基于SPIN的CSCW系统的验证

单卓为[1],鱼 滨[2]
(1.西北大学 信息学院 计算机系,陕西 西安 710127;
2.西安电子科技大学 计算机学院,陕西 西安 710071)

摘 要:近年来,CSCW系统呈现出用户越来越多、权限关系越来越复杂、组织结构规模越来越大、处理的情况越来越复杂的趋势。因此,CSCW系统的访问控制策略以及访问控制策略的验证已成为国内外CSCW领域十分值得研究和探讨的问题。针对CSCW系统设计的特点,提出了一种验证策略。结合具体实例,使用RBAC模型描述CSCW系统的访问控制权限,利用SPIN工具将模型检测应用于验证CSCW系统属性。

关键词:CSCW;角色访问控制;时序逻辑;模型检测;SPIN
中图分类号:TP311 **文献标识码**:A **文章编号**:1673-629X(2008)04-0009-04

Using SPIN to Validate CSCW System

SHAN Zhuo-wei[1], YU Bin[2]
(1. Department of Computer Science, School of Information, Northwest University, Xi'an 710127, China;
2. School of Computer Science and Engineering, Xidian University, Xi'an 710071, China)

Abstract: Recently, CSCW system has shown a development tendency which has more users, more complex permission connection, larger configuration dimension, and more complicated cases to deal with. Consequently, the accessing control policy of CSCW system and the validation of it have become quite study-worthy topics in and out of China. Focusing on the design traits of CSCW system, has proposed one validation tactic of CSCW system. Researchers illustrate this validation tactic with examples, use RBAC model to depict the accessing control permission of CSCW system, and use the SPIN tool to apply model check to validate the attribute of CSCW system.

Key words: CSCW; RBAC; temporary logic; model checking; SPIN

0 引 言

近年来,信息系统、计算机网络应用、多媒体通信等领域都非常关注计算机支持的协同工作。CSCW(Computer Supported Cooperative Work,计算机协同工作)是Gfief和Cashman在1984年提出的,用于描述在计算机技术支持的环境下,一个群体如何协同完成一项共同任务的新兴技术。经过多年的发展,CSCW现已成为一个新的多学科领域[1]。

随着分布式计算环境的进一步发展,CSCW系统呈现出用户越来越多、权限关系越来越复杂、组织结构规模越来越大、处理的情况越来越复杂的趋势。传统的访问控制模型如DAC(Discretionary Access Control)和强制访问控制MAC(Mandatory Access Control)等,已不适合描述CSCW系统的安全策略。因此,20世纪90年代中期,美国人Ravi Sandhu提出了RBAC(Role-Based Access Control,角色访问控制)模型,该模型有效地克服了传统访问控制技术的不足,满足了CSCW系统的需求,有效地描述系统安全策略的约束条件,减少了授权管理的复杂性,为管理员提供了一个易于实现的安全环境。

通过引入RBAC模型,CSCW系统可以根据需要定义各种角色,并设置合适的访问权限,而用户根据职责和任务的需要再被指派为不同的角色。这样整个访问控制过程分为两部分,即访问权限与角色相关联,角色再与用户相关联,从而实现用户与访问权限的逻辑分离。

在RBAC模型中,根据基本的系统规则,基于角色的安全性和一致性的约束可以通过逻辑表达式予以充分表达,可以引入模型检测的方法验证系统的正确性。根据这个思路,可以在CSCW系统的设计阶段验证CSCW系统的安全性需求,即保证指定的约束不违反

收稿日期:2007-07-14
基金项目:国家自然科学基金重点项目(60433010)
作者简介:单卓为(1980-),男,河南周口人,硕士研究生,主要研究方向为软件工程、形式化验证;鱼 滨,副教授,硕士生导师,主要研究方向为分布式应用、中间件技术、时态逻辑。

任何系统所希望的一致性和安全性,如:用户的交互能够相互协调,同时遵循任务流需求;角色没有冲突和不一致的约束;机密信息不能被未授权的用户访问;在访问对象时,任何临时的或者有条件的约束都能被满足;没有权限泄露给未授权的用户。

