计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2015.20140306
52(7):1631-1641, 2015

基于接口精化的广义无干扰性研究

孙 聪[1] 习 宁[1] 高 胜[2] 张 涛[1] 李金库[1] 马建峰[1]
[1](西安电子科技大学网络与信息安全学院 西安 710071)
[2](中央财经大学信息学院 北京 102206)
(suncong@xidian.edu.cn)

A Generalized Non-interference Based on Refinement of Interfaces

Sun Cong[1], Xi Ning[1], Gao Sheng[2], Zhang Tao[1], Li Jinku[1], and Ma Jianfeng[1]
[1](*School of Cyber Engineering, Xidian University, Xi'an* 710071)
[2](*School of Information, Central University of Finance and Economics, Beijing* 102206)

Abstract In the design and implementation of complicated component-based softwares, the security requirements on the whole system are hard to achieve due to the difficulty on enforcing the compositionality of the security properties. The specification and verification of security properties are the critical issues in the development of component-based softwares. In this paper, we focus on the generalization on the security lattice over the information flow security properties enforceable on the component-based softwares. The previous definitions of the information flow security properties in the component-based design are restricted on the binary security lattice model. In this work, we extend the interface structure for security to the generalized interface structure for security (GISS). We define a refinement relation and use this relation to give a non-interference definition (SME-NI) based on the principle of secure multi-execution. This is the first application of secure multi-execution on the information flow security verification of component-based systems. The new definition of non-interference can be adapted to any finite security lattice models. The Coq proof assistant is used to implement the certified library for interface automata, as well as the decision procedure for the refinement relation. The experimental results show the characteristics of SME-NI and the validity of decision procedure.

Key words information flow security; non-interference; interface automata; refinement; component-based design

摘 要 在复杂构件化软件的设计和实现过程中,由于安全属性的可组合性难以实现,使得系统整体的安全需求难以得到有效保证,因而安全属性的规约和验证问题是构件化软件开发过程中关注的关键问题.针对当前构件化软件设计过程中,信息流安全属性仅局限于二元安全级格模型的问题,在现有安全接口结构基础上提出广义安全接口结构,在广义安全接口结构上定义精化关系,并利用这一精化关系定义了能够支持任意有限格模型的基于安全多执行的无干扰属性,首次将安全多执行的思想应用于构件化系统的信息流安全属性验证.使用Coq定理证明工具实现了接口自动机程序库以及对精化关系的判定过程,并用实例验证说明了无干扰属性定义的特点及判定方法的有效性.

收稿日期:2014-04-08;**修回日期**:2014-11-20
基金项目:长江学者和创新团队发展计划项目(IRT1078);国家自然科学基金委员会——广东联合基金重点基金项目(U1135002);国家科技重大专项基金项目(2011ZX03005-002);国家自然科学基金项目(61303033,61272398);陕西省自然科学基础研究计划项目(2013JQ8036);中央高校基本科研业务费专项资金项目(JB140309);航空科学基金项目(2013ZC31003,20141931001)

关键词 信息流安全;无干扰性;接口自动机;精化;构件化设计

中图法分类号 TP309.2

基于构件的软件开发(component-based software development)是一种面向软件复用和复杂软件系统开发的重要技术[1].随着 Web 服务技术和网构软件技术的深入发展,基于构件的软件描述、设计、适配、组装和验证技术正得到迅速推广和广泛应用[2].实现构件化软件的非功能需求是当前基于构件的软件开发所面临的重要而复杂的问题.组合证明理论常用于降低系统属性实现和验证的复杂性,通过构件的属性确认和组合条件检查,保证系统满足功能需求.然而,组合证明理论仍存在很多应用限制,特别是对于安全性需求,传统的组合理论[3]并不适用,因而要求我们找到更有效的方法保证构件化软件的安全性.

在构件的行为建模与设计阶段对非功能属性进行考虑是一种有效提高构件化软件开发效率的方法.接口自动机[4]是一种轻量级的构件行为模型,能够准确描述构件及构件化系统的时态行为,因而广泛地应用于构件化系统的行为建模.早期的接口自动机理论扩展包括时间接口自动机[5]、资源接口自动机[6]、以及接口自动机精化关系的扩展[7-8],这些扩展为接口自动机的应用提供了理论准备.而近年来,对接口自动机的研究已经演化到对复杂系统协议的形式化验证[9]、对信息物理融合系统的建模[10]、以及在行为建模基础上对复杂系统非功能性需求的描述和判定[11-13]等方面.

在复杂构件化系统中,由于安全属性的可组合性难以实现,使得系统整体的安全需求难以得到有效保证[14-15].针对构件化软件的安全属性规约问题,接口自动机已用于描述信任管理中的安全策略[12].经扩展后的接口自动机还用于构件化设计过程中的信息流安全属性规约和安全构件设计[13,16].信息流安全属性描述了构件化系统的端对端安全需求,在构件化软件系统设计中十分重要[17].当前针对构件化系统的信息流安全性分析仅建立在二元安全级格模型上[13,16-17],即将系统信息划分为高/低 2 个安全级,难以支持细粒度的安全策略.针对这一局限,本文基于接口自动机定义,提出广义安全接口结构的概念.在此结构上,利用安全多执行(secure multi-execution)[18]的思想,定义基于安全多执行的无干扰性,利用精化关系将构件化系统的无干扰性从二元格模型扩展到任意的有限格模型.本文还使用 Coq 定理证明器实现了接口自动机程序库以及对精化关系的判定过程,并用具体实例说明了本文无干扰性定义的特点及判定方法的有效性.

本文的主要贡献在于以下 3 方面:1)将构件化系统的信息流安全属性从二元格模型扩展到任意的有限格模型;2)提出了基于安全多执行的无干扰属性,首次将安全多执行的思想应用于构件化系统的信息流安全属性验证;3)实现了接口自动机程序库并实现了对本文无干扰属性的判定过程,判定过程能有效应用于对实际构件化设计的安全属性验证.

1 接口自动机

接口自动机是一种能够描述构件与环境交互行为及相互约束的状态迁移系统.它将描述构件行为的动作划分为输入、输出和内部动作 3 类,以描述接口对环境的输入假设及构件的输出保证.3 类动作互无交集.

定义 1. 接口自动机(interface automata).为 6 元组 $S=\langle Q,q^0,A^{\mathrm{I}},A^{\mathrm{O}},A^{\mathrm{H}},\delta\rangle$,其中 Q 为状态的有限集,$q^0\in Q$ 为初始状态,$A^{\mathrm{I}},A^{\mathrm{O}},A^{\mathrm{H}}$ 分别为输入动作集合、输出动作集合及内部动作集合,$A^{\mathrm{I}},A^{\mathrm{O}},A^{\mathrm{H}}$ 互不相交.$\delta\subseteq Q\times A\times Q$ 为迁移的有限集合,其中动作集合 $A=A^{\mathrm{I}}\cup A^{\mathrm{O}}\cup A^{\mathrm{H}}$.

接口自动机 S 应为输入确定性的,即对任意 $q,q',q''\in Q$ 及任意输入动作 $a\in A^{\mathrm{I}}$,$(q,a,q'),(q,a,q'')\in\delta$ 蕴含 $q'=q''$.定义 $(q_1,\varepsilon,q_n)\in\delta^*$,仅当存在 $a_1,\cdots,a_{n-1}\in A^{\mathrm{H}}$ 及 $q_2,\cdots,q_{n-1}\in Q$,使得 $\forall 1\leqslant i<n,(q_i,a_i,q_{i+1})\in\delta$ 或 $q_1=q_n$.对任意 $a\in A^{\mathrm{I}}\cup A^{\mathrm{O}}$,$(q,\varepsilon a,q')\in\delta^*$ 仅当 $\exists q''$,$(q,\varepsilon,q'')\in\delta^*$ 且 $(q'',a,q')\in\delta$.对于给定状态 $q\in Q$,若存在 $q'\in Q$ 使得 $(q,a,q')\in\delta$,则称动作 $a\in A$ 为激活的(enabled).对于任意 $q\in Q$,$A^{\mathrm{I}}(q),A^{\mathrm{O}}(q),A^{\mathrm{H}}(q)$ 分别表示在状态 q 下处于激活状态的输入动作集合、输出动作集合和内部动作集合.

