基于关联规则的网络异常检测系统设计与实现

刘金龙[1],刘 鹏[1],裴 帅[2],田 冲[2]
(1.海军参谋部,北京 100841;2.信息产业信息安全测评中心,北京 100083)

摘 要: 入侵检测技术是网络安全防御的核心技术之一。由于网络承载的带宽流量日益增多,入侵检测系统需要提供快速的检测能力。Snort 入侵检测系统依靠将抓取的数据与规则匹配来判断是否受到攻击,因此规则的好坏决定了系统性能的高低。结合数据挖掘技术,设计实现一种基于关联规则的关联分析器插件来增强 Snort 对入侵的识别能力。首先利用 Apriori 对 Snort 产生的告警日志进行数据挖掘,搜索隐藏的攻击模式;然后,将关联规则转化为相应的 Snort 规则。最后,利用 SYN Flood 攻击测试规则增强的 Snort 系统的性能,结果表明,改进后的 Snort 能够提高对 SYN Flood 攻击的检测效率。

关键词: 入侵检测;Snort;关联规则;Apriori;SYN Flood
中图分类号: TP393 **文献标识码:** A **DOI:** 10.19358/j.issn.2096-5133.2020.11.003
引用格式: 刘金龙,刘鹏,裴帅,等. 基于关联规则的网络异常检测系统设计与实现[J].信息技术与网络安全,2020,39(11):14-22.

Design and implementation of network anomaly detection system based on association rules

Liu Jinlong[1], Liu Peng[1], Pei Shuai[2], Tian Chong[2]
(1.Naval Staff, Beijing 100841, China;
2.Information Technology & Security Test and Evaluation Center, Beijing 100083, China)

Abstract: Intrusion detection technology is one of the core technologies of network security defense. Due to the increasing network bandwidth traffic, intrusion detection systems need to provide rapid detection capabilities. The Snort intrusion detection system relies on matching the captured data with rules to determine whether the system is under attack, so the quality of the rules determines the performance of the system. This paper combines data mining technology to design and implement an association analyzer plug-in unit based on association rules to enhance Snort to identify intrusions. At first, Apriori is used to mine the alarm logs generated by Snort and search the hidden attack patterns; Furthermore, the association rules are converted into corresponding Snort rules. Finally, the performance of the Snort system is enhanced by using SYN Flood attack test rules. The results show that the improved Snort can improve the detection efficiency of SYN Flood attacks.

Key words: intrusion detection; Snort; association rules; Apriori; SYN Flood

0 引言

入侵检测作为一种重要的网络安全防护技术,由 ANDERSON J P[1]在 1980 年首次提出,经过几十年的发展,在入侵检测系统模型构建[2]、检测数据集获取[3]、检测方法创新[4-6]等方面取得了丰硕的成果,已广泛应用于物联网[7]和智慧城市[8]等多种应用场景。然而随着网络承载带宽流量日益增多,人工分析海量告警日志信息已难以满足日常需求,开发基于数据挖掘的入侵检测系统逐渐成为主流[9]。入侵检测系统的基本原理就是将获取的数据经过处理后,与之前设好的规则进行匹配,从而判断是否为攻击或入侵[10-11]。根据入侵检测的原理,系统需要获取足够多的数据,才能更准确地判断是否为攻击或入侵。

为了能够更有效处理网络中大规模的安全数据,学者们开始研究数据挖掘技术,王洋等[12]利用

贝叶斯攻击图模型从大规模流量中识别异常告警，通过告警关联识别攻击者的意图。李祉岐等[13]对现有告警融合和关联分析方法进行了综合分析，提出了基于告警关联的入侵检测体系架构以及应用准则。胡浩等[14]利用吸收 Markov 链模拟攻击者的入侵行为，解决了用攻击图对攻击路径进行仿真时存在的状态爆炸问题，有效提升入侵路径识别的精度。

Snort 是美国 Sourcefire 公司发布的开源入侵检测软件，提供规范化的接口便于用户对 Snort 进行扩充与改进，因此研究人员选择在 Snort 基础上进行研发或对其进行进一步的功能扩充，以实现从大量日志信息中，快速、有效找到网络流规律及数据信息之间的联系，发现异常的网络数据流的特征信息，提升漏告警和误告警场景中的检测完备性。告警关联规则挖掘是入侵检测的重点环节之一，HU H [15]等认为同一攻击过程中的各个攻击步骤以较高的概率在一个时间窗口内发生，因而同一攻击过程产生的告警在统计上具有相似性，因此提出了基于统计时序的告警关联方法，通过计算告警序列之间的因果关联指数来判断告警是否具有关联关系。上述方法不依赖领域知识，但存在计算量大、参数配置复杂等不足。

针对上述问题，本文以 Snort 为基础，设计实现了能够从大量日志信息中发现网络中攻击与入侵数据流间隐藏关系的入侵检测系统。本文提出的方法能有效融合告警信息，识别入侵过程，帮助管理人员掌握网络安全状况，辅助指导风险评估和入侵响应等后续过程。

