第47卷 第8期 计 算 机 工 程 2021年8月
Vol.47 No.8 Computer Engineering August 2021

·网络空间安全· 文章编号：1000-3428(2021)08-0140-09 文献标志码：A 中图分类号：TP309

基于侧信道特征的安全代理流量分类方法

高 平[1]，广 晖[2]，陈 熹[1]，李光松[1]

（1.河南省网络密码技术重点实验室，郑州 450001；2.郑州大学 体育学院，郑州 450004）

摘 要：安全代理被越来越多的互联网用户用于规避网络审查和访问受限资源，因此安全代理流量的分类对于网络安全和网络管理具有重要意义。为弥补深度包检测技术在过滤和识别不良信息上的不足，提高防火墙流量探测能力，提出一种安全代理流量分类方法。提取用于安全代理流量分类的侧信道特征，包括有效载荷长度序列、信号序列等，使用机器学习和深度学习算法对Shadowsocks、V2Ray、Freegate、Ultrasurf 4种被广泛使用的安全代理流量进行识别。实验结果表明，通过提取与有效载荷内容无关的侧信道特征进行分类，与MLP、LSMP等算法相比，该方法在准确率、F1值等性能方面均有提升。

关键词：安全代理；流量分类；机器学习；深度学习；深度包检测

开放科学（资源服务）标志码（OSID）：

中文引用格式：高平，广晖，陈熹，等. 基于侧信道特征的安全代理流量分类方法[J]. 计算机工程，2021，47(8)：140-148，156.

英文引用格式：GAO P，GUANG H，CHEN X，et al. Traffic classification method based on side-channel features for security proxy[J]. Computer Engineering，2021，47(8)：140-148，156.

Traffic Classification Method Based on Side-Channel Features for Security Proxy

GAO Ping[1]，GUANG Hui[2]，CHEN Xi[1]，LI Guangsong[1]

（1.Henan Key Laboratory of Network Cryptography Technology，Zhengzhou 450001，China；
2.Physical Education College，Zhengzhou University，Zhengzhou 450004，China）

【Abstract】In recent years，security proxy has been utilized by more and more users to circumvent Internet censorship and access restricted resources. The classification of security proxy traffic is of great significance for network security and network management. In order to make up for the insufficiency of the existing deep packet inspection technology and improve firewall traffic detection capability，a secure proxy traffic classification method is proposed. The side-channel features used for security proxy traffic classification are extracted，including the payload length sequences and the signal sequence. Then based on these features，machine learning and deep learning algorithms are used to identify the traffic of widely used security proxies，including Shadowsocks，V2Ray，Freegate，and Ultrasurf. Experimental results show that compared with algorithms such as MLP and LSMP，the proposed traffic classification method provides higher accuracy，F1 value and other performance indicators.

【Key words】secure proxy；traffic classification；machine leaning；deep learning；deep packet inspection

DOI：10. 19678/j. issn. 1000-3428. 0058954

0 概述

为维护网络安全，互联网服务商通常会在网关处的防火墙利用网络审查技术来过滤网络流量中的不良信息，如端口过滤、深度包检测等技术。与此同时，一些恶意用户和组织往往会使用各种各样的方法手段规避网络审查。目前，用户主要使用安全代理、VPN加密、匿名网络3种方式突破防火墙的封锁，达到访问受限资源的目的。这些技术对防火墙的穿透会造成用户网络行为无法识别、不可监控的后果，如非法交易、传播不良信息等。因此，识别和分类这些技术软件产生的流量对网络管理和网络安全具有重要意义。

安全代理部署简单、成本较低，更容易在互联网用户之间传播，已经成为恶意用户突破防火墙的重要手段之一。目前Shadowsocks（SS）、V2Ray、Freegate、Ultrasurf这4种安全代理使用较为广泛[1]。SS和V2Ray是开源的安全代理软件，用户可轻易在

基金项目：国家重点研发计划“网络空间拟态防御技术机制研究”（2016YFB0800100）。

作者简介：高 平（1996—），男，硕士研究生，主研方向为网络安全；广 晖、陈 熹，讲师、硕士；李光松，副教授、博士。

收稿日期：2020-07-15 **修回日期：**2020-08-19 **E-mail：**yuntshark@163.com

虚拟专用服务器(Virtual Private Server,VPS)上部署。通过VPS对客户端流量进行代理,达到穿透防火墙的目的。Freegate和Ultrasurf安全代理软件有较大的受众基数,由专门的公司动态网和UltraReach对软件进行定期的更新和维护。安全代理通过加密用户的流量来规避防火墙的深度包检测技术。因此,安全代理流量分类本质上是加密流量分类问题。加密流量分类主要有基于端口的流量分类、基于深度包检测的流量分类和基于统计特征的机器学习算法流量分类3种方法。

基于端口的分类是早期传统的流量分类方法,使用Internet Assigned Numbers Authority (IANA)[2]来分配知名端口号识别流量类型。然而,对于使用非标准端口的协议和应用,这种监测技术则会失效。例如SS和V2Ray安全代理均可指定非标准端口,Freegate和Ultrasurf通过使用443端口伪装成TLS流量来规避防火墙的端口过滤。

基于深度包检测的分类方法融合了基于端口的分类方法。它不仅在端口过滤网络流量,而且还会检查流经网关的流量载荷来匹配已知协议或应用的关键字段。目前较为流行的深度包检测工具主要包括nDPI[3]、Libprotoident[4]、L7-fliter[5]、OpenDPI[6]4种。文献[7]对这些工具的性能进行了比较。然而,深度包检测技术能够识别的协议种类有限。由于SS和V2Ray通过双方预共享口令生成加密密钥,直接加密双方的通信内容,使得流量中分组有效载荷内容被混淆成随机字符串,深度包检测技术不能对其进行过滤和识别。

基于侧信道特征的机器学习算法分类通过提取网络流量中分组侧信道特征,如有效载荷长度、分组时间间隔、分组总数等,使用机器学习算法实现流量类型的区分。相比基于端口和深度包检测的方法,该方法具有协议无关性,即不依赖有效载荷中协议的具体实现,使用侧信道特征对流量进行分类。

