基于多层抽象的程序行为模型及异常检测研究

程 霞[1,2] 王晓锋[3]
(四川师范大学经济与管理学院 成都 610068)[1] (南开大学商学院 天津 300071)[2]
(华为赛门铁克科技有限公司 北京 100085)[3]

摘 要 在入侵检测领域,对程序行为的异常分析始终缺乏高效的短周期模型,现有模型对程序行为的抽象能力非常有限。为此,首先提出一种新的、具备充分自描述能力的模式:间隙变长频繁短序列模式(GV模式),该模式涵盖描述程序行为的顺序、选择和循环3种基本结构;然后给出GV模式挖掘算法以及基于GV模式库的系统调用流程图模型。实验表明,基于新模型的异常检测算法简单高效,在保持高检测率的前提下具有较低的检测开销和误检率,具备了实时检测能力。

关键词 异常检测,短周期模型,行为分析,模式匹配,数据挖掘
中图法分类号 TP393 **文献标识码** A

Research on Program Behavior Model and Anomaly Detection Based on Multiple Abstraction

CHENG Xia[1,2] WANG Xiao-feng[3]
(Economic and Management School, Sichuan Normal University, Chengdu 610068, China)[1]
(Management School, Nankai University, Tianjin 300071, China)[2]
(Huawei Symantec Technologies Co., Ltd, Beijing 100085, China)[3]

Abstract Efficient short sequence models used in anomaly analysis of program behaviors are not available in anomaly detection field. The current models are short of abstracting program behaviors. Therefore, a new highly self-explanatory pattern called GV pattern(gapped variable frequent pattern) was provided to cover three fundamental structures of program: sequence, selection and circulation. Subsequently, GV pattern mined algorithm and system-call flow chart model based on GV pattern library were presented in details. Experiments show that the anomaly detection algorithm based on new model keeps low detection overhead and false positive rate on the condition of high detection rate, which is crucial in a real-time intrusion detection system.

Keywords Anomaly detection, Short sequence model, Behavior analysis, Pattern matching, Data mining

1 引言

在入侵检测领域,目标系统上观测得到的审计事件流可以作为时间序列进行分析,例如程序的系统调用序列、Shell命令序列、Web页面访问序列等,这些序列描述了系统运作的行为特征。基于行为特征的时序分析为入侵检测提供了充分的依据。

频繁模式是一种常用的时序分析工具,特指序列中频繁出现的小片段。频繁模式引入入侵检测,就会产生用于异常检测的短周期模型,基本思想是从正常的序列中挖掘出由频繁出现的序列片段组成的标准频繁模式库,用于抽象出观测系统的行为规范。异常的序列片段与标准库相悖,从而背离了规范。

目前采用短周期模型对程序行为进行异常分析的研究较少。Forrest[1]最早将程序行为定义为系统调用序列,提取定长的系统调用序列为目标程序建模,这种定长的频繁模式称为N模式。基于N模式的检测算法一般称为STIDE算法[2]:训练阶段,利用长度为n的滑动窗口在训练序列上滑动,落入窗口的新的N模式被存入模式库;检测阶段,用同样的滑动窗口在待测序列上滑动,检查落入窗口的每一个长度为n的短序列,若短序列在模式库中存在,称为匹配,否则称为失配。通常将待测序列的失配率作为计算序列异常指标的基础。

采用N模式的异常检测算法十分简洁直观,但是在不同的应用场合确定合适的n值是个难题。过短的N模式会将异常序列过度分解,造成异常信号太弱而难以检测。而过长的N模式一方面会导致模式库规模较大,检测开销增加,另一方面过于严格,误检率太高。为了灵活地从事件序列的训练集中提取更合适的频繁模式,Wespi[3,4]等提出了变长频繁模式库的挖掘算法。该方法采用TEIRESIAS算法[5]输出候选的变长模式集,然后从候选集中选取最优子集,该子集能够覆盖事件序列的训练集,而子集规模最小。与定长的N模式

到稿日期:2009-04-24 返修日期:2009-07-13 本文受国家自然科学基金(70471040)和国防研究基金(4131605)资助。
程 霞(1980—)女,博士研究生,讲师,主要研究方向为电子商务安全及供应链管理,E-mail:cxshine@yahoo.com.cn;王晓锋(1978—)男,博士,主要研究方向为网络与信息安全。

相比，Wespi 的方法能够用更少的 V 模式灵活地涵盖训练序列集。

本文将对基于频繁模式的程序行为异常检测做深入探讨。对系统调用序列进行多层抽象后，给出新的程序行为异常检测模型：系统调用流程图模型，并结合现有模型进行异常检测分析。