接口自动机 S 和 T 可组合,当且仅当其中一者的输入仅能与另一者的输出相符合,即 $A_S^{\mathrm{H}}\cap A_T=A_S\cap A_T^{\mathrm{H}}=A_S^{\mathrm{I}}\cap A_T^{\mathrm{I}}=A_S^{\mathrm{O}}\cap A_T^{\mathrm{O}}=\varnothing$.可组合的接口自动机 S 与 T 兼容,如果 S 与 T 乘积的初始状态能够使得非法状态在特定的环境下不可达.对于兼容的接口自动机 S 与 T,其组合可表示为 $S\parallel T$.关于接

口自动机兼容性、乘积和组合结果的详细定义参见文献[4].

2　广义安全接口结构

为了将二元安全级格模型扩展到任意格模型,本文提出广义安全接口结构(generalized interface structure for security, GISS),该结构是安全接口结构[13]的一种扩展,它将接口自动机的可见动作$(A^{\mathrm{I}}\cup A^{\mathrm{O}})$映射到一个有限的安全级格模型中的某个安全级.

定义 2. 假设 L 为一个有限的安全级格模型,与 L 相关的广义安全接口结构$\mathbb{S}_L$定义为二元组$\langle S,\Gamma_L\rangle$,其中 $S=\langle Q,q^0,A^{\mathrm{I}},A^{\mathrm{O}},A^{\mathrm{H}},\delta\rangle$为接口自动机,$\Gamma_L:A^{\mathrm{I}}\cup A^{\mathrm{O}}\mapsto L$ 为一个映射,该映射将 S 的可见动作映射到 L 中的一个特定安全级.

一般安全接口结构将接口自动机的可见动作映射到二元格模型 $L_2=(\{l,h\},\sqsubseteq)$,其中$\sqsubseteq=\{(l,l),(l,h),(h,h)\}$.与此不同,广义安全接口结构将接口自动机的可见动作映射到任意有限的安全级格模型,从而扩展了安全接口结构的表达能力.易知,广义安全接口结构在 L_2 上退化为一般安全接口结构.

对于安全级格模型中的任一安全级 $l\in L$,可以通过 $\Gamma_L^{-1}(l)$得到对应于 l 的接口自动机可见动作集合.$\Gamma_L^{-1}(\sqsubseteq l)$和 $\Gamma_L^{-1}(\sqsubset l)$分别表示安全级低于等于 l 和高于等于 l 的可见动作集合.进一步地,$\Gamma_{L,I}^{-1}$和$\Gamma_{L,O}^{-1}$可分别用于获得安全级满足要求的可见输入和可见输出的集合.$\Gamma_L[B]$表示一个部分映射关系,其中 B 为原映射中可见动作集合的子集.

为了定义广义安全接口结构上的信息流安全属性,首先定义接口自动机和广义安全接口结构上的动作限制、动作隐藏和动作替换操作,具体如下.

定义 3. 给定广义安全接口结构$\mathbb{S}_L=\langle S,\Gamma_L\rangle$以及可见动作集合 $X\subseteq A_S^{\mathrm{I}}\cup A_S^{\mathrm{O}}$,

1) 对 S 的 X 动作限制 $S\upharpoonright X$ 表示为接口自动机$\langle Q_S,q_S^0,A_S^{\mathrm{I}}\backslash X,A_S^{\mathrm{O}}\backslash X,A_S^{\mathrm{H}},\delta_{\upharpoonright X}\rangle$,其中$(q,a,q')\in\delta_{\upharpoonright X}$当且仅当$(q,a,q')\in\delta_S$ 且 $a\notin X$.对$\mathbb{S}_L$的动作限制$\mathbb{S}_L\upharpoonright X=\langle S\upharpoonright X,\Gamma_L[A_S^{\mathrm{I}}\cup A_S^{\mathrm{O}}\backslash X]\rangle$.

2) 对 S 的 X 动作隐藏 $S\downharpoonright X$ 表示为接口自动机$\langle Q_S,q_S^0,A_S^{\mathrm{I}}\backslash X,A_S^{\mathrm{O}}\backslash X,A_S^{\mathrm{H}}\cup X,\delta_S\rangle$.对$\mathbb{S}_L$的动作隐藏$\mathbb{S}_L\downharpoonright X=\langle S\downharpoonright X,\Gamma_L[A_S^{\mathrm{I}}\cup A_S^{\mathrm{O}}\backslash X]\rangle$.

3) 将 S 的 X 动作替换为$a(a\notin A_S)$,记作 $S(a/X)$,表示为接口自动机$\langle Q_S,q_S^0,A_x^{\mathrm{I}},A_x^{\mathrm{O}},A_S^{\mathrm{H}},\delta_{S(a/X)}\rangle$,其中,①对任意 $Z\in\{\mathrm{I},\mathrm{O}\}$,若 $X\cap A_S^Z=\varnothing$,则 $A_x^Z=A_S^Z$,否则 $A_x^Z=A_S^Z\backslash X\cup\{a\}$;②对于任意$(q,b,q')\in\delta_S$,若 $b\notin X$,则$(q,b,q')\in\delta_{S(a/X)}$,否则$(q,a,q')\in\delta_{S(a/X)}$.对$\mathbb{S}_L$的 X 动作替换为 a($a\notin A$),定义为$\mathbb{S}_L(a/X)=\langle S(a/X),\Gamma_L'\rangle$,其中 Γ_L' 满足对任意 $b\in A_S^{\mathrm{I}}\cup A_S^{\mathrm{O}}\backslash X$,有 $\Gamma_L'(b)=\Gamma_L(b)$且 $\Gamma_L'(a)=\sqcap L$.其中,$\sqcap L$ 为格 L 的最小上界.为了保证 $S(a/X)$良定义,对被替换动作集合 X,一般要求 $X\subseteq A_S^{\mathrm{I}}$ 或 $X\subseteq A_S^{\mathrm{O}}$.为保证替换结果为输入确定性的,需要对替换结果执行必要的子集构造[19].

3　基于安全多执行的信息流安全属性

本节利用安全多执行(secure multi-execution, SME)[18]的基本思想定义一种信息流安全属性.安全多执行的基本思想如图 1 所示.图 1(a)为程序的正常执行,图 1(b)为对应的安全多执行.SME 将给定程序执行多次,每次执行对应于安全级格模型中的一个安全级,称为 SME 的一个单次执行.单次执行对程序输入进行指派,低安全级输入可以送入高安全级的单次执行,但高安全级输入在低安全级的单次执行中由默认行为替代.SME 保证单次执行能够输出相应安全级上的数据且与其他安全级输出无关.

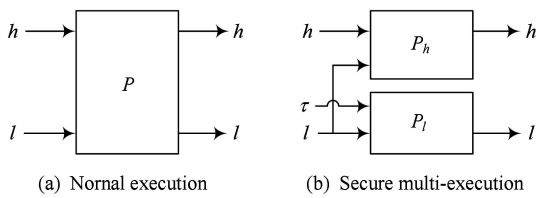


Fig. 1　The principle of secure multi-execution.

图 1　安全多执行基本思想

SME 有 2 方面特点:1)程序的 SME 执行过程满足无干扰性,即程序的高安全级输入与低安全级输出不发生依赖关系;2)虽然不安全程序的 SME 结果与一般执行结果可能存在语义上的不同,但 SME 能够保证透明性,即对于信息流安全的程序,其 SME 结果与一般执行结果无差别.

