基于机器学习和规则的网络异常流量检测研究

尤　刚[1]，徐　蕾[2]，李美鹏[1]，刘文杰[1]，张　鹏[1]，陆振奎[2]
（1. 96941 部队，北京　100085；2. 中国航天时代电子有限公司，北京　100094）

摘　要：网络异常流量检测的主流方法有基于机器学习的和基于规则匹配的，前者可以检测未知异常流量，后者可以精准指出攻击类型。结合两者优势，采用混合的方式实现网络异常流量检测系统。该系统设置了两道过滤器，第一道过滤器采用流聚类算法进行初步过滤，第二道过滤器采用开源工具 Suricata 进行精细识别。基于 DenStream 算法提出了一种可以根据网络中异常流量比例变化而动态确定半径阈值的流聚类算法 DenStream-DRT，此外，为改进 Suricata 存在无法识别未知异常流量的问题，提出了基于 Apriori 的含有效负载约束规则的生成算法 PCRG-Apriori，最后将基于规则的网络入侵检测系统 Suricata 与 DenStream-DRT 分类器进行了整合，形成了一个全新的网络异常流量检测系统。实验证明，集成系统在速率和准确性方面都有较好的表现。

关键词：网络安全；流聚类算法；Apriori 算法；Suricata；异常流量检测系统
中图分类号：TP309　　**文献标识码：**A　　**DOI：**10.19358/j.issn.2097-1788.2025.02.001
引用格式：尤刚，徐蕾，李美鹏，等. 基于机器学习和规则的网络异常流量检测研究［J］. 网络安全与数据治理，2025，44(2)：1-9.

Research on abnormal network traffic detection based on machine learning and rule-based methods

You Gang[1], Xu Lei[2], Li Meipeng[1], Liu Wenjie[1], Zhang Peng[1], Lu Zhenkui[2]
(1. Unit 96941 of PLA, Beijing 100085, China; 2. China Aerospace Times Electronics Co., Ltd., Beijing 100094, China)

Abstract: The mainstream methods of network abnormal traffic detection are machine learning-based and rule matching-based. The former can detect unknown abnormal traffic, and the latter can accurately point out the type of attack. In order to combine the advantages of the two, this paper uses a hybrid method to realize the network abnormal traffic detection system. The system is equipped with two filters. The first filter uses the stream clustering algorithm for preliminary filtering, and the second filter uses the open source tool Suricata for fine identification. Based on DenStream algorithm, this paper proposes a flow clustering algorithm DenStream-DRT, which can dynamically determine the radius threshold according to the change of the proportion of abnormal traffic in the network. In addition, in order to improve the problem that Suricata cannot recognize unknown abnormal traffic, this paper proposes a generation algorithm PCRG-Apriori with payload constraint rules based on Apriori. Finally, the rule-based network intrusion detection system Suricata is integrated with the DenStream-DRT classifier to form a new network abnormal traffic detection system. Experimental results show that the integrated system has good performance in speed and accuracy.

Key words: network security; flow clustering algorithm; Apriori algorithm; Suricata; abnormal traffic detection system

0　引言

在网络安全领域，网络异常流量检测至关重要。当前网络异常流量检测方法主要有基于机器学习、基于规则以及两者混合的。

机器学习中的有监督学习方法依赖标注好的数据，在数据集质量高时能实现较好的检测效果。例如，Hu[1]等人提出了鲁棒性的 SVM 算法，展现出对噪声处理的强大能力，增强了模型的稳定性；Kabir 等人[2]提出了一个改进的 SVM 方法 LS-SVM，实验结果证明该方法在准确性和效率方面有了显著提升。机器学习中的半监督学习介于监督和无监督之间，通过结合已标注正例与未标注数据训练模型，可实现较好分类性能。Jabbar 等人[3]提出了一个以迭代的方式进行聚类的半监督学习器，实验结果显示该方法可以实现较高的准确率和较低的误报率。机

器学习中的无监督学习算法不依赖标注数据集，适应性强，但准确性不如有监督学习，且误报率较高。Syarif 等人[4]研究对比了常用的聚类和有监督学习方法，实验结果显示无监督的聚类算法误报率较高，约为 20%。

基于规则的网络异常流量检测通过将专家定义的规则与流量进行匹配来识别异常流量。Suricata 是一个开源的网络入侵检测和阻止引擎，其在多方面表现出色，但存在无法检测未知流量、实时性差等局限。

混合网络异常流量检测有串行和并行两大方向。并行检测中基于规则的工具和基于机器学习的分类器同步运作。例如，Shah 等人[5]提出了一个并行处理框架，将 Snort 与 SVM 同时运作，实验显示该系统具有较好的检测精度。串行检测则顺序运用两者。例如，Chiba[6]等人介绍了一种以 Suricata 和隔离森林算法为核心的检测框架，其中 Suricata 作为初步过滤器，由隔离森林算法进行进一步的异常流量识别，实现了对未知攻击的有效检测。

