一种基于Petri网的隐蔽信息流分析方法

周从华 鞠时光
(江苏大学计算机科学与通信工程学院 江苏 镇江 212013)

摘 要 隐蔽信息流分析是开发高等级可信计算机系统必须面对的问题.以Petri网作为开发安全系统的形式化建模工具,给出Petri网中隐蔽信息流存在的判定条件.提出该条件成立的两种网结构,进而可以在语法层次上预先判断隐蔽信息流的存在性,并使由此类结构引起的隐蔽信息流在系统的设计阶段得以避免.开发了一种基于Petri网可达图的隐蔽信息流存在性判定算法,算法遵循无干扰方法的思想,但是避免了无干扰方法中等价状态的区分和展开定理的使用.另外,算法采用深度优先搜索的策略,避免了Petri网全局可达图的构造.对复杂的安全系统,分析了子系统的各种组合运算对隐蔽信息流存在性的影响,降低了大规模系统分析的复杂度.

关键词 隐蔽信息流;无干扰;Petri网;组合;展开定理
中图法分类号 TP316 **DOI号**: 10.3724/SP.J.1016.2012.01688

A Petri Net Based Approach to Covert Information Flow Analysis

ZHOU Cong-Hua JU Shi-Guang
(*School of Computer Science and Telecommunication Engineering, Jiangsu University, Zhenjiang, Jiangsu* 212013)

Abstract The covert information flow analysis is a key problem in developing high secure systems. In this paper Petri net is used to model the dynamic behavior of the secure system. A condition characterizing the existence of the covert information flow in Petri net is proposed, and two kinds of net structures are proposed for judging the condition such that the covert information flow could be searched in the structure level. Based on the reachable graph of Petri nets, an algorithm for deciding the condition is proposed, and it follows the idea of noninterference. However, it avoids using the unwinding theorem and distinguishing the equivalent states. In addition, the algorithm is based on the depth first search strategy such that it avoids constructing the global reachable graph. For complex secure systems, we analyze how the composition operating of Petri nets influences the covert information flow such that the complexity of analyzing the large system is reduced.

Keywords covert information flow; noninterference; Petri net; composition; unwinding theorem

1 引 言

保护数据的机密性是安全系统最基本的安全需求.目前,安全计算机系统一般采用自主访问控制和强制访问控制策略来避免直接对机密数据的访问,达到约束系统中信息流动的目的.然而访问控制只能防止以访问操作为途径的显式信息扩散,并不能发现以非访问操作为途径的隐式信息扩散,如通过影响系统的运行环境间接传递信息,这种隐式信息

收稿日期:2007-09-05;最终修改稿收到日期:2012-05-12.本课题得到国家自然科学基金青年基金(61003288)、中德合作交流基金(6111130184)、江苏省自然科学基金(BK2010192)及教育部博士点基金(20093227110005)资助.**周从华**,男,1978年生,博士,副教授,主要研究方向为模型检测、访问控制、模态逻辑.E-mail: chzhou@ujs.edu.cn. **鞠时光**,男,1955年生,教授,博士生导师,主要研究领域为数据库安全、访问控制.

扩散方式通常被称为隐蔽信息流. 隐蔽信息流标识[1]一直是开发多级安全系统必须解决的关键问题，同时也是一个难题.

目前，已经诞生了一些隐蔽信息流标识方法，可以概括为两类：一类是基于程序结构的静态分析方法，如 Kemmerer 等人提出的共享资源矩阵法[2]、隐蔽流树法[3]，Denning 等人[4]提出的语法信息流方法，Tsai 等人[5]提出的语义信息流方法，卿斯汉等人[6]提出的回溯方法等；另一类是基于程序动态行为的信息流分析法，如 Gouen 等人[7-8]提出的无干扰分析法. 静态分析方法均是建立在对程序结构分析的基础上，在实际使用中可以发现一些隐蔽信息流，但是这些方法以手工处理为主，而且依赖于以往经验和个人能力，因此使用时不仅工作量大，而且非常容易出错. 信息流分析法是一种形式化方法，其本质是一个用户不应当知道他不支配的其他任何用户的任何动作. 事实上，不管是直接的信息泄露还是间接的信息泄露，都可以看成系统中信息的流动，因此信息流分析法被认为是一种有前景的方法[1]. 其主要缺点是使用时依赖“展开定理”[9]. 本文的一个研究点就是如何在基于状态转换系统的无干扰分析法中避免使用“展开定理”.

自 Gouen 等人提出无干扰分析法以来，研究人员一般基于进程代数和事件系统的形式规约来研究无干扰分析法. 采用这两种规约方式的缺点非常明显：进程代数太抽象，不易理解，而且关于进程代数的自动化分析工具不多；事件系统的主要问题在于很难实施归纳证明. 目前软件规约大多采用基于状态转换的描述方式. 与进程代数与事件系统相比，这种规约方式具有直观自然的优点，同时存在大量的有效的分析工具. 因此本文主要研究基于有限状态转换系统的无干扰分析法. 我们从基于状态转换的安全系统的形式化规约出发，以 Petri 网[10-11]作为描述安全系统规范的形式化工具来开展我们的研究. 选择 Petri 网作为安全系统动态行为建模的工具，主要基于以下两点原因：(1) Petri 网本身的特点：既有严格的形式化定义，又有直观的图形表示，存在大量设计和分析的算法，有强大的计算机工具来辅助算法的设计与分析，并且可以表示真并发，可以对系统提供不同层次的描述，使得我们的方法可以应用在系统开发的不同阶段；(2) 多级安全系统的特性在于用户实施的操作受到安全机制的监控，造成不同安全级用户被允许执行的操作和对程序输出的观察力度是不一样的，安全系统的形式化模型应该具有刻画此特性的能力，在 Petri 网中通过对变迁和库所进行划分，很容易实现该能力，从而使得信息流的描述更加自然，容易在实际系统中得到应用.

文中采用无干扰方法中隐蔽信息流存在的表现形式，从初始状态开始，删除高安全级用户所有的输入(等价于从来没有这些输入)，低安全级用户的输出没有任何变化，作为隐蔽信息流存在的充分条件. 并将该条件转化为 Petri 网上的表现形式：存在高安全级用户的单个输入引起安全系统的状态发生变化，且这种变化在低安全级用户所见的库所集上有所反映. 在建立系统的 Petri 描述后，从 Petri 网的语法层面给出了隐蔽信息流存在的两种主要结构关系：冲突关系和因果关系. 这两种关系说明什么样的结构可能会产生隐蔽信息流，这样，在安全系统设计阶段，设计人员就可以考虑避免使用这些结构. 文中主要通过计算 Petri 网的可达图过程来检测隐蔽信息流的存在性，而且检测过程是在计算可达图的过程中实施的，这样可以避免全局可达图的构造，提高了效率，节省了存储空间. 与无干扰方法相比，我们方法的优点主要体现在无需区分不同的状态等价类，避免使用展开定理.

与基于有限状态转换系统的无干扰分析法相关的研究主要有 3 个. 在文献[12]中訾小超等对无干扰方法进行了改进，提出了一个隐蔽信息流在自动机模型中存在的充分必要条件，并给出了判定算法. 但是他们的方法仍然没有避免使用展开定理. 在文献[13]中 van der Meyden 等对确定型多级安全系统，提出了一种算术的方法来验证无干扰属性. 他们首先定义了系统的双构造，然后将无干扰属性的验证问题归约为双构造系统的可达性问题. 他们的方法避免了“展开定理”的使用，但是增加了分析的系统规模，具体表现为：令 M 为一确定型安全系统，$|M|$ 表示系统中状态的个数，则 M 的双构造中状态数为 $|M|^2$. 与我们的方法相比，主要的不同点体现在两个方面：(1) 时间复杂性. 文献[13]中提出的方法等价于在 M 的双构造中进行深度优先搜索，其时间复杂性为 $O(|M|^2)$，而我们的方法等价于在系统 M 上进行深度首先搜索，时间复杂性为 $O(|M|)$；(2) 存储空间. M 的双构造中每个状态的存储空间是 M 中每个状态所需存储空间的两倍.

文献[14]提出一种通过考察 Petri 网中库所的前置集和后置集之间的关系来判断系统是否满足无干扰属性的方法，即如果发现某一库所的前置集中有高安全级的变迁，同时后置集中有低安全级的变迁，则认为系统不满足无干扰安全属性. 本文对文献[14]的这项工作进一步精化，提出了变迁之间的冲

突关系和因果关系，并分析两者和隐蔽信息流之间的关系. 此外，文献[14]只考察了 Petri 网的语法结构，而没有考虑 Petri 网的语义，从而导致会发现很多伪的隐蔽信息流. 本文通过引入 Petri 网的 L_1 活性有效地避免了该问题.

本文最后还研究了安全系统的组合问题. 当多级安全系统规模很大、很复杂时对其直接进行安全性分析是一件十分困难的事情. 现在，软件工程领域流行的开发方法是基于模块的开发，即将复杂系统化分成很多个模块，每个子模块单独开发，最后再将这些子模块按照一定的组成规则组合成一个复杂的系统. 因此我们在分析安全系统时也可以按照这种思路进行，从而降低大规模系统分析的复杂度. 文中在假设单个 Petri 网不存在隐蔽信息流的情况下，讨论了在各种组合操作下(顺序、循环、并行、选择、共享、同步)，无干扰属性是否得到保持，为开发大规模复杂的安全系统提供了指导.

2 安全系统的 Petri 网模型

2.1 Petri 网

1962 年联邦德国的 Carl Adam Petri 在他的博士论文《用自动机通信》中首次使用网状结构模拟通信系统，这种系统模型就是 Petri 网的雏形. 目前，Petri 网的研究相当广泛，在系统建模、分析、模拟、验证方面均获得了有效的应用.

