基于系统调用入侵检测的马氏链模型*

钟安鸣[1]，贾春福[1,2]

(1. 南开大学 信息技术科学院，天津 300071；2. 中国科学院 研究生院 信息安全国家重点实验室，北京 100039)

摘　要：研究了利用马氏链提取基于系统调用序列的进程特征，实现入侵检测的方法；并通过实验说明了此方法的可行性和有效性，实验表明一阶马氏链模型的检测效果要比二阶马氏链模型的检测效果好。

关键词：入侵检测；系统调用；马氏链模型

中图法分类号：TP311.11　　**文献标识码**：A　　**文章编号**：1001-3695(2005)04-0134-03

Markov Chain Model of Syscall-based Intrusion Detection

ZHONG An-ming[1]，JIA Chun-fu[1,2]

(1. School of Information Technology Science, Nankai University, Tianjin 300071, China; 2. State Key Laboratory of Information Security, Graduate School, Chinese Academy of Sciences, Beijing 100039, China)

Abstract: Presents a Markov chain method to characterize process based on syscall sequences for intrusion detection, and demonstrates its feasibility and validity through experiments. The experiment results reveal better intrusion detection performance of the first-order Markov model than that of the second-order Markov model.

Key words: Intrusion Detection; Syscall; Markov Chain Model

随着全球信息化进程的加速发展，计算机网络安全技术越来越受到人们的关注。入侵检测是在访问控制和防火墙等安全措施之后的一道主动防护屏障。自从 Denning[1] 在 1986 年首先提出了一个具有重要应用价值的安全模型之后，入侵检测技术和方法的研究已经取得了很多重要的成果，并出现了一些具有一定实用价值的商业产品。

Forrest[2] 等人在研究过程中发现，程序运行过程中所使用的系统调用序列表现出一些重要的性质，主要包括：①进程运行过程中所使用的系统调用序列具有一定的稳定性，表现为系统调用短序列(系统调用序列的片段)具有一定的稳定性；②刻画系统调用的系统调用短序列的数量变化不会很大，趋向于一个定值；③运用系统调用短序列可以描述系统调用的特征。以上特征表明：系统调用短序列可以用于刻画进程的特征，并可以用于构建进程运行的正常行为特征库，以区分系统中不同进程和入侵行为等。因此，基于系统调用可以建立入侵检测模型。

已有的研究表明：利用系统调用序列构建进程行为特征时，随着进程产生的系统调用序列长度增加，特征库长度也会随之增加，这不利于实现系统的实时性。因此，一个重要的研究课题是：如何建立较为精简而又能准确反映用户行为特征的进程行为模式。由于进程产生的系统调用序列中前后系统调用之间存在一定的依赖关系，并且这种依赖关系具备一定的稳定性，本文利用马氏链模型构建进程行为特征，并通过实验验证了此方法的有效性。

收稿日期：2004-02-27；修返日期：2004-06-10

基金项目：天津市自然科学基金资助项目(013600711，033800611，033600311)；信息安全国家重点实验室开放课题资助项目(03-04-02)

1　入侵检测的马氏链模型

现实情况的系统调用序列中，各个系统调用之间可能存在非常复杂的高阶依赖关系，因此，利用马氏链来刻画系统调用序列特征时，必须要进行一定的理论假设。考虑到对系统调用序列中各系统调用之间依赖关系的不同假设以及实现上的可能性，我们提出了两类马氏链模型。当我们假设序列中仅存在一阶依赖关系，即当前系统调用的发生仅取决于其前一个系统调用时，可以采用一阶马氏链模型；当我们假设序列中存在二阶依赖关系，即当前系统调用的发生取决于其前两个系统调用时，可以采用二阶马氏链模型。本文给出了这两种模型的详细描述，并通过实验比较了它们的效果。

1.1　马氏链模型的结构

(1) 一阶马氏链模型

一阶马氏链模型 λ_1 由三元组构成。$\lambda_1 = (Q, M, N)$，其中，Q 为状态集合，即系统调用序列中出现的全体系统调用集；M 为一阶转移矩阵；N 为系统调用的总数，即 Q 中出现的元素的总数。

