采用模型检测器的软件安全模型验证方法

陈峰[1,2]，李伟华[1]，陈昊[2]，吕正[2]
(1. 西北工业大学计算机学院，710072，西安；2. 西北大学信息科学与技术学院，710069，西安)

摘要：针对软件开发过程中安全性分析与设计不足的问题，在研究现有软件安全性建模及形式化验证技术的基础上，提出了一种适用于面向对象的软件安全性建模与验证方法. 建立软件安全属性的非形式化UML模型，采用安全扩展有限自动机创建其形式化模型，并使用线性时序逻辑描述安全属性，将形式化模型与安全属性共同作为模型检测器的输入，得到模型是否满足性质的验证结果，从而实现了软件安全设计与验证技术的有机结合. 实验结果表明，该方法能够在软件设计初期对所涉及的安全性进行有效分析与验证.

关键词：软件安全分析；确定有限自动机；形式化建模；模型检测
中图分类号：TP311 **文献标志码**：A **文章编号**：0253-987X(2011)02-0015-06

A Verification Method of Software Safety Model Based on Model Checker

CHEN Feng[1,2], LI Weihua[1], CHEN Hao[2], LÜ Zheng[2]
(1. School of Computer Science and Technology, Northwestern Polytechnical University, Xi′an 710072, China;
2. School of Information Science and Technology, Northwest University, Xi′an 710069, China)

Abstract: In order to improve the analysis and design of software safety, on the basis of current researches on safety model and verification technology, a modeling and checking method is proposed for object-oriented software safety. Establishing non-formal UML models of software safety, safety extended deterministic finite automata and linear temporal logic are used to create the formal models and describe the safety properties separately. Through a model checker, we get the verification result. This approach realizes the combination of software safety model and verification technology. Experimental results show that the method can analyze and verify the safety properties effectively in the early stage of software design.

Keywords: software safety analysis; deterministic finite automata; formal modeling; model checking

软件安全性是指使软件所控制的系统始终处于不危及人的生命财产和生态环境的安全状态的性质. 随着计算机应用范围的迅速扩大，它将越来越深入人们的日常生活. 计算机及其软件虽然提高了人们的生活与工作条件，但也直接地关系到人们的生命、财产与人类生存环境的安危，因此安全性是计算机在诸多领域应用中对软件提出的一个新的要求.

1 相关工作

软件安全性分析的实施，极大地提高了软件质量与安全性，同时带来了相应的社会效益与经济效益. 国内外研究人员对现有的软件安全分析与建模方法进行了深入研究，提出了相应的策略，文献[1]给出了一种通过创建安全对象以及故障树分析评估

收稿日期：2010-09-02. 作者简介：陈峰(1978—)，男，博士生；李伟华(联系人)，男，教授，博士生导师. 基金项目：国家自然科学基金资助项目(60803150，60803151)；国家自然科学基金委员会-广东联合基金重点资助项目(U0835004)；国家高技术研究发展计划资助项目(2008AA01Z411)；陕西省自然科学基金资助项目(2009jm8012)；陕西省教育厅自然科学专项资助项目(09JK736)；陕西省科技攻关资助项目(2009k08-04).

进行安全分析的软件开发方法;文献[2]介绍了软件系统安全需求的跟踪;文献[3]在软件质量模型基础上给出了软件安全度量与管理的标准. 上述方法和模型与传统的面向对象软件设计存在一定的差异,不便于将安全分析与软件设计进行有机结合,并且涵盖安全性需求的通用软件以及安全性至关重要的软件系统,应使用更为严格的过程管理及形式化的设计与开发方法对软件安全性进行验证.

在面向对象软件设计中,UML 作为一种非形式化建模语言具有广泛应用,其语法结构虽采用了形式化规约,但其语义部分使用自然语言描述,缺乏精确的表述,因而无法采用形式化验证方法来确保使用 UML 建立的模型能够正确刻画系统性质. 为了解决此类问题,本文拟采用模型检测的方法验证系统模型的正确性. 具体做法如下:结合 UML 和形式化方法的优点,将非形式化的 UML 图形转换为具有精确语义定义的形式化规范,在非形式化的图形表示与形式化定义之间建立映射关系,得到软件安全分析的形式化模型,并将其与软件性质的逻辑描述共同作为模型检测器的输入,以验证系统模型是否满足性质要求,从而得到正确的表达模型. 在使用 UML 安全扩展[4]建立软件安全相关模型的过程中,序列图所表达的安全交互是软件安全分析与设计的核心,并且在 UML2.0 序列图所添加的新特征中,neg 组合片段能够描述交互过程中无效的否定迹,用以捕获软件安全属性. 为了保证采用序列图建立的安全相关模型的正确性,可以使用不同的形式化方法对其进行描述与验证,从而实现在软件安全设计正确的基础上对模型的应用.

