基于软件行为的可信评价研究

丁卫涛 徐开勇
(解放军信息工程大学密码工程学院 郑州 450001)

摘 要 为了准确合理地评价软件可信性,提出了基于软件行为的可信评价模型。首先,在软件行为迹中设置监控点,根据监控点各属性的性质及其在可信评价系统中的作用,将监控点的属性分为控制流和数据流两级。其次,针对控制流级属性,提出基于支持向量机(Support Vector Machine,SVM)的软件行为迹的评价方法;针对数据流级属性,提出基于模糊层次分析法的场景属性评价方法。最后,实验分析表明,基于软件行为的可信评价模型能够准确地评价软件可信性,并且具有较高的效率。

关键词 软件行为,支持向量机,模糊层次分析法,监控点,软件可信
中图法分类号 TP309 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.1.045

Research of Trustworthiness Evaluation Model Based on Software Behavior

DING Wei-tao XU Kai-yong
(School of Cryptography Engineering,PLA Information Engineering University,Zhengzhou 450001,China)

Abstract In order to evaluate the trustworthiness of the software accurately and reasonably,a trustworthiness evaluation model based on software behavior was proposed. Firstly,the monitoring points are set up in the software behavior trace. According to the attribute of the monitoring points and the function in the trusted evaluation system,the monitoring points are divided into control flow and data stream. Secondly,for the attribute of control flow,the evaluation method of the software behavior trace based on support vector machine (SVM) is proposed. And for the attribute of data stream,the evaluation method of scene property based on fuzzy AHP is proposed. Finally,the experimental analysis shows that the trustworthiness evaluation model based on software behavior can evaluate the trustworthiness of software accurately and the efficiency.

Keywords Software behavior,Support vector machine,Fuzzy hierarchy analysis,Monitoring points,Software trustworthiness

1 引言

随着软件在金融、军事、政府、经济等国计民生中的应用不断深化,软件系统在生活中的重要性日益突出,软件的可信性要求越来越高。可信计算组织(Trusted Computing Group,TCG)(2007)从实体行为角度对可信进行了定义:"如果一个实体的行为,总是以预期的方式,达到预期的目标,则称其为可信的[1,2]"。软件行为[3]可以看作是一系列的相关动作序列,通过在软件行为中设置一系列的监控点[4],可以对软件行为属性进行检测,因此通过对监控点属性的检测可以实现对软件进行可信性的评价[5]。

在软件行为可信评价方面,国内外许多专家和学者都已经进行了多年的研究。Sekar[6]等总结前人研究成果,提出了基于系统调用的FSA模型,该模型根据系统调用PC值构造出程序的执行路径,通过将软件实际运行路径与软件预期行为轨迹进行对比可判断软件可信性。该模型由于只考虑了系统调用序列的时序关系而没有考虑系统调用的参数信息即具体的场景信息,因此只能检测出那些针对控制流的攻击行为,而无法抵抗那些没有针对控制流的攻击行为,如通过改变系统调用参数来进行攻击的行为。李珍[7]等通过在软件行为轨迹中设置一些监控点,提出了一个基于分级属性的软件监控点可信行为模型,但是未提出针对各级属性进行可信评价的有效方法。傅建明[8]等提出了基于对象的软件行为模型SBO,该模型能够处理不同行为迹的状态合并,可以检测基于控制流的攻击和针对数据语义的攻击,但只对系统对象进行了描述建模,并没有给出具体的可信评价方法。针对上述研究中存在的问题,提出了基于软件行为的可信评价模型。

2 基于软件行为的可信评价模型概述

本文对TCG的信任链进行拓展,引入对应用层软件运行时动态可信的监测,在操作系统和应用软件之间加入度量代理,提出了基于软件行为的可信评价模型。首先将度量代理加入信任链,在操作系统接管系统运行的控制权后,继续度量其二进制哈希值并将结果扩展至PCR;软件运行时,进行基于软件行为的可信评价,度量代理根据动态度量的结果决定软件是否可信。

到稿日期:2015-05-28 返修日期:2015-08-22 本文受国家自然科学基金项目:密码片上系统安全模型结构与验证方法研究(61072047)资助。
丁卫涛(1991—),男,硕士生,主要研究方向为信息安全,E-mail:1975329596@qq.com;徐开勇(1963—),男,博士,研究员,主要研究方向为信息安全、可信计算。

根据可信计算组对软件可信的定义，软件的行为轨迹包括运行轨迹和功能轨迹。运行轨迹从流程角度描述软件轨迹，即表征软件是否按照正确的路径执行，本文称之为控制流(Control Flow)属性；功能轨迹从功能及场景信息的角度描述软件轨迹，即表征软件是否在正确的场景下实现了正确的功能，本文称之为数据流(Data Stream)属性，即通过判断软件控制流和数据流的可信性可以判断软件是否可信。出于性能考虑，将控制流和数据流分开处理，即先判断控制流是否可信，如果可信，再判断数据流是否可信，避免了同时处理时数据量大的问题。