文献[8]基于流中分组总数、有效载荷长度统计量(最大值、最小值、平均值、方差等)和分组时间间隔统计量等特征使用XGBoost算法对SS流量进行识别,但未将有效载荷长度直接作为分类特征,并且没有充分利用流中载荷长度的变化。文献[9]基于流中分组总数、同方向分组持续总时间、同方向分组长度总字节数等特征构成3 000维特征向量,使用随机森林(RF)算法区分SS流量。但使用的特征维度高,对于每一条流而言,并不一定具有多达3 000维特征,特征向量会有大量填充。文献[10]基于流中同方向分组长度总字节数和目的IP分布等特征使用随机森林(RF)等机器学习算法区分SS流量,但未考虑分组时间间隔特征。文献[11]基于流中分组长度平均值、分组总数、分组时间间隔等特征,使用XGBoost算法区分SS流量,但分组长度平均值并不能精确表征流量的特征变化。文献[12-13]基于分组长度的统计量等特征使用机器学习算法对SS安全代理流量进行区分。文献[14]基于载荷内容使用多种深度学习算法区分多种应用。但由于SS和V2Ray不进行协议的协商,载荷内容被加密算法加密并直接传输,因此使用载荷内容构造特征向量会对分类造成混淆。文献[15]基于分组长度和分组方向等特征,使用卷积神经网络(CNN)区分SS、V2Ray等流量。文献[16]基于分组长度特征识别Freegate流量,但由于Freegate版本的更新,该方法对更新后的Freegate流量并不适用。

相比于上述文献使用的特征,除使用流中分组总数、有效载荷长度的统计量、分组时间间隔特征外,流中单个分组的有效载荷长度和流量类型紧密相关,也可以被用来构建特征向量。本文将流中前n个有效载荷长度组合成有效载荷长度序列(Payload Length Sequence,PLS),来表征流量的载荷长度变化情况。从信号分析的角度出发,将流中每一个分组表征为一个信号,该信号由有效载荷长度、分组时间间隔和分组方向构成,从而可以将多个分组对应的信号进行合并构成信号序列,作为分类特征向量。本文基于提取的流量侧信道特征构建特征向量,使用机器学习算法、深度学习算法进行训练得到分类器,最终在测试集上使用分类器区分正常Web浏览和4种安全代理流量。本文使用的分类算法分别包括朴素贝叶斯(NB)、逻辑回归(LR)、支持向量机(SVM)、随机森林(RF)、XGBoost 5种机器学习算法和前馈神经网络多层感知机(MLP)、卷积神经网络(CNN)、长短期记忆(LSTM)网络3种深度学习算法。

1 预备知识

1.1 安全代理介绍

SS和V2Ray一般由Socks5协议代理本地Web流量。文献[17]将加密流量通信分为非加密初始化阶段和加密数据传输阶段。非加密初始化阶段协商双方使用密码算法、DH值、证书等信息,加密数据传输阶段完成数据的加密传输。但SS和V2Ray并没有遵循此规则,双方在发送数据之前未进行协商,而是直接基于双方预共享密钥生成加密密钥,对本地代理和服务器代理之间的数据进行加密封装,如图1所示。

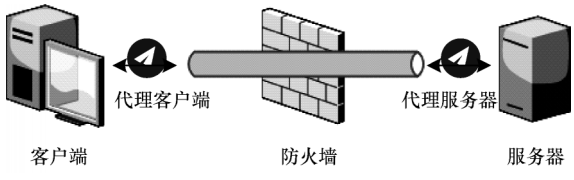

图1 SS和V2Ray通信流程

Fig.1 Communication procedure of SS and V2Ray

Freegate和Ultrasurf的工作原理与SS、V2Ray不同，不能直接与代理服务器传输数据。首先通过DNS解析获取服务器IP，之后与代理服务器进行协商并建立连接，建立连接后与代理服务器进行数据传输，如图2所示。

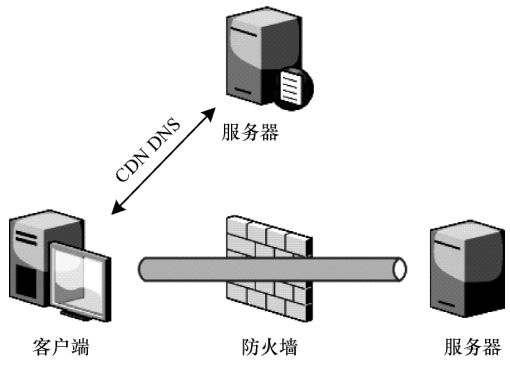

图2 Freegate和Ultrasurf通信流程

Fig.2 Communication procedure of Freegate and Ultrasurf

Freegate有A、F、M 3种通道，通过测试发现A通道和F通道已经被防火墙封锁，目前M通道可以正常使用。A通道使用内容分发网络（Content Delivery Network，CDN）隐藏代理服务器真实IP，使用TLS建立连接后使用UDP传输数据。与A通道不同，F通道使用TLS传输数据，但目前F通道不稳定。除使用CDN隐藏代理服务器真实IP地址外，M通道建立连接和数据传输均使用TLS进行流量伪装，稳定性较好。本文分析M通道协商时产生的TLS流量。

与Freegate的M通道相似，Ultrasurf首先通过CDN隐藏代理服务器真实IP地址，然后使用TLS建立连接并传输数据。

1.2 研究对象

本文以双向流（Bi-flow）为研究对象，设：主机A的IP为IP_A，端口为$Port_A$；主机B的IP为IP_B，端口为$Port_B$；传输层协议号为Proto。在同一会话中通信双方的IP地址、端口号以及传输层协议号不变并且一直持续到会话结束。从新的会话建立开始，称具有相同五元组$(IP_A, Port_A, IP_B, Port_B, Proto)$或者$(IP_B, Port_B, IP_A, Proto)$并持续到会话结束的分组集合为双向流：

$$f=\{h_i,p_i\}_i \tag{1}$$

其中：h_i是流中第i个分组的首部，包括以太网层、IP层、传输层的信息；p_i是第i个分组的应用层载荷。

2 基于侧信道特征的流量识别方法

由于深度包检测分类方法难以对加密流量有效载荷的关键字段进行匹配，对加密流量的分类性能较差，并且基于SS和V2Ray在通信时的特点，整个通信过程的数据均被加密算法封装，在无协议的非加密初始化阶段，使用有效载荷内容作为分类特征，容易造成混淆。因此，本文提取与有效载荷内容无关的流量侧信道特征，使用机器学习算法和深度学习算法构建分类器，对安全代理流量进行识别和分类。