定义 1. Petri 网是一个四元组 $PN=(P,T,F,M_0)$，其中

(1) P 表示库所节点集合，T 表示变迁节点集合；

(2) $P\cap T=\varnothing$，$P\cup T\neq\varnothing$；

(3) $F\subseteq(P\times T)\cup(T\times P)$，表示库所节点与变迁节点之间的有向弧集合；

(4) $M_0:P\to N$ 为初始状态，这里 N 为自然数集.

对 $\forall x\in P\cup T$，令 ${}^{\cdot}x=\{y\mid(y\in P\cup T)\wedge((y,x)\in F)\}$ 和 $x^{\cdot}=\{y\mid(y\in P\cup T)\wedge((x,y)\in F)\}$，称 ${}^{\cdot}x$ 和 $x^{\cdot}$ 分别为 x 的前置集和后置集. 称 Token 在库所节点上的分布为 Petri 网的状态，即 $M:P\to N$. 文中，采用 $M(p_i)$ 表示在库所节点 p_i 上的 Token 数. 变迁 t 在状态 M 下是使能的当且仅当 $\forall p\in{}^{\cdot}t$，$M(p)\geq1$. 使能的变迁 t 是可以引发的，引发后Petri 网的状态发生如下变化：

$$M'(p)=\begin{cases}M(p)-1, & p\in{}^{\cdot}t-t^{\cdot}\\ M(p)+1, & p\in t^{\cdot}-{}^{\cdot}t\\ M(p), & \text{其它}\end{cases}\quad(1)$$

给定 Petri 网 $PN=(P,T,F,M_0)$，在文章中会用到下面几个记号：

(1) $M_1\xrightarrow{t}M_2$：变迁 t 在状态 M_1 是使能的，且在引发后状态由 M_1 变成 M_2(按式(1)计算)；

(2) $M_1\to M_2$：存在变迁 t 使得 $M_1\xrightarrow{t}M_2$；

(3) $M_1\xrightarrow{\sigma}M_2$：变迁序列 $\sigma=t_1,\cdots,t_{n-1}$ 使得 $M_1\xrightarrow{t_1}M_2\xrightarrow{t_2}\cdots\xrightarrow{t_{n-1}}M_n$，并称 M_2 为在 M_1 下输入 σ 的输出；

(4) $M_1\xrightarrow{*}M_2$：存在变迁序列 σ 使得 $M_1\xrightarrow{\sigma}M_2$；

(5) $[M_0\rangle$ 表示 PN 的可达状态集：$[M_0\rangle=\{M\mid M_0\xrightarrow{*}M\}$；

(6) $M_1\xrightarrow{t_1}M_2\xrightarrow{t_2}\cdots\xrightarrow{t_{n-1}}M_n$ 称为路径.

定义 2(可达图). Petri 网 $PN=(P,T,F,M_0)$ 的可达图 RS 是一个三元组 (S,s_0,R)，这里 $S=[M_0\rangle$，$s_0=M_0$ 为可达图的初始状态，$R:S\times T\to S$ 定义为 $s'=R(s,t)$ 当且仅当 $s\xrightarrow{t}s'$.

称 $s_1\xrightarrow{t_1}s_2\xrightarrow{t_2}\cdots\xrightarrow{t_{n-1}}s_n$ 为 RS 中的一条路径当且仅当 $\forall1\leq i\leq n-1(s_{i+1}=R(s_i,t_i))$. 称 Petri 网 PN 是 k 界的当且仅当 $\forall M\in[M_0\rangle$，$\forall p\in P(M(p)\leq k)$. k 界 Petri 网可以转化为与之等价的 1 界 Petri 网. 由于很多系统可以用有界 Petri 网进行描述，因此文中假设所讨论的 Petri 网 PN 均是 1 界的. 在后面的讨论中会涉及到 Petri 网的 L_1 活性，因此下面引入 L_1 活性的定义.

定义 3(L_1 活性)[15]. 给定 $PN=(P,T,F,M_0)$，称变迁 $t\in T$ 具有 L_1 活性当且仅当在 PN 的可达图 RS 中存在路径 $s_0\xrightarrow{*}s_1\xrightarrow{t}s_2$，即存在一个从初始状态可达的状态 s_1，使得在 s_1 下 t 是使能的.

2.2 基于等级划分的安全系统的 Petri 网表示

为方便讨论，先约定系统的保密性要求：系统中只存在高安全级(以下简称 H 用户)和低安全级(以下简称 L 用户)两类用户，从 L 用户到 H 用户的信息流是允许的，而从 H 用户到 L 用户的信息流是禁止的. 这个保密性要求体现了多级安全策略. 这里需要说明的是，本文主要关心的是从高安全级用户到低安全级用户的信息流动. 因此和许多文献一样，为讨论的简便，假设系统中只存在两类用户. 事实上对存在多个安全等级用户的多级安全系统，从信息流动的角度看，第三方主体可能会担当中间人的角色，即信息从高安全级用户流向第三方主体，再由第三方主体流向低安全级用户. 对于满足传递性的信息流安全策略，这种情况可以归约为高安全级用户到低安全级用户的信息流动，因此两类用户的假设把握住了研究的重点，同时又不失一般性的.

每个用户都有特定的操作权限和观察能力，有

些操作对该用户是禁止的，有些输出对该用户是不可见的. 依据 L 用户的操作和观察能力，输入事件可分为 H 用户输入事件和 L 用户输入事件，分别用 T_H 和 T_L 来表示由这些事件构成的集合，且满足 $T_L \cap T_H = \varnothing$. T_L 中的事件对 L 用户是可见的，而 T_H 中的事件对 L 用户是不可见的，否则就已经形成了相应的隐蔽信息流. L 用户到 H 用户的信息流不违背保密性要求，我们不关心 H 用户是否能够观察到 L 中的事件. P_L 表示 L 用户所能观察到的库所集，对于 T 中事件的发生导致的系统状态发生变化，L 用户只能通过集合 P_L 上 Token 分布的变化来观察.

从上面的分析看出，对于安全系统，需要区分 L 用户和 H 用户的动作，因此为方便起见，以下假设 Petri 网为一五元组 $PN=(P,T_L,T_H,F,M_0)$，这里 $T_L \cup T_H = T$，$T_L \cap T_H = \varnothing$. L 用户所能观察到的库所集 P_L 定义为 $P_L = \bigcup_{t \in T_L} ({}^{\cdot}t \cup t^{\cdot})$.

系统的运行受输入事件驱动，但是系统中还存在一类输入事件，比如用户不停观察当前状态，这类事件不会改变系统的当前状态. 我们引入记号 τ 统一表示这类操作，且在任何状态 M 下 τ 都能被引发，同时在引发后状态保持不变. 为了讨论的方便，在本文的后面部分除非特别强调，否则输入事件均不包含 τ.

2.3 *PN* 的用户视图

在 PN 的运行中，由于 L 用户只能观察到 T_L 中事件的发生和 Token 在 P_L 上的分布，L 用户并不能精确观察到 PN 的全部执行活动，这相当于 L 用户只能感受到 PN 的部分执行，我们把 L 用户感受到的这一部分称为 PN 在 L 用户层面的投影，记作 L_RS.

定义 4(L 可达图). 给定 Petri 网 $PN=(P,T_L,T_H,F,M_0)$，定义其可达图 $RS(S,s_0,R)$ 在 L 用户上的投影为 $L\_RS(S_L,s_{0,L},R_L)$，这里：

(1) $S_L=\{s\downarrow_{P_L} \mid s\in S\}$，其中 $s\downarrow_{P_L}$ 定义为 $\forall p\in P_L$，$s\downarrow_{P_L}(p)=s(p)$；

(2) $s_{0,L}=s_0\downarrow_{P_L}$；

(3) $R_L: S_L\times(T_L\cup\{\varepsilon\})\to S_L$ 为状态转换函数，$R_L(s\downarrow_{P_L},t)=s'\downarrow_{P_L}$ 当且仅当下面条件中的一个成立：如果 $t\in T_L$，则 $R(s,t)=s'$；如果 $t=\varepsilon$，则 $\exists t'\in T_H$ 使得，$R(s,t')=s'$.

3　隐蔽信息流形成条件

在实际系统中，隐蔽信息流形成需要两个阶段：H 用户的信息发送和 L 用户的信息接收. 在发送阶段，H 用户执行精心设计的操作(或操作组合)，这些操作隐含了相应的机密信息，并能以一定的方式影响系统行为；在信息接收阶段，L 用户通过观察系统行为，推断 H 用户对系统所执行的操作，并进一步推出其中所隐含的机密信息. 从隐蔽信息流的实际形成来看，这两个阶段缺一不可，而形成这两个阶段的关键在于：L 用户能够感知到 H 用户对系统行为的特定影响. 无干扰方法根据 L 用户观察的系统输出来推断 H 用户对系统所执行的操作. 设 w 为一个输入序列，它的结尾是 L 的输入，w/H 表示从 w 中删除所有 H 的输入后剩下的子序列. 在初始状态输入 w 后，L 得到的输出为 $L(w)$. 显然 $L(w)$ 刻画的是 Token 在 P_L 上的分布. 本节主要考察在 L 可达图上隐蔽信息流的表现形式. 在此之前，先介绍无干扰方法的核心思想，如定理 1.

定理 1[7-8]. H 与 L 之间存在隐蔽信息流当且仅当存在以 L 的输入为结尾的输入序列 w，满足 $L(w)\neq L(w/H)$.

据定理 1 隐蔽信息流的实际关键在于：L 用户能够感知到 H 用户对系统行为的特定影响，这意味着 L 用户所感受到的系统行为并不完全由自己的输入所决定，反映在 $L\_RS(S_L,s_{0,L},R_L)$ 上就是：对同一 L 用户的输入序列，存在多条可能的转移路径，而且在系统运行中，L 用户能够知道实际是按哪一条路径进行转移的.