2 集成安全分析的软件建模

软件安全性建模及其安全属性刻画的正确性验证在安全相关软件的分析与设计中至关重要,而形式化验证技术能够很好地解决模型是否正确表达系统安全性的问题,因此作为本文研究的重点.

2.1 软件安全性建模与验证方法

为了获得能够正确表达软件安全属性设计的系统模型,本文提出了一种软件安全性建模与验证方法,将安全属性添加到系统分析与设计过程中,其步骤如下,基本流程如图 1 所示.

步骤 1 从使用自然语言描述的软件需求中,衍生出使用 UML 刻画的软件安全分析的非形式化模型,用以指导软件的设计与开发活动.

步骤 2 将非形式化模型映射为安全分析的形式化模型,本文采用自动机作为系统的形式化模型.

步骤 3 使用 Promela 语言描述形式化模型.

步骤 4 使用逻辑语言描述软件需求的安全属性.

步骤 5 将步骤 3 与步骤 4 的结果作为检测器的输入条件,以判断所建立的模型是否准确地表达了系统要求的安全属性.

步骤 6 若输出结果中产生反例,说明模型不满足性质,需要对模型进行检查,迭代此过程直至无反例产生,即可得到满足性质描述的系统模型,此模型可作为系统开发的依据.

本文主要研究在软件安全分析的非形式化模型基础上,如何建立其形式化有穷状态自动机模型,并使用模型检测器对安全属性的形式化描述进行验证,为进一步使用软件安全模型奠定基础.

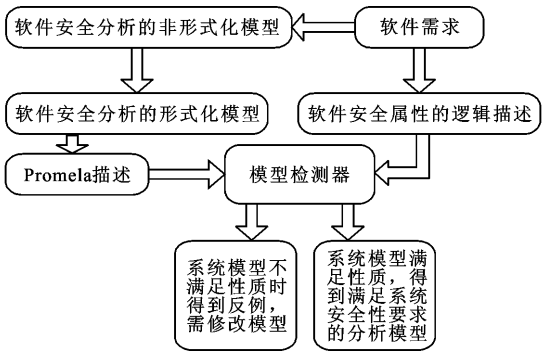

图 1 软件安全性建模及验证过程

2.2 软件安全分析的非形式化模型

在基于 UML 面向对象软件建模与设计中,安全属性作为系统需求的一部分,根据不同应用领域的不同侧重点,通过向活动图、序列图、状态图及部署图添加相应的 profile 文件进行扩展,实现其安全分析策略.

2.2.1 *基于安全分析的 UML 扩展* UML 适用于各种软件开发方法、软件生命周期的各个阶段,可满足大多数情况下对系统的建模需要,但在对软件安全进行建模时,一些构件不能与标准的 UML 建模元素一一对应,因此须对 UML 进行扩展,其 3 种核心扩展机制包括构造型、标记值和约束. UML 安全扩展"UMLsec"是基于 UML 标准扩展机制的一个 UML profile,通过在 UML 元模型中增加安全相关的构造型、标记值、约束等建模元素,来表达安全属性的语义和系统需求与约束. 主要扩展点在于向 UMLsec 的元素中增加相关领域的定义,使其能够更明确更有针对性地表达不同业务领域的特点,同时描述最基本的安全需求.

2.2.2　UMLsec 设计　采用 UML 安全扩展机制描述软件的安全需求，包括数据保密性和完整性，领域相关的安全知识，允许考虑不同的威胁场景，允许包含重要的安全概念，允许加入安全机制等[5-6]. 其中，UML 活动图可对软件系统的安全控制流进行建模，序列图用来表示安全关键的交互，状态图能够建立保护对象安全性的模型，而部署图可实现物理安全性需求. 根据 UML2.0 中各类模型的特性，设计了不同的构造型，详细内容参见文献[7]. 序列图描述的安全关键的交互是软件安全分析的重要组成部分，作为本文研究软件安全模型验证的基础.