2.1 主要框架

本文首先收集真实的Web浏览和安全代理流量，将流量存储在pcap或pcapng文件中。之后对流量数据进行预处理，将具有相同五元组的分组组合成双向流（Bi-flow）。提取双向流PLS序列、分组总数、分组时间间隔、载荷长度统计量、信号序列侧信道特征，构建特征向量。基于构建的特征向量，使用机器学习和深度学习算法得到流量分类器。最后使用这些分类器在测试集上进行标签预测，并评估不同分类器的性能。基于侧信道特征的流量分类框架如图3所示。本文处理流量和提取特征的工具已在Github上公开[18]。

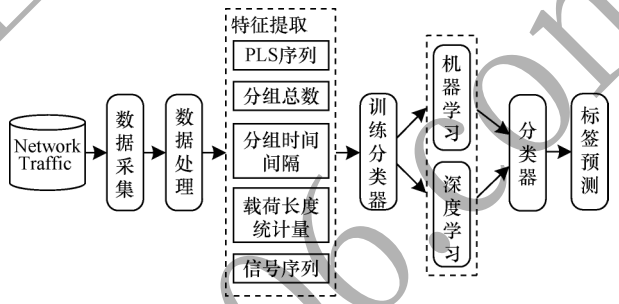

图3 基于侧信道特征的流量分类框架

Fig.3 Traffic classification framework based on side-channel feature

2.2 特征提取

文献[14,19]使用有效载荷内容作为特征向量进行流量分类，但考虑到SS和V2Ray的流量在通信时没有非加密初始化阶段，整个通信过程被加密算法封装，将有效载荷内容作为分类特征会降低分类器的性能。本文使用NIST检测标准说明了SS和V2Ray的有效载荷内容均被加密，难以进行区分。

NIST[20]随机性检测标准是美国国家标准与技术研究院（National Institute of Standards and Technology，NIST）提出用来检测二元比特序列随机性的方法集合，其中包含15种随机性检测方法。本文使用单比特频数检验、块内频数检验、最大1游程检验、近似熵检验、累加和检验5种检验方法对SS和V2Ray流量载荷内容进行分析，使用开源代码[21]计算检验值，其中块内频数检验、最大1游程检验选取的参数如表1所示，其他检验使用代码默认参数。

表1 NIST标准参数选取

Table 1 Selection of NIST standard parameters

参数	块内频数检验	最大1游程检验
分块大小M	20	8

抽取SS和V2Ray各50条流,通过截取流中前80 Byte有效载荷,使用NIST随机性检测标准对SS和V2Ray的通信数据随机性进行度量,表2给出了每种检验值的平均值。

表2 检验标准平均值

Table 2 Average of test standard

安全代理软件	单比特频数	块内频数	最大1游程	近似熵	累加和
SS	0.489 5	0.478 4	0.520 8	0.499 9	0.514 5
V2Ray	0.499 4	0.472 4	0.513 7	0.515 5	0.541 8

在NIST随机性检测标准中,将检验值与P值比较用来判定测试数据是否为随机产生的。当检验值大于P值时测试数据被认为是随机的。本文取P值为0.01,表示序列被认为是随机序列的置信度为99%。通过比较,SS和V2Ray流量均为加密流量,但是载荷的检验值接近,难以区分。因此,需要通过提取和载荷内容无关的并且有区分度的侧信道特征表征安全代理流量,进而对其进行识别和分类。

定义流f中第i个分组为pkt_i,其有效载荷为p_i,有效载荷长度为$l(p_i)$。对于流中前n个有效载荷长度不为0的分组可定义有效载荷长度序列(Payload Length Sequence,PLS):

$$P_{\mathrm{PLS}}=(l(p_{i_1}),l(p_{i_2}),\cdots,l(p_{i_n})) \tag{2}$$

其中:$l(p_{i_k})$表示流中第i_k个有效载荷长度不为0的分组,根据n的不同选取,PLS的长度也不同。

对于流中前n个有效载荷长度不为0的分组,定义第i_{k-1}个分组和第i_k个分组时间间隔为t_k,则分组时间间隔组成的序列如下:

$$T=(t_0,t_1,\cdots,t_{n-1}) \tag{3}$$

其中:t_0表示流中第1个有效载荷长度不为0的分组和TCP三次握手的最后1个分组的时间间隔。

使用有效载荷长度构建PLS序列,可以表征流量载荷长度的变化。为了从整体上对流量进行表征,流中有效载荷长度统计量包括最大值、最小值、平均值、方差、中位数、上四分位数、下四分位数以及分组总数特征被提取作为特征向量的一部分。

基于上述提取的侧信道特征,根据算法1构建机器学习算法使用的特征向量$\boldsymbol{F}$。

算法1 机器学习算法

输入 $f=\{h_i,p_i\}_i$,指定PLS长度为n(不包含重传分组),分组时间间隔序列T,分组总数pkt_cnt,有效载荷统计量S,变量k=0

输出 特征向量$\boldsymbol{F}$

```
1.For pkt_i in f do://处理流f中第i个分组
2. If pkt_i is restransmitted://判断第i个分组是否为重传
//分组
3.      continue
4. Else:
5.      If l(p_i)=0://判断第i个分组有效载荷长度是否为0
6.          continue
7.      Else:
8.          k++
9.          If k≤n:
10.             PLS.append(l(p_i))
11.             T.append(t_k-1)
12.         Else:Break
13.End For
14.pkt_cnt = |f|
15.计算流中有效载荷长度统计量S
16.F = (PLS, T, pkt_cnt, S)//得到特征向量
```

