计算机科学
COMPUTER SCIENCE

一种基于部分数据的多级剪枝Obfs4混淆流量识别方法

徐宸涵，黄河，孙玉娥，杜扬

引用本文

徐宸涵，黄河，孙玉娥，杜扬．一种基于部分数据的多级剪枝Obfs4混淆流量识别方法[J]．计算机科学，2024，51(4)：39-47.

XU Chenhan, HUANG He, SUN Yu'e, DU Yang. Multi-level Pruning Obfs4 Obfuscated Traffic Recognition Method Based on Partial Data [J]. Computer Science, 2024, 51(4): 39-47.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.231000118

一种基于部分数据的多级剪枝 Obfs4 混淆流量识别方法

徐宸涵[1]　黄　河[1]　孙玉娥[2]　杜　扬[1]
1 苏州大学计算机科学与技术学院　江苏 苏州 215006
2 苏州大学轨道交通学院　江苏 苏州 215131
(chenhan_xu@outlook.com)

摘　要　Obfs4 混淆流量是匿名通信网络 Tor 的一种承载流量,因其强匿名的特性而被滥用于非法网络活动,因此识别 Obfs4 混淆流量对预防利用 Tor 网络进行的网络犯罪具有重要作用。现有识别策略往往侧重于分析 Obfs4 流量特征,将完整流样本利用机器学习或深度学习技术进行精细化识别,但处于在线流识别的应用场景下时间开销偏高,且识别准确度在 Obfs4 应用间隔到达时间反检测技术(Inter-arrival Timing,IAT)后有所下降。为此,提出了一种基于部分数据的多级剪枝 Obfs4 混淆流量识别方法,仅收集每个流最先到达的少量数据包进行多轮快速过滤,并重点针对 IAT 模式特性设计识别方法,提升了 Obfs4 流量识别的效率和鲁棒性。该方法将识别过程分为握手阶段和加密通信阶段。在握手阶段,充分挖掘 Obfs4 握手数据包的隐含语义,进行随机性、时序和长度分布特征的粗粒度快速剪枝;在加密通信阶段,先对每个流的前若干数据包进行特征提取,并提高 IAT 相关特征的权重,最后利用 XGBoost 分类方法进行细粒度识别。实验结果表明,在包括了应用 IAT 技术的混淆流量的数据集上,使用流的前 30～50 个数据包能达到 99%的正确率和精确度,平均每条流的处理时间在毫秒级。

关键词:Obfs4;混淆流量识别;多级剪枝;间隔到达时间反检测;极致梯度提升
中图分类号　TP391

Multi-level Pruning Obfs4 Obfuscated Traffic Recognition Method Based on Partial Data

XU Chenhan[1],HUANG He[1],SUN Yu'e[2] and DU Yang[1]
1 School of Computer Science and Technology,Soochow University,Suzhou,Jiangsu 215006,China
2 School of Rail Transportation,Soochow University,Suzhou,Jiangsu 215131,China

Abstract　Obfs4 obfuscated traffic,carried by the anonymous communication network Tor,is often misused for illicit online activities due to its strong anonymity. Consequently,the identification of Obfs4 obfuscated traffic plays a critical role in preventing cybercrime via the Tor network. Existing methods tend to focus on the analysis of Obfs4 traffic features,utilize machine learning or deep learning techniques for the precise identification of entire flow samples. However,in the realm of flow recognition,it often results in considerable time overhead. Recognition accuracy also decreases notably with the incorporation of inter-arrival timing (IAT) technology in Obfs4. In response,a multi-level pruning method for Obfs4 obfuscated traffic recognition based on partial data is proposed. This approach involves collecting only a small number of initial packets from each flow for several rounds of rapid filtering,and is specifically designed to enhance the efficiency and reliability of Obfs4 traffic identification by focusing on the IAT pattern. The approach breaks down the process into two key phases:a handshake phase and an encrypted communication phase. During the handshake phase,it thoroughly explores the underlying meanings in Obfs4 handshake packets,enabling quick filtering based on broad characteristics like randomness,timing,and length distribution. In the encrypted communication phase,it extracts features from the first packets of each flow and places greater importance on features related to IAT. Finally,fine-grained identification is accomplished using the XGBoost classification method. Experimental findings indicate that despite the implementation of IAT technology,leveraging the initial 30～50 data packets from the flow yields a 99% accuracy rate,with an average processing time per flow measured in milliseconds.