考察路径 $s_1\xrightarrow{l_1}s_2\xrightarrow{\varepsilon}s_2\xrightarrow{l_2}s_3$，$s_1\xrightarrow{l_1}s_2\xrightarrow{l_2}s_3$，在不考虑 s_3 以后发生的状态转换关系的情况下 L 用户所观察到的是同一条转移路径 $s_1\xrightarrow{l_1}s_2\xrightarrow{l_2}s_3$. 因此不能简单地通过路径的长度来判定不同的转移路径，需要通过下面定义的路径观察函数 Obs 来区分.

定义 5(路径观察函数). 设 $Path$ 为 $L\_RS(S_L,s_{0,L},R_L)$ 中的一条路径，$Obs(Path)=\{s_1\xrightarrow{t}s_2 \mid (t\in T_L$ 且 $s_1\xrightarrow{t}s_2$ 为 $Path$ 上的一个状态转换或者 $(t=\varepsilon\wedge s_1\neq s_2)\}$ 称为 $Path$ 的观察函数.

定义 6(不同的状态转移路径). 路径 $Path_1$ 与 $Path_2$ 是不同的当且仅当 $Obs(Path_1)\neq Obs(Path_2)$.

以下部分会涉及到在 L 输入序列 σ 下 L 用户的观察，这里的观察指的是对于任意输入序列 w，只要 $w/H=\sigma$，则 PN 在 w 下的运行路径均是 L 用户在输入序列 σ 下观察到的路径.

引理 1. PN 中存在以 L 的输入为结尾的输入序列 w，满足 $L(w)\neq L(w/H)$ 当且仅当 L 用户在输入序列 w/H 下能观察到不同的状态转移路径.

证明. (⇒)设在输入 w 和 w/H 下，状态转移

路径分别为 $Path_1$,$Path_2$. 因为 $L(w)\neq L(w/H)$,所以 $Obs(Path_1)\neq Obs(Path_2)$.

(⇐)通过对 L 用户输入序列(不包括 τ)的长度进行归纳来证明.

(1)设 L 用户输入序列的长度为 0,此时 L 用户必观察到路径 $Path_1=s_1\xrightarrow{\varepsilon}s_2\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s_n\xrightarrow{\tau}s_n$,且 $\forall 1\leq i\leq n-2(s_i=s_{i+1})\wedge s_{n-1}\neq s_n$. 在状态 s_0 下,L 用户可以输入事件 τ,并观察到路径 $Path_2=s_0\xrightarrow{\tau}s_0$. 记 w 为 $Path_1$ 上的输入序列,则 $L(w)\neq L(w/H)$.

(2)设在 L 用户输入序列长度为 $m-1(m\geq 1)$ 时结论成立.

(3)设在 L 用户输入序列 $l_1,\cdots,l_m\in T_L$ 下 L 用户有不同的观察,且在 $l_1,\cdots,l_{m-1}$ 下 L 用户没有不同的观察. 不妨设两个观察为 $s_1\xrightarrow{}\cdots\xrightarrow{l_{m-1}}s_{n-x}\xrightarrow{\varepsilon}\cdots\xrightarrow{l_m}s_n\xrightarrow{}s_{n+1}\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s_{n+u}$,$s'_1\xrightarrow{}\cdots\xrightarrow{l_{m-1}}s'_{j-y}\xrightarrow{\varepsilon}\cdots\xrightarrow{l_m}s'_j\xrightarrow{\varepsilon}s'_{j+1}\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s'_{n+v}$. 因为在序列 $l_1,\cdots,l_{m-1}$ 下 L 用户没有不同的观察,所以必然有 $s_{n-x}=\cdots=s_{n-1}=s'_{j-y}=\cdots=s'_{j-1}$,$s_n=s'_j$,路径 $s_n\xrightarrow{\varepsilon}s_{n+1}\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s_{n+u}$ 与 $s'_j\xrightarrow{\varepsilon}s'_{j+1}\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s'_{n+v}$ 上的观察函数值不一样. 不妨设存在 $0\leq i\leq u-1$ 使得($s_{n+i}\neq s_{n+i+1}$),任意的 $0\leq j<i(s_{n+j}=s_{n+j+1})$. 设 $s_1\xrightarrow{}\cdots\xrightarrow{l_{m-1}}s_{n-x}\xrightarrow{\varepsilon}\cdots\xrightarrow{l_m}s_n$ 中的引发序列为 w_1,在输入序列 w_1/H,w_1-w_1/H 下路径为 $s_1\xrightarrow{l_1}\cdots\xrightarrow{l_{m-1}}s'_m\xrightarrow{l_m}\cdots\xrightarrow{\varepsilon}s'_n$,则因为在序列 $l_1,\cdots,l_{m-1}$ 下 L 用户没有不同的观察,所以 $s'_n=s_n$. 令 $Path_1=s_1\xrightarrow{l_1}\cdots\xrightarrow{l_{m-1}}s'_m\xrightarrow{l_m}s'_{m+1}\xrightarrow{\tau}s'_{m+1}$,$Path_2=s_1\xrightarrow{l_1}\cdots\xrightarrow{l_{m-1}}s'_m\xrightarrow{l_m}s'_{m+1}\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s'_n\xrightarrow{\varepsilon}s_{n+1}\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s_{n+i}\xrightarrow{\varepsilon}s_{n+i+1}\xrightarrow{\tau}s_{n+i+1}$. 记 $Path_2$ 上的输入序列为 w,则 $Path_1$ 上的输入序列为 w/H,且 $L(w)\neq L(w/H)$. 证毕.

定理 2. PN 中存在隐蔽信息流,当且仅当在同一引发序列下 L 用户能观察到不同的状态转移路径.

定理 2 直接由定理 1 和引理 1 可得. 为加深对定理 2 的理解,我们首先讨论空输入事件下的状态转移情况. 设在 $s_{0,L}$ 状态下,H 用户输入事件序列为 $h_1,\cdots,h_n$,导致 $s_{0,L}\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s_1$,$s_{0,L}\neq s_1$. 在事件序列 $h_1,\cdots,h_n$ 没有输入的情况下,L 用户观察到的是初始状态没有变化;在事件序列 $h_1,\cdots,h_n$ 输入的情况下,L 用户观察到的是初始状态发生了变化. 因此 H 用户可以通过是否输入事件序列 $h_1,\cdots,h_n$ 来控制 L 用户的观察.

下面考虑在单个输入事件下的状态转移情况. 设在 $s_{0,L}$ 状态下,L 用户输入事件为 l_1,H 用户输入事件序列为 $h_1,\cdots,h_n$,导致 $s_{0,L}\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s_1\xrightarrow{l_1}s_2\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s_3$,$s_2\neq s_3$. 设当前状态为 s_2,在事件序列 $h_{x+1},\cdots,h_n$ 没有输入的情况下,L 用户观察到的是 s_2 没有发生变化;在事件序列 $h_{x+1},\cdots,h_n$ 输入的情况下,L 用户观察到的是 s_2 发生了变化. 因此 H 用户可以通过是否输入事件序列 $h_{x+1},\cdots,h_n$ 来控制 L 用户的观察.

多个输入事件下的状态转移情况考察,类似于空和单个输入事件,这里不再赘述. 综上所述,在同一输入下 L 用户能够观察到不同的输出,也就能够感受到 H 用户对系统行为的影响,也就能够推导出所隐蔽的机密信息;反过来,若对任意一个输入,L 用户只能观察到一种输出(一条转移路径,或者所有转移路径上都具有相同的观察),H 用户就不能影响到 L 用户的观察,也就不能传递机密信息.

4 隐蔽信息流的判定

在第 3 节中已经给出了隐蔽信息流存在的充分必要条件. 因此如何有效地判定该条件就显得尤为重要,这一部分我们将展示如何从 PN 的网结构和动态行为两个方面进行判定. 基于网结构的静态判断可以非常简便地发现 PN 中有没有隐蔽信息流,从而尽早地引导 PN 的修改. 但该方法是不完全的,因此对于通过网结构判断的 PN,还需运用基于可达图的动态判定方法. 动态判定是完全的,即通过该判定的 PN,肯定不存在隐蔽信息流.

4.1 基于网结构的隐蔽信息流存在性判定

PN 中之所以存在隐蔽信息流,是因为 H 用户的动作可以影响 L 用户的观察,这反映在 PN 的结构上就是 T_H 中的变迁和 T_L 中的变迁存在某种牵连关系. 我们主要考虑冲突关系和因果关系.

定义 7(冲突关系). 给定 $PN=(P,T_L,T_H,F,M_0)$,称 PN 具有冲突关系当且仅当 $\exists l\in T_L$,$\exists h\in T_H$ 满足 $^\cdot l\cap {}^\cdot h\neq\varnothing$ 或者 $l^\cdot\cap h^\cdot\neq\varnothing$.

定义 8(因果关系). 给定 $PN=(P,T_L,T_H,F,M_0)$,称 PN 具有因果关系当且仅当 $\exists l\in T_L$,$\exists h\in T_H$ 满足 $l^\cdot\cap {}^\cdot h\neq\varnothing$ 或者 $^\cdot l\cap h^\cdot\neq\varnothing$.

为了加深对冲突、因果关系的理解,我们给出了几种典型的冲突结构(如图 1(a),(b))和因果结构(如图 1(c),(d)). 冲突关系的本质在于变迁 h 的引发导致变迁 l 无法引发,因果关系的本质在于变迁 h 的引发导致变迁 l 可以引发,这样无论是满足冲突关系还是因果关系,H 都可以通过是否引发 h 向 L 用户传递信息.

