doi:10.11959/j.issn.1000-436x.2016265

基于业务过程挖掘的内部威胁检测系统

朱泰铭[1,2]，郭渊博[1,2]，琚安康[1,2]，马骏[1,2]
（1. 解放军信息工程大学网络空间安全学院，河南 郑州 450001；
2. 数学工程与先进计算国家重点实验室，江苏 无锡 214000）

摘 要：当前的入侵检测系统更多针对的是外部攻击者，但有时内部人员也会给机构或组织的信息安全带来巨大危害。现有的内部威胁检测方法通常未将人员行为和业务活动进行结合，威胁检测率有待提升。从内部威胁的实施方和威胁对系统业务的影响这 2 个方面着手，提出基于业务过程挖掘的内部威胁检测系统模型。首先通过对训练日志的挖掘建立系统业务活动的正常控制流模型和各业务执行者的正常行为轮廓，然后在系统运行过程中将执行者的实际操作行为与预建立的正常行为轮廓进行对比，并加以业务过程的控制流异常检测和性能异常检测，以发现内部威胁。对各种异常行为进行了定义并给出了相应的检测算法，并基于 ProM 平台进行实验，结果证明了所设计系统的有效性。

关键词：内部威胁；过程挖掘；行为轮廓；异常检测
中图分类号：TP391 **文献标识码**：A

Business process mining based insider threat detection system

ZHU Tai-ming[1,2], GUO Yuan-bo[1,2], JU An-kang[1,2], MA Jun[1,2]
(1. School of Cyberspace Security, PLA Information Engineering University, Zhengzhou 450001, China;
2. State Key Laboratory of Mathematical Engineering and Advanced Computing, Wuxi 214000, China)

Abstract: Current intrusion detection systems are mostly for detecting external attacks, but sometimes the internal staff may bring greater harm to organizations in information security. Traditional insider threat detection methods often do not combine the behavior of people with business activities, making the threat detection rate to be improved. An insider threat detection system based on business process mining from two aspects was proposed, the implementation of insider threats and the impact of threats on system services. Firstly, the normal control flow model of business activities and the normal behavior profile of each operator were established by mining the training log. Then, the actual behavior of the operators was compared with the pre-established normal behavior contours during the operation of the system, which was supplemented by control flow anomaly detection and performance anomaly detection of business processes, in order to discover insider threats. A variety of anomalies were defined and the corresponding detection algorithms were given. Experiments were performed on the ProM platform. The results show the designed system is effective.

Key words: insider threat, process mining, behavior profile, anomaly detection

1 引言

信息技术的高速发展促进了信息系统在各类企业和组织中的广泛应用。然而，信息系统在为这些组织带来工作效率提升的同时，也引入了大量的信息安全漏洞，其中，既有技术层面上的软硬件漏洞，也有来自于组织人员管理上的漏洞。相比因软硬件漏洞招致的外部网络攻击，由于组织内部人员管理漏洞造成的内部威胁往往危害更大，也更难被察觉。造成内部威胁的原因主要

收稿日期：2016-09-15
基金项目：国家自然科学基金资助项目（No.61602515）
Foundation Item: The National Natural Science Foundation of China (No.61202515)

有以下几方面：1) 部分缺乏安全意识的员工在工作时可能做出违反安全规定的误操作；2) 部分员工在工作时为了自身方便，提高效率，故意绕过安全措施进行操作；3) 个别员工因受到他人利诱或对组织采取报复行动，对组织的机密信息进行外泄或破坏。总地来说，内部威胁是一个涉及了人为因素和系统因素的综合性问题，检测和防御内部威胁成为了组织管理者面临的巨大挑战。

为提高工作效率，越来越多的企业和组织开始采用各类业务系统来完成业务活动。然而，大多数的业务系统在设计之初通常只考虑如何保证业务功能的正常实现，很少考虑业务活动的安全性，从而十分容易遭受来自内部人员的有意或无意的威胁，使业务系统出现异常。因此，本文尝试从业务活动的角度，通过综合分析业务执行过程中出现的异常情况和业务执行者的异常工作行为来检测内部威胁。

