基于 HMM 的系统调用异常检测

闫 巧[1,2],谢维信[1],宋 歌[2],喻建平[1]
(1. 深圳大学信息工程学院,广东深圳 518060;2. 西安电子科技大学电子工程学院,陕西西安 710071)

摘 要: 我们利用隐马尔可夫模型来描述特权进程正常运行时局部系统调用之间存在的规律性. 具体方法是将 UNIX 特权程序的系统调用轨迹通过隐马尔可夫模型处理得到系统状态转移序列,再经滑窗后得到系统状态转移短序列. 初步的实验证明这样得到的系统状态转移短序列比 TIDE 方法提出的系统调用短序列能更加简洁和稳定地表示系统的正常状态,采用这种状态短序列建立的正常轮廓库比较小,而且对训练数据的不完整性不太敏感. 在同等的训练数据下,检测时本方法比 TIDE 方法的检测速度快,虚警率低.

关键词: 入侵检测; 异常检测; 隐马尔可夫模型; 系统调用; 正常轮廓
中图分类号: TP393 **文献标识码**: A **文章编号**: 0372-2112 (2003) 10-1486-05

System Call Anomaly Detection Method Based on HMM

YAN Qiao[1,2], XIE Wei-xin[1], SONG Ge[2], YU Jian-Ping[1]
(1. *Institute of Information Engineering, Shenzhen University, Shenzhen, Guangdong* 518060 *China*;
2. *Institute of Electronic Engineering, Xidian University, Xi'an, Shaanxi* 710071 *China*)

Abstract: An anomaly intrusion detection method based on a HMM is given. We pass the system call trace of unix privileged process into a HMM to get state transition sequences. Preliminary experiments prove the state transition sequences can express the different mode between normal action and intrusion behavior more stably and more simply than the short sequence in TIDE can do. Although building a HMM is computationally expensive, we can get three advantages, that is, smaller profile database, needing smaller training data, and greater difference between normal data and abnormal data. So we can detect more quickly and with lower false positive rate.

Key words: intrusion detection; anomaly detection; HMM; system call; normal profile

1 基于系统调用的异常检测

入侵检测技术主要采用两种方法即滥用检测和异常检测. 其中异常检测因不需要具有入侵的先验知识,且具有检测新型攻击的能力,所以倍受入侵检测研究者的关注. 1996 年由新墨西哥大学的 FORREST 等人引入了一个简单的异常检测方法叫做时延嵌入序列(TIDE)方法,它是根据 FORREST 等人在反复实验中所观察到的两个重要现象而提出的一种基于实验的方法,这两个现象是[1]:

(1) 在正常操作时,程序执行的系统调用序列是局部连续的;

(2) 当程序的安全漏洞被利用时,将会执行一些不常见的系统调用短序列.

第一个现象之所以产生是因为绝大多数程序的代码是静态的,系统调用在代码的固定位置发生. 程序中的条件语句和函数调用可能改变系统调用的相对顺序,但不会影响小范围系统调用的相关性. 第二种现象之所以发生,是因为如果运行程序的代码被入侵者替代,则很可能执行未在正常数据库中出现的系统调用序列. 另外,很有可能一个成功的入侵者需要 fork 新的进程以利用系统的资源[1].

由这两个现象他们推论进程的正常执行轨迹和异常执行轨迹有着不同的行为模式,并且可以用系统调用的短序列来表征. 所以可以通过监视由特权进程使用的系统调用来完成入侵检测. 他们的具体方法是通过列举出正常执行轨迹中所有唯一的,预先给定长度为 K 的短序列来构造程序正常行为轮廓. 当选择序列长度为 K 时,将长度为 K 的窗口通过每个正常轨迹,一次滑动一个系统调用,向正常轮廓库中添加唯一的序列. 检测时,来自检测轨迹的序列与正常数据库轮廓中的序列相比较,在数据库中找不到一样的序列叫做不匹配. 任何一次不匹配都说明该序列是没包括在正常训练数据库轮廓的序列,它可能是异常行为. 通过计数不匹配的次数,并求出不匹配次数占待检测轨迹中总序列的百分比,再将这个百分比与预先给定的阈值相比较,就可以判断程序此次执行是正常还是异常[1,2]. 该项技术能够检测到对 UNIX 的程序如 SENDMAIL、LPR、FTPD 等的常见的攻击[1].

