入侵检测系统中关联规则挖掘技术的研究*)

王 怡 谢俊元
(南京大学计算机科学与技术系 南京 210016)

摘 要 在入侵检测系统中使用关联规则分析,挖掘网络数据中系统特征之间的关联关系,创建正常行为库,找出异常连接。描述了一种改进的 FP_树算法——NFP_树算法,用以进行入侵检测关联规则的挖掘,实验证明,此算法比传统的关联算法在入侵检测中的应用效果更好。

关键词 入侵检测,关联规则,NFP_树

Research on the Technology of Association Rule Mining in Intrusion Detection System

WANG Yi XIE Jun-yuan
(Department of Computer Science and Technology, Nanjing University, Nanjing 210016, China)

Abstract Intusion detection system uses the analysis of association rule to mine the association relationship of network data and system feature, establish the normal behavior stock and find out the abnormal connection. The report discribes a improved FP_tree algorithm——NFP_tree algorithm and uses it to mine asssociation rule in IDS. The experiment shows that the algorithm has better results than traditional association algorithm in IDS.

Keywords Intrusion detection, Association rule, NFP_tree

1 引言

入侵检测系统是一种主动的信息安全保障和防卫系统,它能够弥补防火墙的不足。当前入侵检测系统通常运用模式匹配、统计分析、完整性分析等方法,对网络行为和系统状态进行实时的监控和检测。模式匹配对于检测已知的攻击是非常有效的,它具有准确性高、速度快的特点,但是对于未知的检测则毫无办法。统计方法和完整性分析将用户的正常状态和合法行为建模,然后将用户活动和正常模式比较,判断入侵行为,这些方法虽然可以检测出未知的攻击,但它的错报率和漏报率非常高[1]。数据挖掘是从大量数据中挖掘出未知的信息和知识的过程,而入侵检测系统希望从大量的网络数据中挖掘出入侵行为,因此将数据挖掘技术应用到入侵检测系统中具有一定的必然性。

2 关联规则挖掘概念

关联规则挖掘发现大量数据中项集之间有趣的关联或相关关系。关联规则的获取是通过某个挖掘方法从事务数据库中找出隐含的频繁模式,在入侵检测系统中就是从大量的网络数据中提取频繁模式作为判断网络是否正常的依据,挖掘关联规则的过程其实就是挖掘频繁模式的过程。挖掘频繁模式通常采用两种算法:Apriori 算法和 FP_树算法。本文提出一种改进的 FP_树算法来生成频繁模式。

2.1 定义

本文研究了一种 FP_树算法的优化算法 NFP_树算法,并将它应用到入侵检测系统中。

定义 1(频繁模式树) 按照树的结构对事务数据库中包含的频繁模式进行压缩存储。

1. 根节点:标记为 NULL,一组频繁项目子树为根节点的子节点;

2. 频繁项目子树:频繁项目子树的每个节点由四个域组成,分别为项目名(item_name)、支持度计数(count)和两个节点链,两个节点链分别指向父节点(pre_link)和指向树中与它具有相同项目名的节点(node_link)。

定义 2(频繁项头表) 频繁项头表包含两个域,分别为项目名(item_name)和一个节点链,与传统 FP_树算法不同的是,这个节点链并不是指向频繁模式树中与项头表中具有相同项目名的第一个出现的项节点,而是指向频繁模式树中与项头表中具有相同项目名的最后一个出现的项节点(tail_link)。

定义 3(频繁模式集) 频繁模式集是指 FP_树中叶节点到根节点的组合,设 $l1$ 是任意一条路径,则可表示为($l1$: *count*),其中 *count* 为该路径的支持度计数。频繁模式集是这些路径组成的集合。

3 一种改进的 FP_树算法——NFP 算法

为了方便,我们采用参考文献[2]中的例子进行说明(见表 1),同时设最小支持数为 2。

3.1 构造 NFP_树

本文提出的 NFP_树算法与传统的 FP_树算法在生成树时存在着两点不同:

1. 采用逆向链表,每生成一个节点其链表指向父节点;

2. 项头表中的节点链并不是指向频繁模式树中与项头表中具有相同项目名的第一个出现的项节点,而是指向频繁模

*)本文研究得到 2005 年度国家发改委信息安全重大专项基金项目《网络安全综合防护系统 NSS-2》(发改办高技[2005]1879)和江苏省科技厅网络安全平台技术(高技术研究 BG2005029)的资助。王 怡 研究生,主要研究方向为网络和信息安全;谢俊元 教授,博导,主要研究方向为网络安全和人工智能。