实际的业务活动包含了多重因素，因此相关的过程模型也必须是多维的，不仅要能反映业务活动之间的顺序信息，还要能反映各业务活动执行者的行为信息、业务案例的特征信息以及业务活动在时间频率方面的信息等。为达到上述目标，大多数企业和组织采取基于日志的过程挖掘方法。采用该方法的原因有以下几点：1) 业务系统日志取用方便，采用系统日志作为数据源是对系统运行影响程度最小的方法之一；2) 日志中详细记录了业务系统的运行情况，便于管理者了解真实的业务活动过程和细节；3) 基于日志来挖掘业务过程使建模过程更加客观和高效。目前的许多过程挖掘研究[1~9]仅从控制流角度建立业务过程模型，很少有研究能结合人员行为角度进行考虑，使模型不能全面地反映业务活动的真实信息，也无法支持业务活动中人员异常行为的检测。本文将人员行为角度纳入到业务过程挖掘中，提出一种基于业务过程挖掘的内部威胁检测系统，如图 1 所示。该系统收集业务活动在正常执行时的事件日志，用于正常业务过程模型的挖掘，然后将正常模型与业务活动的实际执行过程进行对比，检测业务活动的控制流异常和操作人员的行为异常，帮助管理人员及时发现内部威胁。本文利用 ProM 平台[10]和 Java 编程进行了实验，实验结果表明该系统能有效检测出业务活动中存在的内部威胁。

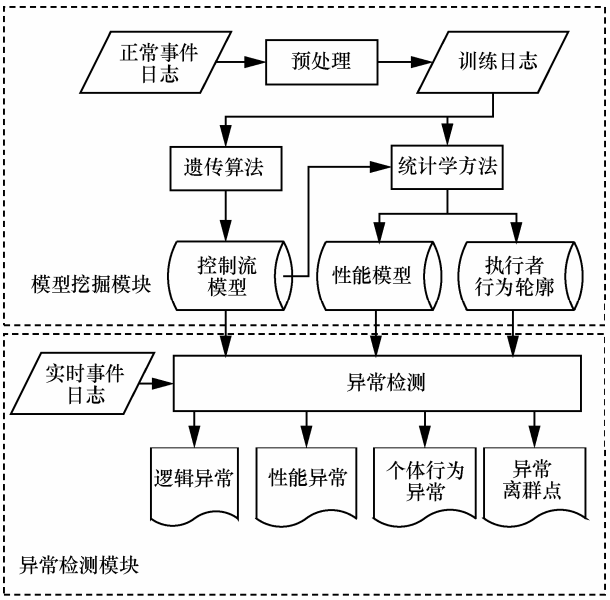


图 1　基于业务过程挖掘的内部威胁检测系统

2　相关研究

本节从内部威胁检测和业务过程挖掘这 2 个方面介绍相关研究成果。

内部威胁一直以来都是危害企业和组织信息安全的重要因素。早在 2000 年，文献[11]就提出了研究这一问题所面临的各类挑战，进一步引发了人们对内部威胁的关注和探索。Spitzner[12]采用蜜罐技术来诱捕内部攻击，然而随着攻击者的手法越来越趋向隐蔽和高级，人们也开始需要更加先进的预防与检测手段。文献[13]采用 RBAC 方法建立用户行为规则，通过发现违反规则的异常行为来检测内部威胁。文献[14]通过关注人员、数据的一般特征和基于上下文的内部威胁建模对 RBAC 模型进行了扩展。此外，Greitzer 等[15]将传统安全审计数据与社会心理数据相结合，使人们不仅可以检测内部威胁，还可以感知并预测潜在的内部威胁发生倾向，并且提出了一个框架，用于整合并分析组织内部数据与网络安全数据，预测可能的内部威胁。与此类似，Brdiczka 等[16]提出了一个将结构性异常检测(SA)和个人心理轮廓(PP)相结合的内部威胁检测方法，最终通过 SA 和 PP 的融合与结果排序来确认内部威胁。Parveen 等[17]采用流数据挖掘和图挖掘方法进行内部威胁检测，并在文献[18]中进行了扩展。本文提出的检测系统不仅关注内部人员的异常行为，还通过检测业务活动本身的异常情况来帮助管理者评估内部威胁对业务系统造成

的影响。

业务过程挖掘是从工作流管理领域中发展出来的一种重构业务过程的技术，目前已有不少研究者提出了相关的挖掘算法。值得一提的是，Aalst 等在文献[2]中给出了完整的业务过程建模过程应当考虑的 3 个方面，即控制流视角、组织人员视角和案例属性视角。其中，控制流视角关注业务活动之间的先后顺序，组织人员视角关注业务的具体参与者以及他们各自负责哪些业务活动，案例属性视角关注业务活动的整体属性，包括了业务的执行过程、参与者以及数据元素之间的关系等因素。然而，目前的大多数过程挖掘方法仅关注控制流层面的挖掘。文献[3]提出了 α 算法，通过发现活动间的二元关系构建控制流结构，但该算法不能处理非局部选择结构。文献[4]提出的 α^{++}算法是对 α 算法的扩展，但不能处理日志噪声。文献[5,6]提出的启发式挖掘算法考虑了活动间跟随关系的频率，因此该算法能够处理日志噪声，但是不能处理非局部选择结构和重复活动。文献[7,8]提出的两阶段挖掘算法，先在日志层建立活动的二元关系模型，然后再将这些模型合并形成顺序和选择结构，但该算法不能挖掘循环结构。文献[9]提出的遗传算法能够解决日志中存在的重复活动、不可见活动以及噪声记录和不完备记录等问题，较好地弥补了上述算法的缺陷。本文首先采用文献[9]中的算法挖掘出业务的控制流模型，然后进一步结合控制流模型和系统日志，为业务执行者建立正常行为轮廓，以此来检测业务活动中可能出现的内部威胁。