模型在软件行为迹中植入一系列的监控点，通过监控点提取软件运行时的特征数据，根据监控点属性及其在可信评价系统中的层次作用，将监控点属性分为控制流和数据流两类。Control Flow 属性是软件行为序列的抽象表现，以数据结构链的形式表示，用来描述软件行为序列是否按照预定的顺序执行，主要包括监控点的上下文属性等；Data Stream 属性是用来描述监控点的场景信息，这些信息可以用数值表示，如时间戳、CPU 占用比、调用线程数、内存占用比等，分别用来表示当前监控点与上一个监控点之间的时间差、CPU 的占用率、线程调用数目以及内存占用率等。基于软件行为的可信评价流程如图 1 所示，其中左侧部分为一条由软件行为监控点及其转移路径组成的软件预期行为轨迹示意图；对软件运行过程中的一条软件行为进行可信评价的过程如图 1 右侧部分所示，包括 Control Flow 级属性可信评价模型和Data-Stream级属性可信评价模型，其可信评价过程如下。

Step 1　针对 Control Flow 级属性，提出了基于支持向量机(SVM)的软件行为迹评价方法。该评价方法对软件行为序列是否可信进行判断，首先判断一个行为序列是否异常，然后对异常点数目进行统计，若累计异常行为序列数目 n 超过阈值 r，即 $n \geqslant r$，则判定软件行为不可信，并进行报警且软件终止运行；反之即 $n<r$，则判定软件行为可信，并转入 Step2。

Step2　针对 Data Stream 级属性，提出了基于模糊层次分析法(FAHP)的场景属性评价方法。场景级属性可信值 T 设定的取值范围为[0,100]，设定软件可信阈值 δ，即 $\delta<T\leqslant 100$ 时，判定软件可信；$0\leqslant T\leqslant\delta$ 时，判定软件不可信，并进行报警且软件终止运行。

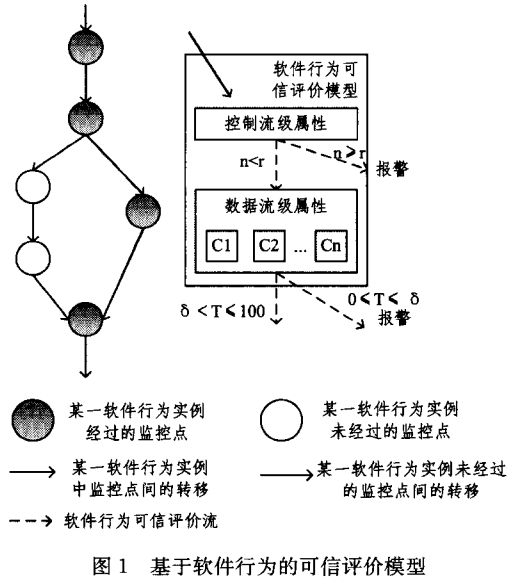

图 1　基于软件行为的可信评价模型

3　软件行为可信评价

3.1　相关定义

定义 1(软件行为，Software Behavior，SB)　软件行为是指软件作为主体运行时对客体实施的操作或者动作。同时，一个软件行为也可以表示为在软件运行中完成某个特定功能的一系列程序语句或系统调用。其形式化表述如下：

$$Behavior=\{behavior=(s)APPLIES(f)TO(obj)\mid s\in Subjects, f\in Functions, obj\in Objects\}$$

其中，*Subjects* 为主体集，*Functions* 为函数集(操作、动作)，*Objects* 为客体集。

定义 2(软件行为迹，Software Behavior Trace，SBT)　软件行为迹是软件行为的表现方式，将软件运行时的行为序列按照时间顺序记录下来，形式化系统调用序列。其形式化表述如下：

$$SBT=\{R_1,R_2,\cdots,R_n\}$$

定义 3(行为信息基，Behavior Information Base，BIB)　对于已知正常或者异常的软件行为轨迹 $R_1,R_2,\cdots,R_n$ 序列，通过对原始数据序列进行相空间重构，即将一维时间序列转化为矩阵形式来寻找数据间的相联关系，建立长度为 M 的滑动窗口进行扫描生成行为 BIB，其形式化描述为二元组$\langle x_i, y_i\rangle$，其中 $x_i\subset R$，$|x_i|=M$；属性变量为

$$y_i\in\begin{cases}0, & \text{异常行为短序列}\\ 1, & \text{正常行为短序列}\end{cases}$$

定义 4(监控点场景，Check Scene，CS)　指预定义的用于在软件运行时进行可信评价的监控点属性，当软件运行至监控点位置时采集该点处的场景信息，包含实时运行的状态信息和环境信息。本文选取 4 个重要的属性信息进行评价，其形式化描述如下：