(2) 二阶马氏链模型

二阶马氏链模型 λ_2 由四元组构成。$\lambda_2 = (Q, M_1, M_2, N)$，其中，$Q$ 为状态集合，即系统调用序列中出现的全体系统调用集；M_1 为一阶转移矩阵；M_2 为二阶转移矩阵；N 为系统调用的总数，即 Q 中出现的元素的总数。

1.2　基于系统调用的入侵检测马氏链模型

基于系统调用的入侵检测马氏链模型主要由四大模块组成，即预处理模块、马氏链训练模块、马氏链测试模块和结果输出模块，如图 1 所示。

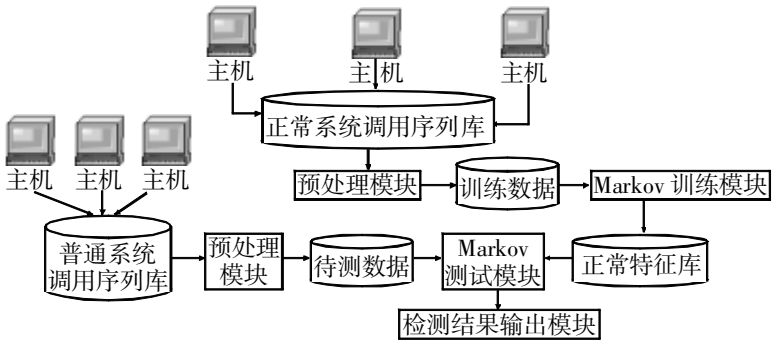

图 1　基于系统调用的入侵检测马氏链模型

1.2.1　预处理模块

预处理此模块负责统计系统调用序列中出现的系统调用总数, 并将系统调用编号以便马氏链模型对其进行处理。编号时, 将各系统调用按照出现的频率排序, 将出现频率高的系统调用排在中间, 将出现频率低的系统调用依次排在两边, 模拟正态分布的情况。经过上述排序后, 各个系统调用的序号就是它们的编号。预处理模块之所以对系统调用采用这种编号原则, 是因为有研究[3]表明, 系统调用序列的频率特性和顺序特性对异常检测都起作用。仅仅用马氏链模型难以刻画系统调用的频率特性, 加上这个预处理模块, 就使我们的模型既能反映系统调用的频率特性, 又能反映系统调用的顺序特性, 从而更加全面地描述系统调用的特征。

1.2.2　马氏链训练模块

马氏链训练模块负责从正常的系统调用序列中提取正常系统调用特征库。通过以下两个步骤进行。

(1) 将系统调用看作状态, 构建马氏链模型, 计算转移矩阵 M

对于一阶马氏链模型, 一阶转移矩阵 M 的计算方法如下:

$$M = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1N} \\ p_{21} & p_{22} & \cdots & p_{2N} \\ \cdots & \cdots & \cdots & \cdots \\ p_{N1} & p_{N2} & \cdots & p_{NN} \end{bmatrix}$$

$$p_{ij} = N_{ij} / N_i$$

其中, p_{ij}为由系统调用 i 转移到系统调用 j 的概率; N_{ij}为系统调用短序列 i, j 在整个系统调用序列中出现的次数; N_i为在存在后继系统调用的条件下, 系统调用 i 在整个系统调用序列中出现的次数。

对于二阶马氏链模型, 一阶转移矩阵 M_1 的计算方法同一阶模型中的 M, 二阶转移矩阵 M_2 的计算方法如下:

$$M_2 = \begin{bmatrix} p_{111} & p_{112} & \cdots & p_{11N} \\ \cdots & \cdots & \cdots & \cdots \\ p_{1N1} & p_{1N2} & \cdots & p_{1NN} \\ p_{211} & p_{212} & \cdots & p_{21N} \\ \cdots & \cdots & \cdots & \cdots \\ p_{N11} & p_{N12} & \cdots & p_{N1N} \\ \cdots & \cdots & \cdots & \cdots \\ p_{NN1} & p_{NN2} & \cdots & p_{NNN} \end{bmatrix}$$

$$p_{ijk} = N_{ijk} / N_{ij}$$

其中, p_{ijk}为在前两个系统调用分别为 i, j 的条件下, 第三个系统调用为 k 的概率; N_{ijk}为系统调用短序列 i, j, k 在整个系统调用序列中出现的次数; N_{ij}为在存在后继系统调用的条件下, 系统调用短序列 i, j 在整个系统调用序列中出现的次数。