3 系统设计与工作原理

3.1 日志预处理

作为过程挖掘的数据来源，事件日志的完整性和正确性直接关系到挖掘结果的准确性，因此，在过程挖掘之前需要对日志进行预处理。首先给出相关概念的定义，然后介绍日志的预处理过程。

定义 1 事件

一个事件 E 是一个六元组，E = (*caseID*, *taskname*, *type*, *operator*, *timestamp*, *extraInfo*)，其中，*caseID* 为该事件所属的业务实例标识，*taskname* 为任务名，*type* 为该任务的执行状态，*operator* 为任务执行者，*timestamp* 为事件的发生时间，*extraInfo* 为其他信息。

定义 2 事件序列

一个事件序列 *ES* 是某一业务在一次执行过程中产生的，并根据产生时间的先后顺序组成的事件集合，$ES=(E_1, E_2, \cdots, E_n)$。其中，$E_1$ 为序列的起始事件，E_n 为序列的终止事件，且满足

$$\forall E_i, E_j \in ES(i<j),\ E_i.timestamp \leqslant E_j.timestamp \quad (1)$$

定义 3 事件日志

一个事件日志 *EL* 是若干事件序列 *ES* 的集合，$EL=(ES_1, ES_2, \cdots, ES_n)$。

日志预处理的第一步是统计收集到的事件日志的整体信息，如该日志中包含了多少种业务活动，每种业务活动有多少对应的事件序列，各事件的出现次数、所占比例以及业务执行者的名称、数量和所参与的事件等信息。这些统计信息可以对该事件日志的整体情况有所了解，进而方便下一步对日志的过滤和筛选。接着，由于要挖掘的是某业务的完整过程模型，因此，只需要该业务被完整执行后产生的事件序列。为此，可根据事件的 *caseID* 对事件序列进行筛选，剔除与所选业务种类无关的事件序列；然后，在剩下的事件日志中筛选出完整的事件序列，即序列具有合法的开始和结束事件，且每一个中间事件都被完成。这些经过预处理的日志称为训练日志。

3.2 业务过程挖掘

3.2.1 挖掘控制流模型

为了从给定的事件序列中挖掘出最为优化的过程模型，选择一个合适的算法十分关键。通常可以采取局部策略或全局策略来衡量一个模型的优化程度[9]。局部策略一般采取逐步方式来建立模型，每进行一步时，通过局部信息决策出最优化的下一步动作。全局策略则通过全局信息进行综合判断，让算法挖掘出全局上最优化的模型，遗传算法就是使用全局策略的典型代表。此外，遗传算法能够解决更多问题，比如日志中存在的重复活动、不可见活动以及噪声记录和不完备记录等。因此，本文采用遗传算法进行控制流模型挖掘，其主要步骤如图 2 所示。

算法大致分为 4 个阶段：初始化、选择、繁殖和终止。初始化阶段用于建立初始种群，每一代种群中可能有成百上千的个体，一个个体指一个过程模型。在建立初始种群时，采用随机化方法，使用训练日志中出现的事件名称进行随机组合，产生大

量随机的个体。由于种群数量巨大，可能产生少量与真实事件序列大体相符的个体。在选择阶段中，算法使用适应度函数计算每个个体的适应度，即综合衡量各过程模型的完整性和准确性。然后，将具有最高适应度的个体直接放入下一代种群，其余的个体则通过竞赛选出用于产生下一代种群的父代个体，剩余的低适应度个体则被丢弃。在繁殖阶段中，使用选出的父代通过交叉和变异来创造新种群。在交叉中，将父代个体两两配对，得到一个“子模型”池，其中的子模型分享父代遗传材料的部分。随后，使用变异操作修改得到的子模型，如随机添加或删除一个因果依赖，这样可以将新的遗传材料插入到下一代种群中。变异的存在使每一代种群得以不断演化。重复上述过程，直到得到的个体满足某些预设条件后，算法结束，返回适应度最好的模型。