$CS=\{$时间戳(T)，线程数(N)，*CPU* 负载(*CPUL*)，内存占用(*MemL*)$\}$

定义 5(时间戳，Time Stamp，TS)　在提取分析检测点场景时，提取出软件从上一个监控点到运行到当前监控点时所经历的时间，将其记为该监控点的时间戳(TS_i)。在软件刚运行时令 $TS_1=0$，TS_i 表示第 i 个监控点的场景的时间戳。

定义 6(线程数，Thread Count，TC)　记录在 CPU 进程堆栈排队等待 CPU 时间片的线程数量。

定义 7(CPU 负载，CPU Load，CPUL)　记录一个时间片内 CPU 时钟忙闲的程度。

定义 8(内存占用情况，Memory Load，MemL)　记录软件运行时占用系统内存的比例。

3.2　基于支持向量机的软件行为迹评价

本文采用支持向量机(SVM)[9]方法对软件行为轨迹进行分类。首先用长度为 M 的滑动窗口[10]扫描软件行为序列获得行为信息基(BIB)，然后用支持向量机对信息基进行训练，评断软件行为序列的可信性。

支持向量机[11]的核心思想是寻找一个满足分类要求的最优分类超平面 $\omega\cdot x+b=0$，该超平面在保证将样本准确分为两类的情况下，应使两类样本到最优超平面的最小距离之和最大，即尽可能地将两类样本分离开来。这样寻找最优超平面的问题就转化为求解如下带约束条件的最值问题。

$$\min_{w,b,\xi}\frac{1}{2}w^2+C\sum_{i=1}^{l}\xi_i$$
$$\text{s. t. } y_i((w\cdot x_i)+b\geqslant 1-\xi_i, i=1,\cdots,l \quad (1)$$
$$\xi_i\geqslant 0, i=1,\cdots,l$$

其中，ξ_i 为误差约束条件；C 为惩罚系数。引入 ξ_i、C 用于解决超平面不能正确分类的样本。利用 Lagrange 乘法算子可以把上述寻找最优超平面的问题转化为其对偶问题：

$$\min_{\alpha}\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}y_iy_j\alpha_i\alpha_j[\varphi(x_i)\cdot\varphi(x_j)]-\sum_{j=1}^{l}\alpha_j$$
$$\text{s. t. }\sum_{i=1}^{l}y_i\alpha_i=0 \quad (2)$$
$$0\leqslant\alpha_i\leqslant C; i=1,\cdots,l$$

其中，$\alpha=(\alpha_1,\cdots,\alpha_l)^{\mathrm{T}}$ 为 Lagrange 乘子向量；α_i 为样本点 x_i 对应的 Lagrange 乘子。这是一个典型的二次函数求最优值问题，存在唯一的解。由此可以得到最优权值向量 ω 和最优偏置 b 分别为

$$\omega=\sum_{i=1}^{l}a_iy_ix_i, b=y_j-\sum_{i=1}^{l}y_ja_i(x_i\cdot x) \quad (3)$$

其对应的最优分类决策函数为

$$f(x)=\mathrm{sgn}[(\omega\cdot x)+b]=\mathrm{sgn}[\sum_{i=1}^{l}y_i\alpha_i(x_i\cdot x)+b] \quad (4)$$

其中，$\mathrm{sgn}=\begin{cases}+1, & \text{判断结果为正常}\\ -1, & \text{判断结果为异常}\end{cases}$。

对于线性不可分情况，先利用一个非线性映射 $\varphi: x\to\varphi(x)$ 将非线性样本数据映射到一个高维特征空间中，在高维特征空间中可以实现线性分类，求解最优分类超平面 $\omega\cdot\varphi(x)+b=0$，则对应的求解最优化问题转化为：

$$\min_{\alpha}\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}y_iy_j\alpha_i\alpha_j[\varphi(x_i)\cdot\varphi(x_j)]-\sum_{j=1}^{l}\alpha_j$$
$$\text{s. t. }\sum_{i=1}^{l}y_i\alpha_i=0 \quad (5)$$
$$0\leqslant\alpha_i\leqslant C; i=1,\cdots,l$$

相应的最优分类决策函数为：

$$f(x)=\mathrm{sgn}[\sum_{i=1}^{l}y_i\alpha_iK(x_i\cdot x)+b]$$

其中 $K(x_i\cdot x)=\varphi(x_i)\cdot\varphi(x)$ 为核函数。核函数的选取应使其为特征空间的一个内积，模型采用支持向量机中最常用的 Gauss 径向基函数作为核函数。Gauss 径向基函数为 $K(x\cdot y)=\exp((-\gamma)\|x-y\|)^2$。