式树中与项头表中具有相同项目名的最后一个出现的项节点。

以表 1 中的事务数据库为例构造 NFP_树，其具体过程如下：

第一步：第一次扫描事务数据库，计算所有项目的支持计数，删去支持计数小于 2 的项，将剩下的项按照支持记数的降序排成一个列表 $L=\langle(I2:7),(I1:6),(I3:6),(I4:2),(I5:2)\rangle$；

第二步：生成树的根结点 T，设置为 NULL；

表 1　事务数据库

TID	项 ID 的列表	排序后的 ID 表
T100	I1,I2,I5	I2,I1,I5
T200	I2,I4	I2,I4
T300	I2,I3	I2,I3
T400	I1,I2,I4	I2,I1,I4
T500	I1,I3	I1,I3
T600	I2,I3	I2,I3
T700	I1,I3	I1,I3
T800	I1,I2,I3,I5	I2,I1,I3,I5
T900	I1,I2,I3	I2,I1,I3

第三步：第二次扫描事务数据库，将每个事务按照前面生成的列表 L 的顺序重新排序，将排序后的项集插入到 NFP_树中，即将每个事务创建为树中的一个分枝。如表中第一个事务扫描生成第一个分枝$\langle(I2:1),(I1:1),(I5:1)\rangle$，在生成树时将这个分枝的所有节点反向链接，即 I5 的 pre_link 指向 I1，I1 的 pre_link 指向 I2，将 I2 的 pre_link 指向根节点 T；而对于第二个事务，由于其排序后包含的项目集$\langle I2,I4\rangle$与已存在的分枝$\langle I2,I1,I5\rangle$共享一段路径$\langle I2\rangle$，于是 I2 的计数加 1，而在 I2 节点下再生成一子树$\langle(I4:1)\rangle$；同理将第三个记录插入。

这样就将事务数据库 D 中的所有频繁项压缩到一棵频繁模式树中，并且保留了原来记录中个项目之间的关联信息。

按上述过程，由表 1 的事务数据库可生成如图 1 所示的 NFP_树。

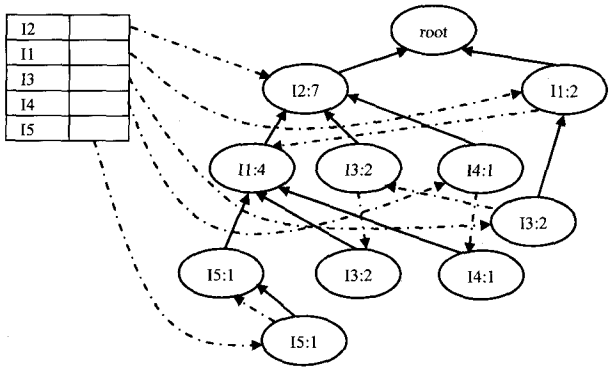

图 1　存放频繁模式的 NFP_树

为了便于对生成的频繁模式树进行遍历，需要创建一个频繁项头表 HTable，每个项通过项头表中的节点链指向它在树中的出现。在传统 FP_树算法中，项头表中的链表指向树中该项的第一次出现，而树中各相同的项通过一个 node_link 相连，但是这里存在一个问题，即当每次在树中插入一个项时，必须从头表到树中一个一个地找出最后一个 node_link 项后，再将新的项加入，这在巨量的网络数据中所耗费的时间是相当大的，因此将项头表中的链表 tail_link 指向该项在树中的最后一次出现，并将树中的该节点链指向它在树中的前一次出现，这样不但解决了上面所讲到的问题，而且仍然保持树中各相同项之间的连接，不会丢失任何的关联信息。

假设新增加一条分路 I6，I3，不考虑生成树的情况，只考虑对于头表的操作，则具体操作为：在树中插入 I6 节点，如图 1 所示，在头表中未找到 I6，则在头表中增加 I6 的项，将其 tail_link 指向新增加的 I6 的位置，并将 I6 节点的 node_link 设为 NULL；插入 I3 节点时，在头表中找到 I3 的 tail_link 域，已经有值存在即头表的 tail_link 域非空，则将新增加的 I3 节点的 node_link 指向头表中的 tail_link 所指的节点，并将 tail_link 域中所指节点的指针指项新增加的 I3 节点所在的位置。