2 GV 模式

无论程序结构多么复杂，都可以分解为 3 种基本结构：顺序、选择和循环。观测程序行为得到的任何系统调用序列，都是这 3 种基本结构组合的快照。V 模式很好地描述了序列的顺序结构。但是对于包含大量选择结构的序列，无论是 N 模式还是 V 模式，描述能力都不充分。为此，对 V 模式加以扩展，允许包含有限的不确定元素，得到间隙型变长频繁短周期模式，简称 GV 模式。

在定义 GV 模式之前，首先给出频繁模式、强弱模式、最大模式和冗余模式[6]等概念。

定义 1 频繁模式是指序列集中频繁出现的短片段。对于给定的训练序列集和支持度阈值 sup，如果模式 P 在超过 sup 个序列中出现，则 P 是频繁模式。模式 P 的出现次数称为支持度。

定义 2 强弱模式是表征模式特殊性的相对概念。如果将模式 P 经过扩展或特化得到模式 Q，则称 Q 是 P 的强模式，或者称 P 是 Q 的弱模式。这种扩展或特化称为强化。例如，以句点表示间隙，对于模式 $AB.C$，扩展得到的 $AB.CD$ 和特化得到的 $ABDC$ 均为 $AB.C$ 的强模式。

定义 3 最大模式是指不可能在支持度不降的情况下进行强化的模式。对于给定的序列集，如果序列包含模式 P，则一定也包含 P 的弱模式 Q，因此强模式的支持度不高于弱模式。如果模式 P 的所有强模式的支持度均低于 P 的支持度，则称 P 是最大模式。

定义 4 冗余模式是可替代的模式。假设有最大模式 P 以及 P 的 n 个强模式 $P_1, P_2, \cdots, P_n$，若序列集中包含模式 P 的所有序列至少也包含 $P_1, P_2, \cdots, P_n$ 中的一个，则称 P 是冗余模式。冗余模式是一个相对概念。冗余是指在一个最大模式库中，如果某个模式在训练序列集中的任意出现都被库中已知的更强模式所覆盖，则这个模式在库中就冗余了。

定义 5 GV 模式是频繁的、最大的、非冗余的$\langle L,W\rangle$模式[5]。GV 模式满足如下条件：

(1)模式长度不固定，但是模式中的起始元素和结尾元素必须是确定元素；

(2)模式中任意以确定元素作为起止元素的子模式，如果子模式的确定元素数为 l，则该子模式的长度不能大于 w；

(3)是频繁模式；

(4)是最大模式；

(5)是非冗余模式。

满足条件(1)和(2)的短序列称为$\langle L,W\rangle$模式。条件(2)利用给定的参数 l 和 w 限定了间隙密度，w 与 l 的差值越大，模式中允许出现的间隙就越多。例如，如果指定 w 和 l 分别为 4 和 3，则短序列 $AB.CD$ 是$\langle L,W\rangle$模式，而 $AB.C.D$ 不是，因为子序列 $B.C.D$ 的确定元素数为 3，但是该子序列的长度为 5，超过了指定的 w 值。表 1 记录了一段简单的应用程序代码可能产生的系统调用观测序列以及建模结果，$A-F$ 是 6 种系统调用。涵盖该程序行为的所有时序信息，需要 17 个长度为 3 的 N 模式，或者 8 个 V 模式，或者 2 个 GV 模式。GV 模式的间隙包含一个任意元素，循环体用方括号表示，$*$ 作用于左侧的循环体，表示一次或者多次循环，循环体是可以嵌套的。3 种频繁模式中，GV 模式最接近程序代码的实际结构，其变长特性描述了顺序结构，间隙描述了选择结构，重复符号描述了循环结构。一般而言，仅凭训练序列集中的数据无法获取全部的行为信息，但是 GV 模式集合能够在很大程度上“复原”出系统行为的原貌。

表 1 3 种短周期模式库对比

代码		系统调用序列训练集	频繁短周期模式库 N模式(n=3)	V模式	GV模式(识别出循环)
	1	AEF	AEF		
	2	ABCFEF	ABC		
A;	3	ABDFEF	BCF		
for(…)	4	ABEFEF	CFE		
{	5	ABCFBDFEF	FEF		
B;	6	ABCFBEFEF	ABD	AEF	
if(…)	7	ABDFBEFEF	BDF	ABCF	
C;	8	ABCFBCFEF	DFE	ABDF	
else if	9	ABDFBDFEF	ABE	ABEF	AEF
(…) D;	10	ABEFBEFEF	BEF	BCF	A[B.F]*EF
else	11	ABCFBDFBEFEF	EFE	BDF	
(…) E;			CFB	BEF	
F;			FBD	EF	
}			FBE		
E;	…	……	DFB		
F;			FBC		
			EFB		

