sentry 维护的是 RBAC 的策略,而 TE-MAC 模型需要维护和提供 MAC 策略查询。3)如何实现 SEHive 的鉴权。

对于第 1 个挑战,本文在插件组件中重新实现 Hive 会话钩子接口以配置访问控制所需要的属性。实现 Hive 权限验证接口以重新定义权限判断机制和返回结果过滤机制。对于第 2 个挑战,本文在策略提供器中实现提供器后台接口类,用于加载和解析策略文件。实现列出权限方法以返回缓存的权限。对于第 3 个挑战,本文在策略引擎组件中,实现策略引擎接口类,用于控制访问请求。

模型在运行时分为两个阶段,第一个阶段是策略制定和加载。安全管理员使用策略文件来制定安全策略,由策略提供器解析策略文件,将策略进行缓存。

第二个阶段是鉴权阶段,如图 2 所示。当在 Beeline 上键入一个连接到 hiveserver2 的命令时。该命令经过解析器阶段和语义分析器阶段。然后进入 SEHive 授权框架阶段。通过在 hive-site.xml 文件中配置的绑定层(Hive 插件)探测到这次请求,将这次访问的主体和客体传递给 SEHive 策略引擎。策略引擎获取参数,从策略提供器中查找与之相关的规则,然后判断主体能否访问客体。

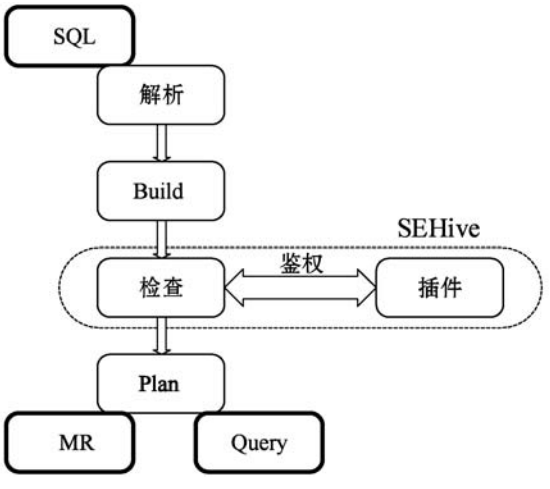

图 2 HiveQL 查询流程

2.1 插 件

首先配置 Hive 的会话钩子(Session Hook)属性,该钩子函数会在用户通过 Beeline 客户端连接到 hiveserver2 的初始化阶段调用,用于配置授权所需准备工作。通过实现 HiveSessionHook 接口类来进行配置。在钩子函数中,实现了以下 9 个操作。1)开启 Hive 授权验证,使得每次访问都经过图 2 中所示鉴权流程。2)配置 SemanticAnalizerHook,指定该会话在提交 HiveQL 请求时调用的鉴权方法。3)设置 DoAS 参数,该参数决定以会话用户身份或是运行 Hive 服务的用户身份访问 HDFS。同时,需要相应地在 Hadoop 配置文件中进行修改以决定是否允许代理。4)设置权限将表的所有权限赋予数据所有者。5)设置安全命令的白名单,仅允许用户访问安全命令,其中白名单设置与 Sentry 相同。6)对于每个连接用户,创建一个 HDFS 暂存目录,设置其目录的权限为 700,即只有会话用户拥有该目录全部权限。7)配置属性限制列表,确保安全属性不被用户修改。8)设置会话用户,为之后的鉴权方法做准备。9)配置作业属性为会话用户,使得 Map Reduce 任务的访问控制列表(ACL)为会话用户。

其次,实现 SemanticAnalizerHook 指定的鉴权方法。这就需要实现 HiveAuthorizationValidator 接口类中的 checkPrivileges 方法和 filterListCmdObjects 方法。其中 checkPrivileges 方法用于检查会话用户是否有在给定的输入和输出对象上执行给定的操作类型的权限。filterListCmdObjects 方法用于根据当前用户的权限对结果选择过滤。首先通过将操作映射到权限的预定义映射表,将操作转换为对应权限。然后树形遍历访问列表中的数据库和表,将可访问的数据库或表作为结果返回。如果没有访问权限或访问列表为空将通过抛出错误来拒绝访问。