考虑到系统的效率，本文选择构建串行的检测系统，即将基于机器学习的检测方法作为第一道过滤器，将基于规则的工具作为第二道过滤器。然而，现行的流聚类算法存在准确率较低的问题，导致过多可疑流量被传递至 Suricata 系统；此外，Suricata 存在无法识别未知异常流量的问题。本文对上述问题进行了改进研究：（1）针对流聚类算法准确率较低的问题，提出了一种可以动态确定半径阈值的流聚类算法，并进行了对比实验；（2）针对 Suricata 系统仅能识别已知的异常流量问题，提出了基于 Apriori 的含有效负载约束的规则生成算法；（3）将基于规则的 Suricata 系统和基于机器学习的流聚类算法集成，并进行了消融实验[7]。

1　DenStream 及其优化算法

1.1　DenStream 算法优缺点分析

DenStream 算法（Density-based Streaming clustering algorithm）包括以下几个优点[8]：（1）在处理数据方面具有实时性；（2）能够发现任意形状的簇，并且不需要事先确定簇的数量；（3）具有处理异常值的能力。但 DenStream 也有很多缺点：（1）在实际的 Web 服务器流量分析中，由于网络流量的正常与异常比例可能发生波动，导致与之相关的潜在核心微簇和异常微簇半径同样会发生变化；（2）实时性是网络异常流量分析的一个关键要求，然而，DenStream 算法离线部分聚合功能不能协助实时异常的判断，且离线部分算法时间复杂度较高。

1.2　DenStream 的优化

本文对于 DenStream 主要从两个方面进行优化：

（1）为了使 DenStream 适应网络流量中正常流量和异常流量的比值变化和提升系统的可移植性，本文提出了一种可以动态确定半径阈值的流聚类算法，并将其命名为 DenStream-DRT（DenStream with Dynamic Radius Threshold），半径阈值公式如式（1）所示：

$$\varepsilon(k) = \bar{r} \pm k_r \sigma_r \tag{1}$$

其中，$\bar{r}$ 是簇半径的平均值的增量估计，其初始值为模型构建开始时前 S 个正常样本合并在一起所获得的簇的半径，之后便随着新样本的进入不断更新；σ_r 是簇半径的标准差的增量估计；k_r 用于计算数据流中半径大小的置信区间，可以允许控制。

（2）鉴于本研究主要利用 DenStream 算法的在线部分来实现异常的实时判定，且无需深入分析其最终的聚类成果，故删除算法中的离线逻辑部分。

具体改进后算法的伪代码如算法 1 所示，算法步骤如下：

（1）初始化潜在核心微簇，使用正常流量输入 DBSCAN 算法；

（2）尝试将新样本合并到潜在核心微簇；

（3）若无法合并到最近的潜在核心微簇，则下一步尝试将此样本合并到异常值微簇；

（4）对潜在核心微簇和异常值微簇进行更新。

算法 1：DenStream-DRT

```
使用正常流量输入 DBSCAN 算法，初始化核心微簇；
while 新样本 p 到来 do
  找到最近的潜在核心微簇并尝试合并；
  if r_c < ε [r̄ + k_r σ_r]  then
    将新样本合并到该潜在核心微簇；
    返回正常标签；
  else
    找到最近的异常值微簇并尝试合并；
    if r_o < ε [r̄ + k_r σ_r]  then
    将新样本合并到该异常值微簇；
    if 该异常值微簇的权重 ω_o > β_μ then
      将该异常值微簇变成潜在核心微簇；
    end
  else
    为该样本新生成一个异常值微簇；
  end
  返回异常标签；
end

更新所有潜在核心微簇的 ω_c；
if  t  mod Tp  = = 0 then
  for 每个潜在核心微簇 do
    if 该异常值微簇的权重 ω_o < β_μ then
```

```
        将该潜在核心微簇变成异常值微簇;
      end
    end
    for 每个异常值微簇 do
      将新样本合并到该异常值微簇;
      if 该异常值微簇的权重 ω_p <β_μ then
        删除该异常值微簇;
      end
    end
  end
```

1.3 构建网络异常流量数据集

本文构建了一个包括 DoS 攻击、XSS 攻击和 SQL 注入攻击的数据集，并将该数据集命名为“NSDWID”。该数据集是从模拟的仿真网络中采集的，包含正常网络流量以及 DoS 攻击、XSS 攻击和 SQL 注入攻击异常流量类型。仿真网络包括 Web 服务器主机、中心路由器、实验室网络和其他网络。图 1 为仿真网络的网络拓扑结构。