2.3　安全特征序列图的语法

为了能够准确地描述安全特征序列图的动态交互过程，抽象对象生命周期内约束事件发生的序列，在文献[8]给出的序列图形式化定义的基础上，本文添加了消息约束构造型以及事件之间的偏序关系，给出如下定义.

定义 1　序列图 SD 可表示为一个八元组 SD=(O, E, M, mg, obj, ste, $\rightarrow$, $\prec$)：

(1)O 是有穷对象集合.

(2)$E=S\cup R$ 是有穷事件集合，包括发送事件集 S 和接收事件集 R.

(3)M 是有穷消息集合. 每个消息与 2 个事件相连，一个是消息发送事件 $!m\in S$，一个是消息接收事件 $?m\in R$，$!m$ 和 $?m$ 构成了一对同步事件.

(4)mg 是从 E 到 M 的函数关系，$\mathrm{mg}(e)\in M$ 表示事件 e 所对应的消息.

(5)obj 是从 E 到 O 的函数关系，$\mathrm{obj}(e)\in O$ 表示事件 e 所对应的对象.

(6)ste 是消息约束构造型，表示来自 profile 的安全约束信息.

(7)$\rightarrow$是消息集合 M 上的全序关系，在此表示序列图中的消息在纵向时间轴上的先后关系，即在一个对象的生命线上，位于上方的事件先于下方的事件发生.

(8)$\prec$是发送事件与接收事件之间的偏序关系，在此表示在两个对象之间，发送消息事件先于接收消息事件发生，即$\{(!m,?m)\mid m\in M\}\subseteq\prec$.

如图 2 所示，$O=\{A,B\}$；$E=S\cup R=\{!m,!n\}\cup\{?m,?n\}$；$M=\{m,n\}$；$\mathrm{mg}=\{<!m,m>,<?m,m>,<!n,n>,<?n,n>\}$；$\mathrm{obj}=\{<!m,A>,<?m,B>,<!n,B>,<?n,A>\}$；$\mathrm{ste}=\{\mathrm{safety}\}$；$\rightarrow=m\rightarrow n$；$\prec=\{!m\prec?m;!n\prec?n\}$.

2.4　软件安全分析的有穷状态自动机模型

为准确表达建模过程中非形式化模型刻画的安全属性，将序列图中单个对象描述的逻辑语义映射到一个对象 o，得到该对象在场景中参与的事件序列. 此序列用有穷自动机进行识别，称其为对象 o 的对象自动机，而整个序列图所描述的行为可以由所有对象的积自动机表示. 本节主要讨论如何将软件安全建模的非形式化模型映射为安全扩展确定有限自动机，从而完成下一步的验证工作.

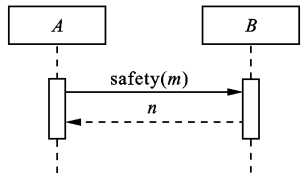

图 2　简单的序列图

2.4.1　*安全扩展确定有限自动机*　为准确表达对象在消息交互过程中的安全约束，我们在确定有限自动机定义的基础上，添加了安全属性和消息事件 2 个元组，提出了安全扩展确定有限自动机 SEDFA，定义如下.

定义 2　安全扩展确定有限自动机(Safety Extended Deterministic Finite Automata，SEDFA)是一个七元组 $M=(S,\Sigma,P,E,\delta,s_0,F)$，其中：$S$ 是一个有限状态集合；Σ 是输入字母表；P 是安全属性；E 是消息事件集合；δ 是状态转移函数；s_0 是 M 的初始状态，$s_0\in S$；F 是 M 的终止状态集合，$F\subseteq S$.

如图 3 所示，SEDFA 中，$M=(\{s_0,s_1,s_2\},\{!m,?n\},\{\mathrm{safety}\},\{!m,?n\},\{\delta(s_0,\mathrm{safety}(!m))=s_1,\delta(s_1,?n)=s_2\},s_0,\{s_2\})$.