2.2 策略提供器

策略提供器负责从策略文件中加载和解析策略,然后在数据结构中进行缓存,并提供其他模块可以获取权限的方法。需要实现 ProviderBackend 接口类。其中包含用于获取域到类型访问规则的 getPrivileges 方法,以及用于获取用户所拥有域和数据对象所拥有类型的 getTypeEnforcement 方法。实现 PolicyFiles 类用于加载和解析基于 TE-MAC 模型的策略文件。

2.3 策略引擎

策略引擎主要提供判断访问是否被允许的方法。根据传入的主体,客体和操作所需权限。首先根据主体请求策略提供器,得到会话的有效域集合 effectiveD(s)。

其次,得到客体的类型。最后,在访问规则中进行匹配,判断是否允许访问。通过实现 PolicyEngine 接口类的 validatePolicy 方法,同时该类需要维护 Provider-Backend 的对象用于获取权限。

2.4　策略文件

策略文件共分为 4 个部分。第一部分描述用户与域的映射和组与域的映射。它们是多对多的关系。第二部分描述不同组之间的偏序关系。组的层次结构可以视为有向无环图,每个偏序关系为图中的一条有向边。第三部分描述数据对象与类型之间的关系,每个数据对象对应一个类型。第四部分描述访问规则,一条规则包括由域,类型和允许的操作组成的三元组,表示允许域以给定的操作方式访问类型。

3　SEHive 的实现

本节将 Hive 置于 Hadoop 生态中,站在整体的层面来分析。图 3 展示了多层访问控制决策和执行点,用于对包括 Apache Hive 和 HDFS 之类的服务中的资源进行授权访问。该架构描绘了几个 Apache 项目如何协同以实现 SEHive。

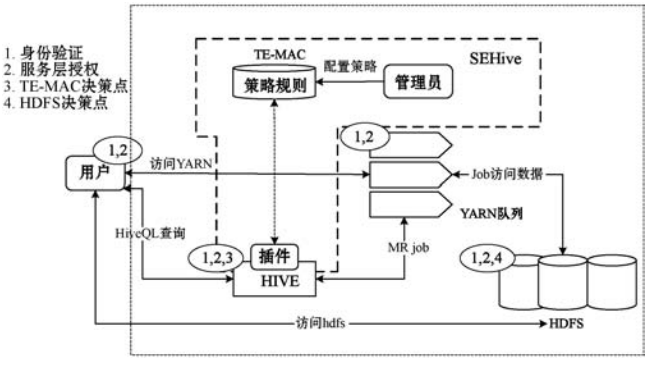

图 3　Hadoop 整体安全机制架构

首先,Kerberos 可用于用户和身份管理以提供身份验证。

用户通过身份验证后,第一层访问控制机制为服务层面的授权。该层控制对 Hive 等 Hadoop 生态系统服务的访问,远早于访问 Hive 服务所提供的数据。它还检查是否允许用户访问 Hadoop 守护程序(例如 Hadoop NameNode,YARN 资源管理器)以提交任务或查询状态。

通过服务层授权之后,当用户尝试通过 Apache Hive 数据服务发出 HiveQL 命令时,被 SEHive 模型在 Hive 中配置的插件检测到,调用 TE-MAC 模型中的策略引擎进行鉴权操作,这是第三层访问控制机制。由于 Hive 中的数据对象存储在 HDFS 中,可以根据需求配置为两种模式,其区别在于用户是否需要具有 HDFS 中对象的访问权限。因此第四层访问控制机制为 HDFS 层面的授权。

我们通过实际应用场景对这两种模式进行讨论。在模式一中,数据分析师可以通过 Apache Hive 使用 HiveQL 访问数据,但可能不允许通过 MapReduce 作业访问 HDFS 中相应的数据文件。在这种情况下,发出 HiveQL 命令的用户被更改为运行 Apache Hive 服务的用户。在模式二中,审计员可能需要同时具有 Apache Hive 和 HDFS 中对应数据访问权限。通过配置 SEHive 的插件组件中 DoAS 属性来满足这两个用例的需求。假设 Alice 是通过会话访问 Apache Hive 的用户,而 Bob 是运行 Apache Hive 服务的用户。将 DoAS 设为 false,运行 HiveQL 的用户是 Bob。对于 DoAS 为 true,Alice 在 Hive 和 HDFS 上都必须具有访问数据的权限。