(2) 用 Viterbi 算法求最大可能短序列

Viterbi 算法曾广泛用于数字通信中做卷积译码。后来 Omura[4] 证明 Viterbi 算法等价于求通过一个加权图的最短路径问题的动态规划算法。它事实上就是卷积码的最大似然译码算法, 即译码器所选择的输出总是能给出对数似然函数值为最大的码字。可以用该算法求出以某一系统调用开头的最大可能系统调用短序列 O, 以用作入侵检测依据。我们规定这类最大可能短序列的长度统一为 T(T 的最佳取值需根据实验来确定) , 则有 $O = s_1, s_2, \cdots, s_T$, 其中 s_i 为系统调用编号。下面详细说明我们的马氏链模型中用来求最大可能短序列的 Viterbi 算法[5]。

对于一阶模型, 由于

$$P(O \mid \lambda_1) = P(s_1, \cdots, s_T \mid \lambda) = p_{s_1 s_2} p_{s_2 s_3} \cdots p_{s_{T-1} s_T}$$

我们定义 $U(s_1, \cdots, s_T) = -\left[\sum_{t=2}^{T} \ln(p_{s_{T-1} s_T})\right]$

于是, $P(O \mid \lambda_1) = \exp(-U(s_1, \cdots, s_T))$

因此最优状态序列 X 所要满足的条件为: $\max_{\{s_t\}_{t=2}^{T}} P(s_1, s_2, \cdots, s_T \mid \lambda_1) = \min_{\{s_t\}_{t=2}^{T}} U(s_1, s_2, \cdots, s_T)$。我们定义 $-\ln(p_{s_i s_j})$ (即从状态 i 转到状态 j 的概率的负对数) 为从状态 i 转到状态 j 的权值, 于是求最大可能状态序列的问题就转化为求权值最小的状态序列(路径) 的问题, 而求权值最小的状态路径问题可以用 Viterbi 算法解决。这就是我们在定义权值时要对概率取负对数的原因。

Viterbi 算法中引入符号 $\delta(i)$ 表示如果要找的可能最优状态路径在时刻 t 通过状态 i, 这条路径在时刻 t 以前(包括时刻 t) 的累计权值(注意这里是可能最优路径, 最后确定的最优路径在时刻 t 并不一定通过状态 i, 这里有一个取舍的问题, 下面将要说明) ; 引入符号 $\Psi_t(j)$ 表示如果可能最优路径在时刻 t 通过状态 j, 在时刻 $t-1$ 应通过的状态。对于一阶模型, 用 Viterbi 算法求以特定状态 s 开始的最大可能状态序列的步骤。对每一个状态 s(即系统调用) , 进行以下四步操作:

① for $1 \le i \le N$　//初始化

$\delta_2(i) = -\ln(p_{si})$

$\Psi_2(i) = s$

② for $3 \le t \le T$　//递推计算

for $1 \le j \le N$

$\delta_t(j) = \min_{1 \le i \le N} \left[\delta_{t-1}(i) - \ln(p_{ij})\right]$

$\Psi_t(j) = \arg\min_{1 \le i \le N} \left[\delta_{t-1}(i) - \ln(p_{ij})\right]$

③ $P^* = \min_{1 \le i \le N} [\delta_T(i)]$　//终止递推

$S_T^* \arg\min_{1 \le i \le N} [\delta_T(i)]$

④ for $t = T-1, T-2, \ldots, 1$　//回溯, 标定最优状态路径

$s_t^* = \Psi_{t+1}(s_{t+1}^*)$

对于二阶模型, 我们引入符号 $\delta(i, j)$ 表示如果要找的可能最优状态路径在时刻 t 通过状态 j 且在时刻 $t-1$ 通过状态 i, 这条路径在时刻 t 以前(包括时刻 t) 的累计权值; 引入符号 $\Psi_t(i, j)$ 表示如果可能最优路径在时刻 t 通过状态 j 且在 $t-1$ 时刻通过状态 i, 在时刻 $t-2$ 应通过的状态。二阶模型用 Viterbi 算法求以特定状态 s 开始的最大可能状态序列的步骤[6]。对每一个状态 s(即系统调用) , 进行以下四步操作:

① for $1 \le i \le N$　//初始化

$\delta_2(s, i) = -\ln(p_{si})$

$\Psi_2(s, i) = 0$

② for $3 \le t \le T$　//递推计算

for $1 \le j \le N$ $1 \le k \le N$

$\delta_t(j, k) = \min_{1 \le i \le N} \left[\delta_{t-1}(j, k) - \ln(p_{ijk})\right]$

$\Psi_t(j, k) = \arg\min_{1 \le i \le N} \left[\delta_{t-1}(i, j) - \ln(p_{ijk})\right]$

③ $P^* = \min_{1\le i\le N, 1\le j\le N} [\delta_T(i,j)]$ //终止递推

$s^*_{T-1}, s^*_T = \arg \min_{1\le i\le N, 1\le j\le N} [\delta_T(i,j)]$

④ for t = $T-1, T-2, \ldots, 2$ //回溯,标定最优状态路径

$s^*_{t-1} = \Psi_{t+1}(s^*_t, s^*_{t+1})$

最后得到的向量 $O^* = \{s, s^*_2, \ldots, s^*_T\}$ 中记录了我们要求的最大可能系统调用短序列,所有这些最大可能系统调用序列就构成了系统调用正常特征库。易知,对一阶模型本算法的复杂度为 $O(N^2T)$,对二阶模型本算法的复杂度为 $O(N^3T)$。

1.2.3 马氏链测试模块

用上一步得到的正常特征库中短序列的长度 T 为滑动窗口,依次从待测系统调用序列中截取长度为 T 的短序列。然后与正常特征库中具有共同首状态的特征短序列相比较,若两者不相同的状态数大于一定的阈值 V(V 是介于 1 和 T 之间的整数,可称之为短列匹配阈值),则认为此次截取的短序列不正常;若两者相同的状态数小于等于阈值 V,则认为此次截取的短序列正常。我们定义正常的短序列在全体短序列中占的比例为待测系统调用序列的匹配率,若匹配率低于一定的阈值,则认为待测系统调用序列不正常。

2 实验

2.1 实验数据

我们通过实验检验了上述模型的效果,实验数据采用美国新墨西哥州立大学研究"计算机免疫系统"时所用的数据。它们是在计算机运行过程中收集的系统调用序列,包括进程正常运行时产生的正常系统调用序列和有入侵时产生的异常系统调用序列。在我们的实验中只用到了其中关于进程 Sendmail 的数据,在文献[7]上可以找到有关实验数据的详细描述。

2.2 实验结果的表示形式

为求实验结果的稳定性,我们用相对匹配率来表示实验结果。为了说明相对匹配率,首先引入正常匹配率的概念。正常匹配率是指对正常系统调用序列测试得到的匹配率;所谓相对匹配率,是指测试其他系统调用序列得到的匹配率与正常匹配率的比值。显然,正常系统调用序列的相对匹配率一定是 1,相对匹配率越小,说明该系统调用序列偏离正常系统调用序列和程度越大。

2.3 实验结果

对 CERT Sendmail 系统调用序列的一组正常数据和六组异常数据的测试结果如表 1 所示,对 UNM Sendmail 系统调用序列的一组正常数据和八组异常数据的测试结果如表 2 所示。为了进一步对比一阶马氏链模型和二阶马氏链模型的检测效果,图 2 和图 3 用条形图表示了这两组实验的检测结果。

