用户行为异常检测模型①

Users' Abnormal Behavior Detection Model

郑红艳　(中国人民解放军通信指挥学院 六系 湖北 武汉 430010)

吴照林　(中国人民解放军通信指挥学院 战场信息融合实验室 湖北 武汉 430010)

摘　要： 从系统安全的角度出发，将异常检测技术用于用户行为分析领域，建立了一个异常检测原型模型UBAD(User Behavior Abnormality Detection)。利用关联规则算法 FP-growth，对系统中的反映用户行为特征的数据进行深入挖掘,得到每个用户的行为模式，将当前模式与正常历史模式进行对比可以判断用户行为是否异常。

关键词： 用户行为 异常检测 模式挖掘 关联规则

计算机安全是一个日益突出的热门话题，当前很多技术单位采用重重软硬兼施的方法来保障安全，使得工作进程大大折扣，而通过捕捉并分析与用户行为紧密相关的数据变化来判断该用户行为是否异常，这种预见性的决策不会给正常业务造成任何 I/O 负担，并且这种产品的成本比硬机制的安全产品要低得多。通过分析系统和用户的状态、行为与正常行为或期望行为的偏差来检测入侵的技术称为异常检测技术。

1　异常检测技术

异常检测(anomaly detection)是目前入侵检测系统的主要研究方向，其特点是通过对系统异常行为的监测，可以发现未知的攻击模式。异常检测的关键在于建立正常使用模式并利用该模式对当前用户行为进行比较和判断。异常检测的思想最早由 Denning 提出，即通过监视系统审计记录上系统使用的异常情况，可以检测出违反安全的事件。Denning 还建立了一种入侵检测模型，该模型包括主体、客体、审计记录、轮廓、异常记录和活动规则 5 个部分[1]。该模型奠定了异常检测的基础，许多异常检测方法都是以它为基础而发展起来的。

2　UBAD模型结构

目前对用户行为研究得比较多,但大部分都是对网站上用户的兴趣挖掘等，带有一定的商业盈利目的，本文将重点放在系统的数据库安全上，建立了一个用户行为异常检测系统模型 UBAD(如下图所示)，它由五个模块组成：其中数据采集处理模块的主要功能就是得到反映用户行为的干净数据，行为库模块定义了系统用户的行为范围及行为属性解释结果输出的实际含义，模式挖掘模块运用关联规则算法对数据进行挖掘得到用户行为模式[2]，模式对比模块将正常用户行为模式与当前用户行为模式进行比较，输出解释模块主要负责对模式对比结果进行图像化描述和判断处理。各个模块的实现过程见后文所述。

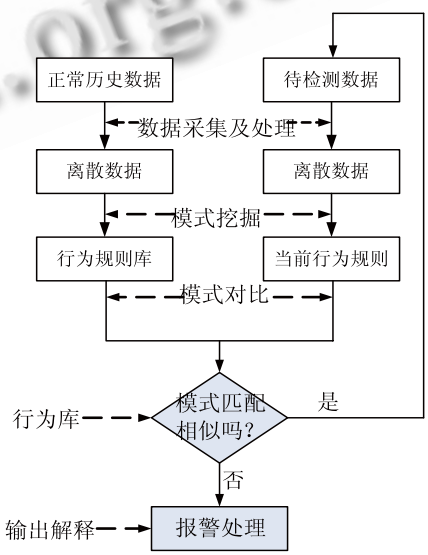

图 1　UBAD 模型结构

① 收稿时间:2008-12-11

2.1 用户行为数据的采集及处理

用户对系统数据的操作在数据库日志中留下了痕迹，日志记录的内容包括操作人员的名称、操作的时间、操作内容等。对这些日志数据进行预处理，可以得到反映用户行为的数据信息， 预处理的算法如下：