虽然他们提出的方法非常简单. 但由于直接从原始审计

收稿日期:2001-12-17;修回日期:2003-07-21
基金项目:国家"863"项目(No. 2001AA142100B)

数据的层次建立正常轮廓,所以在一定程度上影响了所建立的异常检测系统的性能.比如,采用他们提出的方法建立的 sendmail 的正常轮廓包含上千个短序列.虽然对于当前机器的性能,其存储量在存储方面的问题不大.但较大的正常轮廓会显著降低检测时的性能.而且由于在建库时不可能考虑到所有可能发生的正常操作,所以建立的正常轮廓必然是不完整的.而对于不完整的训练数据,TIDE 方法非常敏感,随着训练数据完整性的降低,其正常轮廓库的完备性会显著下降从而引起检测时比较高的虚警率.

从入侵检测技术的发展可以看出,若能从原始的审计轨迹提取比原始数据更抽象的入侵特征或正常行为的高层次模式,将会显著提高入侵检测的鲁棒性和灵活性.比如滥用检测的经典方法 STAT 从原始审计数据中提取特征动作描述入侵[4],而著名的异常检测系统 IDES 从用户行为审计中提取概率分布来描述正常行为[5].所以我们致力于研究从特权程序的系统调用轨迹中提取更高层次的抽象模式,从而建立正常进程更简洁和稳定的表述模式,改善原有的异常检测系统的检测性能.

2　隐马尔可夫模型

根据 FORREST 等人的方法可以看出:程序正常运行时的系统调用轨迹的局部系统调用之间具有一定的规律性,而对该程序的某些攻击会破坏这种规律性.系统调用短序列虽然能够在一定程度上表现出程序正常行为模式和异常行为模式的差异,但也许并不是正常行为最合理最有效的表示.我们希望能够透过短序列所表现出的差异这一现象,找到正常行为模式和异常行为模式之间更为本质的区别,从更抽象的层次描述程序正常运行时局部系统调用之间存在的规律性,从而克服 TIDE 方法直接从原始审计数据建立轮廓而导致缺乏灵活性和鲁棒性的缺点.考虑到隐马尔可夫模型能够根据观察到的序列,揭示出隐含的状态序列,所以我们借助隐马尔可夫模型来帮助我们实现上述目的.

我们之所以考虑采用隐马尔可夫模型是基于以下原因:

(1) 从更本质的角度上讲,实验中所观察到的程序正常运行时局部系统调用存在的连续性和规律性可以用一种统计概率模型来表征.因为它实际上表明在程序正常执行时,某一种系统调用将以某种概率出现在另一系统调用之后;

(2) 系统调用轨迹是一组时变的离散时间数据序列,而隐马尔可夫模型是描述离散时间的数据样本序列的一种强有力的统计工具,具有处理非线性时变信号的能力,表示一种双随机过程.所以我们采用它来描述程序正常运行时局部系统调用之间存在的统计规律是很自然的;

(3) 系统调用是操作系统与应用程序之间的接口,它在一定程度上反映了应用程序利用系统资源的情况或系统当前的状态.虽然绝大多数操作系统的系统调用数都在 100 以上,比如 SUN OS 的系统调用数为 182 个,但系统的状态数可能会远远小于这个数.我们希望利用 HMM 可以从系统调用轨迹这个观察序列中揭示出隐含的系统状态.若系统的状态数比较小,我们据此建立的正常轮廓库就会比 TIDE 方法更小,而且会更具有概括性,因而对正常训练数据的完整性要求会降低;

(4) 根据已有的研究结论[8],数据规律性的不同将会影响检测器的性能.对于具有时间上的序列特征的数据,可以采用条件熵来衡量这种规律性.

令 $X=(e_1,e_2,\cdots,e_n)$,令 $Y=(e_1,e_2,\cdots,e_k)$,其中 $k<n$

定义条件熵为:

$$H(X/Y)=\sum_{x,y\in C_x,C_y}P(x,y)\log_{P(x/y)} \tag{1}$$

则该条件熵越小表示不确定性越小,根据这样的数据集合建立的模型的准确性越好[8].因此,我们若能够基于隐马尔可夫方法建立比系统调用短序列的条件熵更小的数据,我们将会得到更高的检测性能.