3 GV 模式挖掘

GV 模式挖掘过程是对程序行为进行多层抽象的过程，分为 3 步：第一步，对训练序列进行预处理，消除显式循环片段；第二步，采用深度优先遍历算法，得到所有确定元素数为 l 的频繁$\langle L,W\rangle$模式。这些模式称为基本模式；第三步，连接具有相同首部和尾部的基本模式，得到 GV 模式，连接算法确保输出所有频繁的、最大的、非冗余的$\langle L,W\rangle$模式。

3.1 显式循环

消除显式循环片段的算法十分直观。仍以表 1 为例，显式循环的消除如表 2 所列。如果序列中包含嵌套的多层循环，则消除循环是个迭代的过程。

表 2 消显式循环片段

	序列	显式循环片段
初始序列	ABCFBDFBDFBFBFBEFBEFBDFBFGH	—
识别显式循环片段	ABCF BDFBDF BCFBCF BEFBEF BDFGH	BDF,BCF,BEF
消除显式循环片段	ABCF BDF BCF BEF BDFGH	—

消除显式循环中的重复片段并不会损失信息。Long[7]等的研究表明，这种简单算法能够从很大程度上改善序列中大量存在的冗余信息带来的不利影响。然而，包含选择结构的序列中，除了显式循环外，还存在隐式循环。表 1 代码中，由于循环体内还包含了选择结构，因此 BCF，BDF 和 BEF 均是可能的循环片段，这些片段组成了隐式循环。目前没有任何短周期模型能够完全准确地描述由选择结构和循环结构结合形成的隐式循环。但是正如表 1 所列，GV 模式对于程序

行为的描述是相对精确的。

3.2 基本模式

给定参数 l,w 和 sup,基本模式定义为确定元素数恰好为 l 的频繁〈L,W〉模式,长度 len 在 l 到 w 之间,模式的中间位置包含了最多 $w-l$ 个间隙。

Pratt[8]算法是一种深度优先遍历算法,能够挖掘出任意长度的频繁〈L,W〉模式,但是需要计算大量未知扩展模式的支持度,从而极大地增加了训练序列集的扫描次数。为提高效率,在挖掘基本模式时,采用定长模式插入法为长度在 l 到 w 之间的每种模式分别构造模式树。这样只需要计算实际存在模式的支持度,减少了训练序列集的扫描次数。设需要构造长度为 len 的基本模式的模式树,算法步骤如下:

(1)建立模式树 T 的根节点,记为 $NULL$;

(2)以定长 len 的窗口逐一在所有训练序列上滑动,得到一个定长的频繁短序列 P;

(3)基本模式包含 $m=w-l$ 个间隙,将 P 与 T 中所有分枝进行匹配比较,设 T 中有分枝模式 P' 与 P 的不匹配数 d(即 P 与 P'之间的海明距离)最小:如果 d 不大于 m,则将 P 合并入 P'分枝,不匹配处成为候选间隙,记录候选间隙中的所有可选元素;否则,从根节点 $NULL$ 处起为 P 建立新的分枝;

(4)继续滑动窗口,直至遍历完所有训练序列。

图 1 和图 2 是以表 1 的系统调用序列作为训练集,指定各参数 $l=2$,$w=3$ 以及 $sup=1$ 时得到的两棵基本模式树,分别表示长度为 2 的无间隙基本模式和长度为 3 的单候选间隙基本模式。在包含候选间隙的模式树中,当候选间隙中的可选元素超过定阈值 k 时,候选间隙转化为间隙,可以匹配任意元素。图 1 不包含候选间隙,所有模式均为基本模式。假设 $k=2$,图 2 中只有模式 $B[CDE]F$ 转化为基本模式 $B.F$,其它模式的间隙位于模式的起止点,不符合〈L,W〉模式的要求。不同长度的基本模式是分开挖掘的,而较长的基本模式中包含更多的间隙,因此可能会出现较短模式相对较长模式而言是非最大模式的情况。这些已知非最大的较短模式对于接下来通过连接生成更强模式并无贡献,应该将其排除在基本模式之外。例如,由于存在长度为 3 的单间隙模式 $B.F$,图 1 中的 6 个长度为 2 的模式 $BC/BD/BE/CF/DF/EF$ 是非最大的,因此最终的基本模式只有 6 个:$AB/AG/GH/FB/FG/B.F$。

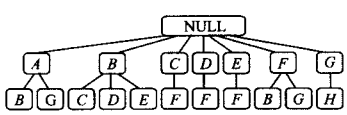

图 1 无间隙基本模式树($l=2$,$w=3$,$sup=1$)

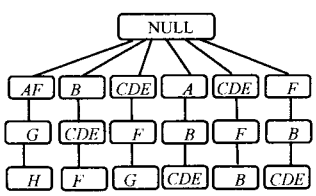

图 2 单候选间隙基本模式树($l=2$,$w=3$,$sup=1$)