通过使用模型检测的方法和工具,能够保证系统设计的正确性,为进一步的系统开发打下坚实基础。

1 模型检测和SPIN工具

模型检测是关于自动验证并行或分布式系统性质的方法,它的基本思想是采用状态空间搜索的方法来检测一个给定的计算模型是否满足某个时序逻辑公式所表示的特定性质[2]。其检查过程如图1所示。

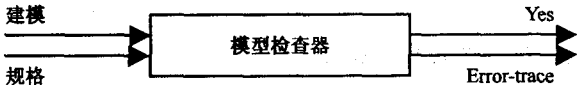

图1 模型检测过程

图中,建模(Modeling):描述系统的可能行为,用模型检查工具可以接受的形式表示;

规格(Specification):声明系统期望达到的性质,通常用时序逻辑表示;

验证(Verification):由模型检查器自动完成。如果正确,输出yes;如果错误,输出一个反例。

模型检测主要是面向某类性质来检测系统是否合乎规约,在系统不满足所要求的性质时,模型检测算法会产生一个反例(一般是一条执行路径)来说明不满足的原因。

SPIN是由贝尔实验室开发的,基于以上的思想和方法,用于并发系统自动验证工具。SPIN通过Promela(Protocol Meta-Language)语言进行建模,用线性时序逻辑公式描述规格,进行自动验证,如不满足系统所要求的规格,将产生一个反例来说明不满足的原因。如今SPIN被广泛地应用于工业界和学术界。其特点如下[3]:

(1)SPIN以Promela为输入语言,可以对软件设计中的规格的逻辑一致性进行检验,并报告系统中出现的死锁、无效的循环、未定义的接收和标记不完全等情况。

(2)SPIN使用on-the-fly技术,即无需构建一个全局的状态图或者Kripke结构,可以根据需要生成系统自动机的部分状态。

(3)SPIN可作为一个完整的LTL(Linear Temporal Logic)模型检验系统来使用,支持所有的可用的线性时态逻辑表示的正确性验证要求,也可以在有效的on-the-fly检验系统中用来检验协议的安全特征。

(4)SPIN可使用会面点来进行同步通信,也可以使用缓冲通道来进行异步通信。

(5)对于给定的一个使用Promela描述的协议系统,SPIN可以对其执行随意的模拟,也可以生成一个C代码程序,然后对该系统的正确性进行有效的检验。

(6)在进行检验时,对于中小规模的模型,可以采用穷举状态空间分析,而对于较大规模的系统,则采用Bit State Hashing方法来有选择地搜索部分状态空间。

2 CSCW系统模型检测的策略

对于CSCW系统,首先利用RBAC描述系统的静态需求,其要素包括活动、权限、角色、操作和数据[4],关系如图2所示。

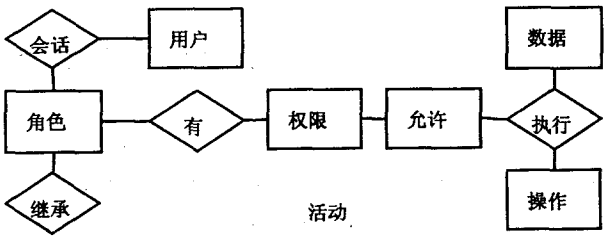

图2 RBAC各元素之间的关系

在RBAC模型中,一个活动(Activity)被定义为针对一系列的共享对象,多个用户(user)通过完成各自的任务而共同完成某个特定的目标。活动可以描述一个受保护的领域以及角色、对象和权限在协作中的作用域。活动中的角色被赋予一定权限执行某种特定的操作(Operation)。操作根据其前置条件执行相应的动作(Action)。这些动作包括对象方法的调用、同步、活动的管理,以及活动和新对象的实例化。活动的模板指定了协作的方式。会话描述用户和角色之间的关系。用户每次必须通过建立会话获得许可(admission)来激活角色,得到相应的访问权限。