根据 SME 的透明性,若 SME 的输出结果与一般执行的输出结果有差别,则目标程序不满足信息流安全属性.本文以此作为检测接口是否安全的基本思想,定义与保证透明性相对应的信息流安全属性,保证 SME 输出结果与一般执行输出结果无差别时目标程序能够满足该属性.根据这一思想,安全接口结构应被执行多次,在 SME 过程中,安全接口

结构的每个输出动作分别由该动作对应安全级上的SME单次执行处理;对于某个安全级上的单次执行,比该安全级更高的安全级所对应的输入动作应看作某种默认动作.在进行信息流安全属性判断时,将一般执行与SME进行输出比较,以判断安全接口结构是否违反信息流安全属性.

以上SME基本思想要求对系统的2种执行的输出进行运行时比较,本文则将动态的输出比较转化为对不同广义安全接口结构的静态比较.这一转化过程描述如下:对于广义安全接口结构$\mathcal{S}_L$,其一般执行和SME的各个单次执行均由相应的接口自动机表示.在任一安全级l上,将l对应的SME单次执行与$\mathcal{S}_L$的一般执行进行比较.此时,对于一般执行,仅考虑$\mathcal{S}_L$在l上的输出动作,其他无关的输出动作可以被隐藏,同时,安全级不低于l的输入动作也应被隐藏,以避免引入对默认动作的额外干扰.因而$\mathcal{S}_L$的一般执行可看作

$$\mathcal{S}_L\downarrow(\Gamma_{L,I}^{-1}(\lnot l))\downarrow(A_S^{O}\backslash\Gamma_L^{-1}(l)). \tag{1}$$

假定$\tau(\tau\notin A)$表示默认动作,与l对应的SME单次执行可用以下广义安全接口结构表示:

$$\mathcal{S}_L(\tau/\Gamma_{L,I}^{-1}(\lnot l))\downarrow(A_S^{O}\backslash\Gamma_L^{-1}(l)). \tag{2}$$

在这一广义安全接口结构中,所有安全级不低于l的输入动作均被低安全级的默认动作τ替换.

以图2中的安全级格模型为例,安全级格模型$L_4=(\{h,m_1,m_2,l\},\sqsubseteq)$包含4个安全级,合法的信息流向由偏序关系$\sqsubseteq=\{(l,m_1),(l,m_2),(m_1,h),(m_2,h),(l,h),(m_1,m_1),(m_2,m_2),(l,l),(h,h)\}$表示.对于使用该安全级格模型的广义安全接口结构,图2(a)为一般执行的格模型表示.假定在安全级l_0上表示SME单次执行的广义安全接口结构将安全级l_1上的输入动作用默认动作替换,则将该广义安全接口结构对应的格模型中所有形如$(_,l_1)$的偏序关系用虚线表示.可知,安全级h上的SME单次执行的格模型与一般执行相同,如图2(a)所示.安全级m_1,m_2,l上的SME单次执行的格模型分别如图2(b)(c)(d)所示.

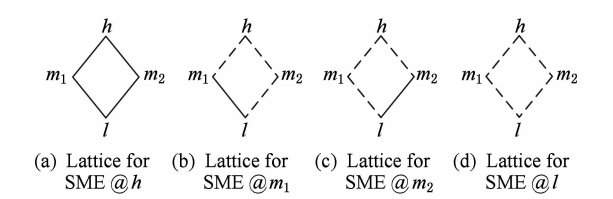


Fig. 2 Examples of security lattice model for normal execution and secure multi-execution.

图2 一般执行与SME中的安全级格模型示例

在以上转化的基础上,本节给出一种部分精化(partial refinement)关系,用以定义表示SME单次执行的广义安全接口结构与表示一般执行的广义安全接口结构二者之间是否存在机密行为信息的泄露.这一部分精化关系称为X-输入部分精化,以下为关系的具体定义.

定义4. 假设$\mathcal{S}_L=\langle S,\Gamma_L\rangle$和$\mathcal{T}_L=\langle T,\Gamma'_L\rangle$均为广义安全接口结构,对于$S$的任一输入动作的集合$X\subseteq A_S^{I}$,由$\mathcal{S}_L$到$\mathcal{T}_L$的$X$-输入部分精化为一个关系$\succ_X\subseteq Q_S\times Q_T$,其中$q_S^0\succ_X q_T^0$,且对任意的$q_S\succ_X q_T$,有以下条件成立:

1) $\forall a\in A_S^{I}\backslash X$,若$(q_S,a,q'_S)\in\delta_S$,则$\exists q'_T\in Q_T,(q_T,a,q'_T)\in\delta_T$且$q'_S\succ_X q'_T$.

2) $\forall a\in A_T^{I}$,若$(q_T,a,q'_T)\in\delta_T$,则$\exists q'_S\in Q_S,(q_S,a,q'_S)\in\delta_S$且$q'_S\succ_X q'_T$.

3) $\forall a\in A_T^{O}$,若$(q_T,a,q'_T)\in\delta_T$,则$\exists q'_S\in Q_S,(q_S,\varepsilon a,q'_S)\in\delta_S^{*}$且$q'_S\succ_X q'_T$.

4) $\forall a\in A_T^{H}$,若$(q_T,a,q'_T)\in\delta_T$,则$\exists q'_S\in Q_S,(q_S,\varepsilon,q'_S)\in\delta_S^{*}$且$q'_S\succ_X q'_T$.

由定义4可知,在X-输入部分精化关系下,$\mathcal{S}_L$仅有一部分输入迁移精化到$\mathcal{T}_L$的对应迁移,这些输入迁移执行的输入动作在$A_S^{I}\backslash X$中.如果表示SME单次执行的广义安全接口结构与表示一般执行的广义安全接口结构满足$\{\tau\}$-输入部分精化关系$\succ_{\{\tau\}}$(简写为$\succ_\tau$),则当前安全级上的执行结果输出没有差别.若所有安全级上的执行结果输出均无差别,则广义安全接口结构满足信息流安全属性,称为基于安全多执行的无干扰性.具体定义如下:

定义5. 广义安全接口结构$\mathcal{S}_L=\langle S,\Gamma_L\rangle$满足基于安全多执行的无干扰性(SME-based noninterference,SME-NI),如果对任意$l\in L$,以下关系成立

$$\mathcal{S}_L(\tau/\Gamma_{L,I}^{-1}(\lnot l))\downarrow(A_S^{O}\backslash\Gamma_L^{-1}(l))\succ_\tau$$
$$\mathcal{S}_L\downarrow(\Gamma_{L,I}^{-1}(\lnot l))\downarrow(A_S^{O}\backslash\Gamma_L^{-1}(l)).$$

SME-NI利用接口自动机之间的部分精化关系来判定广义安全接口结构的SME单次执行与一般执行是否会泄露高安全级的机密动作信息,从而保证在每个安全级上均判定通过的广义安全接口结构满足信息流安全属性.由于在最高安全级$\sqcup L$上式(1)与式(2)均等价于不含τ动作的$\mathcal{S}_L\downarrow(A_S^{O}\backslash\Gamma_L^{-1}(\sqcup L))$,易知在安全级$\sqcup L$上,定义5中的$\succ_\tau$关系总成立,因此只需判断$\succ_\tau$关系在除$\sqcup L$之外的其他安全级上是否成立.以图2为例,在安全级$m_1,m_2,l$上的精化关系判定分别能够捕获的机密动作信息泄露分别如图3(a)(b)(c)所示:

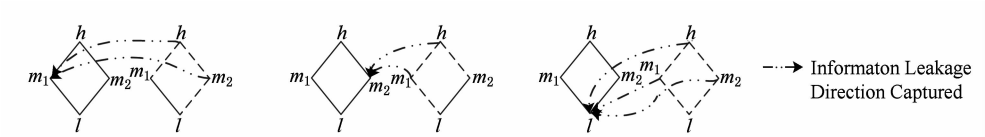


Fig. 3　Information leakage captured by decisions of refinement relation on different security levels.