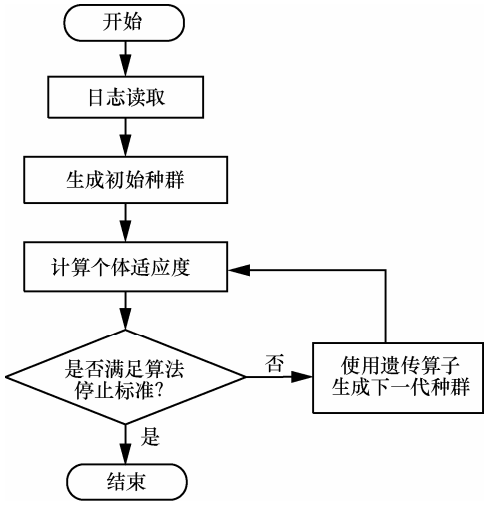


图 2　遗传算法主要步骤

在挖掘出控制流模型后，需要根据专家知识确定模型中对时间和频次有特定要求的节点，统计相应的时间和频次信息，并维护一张“时间—频次约束表”，作为检测业务性能异常的依据。

3.2.2　挖掘执行者正常行为轮廓

通过对训练日志的统计分析和对控制流模型的观察，可以方便地确定业务过程中有哪些执行者，并确定它们应分别执行哪些任务，以及这些任务之间的先后顺序、时间间隔和执行次数等。在确定角色时，将执行相同任务集的不同执行者认定为同一角色。此外，为同时从逻辑层和表现层为执行者建立行为轮廓，可以通过事件中的 *extraInfo* 字段得到更为详细的信息作为补充，如该事件的发生涉及到哪些设备以及设备中的哪些软件、文件或数据，任务执行者对这些设备、软件、文件或数据在什么时间进行了哪些操作等。采用一棵多叉树表示一个执行者的正常行为轮廓，其实例结构如图 3 所示。其中，树根为“*caseID-operator*”，表示该执行者在特定业务案例下的正常行为轮廓。从树根到树叶按层次划分，各层的节点依次为角色，任务，设备，软件、文件与数据，操作。

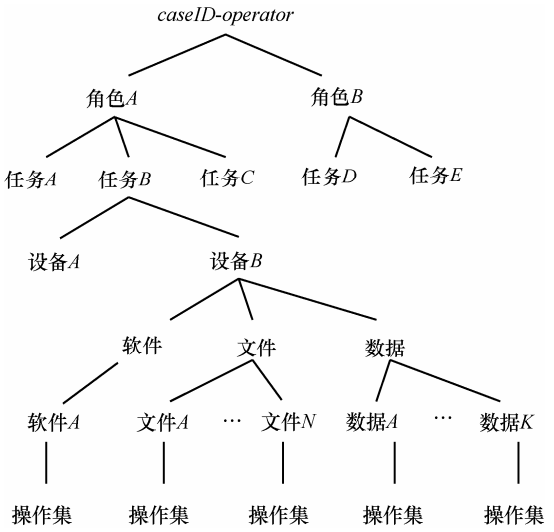


图 3　执行者树状行为轮廓实例

1) 角色：包含了该执行者担任的所有角色。

2) 任务：根据角色，列出该执行者需要执行的所有任务。

3) 设备：根据任务，列出该执行者执行任务时需要操作的所有设备。

4) 软件、文件与数据：根据设备，列出该执行者需要对设备中哪些软件、文件或数据进行操作。

5) 操作：根据设备以及其中的软件、文件和数据，列出该执行者需要对它们执行的具体操作及其频率。

从树根沿着某一条路径一直到叶子节点，可以很方便地得到执行者在执行某些任务时应当进行的操作集合，以及这些操作的正常频率范围。为方便后续的计算过程，用行为向量表示这些操作及其频次范围。

定义 4　行为向量

一个行为向量 ***BV*** 是表示某执行者在执行某项任务时对某设备进行的操作及其频率的多元组，$\boldsymbol{BV}(operator, task, device)=(f(op_1), f(op_2), \cdots, f(op_n))$，其中，*operator* 表示执行者，*task* 表示执行的任

务，*device* 表示设备，$f(op_i)$表示操作 op_i 的频率，且满足

$$f(op_i)=\frac{\#(op_i)}{\sum_{1}^{n}\#(op_i)} \tag{2}$$

其中，$\#(op_i)$表示操作 op_i 的出现次数。

特别地，称 $\boldsymbol{BV}_{\mathrm{N}}$ 为正常行为轮廓中的行为向量，称 $\boldsymbol{BV}_{\mathrm{E}}$ 为实际执行时的行为向量，以示区分。

