基于命令紧密度的用户伪装入侵检测方法

王秀利[1],王永吉[2]
(1.中央财经大学信息学院,北京 100081;2.中国科学院软件研究所,北京 100190)

摘 要: 根据Unix系统中用户的历史命令序列,提出一种基于命令紧密度模型的用户伪装入侵检测方法.该方法从命令组合的角度抽取用户的行为模式.用户经常组合使用的命令,表现出关系紧密;不常被一起使用的命令,表现出关系疏远.通过滑动窗口方法从用户的历史命令序列中生成紧密度矩阵.如果待检测的命令块对于该用户来说表现出紧密度过低,则判断为异常.实验表明该方法计算量小,检测效果好,而且具有很高的实时性.

关键词: 异常检测;伪装检测;命令紧密度;shell;主机
中图分类号: TP393.08 **文献标识码:** A **文章编号:** 0372-2112 (2014)06-1225-05
电子学报URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2014.06.029

Masquerader Detection Based on Command Closeness Model

WANG Xiu-li[1], WANG Yong-ji[2]
(1. *School of Information, Central University of Finance and Economics, Beijing* 100081, *China*;
2. *Institute of Software, Chinese Academy of Sciences, Beijing* 100190, *China*)

Abstract: According to the history of command sequence in Unix system, an approach to masquerader detection based on the closeness model of command was proposed. The behavior patterns of user were extracted from the view of command combinations. Those commands combined frequently by users showed close relationship, and other commands exhibited loose relationship. Command closeness matrix was generated by the sliding window from the sequence of commands. If the command block to be detected exhibited a low closeness for the user, it was judged as abnormal. Experimental results show that a simple calculation, an accurate detection, and a high level of real-time can be achieved by using the proposed approach.

Key words: anomaly detection; masquerader detection; command closeness; shell; host

1 引言

伪装入侵检测多采用异常检测技术[1].根据其建模算法不同,可分为基于信息熵[2,3]、文本挖掘[4]、隐马尔可夫模型[5,6]、朴素贝叶斯[7,8]、序列和生物信息[9,10]、支持向量机[11,12]的检测方法等.基于shell命令的入侵检测也得到了较多研究,文献[1]提出基于多重行为模式并行挖掘和多门限联合判决的检测模型.文献[13]利用特殊的多阶齐次Markov链模型对合法用户的正常行为进行建模.文献[14]提出基于共生矩阵的用户行为异常检测方法.

本文提出一种基于命令紧密度模型的伪装入侵检测方法,使用类Unix平台上的shell命令作为原始审计数据,从命令组合的角度抽取用户的行为模式.通过滑动窗口方法从历史命令序列中生成紧密度矩阵代表用户的行为模式,如果待检测的命令块表现出紧密度过低,则判断为异常.

2 命令紧密度

2.1 基本原理

shell命令序列反映了用户操作计算机的行为习惯.比如写c或c++代码的用户会经常使用gcc,make,gdb等命令.经常组合使用的命令,共现次数多,表现出关系紧密;不常被一起使用的命令,共现次数少,表现出关系疏远.假设伪装用户的活动和合法用户不同,它们组合命令的方式也不同.如果一个命令块中命令之间的关系表现疏远,那么,它很可能是伪装用户.本文根据这一特征,设计检测算法.

2.2 命令紧密度模型

两个命令的共现程度称为紧密度.本文提出一种基于滑动窗口的方法度量命令之间的紧密度.

收稿日期:2013-07-15;修回日期:2013-12-09;责任编辑:覃怀银
基金项目:国家自然科学基金(No.61272398);北京市自然科学基金(No.4112053)

用户的历史操作记录是一串 shell 命令组成的序列.使用滑动窗口采集命令序列,经常一起使用的命令会经常出现在同一个窗口内.如果给滑动的窗口编号,并统计每个命令分别出现在哪些窗口中.那么,每个命令将对应一个向量,代表出现过的窗口,向量之间的相似度即代表了两个命令的紧密度.

滑动窗口涉及到两个参数:窗口的大小 s,步长 l.s 代表窗口能够容纳命令的个数,l 表示窗口每次向前滑过的命令个数.为了解决序列被截断的问题,将 l 设置为 s 的一半.如果 $s=4$,则 $l=2$,窗口滑动的工作方式如图 1 所示.

