文章编号:1001 - 9081(2005)08 - 1744 - 03

一种基于隐 Markov 模型的异常检测技术

安景琦,刘贵全,钱 权
(中国科学技术大学 计算机科学技术系,安徽 合肥 230027)
(jqan@ mail. ustc. edu. cn)

摘 要:给出了一种建立隐 Markov 异常检测模型的算法,并从序列支持度分析、序列预测两个方面研究了该模型在异常检测中的应用,通过实验,分析了影响这一检测方法效果和效率的因素。实验表明,该方法能在不需要任何安全方面背景知识的情况下,有效地检测出入侵行为。

关键词:异常检测;隐 Markov 模型;系统调用;滑动窗口
中图分类号: TP309.2 **文献标识码**:A

Anomaly detection technique based on hidden Markov model

AN Jing-qi, LIU Gui-quan, QIAN Quan
(*Department of Computer Science and Technology, University of Science and Technology of China, Hefei Anhui* 230027, *China*)

Abstract: An algorithm was given to making a kind of HMM(Hidden Markov Model) for anomaly detection. The application in anomaly detetction of the model was introduced from analyzing on support of sequence and sequence prediction. Factions which inflect results and efficiency were discussed by the experiments. And the experiments also show that the method can detect intrusion without any security knowledge.

Key words: anomaly detection; hidden Markov model; system call; slide window

0 引言

入侵检测从检测技术上通常分为误用检测和异常检测两种。误用检测采用模式匹配的方法,优点是检测准确率较高,但缺点是对于未知特征的入侵类型没有检测能力,因此在层出不穷的网络攻击手段面前有明显的不足之处;而异常检测通过预先定义一组系统正常情况下的数值,然后将系统运行时的数值与定义的正常情况作比较,得出系统是否有被攻击的迹象[3],这样,当系统遭受未知特征的入侵时,只要发现与正常情况下的系统数值有偏差,就可以检测出来。异常检测技术的难点在于如何定义所谓的正常情况,这对于降低误报率(虚警)具有重要意义,特别是在缺乏相关的背景知识和领域知识的情况下。

文献[4,5]中为计算机系统的运行状态建立 Markov 模型,然后在这个模型的基础上提出了检测算法。文献[6]对 Markov 模型在异常检测中的应用进行了较为全面地研究,但是由于实际系统并不完全符合 Markov 模型的假设条件,建立的模型不能非常准确的反映系统特征,因此在此基础上提出的检测算法的准确性也不理想。本文提出了一种使用隐 Markov 模型(Hidden Markov Model,HMM)建立系统模型,并探讨了两种使用该隐 Markov 模型检测入侵行为的方法,该模型能较为准确的描述计算机系统的状态,通过实验,发现使用这一模型的检测方法能较准确的区分正常和异常行为并检测出入侵,而且算法简单,效率较高。

1 隐 Markov 模型在异常检测中的应用

如前所言,异常检测需要对系统建立一个正常模式,而系统在某种应用下,一般有自己的日常事务,尤其是长期的事务,这恰是异常检测的基础所在。此外这种日常事务通常具有一定的规律,例如系统所处状态的转换,某种状态下系统的行为特征等。隐 Markov 能很好地描述这样的特性。由此可以想到,隐 Markov 模型可以用来对某些系统应用建立正常行为模型,本文将 sendmail 程序运行时的系统调用序列作为观察值序列,为其建立隐 Markov 模型。

隐 Markov 模型是一种双重随机过程,一个是隐含的有限状态 Markov 链,它描述状态的转移,另一个描述状态与观察值之间的统计对应关系。对应的,使用隐 Markov 模型建立的模型也应具有两个序列:一个隐含的状态序列和一个可见的观察值序列。隐 Markov 模型有三个假设:当前状态只与上一个状态有关;状态转移概率与时刻无关;观察值只与当前状态有关。这三个假设大大降低了模型的复杂度,使得隐 Markov 的算法具有较高的效率。

1.1 使用隐 Markov 模型为系统正常模式建模

在缺乏相关背景知识的情况下,需要利用有限的信息来尽可能精确地描述系统的正常运行模式,并且要求建立这个模型的过程要尽可能简单,不需要人为的干预,而隐 Markov 模型自身的特点恰好能满足这一要求。一个可采用的方案是:记录系统正常行为模式下(没有遭受攻击)的一组观察值(这里采用的是系统调用序列),使用 HMM 的学习算法 Baum-Welch 算法,得到系统正常模式的隐 Markov 模型的参数,该模型(以下记为 λ)是一个五元组$(\Omega_X,\Omega_O,A,B,\pi)$:

Ω_X: $= \{q_1,\cdots,q_N\}$:状态值的有限集合;

Ω_O: $= \{v_1,\cdots,v_M\}$:观察值的有限集合;

$A = \{a_{ij}\}, a_{ij} = p(X_{t+1} = q_j \mid X_t = q_i)$:状态转移概率;

收稿日期:2005 - 01 - 29;**修订日期**:2005 - 04 - 07 **基金项目**:国家自然科学基金资助项目(90104030)

作者简介:安景琦(1981 -),男,安徽巢湖人,硕士研究生,主要研究方向:人工智能、网络安全; 刘贵全(1970 -),男,四川彭山人,副教授,博士,主要研究方向:多 Agent 系统、机器学习、网络安全; 钱权(1972 -),安徽安庆人,博士,主要研究方向:人工智能、网络安全.