基于 SVM 的分类结果只是针对一个系统调用序列的评价，不是对一条完整的软件行为序列的可信评价。异常系统调用序列累计得过多就会造成软件行为不可信。本文设定异常行为系统调用序列个数阈值 r，阈值 r 根据历史行为进行设定，如可统计大量功能发生异常的软件的行为迹的异常点个数，选取其中最小值作为阈值 r。

$$\begin{cases}n\geqslant r, & \text{软件行为不可信，进行报警并终止软件运行}\\ n<r, & \text{软件行为可信，转入 Step2 进行场景评测}\end{cases}$$

其中，n 为软件行为异常序列个数。

3.3 基于模糊层次分析法的场景属性评价

为了了解软件系统当前及未来一段时间的风险发生点，评估这些风险发生时有可能带来的安全问题与损失，就需要进行风险评估[12]。信息系统风险分析和评估可以为安全策略的确定、信息系统的建立及安全运行提供依据[13]，是一个复杂的过程，涉及可能引起系统遭受损失的威胁事件及威胁事件通过资产的脆弱性对资产的危害程度等[14]。因此，风险评估主要依据实体的历史行为，分析并评估其针对资产的威胁事件及产生的严重后果。当一个软件开发完成之后，经过多次正常的训练就能得到该软件正常运行时场景中各属性的取值范围。该软件在实际运行时，对各检查点处监测到的场景信息和正常场景信息按规定的策略进行比对，发现有疑似风险的检查点，风险发生的可能性就越大，当可能发生风险的检查点累积到一定程度时，就可以判断软件行为是否可信。根据风险评估中对风险的表示和文献[15]中 Jøsang A 对风险计算公式的定义：

$$R(e)=P(e)\times L(e) \quad (6)$$

其中，$R(e)$ 表示监控点 e 的风险值，$P(e)$ 表示监控点 e 产生风险的概率，$L(e)$ 表示当监控点 e 处产生风险时所造成的损失。由式(6)知要对软件进行风险评估，首先要确定监控点处产生风险的概率以及当风险发生时所造成的损失。求解风险值 $R(e)$ 的步骤如下。

(1)求解 $P(e)$

本文主要针对场景级属性进行风险评估[16]，选取时间戳、线程数目、CPU 占用比、内存占用比 4 个要素作为分析要素。由于 $P(e)$ 受 CPU 占用比、线程数目、内存占用比、时间戳个这几个属性的影响，并且每个属性影响的权重不相同，通过加权平均法可以确定场景值 $S(e)$。由于各个属性的单位不同，不能直接将其数值用于确定场景值 $S(e)$，需要将数值进行统一处理，即令 $gcpu$、$gthr$、$gmem$、$gtime$ 分别用来表示 CPU 占用比、线程数目、内存占用比、时间戳 4 个属性的当前监控点数值与上个监控点数值的比值，则确定场景值 $S(e)$ 的公式如下。

$$S(e)=gcpu\cdot w_1+gthr\cdot w_2+gmem\cdot w_3+gtime\cdot w_4 \quad (7)$$

其中，w_1, w_2, w_3, w_4 分别表示对应属性的权重。

模型采用 Fuzzy AHP(模糊层次分析法)来求解这 4 个属性的权重[17,18]。求解权重步骤如下：

①求解优先关系矩阵

将 4 个属性根据重要性两两进行比较即可构建优先关系矩阵，属性重要性比较刻度表如表 1 所列。

表 1 属性重要性比较刻度表

FAHP 刻度	定义	说明
A	同等重要	两个风险属性 S_1 和 S_2 的重要性相同
B	轻微重要	S_1 比 S_2 稍微重要一点
C	明显重要	S_1 比 S_2 明显重要
D	重要的多	S_1 比 S_2 重要很多
E	极端重要	S_1 比 S_2 极其重要
F	反比较	如果 S_1 与 S_2 比较得到 r_{ij}，则 S_2 与 S_1 比较得到 $r_{ji}=1-r_{ij}$

根据刻度表得到属性元素 $S_1, S_2, S_3, \cdots, S_n$，对其两两比较得到优先关系矩阵 A 如下：

$$A=\begin{bmatrix}r_{11} & r_{12} & \cdots & r_{1n}\\ r_{21} & r_{21} & \cdots & r_{2n}\\ \cdots & \cdots & & \cdots\\ r_{n1} & r_{n2} & \cdots & r_{nn}\end{bmatrix}$$

②将优先关系矩阵转化为模糊一致矩阵，将优先关系矩阵 A 表示为 $A=(r_{ij})_{n\times n}$，对优先关系矩阵 A 按行求和得到 $r_i=\sum_{j=1}^{n}r_{ij}, i=1,2,\cdots,n$，因此

$$r'_{ij}=(r_i-r_j)/2n+0.5 \quad (8)$$

③求解属性权重

$$\omega_k=\frac{1}{n}-\frac{1}{2a}+\sum_{j=1}^{n}r'_{ij}/(na) \tag{9}$$

其中，$k=1,2,\cdots,n$，$a\geqslant\frac{n-1}{2}$，a 是人们对感知对象的差异程度的一种度量，但与评价对象个数和差异程度有关，当评价的个数或差异程度较大时，a 值可以取得稍微大一点。