Keywords　Obfs4,Obfuscated traffic recognition,Multi-level pruning,Inter-arrival time reverse detection,XGBoost

到稿日期:2023-10-18　返修日期:2023-11-27
基金项目:国家自然科学基金(62332013,62072322,U20A20182,62202322)
This work was supported by the National Natural Science Foundation of China(62332013,62072322,U20A20182,62202322).
通信作者:孙玉娥(sunye12@suda.edu.cn)

1 引言

匿名通信系统能隐藏网络通信双方的身份，保护数据链路上的数据包负载信息免受第三方监听和攻击，其规模和用户数量不断增长。当前最流行的匿名通信系统是第二代洋葱路由 Tor[1]，据官方信息统计网站 Tor Metrics 的数据显示[2]，截至 2023 年 9 月，有活跃中继 6000 多个、网桥 2000 多个，每日为近 60 万用户提供匿名上网服务。然而，Tor 本身被设计为去中心化、不受审查的网络工具，即使是开发者也无法从技术上监管和控制，间接导致大量暗网访问者利用 Tor 进行匿名化的非法网络活动，如贩卖毒品、军火、传播恶意软件等。互联网监管机构可以通过屏蔽潜在的 Tor 中继节点通信或异常用户的 Tor 连接尝试来阻止此类行为，其前提是精确、实时地从互联网背景流量中区分出 Tor 流量。因此，识别 Tor 流量是一项必要的网络监管任务，对打击网络犯罪具有重要的现实意义。

混淆流量技术应用于 Tor 后，成为 Tor 流量识别最主要的障碍。为提高流量匿名性，Tor 先后开发了包括 FTE，ScrambleSuit，Meek[3]，Obfsproxy[4-6] 和 Snowflake[7] 在内的可插拔混淆插件。Tor 网络每天建立的连接中，有 60% 的连接使用了 Obfs4 插件[2]，因此 Obfs4 混淆流量识别是 Tor 流量识别任务的主要目标。Obfs4 协议基于 TCP 协议，采用 Elligator2[8] 椭圆曲线加密、数据包随机填充和间隔到达时间等策略二次包装 Tor 数据包，并将它们组装成 Obfs4 流。在真实的大流量网络应用场景中，识别 Obfs4 流面临两个方面的挑战：1）Obfs4 混淆策略在一定程度上隐藏了 Obfs4 流的静态特征和动态特征，使其表现出与普通 TCP 流类似的行为；2）在保证正常提供网络通信服务的前提下，每条流能占用的计算时间和资源非常有限，这为高精度的实时 Obfs4 流量检测算法提出了更为严格的复杂度要求。

当前主要 Obfs4 流量识别策略可以划分为两种方法路线。大部分方法在机器学习的基础上做了一些适配 Obfs4 的改进[9-11]，例如，文献[9]拓展了相关特征选取角度，如网桥节点、主动连接方法、流量分析结果等，并将它们串行融合；文献[10]分析了 Obfs4 协议，确定了握手数据包的随机性和时序特征，然后将其按位重组以进行一次额外的粗过滤；文献[11]通过高斯混合模型引入混合因子计算数据流包长序列和时间间隔序列的二维概率密度分布描述流量指纹，然后使用隐马尔可夫模型将隐藏状态及其转移概率组合成完全连通的状态拓扑。这些方案最终都回归到使用传统机器学习方法进行识别，如支持向量机[9-10]、K-means[11] 等。然而，此类方法往往容易因为额外的特征提取或处理方法增加时间成本，延长样本平均识别周期不能确保检测的实时性；并且，识别准确度在 Obfs4 应用 IAT 反检测策略后有所下降。还有少量研究选择第二种思路，着力于寻找代表性强的关键序列，例如文献[12]提出将 Obfs4 传输模型根据通信双方的 IAT 模式分成 9 种场景，通过计算 Obfs4 流传输过程中产生的关键单元信号得到对识别最有帮助的 TCP 包序列。虽然在指定的一种场景下该段序列足够精炼有效，但是真实互联网环境并不能提前确定当前流属于哪种具体场景，进行有意义的识别通常需要对全部 9 种场景进行遍历检测，其结果表现为算法效率的显著衰减。