3.3 *GV* 模式生成算法

假设当前模式 P 的匹配列表记为 M_List_P,每个匹配项代表一次匹配,即模式 P 在训练序列集第 No_P 个序列的位置 Pos_P 处匹配。覆盖标识 Ovl_P 是布尔量,表明在这个位置是否有更强的已知模式相匹配(即覆盖)。同样,待连接模式 Q 的匹配列表记为 M_List_Q,连接后得到的模式 R 的匹配列表记为 M_List_R,如式(1)所示。

$$\begin{aligned}&M\_List_P=\{(No\_P_i,Pos\_P_i,Ovl\_P_i)\mid i=1,2,\cdots,M\}\\&M\_List_Q=\{(No\_Q_j,Pos\_Q_j,Ovl\_Q_j)\mid j=1,2,\cdots,N\}\\&M\_List_R=\{(No\_R_k,Pos\_R_k,Ovl\_R_k)\mid k=1,2,\cdots,O\}\end{aligned}\quad(1)$$

不用扫描训练序列集,直接比较 M_List_P 和 M_List_Q 就能够得到 R 的匹配列表 M_List_R。当且仅当模式 P,Q 和 R 各自匹配列表的匹配项满足式(2)的条件时,P 的右端能够与 Q 的左端连接产生新的扩展模式 R,$Len(P)$ 是模式 P 的长度。

$$\begin{aligned}&No\_P_i=No\_Q_j=No\_R_k\\&Pos\_R_k=Pos\_P_i\\&Pos\_Q_j=Pos\_P_i+Len(P)-L+1\end{aligned}\quad(2)$$

同理,当且仅当模式 P,Q 和 R 各自匹配列表的匹配项满足式(3)的条件时,P 的左端才能够与 Q 的右端连接产生新的扩展模式 R,$Len(Q)$ 是模式 Q 的长度。

$$\begin{aligned}&No\_P_i=No\_Q_j=No\_R_k\\&Pos\_R_k=Pos\_Q_j\\&Pos\_P_i=Pos\_Q_j+Len(Q)-L+1\end{aligned}\quad(3)$$

当发现扩展模式 R 为频繁模式时,需要更新 P 的匹配列表,将列表中所有被 R 覆盖的匹配项的覆盖标识域置位。

挖掘 GV 模式的最后一步是以基本模式集合作为输入,采用带冗余控制的 GV 模式生成算法,将基本模式按照深度遍历的方式连接,最终得到无冗余的 GV 模式集合。

为了确保输出所有可能的模式以及更强的最大模式先于非最大模式输出,首先对基本模式集合进行两种排序:前缀排序和后缀排序。前缀排序是将基本模式的前 $l-1$ 个元素组成的子模式转化为 $l-1$ 位二进制数,确定元素取 1,间隙取 0,模式按照对应的二进制大小递减排序;后缀排序与前缀排序类似,只是取基本模式的后 $l-1$ 个元素逆序转化,即最末位元素对应二进制数的最高位。按照这种排序规则,更强的模式会被优先连接扩展,因此较强的最大模式会被更早发现并输出。

GV 模式生成算法基本步骤如下。

算法 GV 模式生成算法

输入:基本模式集合 E_P,〈L,W〉模式参数 l,频繁阈值 sup

输出:GV 模式集合 GV_P

过程 1:GV_Gen(生成所有 GV 模式)

(1)将基本模式集合 E_P 分别按照前缀排序和后缀排序得到两个有序集,前缀集记为 E_P_{prf},后缀集记为 E_P_{sf};

(2)按照顺序每次从前缀集 E_P_{prf} 中取一个基本模式作为当前模式,记为 P;

(3)以 P 作为输入调用过程 Convolution_Rightmost_In_Depth;

(4)以 P 作为输入调用过程 Convolution_Leftmost_In_Depth;

(5)返回步骤(2),直到取完 E_P_{prf} 中的所有基本模式。

过程 2:Convolution_Rightmost_In_Depth(深度优先右扩展)

(1)顺序选择可连接模式。从前缀集 E_P_{prf} 依次寻找能

够与 P 的右端进行连接的模式 Q，如果有多个模式能与 P 的右端连接，则前缀排序在前的模式优先。若搜索到前缀集 E_P_{prf} 末尾没发现可以连接的模式：检查 GV_P 中是否存在比 P 更强且支持度等于 P 的模式，若是则退出过程，否则将 P 输出到 GV 后退出过程。若有可以连接的模式 Q，进入下一步；