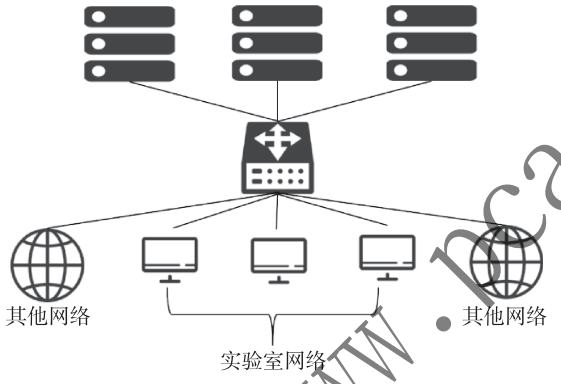

图 1 仿真网络的网络拓扑结构

流量的收集过程分为两步。首先对正常流量进行采集，本文参考文献［9］，采集过程中对正常流量采集的占比 TCP ∶ UDP ∶ ICMP 为 85 ∶ 10 ∶ 5。其次对于异常流量的采集，在实验室网络中使用 DoS 攻击工具 hping3[10] 产生异常流量。hping3 是一个灵活的网络工具，可以用来发送定制的 synflood[11] 攻击、udpflood[12] 攻击和 icmpflood[13] 攻击，其混合比例为 1 ∶ 1 ∶ 1。在实验室网络模拟 SQL 注入攻击和 XSS 攻击，其混合比例为 1 ∶ 1。

1.4 Clustream、DenStream、DenStream-DRT 仿真对比

为了验证所提出算法的可用性，以及该算法在 Web 服务器场景中的效果，本节使用 KDD99 数据集和本文构建的 NSDWID 数据集进行验证。

1.4.1 实验过程

（1）特征属性选取

在 KDD99 数据集中每条记录包含 41 个特征属性，本文参考 Li[14] 等人对 KDD99 数据集选取的 12 个特征属性进行实验。这些特征的详细信息如表 1 所示。NSDWID 数据集选取与 KDD99 数据集一样的特征属性。

表 1 特征选取

序号	属性字段	数据所属类型	属性描述
1	service	符号型	服务类型
2	protocol_type	符号型	协议类型
3	flag	符号型	tcp 标志位
4	src_bytes	数值型	字节数，由源主机发到目的主机
5	dst_bytes	数值型	字节数，由目的主机发到源主机
6	logged_in	数值型	0 或 1，成功登录为 1，失败为 0
7	count	数值型	过去 2 s 内，同当前连接有同一目标主机的连接
8	srv_count	数值型	过去 2 s 内，同当前连接有同一服务的连接
9	same_srv_rate	数值型	过去 2 s 内，在与当前连接具有相同目标主机的连接中，与当前连接具有相同服务的连接的百分比
10	srv_diff_host_rate	数值型	过去 2 s 内，在与当前连接具有相同服务的连接中，与当前连接具有不同目标主机的连接的百分比
11	dst_host_count	数值型	前 100 个连接中，同当前连接有相同主机的连接数
12	dst_host_srv_count	数值型	前 100 个连接中，同当前连接有相同主机和服务的连接数

（2）数据集预处理

数据集预处理包含三个关键步骤：填补缺失信息、数值化非数值型特征属性、归一化处理。这三步的预处理工作为后续的流聚类算法奠定了一个基础，特别是在处理大规模的数据集场景下。

（3）划分数据集

网络中正常流量和异常流量的比例在实验室环境和在实际的应用环境中有巨大差异。实际的应用环境中会受到外部攻击模式的变化和日常流量突变等因素的影响。

网络流量比例的差异直接影响本文所选取算法中簇的大小，本文所提出的 DenStream-DRT 算法可以很好地根据所输入的异常流量的比例自动调节半径阈值，达到较为良好的表现。

为了验证提出的 DenStream-DRT 算法在面对不同异常流量比例时，其自动调整半径阈值功能的有效性以及算法准确率的提升情况，设计了包含不同异常流量比例的测试数据集，分别输入到 DenStream-DRT 算法、Clustream 算法以及 DenStream 算法进行对比实验。由于 KDD99 数据集拥有丰富的数据量，本研究将其分割为三组不同比例的正常流量与异常流量测试集（1 ∶ 1，4 ∶ 1，100 ∶ 1），模拟了从相对均衡的场景到高度非均衡的场景。另外，本研究也在 NSDWID 数据集上进行了实验，该数据集维持了其原始的正常流量与异常流量比例（10 ∶ 1）。各个比例测试集中的流量总和、正常流量总和、异常流量总和以及各异常流量类型数量均在表 2 中进行了明确展示。

表 2　不同场景中的正常数据和异常数据的数量

统计数据标识	KDD99			NSDWID
	1 ∶ 1	4 ∶ 1	100 ∶ 1	10 ∶ 1
流量总和	194 000	121 250	97 970	11 000
正常流量总和	97 000	97 000	97 000	10 000
异常流量总和	97 000	24 250	970	1 000
Probe	4 107	4 107	300	—
DoS	91 715	18 965	300	500
U2R	52	52	52	—
R2L	1 126	1 126	318	—
SQL Injection	—	—	—	250
XSS	—	—	—	250

1.4.2　结果分析

基于不同场景下安全需求和性能需求的不同，实验分别得出以下两种情况算法在准确率、误报率、召回率、*F*2 分数和时间上的表现：（1）在召回率高于 80% 时，三种算法误报率最低的情况下；（2）在召回率高于 95% 时，三种算法误报率最低的情况下。两种情况下实验结果分别如表 3、表 4 所示。