发出 HiveQL 命令时,如果 SEHive 中的 DoAS 属性为 true,还需要用户具有访问 YARN 队列的权限,因为该命令会导致 MapReduce 或 Tez 任务,该任务也将提交给 YARN 队列。由于这些任务将访问 HDFS 中的数据,因此用户也应具有 HDFS 文件的权限。

通过这四层安全机制,SEHive 通过与其他安全机制集成,能够实现用户在发出 HiveQL 命令时得到保护。确保从身份验证到访问 Hive 服务授权,再到访问 Hive 中提供的数据和 HDFS 中存储的数据,以及执行 MapReduce 任务的 Yarn 队列在访问数据时的安全。

4　SEHive 的评估

本节通过实验来说明模型的有效性。实验在一台 CPU 配置为 Intel(R)i7-4790 CPU @ 3.60 GHz,具有 32 GB RAM,安装 64 位 Windows 系统的计算机上进行。使用 VMware 软件构建了 3 台 Ubuntu 16.04 虚拟机,每个虚拟机具有 4 GB RAM 和单核 CPU。每个虚拟机中安装 Hadoop 2.8.1,构成一个由三个节点组成的 Hadoop 集群,其中一个是主节点,另外两个是从节点。Apache Hive 2.1.1 安装在主节点上。

4.1　用　例

考虑以下应用场景,Hadoop 集群中有三个用户 Alice、Bob 和 Manager。这三个用户以不同的权限访问同一 Hadoop 集群中的数据。它们都能使用 Beeline 客户端访问 Hive 服务。Bob 允许查看 car database 下 customer 表中的数据,以及 car database 下 facilities 表中的数据,但不能对这两张表进行修改操作。仅允许 Alice 访问 car database 下 customer 表中的数据,同时

可修改该表中的数据。但 facilities 表对其不可见。而他们共同的上级 Manager 同时拥有他们的权限。

定义组 sale、analyst 和 manager。其中 manager $\geqslant_g$ sale, manager $\geqslant_g$ analyst。sale 组包含的域为 sale_t, analyst 组包含的域为 analyst_t。根据组的偏序关系, manager 组包含 salt_t 和 analyst_t 域。customer 表对应的类型为 customer_t, facilities 表对应的类型为 facilities_t。

权限定义如下:

策略 1: *Allow sale_t customer_t {getattr read write}*

策略 2: *Allow analyst_t customer_t {getattr read}*

策略 3: *Allow analyst_t facilities_t {getattr read}*

根据策略 1 可知,拥有 sale_t 域的用户拥有类型为 customer_t 的数据对象的查看、读和写权限。根据策略 2 和策略 3 可知,拥有 analyst_t 域的用户拥有类型为 customer_t 和 facilites_t 的数据对象的查看和读权限。通过组层次结构,简化了域的分配。

下面通过实验来验证模型的有效性。首先在策略文件中配置组的偏序关系、用户和组与域的对应关系、数据对象与类型的对应关系,以及访问策略。当 Hive 服务启动时,插件从本地策略文件加载策略。

依次使用 Alice、Bob、Manager 和任意其他用户,通过 Beeline 访问 hiveserver2。使用 select * from customer 和 select * from facilities 选择数据,其中 select 需要 read 权限。使用 load data local inpath'/tmp/tmp_file' into table customer 向 customer 表中写入数据,其中 load 需要 write 权限。各用户执行命令成功与否的状态如表 1 所示。实验表明,TE-MAC 能够最大限度地减小用户可访问资源的范围,实现了最小特权原则。

表 1 各用户对于表的访问权限

用户	Customer 表 read 操作	Customer 表 write 操作	Facilities 表 read 操作
Alice	√	√	×
Bob	√	×	√
Manager	√	√	√
Other	×	×	×