从信号处理的角度出发,流f可被看作信号源,流中的分组pkt_i可被看作i时刻发出的信号S_i。S_i由分组pkt_i的有效载荷长度、分组时间间隔、分组方向构成,定义如下:

$$S_i=(l(p_i),t_i,\mathrm{dir}(\mathrm{pkt}_i)) \tag{4}$$

其中:$t_1=0$,$\mathrm{dir}(\mathrm{pkt}_i)$定义如下:

$$\mathrm{dir}(\mathrm{pkt}_i)=\begin{cases}0,\text{客户端}\rightarrow\text{服务器}\\1,\text{服务器}\leftarrow\text{客户端}\end{cases} \tag{5}$$

给定要处理流中的分组个数n,对每一时刻信号S_i进行汇总,构成流的信号序列特征向量:

$$\boldsymbol{Sig}=(S_1,S_2,\cdots,S_n) \tag{6}$$

使用信号序列对网络流量进行表征,作为1D-CNN和LSTM深度学习算法的输入,对安全代理流量进行分类。

2.3 机器学习

本文使用朴素贝叶斯(NB)、逻辑回归(LR)、支持向量机(SVM)、随机森林(RF)、XGBoost 5种机器学习算法对安全代理流量进行分类,利用算法1得到的特征向量$\boldsymbol{F}$组成样本集合进行训练,并得到分类器。在测试集上,比较不同n值下的不同分类器性能,最终得到最佳分类器。

对于不同的n值,PLS长度不同,一般来说,选取流中的分组越多,分类器的性能越优越,当选取的分组总数达到一定数量时,分类器的性能会趋于稳定。

在训练和测试之前首先对数据进行Shuffle操作,打乱样本顺序,训练集和测试集比例为3:1。除了NB算法外,在训练阶段使用GridSearch算法进行参数搜索,并使用4折交叉验证。

2.4 深度学习

本文使用MLP、1D-CNN、LSTM 3种深度学习算法识别和区分安全代理流量。

MLP架构中的隐藏层如表3所示。MLP的输入特征向量为$\boldsymbol{F}=(\mathrm{PLS},T,\mathrm{pkt_cnt},S)$,其中PLS和$T$的长度为8。在每一层隐藏层后加入批量归一化(BN)层提高模型的收敛速度,并防止过拟合。

表3 MLP隐藏层结构

Table 3 Structure of MLP hidden layers

层数	每层结构
1	128个神经元+ReLu激活函数
2	BN层
3	64个神经元+ ReLu激活函数
4	BN层
5	32个神经元+ ReLu激活函数
6	BN层

CNN已经在机器视觉等多个领域内有了较成熟的应用。文献[14,19]基于流中分组有效载荷,使用2D-CNN深度学习算法对加密流量进行分类。根据上文分析,由于SS和V2Ray均使用加密算法对数据进行封装,其流量分组中的载荷均为随机字符串,将载荷的随机字符串加入到分类特征向量中,会降低分类器对SS和V2Ray的分类能力。除分组有效载荷内容特征外,可使用的特征包括有效载荷长度、分组时间间隔、分组方向等流量的测信道特征。对于2D-CNN算法而言,由于卷积核和池化操作均被用来对特征进行降维,使用侧信道特征构成的特征向量$\boldsymbol{F}$=(PLS, T, pkt_cnt, S)特征维度小,不适合使用2D-CNN进行训练。近年来,1D-CNN逐渐被用于语音识别领域[22-23]。将信号序列$\boldsymbol{S}_{\mathrm{Sig}}=(S_1,S_2,\cdots,S_n)$作为1D-CNN的输入,可避免特征维度小而难以进行训练的问题,其中序列的长度n=30。1D-CNN框架如图4所示。

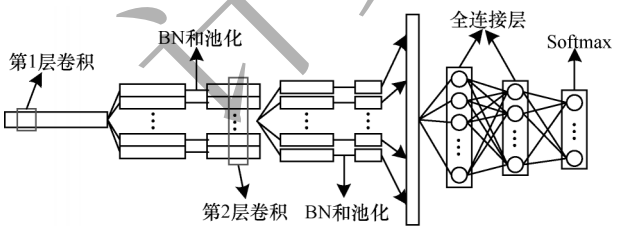

图4 1D-CNN整体架构

Fig.4 Overall architecture of 1D-CNN

第1次卷积后使用最大池化层,第2次卷积后使用均值池化层,每个卷积层和全连接层具体参数如表4和表5所示。

表4 卷积层具体参数

Table 4 Specific parameters of convolutional layers

卷积层	大小	卷积核总数	步长	激活函数
1	8	128	1	ReLU
2	4	64	1	ReLU

表5 全连接层具体参数

Table 5 Specific parameters of full connection layers

全连接层	神经元总数	激活函数
1	128	ReLU
2	64	ReLU

LSTM是专门用来处理时序特征数据的神经网络。将信号序列$\boldsymbol{S}_{\mathrm{Sig}}=(S_1,S_2,\cdots,S_n)$作为模型输入,对安全代理流量进行分类,其中$n$=30。其整体架构如图5所示,其中隐藏层状态为64,在LSTM隐藏层后加入了一层全连接层,其神经元个数为64,并使用ReLU激活函数。最终使用Softmax分类器达到分类的目的。LSTM整体架构如图5所示。

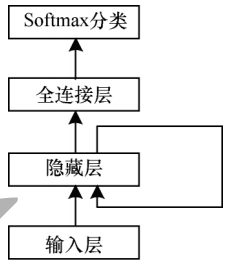

图5 LSTM整体架构

Fig.5 Overall architecture of LSTM

在训练和测试深度学习算法的分类器时,首先进行Shuffle操作打乱样本顺序,训练集和测试集比例为3∶1。3种算法的Batch_Size大小为64,并使用Adam优化算法。

3 实验与分析结果

3.1 数据集

通过Selenium[24]自动爬取ChinaZ[25]排名前840的网页,构造正常的Web浏览流量、SS流量、V2Ray流量。通过Bat脚本自动开启Freegate、Ultrasurf代理软件,构造客户端连接代理服务器的流量。所有的网络流量均存储于pcap或pcapng文件中,数据集总量为10.13 GB,包含109 237条流。各种类型流量在数据集中具体分布如表6所示。