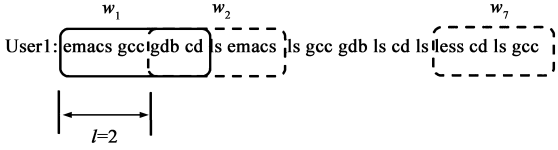


图1 滑动窗口的工作方式

窗口滑动过程中,对窗口内的每个命令,统计该命令出现的次数,生成窗口 × 命令矩阵.假设训练数据的长度是 L,$\sum$表示训练数据中所有不同命令的集合,不同命令个数 $m=|\sum|$,窗口个数 $n=(L\text{-}s)/l+1$.原始数据可以表示为一个 $n\times m$ 的矩阵 $\boldsymbol{V}$,记为

$$\boldsymbol{V}_{n\times m}=\begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1m} \\ v_{21} & v_{22} & \cdots & v_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ v_{n1} & v_{n2} & \cdots & v_{nm} \end{bmatrix}=[\boldsymbol{v}_1 \quad \boldsymbol{v}_2 \quad \cdots \quad \boldsymbol{v}_m]$$

其中,$\boldsymbol{V}$ 中的每个列向量代表与其对应的命令在不同窗口中出现的频率.$\boldsymbol{V}$ 的每个行向量代表其对应的窗口中不同命令出现的频率.假设 $\boldsymbol{V}$ 中第 i 列对应的命令用 $s_i(i=1,2,\cdots,m)$表示,为了度量任意两个命令 s_i,s_j 之间的紧密度.从矩阵 $\boldsymbol{V}$ 生成矩阵 $\boldsymbol{B}_{n\times m}=[b_{ij}]$和 $\boldsymbol{B}'_{n\times m}=[b'_{ij}]$.矩阵 $\boldsymbol{B}$ 和 $\boldsymbol{B}'$ 是二值矩阵,b_{ij}代表命令 s_j 在窗口 w_i 中是否出现.同一个命令在一个窗口中的多次出现看成是该命令与其孪生命令的交替出现,b'_{ij}代表命令 s_j 的孪生命令 s'_j 在窗口 w_i 中是否出现.

两个命令之间紧密度的计算方法如下式.

$$\text{closeness}(s_i,s_j)=\begin{cases}\dfrac{b_i\cdot b'_i}{\|b_i\|\cdot\|b'_i\|}, & \text{if } s_i,s_j\in\sum \text{ 且 } i=j \\ \dfrac{b_i\cdot b_j}{\|b_i\|\cdot\|b_j\|}, & \text{if } s_i,s_j\in\sum \text{ 且 } i\neq j \\ 0, & \text{if } s_i\notin\sum \text{ 或 } s_j\notin\sum\end{cases}$$

窗口内的平均命令紧密度称为窗口聚合度,如果窗口内的命令是合法用户常用的,窗口聚合度的值较高;反之,窗口聚合度的值较低.

3 入侵检测

使用滑动窗口从待检测的序列中采集命令序列,计算每个命令窗口的聚合度,最后生成窗口聚合度的均值,根据预先设定的阈值判断是否是入侵.

3.1 训练过程

(1)从历史命令序列中生成命令-窗口矩阵,如表 1 所示,算法的时间复杂度为 $O(L)$,L 是序列长度.

(2)计算任意两个命令的紧密度,算法如表 2 所示,时间复杂度是 $O(m(m+n))$.

表 1 生成窗口-命令矩阵 $\boldsymbol{V}$ 的算法

```
CreateVMatrix(S)
输入:用户的命令序列 S = s1, s2, s3…sL,滑动窗口的大小 s,步长 l
输出:矩阵 V
m = |∑|      //命令个数
n = (L - s)/l + 1//窗口个数
create matrix V[m][n]
Init all v_ij = 0
for each window i = 1 to n
    //计算第一个属于窗口 w_i 的命令下标
    b = (i - 1) * l
    //计算最后一个属于窗口 w_i 的命令下标
    e = b + s - 1
    for each command j belongs to S[b:e]
      V[s_j][i] + = 1      //s_j 表示命令 j 的索引
    end for
  end for
```

表 2 生成命令紧密度矩阵 $\boldsymbol{C}$ 的算法

```
CreateClosenessMatrix(V)
输入:命令-窗口矩阵 V
输出:命令紧密度矩阵 C
m = |∑|   // 命令个数
n = (L - s)/l + 1//窗口个数
create matrix C[m][n], B[m][n], B'[m][n]
init all c_ij = 0, b_ij = 0, b'_ij = 0
for i = 1 to m
  for j = 1 to n
    B[i][j] = min(V[i][j],1)
    if V[i][j] > 1 then
      B'[i][j] = 1
    end if
  end for
end for
for each command s_i  from ∑
  for each command s_j(j > = i)from ∑
    if j = = i then  //如果命令相同
      C[i][j] = cosine(b[i], b'[i])
    else       //如果命令不同
```

```
        C[i][j] = cosine(b[i], b[j])
    end if
  end for
end for
```

3.2 检测过程

假设待检测命令块为 X,长度为 size(X).检测分为如下 4 个步骤.