(2)右连接扩展。连接 P 与 Q 得到扩展模式 R，根据式(2)的约束条件得到 R 的匹配列表 M_List_R，若 $|M\_List_R| < sup$，丢弃 R，回到步骤(1)，否则，更新 P 的匹配列表，对被 R 覆盖的匹配项覆盖标识置位，将 R 设置为当前模式 P，递归调用过程 Convolution_Rightmost_In_Depth；

(3)冗余控制。检查 P 的匹配列表，若所有匹配项覆盖标识均被置位，退出过程，否则回到步骤(1)，继续按顺序寻找可连接模式。

过程 3：Convolution_Leftmost_In_Depth(深度优先左扩展)

(1)顺序选择可连接模式。从后缀集 E_P_{sf} 依次寻找能够与 P 的左端进行连接的模式 Q，如果有多个模式能与 P 的左端连接，则后缀排序在前的模式优先。若搜索到后缀集 E_P_{sf} 末尾没发现可以连接的模式：检查 GV_P 中是否存在比 P 更强且支持度等于 P 的模式，若是则退出过程，否则将 P 输出到 GV 后退出过程。若有可以连接的模式 Q，进入下一步；

(2)左连接扩展。连接 P 与 Q 得到扩展模式 R，根据式(3)的约束条件得到 R 的匹配列表 M_List_R，若 $|M\_List_R| < sup$，丢弃 R，回到步骤(1)，否则，更新 P 的匹配列表，对被 R 覆盖的匹配项覆盖标识置位，将 R 设置为当前模式 P，递归调用过程 Convolution_Leftmost_In_Depth；

(3)冗余控制。检查 P 的匹配列表，若所有匹配项覆盖标识均被置位，退出过程，否则回到步骤(1)，继续按顺序寻找可连接模式。

算法包含向左和向右两个方向深度优先扩展的递归过程，确保输出所有频繁 GV 模式。扩展过程中，中间模式匹配列表的建立与更新完全以基本模式匹配列表为基础，不需要任何扫描训练序列集的操作。两个递归过程的冗余控制步骤确保一旦发现冗余模式，立即中止进一步扩展，降低了算法的复杂度。

4 系统调用流程图

如果将消除显式循环作为对程序行为的第一层抽象，将挖掘 *GV-GRAM* 模式作为第二层抽象，那么在 *GV-GRAM* 模式上识别可能的隐式循环则是第三层抽象。经过多层抽象得到的系统调用流程图，能够比较精确地复原出程序的行为轮廓。

从 GV 模式得到的系统调用流程图类似 Sekar 等为系统调用序列建立的 FSA 状态机模型[9]，但图中的节点并非状态，而是事件本身。因此，建立系统调用流程图不需要收集额外的状态信息，也不存在动态链接库的限制。系统调用流程图模型集成了短周期模型与状态机模型的优点，是一种高效的事件序列异常检测模型。

图 3 是表 1 中 GV 模式库对应的系统调用流程图。流程图是带回路的树，从根节点 *NULL* 出发的每个完整分枝代表一个独立的 GV 模式，树枝上包含子孙节点到祖先节点的回路，用于表示循环结构。从 GV 模式集合创建的系统调用流程图非常直观：为每个模式建立分枝，将模式中的每个元素作为一个节点，识别出模式中的循环片段作为回路。从 GV 模式的角度来看，这些循环都是显式的，从最内层开始迭代地识别各层循环，每次迭代识别一层循环，直到分枝上不存在重复片段为止。

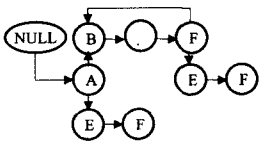

图 3 系统调用流程图

5 检测

基于系统调用流程图的异常检测算法十分简单。对于待测序列，按照如下步骤实施检测：

(1)将待测序列保存入队列 *Seq*；创建系统调用流程图匹配指针 *Ptr*，初始化 *Ptr*，使其指向流程图的根节点 *NULL*，*Ptr* 所指节点称为 *Ptr* 节点；初始化异常指数 A 为 0；

(2)若 *Seq* 为空，输出 A，否则移出 *Seq* 队首事件，并存入 e；

(3)检查 *Ptr* 节点是否有后续节点，若没有，将 *Ptr* 重置为根节点 *NULL*；检查 *Ptr* 节点的所有后续节点(包括回路所指的祖先节点)，若发现能够匹配 e 的子节点 *Node*，则设置指针 *Ptr* 指向 *Node*，回到步骤(2)，否则进入下一步；

(4)将事件 e 定义为异常事件，计算 *Seq* 中最近被连续定义为异常事件的事件数，若该数大于 A，将其赋予 A；

(5)回到步骤(2)。

检测算法只对待测序列进行一次扫描，并且直接通过系统调用流程图匹配当前事件，不需要每次选取最合适的分枝模式进行匹配。这种深度优先匹配的方法与挖掘 GV 模式的算法一致，确保了正常的待测事件序列能够与流程图模型进行高效匹配。

