分布式并行安全操作系统的用户一致性算法

王启科，刘心松，邱元杰，周 涛，黎 屹

(电子科技大学计算机科学与工程学院 成都 610054)

【摘要】在基于Linux内核的分布式并行安全操作系统中，由于没有全局一致的用户鉴别和访问权限管理机制，对系统的安全带来危害。该文在对用户权限验证机制进行分析的基础上，给出了由全局同步算法和用户信息一致性协议组成的实现用户一致性的算法。通过测试表明该算法可以有效地保证系统的用户一致性。

关 键 词 一致性协议; 分布式并行; 全局同步; 操作系统安全; 用户一致性

中图分类号 TP389.1 文献标识码 A

Users Consistency Algorithm of Distributed and Parallel Security Operating Systems

WANG Qi-ke，LIU Xin-song，QIU Yuan-jie，ZHOU Tao，LI Yi

(School of Computer Science and Engineering, Univ. of Electron. Sci. & Tech. of China Chengdu 610054)

Abstract In the Linux kernel-based Distributed and Parallel Security Operating System (DPSOS), the system security is not assured because of the lack of global user identification and access right managing mechanism. On the basis of analyzing the mechanism of user authority verification, this paper presents an algorithm to achieve the user consistency. This algorithm consists of global synchronizing algorithm and user information consistency protocol. Through tests, it has been proved that the algorithm can guarantee the user consistency of the system effectively.

Key words consistency protocol; distributed and parallel; global synchronizing; security of operating system; user consistency

随着网络技术的迅速发展和网络带宽的不断提高，信息安全问题日益显得突出，服务器的性能已经成为制约网络服务能力的瓶颈，分布式并行系统是解决此瓶颈的有效办法[1-2]。因此如何最大限度地实现资源共享、全方位地进行资源保护，是开发分布式并行操作系统中必须认真考虑的问题。文献[3]提出了基于角色的访问控制(Role-Based Access Control，RBAC)模型，该模型是策略中性的，为相关的研发提供了公共的框架。文献[4]提出了将SELinux作为Linux安全模块(Linux Security Module，LSM)的实现方法，该方法将安全域加入到内核数据结构中，并且在内核代码的关键点插入钩子函数的调用，用来管理安全域并实现访问控制。文献[5]提出了分布式系统中面向对象的RBAC模型，并提出了防止域安全管理器将互斥的角色分发给用户的新方法，实现了多安全域的访问控制。但这些模型和实现方法均未涉及分布式并行系统中维护用户一致性的实现方法及全局一致的用户鉴别和访问权限管理机制。基于此，本文提出了用户一致性算法，该算法可以保证在各种情况下分布式并行系统中用户信息的一致性。

1 问题简述

系统是基于Linux内核研发的分布式并行安全操作系统，它使用网络文件系统(Network File System，NFS)来实现资源的共享和多副本存储，通过智能的副本管理、动态可变节点数以及自动故障恢复等方法来保证分布式并行系统的高可靠性和可用性。在安全方面，采用了RBAC模型，并且实现了全局策略一致的RBAC。

当用户合法访问系统中其他节点通过NFS输出的属于该用户的共享资源时，访问操作可能被拒绝。发生这种现象的原因是由于在Linux中对用户的访问权限是通过用户识别符(User Identifier，UID)和文

收稿日期：2005－02－23
作者简介：王启科(1981－)，男，硕士，主要从事分布式并行操作系统及操作系统安全方面的研究.

件所有者UID来进行验证的。因此当同一个用户名在各节点上有不同的UID时，就会造成该用户对相同的NFS共享资源在不同的节点上的访问权限不一致[6]。为解决这一安全隐患并保证用户访问资源的合法性，需要对分布式并行安全操作系统中所有节点的用户信息提供一致性保证。

2 算法描述

2.1 全局同步算法

对各个节点上的用户名和对应的用户UID信息采用基于版本号的同步方法来进行同步。通过执行同步算法使系统中所有活动节点都具有版本号相同的用户信息，而且该版本号必须是最高版本号。当某节点启动后，即发起全局同步操作，将发起全局同步操作的节点称为同步发起节点。全局同步的具体算法为：

1) 同步发起节点进程：

①Deliver(节点启动消息，所有活动节点)；
②Getmsg(节点启动的响应消息)；
GetMaxVersion(节点启动的响应消息集)；
③Deliver(需要更新消息，非最高版本活动节点)；
④Getmsg(需要更新的响应消息)；
⑤Deliver(请求用户信息的消息，有最高版本的活动节点)；
⑥Getmsg(请求用户信息的响应消息)；
Update(最高版本的用户信息)；