(1)用滑动窗口方法扫描命令序列,设置窗口的大小 s 和步长 l 与训练阶段相同.

(2)计算每个窗口的聚合度 closeness(w_i), $i = 1\cdots n$,窗口个数 $n = (\text{size}(X) - s)/l + 1$.

(3)计算所有窗口聚合度的平均值 avg_closeness.

(4)判断 avg_closeness < T? 若成立,判断为伪装;反之,判断为正常.其中 T 是区分正常和异常命令块紧密度的阈值.

4 实验

4.1 数据集

本文采用 SEA 数据集(http://www.schonlau.net).从 70 个用户的 Unix 命令序列中随机选出 50 个作为目标用户,其余 20 个作为入侵者.

该数据集共有 50 个文本文件,每个文件对应一个用户,包含 15000 个命令.前 5000 个命令是由用户本人发出的,是正常数据集,通常用于训练;后 10000 个命令被以一定的概率注入其它 20 个用户的命令,并且以 100 个命令作为一个命令块,每个命令块要么是伪装的,要么是合法的,通常作为测试数据.因此,可以将数据看成由 150 个命令块组成,前 50 个命令块是训练数据,后 100 个是测试数据.此外,文件 masquerade_summary.txt 描述了每个用户的后 100 个命令块中伪装命令块的分布情况.

4.2 实验结果

4.2.1 不同窗口大小的实验结果对比

窗口大小影响方法的性能.窗口过大,使得原本不常在一起使用的命令出现在同一个窗口中,使得命令紧密度度量的准确性降低;窗口过小,使得较多的命令之间紧密度出现 0 的情况,同时,向量长度较大,训练时间相对较长.因此,本文将窗口大小固定在 4~10 之间,图 2 显示窗口大小分别为 4,6,8 时,窗口的大小对实验结果的影响并不明显,即该方法对窗口大小设置不敏感.当 $s = 8$ 时实验效果最好.后续分析选择窗口大小为 8.

4.2.2 用户测试命令块的预测值分布

为了直观看到正常命令块与伪装命令块在预测值上的差异,本节测试了所有 50 个用户的命令序列数据,大部分用户的实验结果都非常好.限于篇幅,仅给出用户 24 的检测结果如图 3 所示.用户的正常命令块与伪装命令块的预测值有明显的区分.

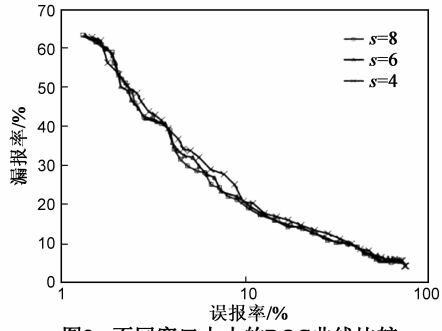


图2 不同窗口大小的ROC曲线比较

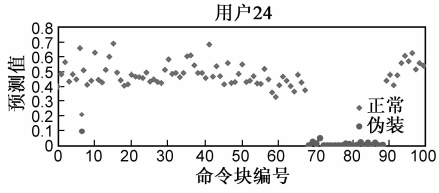


图3 用户24的伪装检测结果.(横坐标表示测试数据中命令序列的块编号,纵坐标表示预测值.圆点代表伪装数据.)

4.2.3 与其他方法的对比

通过选定不同的阈值 T,得到对应各阈值下入侵检测的误报率与漏报率.实验结果如表 3 所示.表中的最后几行是选择不同的阈值 T 所计算得到的实验结果,而前 6 行则是文献[15]中在同一阈值下使用其它 6 种不同方法所得到的实验结果.