1 相关知识

1.1 入侵检测系统

入侵检测系统根据设定的规则进行告警匹配来判断是否存在入侵行为，通常入侵检测系统由以下四部分构成，如图 1 所示。

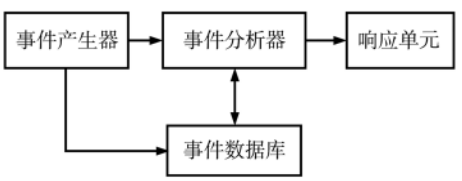


图 1 入侵检测系统组成

(1)事件产生器：负责收集网络或主机上的安全数据然后将其转换为安全事件。

(2)事件分析器：根据提前设定好的判定规则对事件进行分析和判断，判断安全事件是否为网络入侵，如果判定为入侵还需要给出告警信息。

(3)事件数据库：用于存储安全事件数据。

(4)响应单元：根据事件分析器的结果实施入侵响应，比如关闭连接、切断网络。

依据检测原理的不同，入侵检测系统主要包括两类：

(1)异常检测系统

异常检测系统通过对用户正常行为进行分析，提取特征信息，并将提取到的特征信息储存到数据库中用以建立正常行为模式。

(2)误用检测系统

误用检测系统是针对各类攻击、入侵活动进行分析，发掘攻击和入侵特征，根据这些特征建立入侵特征模式库进行判断。

1.2 Snort 原理

Snort 系统是一种轻量级误用检测系统，其最大优点是具有可扩展性，通过外部插件可以很方便地对系统功能进行扩展。Snort 的架构具有模块化的特点，其工作过程如下：首先由嗅探器取得原始元数据，然后经协议处理器和预处理器，使用检测引擎进行分析处理，如果与规则库中的规则匹配成功则识别相应攻击。Snort 架构如图 2 所示。

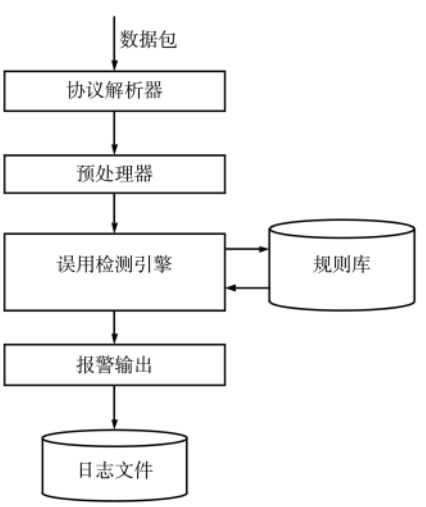


图 2 Snort 体系结构

(1)协议解析器

首先从获取的数据包中解析出协议部分的信息，然后将信息写入定义好的数据结构中。

(2)预处理器

预处理器是实现 Snort 插件机制的主要部分，

通过预处理器可以将外部插件加载到 Snort 中，按照插件的工作方法对数据进行处理。

(3)检测引擎

将数据流信息与规则库逐一匹配，若匹配成功则触发告警信息。

(4)告警输出

输出模块可以根据用户需求，将检测的结果按一定格式进行输出，输出方式主要有：

①输出到文件：Snort 自定义的格式、CSV 格式。

②输出到数据库：MySQL。

Snort 规则结构如图 3 所示。

(1)规则头部

动作：指 Snort 发现匹配规则的数据包后触发的动作。

协议：指规则对应的数据包采用的协议。

方向操作符："->"表示数据传递方向；"<>"表示双向操作符，即规则头部中的两对地址\端口号可以作为源或目标。

(2)规则选项

Snort 给出了 42 个关键字供用户根据实际情况编写规则，具有很强的扩展性。

规则头部							规则选项
动作	协议	源地址	源端口	方向	目的地址	目的端口	

图 3 Snort 规则结构

1.3 数据挖掘技术

数据挖掘是对大量数据经过特殊处理后，利用特定算法挖掘数据之间隐藏的有价值的关系。数据挖掘过程如图 4 所示，步骤如下：

(1)数据清洗：将原始数据中与挖掘主题无关的数据剔除出去，减小待处理数据的规模；

(2)数据集成：将经过清洗后的数据按照相应的规则集成到一起，存储到仓库中；

(3)数据转换：将数据转换成适合挖掘算法进行处理的数据格式，形成格式化数据；

(4)数据挖掘：采用选择的挖掘算法对格式化数据进行处理，提取高价值信息；

(5)模式评估：对步骤(4)得出的信息进行评估，分析挖掘数据的有效性；

(6)知识表示：将评估有效的数据可视化展示给用户。

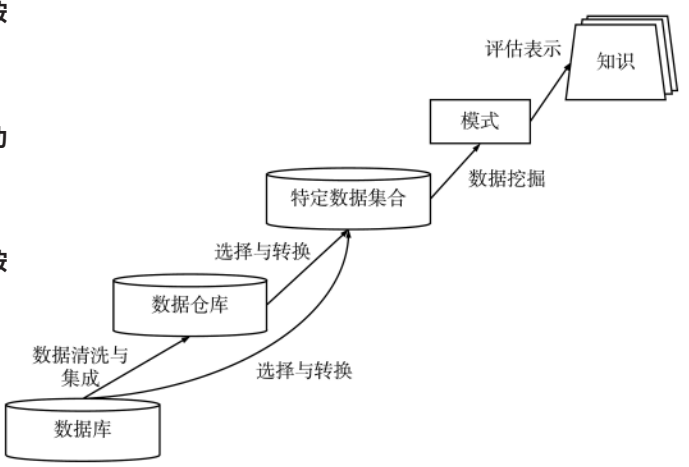


图 4 数据挖掘过程

1.4 关联规则挖掘

关联规则的符号解释如表 1 所示，对于数据项集 A 和 B，关联规则"A=>B"表示如果 A 出现，那么可知 B 也出现，用于刻画不同数据项间的隐藏关联。规则评估包括支持度 s 和置信度 c 两个重要指标，s 表示 A 和 B 同时出现的概率，s 越小则表示 A 和 B 的关联性越小；c 揭示了 A 出现时 B 会同时出现的概率。