鉴于上述分析的方法和局限性，本文提出了一种基于部分数据的多级剪枝 Obfs4 混淆流量识别方法，对应用了 IAT 技术的 Obfs4 流也能完成精确识别，兼顾了算法开销和鲁棒性。首先，基于 Obfs4 握手阶段的随机性特征、时序特征和数据包长度分布特征进行多级粗粒度的快速过滤，对结论明确的非目标流量进行剪枝，省去后续算法步骤。随后，对目标流最先到达的部分数据包进行特征提取，使用 XGBoost 分类器进行精细化分类，在加密通信阶段开始的短时间内输出分类结果。多级剪枝思想在目标类型密度低的问题中对算法效率的提升尤为明显。针对 IAT 反检测技术，本文分析了不同 IAT 模式下握手阶段的共性和加密通信阶段的特性，在时序检测中测量客户端握手阶段静默时间，在特征工程中构建 IAT 相关特征并增大相应权重，提高算法鲁棒性。此外，为缓解数据集依赖问题，本文收集了应用不同 IAT 策略 Obfs4 目标流量的样本，以及未加密和多种类型加密的背景流量样本，并仔细调整了它们之间的比例，以获得更好的适应性。

本文的主要贡献在于：

1）提出了一种基于部分数据的多级剪枝 Obfs4 混淆流量识别策略，通过逐级剪枝消除冗余的算法步骤，降低算法时间开销，提升检测实时性。

2）提出了一种应对 IAT 技术的检测策略，将握手阶段数据包收发时间戳规律和加密通信阶段数据包长度分布规律相结合，提高了算法的识别精度和鲁棒性。

3）构建适应性更广的实验数据集，仔细调整了目标 Obfs4 流量和背景流量的构成和数量，优化了数据集依赖问题。

2 相关工作

本章首先简要介绍了 Obfs4 协议原理，然后整理了当前 Obfs4 流量识别的研究现状。

2.1 Obfs4 协议原理

Obfs4 协议基于 TCP 协议，目标是与未公开的 Obfs4 服务端节点建立安全、隐蔽的通信链路。Obfs4 可插拔混淆插件集成于 Tor 洋葱浏览器中，对 Tor 客户端产生的数据包明文进行二次加密，形成混淆流量在通信链路上传输，称为 Obfs4 流。Obfs4 协议将通信过程划分成握手和加密通信两个阶段。

TCP 连接建立后，立即进行无间隔的 Obfs4 握手，在 Obfs4 客户端与服务端之间进行 3 次数据交换[6]，最终根据椭圆曲线加密算法商定一个共享密钥。在每次数据交换时，客户端或服务端必须对接收数据包进行哈希校验，若校验失败则本次连接将被认为有风险，由一方主动放弃连接。此外，若出现长时间未收到确认，或连续接收到多个确认，这些不符合 3 次数据交换规则的异常活动都将导致握手失败。

握手认证完成后进入加密通信阶段，使用 Obfs4 协议规定的数据帧进行通信，使用握手阶段得到的共享密钥对数据包进行加密和解密。数据帧结构如图 1 所示，其中第 16 字节代表数据包的类型：0x00 表示有效载荷部分是正常通信的

应用数据，而 0x01 表示有效载荷部分是填充的随机数据。Obfs4 通过随机填充隐藏真正的统计特征，将 Tor 流量与普通 TCP 流量进行混淆。

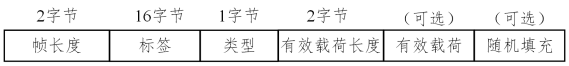


图 1 Obfs4 加密通信数据包结构

Fig. 1 Obfs4 encrypted communication packet structure

此外，Obfs4 支持 IAT 反检测技术，可自定义报文拆解模式，取值为 0，1，2，分别表示不主动拆分、主动拆分成固定的最大传输单元（Maximum Transmission Unit，MTU）大小和拆分成自定义可变大小。Obfs4 协议中 MTU 被定义为 1 448 字节。因此，一个长度大于 1 448 字节的握手数据包请求在 IAT 取值为 1 或 2 的情况下表现为同一方向上连续的多个报文请求，且发送时间间隔经过延迟等待。