表6 流量分布

Table 6 Flow distribution

软件名称	版本	操作系统	流总数
Web Browsing	Firefox/IE	Ubuntu 16.04 Windows 10	20 793
Shadowsocks	ss-Libev 3.3.3 ss-Windows 4.1.9.2	Ubuntu 16.04 Windows 10	19 141
V2Ray	4.21.3	Ubuntu16.04 Windows 10	24 744
Freegate	7.80	Windows 10	21 307
Ultrasurf	20.03	Windows 10	23 252

机器学习算法使用scikit-learn 0.22.2[26],深度学习算法使用Tensorflow 2.0[27]搭建。

3.2 特征分析

不同的安全代理具有不同的流量特征,从有效载荷长度角度来看,不同安全代理的有效载荷长度具有不同的分布,图6~图10分别列出正常Web浏览、SS、V2Ray、Freegate、Ultrasurf流中第1个分组有效载荷长度的分布。

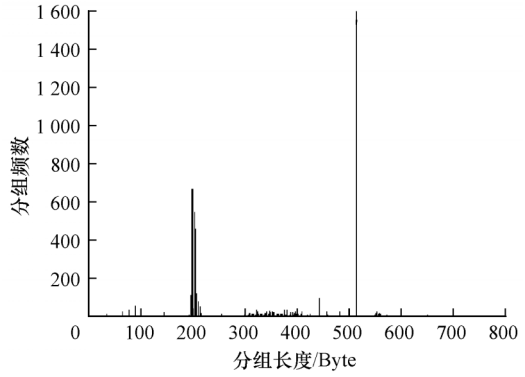

图6 Web载荷长度分布

Fig.6 Payload length distrubution of Web

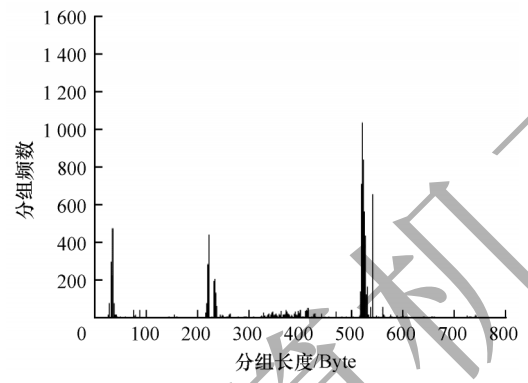

图7 SS载荷长度分布

Fig.7 Payload length distrubution of SS

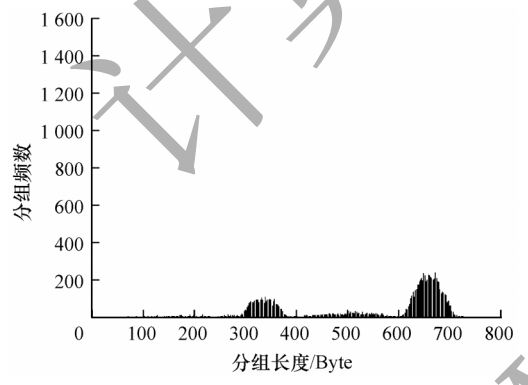

图8 V2Ray载荷长度分布

Fig.8 Payload length distrubution of V2Ray

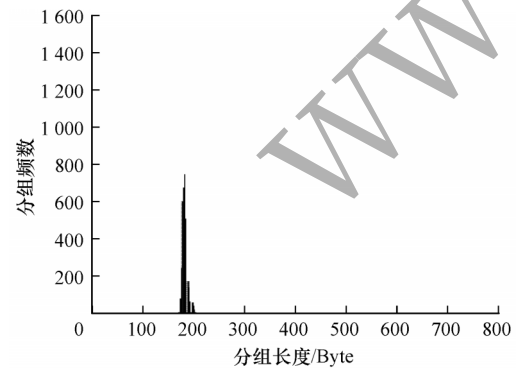

图9 Freegate载荷长度分布

Fig.9 Payload length distrubution of Freegate

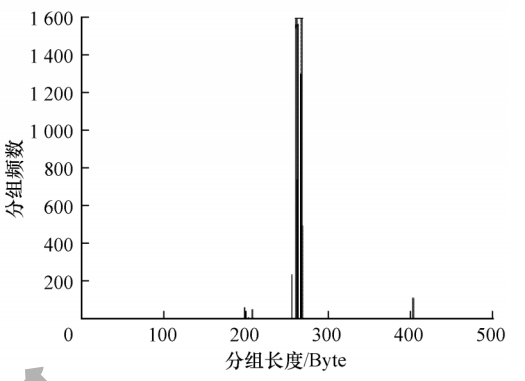

图10 Ultrasurf载荷长度分布

Fig.10 Payload length distrubution of Ultrasurf

正常的Web浏览流量和SS流量,除区间[50~100]外,对于其他区间的载荷长度分布,SS的有效载荷长度均大于正常Web浏览流量,可发生载荷分布平移的现象。这种现象对Ubuntu16.04下的ss-libev软件更为明显。通过结合公开的SS协议设计[28]和源码审计[29]发现,发生载荷长度平移现象是因为SS协议特殊的封装方式造成的。以ss-Libev为例,SS协议一般分为SS请求和SS数据传输,用户在访问网站时,首先会构造SS请求。SS请求使用如图11所示格式将需要访问的目的地址传递给远端服务器。

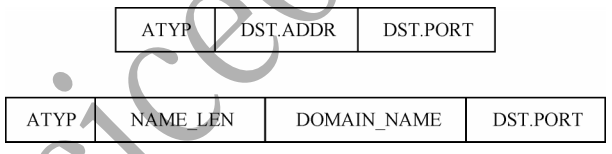

图11 SS请求格式

Fig.11 Request format of SS

在图11中,ATYP代表地址类型,一般为IP和域名2种地址类型。从客户端角度分析,如果访问的网站有其对应的IP地址,则使用IP地址请求。若没有,则会将域名填入SS请求消息中。客户端等待合适的时机将请求发送给远端服务器,让远端服务器进行地址解析。在发送之前,客户端会对SS请求进行加密,如图12所示,其中,灰色部分表示数据加密,PADDING是为了对齐加密算法密钥长度进行的数据填充。