针对RBAC的特点,对于其需求的描述可以抽象设计一个语法级的简单描述模型,然后根据这个语法级的模型,转换成Promela的描述的系统模型;对于系统角色的安全性和一致性约束使用线性时态逻辑表达式来描述其性质;然后利用SPIN工具进行模拟和相关性质的验证。如图3所示。

3 具体应用

图4展示了一个课程活动的层次结构,通过课程活动的实例[5,6]来说明如何使用SPIN验证CSCW系统的安全属性。

根据图4的描述,首先使用RBAC描述该CSCW系统模型的相关静态需求:活动、权限、操作和角色。

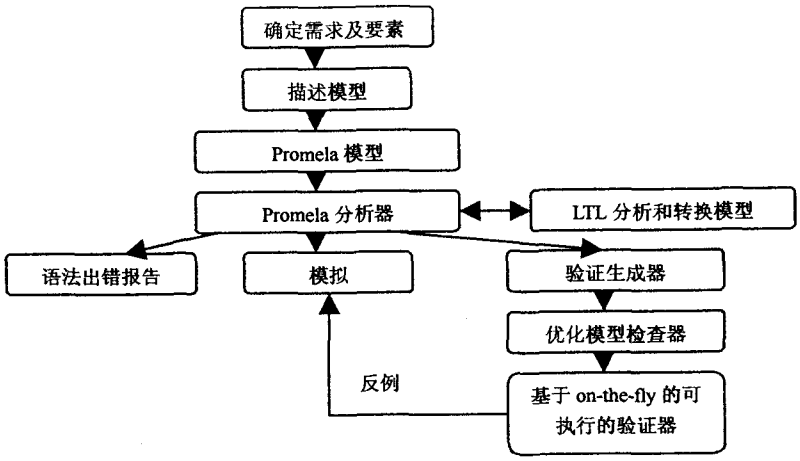

图 3　RBAC 的 CSCW 模型检测的基本过程

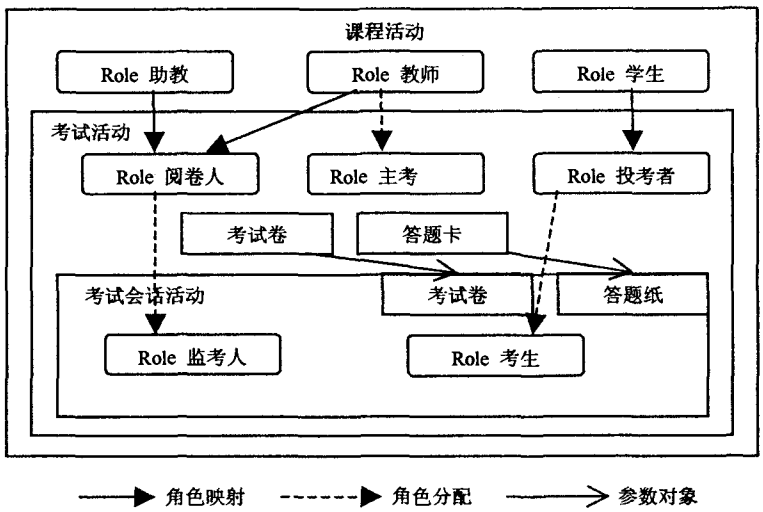

图 4　课程活动的层级结构

课程活动的中包含了三个角色:教师(Instructor),助教(Assistant)和学生(Student)。在课程活动中嵌入了一个考试的活动。在考试活动中,教师可以被许可分配为主考老师(examiner),教师和助教可以被许可分配为阅卷人(Grader)的角色,学生被许可分配为投考者(Examinee)的角色,每个投考者可以创建一个考试会话的实例参与考试。在考试会话活动中,包含了监考人(Checker)和考生(Candidate),其中投考者被许可分配为考生的角色,阅卷人被许可分配为监考人的角色;同时,引入两个考试卷(ExamPaper)和答题纸(AnswerBook)作为考试会话活动的两个参数,其中考试卷仅仅作为一个共享对象,答题纸能够被每次考试会话所创建。