$B = \{b_{ik}\}, b_{ik} = p(O_t = v_k \mid X_t = q_i)$:观察值输出概率;

$\pi = \{\pi_i\}, \pi_i = p(X_1 = q_i)$:初始状态分布。

实际情况中,系统所处的状态之间的转换并不是任意的,通过一次转移,某一状态可能只能转移到其相邻的几个状态,因此,*left* 和 *right* 两个参数被引入该模型,这两个参数限制了某一状态经过一次转移可到达状态的范围,例如 $left = right = 1$ 时,表示状态 q_i 经过一次转移可转移到 q_{i-1}、q_i、q_{i+1} 三个状态,状态转移概率矩阵 A 的第 i 行中只有 a_{ii-1}、a_{ii}、a_{ii+1} 三个元素不为0,此时矩阵 A 是一个三对角矩阵,其中除了对角线以及对角线上面和下面以外的元素全部为 0。*left* 和 *right* 两个参数的引入不仅使得该模型更符合实际情况,而且降低了模型训练和用于检测时的计算量,这对于实时检测非常重要。

Baum-Welch 算法是一种只知道观察值序列而不知道对应的状态序列的情条件下计算模型的 A、B、π 参数值的算法,是极大似然(Expectation-Maximization, EM)算法的一种实现,引入 *left* 和 *right* 两个参数之后,该算法步骤和各步的计算公式被修正如下:

1) 初始化 λ_0, 设置初始的 A、B、π 值。

2) EM 步骤:循环执行以下步骤,直到 λ_i 收敛。

E- 步骤: 根据 λ_i 计算所有可能的状态序列出现的概率,在 t 时刻位于状态 q_i,$t+1$ 时刻位于状态 q_j 的概率是:

$$\xi_t(i,j) = \frac{P(X_t = q_i, X_{t+1} = q_j, O \mid \lambda)}{P(O \mid \lambda)} \tag{1}$$

在 t 时刻位于状态 q_i 的概率为:

$$\gamma_t = \sum_{j=i-left}^{i+right} \xi(i,j) \tag{2}$$

M- 步骤:根据状态序列和输出序列估计参数 λ_{i+1}。

迭代终止的条件是:$|\log P(O \mid \lambda_{i+1}) - \log P(O \mid \lambda_i)| < \varepsilon$,$\varepsilon$ 是事先给定的阈值。当迭代终止时,即可得到描述系统正常模式的隐 Markov 模型的参数。

1.2 使用隐 Markov 模型分析检测数据

1.2.1 滑动窗口序列

首先使用滑动窗口将待检测的观察值序列划分成为短的序列,一般有数量窗口和时间窗口两种划分方法,使用数量窗口可以将观察值序列划分为固定长度的若干短序列,这有利于检测结果的处理和对比,因此这里采用的是数量窗口。例如,对于系统调用序列 $(v_1, \cdots, v_i, v_{i+1}, \cdots, v_{i+s})$,使用长为 t 的滑动窗口划分可得 $s+1$ 个长为 t 的序列,即:$(v_1, v_2, \cdots, v_t)$;$(v_2, v_3, \cdots, v_{t+1})$;$\cdots$;$(v_{s+1}, v_{s+2}, \cdots, v_{t+s})$。

1.2.2 分析方法

如何使用正常模式尽可能明显的区分出待检测的正常行为模式与异常行为模式,是检测技术的要点。为此,设计了以下两种分析方法:

(1) 使用评估算法计算正常模式的 HMM 对待检测观察值序列的支持度。

对于长度为 T 的观察值序列 $O = (v_1, v_2, \cdots, v_T)$,使用前向递推算法计算其出现的概率 $P(O \mid \lambda)$,定义前向变量 $\alpha_t(i)$ 为:t 时刻输出序列为 $v_1, v_2, \cdots, v_t$ 并且位于状态 q_i 的概率,即:$\alpha_t(i) = P(v_1, v_2, \cdots, v_t, q_i \mid \lambda)$。计算公式如下:

初始化:$\alpha_1(i) = \pi_i b_{iv1}$;