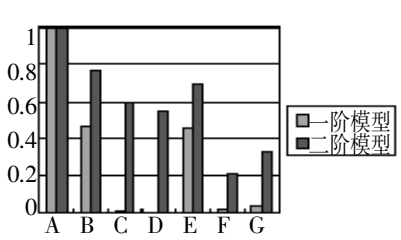

图 2 CERT Sendmail 系统调用序列测试结果

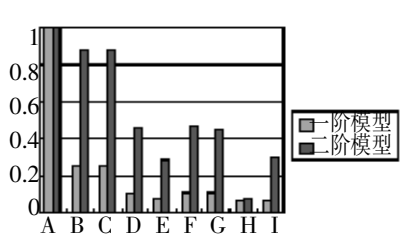

图 3 UNM Sendmail 系统调用序列测试结果

3 结论

实验表明,测试结果的区分度和稳定性受短序列长度 T 和短序列匹配阈值 V 的影响。当 T 取 5, V 取 2 时,测试结果数据的区分度及稳定性最好。表 1 和表 2 列出的就是在这种条件下得到的数据。从表中可以看出,当 T 和 V 取合适的值时,一阶马氏链模型和二阶马氏链模型都能将正常系统调用序列和异常系统调用序列区分开来,其中一阶模型的区分度要比二阶模型的区分度好一些。可见,进程系统调用序列前后之间存在一定的依赖关系。我们将系统调用看作是马氏链模型的状态时,就可以用马氏链模型来提取进程系统调用序列中稳定的短序列,从而刻画进程的特征;而这种前后依赖关系不是太强,用一阶马氏链模型就能很好地刻画,即当前系统调用主要取决于前一个系统调用,而更高阶的依赖特征则不太明显,所以用二阶马氏链模型得到的结果不如一阶马氏模型好。

表 1 CERT Sendmail 系统调用序列测试结果

编号	测试数据(系统调用序列)	一阶模型相对匹配率	二阶模型相对匹配率
A	正常序列	1.000000	1.000000
B	本地 syslog 入侵 1	0.467453	0.765612
C	本地 syslog 入侵 2	0.014068	0.599442
D	远程 syslog 入侵 1	0.005285	0.553906
E	远程 syslog 入侵 2	0.458747	0.693280
F	未成功入侵 1	0.018222	0.212882
G	未成功入侵 2	0.043308	0.332258

表 2 UNM Sendmail 系统调用序列测试结果

编号	测试数据(系统调用序列)	一阶模型相对匹配率	二阶模型相对匹配率
A	正常序列	1.000000	1.000000
B	decode 入侵 1	0.248117	0.881382
C	decode 入侵 2	0.250332	0.877923
D	forwardlooping 入侵 1	0.099879	0.457242
E	forwardlooping 入侵 2	0.081203	0.287942
F	forwardlooping 入侵 3	0.105322	0.459705
G	forwardlooping 入侵 4	0.107199	0.446695
H	forwardlooping 入侵 5	0.069723	0.075068
I	sendmailcp 入侵	0.073119	0.296755

参考文献:

[1] enning D E. An Intrusion-Detection Model [J]. IEEE Transactions on Software and Engineering, 1987, SE-13: 222-232.

[2] Stephanie Forrest, Steven A Hofmeyr. A Sense of Self for UNIX Processes [C]. Los Alamitos, CA: IEEE Computer Society Press, 1996. 120-128.

[3] Nong Ye. Probabilistic Techniques for Intrusion Detection Based on Computer Audit Data [J]. IEEE Transaction on Systems, Man, and Cybernetics, Part A: Systems and Humans, 2001, 31(4): 78-83.

[4] J K Omura. On the Viterbi Decoding Algorithm[J]. IEEE Transactions on Information Theory, 1969, IT-15(2): 177-179.

[5] Rakesh Dugad, U B Desai. A Tutorial on Hidden Markov Models [EB/OL]. http://vision.ai.uiuc.edu/dugad/, 2003-11.

[6] 史笑兴,王太君,何振亚. 二阶隐马尔可夫模型的学习算法及其与一阶隐马尔可夫模型的关系[J]. 应用科学学报, 2001, 19(1): 29-32.

[7] Data Sets & Software[DB/OL]. http://www.cs.unm.edu/~immsec/systemcalls.htm, 2003-11.

作者简介:

钟安鸣(1978-),男,硕士研究生,主要研究方向为信息安全技术、入侵检测系统等;贾春福(1967-),男,教授,主要研究方向为信息安全技术、先进制造系统理论和优化方法及 CIMS。