在上述两个实验中，对提及的三种流聚类算法的实验结果进行对比分析，可以看出，DenStream-DRT 在三种算法中表现出较低的误报率、较高的准确率、较高的 *F*2 分数和更好的时间性能。因此，可以说明 DenStream-DRT 算法能够在高召回率的同时，减少本研究第二道过滤器 Suricata 匹配的数量，大大降低了其丢包的风险，且没有因为动态调节半径阈值增加时间复杂度。

从上述实验结果中还可以看出，本研究提出的算法即使在高召回率的情况下，在低异常流量比例和高异常流量比例的网络环境中均能有效区别正常与异常流量。并且随着异常数据比例的减少，算法的误报率也有所降低，且优于另外两种算法也越来越凸显。

表 3　DenStream-DRT、DenStream、Clustream 的准确率、误报率、召回率、*F*2 分数和时间（召回率高于 80%）

正常 ∶ 异常	算法	准确率	误报率	召回率	*F*2 分数	时间/s
1 ∶ 1	DenStream-DRT	**0. 85**	**0. 12**	**0. 81**	**0. 82**	**13. 82**
	DenStream	0. 76	0. 27	0. 80	0. 79	18. 93
	Clustream	0. 73	0. 35	0. 80	0. 78	24. 19
4 ∶ 1	DenStream-DRT	**0. 89**	**0. 09**	0. 80	**0. 78**	**8. 1**
	DenStream	0. 80	0. 20	0. 80	0. 71	13. 26
	Clustream	0. 70	0. 33	0. 80	0. 66	18. 94
100 ∶ 1	DenStream-DRT	**0. 94**	**0. 06**	0. 81	**0. 37**	**6. 78**
	DenStream	0. 85	0. 15	0. 81	0. 20	9. 2
	Clustream	0. 69	0. 31	0. 81	0. 11	14. 94
10 ∶ 1	DenStream-DRT	**0. 87**	**0. 12**	0. 80	**0. 67**	**0. 89**
	DenStream	0. 82	0. 18	**0. 81**	0. 61	1. 43
	Clustream	0. 68	0. 33	0. 80	0. 50	1. 41

表 4　DenStream-DRT、DenStream、Clustream 的准确率、误报率、召回率、*F*2 分数和时间（召回率高于 95%）

正常：异常	算法	准确率	误报率	召回率	*F*2 分数	时间/s
1 ：1	DenStream-DRT	**0.86**	**0.23**	0.95	**0.92**	**13.2**
	DenStream	0.83	0.30	**0.96**	0.91	19.11
	Clustream	0.77	0.41	0.95	0.89	23.08
4 ：1	DenStream – DRT	**0.82**	**0.21**	**0.96**	**0.83**	**8.48**
	DenStream	0.76	0.29	0.95	0.78	12.77
	Clustream	0.73	0.32	0.95	0.76	19.23
100 ：1	DenStream – DRT	**0.85**	**0.15**	**0.96**	**0.24**	**6.19**
	DenStream	0.80	0.20	0.95	0.19	8.95
	Clustream	0.67	0.33	0.95	0.13	14.59
10 ：1	DenStream – DRT	**0.81**	**0.20**	0.95	**0.32**	**0.77**
	DenStream	0.74	0.28	0.95	0.25	1.21
	Clustream	0.73	0.29	0.95	0.25	1.33

本文所提出的算法在低异常流量比例的实验中 *F*2 分数较低，是由于输入的数据集正常比例和异常比例及其不平衡造成的，在本文提出的框架中的第二道过滤器中会进行更精准的规则匹配，因此会降低系统的总体误报率，提高系统的 *F*2 分数。

相对于在召回率高于 80% 情况下的实验，DenStream-DRT 算法召回率在增加 15% 的同时，误报率只增长了 10% 左右。

2　基于 Apriori 的含有效负载约束的规则生成算法

2.1　算法详细流程

已有研究基于 Apriori 算法生成仅包含规则头部信息的规则[15-16]，这虽然在发现未知异常流量方面有一定优势，但信息的不足可能会导致高误报率。因此，在生成规则时需要添加其他约束信息，以生成更加详细的异常流量规则。Suricata 规则可以根据流量特征定义多种规则选项约束，但通常需要大量的处理资源和时间，因此本研究仅使用常用的字段来建立约束，以实现实时处理。根据该原则，除了选择规则头部、规则选项中的通用选项以外，还添加规则选项中的有效负载选项约束。

根据上述分析，本文提出了基于 Apriori 的含有效负载约束的规则生成算法，并将其命名为 PCRG-Apriori（Apriori-based Payload-Constrained Rule Generation Algorithm）。PCRG-Apriori 算法包括以下步骤：