3.2　挖掘 NFP_树

在对 NFP_树进行挖掘时参考了文献[3]中的频繁模式挖掘方法，但有所不同。对频繁项头表 HTable 进行逆向挖掘，首先取出头表中的最后一项，通过 tail_link 所指的节点 t 得到一条从叶节点到根节点的路径。对该路径上的所有节点进行组合(根节点除外)，按定义 3 设置各组合的标识和计数，然后将所有的组合送入候选频繁模式集 SF 中，若 SF 中已经存在相同的组合，则进行合并。合并的具体操作为：组合标识不变，计数为二者之和，同时对 NFP_树进行修改，将树中该路径上的所有节点的 count 值减去叶节点的 count 值。接着找到节点 t 的 node_link 所指向的节点，重复上面的操作，直到某个节点的 node_link 为空。然后再对项头表中下一项进行挖掘，重复上述过程直到扫描完项头表中的所有项。

删去 SF 中计数小于最小支持度计数的组合，这样 SF 中就保留了所有的频繁模式，并根据给出的最小置信度产生所有满足要求的关联规则。

以图 1 所示的 NFP_树为例，其挖掘过程为：对头表进行逆向操作，首先取出 HTable 中的最后一项 I5 的 tail_link 指向的树中节点，从而得到路径(I5,I3,I1,I2:1)，对该路径的四个节点进行任意组合得到(I5,I3:1)，(I5,I1:1)，(I5,I2:1)，(I5,I3,I1:1)，(I5,I3,I2:1)，(I5,I1,I2:1)，(I5,I3,I1,I2:1)，(I3,I1,I2:1)，(I3,I1:1)，(I3,I2:1)，(I1,I2:1)。此时 SF 为空，因此这些组合直接加入到 SF 中，然后对 FP_树进行修改，将该路径上 I5，I3，I1，I2 的 count 值都减去 1。找到该 I5 的 node_link 指向的下一个 I5 节点，得到路径(I5,I1,I2:1)，对该路径的节点进行任意组合得到(I5,I1:1)，(I5,I2:1)，(I5,I1,I2:1)，(I1,I2:1)，将它们加入到 SF 中，对于在 SF 中已有的组合进行合并，其计数为二者计数之和，对于还没有的组合则加入，并对 NFP_树进行修改。再找到该 I5 的 node_link，发现为空，则对项头表中下一项 I4 进行操作，依次类推，直到扫描完项头表中的所有项。最后，从 SF 中删去计数小于最小支持度的组合，本例中设最小支持度为 2，那么对图 1 进行挖掘后生成的频繁模式集 SF 为：{(I5,I2:2)，(I5,I1:2)，(I5,I1,I2:2)，(I4,I2:2)，(I3,I2:4)，(I3,I1:4)，(I3,I1,I2:2)，(I1,I2:4)}。

3.3　算法描述

Step1　构造 NFP_树

```
扫描数据库，产生频繁项头表 HTable；
M=NULL；
将数据库中各事务按头表的顺序排列；
对于每一个事务 T
{ T=[p|P]；
```

(下转第 89 页)

设计了基于 TPM 的安全模型。通过对模型各个组成部分的分析，并结合 Aglet 平台上的安全实验，我们认为本模型可以从硬件体系结构层次上满足移动 Agent 系统的安全需求，并有望从根本上解决恶意主机所带来的安全问题。下一步的工作重点将是进一步完善该模型的原型系统，扩大网络应用环境，在更复杂的网络环境下测试其安全性能。

参 考 文 献

[1] 王红. 移动 Agent 关键技术研究. 学位论文. 中国科学院计算技术研究所，2002

[2] 柳毅. 移动代理技术中若干安全问题的研究. 学位论文. 西安电子科技大学，2005

[3] 王汝传，孙开翠，张登银，等. 基于 JavaCard 的移动代理保护的研究，计算机学报，2004，27(4)：492-499

[4] Varadharajan V. Security enhanced mobile agents// Proc. of 7th ACM Conference on Computer and Communication Security. 2000

[5] Borselius N. Mobile agent security. Electronics & Communication Engineering Journal，October 2002

[6] 沈志东. 可信计算的关键技术与应用研究. 学位论文. 武汉大学，2006

[7] 方艳湘，沈昌祥，黄涛. 分布式系统中计算安全问题的一种解决方案. 计算机工程，2006(9)