2.2 研究现状

Tor 早期将提供服务的中继节点在互联网上公开，很容易被简单的 IP 黑名单方法屏蔽。随后 Tor 提供了非公开的中继节点，即网桥，研究人员转而使用 DPI 方法对 Tor 的 TLS 指纹进行匹配识别[13]。在 Tor 引入 Obfs4 混淆插件后，基于静态流量特征的 DPI 方法不再有效。

Obfs4 流量识别现有方案主要集中于两种思路：1）结合机器学习或深度学习方法，并实现一些适配 Obfs4 流的改进；2）挖掘关键 TCP 序列，建立不同场景识别模型。大多数解决方案基于机器学习方法，区别在于特征的选择和处理。据我们所知，最早的研究是由文献[14]提出的动态和静态特征相结合的 Obfs4 流量检测解决方案。该方案首先针对 Obfs4 抗静态特征设计随机性检测方法，利用 Obfs4 握手协议的时序特征与其他干扰协议区分。接着，进行大量数据的特征相关性和效度分析，排列组合提取出若干最相关的流特征并以支持向量机为分类器，但很难识别开启了 IAT 模式的 Obfs4 流。文献[9]将有助于识别 Obfs4 流量相关的数据称为源，不仅包括流的特征，也拓展到网桥节点特征、主动连接响应特征等。这些特征并非具有相同的格式，因此文章额外提供了一种串行融合加权计算方法，对它们进行统一处理，最后使用基于加权高斯核函数的支持向量机进行流识别。这种方法虽然提升了一定的算法精度，但多源特征处理流程花费了大量时间，在对实时性要求很高的在线流量识别问题中难以发挥实际作用。文献[11]通过高斯混合模型引入混合因子来计算包长序列和时间间隔序列的密度分布，然后使用隐马尔可夫模型将隐藏状态及其转移概率组合成完全联通的状态拓扑，最后使用 K-means 聚类算法进行分类。然而，该方法需要输入完整的包长和时间间隔序列作为初始特征，不仅等待输入的时间较长，处理数据的计算负载也较大，不适用于在线识别场景。少部分研究者认为，可以找到对流识别起到关键作用的数据包序列，从而降低识别算法的计算开销。例如，文献[12]将 Obfs4 传输模型根据协议分成 9 种典型场景，通过计算 Obfs4 通信过程中发生的关键单元信号得到关键 TCP 数据包流量序列的起始索引和窗口大小。仅使用关键 TCP 序列作为输入，可以在这 9 个场景中实现超过 90% 的准确率和召回率。然而，在未知目标流量具体属于哪种场景时，需要对所有场景进行遍历测试。最坏情况下在进行了全部场景的检测后判定为非目标 Obfs4 流量，此时仍然需要付出高昂的时间代价。

Obfs4 流在现实世界网络流量中样本分布稀疏。文献[15]提出的 XGBoost 模型在解决少样本数据分类问题上表现优异，也产生了在流量分类上的实践。例如，文献[16]使用 XGBoost 方法对 VPN 加密流量进行二分类；文献[17]采用 XGBoost 方法对互联网流量进行多分类，以改进 QoS 服务在多个类不平衡数据集上表现不佳的问题；文献[18]中，XGBoost 方法还被应用于 Tor 流量的识别。

上述有关工作给完善 Obfs4 流量识别任务的解决方案带来了启发。基于对 Obfs4 协议各阶段的特性分析并设计逐级剪枝方法，然后针对 IAT 模式设计特征工程并调整权重，最后将 XGBoost 分类方法引入 Obfs4 流量识别，有望进一步提升 Obfs4 流量识别的精度，同时降低时间开销。

3 问题模型

3.1 问题定义

Obfs4 混淆流量实时识别问题本质上是一个二分类问题。给定 TCP 数据包集合 $P:\{P_1,P_2,P_3,\cdots\}$，首先将其重构为一组流 $F:\{f_1,f_2,f_3,\cdots\}$，其中每个流 f 是一对 TCP 连接按时序所产生数据包的集合，包括发送数据包和接收数据包。规定客户端到服务端方向为流发送方向，源地址为 s、目的地址为 d、源端口为 sp，目的端口为 dp，则发送数据包表示为元组 (s,d,sp,dp)，接收数据包表示为元组 (d,s,dp,sp)。接着，对每个流 f 提取一个 m 维特征向量 $\boldsymbol{X}$。最终，构建一个分类器 $h(\boldsymbol{X})$，输入样本流的特征向量，输出对样本类别标签的预测，即 $h(\boldsymbol{X})\in\{0,1\}$。其中，0 表示非目标流量，1 表示目标流量。