4.2 性能评估

本文通过衡量 SEHive 模型在访问控制阶段所需时间随访问控制策略数量的变化来进行性能评估。

如图 4 所示,SEHive 模型在访问控制阶段的耗时随策略数量增加呈增长趋势。耗时的增加主要是由于在更多的策略中进行检索所带来的。当策略数量在 1 000 以内时,SEHive 耗时在 3 ms 以内。当策略数量达到 10 000 条时,耗时 20 ms。当策略数量达到 100 000 条时,TE-MAC 耗时 356 ms。而 HiveQL 查询耗时通常在秒级别或者更多,因此 TE-MAC 在访问控制阶段的耗时对查询总耗时影响较小。

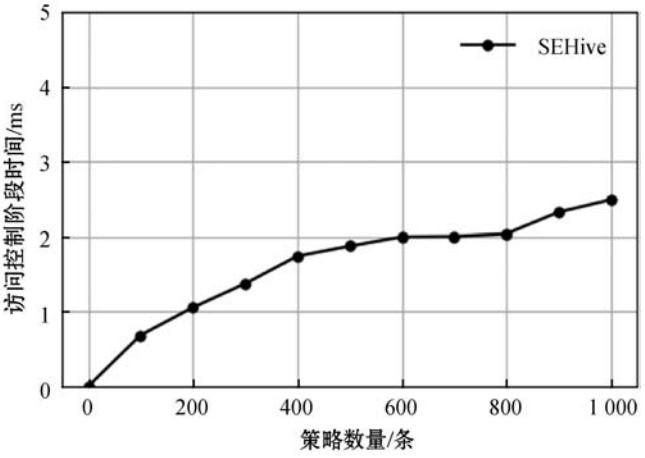

图 4 访问控制阶段时间随策略数量变化

5 结 语

本文针对 Hive 中如何实现强制访问控制,提出了 TE-MAC 模型。该模型采用了与 SELinux 中类似的类型增强的强制访问控制机制,将主体与域关联,客体与类型关联,根据域和类型之间的访问规则控制访问,最大限度地减小用户可访问资源的范围,实现了最小特权原则。同时,引入了组的层次结构,通过偏序关系继承域,便于结构化的权限管理。本文在 Apache Sentry 的基础上实现了 SEHive 模型。同时,将 Hive 置于 Hadoop 中,通过多层访问控制实现了 SEHive。最后,通过实验分析验证了模型的有效性,实验说明 SEHive 在访问控制阶段的耗时对 HiveQL 查询影响较小。

参 考 文 献

[1] Reinsel D, Gantz J, Rydning J. Data age 2025: The evolution of data to life-critical[R]. Don't Focus on Big Data, 2017.

[2] Shvachko K, Kuang H, Radia S, et al. The hadoop distributed file system[C]//2010 IEEE 26th symposium on mass storage systems and technologies(MSST). IEEE, 2010.

[3] Apache Ranger[EB/OL]. [2022-06-28]. http://ranger.apache.org/.

[4] Apache Sentry[EB/OL]. [2022-06-28]. https://sentry.apache.org/.

[5] Gupta M, Patwa F, Benson J, et al. Multi-layer authorization framework for a representative Hadoop ecosystem deployment[C]//Proceedings of the 22nd ACM on Symposium on Access Control Models and Technologies. ACM, 2017: 183-190.

(下转第 294 页)

[6] Boneh D, Lewi K, Raykova M, et al. Semantically secure order-revealing encryption: Multi-input functional encryption without obfuscation[C]//Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2015:563 - 594.

[7] Popa R A, Li F H, Zeldovich N. An ideal-security protocol for order-preserving encoding[C]//IEEE Symposium on Security and Privacy(SP),2013:463 - 477.

[8] Kerschbaum F, Schröpfer A. Optimal average-complexity ideal-security order-preserving encryption[C]//2014 ACM SIGSAC Conference on Computer and Communications Security,2014:275 - 286.

[9] Wang X, Zhao Y. Order-revealing encryption: File-injection attack and forward security[C]//European Symposium on Research in Computer Security,2018:101 - 121.

[10] Ahmed S, Zaman A, Zhang Z, et al. Semi-order preserving encryption technique for numeric database[J]. International Journal of Networking and Computing, 2019, 9(1):111 - 129.