表 1 关联规则符号解释

符号	描述
I	所有项目的集合
T	一个事务(每个事务是一些项目的集合)
D	所有事务 T 的集合
X、Y	某些项目的集合
=>	代表"关联"操作

2 基于关联规则的入侵检测系统设计

本节首先对现有入侵检测系统存在的不足进行总结并提出解决方案，然后设计 Apriori 算法的主要函数实现方法，接着给出系统总体设计，最后对本系统的核心关联分析器内部组件进行详细设计。

2.1 问题分析与解决思路

Snort 的优点是具有很强的可扩展性，但存在两方面问题：一是受其检测原理的限制只能检测已知的攻击和入侵；二是随着网络数据量的增加，Snort 入侵检测效率急剧下降。为解决上述问题，本文解决思路如下：设计一个关联分析器并作为插件连接到 Snort 系统中。关联分析器可以对 Snort 中的海量日志数据进行关联分析，从中挖掘出有价值的信息，最后将这些信息转换成 Snort 的规则并补充进规则

库，使 Snort 可以发现新的攻击行为。

2.2 基于 Apriori 算法的关联规则挖掘

本文以 Snort 入侵检测系统为基础，对其海量日志进行数据挖掘找出关联规则，用以补充 Snort 的规则库，而关联规则挖掘的关键是找出最大频繁项目集。Apriori 算法是一种重要的用于挖掘单维、单层、布尔关联规则频繁项集挖掘算法，其对数据集规模依赖性低。Apriori 算法主要分为两步，第一步是找出一维最大频繁项目集，通过扫描数据库统计得出；第二步执行循环部分，主要由两个函数实现：AprioriGen、$InitL_K$，如表 2 所示，循环结束的标志是不再有新的最大频繁项目集生成。

表 2 Apriori 主要函数

	AprioriGen	$InitL_K$
函数功能	结合 $k-1$ 步生成的最大频繁项集 L_{k-1} 产生候选项集 C_k	分析候选项集 C_k 中项目的支持度，剔除低于最小支持度的项目后输出 L_k
输入参数	$k-1$ 维最大频繁项目集 L_{k-1}	所有事务集合 D 候选项目集 C_k 最小支持度 s
输出结果	k 维候选项目集 C_k	最大频繁项目集 L_k
执行过程	函数的执行过程主要分为 2 步：第 1 步是连接步，基于 L_{k-1} 自连接生成 C_k'；第 2 步是剪枝步，对 C_k' 中所有 k 维项集实施筛选，若某个 k 维项集的所有 $(k-1)$ 维子集中有不属于 L_{k-1} 的情况，则剔除此 k 维项目集	函数执行过程为：统计候选项目集 C_k 中每个 k 维项集在全部事务集合 D 中的支持度，删除支持度小于最小支持度 s 的 k 维项集后输出 L_k

在剪枝步中用到了 Apriori 算法的一个性质，即：最大频繁项目集的子集也一定是最大频繁项目集，反之如果存在不符合条件的子集，则需要将此项剔除，这样可以有效提升关联告警数据挖掘的精度。Apriori 挖掘最大频繁项目集的过程如图 5 所示，其中设置支持度为 2，TID 为事务标志符，items 为事务包含的具体项目。

2.3 系统总体设计

由于 Snort 源码开放，因此开发者可以灵活根据监测需求进行代码的编写，并且以插件的形式插入到 Snort 中。本系统在原有的 Snort 基础上加入关联分析器插件从而可以找出系统日志的数据项间隐藏的关系，利用这些安全数据项之间的联系发现新的入侵行为，最后将关联规则转换为 Snort 规定的规则格式添加至规则库中，改进后的 Snort 不仅可以有效利用海量数据，还可以发现新的入侵行为。总体设计如图 6 所示。