6 实验与分析

本节将系统调用流程图检测算法与已有的两种短周期模型异常检测算法，即基于 N 模式的 STIDE 算法[2]以及基于 V 模式的变长频繁短周期模式匹配算法[4]进行对比实验，测试了 3 种算法的检测模型规模、误检率、检测率以及检测开销等性能指标。

表 3 3 种频繁模式库规模对比

正常序列集		模式库规模			
		N		V	GV l=3, w=4
来源	类别	n=6	n=10		
UNM	bounce1	31	58	21	9
	bounce2	57	97	27	16
	bounce	79	121	35	19
	plus	164	231	42	20
	queue	114	174	29	11
	sendmail daemon	362	598	89	52
	Sendmail. log	131	209	32	21
CERT	Sendmail . daemon	217	335	61	44
	sendmail	71	127	32	27

Forrest[1]最早提出将系统调用序列用作主机异常检测的观测数据源时，采用了跟踪 sendmail 应用程序的系统调用序列得到的两组数组，分别是 UNM sendmail 合成数据集和 CERT sendmail 合成数据集，每个数据集都包含超过 150 万

条的系统调用。合成是指数据集分别包含正常场景下的正常序列和入侵场景下的入侵序列。数据分为两列，分别代表进程 ID 和系统调用序号。将属于同一进程 ID 的系统调用序号分类提取出来，就会构成一个序列，每个场景对应一个序列集，称为一个类别。为每个正常应用场景类别的序列集分别建立 3 种短周期模式库，模式库规模如表 3 所列，识别出循环后的 GV 模式库是最精简的。

算法的误检率水平可以通过计算正常序列的异常指标来衡量。为比较 3 种模式在训练不足时的误检率水平，将 UNM 以及 CERT 的 sendmail 正常系统调用序列按照 3 种不同比例分为两组，一组用来训练得到模式库，另一组作为检测集计算正常行为的异常指标。为便于比较，基于 N 模式库的检测算法采用每个序列中不匹配模式的最大海明距离作为序列的异常指标，而基于 V 模式库和 GV 模式库的算法做同样处理，序列的异常值取最长连续异常调用子序列的长度，然后取检测集中所有序列的异常值均值作为异常指标。3 种算法在训练不足时对正常序列的异常指标如表 4 所列。即使在训练严重不足的情况下，对于正常序列，GV 模式仍然保持了较高的匹配度，而 V 模式在 CERT 数据集下的异常指标甚至超过了 N 模式，这是因为在为 V 模式库选择模式时是以覆盖训练集为导向的，模式库的完备性更加依赖训练集的规模。

表 4　正常序列的异常指标均值对比

正常序列集		模式库规模			
		N		V	GV
来源	训练集/检测集	n=6	n=10		l=3, w=4
UNM	9/1	1.47	2.18	0.64	0.41
	6/4	1.77	2.81	0.91	0.75
	3/7	2.74	4.09	1.97	1.21
CERT	9/1	1.36	2.08	1.15	0.37
	6/4	1.89	2.71	2.48	0.55
	3/7	2.64	3.91	4.02	0.93

UNM 和 CERT 两组数据集包含了充分的入侵场景下的系统调用序列，可以用来检验算法的检测率水平。每类入侵场景有若干组数据，每组数据代表对一次入侵产生的系统调用序列的全程跟踪记录。每个场景下所有序列的平均异常指标代表该类入侵的异常指标，3 种算法对各种入侵场景的异常指标如表 5 所列。

表 5　sendmial 入侵场景的异常指标对比

正常序列集		模式库规模			
		N		V	GV
来源	类别	n=6	n=10		l=3, w=4
UNM	decode	3.2	5.4	11.8	7.3
	fwd_loops	4.1	6.7	15.2	11.4
	sscp	5.2	7.1	19.7	12.2
CERT	sm5x	4.4	9.0	13.2	9.3
	sm565a	3.9	7.3	12.8	11.4
	syslog-local	4.2	9.1	17.5	13.1
	syslog-remote	4.3	8.1	10.1	9.8

从异常信号的强度来看，GV 模式的流程化检测算法配合间隙的存在能够最大限度地保证正常序列的匹配，降低误检率，也可能会导致异常子序列的分解，削弱异常信号。但是另一方面，GV 模式是较长的强模式，这种削弱效应有限。从实验数据来看，基于 GV 模式的检测算法输出的异常指标稍低于基于 V 模式的算法，但是仍然足以检测出入侵。