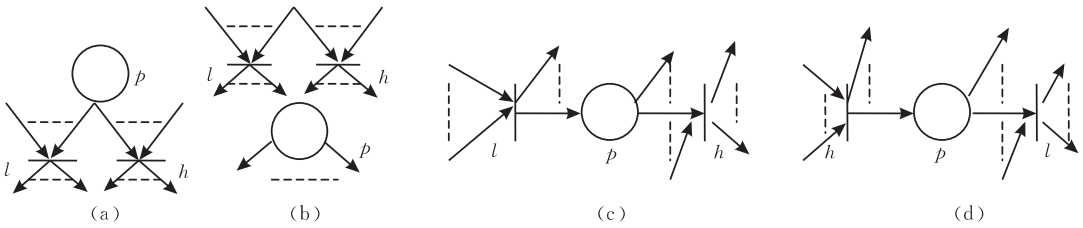


图 1　几种典型的冲突和因果关系

定理 3.　如果 $PN=(P,T_L,T_H,F,M_0)$既没有冲突关系，也没有因果关系，则 PN 中不存在隐蔽信息流.

证明. 设 PN 中存在隐蔽信息流. 由定理 2，在 PN 的 L 可达图中，存在输入变迁序列 $\sigma=l_1,\cdots,l_m$，使得在该序列下，L 用户有不同的观察 $Obs(Path_1)$，$Obs(Path_2)$. 由观察函数 Obs 的定义可知，在路径 $Path_1$ 或者 $Path_2$ 上必存在转换$(s_1\xrightarrow{\varepsilon}s_2\wedge s_1\neq s_2)$. 在状态 s_1,s_2 下，L 用户能够观察到的是 Token 在 P_L 上的分布. $s_1\neq s_2$，所以在状态 s_1，H 用户的输入变迁 h 的引发导致了 P_L 上的分布发生了变化. 由变迁引发的定义，必有 ${}^{\cdot}h\cap P_L\neq\varnothing$，或者 $h^{\cdot}\cap P_L\neq\varnothing$. 不妨设 $l\in\{{}^{\cdot}h\cap P_L\}$，则存在变迁 $t\in T_L$ 使得 $l\in{}^{\cdot}t$ 或者 $l\in t^{\cdot}$. 设 $l\in{}^{\cdot}t$，则 ${}^{\cdot}h\cap{}^{\cdot}t\neq\varnothing$，与 PN 没有冲突关系矛盾.　　证毕.

定理 3 也可以从另外一角度理解. PN 没有冲突关系，也没有因果关系，说明 L 用户和 H 用户之间没有传递信息的媒介. 没有媒介，就不可能有信息的传递.

定理 4.　给定 $PN=(P,T_L,T_H,F,M_0)$，

(1) 如果 PN 满足存在变迁 $l\in T_L$，$h\in T_h$ 使得 $({}^{\cdot}l\cap{}^{\cdot}h\neq\varnothing)$，$h$ 具有 L_1 活性，则 PN 中存在隐蔽信息流；

(2) 如果 PN 满足存在变迁 $l\in T_L$，$h\in T_h$ 使得 $(l^{\cdot}\cap h^{\cdot}\neq\varnothing)$，$h$ 具有 L_1 活性，则 PN 中存在隐蔽信息流；

(3) 如果 PN 满足存在变迁 $l\in T_L$，$h\in T_h$ 使得 $(l^{\cdot}\cap{}^{\cdot}h\neq\varnothing)$，$l,h$ 具有 L_1 活性，则 PN 中存在隐蔽信息流；

(4) 如果 PN 满足存在变迁 $l\in T_L$，$h\in T_h$ 使得 $(h^{\cdot}\cap{}^{\cdot}l\neq\varnothing)$，$l,h$ 具有 L_1 活性，则 PN 中存在隐蔽信息流.

证明.　我们只证明该定理中的第一个结论，其它结论的证明与该结论的证明类似. 变迁 h 具有 L_1 活性，因此在 PN 的可达图 RS 中存在路径 $s_0\xrightarrow{*}s_1$，且在状态 s_1 下 h 是使能的. 设 $s_0\xrightarrow{*}s_1$ 中，L 用户的输入序列为 $l_1,\cdots,l_m$. 因为在状态 s_1 下 h 是使能的，H 用户可以决定引不引发 h. 因此在 $l_1,\cdots,l_m$ 下至少可以观察到两条转移路径 $s_0\downarrow_{P_L}\xrightarrow{*}s_1\downarrow_{P_L}$ 和 $s_0\downarrow_{P_L}\xrightarrow{*}s_1\downarrow_{P_L}\xrightarrow{\varepsilon}s_2\downarrow_{P_L}$，且 $Obs(s_0\downarrow_{P_L}\xrightarrow{*}s_1\downarrow_{P_L})\neq Obs(s_0\downarrow_{P_L}\xrightarrow{*}s_1\downarrow_{P_L}\xrightarrow{\varepsilon}s_2\downarrow_{P_L})$. 由定理 2，可知 PN 中存在隐蔽信息流.　　证毕.

定理 3 和 4 说明，在系统的设计阶段应尽量少用这两种结构，从而使得由此类结构引起的隐蔽信息流在系统的设计阶段得以避免. 另外，从定理 4 可以看出隐蔽信息流的存在依赖于 PN 的初始状态 M_0. 以图 1(a)结构为例，设在某个初始状态 M_0 下，变迁 l 无法被引发，进而 h 也不可能被引发. 这样 H 就不能通过是否引发变迁 h 来传递信息.

4.2　基于可达图的隐蔽信息流存在性判定

定理 5.　Petri 网 $PN=(P,T_L,T_H,F,M_0)$ 中存在隐蔽信息流当且仅当其 L 可达图 $L\_RS(S_L,s_{0,L},R_L)$ 满足：

$$\exists n\in N(\exists s_1\in S_L,\cdots,s_n\in S_L,\ i\in\{1,\cdots,n-1\}((s_1\xrightarrow{\varepsilon}s_2\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s_n)\wedge(s_i\neq s_{i+1})))\tag{2}$$

证明.　(⇒)设 PN 存在隐蔽信息流. 通过对 L 用户引发序列的长度进行归纳来证明该充分条件.

(1) 设引发序列的长度为 0，由定理 2 可知存在路径 $Path_1=s_{0,L}\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s$，$Path_2=s_{0,L}\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s'$ 满足 $Obs(Path_1)\neq Obs(Path_2)$. 因为 $Path_1$ 与 $Path_2$ 的初始状态相同，所以在路径 $Path_1$ 或者 $Path_2$ 上必然存在状态 s_x 使得 $s_x\neq s_{x+1}$. 不妨设 s_x 在路径 $Path_1$ 上，则 $Path_1$ 满足条件(2).

(2) 设在输入序列长度为 $m-1(m\geq1)$ 时结论成立.

(3) 设在输入序列 $l_1,\cdots,l_m\in T_L$ 下 L 用户有不同的观察，且在 $l_1,\cdots,l_{m-1}$ 下 L 用户没有不同的观察.

不妨设两个观察为 $s_1\rightarrow\cdots\xrightarrow{l_{m-1}}s_{n-x}\rightarrow\cdots\xrightarrow{l_m}s_n\xrightarrow{\varepsilon}s_{n+1}\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s_{n+u}$，$s'_1\rightarrow\cdots\xrightarrow{l_{m-1}}s'_{j-y}\rightarrow\cdots\xrightarrow{l_m}s'_j\xrightarrow{\varepsilon}s'_{j+1}\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s'_{n+v}$. 因为在序列 $l_1,\cdots,l_{m-1}$ 下 L 用户没有不同的观察，所以必然有 $s_{n-x}=\cdots=s_{n-1}=s'_{j-y}=\cdots=s'_{j-1}$. 再由于 l_m 的引发和引发所导致的状态变化只

涉及到 $^{\cdot}l_m, l_m{}^{\cdot}$，且 $^{\cdot}l_m \subseteq P_L, l_m{}^{\cdot} \subseteq P_L$，所以 $s_n = s'_j$. 这样，观察不同体现在 $s_n \xrightarrow{\varepsilon} s_{n+1} \xrightarrow{\varepsilon} \cdots \xrightarrow{\varepsilon} s_{n+u}$ 与 $s'_j \xrightarrow{\varepsilon} s'_{j+1} \xrightarrow{\varepsilon} \cdots \xrightarrow{\varepsilon} s'_{n+v}$ 上，因此下面的两种情况必有一个成立：

(a) $\exists e \in \{0, \cdots, u-1\}(s_e \neq s_{e+1})$；

(b) $\exists e \in \{0, \cdots, v-1\}(s'_e \neq s'_{e+1})$.

无论哪种情况成立，均有式(2)成立.

($\Leftarrow$)设 $\exists n \in N(s_1 \in S_L, \cdots, s_n \in S_L, i \in \{1, \cdots, n-1\}((s_1 \xrightarrow{\varepsilon} s_2 \xrightarrow{\varepsilon} \cdots \xrightarrow{\varepsilon} s_n) \wedge (s_i \neq s_{i+1})))$ 成立. 因为 $s_1 \in S_L$，所以存在从初始状态 $s_{0,L}$ 到 s_1 的转移路径 $s_{0,L} \xrightarrow{*} s_1$，不妨设该转移路径上的 L 用户引发序列为 $l_1, \cdots, l_m$. 则由观察的定义 $Obs(s_{0,L} \xrightarrow{*} s_1) \neq Obs(s_{0,L} \xrightarrow{*} s_1 \xrightarrow{\varepsilon} \cdots s_i \xrightarrow{\varepsilon} s_{i+1})$，即在输入序列 $l_1, \cdots, l_m$ 下，L 用户有不同的观察. 证毕.

定理 5 表明，隐蔽信息流存在的关键在于，在 PN 的可达图中存在引发 $s \xrightarrow{h} s'$ 满足 $s\downarrow_{P_L} \neq s'\downarrow_{P_L}$. 因此有下面的推论.