（1）使用 Apriori 算法对频繁内容进行提取

使用 Apriori 算法对频繁内容进行提取，对应的伪代码如算法 2 所示。

算法 2：频繁内容提取算法

```
Input: SequenceSet = {S1, S2, …, Ss}, MINSupport
Output: ContentSet = {L1, L2, …, Lc}
for 序列 S in SequenceSet  do
  for 字符 a in S do
    if a 不在 C1 集合中 then
      加入到 C1 集合中;
    end
  end
end
k = 2;
while Ck = Φ do
    for k – 1 项集 c in Ck-1 do
      for 序列 S in SequenceSet do
        if C 存在于 Si 中 then
          count = count + 1;
        end
      end
      if (count/s) < MINSupport then
        将该项集 C 从 Ck-1 中删除;
      end
    end
    Lk = Ck;
    k + +;
for k – 1 项集 l1  in Lk-1 do
      for k – 1 项集 l2 in Lk-1 do
        if (l1.a1 = = l2.a2) && (l1.a2 = = l2.a3)
                  && (l1.ak-2 = = l2.ak-1) then
```

```
            将 l_{2·a_1}, l_{1·a_1}, l_{1·a_2}…, l_{1·a_{k-1}}, 加入到 C_k 集合中;
        end
      end
    end
    for k 项集 c  in C_k  do
      for “（k-1）项子集” of “项集 c”  do
        if c 不在 L_{k-1}集合中 then
          从 C_k 集合中删除 C_i;
        end
      end
    end
  end
```

（2）检查所有长度的频繁内容之间包含关系并挑选出异常内容

检查所有长度的频繁内容之间的包含关系算法对应伪代码如算法 3 所示。该算法逻辑为：如果一个序列被包含于另一个序列，则将其从集合中删除。

此时集合中序列的数量会大大减少，再根据专家知识，从筛减后的集合中挑选出异常内容。

算法 3：检查包容关系算法

```
Input：项集 C
Output：项集 C
for S_1 in 项集 C  do
  for S_2 in 项集 C  do
    if  S_2 中包含 S_1  then
      将 S_1从项集 C 中删除;
      break;
    end
  end
end
```

（3）确定异常内容位置信息（有效负载约束）

确定异常内容位置信息对应的伪代码如算法 4 所示。

算法 4：内容定位算法

```
Input：项集 C，数据流集合 = {P_1, P_2, …, P_n}, L_K
Output：offset, depth
offset = Maxpacketsize;
depth = 0;
for 每个数据流 p  do
  if C 存在 p 中 then
    offset = MIN（offset）;
    depth = MAX（depth）;
  end
end
```

（4）生成规则头部信息

第三步已经得到包含该异常内容的数据流序号。若序号唯一，则将该数据流的头部信息添加到规则中。如果序号不唯一，则需要对 IP 和端口进行泛化处理。例如有多个端口，可以写成端口段或者端口列表的形式；若有多个 IP 地址，则需写成 IP 列表或 CIDR 的形式。

（5）组合信息生成规则

将以上步骤所生成的规则头部信息和规则选项部分进行组合，并定义规则选项中的常规选项等部分，生成符合标准的 Suricata 规则。

2.2 数据集介绍

本研究是面向 Web 服务器场景进行网络异常流量检测，因此本文仅对 NSDWID 数据集中 HTTP 协议的有效负载字段进行分析。

对 NSDWID 原始数据集 PCAP 文件中的有效负载字段内容通过 Wireshark 工具以及 Python 中的 Dpkt 、Scapy 等模块进行提取，得到输入到算法中的序列。上述数据流中有效负载字段如图 2 所示。将以上有效负载字段以 TXT 文件格式进行保存。

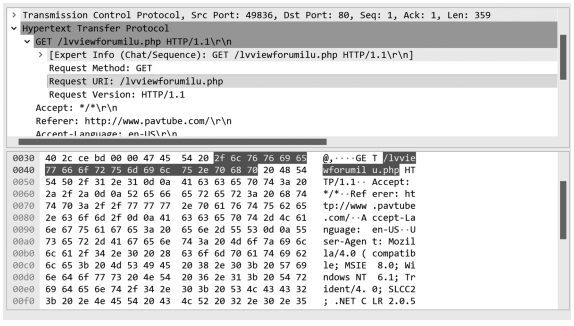

图 2　HTTP 请求中的有效负载

2.3 结果分析

Suricata 是一个开源的网络入侵检测系统，可以被用来检测和预防各层协议的网络攻击[17]。当找到一个异常内容，并根据该异常内容得到对应数据流的信息后，就可以根据以上得到的信息生成各层所对应的规则。以下是根据 PCRG-Apriori 算法生成的规则。

（1）空字节注入攻击

图 3 中展示的是空字节注入异常内容与 483 号数据流

相匹配生成的两层协议规则：第 3/4 层规则和第 7 层规则。

```
----- Suricata Rules For Packet Number 483-----
----- Layer 3/4 Rules -------
----- TCP ---

alert tcp 172.16.2.96 49192-> $HOME_NET any (msg: "Suspicious IP 172.16.2.96 and port 49192 detected!"; re
ference:Packet2Suricata; classtype:trojan-activity; sid:xxxx; rev:1;)
alert tcp $HOME_NET any -> 65.181.112.240 80 (msg: "Suspicious IP 65.181.112.240 and port 80 detected!"; r
eference:Packet2Suricata; classtype:trojan-activity; sid:xxxx; rev:1;)

----- Layer 7 Rules -----
----- HTTP -----


alert HTTP any any -> any $HTTP_PORTS (msg: "Null Byte Injection Attack";flow:established,to_server; http_
uri; content:"/etc/passwd\%00"; nocase; offset:1, depth:43;reference:Packet2Suricata; classtype:Null Byte
Injection Attack; sid:xxxx; rev:1;)

Don't forget to change the sid of the generated rule(s)!
```