图 4 是 3 种检测算法对 CERT 数据集中所有序列的检测开销均值。基于 V 模式的检测算法，每序列检测开销较大，这是因为算法每次从 V 模式库中选择一个匹配模式时都需要执行一个前向的预匹配过程，选择一个能够在未来有限范围内最大限度地匹配待检测序列的最优模式，预匹配过程的计算开销较大。基于 N 模式的检测算法主要开销来源于较大规模的模式库，模式库越大，每次匹配的开销越大，而且由于滑动窗口步长为 1，每个事件事实上被重复遍历了 n 次。采用系统调用流程图作为检测模型时，每个事件只需要遍历一次，而且模式库规模较小，因此检测开销远低于另外两种模型，能够满足实时检测的要求。

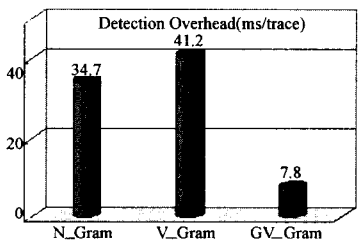

图 4　3 种短周期模型的每序列检测开销

结束语　事件序列是入侵检测的重要数据模型，以事件序列作为训练集可以挖掘出短周期模式库以及 FSA 有限状态机等检测模型。在基于短周期模式的检测算法中，基于定长的 N 模式的 STIDE 等算法难于确定模式长度，且存在局部性缺陷带来的漏检可能；变长的 V 模式能够更有效地描述程序过程调用中的顺序结构，但是没有充分考虑到大量存在的选择结构和循环结构，而且基于 V 模式库的检测算法由于存在前向的预匹配，检测开销较大。本文提出的 GV 模式充分考虑了程序过程调用的 3 种基本结构，能够更加高效地描述程序流程。基于 GV 模式的系统调用流程图匹配算法在保持高检测率的前提下降低了误检率，且具有较小的检测模型规模和极低的检测开销，能够用于事件序列的在线异常检测。

参 考 文 献

[1] Forrest S, Hofmeyr S A, Somayaji A, et al. A sense of self for Unix processes[C]// Proceedinges of the IEEE Symposium on Research in Security and Privacy. 1996: 120-128

[2] Hofmeyr S A, Somayaji A, Forrest S. Intrusion Detection System Using Sequences of System Calls[J]. Journal of Computer Security, 1998, 6(3): 151-180

[3] Wespi A, Dacier M, Debar H. An intrusion detection system based on the Teiresias pattern discovery algorithm[C]// EICAR Proceedings. 1999

[4] Wespi A, Dacier M, Debar H. Intrusion detection using variable-length audit trail patterns[C]// Third International Workshop on the Recent Advances in Intrusion Detection (RAID 2000). Toulouse, France, 2000: 110-129

[5] Rigoutsos I, Floratos A. Combinatorial pattern discovery in biological sequences[J]. Bioinformatics, 1998, 14(1): 55-67

[6] Parida L, Rigoutsos I, Floratos A, et al. Pattern discovery on character sets and real-valued data: linear bound on irredundant motifs and an efficient polynomial time algorithm[C]// Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA). 2000: 297-308

(下转第 143 页)

v_D)为低空平台的北向、东向和地向速度；(q_0，q_1，q_2，q_3)是用四元数表示低空平台的姿态；b_a 和 k_a 分别表示加速度计的固定偏差和刻度因子；b_g 和 k_g 分别表示陀螺仪的固定偏差和刻度因子。

式(5)中的系统噪声 W_{INS} 为：

$$W_{INS}=[w_{ax}^b, w_{ay}^b, w_{az}^b, w_{rx}^b, w_{ry}^b, w_{rz}^b, w_{gx}^b, w_{gy}^b, w_{gz}^b]^T \quad (7)$$

其中，w_a 是加速度计噪声项，w_r 和 w_g 均是陀螺仪噪声项；其中 w_r 为一阶马氏过程，w_r 和 w_g 均为高斯白噪声。

将 GPS 接收单元和惯性导航单元(INS)组合后，将式(5)作为组合导航系统的状态方程，用 GPS 的模型作为测量方程，即：

$$Z=HX_{INS}+V \quad (8)$$

其中，$H=[I_{10\times10}, 0_{10\times15}]$，$V$ 是系统测量噪声。

将组合导航系统测量得到的各种参量送入机载导航主控系统，结合微波探测火场中心的位置信息，经过卡尔曼滤波器和数据融合后对低空平台执行相应的引导导航与飞行控制，确保低空平台在作业时保持近似“水平”状态来探测森林火场的微波辐射。