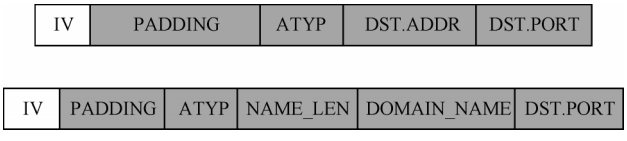

图12 加密后的SS请求

Fig.12 Encrypted request of SS

SS本地代理通过Socks5协议收到浏览器发来的数据，对数据加密封装后和SS请求一起发往SS服务器，如图13所示。

IV	PADDING	ATYP	DST.ADDR	DST.PORT
IV	PADDING	PAYLOAD		

IV	PADDING	ATYP	NAME_LEN	DOMAIN_NAME	DST.PORT
IV	PADDING	PAYLOAD			

图13 SS数据发送

Fig.13 Data transmission of SS

V2Ray安全代理使用VMess协议在客户端和服务器之间进行通信。V2Ray客户端消息格式如图14所示。CMD的具体格式如图15所示。

AUTH	CMD	PAYLOAD

图14 VMess客户端消息格式

Fig.14 Client message format of VMess

VER	IV	KEY	Other

图15 CMD具体格式

Fig.15 Specific format of CMD

在图15中，IV、KEY为16 Byte，Other字段包括选项、余量、加密方式、指令、端口、地址类型、地址、随机值、校验，至少18 Byte。其中余量字段指定了随机值字段的长度，随机值字段代表随机填充的数据。通过随机填充，V2Ray重塑了所封装流量的分组有效载荷长度分布。因此，V2Ray载荷长度也会产生平移现象。

由于Freegate和Ultrasurf的TLS流量中的密码套件和拓展字段与正常的TLS流量有差异，因此这些流量的有效载荷长度和正常的Web浏览具有一定的区分度。

3.3 实验结果

3.3.1 评价指标

为了评估分类器的性能，本文使用准确率、宏精确率、宏召回率以及宏F1值4个指标，最终给出模型分类结果的混淆矩阵。以上指标的计算公式如下：

$$A_{\text{Accuracy}}=\frac{\sum_{i=1}^{n}TP_i}{N} \tag{7}$$

$$m_{\text{macro-}P}=\frac{1}{n}\sum_{i=1}^{n}P_i \tag{8}$$

$$m_{\text{macro-}R}=\frac{1}{n}\sum_{i=1}^{n}R_i \tag{9}$$

$$m_{\text{macro-F1}}=\frac{2\times m_{\text{macro-}P}\times m_{\text{macro-}R}}{m_{\text{macro-}P}+m_{\text{macro-}R}} \tag{10}$$

其中：TP_i是每一类正确预测的样本；P_i是每一类别的精确率；R_i是每一类别的召回率；N是测试样本总数；n为类别总数。

3.3.2 深度包检测

本文运用4种开源深度包检测工具进行测试，检视这些工具对安全代理流量识别的能力。检测工具具体信息如表7所示。

表7 深度包检测工具

Table 7 Inspection tools of deep packet

DPI工具	版本	可识别的协议和应用数
nDPI	3.0	246
Libprotoident	2.0.12	500
L7 fliter	2020.05.22	127
OpenDPI	3.1.0	90

由于SS和V2Ray对流量的加密封装，这4种深度包检测工具均不能识别SS和V2Ray流量。Freegate和Ultrasurf均被识别为TLS流量。深度包检测工具主要是通过内置的签名库进行字段匹配，若没有相应的签名匹配则不能识别流量的种类，并且容易对Freegate和Ultrasurf的TLS伪装流量造成误报。因此，防火墙使用深度包检测方法不能有效过滤安全代理流量。

3.3.3 机器学习算法

NB、LR、SVM、RF以及XGBoost 5种机器学习算法以$\boldsymbol{F}=(\text{PLS},T,\text{pkt_cnt},S)$作为特征向量对4种安全代理流量进行识别和分类。在不同PLS长度下，图16和图17分别展示了5种算法准确率和F1值的变化。

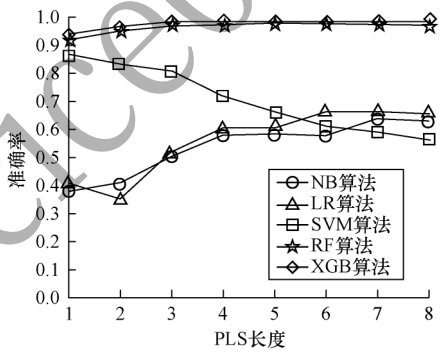

图16 机器学习算法准确率

Fig.16 Accuracy of machine learning algorithms

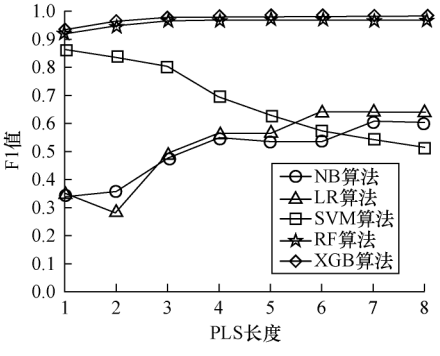

图17 机器学习算法F1值

Fig.17 F1-score of machine learning algorithms

从图16和图17可以看出:使用XGBoost和RF构建的分类器均能够获得良好的分类性能,并且XGBoost的准确率比RF平均高1.21%,F1值平均高1.2%。当PLS长度大于3时,XGBoost和RF的分类器性能趋于稳定。当PLS长度为8时,XGBoost的准确率为98.4%,F1值为98.36%。相比于NB、LR、SVM,使用属于集成学习的XGBoost和RF算法性能有明显提升。图18给出了PLS长度为8时,XGBoost的分类混淆矩阵。