为了简化计算,我们这里采用的是离散化输出的一阶隐马尔可夫模型.至于隐马尔可夫模型的结构我们采用全连接的拓扑结构,虽然隐马尔可夫模型在语音和手写体识别经常采用从左到右的结构,但那是由于其应用背景的物理性质导致的一种简化.而在我们的方法中隐马尔可夫状态的含义是一种隐含的系统状态,这些系统状态之间都有可能发生转移,所以我们认为采用全连接的拓扑结构更合适.

隐马尔可夫模型 $\Phi=(A,B,P)$ 由两部分组成:第一部分是马尔可夫链发生器,由 $\{P,A\}$ 描述,其输出是状态序列,第二部分是随机过程发生器,由 B 描述,其输出是观察序列.具体参数如下[3]:

(1) $O=\{o_1,o_2,\cdots,o_M\}$ 表示输出观察字母表.在我们的问题中即表示进程执行轨迹中可能出现的系统调用;

(2) $\Omega=\{1,2,\cdots,N\}$ 表示状态空间的一组状态.我们希望这组状态能成为程序正常行为模式更稳定的表述;

(3) $A=\{a_{ij}\}$ 表示状态转移矩阵即 $a_{ij}=P(s_t=j/s_{t-1}=i)$;

(4) $B=\{b_{ij}(k)\}$ 表示输出的概率矩阵即 $b_{ij}(k)=P(X_t=o_k/s_{t-1}=i,s_t=j)$;

(5) $P=\{\pi_i\}$ 初始状态分布即 $\pi_i=P(s_0=i)\ 1\leq i\leq N$.

3　利用隐马尔可夫模型进行异常检测

我们利用隐马尔可夫模型进行异常检测的思想是根据正常进程系统调用轨迹训练一个隐马尔可夫模型,利用该隐马尔可夫模型分别计算出正常进程系统调用的最可能的状态转移序列(经处理后作为正常轮廓)和待检进程系统调用的最可能的状态转移序列,经模式匹配来判断是否有入侵.

定义 1　(状态转移事件)即隐马尔可夫模型定义的状态,用 q_i 表示$\{0\leq i\leq n\}$,其中 n 为隐马尔可夫模型的状态数.

定义 2　(状态转移事件向量)设 k 为滑窗大小,$q_0,q_1,q_2,\cdots,q_m\ (m>k)$ 为进程的系统调用序列通过训练好的隐马尔可夫模型得到的最佳状态转移序列,用滑窗在最佳状态转移序列上滑动,则令 $f(k;q_0,q_1,\cdots,q_m)$ 表示所有在滑窗 k 下的状态转移事件向量集合,即

$$f(k;q_0,q_1,\cdots,q_m)=\{(q_0,q_1,\cdots,q_{k-1}),(q_1,q_2,\cdots,q_k),\cdots,(q_{m-k+1},q_{m-k+2},\cdots,q_m)\} \tag{2}$$

其中包括的每一个序列称作状态转移短序列或状态转移事件

向量 μ,如$(q_0, q_1, \cdots, q_{k-1})$等.

定义 3 (正常状态转移事件向量)定义任何由正常进程的系统调用序列经隐马尔可夫模型得到的状态转移事件向量,记为 N.

定义 4 (正常轮廓库)定义所有由正常进程的系统调用序列经隐马尔可夫模型得到的唯一状态转移事件向量集合,记为 $NL=(N_1, N_2, \cdots, N_l)$,其中 l 为正常轮廓库的大小,且当 $i \neq j$ 时有 $N_i \neq N_j$.

基于隐马尔可夫异常检测的过程框图如图 1 所示. 整个异常检测系统包括两部分:离线训练部分(如图 1 虚线所围部分所示)和在线检测部分(如图 1 虚线外部所示).