3.3 异常检测

3.3.1 执行者行为异常检测

在 3.2.3 节中，挖掘出了执行者的个人正常行为轮廓，从而可以通过将执行者的当前行为与其个人正常行为轮廓进行纵向对比，检测出异常行为。然而，当组织的业务执行情况发生正常变化时，各角色对应的任务和操作集合也会随之变化，如果仅依靠纵向对比结果便有可能产生误报。考虑到具有相同角色的执行者的行为具有高度相似性，可以将属于同一角色的执行者的实际行为进行横向比较，通过发现行为的离群点来进一步检测潜在的恶意执行者。

定义 5 个体异常行为

个体异常行为是指行为向量 $\boldsymbol{BV}_{\mathrm{E}}$ 中有 $\boldsymbol{BV}_{\mathrm{N}}$ 中不存在的行为，或 $\boldsymbol{BV}_{\mathrm{E}}$ 与 $\boldsymbol{BV}_{\mathrm{N}}$ 中同一行为的频率之差绝对值大于阈值 σ。

定义 6 离群异常行为

离群异常行为是指经过聚类之后，某些执行者当前的行为向量严重偏离同角色其他执行者的情况。

算法 1 检测个体异常行为的算法

Individual-Behavior_AD(Normal Profile, Execution Log)

1) for each *T* and *D*
2) for each $\boldsymbol{BV}_{\mathrm{E}}$ in Execution Log
3) find the corresponding $\boldsymbol{BV}_{\mathrm{N}}$ in normal profile
4) if $\{op \mid op \in BV_E\} \neq \{op \mid op \in BV_N\}$
5) trigger an operation-content alarm
6) end if
7) if $\exists op_i$ s.t.$|\boldsymbol{BV}_{\mathrm{E}}.f(op_i)-\boldsymbol{BV}_{\mathrm{E}}.f(op_i)| > \sigma$
8) trigger an operation-frequency alarm
9) end if
10) end for
11) end for

在检测离群异常时，同样从操作内容和操作频率这 2 个方面进行考察，并采用基于行为向量距离的聚类来发现离群个体。由于某些执行者可能进行与其他执行者不同的操作，导致各执行者的行为向量维度不同，因此，首先需要将同任务同设备下各执行者的行为向量做同维处理，即将同角色执行者在同一任务同一设备上的操作的并集作为各执行者的操作集。对于每个执行者，将那些不属于自身的操作(本文称之为差异操作)的频率置为 0，其余频率保持不变；随后需要为向量的各个维度赋予权值。考虑到因差异操作造成异常情况的可能性较高，为这些差异操作赋予较高权值，其余操作赋予的权值则相对较低；然后计算各向量之间的欧氏距离，并采用基于距离的方法对向量聚类，聚类算法选择层次凝聚法；最后根据聚类结果判断出异常类簇。

算法 2 检测离群异常的算法

Outlier_AD(Execution Log)

1) for each *R*
2) for each *T* and *D*
3) set *op-set* to ∅
4) for each *O* belongs to *R*
5) $op\text{-}set = op\text{-}set \cup O.\boldsymbol{BV}_{\mathrm{E}}.op_i$
6) end for
7) unify the *op* of each $\boldsymbol{BV}$ according to *op-set*
8) for each *O* belongs to *R*
9) for each *op* which is newly added into $O.\boldsymbol{BV}_{\mathrm{E}}$
10) set $f(op_i) = 0$
11) end for
12) for each $O.\boldsymbol{BV}_{\mathrm{E}}$
13) set the weight of each *op*
14) end for
15) end for
16) cluster the $\boldsymbol{BV}_{\mathrm{ES}}$ according to the Euclidean distance between each $\boldsymbol{BV}$-*pair*
17) for each cluster of $\boldsymbol{BV}_{\mathrm{ES}}$
18) find out the abnormal cluster and trigger an outlier alarm
19) end for
20) end for
21) end for

3.2.2 控制流异常检测

在业务控制流方面，主要有 2 类异常，即逻辑

异常和性能异常。

定义 7　逻辑异常

逻辑异常是指当前业务事件的发生顺序没有遵循正常的业务过程逻辑结构，从而导致业务异常终止或返回错误结果的情况。

检测逻辑异常的方法通常为检查执行日志中事件序列与正常模型之间的一致性。

算法 3　检测逻辑异常的算法

Logic_AD (normal model, execution Log)