图 3　不同安全级上精化关系判定所捕获的信息泄露

4　算法及工具实现

本文的信息流安全属性判定方法建立在对 2 个广义安全接口结构的精化关系判定之上,本节主要给出这一精化关系判定过程算法,具体如算法 1 所示.其中 $obsA^{\mathrm{I}}$,$obsA^{\mathrm{O}}$ 定义由某个特定状态的 ε-闭包(ε-$closure$)中的状态所发出的输入/输出动作集合,$obsA^{\mathrm{O}}$ 的具体定义与文献[20]相同,$obsA^{\mathrm{I}}$ 具体定义为

$$obsA^{\mathrm{I}}(q)=\{a\in A^{\mathrm{I}}\mid(\exists r\in\varepsilon\text{-}closure(q))(a\in A^{\mathrm{I}}(r))\}.$$

由于 X-输入部分精化关系对输入行为的限制与交互模拟(alternating simulation)关系[21]不同,因而 $obsA^{\mathrm{I}}$ 不要求 ε-闭包中的所有状态均接受输入动作 a.$ExtDest(q,a)$ 表示由 q 的 ε-闭包所发出的输入/输出动作 a 的到达状态集合,即:

$$ExtDest(q,a)=\{u'\mid u\in\varepsilon\text{-}closure(q),(u,a,u')\in\delta\}.$$

$refine\_X$ 与算法 2 和算法 3 中的 rec_1 和 rec_2 形成交互递归,用以判断定义 4 所示的 X-输入部分精化关系是否成立.在实现中算法 2,3 的可终性由特定的递减参数保证.算法 1 实现并未直接判断状态关系,而是在状态元组违反精化关系时向关系集合 $SRels$ 中加入一个表示错误的元组(err,err'),并通过判断 $refine_X(q_S^0,q_T^0,X)$的返回结果集合中是否存在元组(err,err')来确定 X-输入部分精化关系是否成立.

本文作者已使用 Coq 定理证明器[22]实现了可证明的接口自动机程序库 CertIA[23].在此基础上实现了 SME-NI 使用的精化关系判定过程.从而支持对广义安全接口结构实例进行信息流安全属性判定.

算法 1. $refine\_X(v,u,X)$.

输入:v,u 为状态,X 为输入动作子集;

输出:状态关系.

① if $obsA_S^{\mathrm{I}}(v)\setminus X\neq obsA_T^{\mathrm{I}}(u)\wedge obsA_T^{\mathrm{O}}(u)\not\subseteq obsA_S^{\mathrm{O}}(v)$ then

②　返回$\{(err,err')\}$;

③ else

④　for all $a\in obsA_S^{\mathrm{I}}(v)\cup obsA_T^{\mathrm{O}}(u)$ do

⑤　　for all $u'\in ExtDest_T(u,a)$ do

⑥　　$rec_1(v,u,X,u',ExtDest_S(v,a))$.

算法 2. $rec_1(v,u,X,u_0,dest_p)$.

输入:v,u,u_0 为状态,X 为输入动作子集,$dest_p$ 为特定到达状态集合;

输出:状态关系.

① if $dest_p=\varnothing$ then

②　返回$\{(err,err')\}$;

③ else

④　for all $v'\in dest_p$ do

⑤　　if $obsA_S^{\mathrm{I}}(v)\setminus X=\varnothing$ then {

⑥　　$SRels\leftarrow refine\_X(v',u_0,X)$;

⑦　　if $(err,err')\notin SRels$ then

⑧　　$SRels\leftarrow\{(v,u)\}\cup SRels$;

⑨　　else

⑩　　$dest_p\leftarrow dest_p\setminus\{v'\}$;$rec_1(v,u,u_0,dest_p)$; }

⑪　　else

⑫　　for all $a_0\in obsA_S^{\mathrm{I}}(v)\setminus X$ do

⑬　　if $ExtDest_S(v,a_0)=\varnothing$ then {

⑭　　$SRels\leftarrow refine\_X(v',u_0,X)$;

⑮　　if $(err,err')\notin SRels$ then

⑯　　$SRels\leftarrow\{(v,u)\}\cup SRels$;

⑰　　else

⑱　　$dest_p\leftarrow dest_p\setminus\{v'\}$;

⑲　　$rec_1(v,u,X,u_0,dest_p)$; }

⑳　　else

㉑　　for all $v''\in ExtDest_S(v,a_0)$ do

㉒　　$rec_2(v'',u_0,v',v,u,X,ExtDest_T(u,a_0),ExtDest_S(v,a_0))$.

算法 3. $rec_2(v_0,u_0,v',v,u,X,dest_q,dest_p)$.

输入:v_0,u_0,v',v,u 为状态,X 为输入动作子集,$dest_p,dest_q$ 为特定到达状态集合;

输出:状态关系.

① if $dest_q=\varnothing$ then
② 返回{(err,err′)};
③ else
④ for all $u''\in dest_q$ do {
⑤ $SRels\leftarrow refine\_X(v_0,u'',X)$;
⑥ if $(err,err')\notin SRels$ then {
⑦ $SRels'\leftarrow\{(v,u)\}\bigcup SRels$;
⑧ if $(err,err')\notin SRels'$ then
⑨ $SRels\leftarrow\{(v,u)\}\bigcup SRels$;
⑩ else
⑪ $dest_p\leftarrow dest_p\setminus\{v'\}$; $rec_1(v,u,X,u_0,dest_p)$; }
⑫ else {
⑬ $dest_q\leftarrow dest_q\setminus\{u''\}$;
⑭ $rec_2(v_0,u_0,v',v,u,X,dest_q,dest_p)$. }}

5 安全属性比较研究

本节通过对具体的实例分析及相关定理说明本文提出的 SME-NI 与已有的无干扰性定义(strict input refinement-based non-deterministic non-interference, SIR-NNI)[16]的区别与联系.实例分析与实验验证的前提条件是:1)构件接口有明确的定义;2)构件的输入/输出动作与安全级格模型有明确的映射关系.SIR-NNI 在精化关系判定时使用了限制高安全级动作的接口自动机,以此区分高安全级动作是否执行对后续动作的影响.而本文 SME-NI 在进行精化关系判定时使用的是由低安全级默认动作替换高安全级动作的接口自动机,并通过精化关系定义对高安全级动作进行限制.

在二元安全级格模型下,SME-NI 可达到与 SIR-NNI 同样的严格性.如图 4 所示,h,h_1,h_2 为高安全级输入,a,b 为低安全级输出,s_1 为接口自动机初始状态.沿用文献[16]对输入、输出和内部动作的表示方法:输入动作用"?"表示,输出动作用"!"表示,内部动作用";"表示.无干扰性判定应能正确判定攻击者是否能够通过观察接口自动机的低安全级动作获知高安全级动作是否被执行.图 4(a)所示的安全接口结构满足 SME-NI,图 4(d)为在低安全级上对图 4(a)进行的{τ}-输入部分精化关系判定过程.因 SME 单次执行对应的接口自动机的 τ 输入被排除在迁移精化之外,故只有$(s_1,a!,s_3)$与一般执行的$(s_1,a!,s_3)$和$(s_2,a!,s_3)$分别匹配.同理,在图 4(e)所示的图 4(b)精化关系判定过程中,一般执行的$(s_2,b!,s_3)$因不存在相匹配的$(s_1,\varepsilon b!,_)$,从而有$\succ_\tau$不成立,因而图 4(b)所示的安全接口结构不满足 SME-NI.图 4(c)为文献[16]中图 4 所示的接口自动机,进行 SME-NI 判定时,低安全级上的一般执行对(s_1,h_1,s_3),(s_1,h_2,s_2)进行动作隐藏,SME 单次执行对(s_1,h_1,s_3)进行动作隐藏并将(s_1,h_2,s_2)替换为(s_1,τ,s_2).SME-NI 将此接口自动机判定为安全.