在引入 *left* 和 *right* 参数之后,递推公式为:

$$\alpha_{t+1}(j) = \left[\sum_{i=j-left}^{j+right} \alpha_t(i) a_{ij}\right] b_{jv_{t+1}} \tag{3}$$

最终结果:$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_t(i)$,算法的时间复杂度为 $O(N(left + right + 1)T)$。

为每一个用滑动窗口划分出的观察值序列计算出支持度,若此支持度小于某一阈值,则认为该序列为"异常",统计所有序列中"异常"的比例,可以发现正常和入侵数据中异常短序列的比例有非常明显的差别。

(2) 使用正常模式的 HMM 根据观察值序列进行观察值预测。

对每一个滑动窗口划分的观察值序列 $O = (v_1, v_2, \cdots, v_T)$,预测其下一时刻输出的观察值 v_{T+1} 的分布,并于待检测序列的实际观察值进行比较,如果与预测不相符合的观察值超过一定比例,则认为发现异常。

预测的 $T+1$ 时刻输出为 v^*,则有:$v^* = \arg\max_v P(v \mid O, \lambda)$,即在正常模型下观察值序列 O 之后的下一时刻出现的概率最大的观察值,这一算法需要对 $T+1$ 时刻可能出现的观察值都进行评估,因此时间复杂度是 $O(N^2(left + right + 1)T)$。

由于系统某一时刻的观察值有多个,很多情况下出现概率最大的观察值并不一定总会出现,因此可以适当地放宽判定正常的标准,即当观察值属于出现概率较大的几个观察值的集合时,都认为是正常的。例如,选取出现概率最高的 5 个观察值作为可接受的观察值范围,若 $T+1$ 时刻出现概率从大到小排列的观察值依次是 5,4,6,2,1,$\cdots$,如果出现不属于这五个观察值的输出,则认为是异常的,当异常数目达到某一比例时,触发报警,报告发现人侵迹象。序列预测方法的主要运算(预测步骤)是在新的观察值输人之前完成(在已知输人序列 O 时,即可以预测下一时刻的观察值分布),因此该方法具有较好的实时性,反应迅速。

2 实验及结果分析

实验采用 UNM 和 CERT 提供的数据[6],选取了正常模式下(未发生人侵事件)的一组数据(长度为 7071),其中长为 5000的序列用来学习,其余 2071 用于测试模型,另外还选取了两组测试数据,分别是正常模式下(长 899)和发生人侵行为时(长 1179)的数据。这样,总共就有了三组测试数据记为 normal1, normal2 和 intrusion。

状态数对测试结果有一定影响,状态数太少不能反应系统的复杂度,过多则导致运算所需的时间和空间成本急剧上升。经过试验比较,选定的状态数为 9,模型的 left 和 right 值都是 3。此时使用长度为 5 000 的序列学习建模的时间在 1 000ms 以内(PⅢ800,256M)。

2.1 序列支持度分析

计算出输出值序列出现的概率之后,用来判定异常的阈值对于结果也有较大影响,阈值过大,则大部分序列都会被判定位异常,过小则相反。阈值与异常比例的关系如图 1 所示(滑动窗口长度为 10,测试数据为 normal1)。由于序列出现的概率必然随着序列长度的增加而迅速减小,因此计算出的概率需要经过标准化处理,方法是使用 P' 替代 P,计算公式为:

$P' = P^{1/L}$ （L 为序列长度） (4)

从图 1 中可以明显看出，阈值过大或过小，都会使的两条曲线之间的差别减小，不利于区分正常与入侵数据，在后面的实验中都使用 0.10 作为阈值。

取不同的窗口大小划分待测试数据，然后对测试数据进行分析，得到异常序列占序列总数的比例如表 1 所示。

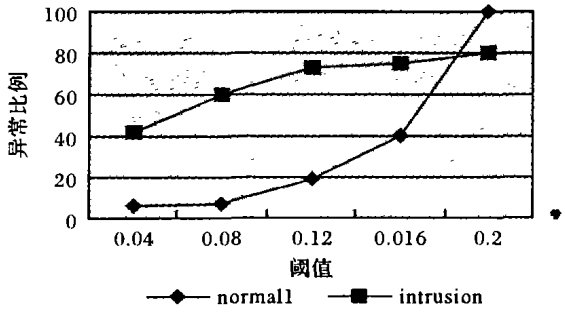

图 1 异常序列比例与阈值的关系

从表 1 可以看出，窗口长度增加对入侵数据的结果影响不大，而正常测试数据中的异常比例随窗口长度的增加有明显减小的趋势，原因一是些偶然因素可能会造成正常序列的中出现个别小概率的观察值，窗口长度太小时，不容易消除这些偶然因素的影响。因此计算结果随窗口长度的增加而变好，但窗口长度的增加会增加计算的时间和空间复杂度，因此，实验发现比较合适的长度为 10。此时 normal1 中异常序列的比例仅为 0.30%，而另一组 normal2 中异常序列的比例为 7.77%，入侵数据 intrusion 中异常序列的比例高达 59.65%，与第一组的差别高达 200 倍，这要优于文献[6]中使用 Markov 模型对同样的数据得出的结果。该分析方法对长为 1000 左右的序列进行分析，所需时间约为 10ms。