1) for each *ES* in execution Log

2)　if *ES* can't be parsed by normal model

3)　trigger a control-flow-logic alarm and log the fault event in *ES*

4)　end if

5) end for

定义 8　性能异常

性能异常是指在当前业务事件序列中，某些特定事件的性能指标超出了正常范围。

具体而言，本文关注的性能异常通常包括以下几点。

1) 时间点异常：某事件在规定范围 τ_1 之外的时间被执行。

2) 时间间隔异常：某些事件之间的时间间隔小于或超过规定值 τ_2。

3) 频次异常：业务活动中某些部分的执行频次多于或少于规定值 μ。

当检测性能异常时，需要根据“时间—频次约束表”中的内容，判断执行日志中的事件序列是否满足表中约束关系。

算法 4　检测性能异常的算法

Performance_AD(Time-Frequency Constraint Table, Execution Log)

1) for each *ES* in Execution Log

2)　if event *E* has moment-constraint

3)　if *E.timestamp* $\notin$ τ_1

4)　trigger a momont-anomaly alarm and log *E*

5)　end if

6)　end if

7)　if event E_1 and E_2 have time-interval constraint

8)　if $|E_1.timestamp - E_2.timestamp| > \tau_2$

9)　trigger a time-interval-anomaly alarm and log E_1 and E_2

10)　end if

11)　end if

12)　let *S* be the set of events which have frequency constraint

13)　if $\#(S) \notin \mu$

14)　trigger a Frequency-Anomaly alarm and log *S*

15)　end if

16) end for

4　实验与结果

实验过程分为 3 步。首先，利用过程日志生成工具生成了人造数据集（业务执行日志）作为原始数据，并向其中注入了一些异常情况，将其作为测试数据；然后，采用过程挖掘软件 ProM[10]对原始数据进行预处理，挖掘出业务的控制流模型，并进行一致性检测。同时，根据控制流模型和训练日志，利用编写的 Java 程序挖掘执行者的行为轮廓；最后，采用本文提出的方法对注入的异常情况进行了检测，并评估了模型的异常检测效果。

采用人工生成数据集的原因有多个方面。首先，企业和组织的真实内部业务审计日志通常有较高的机密性和隐私性，难以被研究者收集和使用；其次，人工生成的数据集在时间范围、数量规模上都可以十分灵活地根据研究需求进行修改，因此使用起来较为方便。下面将详细介绍实验步骤。

4.1　生成数据集

为生成实验数据集，采用过程日志产生工具(PLG, process log generator)[18]模拟了 5 个业务案例的执行过程。通过设置某些参数，PLG 可以生成一个随机的近似真实的业务过程模型，并模拟该业务的执行过程，以日志方式记录下所执行活动的详细情况。为得到更为准确的结果，对每个业务案例各模拟了 10 次执行过程，每次执行过程的事件信息被 PLG 记录到一个日志文件中，日志格式为 MXML，该格式可以支持 $\text{ProM}_{\text{import}}$ 插件的导入，方便后续利用 ProM 进行过程挖掘。

在得到正常的原始数据后，需要向其中注入异常数据，以模拟系统遭受的内部威胁。在此之前，人工地确定了相关的检测阈值，实际应用中可以根据专家知识来进行这一工作。每个被注入的异常数据都对应了一个事先设计的内部攻击场景，它描述

了攻击者的角色类型、攻击的操作细节和受影响的业务活动等信息。例如，某个攻击场景可能描述了某个角色的一个执行者在业务执行过程中比平时更为频繁地使用复制粘贴命令，并导致某个业务活动的执行时间比正常情况下显著增加。为使攻击场景的模拟尽量接近真实，异常数据是被直接关联到某些执行者个体及其所执行的业务活动上的，业务的控制流和性能异常是与执行者的异常行为相关的。原始数据与注入的异常数据的细节情况如表 1 所示，其中，OCA 表示操作内容异常（operation-content anomaly）、OFA 表示操作频率异常（operation-frequency anomaly）、LA 表示逻辑异常（logical anomaly）、MA 表示时间异常（moment anomaly）、TIA 表示时间间隔（time-interval anomaly）、FA 表示频率异常（frequency anomaly）。

4.2 模型实现

模型挖掘首先从日志预处理开始。每次挖掘时，通过 $\text{ProM}_{\text{import}}$ 插件导入某个业务一次执行过程产生的日志，然后利用 ProM 提供的日志过滤工具对日志进行过滤，保留那些完整的业务日志。随后，采用 ProM 中的遗传算法插件对预处理后的日志进行挖掘，得到以 C-net 表示的控制流模型。最后，结合挖掘出的控制流模型和训练日志中的信息，利用 Java 编程挖掘出各执行者的行为轮廓，并统计相关参数。

为减小实验误差，得到更加稳定可靠的模型，采取常用的 10-折交叉验证的方法重复上述步骤。即对每个业务而言，将得到的 10 次执行日志分为 10 份，将其中的 9 份作为训练集，剩余 1 份作为验证集，用以执行上述过程。下次验证再对训练集和验证集重新划分，保证重复的 10 次过程中每份日志都充当过一次验证集。最后，从 10 次验证得到的结果中取相关参数的平均值，作为最终得到的挖掘模型。