表 3 7 种入侵检测方法实验结果的比较

入侵检测方法		误报率(%)	漏报率(%)
Compression		5.0	65.8
Sequence-Match		3.7	63.2
IPAM		2.7	58.9
Hybrid multistep Markov		3.2	50.7
Bayes one-step Markov		6.7	30.7
Uniqueness		1.4	60.6
本文方法	$T = 0.27$	7.0	25.5
	$T = 0.25$	5.7	28.1
	$T = 0.23$	4.6	33.3
	$T = 0.20$	3.7	38.0
	$T = 0.18$	3.1	41.1

另外,本文还与文献[14]进行了对比,文献[14]只采用了 4 个用户的数据,即用户 1,用户 2,用户 3,用户 4.本文采用相同的实验设置,利用用户 3 的 5000 个命

令的前 4000 个作为训练数据用于正常行为建模,后 1000 个作为测试数据用于测试虚警概率;用户 1、用户 2 和用户 4 的 5000 个命令均作为测试数据用于测试检测概率.在 $T=0.23$ 时,本文方法的检测率是 98.6%,误报率为 0.文献[14]的检测率为 100%,误报率为 4%.

本文从检测效果、计算量和建模方法三个方面进行综合比较.

(1)从检测效果上看,其它 6 种方法的检测效果明显不如本文方法.Compression 与 Sequence-Match 这两种方法分别从数据可逆特性与相似匹配度来考虑入侵检测问题,但都没有考虑命令之间的顺序特性,而且,用户行为的建模方法复杂.而本文考虑命令间的组合关系,挖掘命令间的紧密度,因此建模方法更简单,计算量更小.

IPAM 以及 Hybrid multistep Markov 考虑的是用户命令的转移特性,计算量非常大,而且所有的命令都需要逐一考察,建模复杂,除了检测效果没有本文方法好之外,建模方法上也没有优势.

(2)Bayes one-step Markov 的漏报率好,但是误报率差;Uniqueness 的误报率低,但是漏报率高.Bayes one-step Markov 考虑的也是用户命令的一步转移特性,需要较大的计算量.Uniqueness 考虑用户命令的使用频繁度,考察的是很少使用或极少使用的命令.要对每个用户所有的命令进行考察并计算一个统计值,统计值的计算比较复杂[15].另外,如果用户命令的数据量很大,那么建模过程中所需的计算量也会变得非常庞大,给实时入侵检测带来困难.本文方法仅统计用户每个命令间的紧密度,而且在异常检测时也将用户的每个命令组数据作为一组整体考虑,最后用一个简单的数值就能表示出每个命令组是否异常,因此在数据的处理过程中建模方法更为简洁,计算量也更小,更适合实时检测.

(3)文献[14]的检测率很高(100%),但误报率也高.在线实时检测时较高的误报率会影响用户的体验.文献[14]利用部分正则化共生矩阵对用户正常行为建模,综合利用了命令频率特性和共现特性,这一点和本文是一致的.不同的是,文献[14]建模和操作的对象是矩阵,这是以高存储和计算量为代价的.本文则抽象出命令紧密度,易于理解和解释,同时存储和计算量更小.

4.2.4 错误分析

当阈值 $T=0.23$ 时,统计了本文方法在不同用户上的误报与漏报情况,如图 4 所示.

用户 10,13,20,33,36 只占用户总数的 10%,而它们带来的误报却占总误报的约 51%,结合原始数据发现这 5 个用户的测试数据中出现了训练数据中一些不曾出现的命令,导致误报的发生.

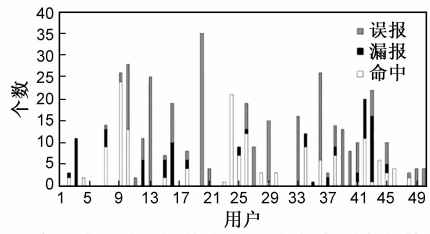


图4 窗口大小为8,阈值为0.23时,各个用户的检测结果

漏报也在一定范围内存在.通过观察原始数据,发现产生漏报的这几组命令序列都曾经在训练数据中出现过.另外,用户 3,16,43 仅占用户总数的 6% 却带来总漏报的约 47%.只有少数用户给入侵检测问题带来较为严重的误报和漏报,说明用户命令序列在一定程度上可以用来对用户行为进行建模,如果能带上命令序列的执行参数并将其作为用户行为的另外一个特征,效果会更好.