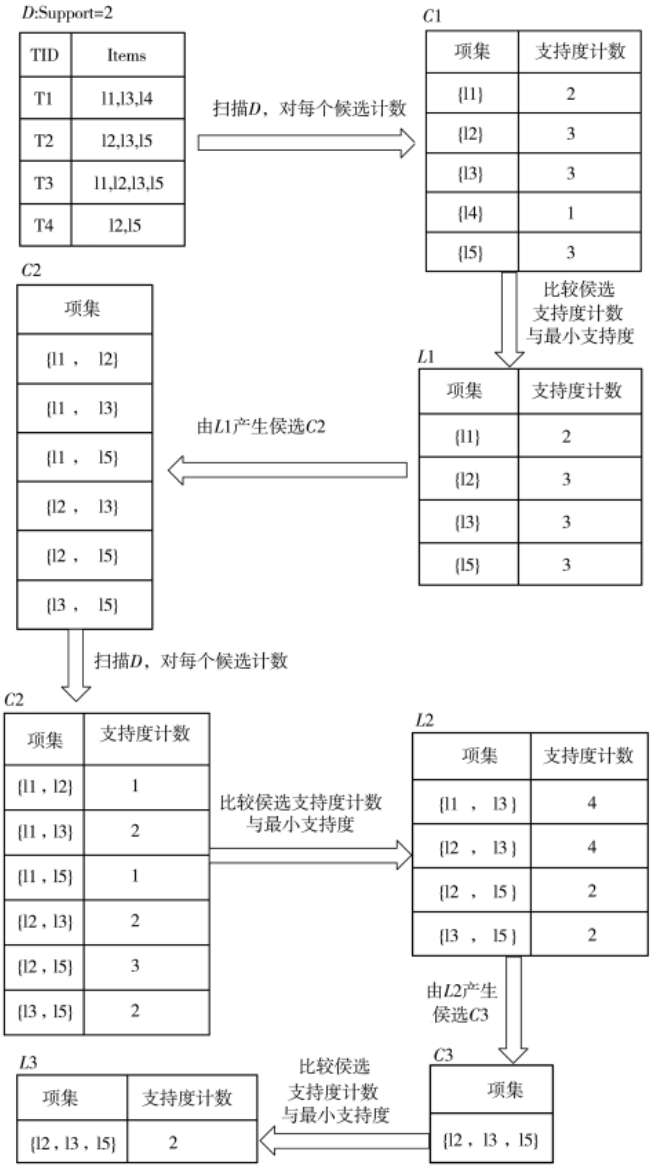


图 5 Apriori 算法流程图

2.4 关联分析器设计

关联分析器的作用是找出 Snort 日志中攻击数据项间尚未被发现的联系，然后将这些关系转换为 Snort 要求的规则格式并补充至规则库，增强系统的防御效能。关联分析器基本结构如图 7 所示，工作流程如下：

(1)数据预处理：Snort 系统日志中包含众多的字

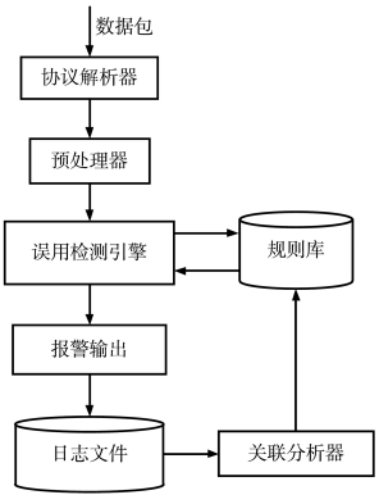


图 6 基于关联规则挖掘的 Snort 入侵检测系统设计

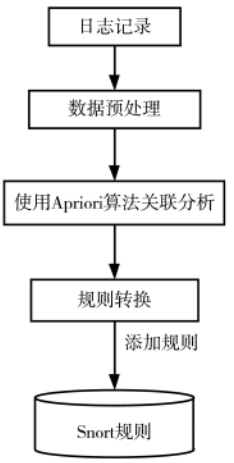


图 7 关联分析器

段，从中筛选出有效字段，剔除与关联分析无关的字段；

(2)使用 Apriori 算法关联分析：用户依据实际需求设置最小支持度和最低置信度；

(3)规则转换：对于步骤(2)输出的关联规则的规则头部放入 Snort 规则头，对应规则选项部分放入 Snort 规则选项，完成格式合并后存入 Snort 规则库。

3 基于关联规则的入侵检测系统实现

3.1 系统运行环境

本系统运行的环境如下：

(1)硬件环境

4 GB 内存、500 GB 外存、CORE i7 处理器。

(2)软件环境

①操作系统：Windows 7；

②数据库：SQL Server 2008；

③Snort 版本：Snort 2.9.5.5；

④抓捕工具：WinPcap 4.1.3；

⑤规则库：Snortrules-snapshot-2920.tar.gz。

3.2 系统实现流程

系统实现流程包含：Snort 安装与配置、数据库安装与配置、关联分析器的实现，其中关联分析器的实现是核心，实现流程如图 8 所示。

3.3 Snort 安装与配置

Snort 安装与配置流程如下：

(1)工具准备：Snort 安装包、Snort 规则库、抓捕工具 WinPcap。

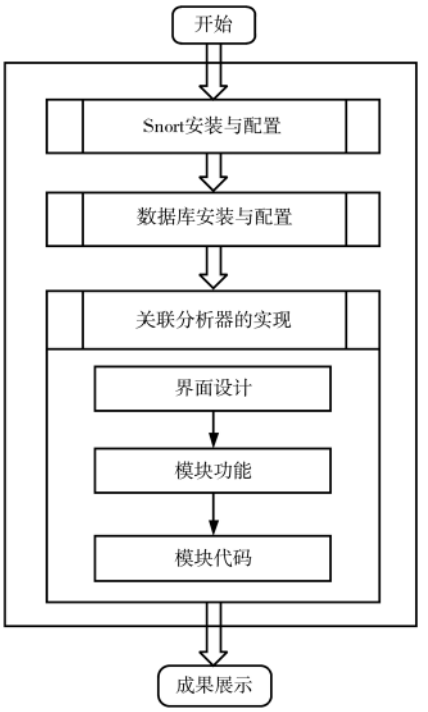


图 8 系统实现流程图

(2)安装软件 Snort 组件。

(3)安装抓捕工具 WinPcap，WinPcap 负责抓取数据包以获取原始数据。

(4)检测 Snort 是否安装成功，通过 cmd 命令行找到 Snort 启动程序的地址。

(5)安装 Snort 规则库，Snort 规则库的压缩包下载完成后解压到 Snort 的安装文件夹内，替换原有文件。

(6)修改配置文件，打开 d:\Snort\etc\Snort.conf 文件，找到以下三个变量：var RULE_PATH，dynamicpreprocessor，dynamicengine，分别在后面添加路径，如图9 所示。

```
# such as:  c:\snort\rules
var RULE_PATH d:\snort\rules
var SO_RULE_PATH d:\snort\so_rules
var PREPROC_RULE_PATH d:\snort\preproc_rules

# path to dynamic preprocessor libraries
dynamicpreprocessor directory d:\snort\lib\snort_dynamicpreprocessor

# path to base preprocessor engine
dynamicengine d:\snort\lib\snort_dynamicengine\sf_engine.dll
```