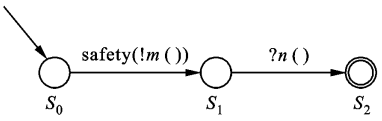

图 3　简单的 SEDFA

序列图中包括了多个交互对象，以下给出单个对象的安全扩展确定有限自动机的构造方法.

2.4.2　*序列图的安全自动机模型*　令序列图 SD=(O, E, M, mg, obj, ste, $\rightarrow$, $\prec$)，安全扩展确定有限自动机 AM=(S, Σ, P, E, δ, s_0, F)，序列图中任意对象对应于一个自动机. 状态创建：s_0 是初始状态；S 为状态的集合，对应于事件序列到对象 o 的映射；F 是自动机的可接受状态，在描述不包含组合片段的序列图时，可接受状态唯一. 安全属性：P 为所有来自于 UMLsec profile 的安全属性. 消息事件：E 为消息事件的集合，对应于对象上所有事件的集合 E. 状态转移函数：δ 为状态迁移的集合，每个迁移可

表示为(P,E),即(安全属性,消息事件),当消息事件不存在安全属性的约束时,安全属性省略.

neg组合片段能够捕获软件安全属性,在构造自动机时,若序列图包含此片段,则对应于该片段内消息事件所产生的状态均为无效状态.图3对应于图2所示序列图中对象A的SEDFA,创建步骤如下:①创建初始状态s_0;②s_1、s_2分别对应于步骤1、步骤2中事件到对象的映射;③状态之间的迁移为$s_0\times(\text{safety}\times!m)\rightarrow s_1$、$s_1\times(?n)\rightarrow s_2$;④$s_2$是自动机的可接受状态.通过以上步骤,可得到序列图中各对象的自动机模型.

3 基于模型检测的安全模型验证

模型检测是一种关于系统性质验证的策略,它通常采用状态空间搜索的方法来检查一个给定的计算模型是否满足某个用时序逻辑公式表示的特定性质[9].本文通过讨论系统自动机是否满足给定线性时序逻辑(Linear Temporal Logic,LTL)公式所描述的性质,来验证软件的安全属性.

3.1 软件安全属性的形式化描述

LTL将路径作为命题的判断对象,在状态序列上解释真值,主要用于描述安全性和活跃性2种特性.安全性描述某种不良性质ψ永不出现$(G\neg\psi)$,活跃性描述某种良好的性质一直保持.

线性时序逻辑主要包括以下时序算子:□为任一时刻算子,□p表示p总是为真,或者p永远为真;◇为某一时刻算子,◇p表示p最终为真,或者p有时为真;○为下一时刻,○p表示p在下一时刻为真;μ为直到算子,$p\mu q$表示p一直为真直到q为真;ω为除非算子,$p\omega q$表示p一直为真直到q为真,或p永远为真.

3.2 软件安全模型验证

本文采用Spin(Simple Promela Interpreter)作为软件安全模型的验证工具,Spin要求输入以Promela描述的模型M和用Never Claim结构描述的性质$\neg\psi$.Never Claim实际上是对从$\neg\psi$转换来的Büchi自动机$A_{\neg\psi}$的文字描述.验证M是否满足ψ的工作由检测器自动完成,其原理是将M转换为自动机A_M,再求积自动机$A_M\times A_{\neg\psi}$,若接受的语言为空,则M满足ψ,否则该积自动机接受的字就是M违反ψ的执行轨迹,并且检测器会反馈这些轨迹.因此,文中Promela和LTL用来描述软件安全模型和安全属性,具体验证步骤如下:①使用序列图表示软件安全分析的非形式化模型,并用SEDFA描述序列图中各个对象的逻辑语义,得到安全分析的形式化模型;②使用Promela描述步骤1所得的SEDFA;③使用LTL公式表示系统安全需求;④将步骤2与步骤3所得作为Spin的输入,观察验证结果.若存在反例,说明模型不满足安全属性;若无反例,表明模型满足性质,即系统设计满足安全需求.

4 实例验证

本节以在线订票系统为例,根据第2节所提出的软件安全性建模与验证方法,说明如何通过建立软件安全分析的序列图及其形式化的有穷状态自动机来验证软件安全模型的正确性.