根据加权平均公式可以得到监控点 e 处的场景值 $S(e)$，$S(e)$不是一个固定值，经过多次训练可以得到 $S(e)$的取值范围。

④求解风险概率 $P(e)$

监控点 e 处发生风险的概率是受场景值 $S(e)$影响的，其关系如下所示：

如果场景值 $S(e)\in[\min S(e),\max S(e)]$，说明监控点 e 是正常点，在此处不会发生风险，即 $P(e)=0$；

如果场景值 $S(e)\notin[\min S(e),\max S(e)]$，说明监控点 e 是异常点，即有可能在此处发生风险。

当 $S(e)<\min S(e)$时，则

$$P(e)=\min\left(\frac{\min S(e)-S(e)}{\max S(e)-\min S(e)},1\right)$$

当 $S(e)>\max S(e)$时，则

$$P(e)=\min\left(\frac{S(e)-\max S(e)}{\max S(e)-\min S(e)},1\right)$$

(2)评估监控点发生风险造成的损失

监控点发生风险时会对所在模块造成损失，根据模块的重要性即可评价此监控点发生风险造成的损失。为了获得一个比较客观的结果，通过模块运行的时间来衡量该模块的价值。计算方法如下：

$$L(e)=E(e)/E_{\max} \tag{10}$$

其中，$E(e)$是监控点 e 对应模块价值的运行时间的占用比；$E_{\max}$是所有模块中最大的运行时间占用比。

由于各个模块在状态转移时符合 Markov Model[19]，通过 Markov Model 可以计算每个模块的运行时间。

(3)求解风险值

根据得到的 $P(e)$和 $L(e)$，通过式(6)可以求得 $R(e)$。

(4)评估可信性

目前，关于风险和可信的相关研究已经很多，但是风险和可信却没有一个统一的公式[20]。本文设定 θ 是风险阈值常数，T_{old}是上个监控点的可信值，T_{new}是当前监控点 e 的可信值。$R(e)$是监控点 e 的风险值。在本文中，$T\in[0,100]$，T 的初始值是 50。如果 $R(e)\in[0,\theta]$，监控点 e 是一个低风险的监控点，则 T_{new}的计算公式如下：

$$T_{new}=\min(T_{old}+w,100) \tag{11}$$

其中，w 是可信值增加的部分。

如果 $R(e)\in[\theta,1]$，监控点 e 是一个高风险的监控点，则 T_{new}的计算公式如下：

$$T_{new}=T_{old}\cdot\frac{\theta}{R(e)} \tag{12}$$

当监控点可信值累积到一定程度时，可判断软件行为的可信性。设定可信阈值 δ，当 $T\in[0,\delta]$时判定软件行为不可信；当 $T\in[\delta,100]$时判定软件行为可信。阈值 δ 可根据软件历史行为进行设定，如可计算大量功能发生异常监控点场景的可信值，选取其中的最大值作为阈值 δ。

4 实验分析

为了验证所提方法的有效性，本文设计了软件行为可信评价模型比较实验，采用处理器为 Intel Core 2 Duo CPU E8400 3GHz、内存为 2GB 的计算机，操作系统为 Ubuntu 10.04，其内核版本为 Linux-2.6.38.2。针对基于支持向量机的软件行为迹评价模型，利用美国新墨西哥大学(UNM)公开发布的 Sendmail 人工数据集[21]进行实验。该数据集通过人为运行 Sendmail 尽可能全面的功能而生成，类似于软件功能测试人员测试 Sendmail 时生成的真实轨迹。本实验采用其中 185 个行为迹实验数据作为训练数据进行相关实验，其中正常的行为迹有 169 个，包含 16 个攻击行为的迹。检测数据选取 Sendmail 程序中另外的 157 个行为迹，其中包含 13 个攻击行为迹，判断这 157 个行为迹是否异常，根据判断正确的个数计算其正确率。出于性能和精确度两方面考虑，选择长度为 7 的滑动窗口对原始数据进行处理，得到软件行为信息基。算法在 Matleb2012 V7.0 上实现，与其它分类算法的比较结果如表 2 所列。徐婵等提出了基于 BP 神经网络的软件行为评估系统[22]，其判断准确率仅为 93.25%，远低于本文所提方法的 98.45%。即可以看出针对软件行为序列可信问题，本文基于支持向量机(SVM)的方法的分类准确率比较高，而且分类速度比较快。

表 2 支持向量机与其他分类算法的对比

检测方法	训练时间/s	测试时间/s	准确率/%
贝叶斯	2.432	0.925	85.36
神经网络	3.627	0.625	90.65
支持向量机(SVM)	1.532	0.275	98.45