[11] Goldwasser S, Gordon S D, Goyal V, et al. Multi-input functional encryption[C]//Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2014:578 - 602.

[12] Chenette N, Lewi K, Weis S A, et al. Practical order-revealing encryption with limited leakage[C]//International Conference on Fast Software Encryption,2016: 474 - 493.

[13] Chang Y C, Mitzenmacher M. Privacy preserving keyword searches on remote encrypted data[C]//International Conference on Applied Cryptography and Network Security, 2005:442 - 455.

[14] Curtmola R, Garay J, Kamara S, et al. Searchable symmetric encryption: Improved definitions and efficient constructions[J]. Journal of Computer Security, 2011, 19(5): 895 - 934.

[15] Kerschbaum F. Frequency-hiding order-preserving encryption[C]//22nd ACM SIGSAC Conference on Computer and Communications Security,2015:656 - 667.

[16] Grubbs P, Sekniqi K, Bindschaedler V, et al. Leakage-abuse attacks against order-revealing encryption[C]//2017 IEEE Symposium on Security and Privacy (SP), 2017: 655 - 672.

[17] Maffei M, Reinert M, Schröder D. On the security of frequency-hiding order-preserving encryption[C]//International Conference on Cryptology and Network Security, 2017: 51 - 70.

[18] Cash D, Liu F H, O'Neill A, et al. Parameter-hiding order revealing encryption[C]//International Conference on the Theory and Application of Cryptology and Information Security,2018:181 - 210.

[19] Zhao Y. Identity-concealed authenticated encryption and key exchange[C]//2016 ACM SIGSAC Conference on Computer and Communications Security,2016:1464 - 1479.

[20] Wang H, Zhao Y. Identity-Based Higncryption[J]. IACR Cryptology ePrint Archive,2019,2019:106.

[21] Diffie W, Hellman M. New directions in cryptography[J]. IEEE Transactions on Information Theory, 1976, 22(6): 644 - 654.

[22] Joux A. A one round protocol for tripartite Diffie-Hellman[C]//International Algorithmic Number Theory Symposium, 2000:385 - 393.

[23] Boneh D, Franklin M. Identity-based encryption from the Weil pairing[C]//Annual International Cryptology Conference, 2001:213 - 229.

[24] Galbraith S D, Paterson K G, Smart N P. Pairings for cryptographers[J]. Discrete Applied Mathematics, 2008, 156(16):3113 - 3121.

[25] Freeman D, Scott M, Teske E. A taxonomy of pairing-friendly elliptic curves[J]. Journal of Cryptology, 2010,23(2):224 - 280.

(上接第 286 页)

[6] Gupta M, Patwa F, Sandhu R. Object-tagged RBAC model for the Hadoop ecosystem[C]//IFIP Annual Conference on Data and Applications Security and Privacy. Springer,2017: 63 - 81.

[7] 苏秋月,陈兴蜀,罗永刚. 大数据环境下多源异构数据的访问控制模型[J]. 网络与信息安全学报,2019,5(1): 78 - 86.

[8] 苏秋月. 大数据平台上基于属性的角色访问控制模型[J]. 现代计算机(专业版),2019(3):21 - 24.

[9] 王于丁,杨家海. 一种基于角色和属性的云计算数据访问控制模型[J]. 清华大学学报(自然科学版),2017,57(11):1150 - 1158.

[10] Gupta M, Patwa F, Sandhu R. An attribute-based access control model for secure big data processing in Hadoop ecosystem[C]//Proceedings of the Third ACM Workshop on Attribute-Based Access Control. ACM, 2018: 13 - 24.

[11] 陈垚坤,刘文丽. 一种适用于 Hadoop 平台的基于属性访问控制模型[J]. 河南师范大学学报(自然科学版), 2016,44(5):146 - 153.

[12] 刘敖迪,杜学绘,王娜,等. 基于区块链的大数据访问控制机制[J]. 软件学报,2019,30(9):2636 - 2654.

[13] Badger L, Sterne D F, Sherman D L, et al. A domain and type enforcement UNIX prototype[J]. Computing Systems, 1996, 9(1): 47 - 83.