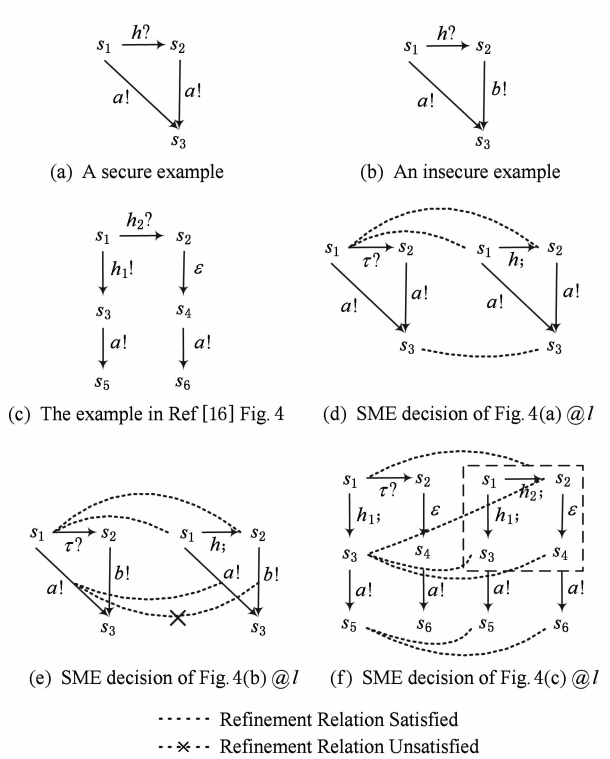


Fig. 4 Comparison between SME-NI and SIR-NNI.

图 4 SME-NI 与 SIR-NNI 的比较

SIR-NNI 也将图 4(a)(c)判断为安全,而将图 4(b)判定为不安全.由以上实例可见,在二元安全级格模型上,SME-NI 与 SIR-NNI 具有相同的严格性.实际上,{τ}-输入部分精化将 τ 输入排除在被精化输入动作之外,达到了与动作限制类似的效果.以下定理说明 SIR-NNI 与 SME-NI 在二元安全级格模型上的等价性.

定理 1. 广义安全接口结构$\mathcal{S}_{\mathbb{L}_h}=\langle S,\Gamma_{\mathbb{L}_h}\rangle$满足 SME-NI 当且仅当$\mathcal{S}_{\mathbb{L}_h}$满足 SIR-NNI.

证明. 在安全级 l 上,$\Gamma^{-1}_{\mathbb{L}_h,l}(h)=A_S^{I,h}$ 且 $A_S^O\setminus\Gamma^{-1}_{\mathbb{L}_h}(l)=A_S^{O,h}$.由$\mathcal{S}_{\mathbb{L}_h}$满足 SME-NI,有

$$\mathcal{S}_{\mathbb{L}_h}(\tau/A_S^{I,h})\upharpoonright A_S^{O,h}\succ_\tau\mathcal{S}_{\mathbb{L}_h}\upharpoonright A_S^{I,h}\upharpoonright A_S^{O,h},$$

即

$$\mathcal{S}_{\mathbb{L}_h}(\tau/A_S^{I,h})\upharpoonright A_S^{O,h}\succ_\tau\mathcal{S}_{\mathbb{L}_h}\upharpoonright A_S^h.$$

为证明上式等价于$\mathcal{S}_{\mathbb{L}_h}\upharpoonright A_S^{I,h}\upharpoonright A_S^{O,h}\rhd\mathcal{S}_{\mathbb{L}_h}\upharpoonright A_S^h$,假定

$$\mathcal{S}_{l\sqsubseteq_h}(\tau/A_S^{I,h}) \setminus A_S^{O,h} \equiv \langle S_1, \Gamma_{l\sqsubseteq_h}[A_S^l] \cup \{\tau \mapsto l\}\rangle,$$
$$\mathcal{S}_{l\sqsubseteq_h} \upharpoonright A_S^{I,h} \setminus A_S^{O,h} \equiv \langle S_2, \Gamma_{l\sqsubseteq_h}[A_S^l]\rangle,$$
$$\mathcal{S}_{l\sqsubseteq_h} \setminus A_S^h \equiv \langle T, \Gamma_{l\sqsubseteq_h}[A_S^l]\rangle.$$

S_1 和 S_2 的输出行为和隐藏行为在同一精化规则下完全相同. $\forall a \in A_{S_2}^1$,有 $a \in A_{S_1}^1 \setminus \{\tau\}$ 且 $\Gamma_{l\sqsubseteq_h}(a) = l$. 由 $(s,a,s') \in \delta_{S_2}$ 有 $\exists t', (t,a,t') \in \delta_T \wedge (s' \succ_\tau t')$ 使得 $\forall a \in A_{S_1}^1 = A_{S_2}^1 \cup \{\tau\}$,有 $\Gamma_{l\sqsubseteq_h}(a) = l$. 由 X-输入部分精化的第 1)条规则,当 $a \neq \tau$,有 $(s,a,s') \in \delta_{S_1}$ 蕴含 $(s,a,s') \in \delta_{S_2}$,从而有 $\exists t', (t,a,t') \in \delta_T \wedge s' \rhd t'$.

证毕.

根据以上分析可知,使用 SIR-NNI 无法直接对复杂安全级格模型上的接口结构进行安全属性判定,而 SME-NI 则支持任意的有限复杂格模型,且 SME-NI 在二元安全级格模型上退化为 SIR-NNI.

对于 SIR-NNI 属性,还可通过直接扩展的方式使之能够应用于复杂格模型,扩展的思想是将复杂格模型进行多次划分,每次划分都将安全级分为高/低 2 类,根据每次划分的结果将接口自动机的输入/输出动作映射到这 2 类安全级,再应用 SIR-NNI. 以图 2 安全级格模型为例,4 次划分结果分别为:$(l, m_1, m_2) \sqsubseteq (h)$,$(l, m_1) \sqsubseteq (m_2, h)$,$(l, m_2) \sqsubseteq (m_1, h)$,$(l) \sqsubseteq (m_1, m_2, h)$. 以下通过图 5 中的实例说明 SME-NI 与直接扩展 SIR-NNI 的严格性差别. 图 5 中接口自动机均使用图 2 所示安全级格模型 L_4. 假定图 5 中 h_1, h_2 为安全级 h 上的动作;m_1, m_1' 为安全级 m_1 上的动作;m_2, m_2' 为安全级 m_2 上的动作;a, b 为安全级 l 上的动作.

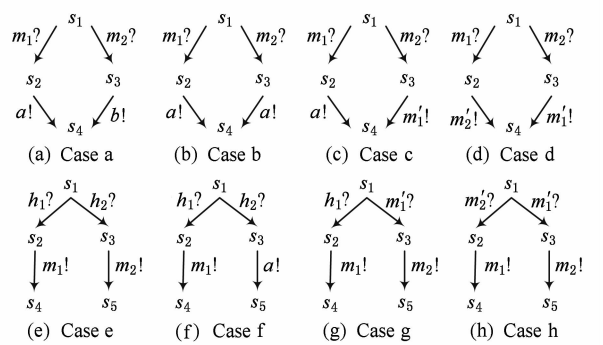


Fig. 5　Interface structures on complex security lattice models.

图 5　复杂安全级格模型上的接口结构用例

表 1 中 SME-NI 为使用本文算法对图 5 中用例进行判定的结果. 为了进行比较,我们还实现了文献[16]所用的严格输入精化(strict input refinement)关系的判定过程,使用这一判定过程得到的判定结果如表 1 中 SIR-NNI 所示. 由表 1 结果可见,SIR-NNI 直接扩展所对应的安全属性比本文 SME-NI 属性更为严格,判定出更多的违反安全属性的情况. 二者的区别之处如图 6 中的判定过程所示.