权限是指所有者(Owner)角色指定哪个用户能够管理活动以及活动的内部实体,具体关系如下:

(1)活动的模板指定了哪个角色拥有哪些实体;

(2)对于一个活动,它的所有者是上一级活动(parent activity)的所有者;对于一个角色,它的所有者默认是活动所有者;对于一个对象,它的所有者是创建该对象的角色。

用来动态地指定系统模型的一致性和安全性的事件需求,被系统模型隐式地产生。在每个角色相关的操作(Operation)中必须包括三个事件:请求(request),开始(start)和完成(finish)。对于特定事件的发生的计数,用#来表示。

角色的需求包括了角色名、已分配的角色、角色的许可、活动的约束和角色的带有前置条件的操作。

角色许可约束:

```
AdmissionConstraints
    member(thisUser, parentActivity.Examinee)
    & member(thisActivity.Creator, thisUser)
    & #members(thisRole)<1
```

角色操作的约束:

```
Operation OpenExam{
    Precondition #(OpenExam.start)=0
    Action exam.readPaper()}
```

活动的约束:

```
ActivationConstraints
    date > DATE(May, 10, 2007, 9:00)
    & date < DATE(May, 10, 2007, 11:00)
```

在这里,用函数 member(role, user)检查在一个角色中是否有一个参与者(participant)出现,members(role)回送一个参与者的列表,#(members(role))表示一个角色中参与者的数量。

根据以上相关描述,使用伪代码创建一个如图 5 所示的考试活动的基本结构模板。

```
ActivityTemplate Course (AssignedRoles Assistant, Instructor, Student) {
    ......
    ActivityTemplate Examination (Owner Instructor, AssignedRoles Examiner){
        ObjectType ExamPaper ......
        ObjectType AnswerBook ......
        Role Examiner {...}
        Role Examinee (Reflect parentActivity.Student) {...}
        Role Grader (Reflect parentActivity.Assistant, parentActivity.Instructor) {...}
        ActivityTemplate ExamSession(Owner Grader,Objects (ExamPaper exam,AnswerBook ans),AssignedRoles Candidate) {
            TerminationCondition #(Checker.Grade.finish)>0
            Role Candidate {
            AdmissionConstraints
```

```
    member(thisUser, parentActivity.Examinee)
    & member(thisActivity.Creator, thisUser)
    & #members(thisRole)<1
    ActivationConstraints
    date > DATE(May, 10, 2007, 9:00)
    & date < DATE(May, 10, 2007, 11:00)
    Operation OpenExam{
    Precondition #(OpenExam.start)=0
    Action exam.readPaper()}
    Operation Write {
    Precondition #(OpenExam.finish)>0
    Action ans.writeAnswer(data) }
    Operation Submit { Precondition Write.finish>0 }
    }
    Role Checker {
        AdmissionConstraints #members(thisRole)<1
    & member(thisUser, parentActivity.Grader)
    Operation Grade {
    Precondition #(Candidate.Submit.finish)=1
    Action ans.setGrade(data)}
   }
  }
 }
}
```

图 5 考试活动的基本结构模板

对于基于角色访问控制的 CSCW 系统的安全属性的验证包括对操作的可达性、任务流、角色约束、信息流和机密性、完整性和访问泄露等的验证。为了介绍模型检测的验证方法，选择对任务流作为一个实例进行验证。简便起见，仅仅针对考试会话模型，使用 Promela 语言进行建模。

根据系统行为需求的描述可以建立一个考试会话的任务流验证模型，用来验证系统期望的性质。

```
typedef Event{
    bit finish;
    bit start;
    bit request
};
Event Write,OpenExam,Submit,Grade;
bit ExamSession_finish=0;
active proctype ExamSession( )
{ Write.finish = 0; OpenExam.start = 0;
    OpenExam.finish = 0; Submit.finish
= 0;
    Grade.finish =0;
    do
    ::Grade.finish = 0 ->
    if
    /* OpenExam */
    :: atomic { OpenExam.start = 0
    -> OpenExam.start = 1; } OpenExam.finish = 1;
    /* Write */
    :: OpenExam.finish ! = 0 -> Write.finish = 1;
    /* Submit */
    :: Write.finish ! = 0 -> Submit.finish = 1;
    /* Grade */
    ::Submit.finish = 1 -> Grade.finish = 1;
    fi
    :: Grade.finish ! = 0 ->
    ExamSession_finish++; break;
    od
}
```