[8] Trusted Computing Group. TCG Specification Architecture Overview. Specification Revision 1. 4. http://www. trustedcomputinggroup. org. 2007. 8

[9] Trusted Computing Group. TPM Main Part 2 TPM Structures. specification version 1. 2. http://www. trustedcomputinggroup. org，2007，7

[10] Wu Xiaoping，Shen Zhidong，Zhang Huanguo. The Mobile Agent Security Enhanced by Trusted Computing Technology// The 2nd International Conference on Wireless Communications，Networking and Mobile Computing (WiCOM 2006). Wuhan：IEEE Press，2006

[11] 王新成，袁子建. 可信计算与系统安全芯片设计研究. 国家信息安全测评认证，2005. 5

[12] Lin Ching，Varadharajan V，Wang Yan，et al. Trust Enhanced Security for Mobile Agents// Proceedings of the Seventh IEEE International Conference on E-Commerce Technology (CEC'05). IEEE，2005

[13] The Aglets 2. 0. 2 User's Manual. http://www. trl. ibm. com/aglets/，2004，10

(上接第 82 页)

```
  p 为 T 的第一个元素；
  insert _tree([p|P],M)；}
insert_tree([p|P],M)
{ 设 N 为 T 的子女；
  If N. item_name=p. item_name then
       N. count++；
              else
       { new(N)；N. count=1；
         N. pre_link=T；
         if HTable. tail_link! =NULL then N. node_link=HTable.
         tail_link；
         Htable. tail_link=N；}
  If P 非空 then insert_tree(P,N)；}
```

Step2 挖掘 FP_树

```
按逆序扫描频繁项头表；
t=HTable. tail_link；SF=NULL；
Procedue FP_growth(t,SF)
{repeat
    {得到从 t 到根的节点形成组合 S1，S2，…Sm；
    for (i=1，i<=m；i++)
      {Si. count=t. count；
      If (Si∈SF)then
    SF. Si. count=SF. Si. count+Si. count；
            else Si 加入到 SF 中；
      该路径上其它节点的 count 值减去叶节点的 count 值；}
      t=t. node_link；}
    until t. node_link=NULL；
  取头表中的下一项；
t=HTable. tail_link；}
```

4 实验分析

我们采用美国国防部高级计划研究署 1999 年提供的评估入侵检测系统的测试数据集[4]来测试系统的性能，主要利用 IDS 系统进行异常检测以发现未知的攻击。实验在设定支持度为 50%、置信度为 80%的情况下进行，在 180 个攻击中能检测到 151 个，检测率达到 83. 8%。实验还分别利用 Apriori 算法和 NFP_树算法进行了测试，其结果见图 2 所示。

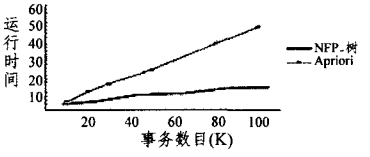

图 2 两种算法运行时间随事务数变化情况

由图 2 可知，当事务数在 20k 以内时，两种算法性能相当，但随着数据的增加，NFP_数算法所用的时间明显比 Apriori 算法的时间少很多，其性能优势有了明显的体现。

结束语 本文描述了一种改进的 FP_树算法-NFP_树算法，并使用它进行入侵检测关联规则的挖掘。实验证明 NFP_树算法比传统关联规则挖掘算法效果更好，但是 NFP_树算法也不能检测出所有的入侵行为，因此如何进一步提高检测的精确度将是我们下一步工作的目标。

参 考 文 献

[1] 胡昌振. 网络入侵检测原理与技术[M]. 北京理工大学出版社，1996

[2] Han Jiawei，Kamber M. Data Mining Concepts and Techniques [M]. 范明，孟小锋，等译. 机械工业出版社，2000

[3] 高俊，施伯乐. 快速关联规则挖掘算法研究[J]. 计算机科学，2005(3)：200-202

[4] Lippmann R，et al. The 1999 DARPA Off-Line Intrusion Detection Evaluation. Computer Networks，2000，34(4)：579-595

[5] Xia Hongxia，Shen Qi，et al. Application of Data Mining Technology to Intrusion Detection System// Proceedings of the 2004 International Symposium on Distruibured Computing and Applications to Business Engineering and Science[R]. Wuhan，china，2004