图 9 Snort 配置

(7)修改 Snort 输出方式，过程如下：打开 Snort.conf 文件添加按钮，将告警日志输出到 CSV 文件的代码：output alert_CSV：filename，timestamp，msg，proto，

src，srcport，dst，dstport，ethsrc，ethdst，ethlen，tcpflags，tcpseq，tcpack，tcplen，tcpwindow，ttl，tos，id，dgmlen，iplen，icmptype，icmpcode，icmoid，icmpseq。Snort 将告警日志以.CSV 格式输出到 d:\Snort\log 文件夹内，如图 10 所示。

3.4 数据库的安装与配置

下载并安装数据库 SQL Server 2008，将 Snort 的.CSV 日志文件导入数据库，在数据库中查看导入的 Snort 告警日志数据，如图 11 所示。

3.5 关联分析器的实现

3.5.1 模块功能实现

(1)数据库操作模块

本模块主要有如下两个作用：

①与数据库建立连接

关键代码如下，通过与数据库建立连接，为下一步传输数据做好准备。

```
string connectionString =@"Data Source =localhost;
Initial Catalog=“数据库名”;Integrated Security=True";
SqlConnection sqlCon=new SqlConnection(connectionString);
sqlCon.Open();
```

②数据预处理

这部分主要包括两项操作：

剔除无关的数据项。数据库中包含众多 Snort 告警日志中的告警项目，由于每一项都包含大量无关数据，为提升系统的效率，在进行关联分析前，剔除无关的数据项，保留可能存在隐含关联的数据项。

根据 Apriori 算法导出数据。根据 Apriori 的输入设置，按照需求取出两部分数据：一项候选集和项目集，并以 ArrayList 格式存储，可以灵活设置数组的大小，动态增加或减少元素。

一项候选集示例：("I1""I2""I3""I4""I5")

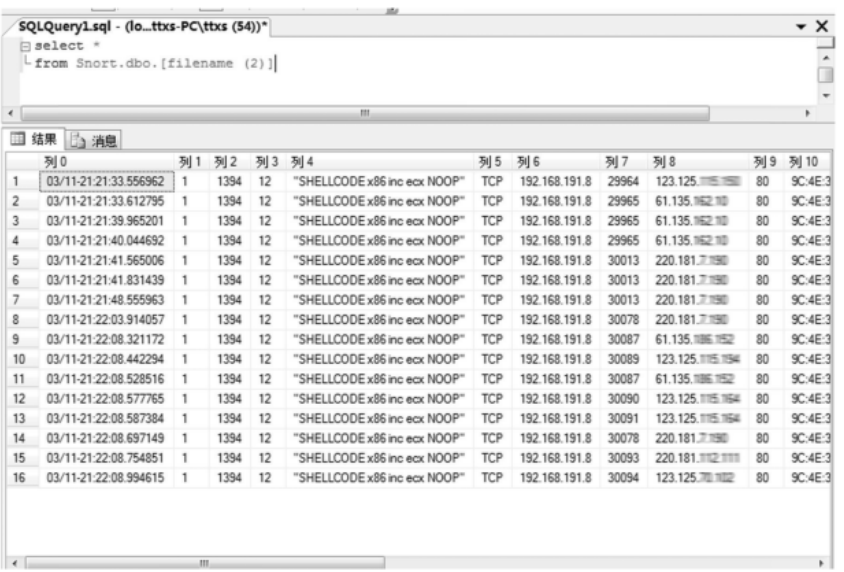
项目集示例：("I1，I2，I5""I2，I4""I2，I3""I1，I2，I4""I1，I3""I1，I2，I3，I5""I1，I2，I3")