输入：审计日志

输出：用户对数据库表的重点字段的操作时序集合。

① AddFile(logs) 加载审计日志。

② Select user by ID from logList 通过用户 ID 识别用户。

③ Create list(userID,time,dml) 建立用户行为列表，其中包括用户 ID，时间，操作的内容。

④ Create table named by UserID ,Insert list into table 新建表并以用户 ID 命名，将行为列表添加到表中。

最后得到的数据库日志中反映用户行为的表字段如下：

表 1 用户行为表内容描述

字段名称	属性	含义
UserID	Varchar(20)	用户的识别码
Time	Datetime	操作时刻
Insert	Varchar(10)	插入语句记录
Delete	Varchar(150)	删除语句记录
Update	Varchar(150)	更新语句记录
Select	Varchar(150)	查询语句记录
Object	Varchar(50)	被操作的对象名

在数据库管理系统中，待分析的数据库审计数据为多值型数据，而产生频繁项集 Apriori 算法和生成强关联规则的 Rule-Generate 算法处理的都是布尔型数据。多值属性可分为单纯数量属性(如读某个表的记录数、更新某个表的记录数等)和复合数量属性(如更新某个表中某一字段的值)及类别属性(如主体、客体等)。对于类别多值属性预处理的方法是：直接把每一类别映射为布尔值[3]，例如将类别多值属性数据库主体转化为 YON-SCOOT、YON—SUMMER、YON—PETER 等布尔型属性。对于数量多值属性预处理的方法是：对属性值进行等距离的划分，每一种操作数量值按 0～99、100～199、200～299、300～399 等划分。

布尔处理结果：

表 2 离散化后的表内容描述

字段名称	属性	含义
SessionID	Varchar(100)	会话序号
YON-USER	NUMBER(1)	主体是否为USER
YON-TIME-PART	NUMBER(1)	时间是否为TIMEPART
YON-SELECT-WHERE	NUMBER(1)	查询语句是否不附加条件
YON-UPDATA-VALUE-PART	NUMBER(1)	更新语句的数据变化是否在DATAPART区间
YON-OBJECT-TABLE-NAME	NUMBER(1)	被操作的表名是否为NAME
YON-OBJECT-COLUMN-NAME	NUMBER(1)	被操作的列名是否为NAME
YON-SELECT-TIMES-PART	NUMBER(1)	进行查询操作的次数是否在PART区间

其中，USER 是所有主体的集合， PART 是所有数值区间的集合，NAME 为所有表和列的名称集合，它们的具体取值依据审计日志而定。(篇幅所限，此处只列出部分字段名)

2.2 模式挖掘的实现原理

模式挖掘是指运用某种算法找出隐含在数据中的规律。本文采用基于 FP-树的关联规则算法 FP-growth，它采用“分而治理之”的策略，将提供频繁项目集的数据库压缩成一颗频繁模式树(FP-树)，然后将这种压缩后的数据库分成一组条件数据库，并分别挖掘每个数据库[4,5]。(版面所限，此处不赘述关联规则相关概念)

2.2.1 算法描述

(1) 构造频繁模式树算法

① 扫描事务数据库 D 一次。收集频繁项的集合(L-项集)以及相应的支持度。按照支持度降序排序，构成频繁项集表 L。

② 创建 FP-树的根节点，以 null 标记。对于 D 中的每个事务 T，进行如下处理：选择 T 中的频繁项目，并按照 L 中的次序排列。设排列之后的频繁项集表为[p|P]，其中 p 是第一个项目，P 是剩余的项目表；如果[p|P]非空，调用 insert_tree([p|P],T)。其执行过程为：如果 T 有子女 N 使得 N.item_name=p.item_name，则 N 的计数加一；否则创建一个新节点 N，

将其计数设置为 1，链接到它的父节点 T，并且通过节点链将其链接到具有相同 item_name 的结点。如果 P 非空，递归地调用 insert_tree(P,T)。

(2) 挖掘频繁项目集算法

FP- 树的频繁项目集挖掘通过调用 FP_growth(FP-tree,null)实现。该实现过程如下：

Procedure FP_growth(Tree,α)