Table 1　Difference on Rigorousness between SME-NI and SIR-NNI on Complex Lattice Models

表 1　SME-NI 与 SIR-NNI 在复杂格模型上的严格性区别

Fig. 5	m_1		m_2		l	
	SME-NI	SIR-NNI	SME-NI	SIR-NNI	SME-NI	SIR-NNI
(a)	√	×	√	×	×	×
(b)	√	×	√	×	×	×
(c)	×	×	√	×	×	×
(d)	×	×	×	×	√	√
(e)	×	×	×	×	√	√
(f)	×	×	√	×	×	×
(g)	×	×	×	×	√	√
(h)	×	×	×	×	√	√

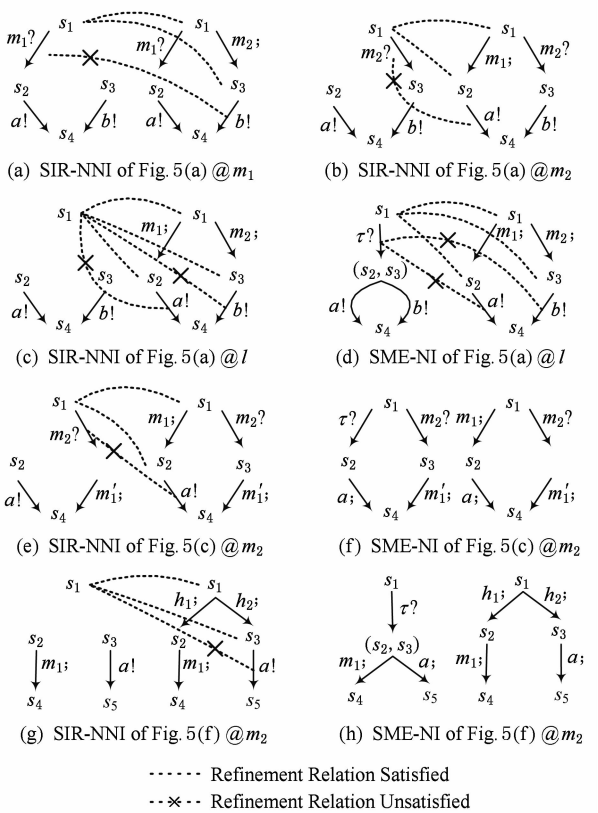


Fig. 6　Decision procedures illustrating the difference of SME-NI and SIR-NNI.

图 6　SME-NI 与 SIR-NNI 的判定过程区别

图 6(a)～(c)分别为图 5(a)接口结构在安全级 m_1, m_2, l 上的 SIR-NNI 判定过程,由于输出动作 a,b 在这 3 个安全级上均为可见,因而在 3 个安全级

上均检测到广义安全接口结构违反严格输入精化关系. 而在 SME-NI 判定过程中,由于 a,b 仅在安全级 l 上可见,因而仅在安全级 l 上检测到广义安全接口结构违反$\{\tau\}$-输入部分精化关系,如图 6(d)所示. 图 5(b)与图 5(a)的判定过程类似. 图 6(e)和图 6(f)分别说明图 5(c)广义安全接口结构在安全级 m_2 上的 SIR-NNI 和 SME-NI 判定过程,图 6(g)和图 6(h)分别为图 5(f)广义安全接口结构在安全级 m_2 上的 SIR-NNI 和 SME-NI 判定过程. 易见,SIR-NNI 直接扩展对应的安全属性比 SME-NI 更严格,而在 SME-NI 属性判定过程中,由于在较高安全级上将较低安全级的输出视为内部动作,因而可以接受一部分精化关系判断.

6 方法评价

第 5 节已通过实例和理论分析说明了 SME-NI 与 SIR-NNI 的区别与联系. 将理论上的接口安全属性满足性结果与利用本文所实现的工具进行分析得到的精化关系判定结果相比较,二者完全一致,说明算法实现的有效性. 本节将通过具体实例说明:1)精化关系判定过程的判定效率可接受;2)在实际构件化设计中,安全属性的变化及安全属性判定开销的变化.

图 7 说明本文 SME-NI 的判定过程开销大于对 SIR-NNI 的判定过程开销,图 7 中用例即图 4 和图 5 中的用例,对于图 5 中的用例,图 7 中所示时间为在安全级 m_1,m_2 和 l 上的精化关系判定时间之和. 开销增长的原因在于,SME-NI 判定要求对广义安全接口结构的输入动作替换为 τ 输入动作,而这一替换过程往往会破坏接口自动机的确定性. 使用子集构造算法恢复替换过后的接口自动机的输入确定性是 SME-NI 判定的前提,而子集构造过程是一个耗时过程. 相反地,SIR-NNI 的动作限制不会引入输入不确定性,因而可以避免子集构造过程. 图 7 中的用例名后括号中的数字表示 SME-NI 判定过程中要求的子集构造次数,可见子集构造过程对安全属性判定效率的影响. 另一方面,从总体判定时间看,2 种判定过程的开销均处于可接受范围.

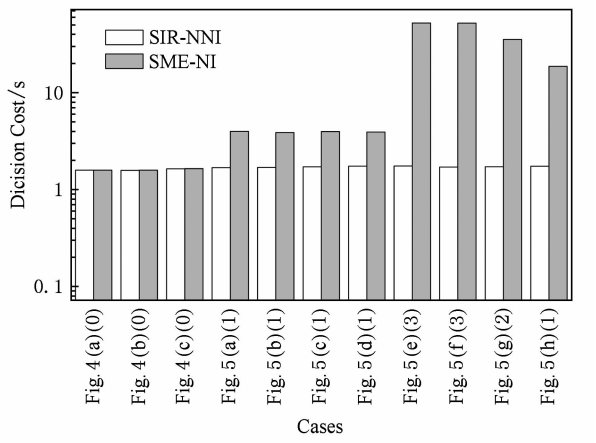


Fig. 7 Comparison on the decision cost of SME-NI and SIR-NNI.

图 7 SME-NI 与 SIR-NNI 的判定开销比较

以下通过实际构件的接口实例说明在实际设计过程中安全属性的判定. 考虑一系列典型的构件化系统的设计,要求构件的行为由接口自动机明确描述,本文在此基础上对构件及系统的输入/输出动作进行安全级限定,使之满足第 5 节提出的实验验证前提条件. 具体考虑的目标构件化系统包括:

1) CyCab 自动驾驶汽车系统[24]:系统描述由传感器标识的站点发出信息指引汽车在预定路线上行驶. 系统涉及 4 类构件:汽车 Vehicle、站点 Station、启动器 Starter、紧急停止器 EmrgHalt,如图 8 所示. 相应的广义安全接口结构建立在三元格模型 $l \sqsubseteq m \sqsubseteq h$ 上. 假设汽车与站点间的交互动作为 l 安全级,与启动器间的交互动作为 m 安全级,与紧急停止构件间的交互动作为 h 安全级.

2) 带适配器的客户端/服务器组合模型[25]:客户端 Client 和服务器 Server 满足适配条件,可以通

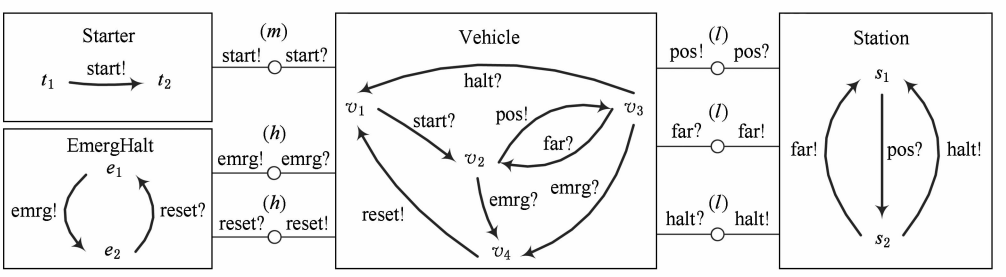


Fig. 8 Use case for the components of CyCab system.