4.1 在线订票系统

在线订票系统主要包括用户登录、票务信息浏览、确认购买、网上支付4个模块,用例图如图4所示,该4部分可作为系统的基本需求.针对用户登录与网上支付需附加额外的安全约束,图5给出了添加安全信息后的序列图,构造型access对登录操作进行安全限制,加密构造型encrypted对传输的账号信息进行保护.同时,在用户订票成功并返回确认消息之后,不允许执行撤销操作,该消息在序列图中的neg组合片段中出现.

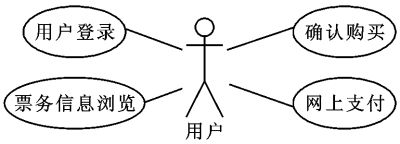

图4 在线订票系统用例图

4.2 在线订票系统序列图对象自动机的构造

图5所示的序列图包含5个对象,分别为用户、安全登录、票务信息、订单以及账单,图6a、6c、6d、6e、6f分别给出了这5个对象的安全扩展确定有限自动机模型,图6b描述了该序列图表示的一个否定迹,即用户对象执行完接收确认消息后发送撤销订单被看作不合法的动作执行过程,在验证时可作为不合法的安全属性,图6g同样描述了序列图交互过程中的一个否定迹.

4.3 在线订票系统安全模型验证

在使用自动机表示的在线订票系统安全属性序列图基础上,采用Promela描述各对象自动机,与LTL表达的安全属性共同作为模型检测器的输入.

4.3.1 *系统安全属性的LTL表示* 待验证的性质包括购买消息发生后不能撤销订单,即P之后Q不成立,即□$(P\rightarrow(\square(\neg Q)))$.$P$表示购买成功,定义为successful;$Q$表示撤销订单,定义为cancel.

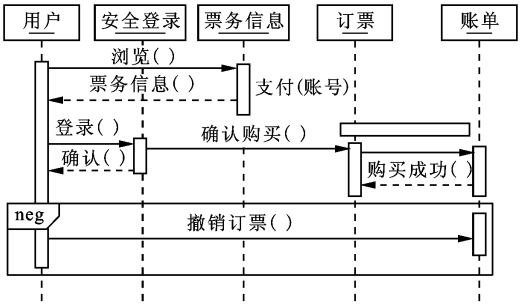

图 5　在线订票系统包含安全属性的序列图

4.3.2　实验结果　将对象自动机的 Promela 描述与待验证性质的公式描述输入到模型检测器 Spin 中，得出的验证结果如图 7 所示. 图中显示检测器发现模型有误，并给出了该错误执行轨迹的片段，其中矩形表示系统状态，数字代表状态编号，状态编号自上而下递增，表示系统演化过程，进程间的交互用箭头表示. 以上过程说明购买行为之后发生的撤消订单消息为一条无效的否定迹，所以撤销消息必发生在neg组合片段内，即系统模型满足于需求的安全

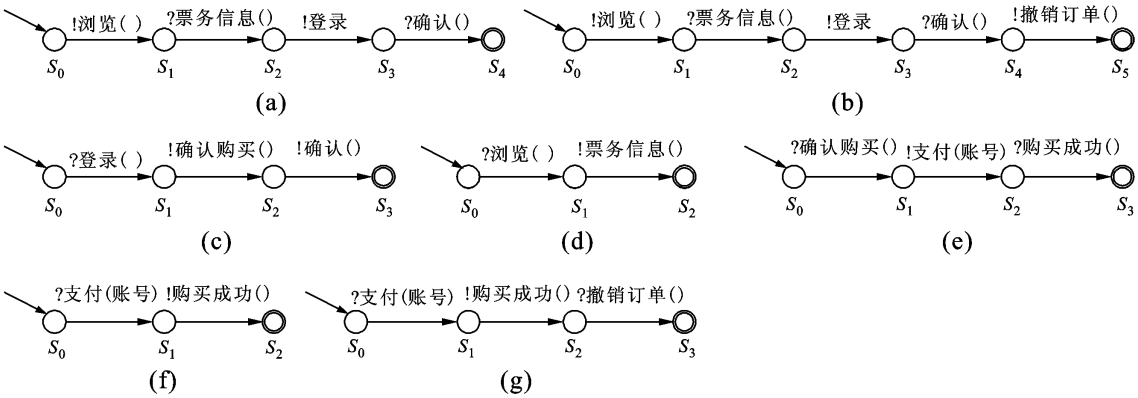

图 6　在线订票系统各对象的安全扩展确定有限自动机模型

性设计，不满足待验证性质. 此外，用与验证上述性质相同的方法可验证系统模型是否满足其他相关性质，从而判断系统设计的正确性.