4 低空微波探火的应用分析

低空平台森林上空飞行时，根据微波探测接收系统接收到的微波辐射强度和波长来确定森林火灾是否存在、火场的大小以及位置。根据文献[9]可知，森林着火后，上空微波辐射的强度增加，而且辐射亮度温度增加大小与火灾的种类有关。急进地表火的燃烧区，辐射亮度温度增加不大，这是因为火蔓延快，仅燃烧地表枯叶和枯草，释放能量低，也有林冠对微波辐射的屏蔽作用。地表火发生在疏林地和无林地，辐射亮度温度增加。稳进地表火和地下火的燃烧比较彻底，释放能量多，强度大，温度增加，辐射亮度温度增加较大。林冠火烧毁整个林区，燃烧更彻底，强度更大，具有最高的辐射亮度温度。

一般情况下，根据接收到的较高的辐射亮度温度来发现森林火场。在相同条件下，这种可能性还与微波探测接收系统的灵敏度、无线方向图上波瓣的宽度(解象力的大小)和对火灾林区的探测时间长短有关。第 3 节中的微波探测接收系统具有较高的灵敏度。解象力与波长成反比，与天线直径成正比。因此，增加解象力就要在选定波长时尽量选用较小波长和增加天线的直径。对森林火灾区域的探测时间与飞机的飞行高度和速度有关，在可能的情况下应尽量增加对地面的探测时间。

微波探测接收系统搭载在地空飞行平台上，其工作方式有扫描和不扫描两种。扫描工作方式是微波探测接收系统的天线垂直与航线连续变换角度，按一定探测宽度在地面形成扫描带。将微波探测接收系统接收到的辐射亮度温度数值转换为图像输出，得到辐射热图。发生森林火灾时，可在热图上出现热点、热区。所以，该系统可用于发现林火、拍摄火场、计算火灾面积等，而且应用微波探测接收系统可拍摄到浓烟下的大火灾。不扫描工作方式是微波探测接收系统的天线固定一个角度，随着低空飞行平台飞行观测区呈线状。微波探测接收系统输出信号可显示在示波器上并打印出来。这种比较适合于森林火灾面积的测量。

结束语 微波技术的一个主要特点在于微波探测的灵敏度受火场中浓烟和水蒸气的影响比较小，具有红外光学不可比拟的优势。考虑到微波火灾探测灵敏度问题，在微波探测接收系统的设计上采用宽频阵列天线等技术。低空平台是应用研究中的一个重要载体，姿态控制要有一个精度相对较高的动态控制过程，因此需要组合导航系统来确保低空平台的稳定悬停和实时调整姿态，并以此来探测火焰的微波辐射。

近年来，国内外研究发展了针对森林火灾探测的诸多新型技术，这些是实现林火监测数字化和智能化控制的重要发展趋势。本文分析和研究的微波探测火灾技术由于受到火场复杂信号的限制，需要做更深层次的技术分析与定量化试验研究。后续的研究工作将配备一个地面微波试验系统，对诸多地被物的微波性能进行定量探测，即微波探测接收系统搭载在低空飞行平台上，飞行高度在 200～500 米对地面的沼泽、森林火焰、湿森林、干森林等做微波探测特性曲线研究。

参 考 文 献

[1] 杨亮，庄爽，马建明，等. 浅谈微波技术应用于火灾探测[J]. 消防科学与技术，2006，25(1)

[2] 吴龙标，方俊，谢启源. 火灾探测与信息处理[M]. 北京：化学工业出版社，2006

[3] Kempka T, Kaiser T, Solbach K. Microwaves in fire detection [J]. Fire Safety Journal, 2006, 41: 327-333

[4] Kaiser T, Kempka T. Is *microwave radiation useful for fire detection*? [C]// Proceedings of 12th international conference on automatic fire detection AUBE'01. vol. 965, Gaithersburg: NIST Special Publication, March 2001

[5] Pozar D M. Microwave and RF wireless systems [M]. New York: Wiley, 2001

[6] Derbel F. Wireless communication for alarm systems with short range radio[C]// International Conference on Automatic Fire Detection AUBE'04. 2004

[7] 吴龙标，袁宏永. 火灾探测与控制工程[M]. 合肥：中国科学技术大学出版社，1999

[8] 姬渊，秦志远，王秉杰，等. 小型无人机遥感平台在摄影测量中的应用研究[J]. 测绘技术装备，2008，10(1)

[9] 姚树人，文定元. 森林消防管理学[M]. 北京：中国林业出版社，2002

(上接第 90 页)

[7] Long Jidong, Schwartz D G, Stoecklin S, et al. Application of Loop Reduction to Learning Program Behaviors for Anomaly Detection[C]// ITCC. 2005: 691-696

[8] Jonassen I. Efficient discovery of conserved patterns using a pattern graph[J]. Comput Appl Biosci, 1997: 509-522

[9] Sekar R, Bendre M, Bollineni P, et al. A Fast Automaton-Based Method for Detecting Anomalous Program Behaviors [C]// IEEE Symposium on Security and Privacy. Oakland, CA, 2001: 144-155