图 3　针对空字节注入攻击所形成的规则

（2）SQL 注入攻击

图 4 中展示的是 SQL 注入异常内容与 1885 号数据流相匹配生成的两层协议规则：第 3/4 层规则和第 7 层规则。

（3）HTTP 响应拆分攻击

图 5 中展示的是 HTTP 响应拆分攻击的异常内容与 378 号数据流相匹配生成的两层协议规则：第 3/4 层规则和第 7 层规则。

3　面向 Web 服务器的流量入侵检测系统的设计与实现

3.1　总体架构

本文综合基于机器学习检测与基于规则检测这两种检测算法的优势，同时结合聚类分析和关联规则的特性，构建一个能够最大化利用两类方法优点的入侵检测模型，如图 6 所示。入侵检测系统的具体设计共包括流量采集模块、流量检测模块、规则生成模块和流量可视化模块。

```
----- Suricata Rules For Packet Number 1885-----
----- Layer 3/4 Rules -------
----- TCP ---

alert tcp 172.16.2.96 49192-> $HOME_NET any (msg: "Suspicious IP 172.16.2.96 and port 49192 detected!"; re
ference:Packet2Suricata; classtype:trojan-activity; sid:xxxx; rev:1;)
alert tcp $HOME_NET any -> 65.181.112.240 80 (msg: "Suspicious IP 65.181.112.240 and port 80 detected!"; r
eference:Packet2Suricata; classtype:trojan-activity; sid:xxxx; rev:1;)

----- Layer 7 Rules -----
----- HTTP -----


alert HTTP any any -> any $HTTP_PORTS (msg: "Possible SQL Injection Attack - Detected Malicious Keyword";f
low:established,to_server; http_uri; content:"union|select|insert|delete|update|drop"; nocase; offset:20,
depth:34;reference:Packet2Suricata; classtype:Possible SQL Injection Attack - Detected Malicious Keyword;
sid:xxxx; rev:1;)

Don't forget to change the sid of the generated rule(s)!
```

图 4　针对 SQL 注入攻击所形成的规则

```
----- Suricata Rules For Packet Number 378-----
----- Layer 3/4 Rules -------
----- TCP ---

alert tcp 172.16.2.96 49191-> $HOME_NET any (msg: "Suspicious IP 172.16.2.96 and port 49191 detected!"; re
ference:Packet2Suricata; classtype:trojan-activity; sid:xxxx; rev:1;)
alert tcp $HOME_NET any -> 65.181.112.240 80 (msg: "Suspicious IP 65.181.112.240 and port 80 detected!"; r
eference:Packet2Suricata; classtype:trojan-activity; sid:xxxx; rev:1;)

----- Layer 7 Rules -----
----- HTTP -----


alert HTTP any any -> any $HTTP_PORTS (msg: "HTTP Response Splitting Attack";flow:established,to_server; h
ttp_uri; content:"%0aset-cookie%3a+tamper%5d"; nocase; offset:20, depth:34;reference:Packet2Suricata; clas
stype:HTTP Response Splitting Attack; sid:xxxx; rev:1;)

Don't forget to change the sid of the generated rule(s)!
```

图 5　针对 HTTP 响应拆分攻击所形成的规则

3.2　系统速率测试

检测系统的速率实验是在保证流量相同的情况下，记录检测花费的时间长短。该实验不重点关注检测结果的误报率和准确率。由于本文构建的数据集数据量较小，因此本节实验仿真时采用 KDD99 数据集的原始数据集作为系统的测试数据，并通过多次取平均值使实验数据更具有说服力。此外，需要注意的是 DenStream-DRT 算法所输入的是处理过的数据，不包括数据采集模块的解析数据处理时间。不同算法输入不同数据量的用时结果如表 5 所示。