推论 1. Petri 网 $PN=(P, T_L, T_H, F, M_0)$ 中存在隐蔽信息流当且仅当其 L 可达图 $L\_RS(S_L, s_{0,L}, R_L)$ 满足：

$$\exists s_1 \in S_L, s_2 \in S_L (s_1 \xrightarrow{\varepsilon} s_2 \wedge s_1 \neq s_2) \quad (3)$$

4.3 隐蔽信息流的 on-the-fly 判定

计算复杂系统的全局可达图是一件十分困难的事情，如何避免计算 PN 的全局可达图对提高隐蔽信息流的判定至关重要. 这一小节，我们将探讨如何在计算 PN 可达图的过程中判定隐蔽信息流的存在性.

由推论 1 可知，隐蔽信息流的判定关键在于检测 H 用户的输入变迁有没有引起 Token 在 P_L 上的分布变化. 而分布变化可以通过比较相连状态在 P_L 上的值是否相等来判断. 算法的基本思想是采用深度优先搜索策略计算 PN 的可达图，在计算可达图的过程通过比较相连状态在 P_L 上的值是否相等来完成隐蔽信息流存在性的判定. 设 s 为当前状态，状态 s' 是在 s 下通过引发 $t \in T$ 而得到的 s 的后继. 如果 $t \in T_H$ 且 $s\downarrow_{P_L} \neq s'\downarrow_{P_L}$，则断定隐蔽信息流存在，否则继续计算 s' 的后继. 具体的判定过程如下算法 1 所示.

算法 1. Petri 网中的隐蔽信息流判定算法.

输入：Petri 网 $PN=(P, T_L, T_H, F, M_0)$

输出：PN 中是否存在隐蔽信息流

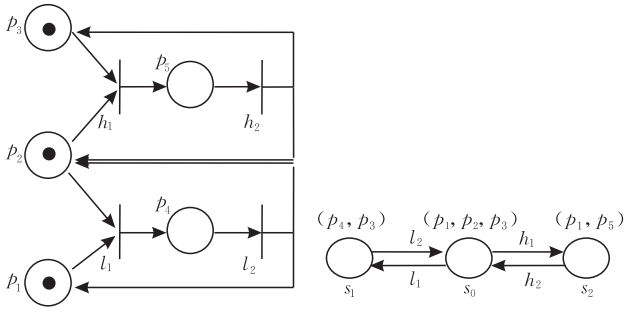
```
Main(){
  初始化：置栈 stack 为空；
  Search(M0)}
Procedure Search(s)
{Enabled(s)=∅；
While(Enabled(s)≠Enable(s)){
在集合 Enable(s)-Enabled(s)中选择变迁 t；
s'=fire(s,t)；
如果t∈TH且 s↓PL≠s'↓PL，则输出：隐蔽信息流存
    在，并退出；
Enabled(s)=Enabled(s)∪{t}；
将 s 压入栈 stack；
Search(s')；}
检查栈是否为空，如果为空，则输出：隐蔽信息流不
    存在，并退出；
从 stack 中取出栈顶元素 s''；
Search(s'')；}
```

算法 1 中的 $Enable(s)$ 表示在状态 s 下使能的变迁集合，$s'=fire(s,t)$ 表示 s' 的计算按照 2.1 节中的式(1)计算. 另外，PN 是 1 界的，所以算法肯定会终止.

4.4 示 例

这一小节我们通过一个简单的隐通道示例来说明 Petri 网中隐蔽信息流的标识. 限制进程 p 不能与进程 q 通信. 然而，进程 p 和 q 共享一个临界区，且对临界区均具有写的能力. 进程 p 和 q 对临界区的写访问满足进程间的互斥关系. 进程 p 通过周期性的对临界区执行写操作可以向进程 q 传递信息，比如在某个周期，进程 p 对临界区执行写操作，此时进程 q 就不能对临界区执行写操作，这样 p 和 q 依据事先约定的方式，q 可认为 p 向传递了一个位信息 0；相反如果进程 q 可以对临界区执行写操作，q 可认为 p 向传递了一个位信息 1.

图 2(a)是进程 p 和 q 对临界区进行访问的一个抽象模型，其中 h_1 表示进程 p 对临界区执行写操作，h_2 表示进程 p 退出访问，l_1 表示进程 q 对临界区执行写操作，l_2 表示进程 q 退出访问，p_2 是临界区是否在执行写操作的标记，p_3，p_5 是进程 p 对临界区执行写操作的标记，p_1，p_4 是进程 q 对临界区执行写操作的标记. 现在依据算法 1 标识 PN_1 中的隐蔽信



(a) 互斥算法的Petri 网模型PN_1 (b) PN_1的可达图

图 2 利用进程互斥算法实现的隐蔽信息流

息流. 首先计算 PN_1 的可达图，如图 2(b)所示，依据算法 1，在计算的过程中发现变迁 h_1 的引发导致 $s_0 \downarrow_{P_L} \neq s_2 \downarrow_{P_L}$，因此 PN_1 中存在隐蔽信息流.

5　系统组合下的隐蔽信息流分析

大规模复杂系统的分析与验证是经常遇到的难题. 通过一些较为简单的小系统利用某种运算或组合而得到较为复杂的系统，且在组合过程中保持某些性质不变，对于分析复杂系统提供了很好的途径. 假设单个 TCB 的隐蔽信息流分析已经完毕，安全系统由多个 TCB 组合而成. 下面讨论各种组合方式(连接、选择、循环、并行、共享合成、同步合成)对系统安全性的影响. 为讨论方便，我们先引入标准 Petri 网的概念. 文献[16]指出任何一个 Petri 网都可以由一个标准的 Petri 网产生. 这样，讨论连接、选择、循环、并行合成方式对安全性的影响可以直接在标准 Petri 网上进行.

定义 9(标准 Petri 网).　一个标准 Petri 网是一个六元组 $PN=(P,T,F,M_0,i,o)$，这里：

(1) P,T,F 与定义 1 中一样；

(2) i,o 满足：$i,o\in P \wedge {}^{\cdot}i=\varnothing \wedge o^{\cdot}=\varnothing$；

(3) M_0 满足：$M_0(i)=1, \forall p\in P\backslash\{i\}(M_0(p)=0)$；

(4) $\forall x\in P\cup T:(i,x)\in F^* \wedge (x,o)\in F^*$，其中 F^* 表示 F 的自反传递闭包.

为讨论方便，事先引入几个约定：

(1) 引入记号 $T_L(PN)$，$T_H(PN)$ 分别表示 PN 中 L 用户，H 用户的输入事件集合；

(2) 无论 PN 由 PN_1，PN_2 以何种方式合成所得，均有 $T_L(PN_1)\cup T_L(PN_2)\subseteq T_L(PN)$，$T_H(PN_1)\cup T_H(PN_2)\subseteq T_H(PN)$；

(3) $P_L(PN)=\{{}^{\cdot}t\cup t^{\cdot} \mid t\in T_L(PN_1)\vee t\in T_L(PN_2)\}$；

(4) 下面涉及到的任意标准 Petri 网 $PN=(P,T,F,M_0,i,o)$，均有 $i,o\in P_L(PN)$.

定义 10(Petri 网的顺序合成，如图 3)[17-18]. 设 $PN_1=(P_1,T_1,F_1,M_{01},i_1,o_1)$，$PN_2=(P_2,T_2,F_2,M_{02},i_2,o_2)$ 为标准 Petri 网，满足 $P_1\cap P_2=\varnothing$，$T_1\cap T_2=\varnothing$. 如果 $PN=(P,T,F,M_0,i,o)$ 满足：

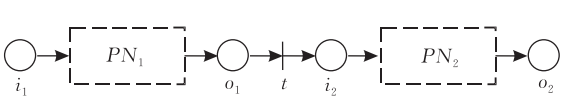


图 3　Petri 网的顺序合成

$P=P_1\cup P_2$，$T=T_1\cup T_2\cup\{t\}$，$t\in T_L(PN)$，$F=F_1\cup F_2\cup\{(o_1,t),(t,i_2)\}$，$i=i_1$，$o=o_2$，则称 PN 为 PN_1 与 PN_2 的顺序合成网，记为 $PN=PN_1\cdot PN_2$.

定理 6.　如果 PN_1 与 PN_2 中没有隐蔽信息流，则 $PN=PN_1\cdot PN_2$ 中也没有隐蔽信息流.

证明.　设 PN 存在隐蔽信息流，则由定理 5 在 PN 的可达图中存在从初始状态 s_0 出发的路径 $Path_1=s_0\xrightarrow{t_1}s_1\xrightarrow{t_2}\cdots\xrightarrow{t_n}s_n$ 满足 $s_{n-1}\downarrow_{P_L(PN)}\neq s_n\downarrow_{P_L(PN)}$. 因为 PN_1 中没有隐蔽信息流，所以 t 必然出现在 $Path_1$ 中，且 $\forall 0\leq i<n(s_i\downarrow_{P_L(PN_1)}=s_{i+1}\downarrow_{P_L(PN_1)})$. 不妨设 $t=t_i$，则 PN_2 在输入 $t_{i+1},\cdots,t_n$ 下，L 用户观察到路径 $s_{i+1}\downarrow_{P_L(PN)}\xrightarrow{t_{I+1}}\cdots\xrightarrow{t_n}s_n\downarrow_{P_L(PN)}$. 因为 $s_{n-1}\downarrow_{P_L(PN_1)}=s_n\downarrow_{P_L(PN_1)}$，所以 $s_{n-1}\downarrow_{P_L(PN_2)}\neq s_n\downarrow_{P_L(PN_2)}$，由定理 5 这与 PN_2 中没有隐蔽信息流是矛盾的.　证毕.