2) 活动节点监听进程：

While(节点活动){Getmsg(Msg)；
Switch(Msg){
Case：①节点启动消息
②Deliver(节点启动的响应消息，同步发起节点)；break；
Case：③需要更新消息
④Deliver(需要更新的响应消息，同步发起节点)；⑤Deliver(请求用户信息的消息，有最高版本的活动节点)；⑥Getmsg(请求用户信息的响应消息)；Update(最高版本的用户信息)；break；
Case：⑤请求用户信息的消息
⑥Deliver(请求用户信息的响应消息，请求用户信息的节点)；break；
… …
}
}

其中，Deliver(Msg, Dest)为将消息Msg发送到节点组Dest；Getmsg(Msg)为接收到来的消息Msg；GetMaxVersion(MsgSet)为从获得的消息中找出用户信息的最高版本号及其节点信息；Update(UserInfo)为通过用户信息UserInfo对本节点的用户信息更新。

算法的典型流程如图1所示。图中，同步发起节点也可能是最高版本活动节点，而且同步发起节点会并发执行同步发起进程和活动节点监听进程。

在执行完一轮同步算法后可以保证系统中所有活动节点都具有版本号相同的用户信息，而且该版本号是最高版本号。

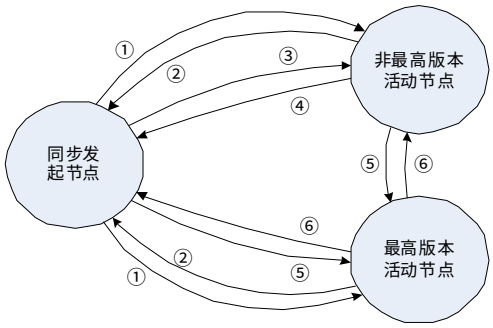

图1 全局同步算法消息流

2.2 用户信息一致性协议

在系统中任意节点上执行了能改变用户信息的操作后，要使所有活动节点的用户信息仍然保持一致。为了达到此目的，本文提出了用户信息一致性协议。该协议通过两阶段提交的方式，保证所有活动节点都会执行相同的用户操作，而且执行操作之后会有相同的执行结果。

两阶段提交方式的用户操作的处理流程是：

1) 用户操作的发起节点的执行进程：

if(Local_validate(用户操作)){
⑦Deliver(询问用户操作的消息，所有活动节点)；
⑧Getmsg(用户操作通过验证的消息)；
if(获得所有用户操作通过验证的消息){
⑨Deliver(允许用户操作的消息，所有活动节点)；
Execute(用户操作)；
IncVersion；
}
else{
⑨Deliver(取消用户操作的消息，所有活动节点)；
}
}
Execute(全局同步)；

2) 活动节点监听进程：

```
While(节点活动){ Getmsg(Msg)；
Switch(Msg){
… …
Case：⑦询问用户操作的消息
    if(Local_validate(用户操作)){
        ⑧Deliver(用户操作通过验证的消息，用户操作发起节点)；
        if(⑨Getmsg(允许用户操作的消息))
            {Execute(用户操作)；IncVersion；}
        }
    Break；
    }
}
```

其中，Local_validate(Action)为本地验证操作Action的可执行性；Execute(Action)为执行Action操作；IncVersion为本节点版本号加1。

在用户操作的处理流程结束前要进行一次全局同步，这是为了消除用户操作执行之前由于节点故障或网络故障引起的各节点间的用户信息不一致。用户操作消息流如图2所示。

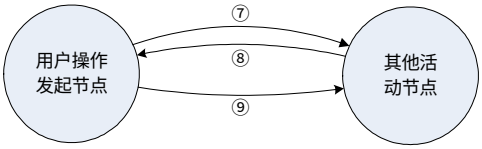

图2　用户操作消息流

2.3　关于安全方面的处理

用户一致性涉及用户信息的全局同步操作和对用户信息的更改操作，攻击者可以通过伪造的报文来获取系统中节点的用户信息，甚至可以更改节点的用户信息，从而获得节点甚至是整个系统的控制权限[7]。因此对全局同步流程和用户操作处理流程需要有很高的安全性要求。在具体实现时，主要采取三方面的措施来提高用户一致性模块的安全性。

(1) 对报文的源地址进行验证：攻击者可以从分布式并行安全操作系统外发送伪造的报文到系统内的节点，以试图获得节点的用户信息或更改节点的用户信息。因此，当节点收到任何发往用户一致性通信端口上的报文时，首先要对该报文的源地址进行验证，若报文的源地址不是分布式并行安全操作系统中某个节点的IP地址，则认定该报文是非法报文并将其丢弃。

(2) 对报文的内容进行散列：尽管对报文的源地址进行了验证，攻击者仍然可以通过将报文伪装成来自系统内的节点，以达到窃取用户信息或更改用户信息的目的。为了防止遭到伪装报文的攻击，需要对报文的内容进行散列处理。在用户一致性模块中使用的所有报文中都有一个散列值，该值是用报文中其他部分的内容通过散列函数得到的。对收到的报文的散列值要进行验证，若散列值不正确，则认定该报文是非法报文并将其丢弃。