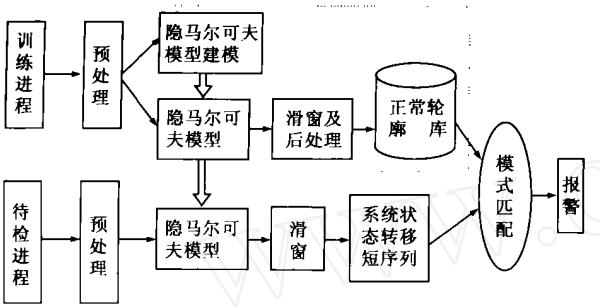

图 1 基于 HMM 异常检测过程图

3.1 离线训练部分

离线训练部分主要目的是建立进程的正常的轮廓库,耗时较长,一般可离线运行. 具体包括以下几步:

(1) 首先收集训练进程的系统调用轨迹. 这些轨迹由进程正常运行时产生,可由三种方法构造. 第一种方法是人工构造,为了执行尽可能多的进程执行路径,可采用软件开发人员用于检测可由进程执行的所有不同子命令的 FVT(功能效验测试)的方法来构造进程. 第二种方法是在实际环境下收集所有进程,但要尽量保证收集阶段没有攻击发生. 第三种方法可以混合采用前两种方法;

(2) 预处理. 把由前一步得到的系统调用轨迹(通常是多个进程的系统调用轨迹按进程执行的时间先后存储在一起)通过数据预处理模块,得到所需的单进程的系统调用轨迹;

(3) 训练隐马尔可夫模型. 将一部分预处理得到的进程的正常系统调用轨迹作为训练数据,训练得到一个隐马尔可夫模型 $\Phi=(A, B, P)$. 隐马尔可夫模型训练算法采用 BAUM-WELCH 算法[3].

- 初始化:选择一个初始估计 Φ;
- E 步骤:计算基于 Φ 的辅助 Q 函数 $Q(\Phi, \Phi)$;
- M 步骤:根据重估方程最大化辅助 Q 函数,来计算 Φ;
- 叠代:令 $\Phi=\Phi$,从 E 步骤重复直到收敛;

(4) 通过 HMM 模型,得到最佳状态转移序列. 将全部的预处理得到的进程的正常系统调用通过训练好的隐马尔可夫模型,用 VITERBI 算法[3,7]求出最佳状态转移序列. 因为状态转移序列是在当前的马尔可夫模型条件下,可能出现所表现的系统调用的最可能的系统内部状态转移,所以我们相信它比系统调用轨迹具有更大的稳定性,因而能更准确的描述进程正常运行时的规律.

- 初始化:$V_0(i)=\pi_i \quad 1 \leq i \leq N$

  $B_0(0)=0$
- 归纳:$V_t(j)=\max\limits_{1 \leq i \leq n} \lfloor V_{t-1}(i)\, a_{ij} b_{ij}(X_t) \rfloor$

  $1 \leq t \leq T; \quad 1 \leq j \leq N$

  $B_t(j)=\operatorname{argmax}\limits_{1 \leq i \leq N}[V_{t-1}(i)\, a_{ij} b_{ij}(X_t)]$

  $1 \leq t \leq T; \quad 1 \leq j \leq N$
- 终止:最佳分值 $=\max\limits_{1 \leq i \leq N}[V_t(i)]$

  $s_t=\operatorname{argmax}\limits_{1 \leq i \leq N}[B_t(i)]$
- 回溯:$s_t=B_{t+1}(s_{t+1}) \quad t=T-1, T-2, K, 0$

  $S=(s_0, s_1, s_2, \cdots, s_T)$ 是最佳序列;

(5) 建立正常轮廓库. 将状态转移序列经滑窗处理,用类似 TIDE 的方法将长度为 K 的窗口通过每个正常状态转移序列,一次滑动一个状态,向正常轮廓库中添加唯一的状态短序列即状态转移事件向量构成正常轮廓库 NL.

3.2 在线检测部分

在线检测部分实时检测可能出现的入侵. 在线检测部分与离线训练部分的工作步骤类似,只是不需对 HMM 建模,而是直接使用训练部分已训练好的 HMM 模型. 首先将待检进程通过预处理模块进行预处理,并将输出的结果通过 HMM 模型,然后利用 VITERBI 算法求出状态序列,将状态序列通过滑窗得到状态短序列即状态转移事件向量集合 $f(k; q_0, q_1, \cdots, q_m)$(与构建正常轮廓库不同的是,这里不能去除相同项).

最后进行匹配,匹配的算法有很多,我们目前采用最简单的一种即将由待检数据得到的状态转移事件向量集合 $f(k; q_0, q_1, \cdots, q_m)$ 中的每一个状态转移向量 μ 与正常轮廓库 NL 中的每一个正常状态转移向量 N 进行完全匹配,若能在库中找到完全相同的序列,则认为待检状态短序列是匹配的,否则判为一次不匹配,即:

对任意状态转移向量 $\mu \in f(k; q_0, q_1, \cdots, q_m)$,若 $\mu \in NL$ 则认为待检状态短序列是匹配的,若 $\mu \notin NL$ 则统计一次不匹配.