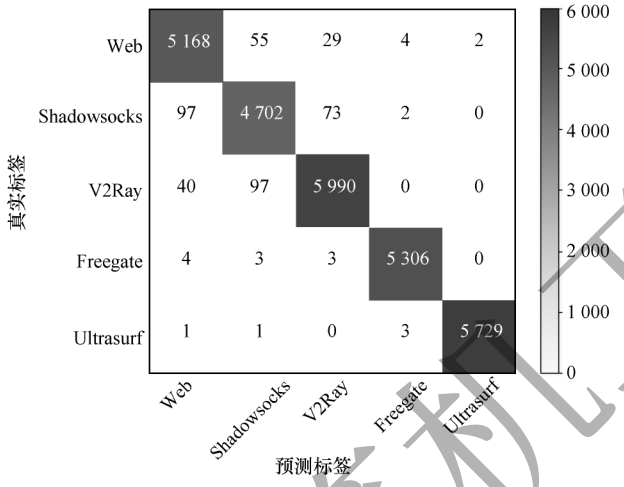

图18　XGBoost混淆矩阵

Fig.18　Confusion matrix of XGBoost

3.3.4　深度学习算法

前馈神经网络使用PLS长度为8时的特征向量$\boldsymbol{F}=(\mathrm{PLS},T,\mathrm{pkt_cnt},S)$作为输入。1D-CNN和LSTM使用构建的信号序列$\boldsymbol{S}_{\mathrm{Sig}}=(S_1,S_2,\cdots,S_{30})$分别对4种安全代理流量进行识别和区分。算法的准确率和损失随轮数(Epoch)增加的变化趋势分别如图19和图20所示。从图19和图20可以看出:与MLP和LSTM算法相比,1D-CNN算法在训练时的准确率和损失随着轮数的增加分别在平稳地增加和减少。最终相比分类器在测试集上的性能,1D-CNN算法的性能最优越。1D-CNN算法的分类混淆矩阵如图21所示。

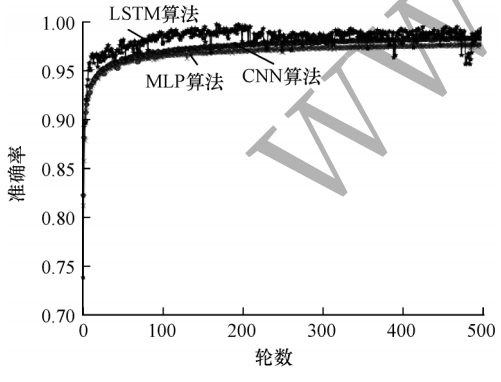

图19　准确率变化趋势

Fig.19　Change trend of accuracy

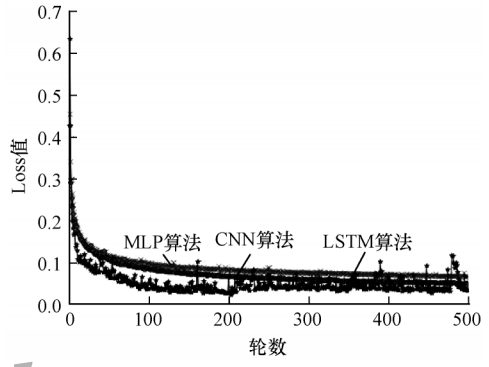

图20　损失变化趋势

Fig.20　Change trend of loss

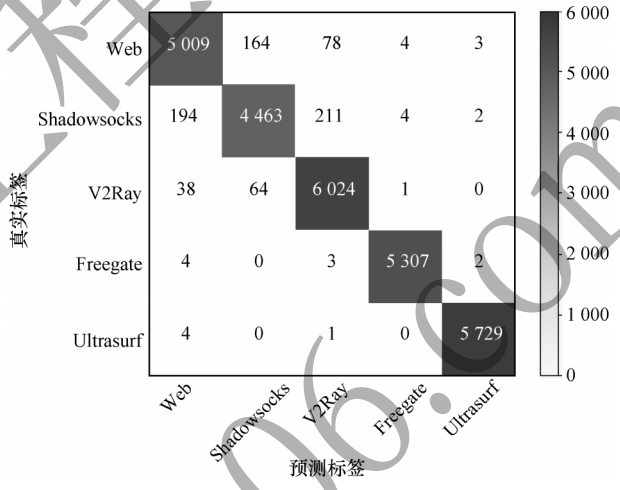

图21　1D-CNN混淆矩阵

Fig.21　Confusion matrix of 1D-CNN

3.3.5　结果对比

从分类器性能分析可以看出,在5种机器学习算法中,XGBoost算法性能最优越。在3种深度学习算法中,1D-CNN算法性能最优越。表8对比了RF、XGBoost、LSTM以及1D-CNN算法在测试集上的性能。

表8　各算法性能对比

Table 8　Performance comparison of each algorithm %

算法	准确率	精确率	召回率	F1值
RF	97.10	96.99	97.08	97.03
XGBoost	98.48	98.43	98.45	98.44
LSTM	97.02	96.94	96.82	96.88
1D-CNN	97.15	97.13	96.98	97.04

从表8分析可以看出:XGBoost算法的性能是最优的,与1D-CNN算法相比较,准确率、精确率、召回率和F1值分别提高了1.33、1.30、1.47、1.40个百分点。

文献[8-12]分别从不同的特征角度对SS流量进行区分,除使用载荷统计量、分组总数表征流量整体的特征外,本文提取了反映流量特征变化的PLS序列和信号序列,利用序列中特征的不同分布能够获得更好的性能。性能结果对比如表9所示。

表9 不同算法识别SS流量方法的性能对比

Table 9 Performance comparison of different algorithms for identifying SS traffic %

算法	精确率	召回率	F1值
文献[8]算法	—	95.14	—
文献[9]算法	85.87	93.28	89.42
文献[10]算法	93.74	90.66	92.17
文献[11]算法	94.63	93.28	93.95
文献[12]算法	—	96.00	—
本文算法	96.79	96.47	96.63

文献[15]运用2D-CNN对SS和V2Ray流量进行区分，当训练轮数达到5 000时，准确率达到93.3%；本文使用1D-CNN在训练轮数达到500时，准确率达到98.22%，计算开销更小。