(3) 对用户信息进行同步时，不同步用户密码：出于对整个系统的安全考虑，在对用户信息进行同步时，不同步用户的密码。假设在节点 S_i 上用户 U_j 的密码为 P_{ij}，该密码被窃取甚至被窜改了，攻击者获得了用户 U_j 在节点 S_i 上的所有权限。但是不能用 P_{ij} 获得系统其他节点上用户 U_j 的权限，从而最大限度地限制了对系统安全造成的损害。

3　测试结果

3.1　指标定义

将用户一致性模块应用于以Linux内核为平台研发的分布式并行安全操作系统中，其测试环境为：(1) 硬件环境：20台P4 2G，512 M内存，100 M网卡的普通PC作为分布式并行安全操作系统服务器，采用100 M交换机。(2) 软件环境：基于Linux内核的分布式并行安全操作系统采用版本为2.4.20的内核，编译器是gcc 3.2。测试结果如表1、2所示。

表1　算法正确性和可用性的测试结果

单节点操作	多节点同时操作	全局同步处理
操作成功，且能保持一致性	能保持一致性	同步成功，且能保持一致性

表2　算法性能测试结果

	单机操作耗时/s	20个节点操作耗时/s	性能下降率/(%)
添加1 000个用户	66.08	137.12	107.5
删除1 000个用户	38.94	101.09	159.6

由于在多节点的系统中执行保证一致性的用户操作时，算法的处理过程和通信时延对执行的效率有负面的影响，会造成性能下降。而本文测试结果表明执行时间在可以接受的范围之内。

(下转第270页)

条件属性 x_2 即道路几何线形的影响强度次之。车辆速度是否合理是导致该路段事故发生的最大诱因，而道路的几何线形是其次的诱因。针对这两个条件，交通管理部门就可以制定出相应的整治措施，如在适当的路段对经过车辆的最低速度和最高速度作出限制，在弯道处设置警示牌，安置凸面镜以方便司机观察弯道对面来车情况等。

4 结 束 语

本文通过运用粗集来对形成交通事故黑点的诸多因素进行筛选，能准确合理地分析出产生交通黑点的主要因素和一般因素，从而能针对其主要因素进行整治，节省了治理交通黑点的时间和费用，有效地避免了人力物力的浪费。

参 考 文 献

[1] PAWLAK Z. Rough sets[J]. Communication of ACM, 1995, 38(11): 89-95.
[2] 陈世清, 唐志航, 肖建华. 基于粗糙集联系度的数据挖掘算法及应用研究[J]. 计算机应用, 2004, 24(6): 74-75.
[3] 王国胤. Rough集理论与知识获取[M]. 西安: 西安交通大学出版社, 2001.
[4] 曾黄麟. 粗集理论及其应用[M]. 重庆: 重庆大学出版社, 1998.
[5] 王德松, 舒 兰. 粗集决策表与决策表简化的可信度比较[J]. 电子科技大学学报, 2004, 33(5): 611-612.
[6] 曾黄麟 智能计算–关于粗集理论、模糊逻辑、神经网络的理论及其应用[M]. 重庆: 重庆大学出版社, 2004.

编 辑 漆 蓉

(上接第241页)

4 结 论

出于对系统的安全性和用户使用的方便性考虑，需要保证分布式并行安全操作系统的各个节点的用户信息的一致性。为了达到这个目的，本文提出了一种解决方案，并且予以实现。实验证明：该方案能够有效地对分布式并行安全操作系统的各个节点的用户信息进行全局同步，并且能保证在任意节点上执行用户操作后用户信息及在遇到节点故障或网络故障之后恢复各个节点用户信息的一致性。

参 考 文 献

[1] GALLI D L. 分布式操作系统: 原理与实践[M]. 徐良贤, 译. 北京: 机械工业出版社, 2003.
[2] 刘详岐, 卢显良. 分布式集成防御系统中的任务迁移[J]. 实验科学与技术, 2006, 4(3): 28-30.
[3] SANDBU R S, COYNE E J, FEINSTEIN H L, et al. Role-based access control models[J]. IEEE Computer, 1996, 29(2): 38-47.
[4] SMALLEY S, VANCE C, SALAMON W. Implementing SELinux as a Linux security module[EB/OL]. http://www.nsa.gov/selinux/papers/module.pdf, 2004-11-05.
[5] ZHANG C N, YANG Cun-gang. An object-oriented RBAC model for distributed System[C]// Proceedings of the Working IEEE/IFIP Conference on Software Architecture (WICSA'01). Amsterdam Netherlands: IEEE Computer Society Washington, 2001, 08: 24-32.
[6] ZADOK E. Linux网络文件系统管理指南[M]. 邱仲潘, 译. 北京: 电子工业出版社, 2001.
[7] SCHNEIER B. 网络信息安全的真像[M]. 吴世忠, 译. 北京: 机械工业出版社, 2001.

编 辑 漆 蓉