(2)生成频繁项集模块

生成频繁项目集语句为：L=Apriori(D，I，s)，其

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	03/11-21:21:33.556	1	1394	12	SHELLCODE	TCP	192.168.191.8	29964	123.125.	80	9C:4E:36:64:27:37:0x36E			***AP***	0x76704A	0xC30B7D26	
2	03/11-21:21:33.612	1	1394	12	SHELLCODE	TCP	192.168.191.8	29965	61.135.	80	9C:4E:36:64:27:37:0x3C3			***AP***	0x3A1623	0x8CFABF77	
3	03/11-21:21:39.965	1	1394	12	SHELLCODE	TCP	192.168.191.8	29965	61.135.	80	9C:4E:36:64:27:37:0x413			***AP***	0x3A1627	0x8CFAC073	
4	03/11-21:21:40.044	1	1394	12	SHELLCODE	TCP	192.168.191.8	29965	61.135.	80	9C:4E:36:64:27:37:0x454			***AP***	0x3A162A	0x8CFAC516	
5	03/11-21:21:41.565	1	1394	12	SHELLCODE	TCP	192.168.191.8	30013	220.181.	80	9C:4E:36:64:27:37:0x3A0			***AP***	0xAF2252	0x5FFB20C9	
6	03/11-21:21:41.831	1	1394	12	SHELLCODE	TCP	192.168.191.8	30013	220.181.	80	9C:4E:36:64:27:37:0x553			***AP***	0xAF2256	0x5FFB216E	
7	03/11-21:21:48.555	1	1394	12	SHELLCODE	TCP	192.168.191.8	30013	220.181.	80	9C:4E:36:64:27:37:0x5D6			***A****	0xAF225B	0x5FFB226E	
8	03/11-21:22:03.914	1	1394	12	SHELLCODE	TCP	192.168.191.8	30078	220.181.	80	9C:4E:36:64:27:37:0x56C			***AP***	0x6614C7	0x866566DC	
9	03/11-21:22:08.321	1	1394	12	SHELLCODE	TCP	192.168.191.8	30087	61.135.	80	9C:4E:36:64:27:37:0x4D7			***AP***	0x7F0096	0x35DAC92E	
10	03/11-21:22:08.442	1	1394	12	SHELLCODE	TCP	192.168.191.8	30089	123.125.	80	9C:4E:36:64:27:37:0x5D6			***A****	0xA3034E	0xF9AE2F74	
11	03/11-21:22:08.528	1	1394	12	SHELLCODE	TCP	192.168.191.8	30087	61.135.	80	9C:4E:36:64:27:37:0x532			***AP***	0x7F00E0	0x35DACA4F	
12	03/11-21:22:08.577	1	1394	12	SHELLCODE	TCP	192.168.191.8	30090	123.125.	80	9C:4E:36:64:27:37:0x426			***AP***	0xACDB8D	0x25FD19C1	
13	03/11-21:22:08.587	1	1394	12	SHELLCODE	TCP	192.168.191.8	30091	123.125.	80	9C:4E:36:64:27:37:0x437			***AP***	0x3C0289	0x23482E97	
14	03/11-21:22:08.697	1	1394	12	SHELLCODE	TCP	192.168.191.8	30078	220.181.	80	9C:4E:36:64:27:37:0x5A7			***AP***	0x6614CC	0x866567DC	
15	03/11-21:22:08.754	1	1394	12	SHELLCODE	TCP	192.168.191.8	30093	220.181.	80	9C:4E:36:64:27:37:0x473			***AP***	0x49C3E5	0xA87EE60A	
16	03/11-21:22:08.994	1	1394	12	SHELLCODE	TCP	192.168.191.8	30094	123.125.	80	9C:4E:36:64:27:37:0x5D6			***A****	0x9DE436	0xCED9AA7D	

图 10 Snort 告警日志



	列 0	列 1	列 2	列 3	列 4	列 5	列 6	列 7	列 8	列 9	列 10
1	03/11-21:21:33.556962	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	29964	123.125.	80	9C:4E:3
2	03/11-21:21:33.612795	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	29965	61.135.	80	9C:4E:3
3	03/11-21:21:39.965201	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	29965	61.135.	80	9C:4E:3
4	03/11-21:21:40.044692	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	29965	61.135.	80	9C:4E:3
5	03/11-21:21:41.565006	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	30013	220.181.	80	9C:4E:3
6	03/11-21:21:41.831439	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	30013	220.181.	80	9C:4E:3
7	03/11-21:21:48.555963	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	30013	220.181.	80	9C:4E:3
8	03/11-21:22:03.914057	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	30078	220.181.	80	9C:4E:3
9	03/11-21:22:08.321172	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	30087	61.135.	80	9C:4E:3
10	03/11-21:22:08.442294	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	30089	123.125.	80	9C:4E:3
11	03/11-21:22:08.528516	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	30087	61.135.	80	9C:4E:3
12	03/11-21:22:08.577765	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	30090	123.125.	80	9C:4E:3
13	03/11-21:22:08.587384	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	30091	123.125.	80	9C:4E:3
14	03/11-21:22:08.697149	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	30078	220.181.	80	9C:4E:3
15	03/11-21:22:08.754851	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	30093	220.181.	80	9C:4E:3
16	03/11-21:22:08.994615	1	1394	12	"SHELLCODE x86 inc ecx NOOP"	TCP	192.168.191.8	30094	123.125.	80	9C:4E:3

图 11 数据库中的告警日志

中输入D为项目集，格式为ArrayList，I为一项候选集，s为用户设置的支持度，格式为double；输出L为频繁项目集，格式为List<ItemSet>，包含了项集的元素和支持度，通过以下语句将L的数据加载到插件中。

this.dataGridView1.Rows[i].Cells[0].Value=L[i].Items;

this.dataGridView1.Rows[i].Cells[1].Value=L[i].Sup;

(3)生成关联规则模块

生成关联规则的语句为：R=AssociationRules(L,c)，其中输入L为生成频繁项集语句的输出项，c为提前设定好的最低置信度，格式为double；输出结果R为得到的关联规则，以List<AssoRule>格式存储；函数AssociationRules用于搜索每个最大项目集A的所有非空子集。

3.5.2 模块代码实现

系统的代码主要分为数据库操作、算法、功能控件。数据库操作使用LINQ to SQL语句进行实现；算法代码主要依据经典的Apriori算法采用C#语言进行编写；同时利用DataGridView、NumericUpDown等功能控件完成响应功能。

4 实验与分析

为验证系统的性能，搭建如图12所示的办公局域网环境，实验过程分为三个部分，第一部分是数据采集，编写SYN Flood脚本对目标主机进行攻击，时间持续1 h，在攻击前运行本系统进行监测和日志记录；第二部分是数据挖掘，利用关联分析器对模拟攻击阶段获取的Snort告警日志进行数据挖掘，产生新的Snort规则，并补充进Snort规则库；第三部分是入侵检测阶段，操作与数据采集阶段相同，之后查看产生的告警日志，验证本文系统是否发现SYN Flood攻击，最后是实验结果分析。