当待检进程得到的状态转移向量与正常轮廓的不匹配数目超过预先给定的阈值时认为待检进程是异常的.

这种匹配方法类似于 FORREST 提出的 TIDE 方法. 但不同的是我们比较的是系统状态转移事件向量而非系统调用的短序列. 由于状态转移事件向量来自 HMM 模型得出的最优状态序列,例如对于正常的 sendmail 轨迹由状态转移事件向量集合 $f(k; q_0, q_1, \cdots, q_m)$ 计算得到的相对条件熵为 0.084,而由系统调用的短序列集合计算得到的相对条件熵为 0.13[8],说明经过隐马尔可夫处理过的数据不确定性更小,它能够更简洁和稳定地表示系统的正常状态.

4 实验仿真:

4.1 数据来源

为利于与其他方法比较,我们使用新墨西哥大学的 SENDMAIL、LPR 等数据. 该数据可从 http://www.cs.unm.edu/~immsec/data-sets.htm 下载.

4.2 参数选取

创建一个隐马尔可夫模型时,首先要确定的两个参数就

是观察空间和状态空间. 对我们的实验而言,状态空间的选取尤为重要,因为它选取的好坏密切关系到采用 HMM 后存储空间的大小和检测效果的好坏.

(1) 观察空间 O

将系统调用序列作为 HMM 观察值. 由于 SUN-OS 的系统调用共有 182 种,因此 HMM 的观察空间 O 的大小取为 182;

(2) 状态空间 Q

运用 HMM 的主要目的之一是希望能用较少的隐含状态来表示观察到的系统调用序列,用隐含状态序列来反应系统调用的变化趋势,从而将其特征记忆在状态序列中. 也就是说,将系统调用的变化特征浓缩到 HMM 的隐状态序列中,从更抽象更概括的层次描述正常程序行为. 以我们分析的 sendmail 特权程序为例,其中总共出现了 43 种系统调用,因此状态数 Q 应该小于 43 才能满足我们的要求. 实验中,我们分别取状态数为 10、15、20、25、30、35、40,并对每种状态数对应的单进程数据库的大小进行了比较分析,其部分结果如下图 2 所示. 实验证明,状态数取 15 时大部分进程得到的 HMM 状态序列的特征库最小,当 Q 值超过 40 以后特征库的大小就接近 TIDE 的特征库,并且基本不再变化. 图中画出的是几个典型的单进程特征库随状态数变化的情况,可以看出,对于多数单进程,状态数等于 15 时 HMM 序列数都出现了谷点,所以在后面的实验中我们的 Q 值取 15;

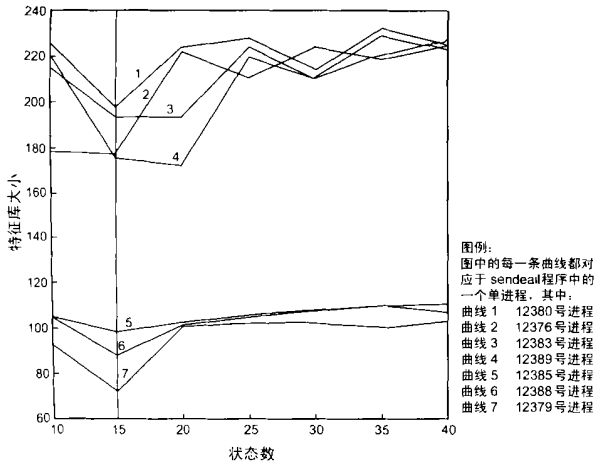

图 2　不同状态下单进程的 HMM 状态序列特征库

(3) 初值

状态转移矩阵随机初始化,观测矩阵利用一定的先验知识进行初始化,具体说来给经常出现的系统调用如 OPEN、CLOSE 等赋予较大的概率值,对其余的系统调用随机赋初值;

(4) 窗口大小

根据 Wenke Lee 的研究[8],从条件熵和计算成本的综合角度考虑,将窗口选择为 6,由于 Forrest 的试验窗口也选为 6[1],所以选择窗口大小 $k = 6$ 也便于与其方法进行比较.