① 如果 Tree 含单个路径 P，则

② 对于路径 P 中的每个组合(记作β)

③ 产生模式β∪α，其支持度 support=β中结点最小支持度

④ 否则对于在 Tree 头部的每个αi

⑤ 产生一个模式β=αi∪α，其支持度 support=αi 的支持度

⑥ 构造β的条件模式基，然后构造β的条件 Treeβ

⑦ 如果 Treeβ非空，则调用 FP_growth(Treeβ，β)

2.2.2 挖掘结果

本文根据给定的最小支持度(min_sup)和最小可信度(min_conf)，使用 FP-growth 算法对我校网管系统中的 Oracle 8i 数据库归档日志进行挖掘，选择其中的 50000 条数据进行预处理挖掘后结果如下(版面所限，本文只列出了 1 条规则)：

表 3　关联规则挖掘结果

规则序号	最小支持度	最小可信度	规则描述
1	29.94%	96.31%	YON-lzp, YON-UPDATA-TIMES-300～399, YON-TIME-08:00～09:00, YON-OBJECT-TABLE-SJSC, YON-OBJECT-COLUMN-RJMC →YON-SELECT-0～100

其中规则 1 表明所有记录中有 29.94%的用户为 lzp，并且他更新的次数在 300～399 之间，时间段在上午 8 点至 9 点，操作了表 SJSC 中的 RJMC 字段，查询次数在 100 以内；用户 lzp 更新的次数在 300～399 之间，时间段在上午 8 点至 9 点，操作了表 SJSC 中的 RJMC 字段有 96.31%查询次数在 100 以内。

2.3 模式对比

将当前审计数据中挖掘出的行为规则与正常历史行为规则库中的所有规则逐一比较，来判断用户行为是否异常，检测算法如下：

① i=0，开始计时，读取规则库中的规则

② i++，while(i< count&&当前规则==规则 i)

{if 两种规则的最小支持度和最小可信度相差 20% go to ④

else　contine}

③ end

④ 发出入侵警告消息，并警示客户端

3　结论

对我校网管系统的 Oracle 日志中 20000 条记录进行了处理，采用关联规则算法 FP-growth 进行了挖掘，实践证明该算法响应速度快，耗用内存小，其中数据记录的条数与所用时间的关系见下图所示：

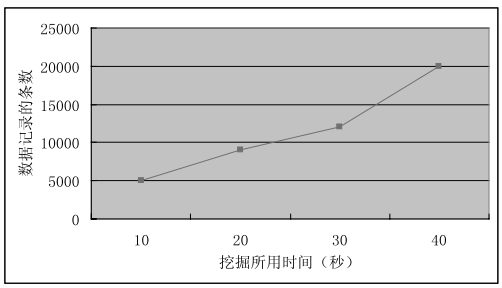

图 2　挖掘时间与数据量的关系

误检率比较低，最多出现 32.1%，虚警率较高，平均达到 46.4%。原因是模式比较时的最小支持度和最小可信度的相差值设置需要动态制定，由于时间关系，本文并没有在这方面进一步深入研究，这也是今后继续努力的一个方面。

参考文献

1 陈文伟.数据仓库与数据挖掘教程.北京:清华大学出版社, 2006.

2 王爱冬,邝祝芳,阳国贵.基于关联规则挖掘的数据库异常检测系统研究.计算机软件与应用,2008,25(5):264－266.

3 邝祝芳,谭骏珊.KMApriori:一种有效的数据库异常检测方法.计算机工程与科学,2008,30(6):18－21.

4 吴玉,李岚.基于数据挖掘的入侵检测行为数据辨析.计算机技术与发展, 2007,17(7):139－141.

5 佟强,周园春,吴开超,阎保平.一种量化关联规则挖掘算法.计算机工程, 2007,33(10):34－35.