定义 11(Petri 网的循环合成，如图 4)[17-18]. 设 $PN_1=(P_1,T_1,F_1,M_{01},i_1,o_1)$ 为标准 Petri 网，如果 $PN=(P,T,F,M_0,i,o)$ 满足 $P=P_1\cup\{i,o\}$，$F=F_1\cup\{(i,t_1),(t_1,i_1),(o_1,t_2),(t_2,o),(o_1,t_3),(t_3,i_1)\}$，$\{t_1,t_2,t_3\}\subseteq T_L(PN)$，$T=T_1\cup\{t_1,t_2,t_3\}$，则称 PN 为 PN_1 上的循环合成网，记为 $PN=PN_1^*$.

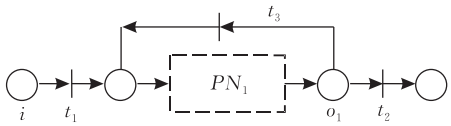


图 4　Petri 网的循环合成

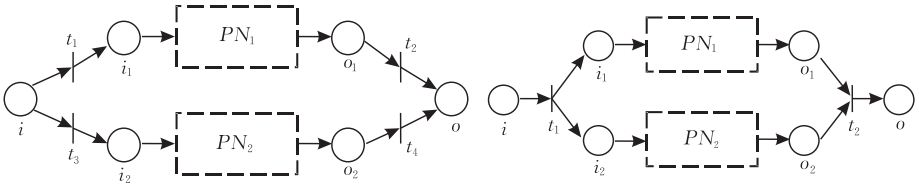
定理 7.　如果 PN_1 没有隐蔽信息流，则 $PN=PN_1^*$ 中也没有隐蔽信息流.

证明.　设 PN 存在隐蔽信息流. 在初始状态下只有 t_1 可以被引发，且 $t_1\in T_L(PN)$，所以在空输入变迁序列下，L 无法观察到这样的路径：$s_0\xrightarrow{\varepsilon}s_1\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s_n$ $(n\geq 1)$ 满足 $s_{n-1}\downarrow_{P_L(PN)}\neq s_n\downarrow_{P_L(PN)}$. 不妨设在输入 $\sigma=t_1,t_2',\cdots,t_n'(n\geq 1)$ 下导致的路径 $Path=s_0\xrightarrow{t_1}s_1\xrightarrow{t_2^1}\cdots\xrightarrow{t_n'}s_n$ 满足 $s_{n-1}\downarrow_{P_L(PN)}\neq s_n\downarrow_{P_L(PN)}$，$t_n'\in T_H(PN)$. 如果 t_2 出现在 σ 中，由 PN 的网结构可以断定 $t_2=t_n'$，与 $t_n'\in T_H(PN)$ 矛盾，所以 t_2 不会出现在 σ 中.

另外，PN_1 没有隐蔽信息流，所以 t_3 一定出现在 σ 中，不妨假设 $t_{j_1}'=\cdots=t_{j_k}'=t_3$. 将 $t_2',\cdots,t_n'$ 分成：$\sigma_1=t_2',\cdots,t_{j_1-1}',\cdots,\sigma_{k+1}=t_{j_k+1}',\cdots,t_n'$. 变迁 t_3 引发造成的后果是使 PN_1 回到了初始状态. 因此，每一个 $\sigma_i(1\leq i\leq k+1)$ 均可以在 PN_1 中引发，且在 $\sigma_{k+1}=$

$t'_{j_k+1},\cdots,t'_n$下导致的路径 $Path_1=s_0\xrightarrow{t'_{j_k+1}}s_{j_k+1}\xrightarrow{t'_{j_k+2}}\cdots\xrightarrow{t'_n}s_n$ 满足 $s_{n-1}\downarrow_{P_L(PN)}\neq s_n\downarrow_{P_L(PN)}$，这与 PN_1 没有隐蔽信息流矛盾. 证毕.

定义 12(Petri 网的选择合成，如图 5(a))[17-18]. 设 $PN_1=(P_1,T_1,F_1,M_{01},i_1,o_1)$，$PN_2=(P_2,T_2,F_2,M_{02},i_2,o_2)$为标准 Petri 网，满足 $P_1\cap P_2=\varnothing$，$T_1\cap T_2=\varnothing$. 如果 $PN=(P,T,F,M_0,i,o)$满足：$P=P_1\cup P_2\cup\{i,o\}$，$T=T_1\cup T_2\cup\{t_1,t_2,t_3,t_4\}$，$F=F_1\cup F_2\cup\{(i,t_1),(t_1,i_1),(o_1,t_2),(t_2,o),(i,t_3),(t_3,i_2),(o_2,t_4),(t_4,o)\}$，$\{t_1,t_2,t_3,t_4\}\subseteq T_L(PN)$，$i=i_1$，$o=o_2$，则称 PN 为 PN_1 与 PN_2 的冲突合成网，记为 $PN=PN_1+PN_2$.



(a) Petri网的选择合成　　(b) Petri网的并行合成

图 5 Petri 网的合成

定理 8. 如果 PN_1 与 PN_2 中没有隐蔽信息流，则 $PN=PN_1+PN_2$ 中也没有隐蔽信息流.

证明. 在初始状态下只有 t_1,t_3 可以被引发，且 $t_1,t_3\in T_L(PN)$，所以在空输入变迁序列下，L 无法观察到不同的状态转移路径. 设在序列 $\sigma=t'_1,t'_2,\cdots,t'_n$ 下，L 用户观察到路径 $Path_1=s_0\xrightarrow{\varepsilon}s_1\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s_n(n\geq 2)$ 满足 $s_{n-1}\downarrow_{P_L(PN)}\neq s_n\downarrow_{P_L(PN)}$. 从 PN 的结构上可以看出 $t'_1=t_1$ 或者 $t'_1=t_3$，不妨设 $t'_1=t_1$. 当 t_1 引发以后，PN 就按照 PN_1 的运行轨迹运行，因此在 PN_1 中存在路径 $Path_2=s_1\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s_n(n\geq 2)$ 且满足 $s_{n-1}\downarrow_{P_L(PN_1)}\neq s_n\downarrow_{P_L(PN_1)}$，这与 PN_1 中没有隐蔽信息流矛盾. 证毕.

定义 13(Petri 网的并行合成，如图 5(b))[17-18]. 设 $PN_1=(P_1,T_1,F_1,M_{01},i_1,o_1)$，$PN_2=(P_2,T_2,F_2,M_{02},i_2,o_2)$为标准 Petri 网，满足 $P_1\cap P_2=\varnothing$，$T_1\cap T_2=\varnothing$. 如果 $PN=(P,T,F,M_0,i,o)$满足：$P=P_1\cup P_2\cup\{i,o\}$，$T=T_1\cup T_2\cup\{t_1,t_2\}$，$F=F_1\cup F_2\cup\{(i,t_1),(t_1,i_1),(o_1,t_2),(t_2,o),(t_1,i_2),(o_2,t_2)\}$，$\{t_1,t_2\}\subseteq T_L(PN)$，$i=i_1$，$o=o_2$，则称 PN 为 PN_1 与 PN_2 的并行合成网，记为 $PN=PN_1\ /\!/\ PN_2$.

定理 9. 如果 PN_1 与 PN_2 中没有隐蔽信息流，则 $PN=PN_1\ /\!/\ PN_2$ 中也没有隐蔽信息流.

证明. 设在序列 $\sigma=t'_1,t'_2,\cdots,t'_n$ 下，L 用户观察到路径 $Path_1=s_0\xrightarrow{\varepsilon}s_1\xrightarrow{\varepsilon}\cdots\xrightarrow{\varepsilon}s_n(n\geq 1)$，且满足 $s_{n-1}\downarrow_{P_L(PN)}\neq s_n\downarrow_{P_L(PN)}$. 从 PN 的结构上可以看出，可以将 $\sigma=t'_1,t'_2,\cdots,t'_n$ 分别投影在 $\{t_1,t_2\}\cup T_1$ 和 $\{t_1,t_2\}\cup T_2$ 上，分别将投影记为 σ_1,σ_2. 从 PN 的结构还可以断定，σ_1,σ_2 的运行互不干扰，因此必有 $s_{n-1}\downarrow_{P_L(PN_1)}\neq s_n\downarrow_{P_L(PN_1)}$ 或者 $s_{n-1}\downarrow_{P_L(PN_2)}\neq s_n\downarrow_{P_L(PN_2)}$. 无论哪种情况成立，与 PN_1,PN_2 中没有隐蔽信息流矛盾. 证毕.

下面我们讨论共享合成与同步合成对隐蔽信息流的影响，为讨论的方便采取的 Petri 形式为定义 1 中给出的四元组形式.

定义 14(共享合成，如图 6)[19]. 设 $PN_1=(P_1,T_1,F_1,M_{01})$，$PN_2=(P_2,T_2,F_2,M_{02})$为 Petri 网，满足 $P_1\cap P_2\neq\varnothing$，$T_1\cap T_2=\varnothing$. 如果 $PN=(P,T,F,M_0)$满足：$P=P_1\cup P_2$，$T=T_1\cup T_2$，$F=F_1\cup F_2$，若 $p\in P_1\cap P_2$，$M_0(p)=\max\{M_{01}(p),M_{02}(p)\}$，若 $p\in P_i-(P_1\cap P_2)$，$M_0(p)=M_{0i}(p)$ $(i\in\{1,2\})$，则称 PN 为 PN_1 与 PN_2 的共享合成网，记为 $PN=PN_1\Delta PN_2$.