根据异常点个数判断软件行为序列是否可信。当判定软件行为序列不可信时，即说明软件行为不可信；当判定软件行为序列可信时，即控制流可信，需要再对数据流进行判定才能判断软件行为是否可信。

针对基于模糊层次分析法的场景属性评价方法，首先通过模糊层次分析法(FAHP)评估属性 $gcpu$、$gthr$、$gmem$、$gtime$ 对应的权重。由于采用数值刻度(0.1～0.9)比较均匀而且容易评估，算法选用 0.1～0.9 刻度作为模糊层次分析法的评估值，其对应关系如表 3 所列。

表 3 刻度取值对照表

刻度	A	B	C	D	E	F
取值	0.5	0.6	0.7	0.8	0.9	0.1,0.2,0.3,0.4

根据各属性重要性及刻度取值对应表得到其优先关系矩阵为

$$A=\begin{bmatrix}0.5 & 0.9 & 0.7 & 0.6\\ 0.1 & 0.5 & 0.8 & 0.6\\ 0.3 & 0.2 & 0.5 & 0.7\\ 0.4 & 0.4 & 0.3 & 0.5\end{bmatrix}$$

然后根据式(9)计算各属性对应权重 w_1，w_2，w_3，w_4 分别为 0.47，0.28，0.18，0.07。

根据 Markov Model 计算 4 个场景属性对应的模块 A，B，C，D 在任务中所占时间比例为 20%，30%，40%，10%，由式(10)计算其模块价值为 0.5，0.75，1，0.25。

设定初始可信值和相关的参数：$T_0=50$，$\theta=0.5$，$\omega=2$；分别对 12 个监控点进行评价，根据式(11)、式(12)分别求解各

监控点的风险值和可信值，如表 4 所列，监控点对应可信值的变化如图 2 所示。

表 4　监控点处的风险值与可信值

监控点	风险值	可信值
e1	0.355	52
e2	0.35	54
e3	0.405	56
e4	0.525	53.33
e5	0.375	55.33
e6	0.475	57.33
e7	0.55	52.12
e8	0.35	54.12
e9	0.425	56.12
e10	0.45	58.12
e11	0.25	60.12
e12	0.575	52.28

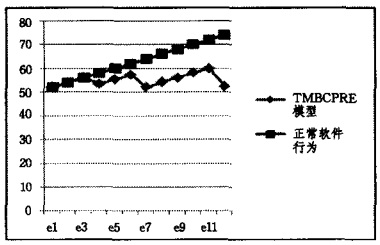

图 2　正常软件行为与异常行为折线对比

由图 2 可知，正常的监控点对软件行为可信值的提高比较缓慢，而异常的监控点对软件行为的可信值下降比较明显，该模型可以准确合理地对软件行为可信性进行评价。

结束语　本文在对软件行为可信度量进行了研究后，提出了基于软件行为的可信评价模型，根据监控点各属性的性质和作用将其分为控制流和数据流，分别对这两级属性进行可信评价，以判断软件的可信型。根据控制流属性的性质，采用滑动窗口对软件行为迹进行扫描，生成软件行为信息基，提出了通过统计学中的支持向量机算法进行可信评价的方法；根据数据流的性质，选取检测场景中 CPU 占用比、线程数目、内存占用比、时间戳等 4 个重要的属性，提出了基于模糊层次分析法的评价模型。通过实验证明，该可信评价模型可以有效快速地评价软件可信性。

参 考 文 献

[1] Rotondo S A. Trusted Computing Group[M]//Encyclopedia of Cryptography and Security. 2011:1331-1331

[2] Shen Chang-xiang, Zhang Huan-guo, Wang Huai-min, et al. Research and development of trusted computing [J] Chinese Science: Information Science, 2010, 40(2): 139-166(in Chinese)
沈昌祥，张焕国，王怀民，等. 可信计算的研究与发展[J]. 中国科学：信息科学，2010，40(2)：139-166

[3] Qu Yan-wen. Software Behavior[M]. Beijing: Publishing House of Electronics Industry, 2004(in Chinese)
屈延文. 软件行为学[M]. 北京：电子工业出版社，2004

[4] Shen Guo-hua, Huang Zhi-qiu, Qian Ju, et al. Research on Software Trustworthiness Evaluation Model and Its Implementation [J]. Journal of Frontiers of Computer Science and Technology, 2011, 5(6): 553-561(in Chinese)
沈国华，黄志球，钱巨，等. 软件可信评估模型及其工具实现[J]. 计算机科学与探索，2011，5(6)：553-561

[5] Huang Jian-feng. Journal of Computer Research and Development[J]. Electronics World, 2014(16): 374-374(in Chinese)
黄见峰. 基于马尔可夫的软件可信评估模型研究[J]. 电子世界，2014(16)：374-374