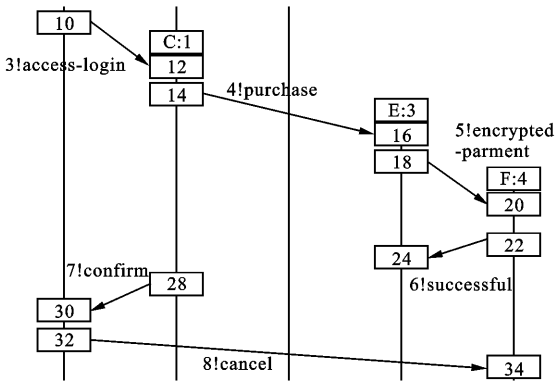

图 7　路径片段

5　结　论

本文阐述了软件安全性建模及验证方法，将安全属性添加到 UML 所表达的非形式化模型中，使用自动机建立其形式化模型，同时采用线性时序逻辑描述软件需求的安全特性，最后通过模型检测器能够证明系统模型是否满足需求中给出的安全属性，为设计与开发过程中对安全模型的进一步使用奠定了基础. 在系统规模较大的情况下，可能存在状态数过多的问题，将会影响模型检测的状态空间搜索，因此在使用序列图建模时，可使用多个序列图，从不同角度建立软件安全性的非形式化模型.

为了提高软件可靠性，改进软件开发过程，需要从不同角度对安全特征进行描述. 我们将在下一步工作中，重点研究不同的 UML 模型在安全性建模中所起的作用，同时为支持软件安全建模与验证的自动化过程，需要设计和实现一个工具，以便能够将软件安全建模过程集成到 Spin 中，完善软件安全性的自动化建模与验证，还需要对不同应用领域软件系统存在的典型安全问题进行分析，设计出具有针对性的 UMLsec profile，将该方法用于更多的软件项目中.

参考文献：

[1]　DE MIGUEL M A, BRIONES J F, SILVA J P, et al. Integration of safety analysis in model-driven software development[J]. IET Software, 2008, 2(3): 260-280.

[2] RUSSOL L. Identification, integration and tracking of software system safety requirements, CECOM-TR-94-10 [R]. Fort Monmouth, NJ, USA: CECOM, 1994.

[3] PSM. Safety measurement white paper [R]. York, UK: PSM Safety and Security TWG, 2006.

[4] JURJENS J, MUNICH T. UMLsec: presenting the profile [EB/OL]. (2002-03-18) [2008-10-27]. http://www.omg.org/news/meetings/workshops/DOC sec-2002_Proceedings/01-2_Juergens_UMLsec_Tutorial.pdf.

[5] JURJENS J, MUNICH T. Secure software architecture description using UML [EB/OL]. (2007-3-21) [2008-10-27]. http://wiki.lassy.uni.lu/tiki-download_file.php? fileId=165.

[6] BEST B, JURJENS J, NUSEIBEH B. Model-based security engineering of distributed information systems using UMLsec [C]// Proceedings of the 29th International Conference on Software Engineering. Piscataway, NJ, USA: IEEE, 2007: 581-590.

[7] CHEN Feng, LI Weihua, FANG Dingyi, et al. Research on integration of safety analysis in model-driven software development [C]// Proceedings of the 5th International Conference on Information Assurance and Security. Piscataway, NJ, USA: IEEE, 2009: 303-306.

[8] 江泽凡,王林章,李宣东,等. 基于UML顺序图的测试方法[J]. 计算机科学,2004,31(7):131-136.
JIANG Zefan, WANG Linzhang, LI Xuandong, et al. An approach to generate test cases based on UML sequences diagrams [J]. Computer Science, 2004, 31(7): 131-136.

[9] 程亮,张阳. 基于UML和模型检测的安全模型验证方法[J]. 计算机学报,2009,32(4):699-708.
CHENG Liang, ZHANG Yang. A verification method of security model based on UML and model checking [J]. Chinese Journal of Computers, 2009, 32(4): 699-708.

(编辑 武红江)