图 8 CyCab 系统构件用例

过适配器 Adapter 进行适配组合. 相应的广义安全接口结构建立在二元格模型 $l \sqsubseteq h$ 上,并假定 ack, arg, req, value, query, service 动作对应的安全级为 h.

3) 分布式交易处理系统[13]:在监控器 SV 的监控下,远程交易处理单元 TPU 连接主服务器 TS 完成交易. 相应的广义安全接口结构建立在二元格模型 $l \sqsubseteq h$ 上,并假定 startM, logM, endM, mOn 动作对应的安全级为 h.

首先分析以上 3 组实际构件接口的 SME-NI 判定结果. Vehicle 构件在 m 安全级上满足精化关系而在 l 安全级的输出动作 pos 上违反精化关系,因而 Vehicle 构件不满足 SME-NI. 由于 Station 构件、Starter 构件和 EmrgHalt 构件均只涉及单个安全级上的动作,因而各自均为安全. 在本文的动作安全级设定下,Client, Server, Adapter 构件均为不安全,原因是 3 个构件对高安全级输入的 τ 替换均影响到了后续的低安全级输出. TPU 构件仅包含低安全级动作,因而判定为安全;SV 构件的低安全级动作均为输入动作,故也判定为安全;TS 构件安全,因为高安全级输出动作 logM 在 SME-NI 判定过程中被隐藏,使得上下 2 分支的可观察行为没有差别.

以下给出对于特定构件的组合结果的信息流安全属性判定结果,用以说明安全属性在构件组合过程中的变化. 在 Vehicle 与 Station 组合过程中,Vehicle 在安全级 l 上的输入输出动作均变为组合结果的内部动作,而无论是在 l 安全级还是 m 安全级上,组合结果的唯一输出动作 reset 在精化关系判定时均被隐藏,故组合结果被判定为满足 SME-NI. 由此可见,单个不安全的构件在组合过程中可能由于引起机密泄露的低安全级输入输出动作变为内部动作而使得组合结果为安全. 另一方面,各自判定为安全的构件也可能在组合过程中引入信息泄露,得出不安全的组合结果. 如构件 TS 与 SV 组合时,对组合结果 TS ‖ SV 的高安全级输入 mOn 的 τ 替换会影响到低安全级输出 startT,因而 TS ‖ SV 不安全.

图 9 给出了对组合结果安全性判定开销与对各构件安全性判定开销的比较结果. 用例名 A_B 中 A 与 B 均为构件,图 9 中显示构件 A 与 B 各自的判定开销之和与构件 $A \parallel B$ 判定开销的比较结果. Vehicle 构件与 EmrgHalt 构件不兼容因而考虑(Vehicle ‖ Starter)与 EmrgHalt 的组合过程(见图 9). 可知,对组合结果的安全性判定开销通常大于对各构件的安全性判定开销.

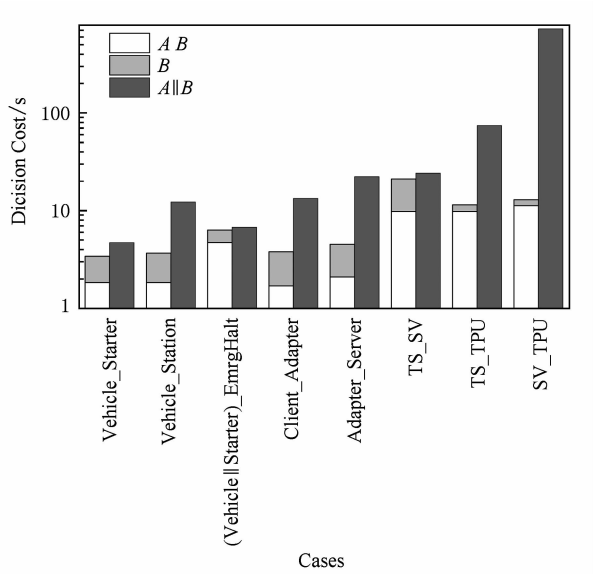


Fig. 9　Increment on the decision cost of composition results.

图 9　组合结果的判定开销增长

7　相关工作

在信息流安全领域,针对泛化的安全属性进行形式化验证是一个重要的研究方向[26]. 作为一种支持泛化信息流安全属性的实现机制,安全多执行用于保证可能不安全的程序以安全的方式执行. 近年来对安全多执行的研究主要分 3 个方向:1)为 Web 脚本程序的信息流安全提供浏览器模型和相应的动态安全监控机制[27-28];2)在反应式程序(reactive program)上研究如何在经典 SME 基础上引入机密消去模型和新的调度算法[29-30];3)将 SME 原理应用于程序变换和静态分析等静态信息流安全机制中[31-32]. 本文沿承了第 3 个方向. 反应式程序与接口自动机有一定相似性,但二者处于不同抽象层次,且二者与环境的交互方式不同,反应式程序通常通过消息信道与环境交互,接口自动机则通过输入/输出动作与环境交互,并强调与环境构件的组合. Zanarini 等人[29]将运行时监控与 SME 结合监测反应式程序是否违反 ID 安全属性,该工作与本文作者此前工作[32]均观测一般执行与 SME 之间的差别,区别在于前者使用运行时监控机制而后者使用下推系统模型检测. Barthe 等人[31]指出 SME 的缺点在于 SME 的运行时实现机制要求对计算平台进行更改,他们提出了一种程序转换方法,转换后程序的行为与原程序 SME 的行为相同,该工作中的安全性实现机制仍依赖于运行时环境,而文献[32]的静态

检测与运行时环境无关.此外,以上工作均没有提及构件化设计与构件组合过程,本文提出的基于安全多执行的无干扰性则将作者此前提出的安全多执行静态分析[32]扩展到接口自动机模型和构件化系统设计场景下.

8 总结与展望

本文提出了支持任意有限安全级格模型的基于安全多执行的无干扰性定义,并在实现接口自动机程序库的基础上实现了对无干扰性的判定过程.SME-NI属性比现有SIR-NNI属性更宽松且在二元安全级格模型上退化为SIR-NNI属性.属性判定过程的实现能够支持实际构件化设计且判定开销可接受.

未来将继续深入研究2方面工作:1)要减小对组合结果的判定开销,应找到特定的组合条件,使得单个构件的判定结果可用于对组合结果的判定且组合条件本身的判定开销较小.本文虽给出了具体实例中各构件与构件组合结果的无干扰性判定,但目前尚未给出安全属性的可组合条件,尚无法将单个构件的安全性判定结果应用于对组合结果的安全性判定,未来将研究如何开发适当的组合条件判定方法,实现安全属性在构件化系统设计过程中可保持.2)在实现本文无干扰性判定方法时,安全级格模型尚未采用归纳定义,因而对定理1的证明尚无法用Coq实现,后续将加入对安全级格模型的Coq实现和对定理1的机器证明.