4.1 实验环境

本实验的目的是测试关联分析器能否找出日志数据项间隐藏的关系，以及改进后的Snort是否具有未知攻击识别能力。实验网络部署如图12所示。

(1)网络攻击工具：端口扫描工具ScanPort、地址扫描工具Advanced IP Scanner、漏洞扫描工具Nessus、SYN Flood攻击脚本。

(2)入侵检测系统：入侵检测系统采用Snort 2.8，加载全部检测规则，用于检测攻击行为，并产生原始的报警信息。由于Snort的包嗅探模式和入侵检测模式需要捕获网络数据包，因此安装网络数据包截取驱动程序WinPcap。

(3)报警存储及分析工具：采用MySQL数据库提供报警数据的存储服务；同时安装PHP5、jpgrapg、ACID进行初步的数据分析，其中PHP5为网页程序开发语言，jpgrapg为图形库，ACID为图形接口，用于将报警数据图形化。

(4)方案实现工具：使用VC 6.0开发平台编程实现报警挖掘与关联。

4.2 实验过程

4.2.1 数据采集

(1)利用C语言编写SYN Flood攻击程序。

(2)运行Snort，输入口令：Snort -d -h 10.104.178.231/24-l\Snort\log-c Snort.conf，启动Snort，如图13所示。

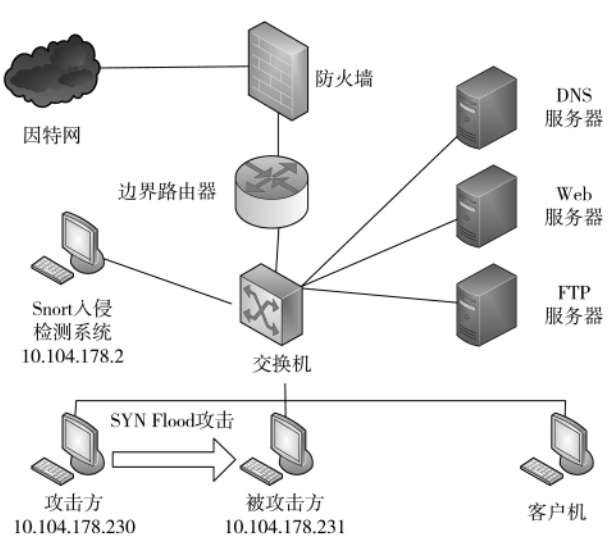


图12 实验网络拓扑

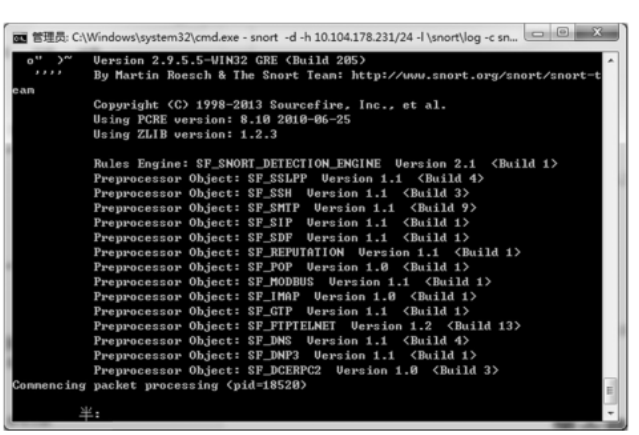


图13 Snort监测

(3)设置被攻击的 IP:#define SYN_DEST_IP "10.104.178.231";设置伪装的 IP 起始值:#define FAKE_IP "10.104.178.3",运行 SYN Flood 攻击程序进行 DoS 攻击,如图 14 所示(一个点代表进行了一次连接请求)。

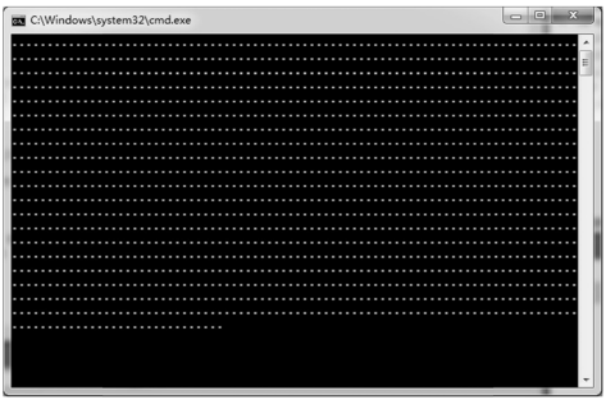


图 14 SYN Flood 攻击

4.2.2 数据挖掘

(1)将数据采集阶段的 Snort 的告警日志导入 SQL Server 数据库,如图 15 所示。

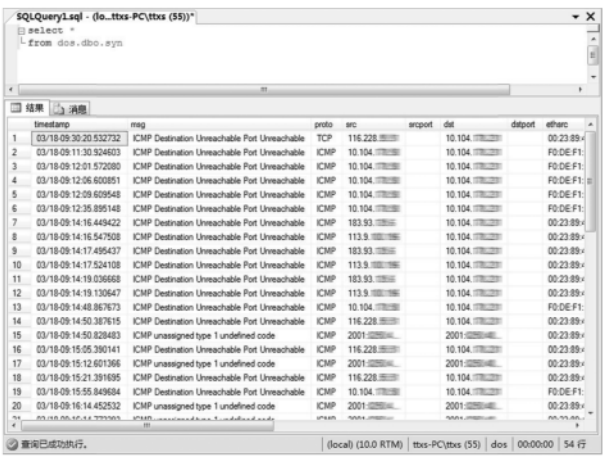


图 15 告警日志导入数据库

(2)运行程序,这里选择将支持度设为 0.3,最小置信度阈值设为 0.7,点击按钮进行计算。

(3)将关联规则转换为 Snort 规则:alert icmp $EXTERNAL_NET any -> 10.104.178.31 any(msg:"SYN Flood!!! ";ttl:54;),并将此规则补充进 Snort 规则库,如图 16 所示。