4.3　实验及结果

(1) 观察库的大小以及训练数据完备性对正常轮廓库的影响

我们采用 TIDE 方法和 HMM 方法分别建库,观察所用正常训练数据的量对我们库的影响. 具体实验结果如表 1 所示. 首先我们发现,用所有正常数据做训练建立的完备的正常轮廓库的大小有显著差异. TIDE 方法大小为 1298 个记录,HMM 方法的大小为 861 个记录. HMM 方法建立的库仅为 TIDE 方法建立的库的 66.3 %. 而且我们还发现仅用 30 %左右的正常训练数据,我们的 HMM 方法就能够得到比较稳定的正常轮廓库,基本上为用全部正常数据训练所得到的数据库的 90.7 %. 而同样的数据得到的 TIDE 的轮廓库非常不完备,仅占用全部正常数据训练所得到的数据库的 66.6 %. 我们以训练数据的完整性为横轴,以所得到的正常轮廓库的完整性为纵轴比较两种方法如图 3 所示. 从中我们可以看到,HMM 方法具有比 TIDE 方法更强的稳定性,即使在正常训练数据不充分时也能得到近似完备的正常轮廓数据库;

(2) 对比检测效果

我们用 70 %的 sendmail 正常进程系统调用作为训练数据来建立正常轮廓,然后用该轮廓来检测程序的四种异常使用. 虽然使用不同的 70 %数据进行训练,所得的实验结果有一定差异,但由 HMM 方法得到的结果总是比由 TIDE 得到的结果好,我们给出其中一次实验的结果,如表 3 所示. 实验表明,我们的方法能够检测到入侵,而且对异常,我们得到的不匹配率比 TIDE 方法得到的不匹配率高,而且因为我们的方法库比较小,检测时速度更快. 我们用 70 %的正常的 SENDMAIL 数据作为训练数据,用剩下的正常数据作为检测数据得到的结果如表 2 所示. 可以看出我们的方法对正常轨迹的不匹配率一般比 TIDE 方法低.

表 1　库的大小和建库时间比较

	建库时间($K=6$) 100 %的正常数据	库的大小($K=6$) 15 %的正常数据	库的大小($K=6$) 30 %的正常数据	库的大小($K=6$) 70 %的正常数据	库的大小($K=6$) 100 %的正常数据
HMM 方法	320 分钟	681	781	832	861
TIDE 方法	38 分钟	732	864	945	1298

表 2　用 70 %的正常数据训练,30 %的正常进程的最大不匹配率

	正常进程的最大不匹配率(%)					
	Bouce	Bouce1	Bouce2	Plus	Queue	Sendmail
HMM 方法	14.65	19.71	34.46	1.11	0.305	13.76
TIDE 方法	14.52	24.37	39.29	0.925	0.37	15.75

表 3　用 70 %的正常数据训练,对异常进程的最大不匹配率

	异常进程的最大不匹配率(%)					
	Syslog-local-1	Syslog-local-2	Syslog-remote-1	Syslog-remote-2	Cert-sm565a-1	Cert-sm5x-1
HMM 方法	94.4	99.1	99.2	88.3	83.3	88.2
TIDE 方法	56.1	58.2	52.6	78.5	37.2	54.7