4.3 实验结果

利用提出的检测方法对测试数据进行了测试，结果如表 2 所示。

表 2 异常检测结果

案例编号	异常报警情况
1	5 OCA, 1 LA
2	9 OCA, 4 OFA, 3 LA, 2 TIA
3	14 OCA, 4 OFA, 4 LA, 4 MA
4	16 OCA, 8 OFA, 3 FA
5	22 OCA, 12 OFA, 7 LA

为评估检测效果，采用信息检索和分类领域中常用的 F_1-measure 作为评估指标。F_1-measure 满足

$$F_1 = \frac{2\times P\times R}{P+R} \tag{3}$$

$$P = \frac{TP}{TP+FP} \tag{4}$$

$$R = \frac{TP}{TP+FN} \tag{5}$$

其中，P 代表准确率，R 代表召回率，TP 代表正阳率，FP 代表假阳率，FN 代表假阴率。根据表 2 中的结果，计算出了各业务的 P、R 和 F_1-measure 值，如表 3 所示。从表 3 可以看出，本实验中对业务 1 的检测准确率最高，达到了 100%，对业务 4 的召回率最高，达到了 100%。由于实验数据集的规模和注入的异常数据有限，较少的误报和漏报都会对准确率和召回率造成较大影响，因此通过准确率和召回率的调和均值 F_1-measure 来判断检测方法的整体效果。结合表 1 和表 3 可以得知，无论业务事件的规模大小，F_1-measure 值都能维持 90%以上的较高水平，表明本文提出的方法针对具体的异常情况具备良好的检测性能。

表 1 人造训练集与测试集

案例编号	事件平均数量	角色数量	执行者数量	注入异常情况	恶意执行者数量
1	574	3	7	5 OCA, 2 LA	1
2	1 227	3	9	9 OCA, 3 OFA, 3 LA, 3 TIA	1
3	2 072	4	12	13 OCA, 5 OFA, 5 LA, 4 MA	2
4	3 356	4	14	16 OCA, 7 OFA, 2 FA	2
5	5 184	5	18	21 OCA, 12 OFA, 6 LA	3

表 3　　准确率、召回率与 F_1 分数

案例编号	准确率	召回率	F_1 分数
1	100%	85.7%	92.3%
2	94.4%	94.4%	94.4%
3	96.2%	92.6%	94.4%
4	92.6%	100%	96.2%
5	92.7%	97.4%	95.0%

在离群异常检测测试中，选择了业务案例 5 作为测试案例，采用层次凝聚法对同角色各执行者的行为向量进行聚类，并发现异常类簇，结果如表 4 所示。

表 4　　离群点检测结果

角色编号	执行者数量	类簇数量	离群点数量
1	3	2	1
2	4	1	0
3	5	2	1
4	3	1	0
5	3	2	1

可以看出，第 1、3 和 5 号角色的聚类结果中各存在一个明显的离群点，这与人工注入的异常情况相符。由于本文的实验数据是静态环境，没有业务活动的动态变更，出现个体异常的执行者与其他同角色执行者的行为向量距离会明显增加，即同时表现为离群异常，因此表 4 中的离群异常数量与表 2 中注入的异常执行者数量相等。在实际情况下，如果存在角色层面的业务活动调整，则该角色所有执行者的行为向量都会发生相似变化，使离群异常执行者数量少于个体异常执行者数量。

5　结束语

本文提出了一个基于业务过程挖掘的内部威胁检测系统。该系统以正常业务日志为数据源，通过过程挖掘和一致性检测的方式得到业务活动的控制流模型，它反映了正常的业务活动应该遵循的逻辑顺序和结构。在此基础上，结合控制流模型和日志信息进一步挖掘出业务执行者的行为轮廓，它以树状结构反映了执行者在担任不同角色、执行不同任务时的正常操作的范围。此外，还通过专家知识和统计方法得到了业务活动在性能方面的正常数据。随后，分析了执行者在进行内部攻击时可能进行的异常行为，以及对业务活动本身造成的异常情况，并详细描述了异常检测方法。在 5 个人工模拟的业务过程场景中进行了实验，并利用 F_1-measure 值评估了检测算法在准确率和召回率上的综合表现，结果表明本文提出的方法能够有效地检测出注入的异常情况，发现实施内部攻击的恶意人员。

然而，本文的方法也存在一些局限性。本文提出的检测系统目前所挖掘的都是静态的业务模型和人员行为轮廓，一旦业务活动发生变化，人员分工进行调整，则原有模型不再适用。而且，由于缺乏真实的业务日志数据，使对业务活动的细节理解上必然存在一些偏差。在以后的工作中，将重点关注如何建立业务过程和人员行为的动态模型，使之具备更强的适应力。同时，还将研究如何通过挖掘组织内部人员之间的社交关系来检测多人共谋的内部攻击。

参考文献：

[1] PARVEEN P, THURAISINGHAM B. Unsupervised incremental sequence learning for insider threat detection[C]//2012 IEEE International Conference on Intelligence and Security Informatics (ISI). 2012: 141-143.