[6] Sekar R, Bendre M, Dhurjati D, et al. A fast automaton-based method for detecting anomalous program behaviors[C]//2001 IEEE Symposium on Security and Privacy(S&P 2001). IEEE, 2001: 144-155

[7] Li Zhen, Tian Jun-feng, Yang Xiao-hui. Dynamic Trustworthiness Evaluation Model of Software Based on Checkpoint's Classification Attributes[J]. Journal of Computer Research and Development, 2013, 50(11): 2397-2405(in Chinese)
李珍，田俊峰，杨晓晖. 基于检查点分级属性的软件动态可信评测模型[J]. 计算机研究与发展，2013，50(11)：2397-2405

[8] Fu J M, Tao F, Wang D, et al. Software behavior model based on system objects[J]. Journal of Software, 2011, 22(11): 2716-2728(in Chinese)
傅建明，陶芬，王丹，等. 基于对象的软件行为模型[J]. 软件学报，2011，22(11)，2716-2728

[9] Zhang Xue-gong. Introduction To Statistical Learning Theory And Support Vector Machines[J]. Acta Automatica Sinica, 2000, 26(1): 32-42(in Chinese)
张学工. 关于统计学习理论与支持向量机[J]. 自动化学报，2000，26(1)：32-42

[10] Luo Qing-hua, Yan Xiao-zhen, Peng Yu, et al. A dynamic RSSI-based ranging method using pattern matching of slide window [J]. Chinese Journal of Scientific Instrument, 2015, 36(3): 499-506(in Chinese)
罗清华，焉晓贞，彭宇，等. 基于滑动窗口模式匹配的动态距离估计方法[J]. 仪器仪表学报，2015，36(3)：499-506

[11] Peng N. A SVM reliability evaluation model for component-based software systems[C]//2013 2nd International Symposium on Instrumentation and Measurement, Sensor Network and Automation (IMSNA). IEEE, 2013: 704-708

[12] Pirzadeh H, Hamou-Lhadj A. A software behaviour analysis framework based on the human perception systems: NIER track [C]//2011 33rd International Conference on Software Engineering (ICSE). IEEE, 2011: 948-951

[13] Söderström O, Moradian E. Secure Audit Log Management[J]. Procedia Computer Science, 2013, 22: 1249-1258

[14] Chen Song, Wang Guang-wei, Liu Xin-yu, et al. Research on the Evaluation of Information System Security[J]. Communications Technology, 2012, 45(1): 128-130(in Chinese)
陈颂，王光伟，刘欣宇，等. 信息系统安全风险评估研究[J]. 通信技术，2012，45(1)：128-130

[15] Jøsang A, Bradley D, Knapskog S J. Belief-based risk analysis [C]//Proceedings of the second workshop on Australasian Information Security, Data Mining and Web Intelligence, and Software Internationalisation-Volume 32. Australian Computer Society, Inc., 2004: 63-68

[16] Tian J F, Zhu Y. Trusted Software Construction Model Based on Trust Shell[J]. Advanced Materials Research, 2011, 186: 251-255

[17] Guo Jin-yu, Zhang Zhong-bin, Sun Qing-yun. Study and Application of Analytic Hierarchy Process[J]. China Safety Science Journal, 2008, 18(5): 148-153(in Chinese)
郭金玉，张忠彬，孙庆云. 层次分析法的研究与应用[J]. 中国安全科学学报，2008，18(5)：148-153

[18] Shi L, Yang S. The evaluation of software trustworthiness with FAHP and FTOPSIS methods[C]//International Conference on Computational Intelligence and Software Engineering, 2009 (CiSE 2009). IEEE, 2009: 1-5

(下转第 225 页)

达性等特性。但由于采用库所和变迁作为基本描述元素，使得其本身的节点数比描述同一系统的 FSM 的节点数多。CPN 通过颜色集进一步增强了 Petri 网的描述能力，并且 CPN Tools 可以基于建立的模型生成状态空间，进而检验模型是否有活锁、死锁以及可达性等。Petri 网的可达性是通过 marking(标记)构成，当系统中有并发行为时，极有可能导致状态空间爆炸。

本文的测试方法是基于 CPN 进行测试推导研究的，并且采用 TTCN-3 进行了测试例的实现，并设计了测试方案进行测试执行。由于本文的主要工作是针对并发系统进行 CPN 建模，并有模型化简的考虑，因此在建模初期就对模型的规模有所控制，然后在并发覆盖测试序列生成时针对模型状态空间进行研究，此时由于令牌的选取，使得状态空间形成的状态图已经成为一个确定的图。本文在建模时针对并发行为测试问题进行了建模的优化考虑；在测试序列生成时，先生成覆盖并发行为的测试子序列，再构成完整测试序列，从而避免了大量冗余序列的生成。因此，本文得到了既满足并发行为覆盖要求，又减少了测试序列数目的测试例。