4 结束语

本文根据流的不同侧信道统计特征，提出一种基于侧信道特征的安全代理流量分类方法。通过构建特征向量对安全代理流量进行表征，分别使用5种机器学习算法和3种深度学习算法识别和区分安全代理流量。实验结果表明，相比深度包检测方法，该方法使用机器学习和深度学习算法，避免了关键字段匹配技术的缺陷，提高了分类性能。在本文使用的分类算法中，XGBoost算法的性能最优越，与其他算法使用特征获得的分类器性能相比，本文提取与载荷内容无关的侧信道特征表征载荷长度变化，区分安全代理流量，达到较好的分类效果。下一步将对Lantern、Trojan、赛风等安全代理软件进行分类，加强防火墙的网络审查能力，以弥补深度包检测技术的不足。

参考文献

[1] CALLANAN C, DRIES-ZIEKENHEINER H. Leaping over the firewall: a review of censorship circumvention tools[EB/OL]. [2020-06-10]. https://freedomhouse.org/sites/default/files/inline_images/Censorship.pdf.

[2] IANA. DNS root, IP addressing, and Internet protocol resources[EB/OL]. [2020-07-09]. http://www.iana.org/.

[3] Open and Extensible LGPLv3 Deep Packet Inspection Library [EB/OL]. [2020-07-09]. https://www.ntop.org/products/deep-packet-inspection/ndpi/.

[4] Libprotoident [EB/OL]. [2020-07-09]. https://research.wand.net.nz/software/libprotoident.php.

[5] L7 filter [EB/OL]. [2020-07-09]. http://l7-filter.sourceforge.net/.

[6] OpenDPI [EB/OL]. [2020-07-09]. https://github.com/thomasbhatia/OpenDPI.

[7] BUJLOW T, CARELA-ESPANOL V, BARLET-ROS P. Independent comparison of popular DPI tools for traffic classification[J]. Computer Networks, 2015, 76: 75-89.

[8] 何杭松. 基于Xgboost算法的Shadowsocks流量识别研究[J]. 软件导刊, 2018, 17(12): 200-203.
HE H S. Research on Shadowsocks traffic identification based on Xgboost algorithm[J]. Software Guide, 2018, 17(12): 200-203. (in Chinese)

[9] DENG Z, LIU Z, CHEN Z, et al. The random forest based detection of Shadowsocks traffic[C]//Proceedings of the 9th IEEE International Conference on Intelligent Human-Machine Systems and Cybernetics. Washington D. C., USA: IEEE Press, 2017: 75-78.

[10] ZENG X, CHEN X, SHAO G, et al. Flow context and host behavior based Shadowsocks traffic identification[J]. IEEE Access, 2019, 7: 41017-41032.

[11] CHENG J X, LI Y, HUANG C, et al. ACER: detecting Shadowsocks server based on active probe technology[J]. Journal of Computer Virology and Hacking Techniques, 2020, 16(3): 217-227.

[12] HAN Z H, CHEN X S, ZENG X M, et al. Detecting proxy user based on communication behavior portrait[J]. The Computer Journal, 2019, 62(12): 1777-1792.

[13] ACETO G, CIUONZO D, MONTIERI A, et al. Multi-classification approaches for classifying mobile app traffic[J]. Journal of Network and Computer Applications, 2018, 103: 131-145.

[14] ACETO G, CIUONZO D, MONTIERI A, et al. Mobile encrypted traffic classification using deep learning: experimental evaluation, lessons learned, and challenges[J]. IEEE Transactions on Network and Service Management, 2019, 16(2): 445-458.

[15] ZHANG Y, CHEN J, CHEN K, et al. Network traffic identification of several open source secure proxy protocols[J]. International Journal of Network Management, 2019, 31(2): 209-220.

[16] ZHUO Z, ZHANG X, LI R, et al. A multi-granularity heuristic-combining approach for censorship circumvention activity identification[J]. Security and Communication Networks, 2016, 9(16): 3178-3189.

[17] VELAN P, ČERMAK M, ČELEDA P, et al. A survey of methods for encrypted traffic classification and analysis[J]. International Journal of Network Management, 2015, 25(5): 355-374.

[18] Pkt_Analyser Github [EB/OL]. [2020-06-10]. https://github.com/gpsvncl/Pkt_Analyser.

[19] 陈雪娇, 王攀, 俞家辉. 基于卷积神经网络的加密流量识别方法[J]. 南京邮电大学学报(自然科学版), 2018, 38(6): 36-41.
CHEN X J, WANG P, YU J H. CNN based encrypted traffic identification method[J]. Journal of Nanjing University of Posts and Telecommunications (Natural Science), 2018, 38(6): 36-41. (in Chinese)

[20] RUKHIN A, SOTO J, NECHVATAL J, et al. A statistical test suite for random and pseudorandom number generators for cryptographic applications[EB/OL]. [2020-06-10]. https://www.onacademic.com/detail.

[21] NIST sp 800_22_tests[EB/OL]. [2020-06-10]. https://github.com/dj-on-github/sp800_22_tests.

（下转第156页）

（上接第148页）

[22] OORD A V D, DIELEMAN S, ZEN H, et al. WaveNet: a generative model for raw audio[EB/OL]. [2020-06-10]. https://arxiv. org/abs/1609. 03499.

[23] AYTAR Y, VONDRICH C, TORRALBA A. SoundNet: learning sound representations from unlabeled video[C]// Proceedings of the 29th Conference on Neural Information Processing Systems. Washington D. C. , USA: IEEE Press, 2016: 892-900.

[24] Selenium[EB/OL]. [2020-06-10]. https://www. selenium. dev/.

[25] ChinaZ[EB/OL]. [2020-06-10]. https://www. chinaz. com/.

[26] Scikit-Learn [EB/OL]. [2020-06-10]. https://scikit-learn. org/stable/.

[27] Tensorflow[EB/OL]. [2020-06-10]. https://www. tensorflow. org/? hl=zh-cn.

[28] Shadowsocks[EB/OL]. [2020-06-10]. https://shadowsocks. org/en/spec/Protocol. html.

[29] Shadowsocks Github[EB/OL]. [2020-06-10]. https://github. com/shadowsocks.

编辑　索书志