[2] AALST W M P, MEDEIROS A K A. Process mining and security: detecting anomalous process executions and checking process conformance[J]. Electronic Notes in Theoretical Computer Science, 2005, 121: 3-21.

[3] AALST W, WEIJTERS T, MARUSTER L. Workflow mining: Discovering process models from event logs[J]. IEEE Transactions on Knowledge and Data Engineering, 2004, 16(9): 1128-1142.

[4] WEN L, WANG J, SUN J. Detecting implicit dependencies between tasks from event logs[J]. Frontiers of WWW Research and Development-APWeb 2006, 2006: 591-603.

[5] WEIJTERS A, RIBEIRO J T S. Flexible heuristics miner (FHM)[C]// 2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM). 2011: 310-317.

[6] WEIJTERS A J M M, VAN der AALST W M P. Rediscovering workflow models from event-based data using little thumb[J]. Integrated Computer-Aided Engineering, 2003, 10(2): 151-162.

[7] DONGEN B F, AALST W M P. Multi-phase process mining: Aggregating instance graphs into EPCs and Petri nets[C]//PNCWB 2005 Workshop. 2005: 35-58.

[8] VAN DONGEN B F, VAN der AALST W M P. Multi-phase process mining: Building instance graphs[C]//Conceptual Modeling–ER 2004. 2004: 362-376.

[9] DE MEDEIROS A K A, WEIJTERS A. Genetic process mining[C]// 26th International Conference on Applications and Theory of Petri

Nets. 2005.

[10] AALST W M P, DONGEN B F, GÜNTHER C W, et al. ProM: the process mining toolkit[J]. BPM (Demos), 2009, 489: 31.

[11] ANDERSON R H, BOZEK T, LONGSTAFF T, et al. Research on mitigating the insider threat to information systems-# 2[R]. Rand National Defense Research Inst Santa Monica CA, 2000.

[12] SPITZNER L. Honeypots: catching the insider threat[C]//19th Computer Security Applications Conference. 2003: 170-179.

[13] HU N, BRADFORD P G, LIU J. Applying role based access control and genetic algorithms to insider threat detection[C]//The 44th Annual Southeast Regional Conference. 2006: 790-791.

[14] BISHOP M, ENGLE S, PEISERT S, et al. We have met the enemy and he is us[C]//The 2008 Workshop on New Security Paradigms. 2009: 1-12.

[15] GREITZER F L, FRINCKE D A. Combining traditional cyber security audit data with psychosocial data: towards predictive modeling for insider threat mitigation[C]//Insider Threats in Cyber Security. 2010: 85-113.

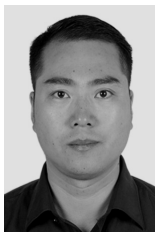
[16] BRDICZKA O, LIU J, PRICE B, et al. Proactive insider threat detection through graph learning and psychological context[C]//2012 IEEE Symposium on Security and Privacy Workshops (SPW). 2012: 142-149.

[17] PARVEEN P, EVANS J, THURAISINGHAM B, et al. Insider threat detection using stream mining and graph mining[C]//2011 IEEE Third International Conference on Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom). 2011: 1102-1110.

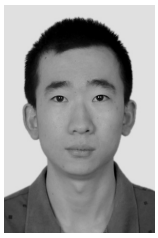
[18] BURATTIN A, SPERDUTI A. PLG: a framework for the generation of business process models and their execution logs[C]//Business Process Management Workshops. 2011: 214-219.

作者简介：

朱泰铭（1991-），男，湖北荆州人，解放军信息工程大学硕士生，主要研究方向为大数据分析、内部威胁检测。



郭渊博（1975-），男，陕西周至人，解放军信息工程大学教授、博士生导师，主要研究方向为大数据安全、态势感知。



琚安康（1995-），男，河南新乡人，解放军信息工程大学硕士生，主要研究方向为多步网络攻击检测、威胁情报。

马骏（1981-），男，山西阳泉人，解放军信息工程大学讲师，主要研究方向为物联网安全、大数据安全。