4.2.3 入侵检测

再次运行 SYN Flood 程序进行攻击同时运行 Snort 进行监测,1 h 后观察告警日志,如图 17 所示。

图 16 补充 Snort 规则库

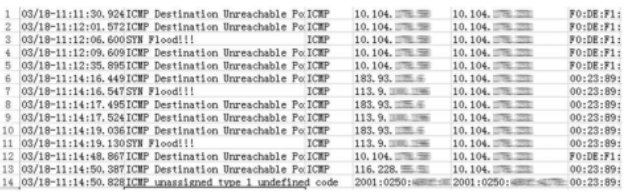


图 17 Snort 告警日志

4.3 结果分析

对比图 15 和图 17 可以看出,原始 Snort 入侵检测系统在第一次遇到 SYN Flood 攻击时无法识别攻击,只能简单地检测出一些 ICMP 数据包产生错误,但 Snort 系统经过基于关联规则的自学习后,规则库得到补充,当再次遇到 SYN Flood 攻击时,可以很快发现此攻击,产生图 17 所示的告警信息。上述结果表明所设计的关联分析器能够从 Snort 告警日志中挖掘出攻击间隐藏的关联信息,同时利用这些关联信息,可使 Snort 入侵检测系统具备一定的未知攻击检测能力。

5 结束语

针对现有 Snort 入侵检测系统本身不能发现海量安全日志数据背后隐含的有价值的信息,且仅能检测出已知攻击的局限性,本文设计了一种基于 Apriori 算法的关联分析器,以插件的形式加载到 Snort 中,克服现有 Snort 入侵检测系统的局限性。采用 Apriori 算法对 Snort 告警日志进行运算,挖掘出 Snort 海量告警日志间隐藏的有价值的关联规则,当数据流中出现关联规则选项时,系统会自动检测识别入侵行为,同时将关联规则经过相应转换后补充进 Snort 的规则库,使原先“未知”攻击变为“已知”攻击,从而使 Snort 可以间接识别尚未发现的攻击。实验结果验证了本文系统的有效性和实用性。

参考文献

[1] ANDERSON J P. Computer security threat monitoring and surveillance[R]. Fort Washington, Pennsylvania,

1980.

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

[3] SHARAFALDIN I, LASHKARI A H, GHORBANI A A. Toward generating a new intrusion detection dataset and intrusion traffic characterization[C]. Proceedings of the 4th International Conference on Information Systems Security and Privacy, 2018: 108-116.

[4] SHONE N, NGOC T N, PHAI V D, et al.A deep learning approach to network intrusion detection[J]. IEEE Transactions on Emerging Topics in Computational Intelligence, 2018, 2(1): 41-50.

[5] KWON D, KIM H, KIM J, et al.A survey of deep learning-based network anomaly detection[J]. Cluster Computing, 2019, 22(1): 949-961.

[6] VINAYAKUMAR R, ALAZAB M, KP S, et al.Deep learning approach for intelligent intrusion detection system[J]. IEEE Access, 2019: 41525-41550.

[7] LIN F, ZHOU Y, AN X, et al.Fair resource allocation in an intrusion-detection system for edge computing: ensuring the security of Internet of Things devices[J]. IEEE Consumer Electronics Magazine, 2018, 7(6): 45-50.

[8] ALOQAILY M, OTOUM S, AL RIDHAWI I, et al.An intrusion detection system for connected vehicles in smart cities[J]. Ad Hoc Networks, 2019, 90(JUL): 101842.1-101842.14.

[9] BUCZAK A, GUVEN E. A survey of data mining and machine learning methods for cyber security intrusion detection[J]. IEEE Communications Surveys & Tutorials, 2017, 18(2): 1153-1176.

[10] 王健, 王语杰, 韩磊.基于突变模型的SDN环境中DDoS攻击检测方法[J].信息网络安全, 2020, 20(5): 11-20.

[11] 王蓉, 马春光, 武朋.基于联邦学习和卷积神经网络的入侵检测方法[J].信息网络安全, 2020, 20(4): 47-54.

[12] 王洋, 吴建英, 黄金垒, 等.基于贝叶斯攻击图的网络入侵意图识别方法[J].计算机工程与应用, 2019, 55(22): 73-79.

[13] 李祉岐, 黄金垒, 王义功, 等.入侵告警信息聚合与关联技术综述[J].计算机应用与软件, 2019, 36(4): 286-294.

[14] 胡浩, 刘玉岭, 张红旗, 等.基于吸收Markov链的网络入侵路径预测方法[J].计算机研究与发展, 2018, 55(4): 831-845.

[15] HU H, LIU J, ZHANG Y, et al.Attack scenario reconstruction approach using attack graph and alert data mining[J]. Journal of Information Security and Applications, 2020, 54: 102522.1-102522.9.

(收稿日期: 2020-10-08)

作者简介:

刘金龙(1970-), 男, 本科, 主要研究方向: 网络安全与管理。

刘鹏(1986-), 男, 本科, 主要研究方向: 网络安全与维护。