表 4 不同特权程序的 HMM 状态序列数据库和 TIDE 方法数据库的比较

程 序 名	TIDE 数据库大小(100 %)	HMM 数据库大小(100 %)
FTP	613	440
LPR	178	170

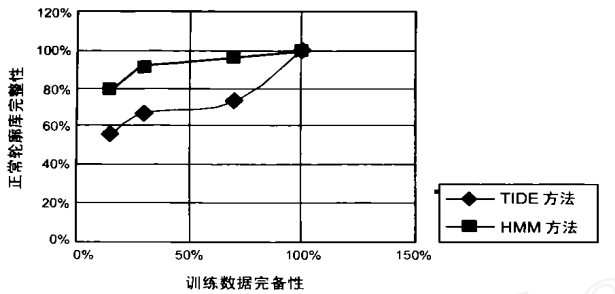

图 3 TIDE 方法与 HMM 方法的训练数据完备性对正常轮廓库的影响比较

5 结论及进一步研究方向

我们对其他 UNIX 特权程序 FTP 和 LPR 做实验也有类似结果,如表 4 所示. 初步的实验证明将 UNIX 特权程序的系统调用轨迹通过隐马尔可夫模型处理得到系统状态转移事件比 TIDE 方法提出的系统调用短序列能更加简洁和稳定地表示系统的正常状态,采用这种短序列作为异常检测的基本量度有更大的精确度. 虽然在我们的方法提出之前,隐马尔可夫模型曾经被应用于系统调用短序列的异常检测[11],但他们采用的是直接利用隐马尔可夫模型作为有限状态机进行概率估计,他们的方法在学习时耗时非常长,如对 LPR 系统特权程序的训练时间多达 5 天,而我们的方法本质上是用训练好的隐马尔可夫模型来过滤数据,使得数据具有更好的规律性,所用的训练时间(对最复杂的包含系统调用数目最多的 SENDMAIL 程序也只需要五个多小时). 所以基于隐马尔可夫的系统调用异常检测方法更加切实可用.

具体说来,我们的 HMM 方法得到的好处有以下三点:

(1) 所建立的正常轮廓的数据库比 TIDE 方法建立的数据库要小,HMM 方法建立的库仅为 TIDE 方法建立的库的 66.3 %. 因此用 HMM 方法检测时速度更快;

(2) 在建立库的过程中,用比较少的训练数据就能得到近似完备的正常轮廓数据库. 用 30 %的正常数据建立的 TIDE 的正常轮廓数据库仅为用全部正常数据建立 TIDE 正常轮廓数据库的 66.6 %,而用 30 %的正常数据建立的 HMM 的正常轮廓数据库为用全部正常数据建立 HMM 数据库的 90.7 %;

(3) 从实验结果上可以看出,HMM 方法在检测异常进程时所得到的最大不匹配率比 TIDE 方法要大,而在检测正常进程时最大不匹配率一般比 TIDE 低.

当然所得到的这些好处是有代价的,这就是 HMM 在建库时所需的时间几乎为 TIDE 的 8 倍,但即使是这样比起其他基于隐马尔可夫的方法,这个学习时间也是可以接受的. 而且我们通常可以离线建立轮廓库,从而换取检测时的高效等以及在建立库的训练数据不充分时仍能保证轮廓库较高的完备性等优点.

参考文献:

[1] Stephanie Forrest, Steven, A. Hofmeyr, Anil Somayaji. A Sence of Self for Unix Processes[A]. IEEE Symposium on Security and Privacy[C]. Oakland, California IEEE Computer Society, 1996. 120 - 128.

[2] Christina Warrender, Stephanie Forrest, Barak Pearlmutt. Detecting Intrusions Using System Calls: Alternative Data Model [A]. 1999 IEEE Symposium on Security and Privacy[C]. 1999. 133 - 145.

[3] R Dugad, U B Desai. A Tutorial on Hidden Markov Models [OL] url. http://vision.ai.uiuc.edu/dugad/guestbook/addHMMguest.html.

[4] Koral Ilgun, Richard A. Kemmerer, Phillip A. Porras. State transition analysis: a rule-based intrusion detection approach[J]. IEEE Trans. on Software Engineering, March 1995, 21(3): 181 - 199.

[5] Teresa F lunt, R Jagannnthan, Menlo Park. A Prototype Real-Time Intrusion-Detection Expert System[A]. 1988 IEEE Symposium on Security and Privacy[C]. 1988. 59 - 65.

[6] Yanqiao. Xie Weixin. Yangbin Songge. An anomaly intrusion detection method based on HMM[J]. Eletronics Leeters, 2002, 38(13): 663 - 664.

[7] 刘海峰,卿斯汉,蒙杨,刘文清. 一种基于基于审计的入侵检测模型及其实现机制[J]. 电子学报 2002, 30(8): 1167 - 1171.

[8] Wenke Lee Dong Xiong. Information-Theoretic Measures for Anomaly Detection[A]. Proceedings IEEE Symposium on Security and Privacy [C]. IEEE Computer Society, Oakland, California, USA. May 14 - 16, 2001.

作者简介:

闫 巧 女,1972 年出生于河南省开封市,博士,讲师,主要研究方向为网络安全,已发表文章十多篇. Email: murongchuchu@hotmail.com.

谢维信 男,1941 年生于广东省花县,博士生导师,深圳大学校长,主要研究方向为智能信息处理.