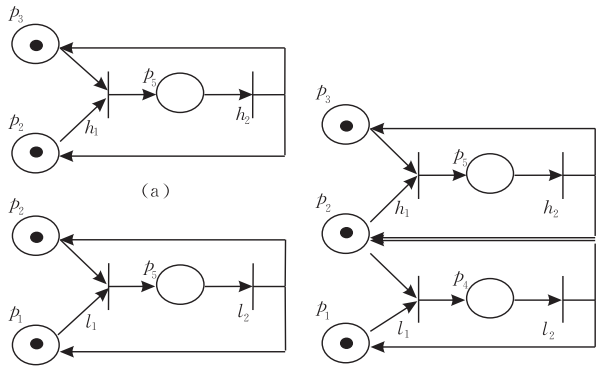


图 6 Petri 网的共享合成

定理 10. 如果 PN_1 与 PN_2 中不存在隐蔽信息流，则 $PN=PN_1\Delta PN_2$ 中可能存在隐蔽信息流.

图 6(c)中的 Petri 网是由图 6(a)，图 6(b)通过共享合成而得. 图 6(a)中没有 L 用户，图 6(b)中没有 H 用户，因此两者均不会存在隐蔽信息流，但是图 6(c)中却存在隐蔽信息流.

定义 15(同步合成Ⅰ)[20-21]. 设 $PN_1=(P_1,T_1,F_1,M_{01})$，$PN_2=(P_2,T_2,F_2,M_{02})$为 Petri 网，

满足 $P_1 \cap P_2 = \varnothing$，$T_1 \cap T_2 = \varnothing$. 如果 $PN=(P,T,F,M_0)$ 满足：$P=P_1 \cup P_2$，$T=T_1 \cup T_2$，$F=F_1 \cup F_2$，$M_0(p)=M_{0i}(p)(i\in\{1,2\})$，则称 PN 为 PN_1 与 PN_2 的同步Ⅰ型合成网，记为 $PN=PN_1 \nabla_1 PN_2$.

定理 11. 如果 PN_1 与 PN_2 中不存在隐蔽信息流，则 $PN=PN_1 \nabla_1 PN_2$ 中也不存在隐蔽信息流.

证明. 设在序列 $\sigma=t'_1,t'_2,\cdots,t'_n$ 下，L 用户观察到路径 $Path_1=s_0 \xrightarrow{\varepsilon} s_1 \xrightarrow{\varepsilon} \cdots \xrightarrow{\varepsilon} s_n (n\geq 1)$，且满足 $s_{n-1}\downarrow_{P_L(PN)} \neq s_n \downarrow_{P_L(PN)}$. 从 PN 的结构上可以看出，可以将 $\sigma=t'_1,t'_2,\cdots,t'_n$ 分别投影在 $T(PN_1)$ 和 $T(PN_2)$ 上，分别将投影记为 σ_1,σ_2. 从 PN 的结构可以断定，σ_1,σ_2 的运行互不干扰，因此必有 $s_{n-1}\downarrow_{P_L(PN_1)} \neq s_n \downarrow_{P_L(PN_1)}$ 或者 $s_{n-1}\downarrow_{P_L(PN_2)} \neq s_n \downarrow_{P_L(PN_2)}$. 无论哪种情况成立，与 PN_1，PN_2 中没有隐蔽信息流矛盾. 证毕.

定义 16（同步合成Ⅱ）[20-21]. 设 $PN_1=(P_1,T_1,F_1,M_{01})$，$PN_2=(P_2,T_2,F_2,M_{02})$ 为 Petri 网，满足 $P_1 \cap P_2=\varnothing$，$T_1 \cap T_2 \neq \varnothing$，$T_L(PN_1) \cap T_H(PN_2)=\varnothing$，$T_H(PN_1) \cap T_L(PN_2)=\varnothing$. 如果 $PN=(P,T,F,M_0)$ 满足：$P=P_1 \cup P_2$，$T=T_1 \cup T_2$，$F=F_1 \cup F_2$，$M_0(p)=M_{0i}(p)(i\in\{1,2\})$，则称 PN 为 PN_1 与 PN_2 的同步Ⅱ型合成网，记为 $PN=PN_1 \nabla_2 PN_2$.

定理 12. 如果 PN_1 与 PN_2 中不存在隐蔽信息流，则 $PN=PN_1 \nabla PN_2$ 中也不存在隐蔽信息流.

证明. 设在序列 $\sigma=t'_1,t'_2,\cdots,t'_n (t'_n\in T_H(PN))$ 下，L 用户观察到路径 $Path_1=s_0 \xrightarrow{*} s_1 \xrightarrow{*} \cdots \xrightarrow{\varepsilon} s_n (n\geq 1)$，且满足 $s_{n-1}\downarrow_{P_L(PN)} \neq s_n \downarrow_{P_L(PN)}$. 首先考虑 $t'_n \in T_1 \cap T_2$，因为 $P_1 \cap P_2=\varnothing$，$s_{n-1}\downarrow_{P_L(PN)} \neq s_n \downarrow_{P_L(PN)}$，所以必有 $s_{n-1}\downarrow_{P_L(PN_1)} \neq s_n \downarrow_{P_L(PN_1)}$ 或者 $s_{n-1}\downarrow_{P_L(PN_2)} \neq s_n \downarrow_{P_L(PN_2)}$. 无论哪种情况成立，与 PN_1，PN_2 中不存在隐蔽信息流矛盾. 下面考虑 $t'_n \notin T_1 \cap T_2$，不妨设 $t'_n \in T_1$. 因为 $P_1 \cap P_2=\varnothing$，$s_{n-1}\downarrow_{P_L(PN)} \neq s_n \downarrow_{P_L(PN)}$，所以必有 $s_{n-1}\downarrow_{P_L(PN_1)} \neq s_n \downarrow_{P_L(PN_1)}$，与 PN_1 中不存在隐蔽信息流矛盾. 证毕.

6　实例分析

本小节我们将采用文献[14]中的一个实例来说明如何标识隐蔽信息流. 在一个支持强制安全等级的系统中，如 UNIX 操作系统，当进程新建和删除一个特权目录时就会受到某些限制，图 7 对这种限制给出了说明. 图 7(a)和(b)说明如果 $L_h > L_i$，则等级是 L_h 的进程不能创建和删除等级是 L_i 的目录 D 中的特权目录 UD. 图 7(c)和(d)说明特权目录只能被低安全级的进程创建和删除.

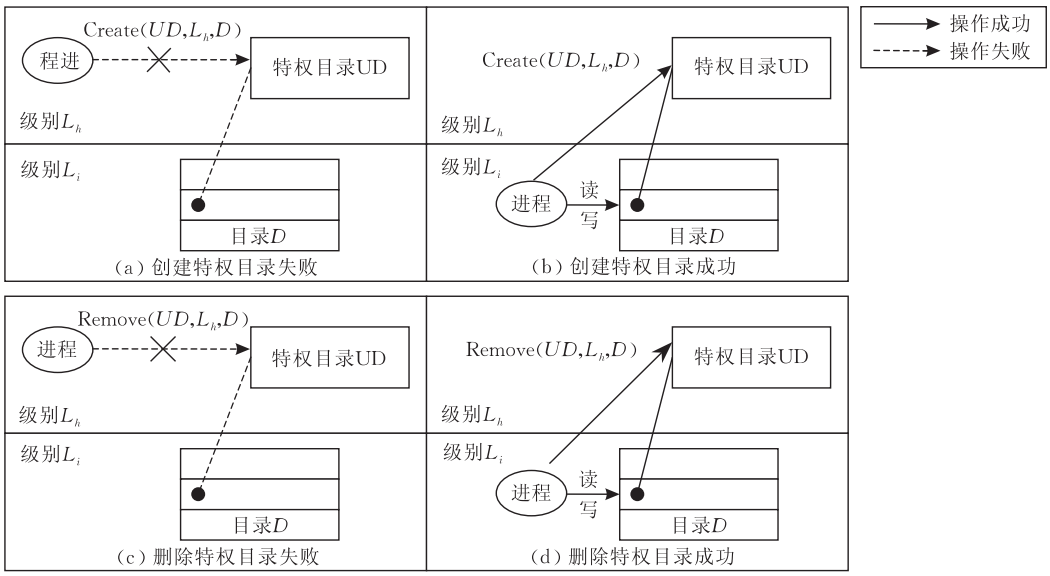


图 7　新建和删除特权目录

第 4 节的结论说明在使用 Petri 网对该系统进行建模时，只需考察高低进程有交互的部分，即特权目录 UD. 在特权目录中，高安全级进程可以读、写、新建、删除文件，而低安全级进程是不允许的. 在 Petri 网中定义一些变迁来表示高低安全级进程可以执行的操作，如表 1 所示. 注意对于低安全级进程，只有当特权目录中没有任何文件时才能被删除.

表 1　Petri 网中的变迁

	读文件	写文件	新建特权目录	删除特权目录	新建文件	删除文件
高安全级进程	Read	Write	×	×	New	Delete
低安全级进程	×	×	Create	Remove	×	×

图 8 是高低进程访问特权目录 UD 的抽象过程的 Petri 网模型. 在该模型中 L_{Creat} 和 L_{Remove} 分别表示低安全级进程创建和删除特权目录 UD. H_{New}, H_{Delete}, $H_{Read/Write}$ 分别表示高安全级进程创建文件、删除文件、读和写文件. 模型描述的主要访问过程是当 UD 创建好之后,如果里面没有文件,则可以直接删除,如果高安全级进程创建了文件,则必须等待文件删除后 UD 才能被删除. 图 8 中的 Petri 网比较简单,可以直接看出每一个变迁都具有 L_1 活性,且 ${}^{\cdot}L_{Remove} \cap {}^{\cdot}H_{New} \neq \varnothing$. 由定理 4 直接可知存在隐蔽信息流.