5 结论

本文提出了一种基于命令紧密度的用户伪装入侵检测方法.基于用户使用习惯分析,充分考虑了审计数据和用户行为的特点,利用了命令之间的组合特征,创新地提出了新的用户行为模式的表示方法——命令紧密度矩阵.实验证明该方法建模简单,计算量小,检测性能令人满意,且适合实时检测.更重要的是,该方法在训练时不需要伪装训练数据.因为,在现实情况下,伪装攻击数据很难收集,依赖两类训练数据的方法往往很难在实际中使用.

致谢 感谢本文审稿专家提出的宝贵意见.

参考文献

[1] 田新广,段▮毅,程学旗.基于 shell 命令和多重行为模式挖掘的用户伪装攻击检测[J].计算机学报,2010,33(4):697-705.
TIAN Xin-guang, DUAN Mi-yi, et al. Masquerade detection based on shell commands and multiple behavior pattern mining [J]. Chinese Journal of Computers, 2010, 33(4): 697-705. (in Chinese)

[2] BERTACCHINI M, FIERENS P I. Preliminary results on masquerader detection using compression based similarity metrics [J]. Electronic Journal of SADIO, 2007, 7(1): 31-42.

[3] EVANS S, EILAND E, et al. MDLcompress for intrusion detection: Signature inference and masquerade attack[A]. Military Communications Conference[C]. Orlando: IEEE Press, 2007. 1-7.

[4] LATENDRESSE M. Masquerade detection via customized

grammars[J]. Lecture Notes in Computer Science, 2005, 3548:141 – 159.

[5] POSADAS R, MEX-PERERA J C, et al. Hybrid method for detecting masqueraders using session folding and hidden Markov models[J]. Lecture Notes in Computer Science, 2006, 4293: 622 – 631.

[6] OKAMOTO T, ISHIDA Y. Framework of an immunity-based anomaly detection system for user behavior[J]. Lecture Notes in Computer Science, 2007, 4694:821 – 829.

[7] KILLOURHY K S, MAXION R A. Learning from a flaw in a naive-Bayes masquerade detector[A]. NIPS 2007 Workshop on Machine Learning in Adversarial Environments for Computer Security[C]. Whistler: NIPS Press, 2007. 1 – 2.

[8] DASH S K, REDDY K S, et al. Adaptive naive Bayes method for masquerade detection[J]. Security and Communications Networks, 2011, 4(4):410 – 417.

[9] GEBSKI M, WONG R K. Intrusion detection via analysis and modeling of user commands[J]. Lecture Notes in Computer Science, 2005, 3589:388 – 397.

[10] COULL S E, SZYMANSKI B K. Sequence alignment for masquerade detection[J]. Computational Statistics & Data Analysis, 2008, 52(8):4116 – 4131.

[11] KIM H S, CHA S D. Empirical evaluation of SVM-based masquerade detection using Unix commands[J]. Computers and Security, 2005, 24(2):160 – 168.

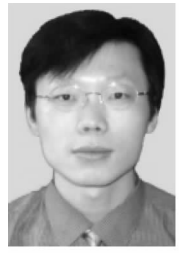
[12] CHEN L, ARITSUGI M. A SVM-based masquerade detection method with online update using co-occurrence matrix[J]. Lecture Notes in Computer Science, 2006, 4064:37 – 53.

[13] 肖喜,翟起滨,田新广,陈小娟,叶润国.基于 shell 命令和多阶 Markov 链模型的用户伪装攻击检测[J].电子学报,2011,38(5):1199 – 1204.
XIAO Xi, ZHAI Qi-bin, et al. Masquerade detection based on shell commands and high-order Markov chain models[J]. Acta Electronica Sinica, 2011, 38(5):1199 – 1204. (in Chinese)

[14] 李超,田新广,肖喜,段▮毅.基于 shell 命令和共生矩阵的用户行为异常检测方法[J].计算机研究与发展,2012,49(9):1982 – 1990.
LI Chao, TIAN Xin-guang, et al. Anomaly detection of user behavior based on shell commands and co-occurrence matrix [J]. Journal of Computer Research and Development, 2012, 49 (9):1982-1990. (in Chinese)

[15] SCHONLAU M, DUMOUCHEL W, et al. Computer intrusion: detecting masquerades[J]. Statistical Science, 2001, 16 (1):58 – 74.

作者简介



王秀利　男,1977 年生,博士,中央财经大学信息学院副教授、硕士生导师.研究方向为信息安全、博弈论.
E-mail: xlwang.cufe@gmail.com

王永吉　男,1962 年生,博士,中国科学院软件研究所研究员、博士生导师.研究方向为隐蔽信道、虚拟化技术.