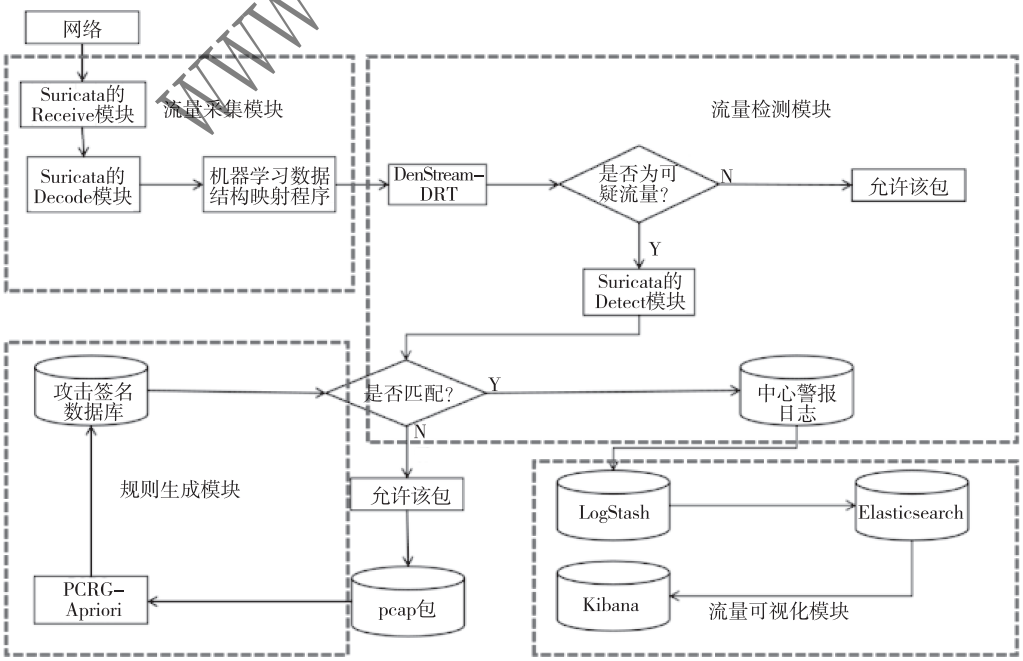

图 6　面向 Web 服务器的流量入侵检测系统模型

表 5　KDD99 数据集输入传统 Suricata 系统、DenStream-DRT 和本文改进的系统时间对比

	传统 Suricata 系统/s	DenStream-DRT /s	本文改进的系统/s	相较于传统 Suricta 系统提升百分比/%
1 000	32. 23	0. 12	31. 44	2. 45
2 000	62. 45	0. 23	54. 35	12. 97
5 000	226. 25	0. 58	162. 90	28
10 000	947. 71	1. 16	594. 12	37. 31
50 000	2 081. 20	5. 84	1 121. 83	46. 10
100 000	5 172. 77	7. 21	2 792. 88	46. 01
200 000	12 160. 96	13. 91	6 810. 13	44

从表 5 可以看出，由于第一道过滤器对全量的流量进行分流，使得输入到 Suricata 的流量大幅度减少，继而节省了很多匹配大量规则的时间。本文改进后的系统随着数据量的增加，其系统实时性的优势体现得更加明显。相较于传统 Suricata 系统，本文所提出的系统在性能上提升稳定于 45%。这降低了传统 Suricata 系统丢包的风险。

3.3　系统误报率、准确率测试

本节实验所使用的数据集各类样本的数量如表 6 所示。

表 6　本实验数据集各类型数据数量分布

Normal	DDoS	Web 应用程序注入攻击
10 000	500	500

将上述数据集分别输入传统 Suricata 系统、DenStream-DRT 算法、DenStream-DRT 算法 + 规则头部信息生成算法、本文集成系统以及加入新规则后的系统（新规则是指：将本文数据集输入到本文所提出的系统中，经过 DenStream 过滤器输出可疑流量，对可疑流量再由 Suricata 系统进行匹配，对于未被匹配成功的可疑流量根据 2. 1 节的规则生成算法生成的新规则），准确率、召回率和误报率如表 7 所示。

表 7　使用本文数据集输入四种框架和加入生成规则后系统的性能对比

	准确率/%	召回率/%	误报率/%
传统 Suricata 系统	94. 06	92. 1	5. 74
DenStream-DRT 算法	79. 5	95. 6	22. 11
DenStream-DRT 算法 + 规则（规则头部信息）生成算法	89. 5	45. 64	2. 05
本文系统	97. 47	89. 1	1. 69
加入生成规则后的系统	97. 57	93. 5	2. 02

传统 Suricata 系统在认真筛选规则后确实可以达到较高的准确率和召回率，在本实验中准确率和召回率可以分别达到 94. 06% 和 92. 1%；但存在时间效率较低和误报率较高的问题。DenStream-DRT 算法时间效率高，但在保证召回率的前提下，准确率较低，误报率较高，在本文实验中分别为 79. 5% 和 22. 11%。所以本文将 DenStream-DRT 算法作为第一道过滤器，这样可以过滤掉接近 80% 的正常流量，以提升第二道过滤器的时间效率。对于本文构建的系统，由于过滤掉了接近 80% 的正常流量，因此准确率、误报率相较传统的 Suricata 系统有较大提升，在本文实验中分别为 97. 47% 和 1. 69%；由于 DenStream-DRT 算法存在漏报现象，导致本文系统最终的召回率下降。但本文所提出的系统会根据第 2 节所提出的算法不断添加新的规则，其召回率会有所提升，当加入生成的新规则后，召回率由原来的 89. 1% 提升到 93. 5%（在实际运行中由于加入新规则的滞后性，其召回率的提升幅度会受影响）。

4　结论

本文在 Web 服务器的场景下，对基于规则和机器学习的网络入侵检测框架所存在的问题进行了研究，针对流聚类算法准确率低的问题，提出了一种可以动态确定半径阈值的流聚类算法 DenStream-DRT；针对 Suricata 无法进一步识别未知异常流量问题，本文基于网络异常流量通常具有重复模式，提出了基于 Apriori 的含有效负载约束的规则生成算法 PCRG-Apriori。为了评估所提出的系统在 Web 服务器中的实际应用效能，本文将基于规则网络入侵检测系统 Suricata 与 DenStream-DRT 分类器进行了整合，并给出最终测试结果，对工程实践具有一定的指导意义。下一步需要针对更多的分类器进行对比测试，以期给出更全面的结论。

参考文献

[1] HU W J, LIAO Y H, VEMURI V R. Robust anomaly detection using support vector machines [C]//Proceedings of the International Conference on Machine Learning, 2003.