表 1 序列支持度分析结果

窗口长度	normal1(%)	normal2(%)	intrusion(%)
5	4.31	17.75	64.42
6	4.29	17.33	57.89
7	4.48	17.23	55.95
8	0.45	12.31	56.75
9	0.40	12.53	67.24
10	0.30	7.77	59.65

2.2 序列预测

将测试数据划分成长度为 30 的若干组数据，对每一个观察值预测其下一时刻可能的观察值分布，与预测不符的观察值，认为是异常观察值，通过计算异常观察值的出现频率，也可以区分正常与入侵数据。

这一分析方法中，异常观察值的判定依据是影响结果的关键因素，表 2 反映的是，不同判定依据下，异常观察值出现频率的变化。

表 2 序列预测分析结果

符合预测的观察值数据	normal2	intrusion
1	0.654	0.986
2	0.578	0.983
3	0.407	0.909
4	0.217	0.878
5	0.125	0.818
6	0.125	0.689

表 2 中最左边一列的数字表示的是判定异常的标准，即可以被认为符合预测的观察值数目，显然，该数目越小，被判定为异常的概率越大。图 2 是根据表 2 中的数据对作的变化趋势图。从图 2 中可以看出，随着判定标准的变化，测试数据中异常观察值出现频率的比值也在变化，选择合适的判定标准，目的就是尽量放大这一比值，可以看出当横坐标为 5 时，比值达到最大。

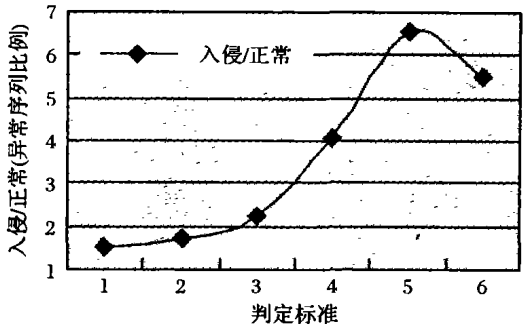

图 2 异常观察值出现概率的变化趋势

3 结语

通过实验可以发现，使用隐 Markov 模型计算正常行为模式对入侵序列的支持度，得出的异常序列的出现频率最高可达正常值的 200 倍，而其他不包含入侵行为的测试数据的这一指标最高是 20～30 倍，采用序列预测的方法进行分析，得到的入侵数据和正常数据的异常观察值出现频率的差别最大可以达到 6～7 倍，这两种分析方法都能很明显的检测出异常行为。第一种方法的时间复杂度较低，并且分析结果也较好；而序列预测方法在实际应用中具有实时性的特点。确定合适隐 Markov 模型的状态数和 *left*、*right* 参数对于改善算法效果、提高运算效率有重要意义。通过分析还发现，滑动窗口长度的增加对结果有好的影响，但会增加运算代价，无论哪种方法，选择合适的阈值和异常判定标准都能明显改善实验结果。

实验中使用的学习数据只是正常模式下的系统调用记录，没有任何关于攻击的先验知识，可见文中使用的隐 Markov 模型及检测方法是一种较好的异常检测模型和检测方法。

参考文献：

[1] ANDERSON JP. Computer security threat monitoring[R]. TechnicalReport, TR80904, Washington: Anderson Co., 1980.

[2] DENNING DE. An intrusion detection model[J]. IEEE Transactions on Software Engineering, 1987, 13(2): 222－232.

[3] ANDERSON R, KHATLOK A. The use of information retrieval techniques for intrusion detection[A]. Web proceedings of the First International Workshop on Recent Advances in Intrusion Detection (RAID 98)[C]. http://www.raid－symposium.org/raid, 1998－03.

[4] YE N. A Markov chain model of temporal behavior for anomaly detection[A]. Proceedings of the 2000 IEEE Systems, Man, and Cybernetics Information Assurance and Security Workshop[C]. http://citeseer.nj.nec.com/ye00markov.html, 2000.

[5] WARRENDER C, FORREST S, PEARLMUTLER B. Detecting intrusions using system calls: alternative data models[A]. Proceeding of the 1999 IEEE Symposium on Security and Privacy[C]. CA: IEEE Computer Society, 1999. 133－145.

[6] 钱权，蔡庆生，安景琦. Markov 链模型在异常检测上的应用研究[J]. 中国科学技术大学学报，2003，33(2).