考试会话活动的任务流主要描述如何完成一个考试活动，可以通过路径表达。该模型的任务流期望达到的性质之一是：对于考试会话，在考生角色完成开卷(Open Exam)的操作之后，在指定的时间内写操作可以被执行一次或者多次。

转换为时态逻辑公式表达如下：

性质 1：□(Candidate.OpenExam.finish -> □◇Candidate.Write.finish)

这里□和◇都是时序逻辑的时序算子，分别表示总是(always)和有时(sometimes)。

把上述公式转换为 Promela 语言 never claim[7]，利用 SPIN 工具进行验证上述属性，可以得到结果如图 6

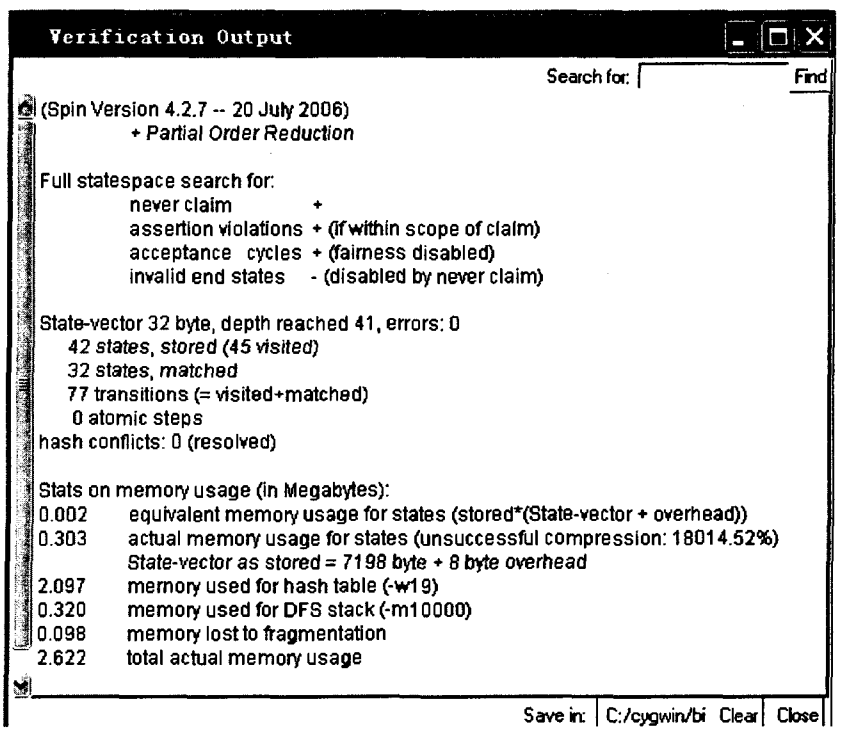

图 6 SPIN 的检测结果

(下转第 15 页)

2.2 不同噪音环境下语速适应性测试

随机选择某英语课文两段语音数据，这两段语音分别由男女声朗读，环境噪音不同，语速也明显不同，以测试聚类分析针对不同语音材料的适应性，实验结果如表 2 所示。

表 2　句子端点门限值

语音素材	A	B
检测门限值	800ms	1100ms

实验结果显示，文中的方法可以对不同语速的语音，进行相应的句子端点门限检测，具有一定的自适应性，如图 2 所示，素材 A 的门限为 800ms，其中的间隙 a 为单词间隙，b 为句子间隙；如图 3 所示素材 B 的门限为 1100ms，其中的 a 为单词间隙，b 为句子间隙。