[2] KABIR E, HU J, WANG H, et al. A novel statistical technique for intrusion detection systems [J]. Future Generation Computer Systems, 2018, 79: 303 – 318.

[3] JABBAR M A, ALUVALU R, REDDY S S S. Cluster based ensemble classification for intrusion detection system [C]//Proceedings of the 9th International Conference on Machine Learning and Computing, 2017: 253 – 257.

[4] SYARIF I, PRUGEL-BENNETT A, WILLS G. Unsupervised clustering approach for network anomaly detection [C]//Networked Digital Technologies: 4th International Conference.

Springer Berlin Heidelberg, 2012: 135 – 145.

[5] SHAH S A R, ISSAC B. Performance comparison of intrusion detection systems and application of machine learning to Snort system [J]. Future Generation Computer Systems, 2018, 80: 157 – 170.

[6] CHIBA Z, ABGHOUR N, MOUSSAID K, et al. Newest collaborative and hybrid network intrusion detection framework based on suricata and isolation forest algorithm [C]//Proceedings of the 4th International Conference on Smart City Applications, 2019: 1 – 11.

[7] CAO F, ESTERT M, QIAN W, et al. Density-based clustering over an evolving data stream with noise [C]//Proceedings of the 2006 SIAM International Conference on Data Mining, 2006: 328 – 339.

[8] FAHY C, YANG S, GONGORA M. Scarcity of labels in non – stationary data streams: a survey [J]. ACM Computing Surveys (CSUR), 2022, 55 (2): 1 – 39.

[9] BRAGA R, MOTA E, PASSITO A. Lightweight DDoS flooding attack detection using NOX/OpenFlow [C]//IEEE Local Computer Network Conference. IEEE, 2010: 408 – 415.

[10] 李传煌, 孙正君, 袁小雍, 等. 基于深度学习的实时 DDoS 攻击检测 [J]. 电信科学, 2017, 33 (7): 53 – 65.

[11] SUDAR K M, DEEPALAKSHMI P, SINGH A, et al. TFAD: TCP flooding attack detection in software-defined networking using proxy-based and machine learning-based mechanisms [J]. Cluster Computing, 2023, 26 (2): 1461 – 1477.

[12] WANG Y C, DING J X, ZHANG T, et al. From replay to regeneration: recovery of UDP flood network attack scenario based on SDN [J]. Mathematics, 2023, 11 (8). DOI: 10.3390/math11081897.

[13] ANUSUYA R, PRABHU M R, PRATHIMA C, et al. Detection of TCP, UDP and ICMP DDOS attacks in SDN using machine learning approach [J]. Journal of Survey in Fisheries Sciences, 2023, 10 (4S): 964 – 971.

[14] LI Y, XIA J, ZHANG S, et al. An efficient intrusion detection system based on support vector machines and gradually feature removal method [J]. Expert Systems with Applications, 2012, 39 (1): 424 – 430.

[15] 李恩燕. 基于经典聚类算法和关联算法的入侵检测系统研究 [D]. 重庆: 重庆邮电大学, 2020.

[16] 董福洋. 基于数据挖掘的入侵检测系统设计与实现 [D]. 南京: 南京理工大学, 2020.

[17] ALBIN E, ROWE N C. A realistic experimental comparison of the Suricata and Snort intrusion-detection systems [C]//2012 26th International Conference on Advanced Information Networking and Applications Workshops. IEEE, 2012: 122 – 127.

(收稿日期: 2024 – 12 – 01)

作者简介:

尤刚 (1985 –), 男, 博士, 工程师, 主要研究方向: 大数据分析与应用、云计算。

徐蕾 (1998 –), 女, 硕士, 工程师, 主要研究方向: 数据挖掘、云计算。

李美鹏 (1996 –), 男, 本科, 助理工程师, 主要研究方向: 大数据分析与应用、数据挖掘。

版权声明

凡《网络安全与数据治理》录用的文章，如作者没有关于汇编权、翻译权、印刷权及电子版的复制权、信息网络传播权与发行权等版权的特殊声明，即视作该文章署名作者同意将该文章的汇编权、翻译权、印刷权及电子版的复制权、信息网络传播权与发行权授予本刊，本刊有权授权本刊合作数据库、合作媒体等合作伙伴使用。同时，本刊支付的稿酬已包含上述使用的费用，特此声明。

《网络安全与数据治理》编辑部