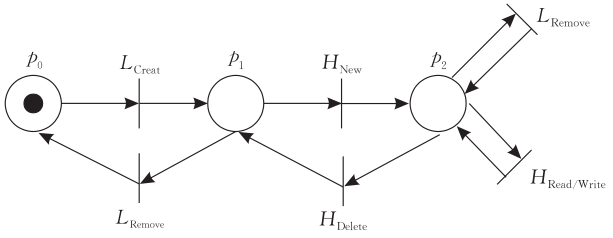


图 8 高低进程访问特权目录 UD 的 Petri 网模型

${}^{\cdot}L_{Remove} \cap {}^{\cdot}H_{New} \neq \varnothing$ 说明 H_{New} 的触发与否直接决定了 L_{Remove} 能否被触发. 直观上的解释就是,当低安全进程创建好 UD 目录后,高安全级进程可以通过是否创建新文件来向低进程传递信息. 如果高安全级进程创建新文件,则低安全级进程不能删除目录 UD,此时可以理解为高安全级进程向低安全级进程传递了一个比特信息. 如果高安全级进程没有创建新文件,则低安全级进程可以成功删除目录 UD,此时可以理解为高安全级进程向低安全级进程传递了另外一个比特信息,从而造成了信息从高安全级进程向低安全级进程的泄漏.

7 结 论

隐蔽信息流标识是一个众所周知的难题,目前尚无理论上成熟、实际上可行的标识方法. 其中,无干扰方法的主要问题在于必须依靠“展开定理”. 本文遵循无干扰方法的思想,以基于状态转换系统的形式化规约为处理对象,使用 Petri 网作为开发 TCB 的形式化建模工具,借助 Petri 网中比较成熟的分析技术对 TCB 中的隐蔽信息流进行分析,并提出一种基于 Petri 网可达图的隐蔽信息流存在性判定算法. 和无干扰方法相比,该算法的主要优点在于避免了等价状态的区分和展开定理的使用,同时可以直接定位隐蔽信息流的位置. 另外,还探讨了导致隐蔽信息流发生的网结构特征,从而使得由此类结构引起的隐蔽信息流在系统的早期设计阶段得以避免. 文中最后还提出可以通过组合分析的方法降低大规模可信系统分析的复杂度,并讨论了各种组合操作下隐蔽信息流的存在性.

文中采用有界 Petri 网为 TCB 建模,因而其可达图也是有限的. 对于具有无穷状态的 TCB,其 Petri 网模型将是无界的,因而其可达图是无穷的. 如何在无穷的空间上判定隐含信息流的存在性是下一步的主要工作. 我们可借助于模型检测技术[22-23]和定理证明器 PVS[24]来完成无穷空间上隐蔽信息流的分析:(1)利用模型检测中的抽象技术[25],在保留全部信息流的前提下,建立 TCB 的有穷状态模型;(2)假设在当前可达空间上隐蔽信息流不存在,利用 PVS 归纳证明在下一步可达空间上隐蔽信息流也不存在.

参 考 文 献

[1] Qing Si-Han. Covet channel analysis in secure operating systems with high security levels. Journal of Software, 2004, 15(12): 1837-1849(in Chinese)
(卿斯汉. 高安全等级安全操作系统的隐蔽通道分析. 软件学报, 2004, 15(12): 1837-1849)

[2] Kemmerer R A. Shared resource matrix methodology: An approach to identifying storage and timing channels. ACM Transactions on Computer Systems, 1983, 1(3): 256-277

[3] Kemmerer R A. Covert flow trees: A visual approach to analyzing covert storage channels. IEEE Transactions on Software Engineering, 1991, 17(11): 1166-1185

[4] Denning D E. A lattice model of secure information flow. Communications of the ACM, 1976, 19(5): 236-243

[5] Tsai C, Gligor V D, Chandersekaran C S. A formal method for the identification of covert storage channels in source code//Proceedings of the IEEE Symposium on Security and Privacy. Oakland, California, USA, 1987: 74-87

[6] Qing Si-Han, Zhu Ji-Feng. Covet channel analysis on Ansheng secure operating system. Journal of Software, 2004, 15(9): 1385-1392(in Chinese)
(卿斯汉,朱继锋. 安胜安全操作系统的隐蔽通道分析. 软件学报, 2004, 15(9): 1385-1392)

[7] Goguen J A, Meseguer J. Security policies and security models//Proceedings of the IEEE Symposium on Security and Privacy. Oakland, California, USA, 1982: 11-21

[8] Zakintihinos A, Lee E S. A general theory of security properties//Proceedings of the IEEE Symposium on Security and Privacy. Oakland, California, USA, 1997:94-102

[9] Jonathan K M. Unwinding forward correctability//Proceedings of the 7th IEEE Computer Security Foundations Workshop. New Hampshire, USA, 1994: 2-10

[10] Yuan Chong-Yi. Principle and applications of Petri net. Beijing: Publishing House of Electronics Industry, 2005(in Chinese)
(袁崇义. Petri 网原理与应用. 北京:电子工业出版社, 2005)

[11] Reisig W. Petri Nets: An Introduction. New York: Spring-Verlag, 1982

[12] Zi Xiao-Chao, Yao Li-Hong, Li Lan. A state-based approach to information flow analysis. Chinese Journal of Computers, 2006, 29(8): 1469-1467(in Chinese)
(訾小超，姚立红，李斓. 一种基于有限状态机的隐蔽信息流分析方法. 计算机学报，2006，29(8)：1469-1467)

[13] var der Meyden R, Zhang C. Algorithmic verification of noninterference properties. Electronic Notes in Theoretical Computer Science, 2007, 168: 61-75

[14] Chen Song, Zhou Cong-Hua, Ju Shi-Guang, Wang Ji. Petri net-based analysis and verification of information flow security properties. Application Research of Computers, 2010, 27(12): 4638-4642(in Chinese)
(陈松，周从华，鞠时光，王基. 基于 Petri 网的信息流安全属性的分析与验证. 计算机应用研究，2010，27(12)：4638-4642)

[15] Murata T. Petri nets: Properties, analysis and application. Proceedings of the IEEE, 1989, 77(4): 541-580

[16] Peterson J L. Petri Net Theory and the Modeling of Systems. New Jersey: Prentice Hall, 1981

[17] Fan Yu-Shun. Techniques for Managing Workflow in Enterprise. Beijing: Tsinghua University Press, 2001(in Chinese)
(范玉顺. 工作流企业管理技术基础. 北京：清华大学出版社，2001)

[18] Zhang Ji-Jun. Compose operation of Petri net control structure. Computer Engineering and Applications, 2005, 41(20): 85-88(in Chinese)
(张继军. Petri 网控制结构的合成运算. 计算机工程与应用，2005，41(20)：85-88)

[19] Pang Shan-Chen, Jiang Chang-Jun, Sun Ping, Zhou Chang-Hong. Property analysis of shared composition Petri nets. Acta Automatica Sinica, 2004, 30(6): 944-948(in Chinese)
(庞善臣，蒋昌俊，孙萍，周长红. 共享合成 Petri 网的性质分析. 自动化学报，2004，30(6)：944-948)

[20] Pu Fei, Lu Wei-Ming. Preservation of liveness and deadlock-freeness in sysnchronous synthesis of Petri net systems. Journal of Software, 2003, 14(12): 1977-1988(in Chinese)
(蒲飞，陆维明. 同步合成 Petri 网系统活性与无死锁性的保持性. 软件学报，2003，14(12)：1977-1988)

[21] Jiang Chang-Jun, Yan Chun-Gang. Research on process characteristics of synchronous composition netws. Acta Electronica Sinica, 1997, 25(2): 57-60(in Chinese)
(蒋昌俊，闫春刚. 同步合成网的进程特性研究. 电子学报，1997，25(2)：57-60)

[22] Clarke E M, Grumberg O, Peled D. Model Checking. MA: MIT Press,1999

[23] Lin Hui-Min, Zhang Wen-Hui. Model checking: theories, techniques and applications. Acta Electronica Sinica, 2002, 30(12A): 9-14(in Chinese)
(林惠民，张文辉. 模型检测：理论、方法与应用. 电子学报，2002，30(12A)：9-14)

[24] Owre S, Rushby J, Shankar N. Integration in PVS: Tables, types, and model checking. Lecture Notes in Computer Science, 1997, 1217: 366-383

[25] Chen Z Y, Zhou C H, Ding D C. Automatic abstraction refinement for petri nets verification//Proceedings of the 10th Annual IEEE International High Level Design Validation and Test Workshop. Napa Valley, CA, USA, 2005: 168-174

ZHOU Cong-Hua, born in 1978, Ph. D., associate professor. His research interests include model checking, access control, modal logic.

JU Shi-Guang, born in 1955, professor, Ph. D. supervisor. His research interests include database security, access control.

Background

This work is supported by the National Nature Science Foundation of China (Nos. 61003288, 6111130184), the Ph. D. Programs Foundation of Ministry of Education of China (No. 20093227110005), and the Natural Science Foundation of Jiangsu Province (No. BK2010192).

The covert information flow is considered as a threat to information leaks. Their detection is considered as an important task that should be automated as much as possible. In recent years there has been an increasing interest in covert information flow analysis. One of the reasons for the increase is that covert information flow analysis is a part of the evaluation criteria used by the National Computer Security Center to classify secure systems.

There are a lot of approaches for covert information flow identification. However, none of them is effective. Specially, the main difficulty in the noninterference approach is that it depends on unwinding theory. The object of the paper is to modify the noninterference approach such that it can avoid using the unwinding theorem. A Petri net based approach to identifying convert channels is proposed in this paper. Its main idea is to model secure systems as safe Petri nets, and analyze the covert information flow from two aspects: structure character of Petri nets and its dynamic behavior. It is always very difficult for analyzing the complex system. In this paper we further propose that the covert information flow in complex system can be analyzed by composition approach. We discuss the effect of some composition patterns including sequence, choice, repetition, etc, for the existence of covert information flow.