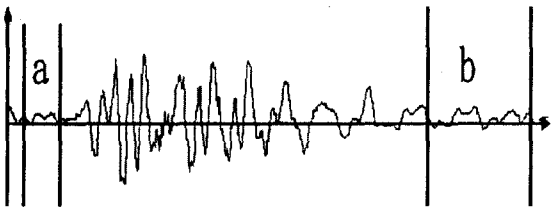

图 2　语音素材 A 的门限值

3 结束语

此检测法相比常规的语音端点检测方法，最显著的是简化了所要考虑的特征值，不再使用过零率特征值（过零数门限）等常规特征值。能量门限也只采用了单能量门限。并且针对语音复读系统等背景噪声相对较小且稳定的实际应用环境，在实现过程上进行了简化和改进，从而在语音端点检测的实现过程上更加简单方便，实验表明能针对不同语速的语音材料较准确地判断出特定的句子语音门限。对于语音复读机软件等具有较低背景噪音的环境中的句子语音端点检测是一种简单有效的检测方法。

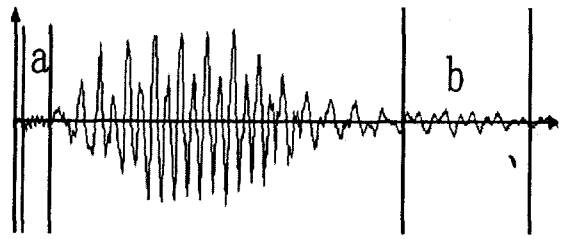

图 3　语音素材 B 的门限值

参考文献：

[1] 古井贞熙. 数字声音处理[M]. 朱家新，张国海，易武秀，译. 北京：人民邮电出版社，1993.

[2] 胡广书. 数字信号处理[M]. 北京：清华大学出版社，2003.

[3] Rabiner L, Juang Biing - Hwang. Fundamentals of Speech Recognition[M]. [s.l.]: PTR Prentice - Hall, Inc, 1993.

[4] 吴亚栋. 语音识别基础[D]. 上海：上海交通大学，1999.

[5] 李祖鹏，姚佩阳. 一种语音段起止端点检测方法[J]. 电讯技术，2000(3): 68 - 70.

[6] 张新宇. Windows 声音应用程序开发指南[M]. 西安：西安电子科技大学出版社，2003.

（上接第 12 页）

所示。从结果可以看出：在验证时，使用了偏序规约的方法，搜索了全部的空间状态，并使用了 never claim，搜索深度最大为 41，共搜索了 42 个状态，77 个转换，其中 32 重复检测，发现错误为 0 个。从而验证了考试会话模型满足期望的性质 1。

4 结　论

引入了验证 CSCW 系统静态需求的一种策略。根据 CSCW 系统的特点，使用角色访问控制描述，利用模型检测的工具 SPIN，结合具体的实例，验证了 CSCW 系统中任务流模型的具体性质。模型检测具有高度自动化，覆盖全部状态和能够生成反例的特点，把这种方法应用于验证软件设计的正确性将有更加广阔的前途。

参考文献：

[1] 郑庆华. CSCW 建模与实现方法[J]. 计算机学报，1998，21(8): 270 - 276.

[2] Joost - Pieter K. Concepts Algorithms and Tools for Model Checking[M]. [s.l.]: [s.n.], 1999.

[3] 古天龙，蔡国永. 网络协议的形式化分析与设计[M]. 北京：电子工业出版社，2003.

[4] 李成锴. 基于角色的 CSCW 系统访问控制模型[J]. 软件学报，2000，11(7): 931 - 937.

[5] Tripathi A, Ahmed T, Kumar R. Specification of Secure Distributed Collaboration Systems[C]// In IEEE International Symposium on Autonomous Distributed Systems (ISADS). Pisa, Italy: IEEE Computer Society, 2003.

[6] Tripathi A, Ahmed T, Kumar R, et al. Design of a Policy - Driven Middleware for Secure Distributed Collaboration[C]// In Proc. of International Conference on Distributed Computing Systems 2002. Vienna, Austria: IEEE Computer Society, 2002: 393 - 400.

[7] Holzmann G J. The SPIN Model Checker[M]. [s.l.] Addison - Wesley, 2003.