当系统中的并发行为很多时，仍会导致状态空间的爆炸，因此下一步的研究工作是考虑如何能尽量地化简 CPN 模型；另外，令牌的选取、变迁点火时间的限制方面也需要深入研究。

参 考 文 献

[1] Lee D, Yannakakis M. Principles and methods of testing finite state machines-a survey[J]. Proceedings of the IEEE, 1996, 84(8): 1090-1123

[2] Tretmans J. A formal approach to conformance testing[D]. University of Twente, Enschede, Netherlands, 1992

[3] Tretmans J. Test generation with inputs, outputs and repetitive quiescence [J]. Software Concepts and Tools, 1996, 17(3): 103-120

[4] Petrenko A, Yevtushenko N, Huo J L. Testing transition systems with input and output testers[J]. Testing of Communicating Systems, Springer Berlin Heidelberg, 2003, 2644 (0302-9743): 129-145

[5] Peterson J L. Petri net theory and the modeling of systems [M]. Englewood Cliffs, Nj, Prentic-Hall, Inc. 1981

[6] Wu Z H. Introduction of Petri Nets[M]. Beijing: Mechanical Industry Press, 2006(in Chinese)
吴哲辉. Petri 网导论[M]. 北京：机械工业出版社，2006

[7] Jensen K. Coloured Petri Nets: Basic concepts, analysis methods and practical use. Vol. 3[M] // Practical use, Monographs in Theoretical Computer Science. Springer, 1997

[8] Jensen K, Kristensen L M. Coloured Petri Nets: modelling and validation of concurrent systems[M]. Spring-Verlag, 2009

[9] Wang G, Wu J, Xu L, et al. Research on test adapter framework for distributed TTCN-3 test execution platform[J]. Acta Electronica Sinica, 2009, 37(1): 125-130(in Chinese)
王冠，吴际，徐珞，等. 面向 TTCN-3 分布式测试执行平台的测试适配器框架的研究与设计[J]. 电子学报，2009，37(1)：125-130

[10] Lei Y, Carver R H. Reachability testing of concurrent programs [J]. IEEE Transactions on Software Engineering, 2006, 32(6): 382-403

[11] Sen K, Marinov D, Agha G. CUTE: A concolic unit testing engine for C[C] // Proceedings of the 13th ACM SIGSOFT Symposium on Foundations of Software Engineering jointly with 10th European Software Engineering Conference. Lisbon, Portugal, ACM Press, 2005

[12] Merz S. Model Checking: A tutorial overview[C] // Proceedings of the 4th Summer School on Modeling and Verification of Parallel Processes. Nantes, France, Springer Press, 2000: 3-38

[13] Edelstein O, Farchi E, Nir Y, et al. Multithreaded java program test generation[J]. IBM Systems Journal, 2002, 41(1): 111-125

[14] Remenska D, Templon J, Willemse T A C, et al. From UML to process algebra and back: An automated approach to model-Checking software design artifacts of concurrent systems[M] // NASA Formal Methods. Springer Berlin Heidelberg, 2013: 244-260

[15] Yang H B, Li Y P. Functional test scenarios generation method based on UML activity diagrams[J]. Computer Engineering, 2011, 37(21): 55-57(in Chinese)
杨鹤标，李云平. 基于 UML 活动图的功能测试场景生成方法[J]. 计算机工程，2011，37(21)：55-57

[16] Farooq U, Lam C P, Li H. Towards automated test sequence generation[C] // Proceedings of the 19th Australian Conference on Software Engineering. Perth, Australia, 2008: 441-450

[17] Din G, Tolea S, Schieferdecker I. Distributed load tests with TTCN-3[M] // Testing of Communicating Systems: TestCom 2006, LNCS 3964, 2006. IFIP International Federation for Information Processing, 2006: 177-196

(上接第 206 页)

[19] Tian Jun-feng, Zhang Ya-jiao. Checkpoint trust evaluation method based on Markov[J]. Journal on Communications, 2015, 36(1): 230-236(in Chinese)
田俊峰，张亚姣. 基于马尔可夫的检查点可信评估方法[J]. 通信学报，2015，36(1)：230-236

[20] Hu Yong. Research on the Evaluation of Information System Security[D]. Chengdu: Sichuan University, 2007(in Chinese)
胡勇. 网络信息系统风险评估方法研究[D]. 成都：四川大学，2007

[21] Forrest S, Hofmeyr S A, Somayaji A, et al. A sense of self for unix processes[C] // Proceedings of IEEE Symposium on Computer Security and Privacy. 1996: 120-128

[22] Xu Chan, Liu Xin, Wu Jian, et al. Software Behavior Evaluation System Based on BP Neural Network[J]. Computer engineering, 2014, 40(9): 149-154(in Chinese)
徐婵，刘新，吴建，等. 基于 BP 神经网络的软件行为评估系统[J]. 计算机工程，2014，40(9)：149-154