参 考 文 献

[1] Mei Hong, Chen Feng, Feng Yaodong, et al. ABC: An architecture based, component oriented approach to software development [J]. Journal of Software, 2003, 14(4): 721-732 (in Chinese)
(梅宏,陈锋,冯耀东,等. ABC:基于体系结构、面向构件的软件开发方法[J]. 软件学报, 2003, 14(4): 721-732)

[2] Yang F Q, Lü J, Mei H. Technical framework for Internetware: An architecture centric approach [J]. Science in China Series F: Information Sciences, 2008, 51(6): 610-622

[3] Abadi M, Lamport L. Conjoining specifications [J]. ACM Trans on Programming Languages and Systems, 1995, 17(3): 507-534

[4] de Alfaro L, Henzinger T A. Interface automata [C] //Proc of the 8th European Software Engineering Conf Held Jointly with 9th ACM SIGSOFT Int Symp on Foundations of Software Engineering. New York: ACM, 2001: 109-120

[5] Alfaro L, Henzinger T A, Stoelinga M. Timed interfaces [G] //LNCS 2491: Proc of the 2nd Int Conf on Embedded Software. Berlin: Springer, 2002: 108-122

[6] Chakrabarti A, Alfaro L, Henzinger T A, et al. Resource interfaces [G] //LNCS 2855: Proc of the 3rd Int Conf on Embedded Software. Berlin: Springer, 2003: 117-133

[7] Wen Y J, Wang J, Qi Z C. Bridging refinement of interface automata to forward simulation of I/O automata [G] //LNCS 3308: Proc of the 6th Int Conf on Formal Engineering Methods. Berlin: Springer, 2004: 259-273

[8] Wen Y J, Wang J, Qi Z C. 2/3 alternating simulation between interface automata [G] //LNCS 3785: Proc of the 7th Int Conf on Formal Engineering Methods. Berlin: Springer, 2005: 173-187

[9] Zhang Y, Tang T, Li K P, et al. Formal verification of safety protocol in train control system [J]. Science China Technological Sciences, 2011, 54(11): 3078-3090

[10] Zhang Y, Zhang T. Hybrid interface automata [C] //Proc of the 19th Asia-Pacific Software Engineering Conf. Piscataway, NJ: IEEE, 2012: 624-633

[11] Liu L, Zhu H B, Huang Z Q, et al. Minimal privacy authorization in Web services collaboration [J]. Computer Standards and Interfaces, 2011, 33(3): 332-343

[12] Costa G, Matteucci I. Gate automata-driven run-time enforcement [J]. Computers and Mathematics with Applications, 2012, 63(2): 518-524

[13] Lee M, D'Argenio P R. Describing secure interfaces with interface automata [J]. Electronic Notes in Theoretical Computer Science, 2010, 264(1): 107-123

[14] Mantel H. On the composition of secure systems [C] //Proc of the IEEE Symp on Security and Privacy. Piscataway, NJ: IEEE, 2002: 88-101

[15] Hedin D, Sabelfeld A. A perspective on information-flow control [G] //NATO Science for Peace and Security Series-D 33: Software Safety and Security. Amsterdam: IOS, 2012: 319-347

[16] Lee M, D'Argenio P R. A refinement based notion of non-interference for interface automata: Compositionality, decidability and synthesis [C] //Proc of the XXIX Int Conf of the Chilean Computer Science Society. Piscataway, NJ: IEEE, 2010: 280-289

[17] Amtoft T, Hatcliff J, Rodriguez E, et al. Specification and checking of software contracts for conditional information flow [G] //LNCS 5014: Proc of the 15th Int Symp on Formal Methods. Berlin: Springer, 2008: 229-245

[18] Devriese D, Piessens F. Noninterference through secure multi-execution [C] //Proc of IEEE Symp on Security and Privacy. Piscataway, NJ: IEEE, 2010: 109-124

[19] Hopcroft J E, Ullman J D. Introduction to Automata Theory, Languages, and Computation [M]. 2nd ed. Boston: Addison-Wesley, 2001: 60-66

[20] de Alfaro L, Henzinger T A. Interface-based design [G] // NATO Science Series 195: Proc of the NATO Advanced Study Institute on Engineering Theories of Software Intensive Systems. Berlin: Springer, 2005: 83-104

[21] Alur R, Henzinger T A, Kupferman O, et al. Alternating refinement relations [G] //LNCS 1466: Proc of the 9th Int Conf on Concurrency Theory. Berlin: Springer, 1998: 163-178

[22] The Coq Development Team. The Coq Proof Assistant Reference Manual version 8.4 [OL]. 2012 [2014-04-08]. https://coq.inria.fr/distrib/current/refman/

[23] Cong Sun. CertIA: A coq library for interface automata [OL]. 2012 [2014-04-08]. http://infosec.pku.edu.cn/~suncong/certia.html

[24] Chouali S, Mountassir H, Mouelhi S. An I/O automata-based approach to verify component compatibility: Application to the CyCab car [J]. Electronic Notes in Theoretical Computer Science, 2010, 238(6): 3-13

[25] Chouali S, Mouelhi S, Mountassir H. Adapting component bahaviours using interface automata [C] //Proc of the 36th EUROMICRO Conf on Software Engineering and Advanced Applications. Piscataway, NJ: IEEE, 2010: 119-122

[26] Zhou Conghua, Wu Hailing, Ju Shiguang. Symbolic algorithmic verification of generalized noninference [J]. Journal of Computer Research and Development, 2012, 49(12): 2591-2602 (in Chinese)
(周从华,吴海玲,鞠时光. 广义不可推断属性符号化算术验证的研究[J]. 计算机研究与发展, 2012, 49(12): 2591-2602)

[27] Austin T H, Flanagan C. Multiple facets for dynamic information flow [C] //Proc of the 39th Annual ACM SIGPLAN-SIGACT Symp on Principles of Programming Languages. New York: ACM, 2012: 165-178

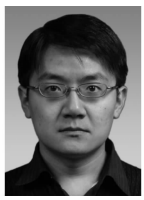
[28] De Groef W, Devriese D, Nikiforakis N, et al. FlowFox: A Web browser with flexible and precise information flow control [C] //Proc of the 19th ACM Conf on Computer and Communications Security. New York: ACM, 2012: 748-759

[29] Zanarini D, Jaskelioff M, Russo A. Precise enforcement of confidentiality for reactive systems [C] //Proc of the 26th Computer Security Foundations Symp. Piscataway, NJ: IEEE, 2013: 18-32

[30] Rafnsson W, Sabelfeld A. Secure multi-execution: Fine-grained, declassification-aware, and transparent [C] //Proc of the 26th Computer Security Foundations Symp. Piscataway, NJ: IEEE, 2013: 33-48

[31] Barthe G, Crespo J, Devriese D, et al. Secure multi-execution through static program transformation [G] //LNCS 7273: Proc of IFIP WG 6.1 Int Conf on Formal Techniques for Distributed Systems. Berlin: Springer, 2012: 186-202

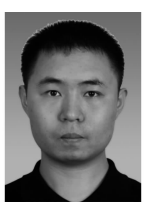
[32] Sun C, Zhai E N, Chen Z, et al. A multi-compositional enforcement on information flow security [G] //LNCS 7043: Proc of the 13th Int Conf on Information and Communications Security. Berlin: Springer, 2011: 345-359



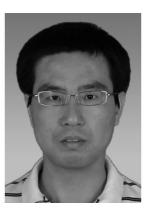
Sun Cong, born in 1982. PhD and associate professor. Member of China Computer Federation. His main research interests include information flow security and software verification.

Xi Ning, born in 1986. PhD and postdoc. His main research interests include heterogeneous network convergence, service computing and network security (xiningtom_1986@126.com).

Gao Sheng, born in 1987. PhD and assistant professor. His current research interests include mobile computing and big data with focus on security and privacy issues(sgao555@gmail.com).



Zhang Tao, born in 1986. PhD candidate. His main research interests include Web services, trusted computing, security, and social network (tzhang@stu.xidian.edu.cn).



Li Jinku, born in 1976. PhD and associate professor. His main research interests include malware defense and computer system security(jkli@xidian.edu.cn).

Ma Jianfeng, born in 1963. Professor and PhD supervisor at the School of Computer Science and Technology, Xidian University. Senior member of China Computer Federation. His main interests include distributed systems, wireless and mobile computing systems, and network and information security (jfma@mail.xidian.edu.cn).