3.2 设计目标

Obfs4 流量识别方法部署在互联网链路中的网络监控设备或网络流量管理设备上，在不影响正常网络通信的前提下识别目标 Obfs4 流。方案应该保证较高的准确率和鲁棒性，可以正确识别应用不同 IAT 模式的 Obfs4 流，减少误判和漏判；同时也不能消耗大量存储资源和时间资源，保证正常数据包的转发。根据上述目标设计的方案应该满足以下 3 条设计目标。

1）分类方法区分度高：用于判定目标流量和非目标流量的规则和逻辑应该基于两者明确的不同点，减少误判和歧义。在使用机器学习方法分类时，所选特征必须能准确反映目标流量不同于非目标流量的特点。

2）分类方法效率高：对流的处理不应该太复杂，否则会影响正常转发速度。因此，应该保证大部分特征明显的流使用简单的方法即可快速完成识别，给少部分易混淆的流使用复杂算法分类保留时间。

3）分类样本收集快：快速收集分类样本是机器学习方法中尽早开始识别的先决条件。因此，应该在流建立后在短时间内收集样本，及时识别，确保后续流量的处理实际有效。

4 基于部分数据多级剪枝的实时 Obfs4 混淆流量识别方法

本章详细介绍了基于 XGBoost 的实时 Obfs4 混淆流量识别方法。首先总览了识别策略，展示了完整的工作流；然后针对 Obfs4 通信的两个主要阶段所具有的不同特征，分别进行方法可行性的讨论。

4.1 系统工作流

本文提出了一种基于部分数据的多级剪枝的 Obfs4 混淆流量识别方法，主要通过握手阶段过滤和加密通信阶段过滤来完成对目标流量的精确识别。

当部署了识别程序的网络设备检测到新到达的 TCP 流，流的数据包特征序列被依次保存以执行握手阶段的多级剪枝，包括随机性检测、时序检测和数据包长度分布检测 3 个部分。

通过检测的流继续执行数据包特征序列收集，未通过检测的流被剪枝，不再执行后续检测步骤并从内存中移除。当某条流的数据包序列收集达到预设的阈值，则进入加密通信过滤阶段。首先根据所收集的流数据构建特征工程，接着通过训练好的 XGBoost 模型进行分类预测。详细的系统工作流如图 2 所示。

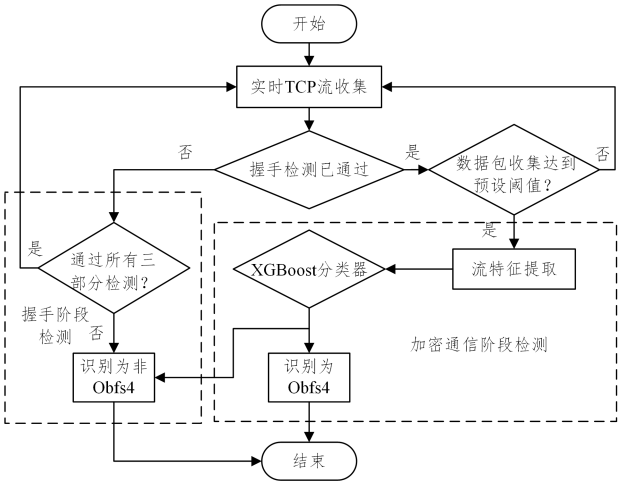


图 2 系统工作流

Fig. 2 System workflow

4.2 握手阶段过滤方法

根据 Obfs4 协议，TCP 连接建立后应当无间隔地开始 Obfs4 客户端与服务端的握手。本节将分析可以用于明确区别 Obfs4 和背景流量的协议内容，然后基于这些内容提出过滤方法。

4.2.1 随机性检测

根据 Obfs4 协议，握手报文的前 32 字节是经过 Elligator2 加密的公钥 X，第 32 字节起是长度在 85～8128 字节之间的随机填充序列，其余部分则由 HMAC-SHA256-128 算法计算出的校验序列组成。随机填充序列让负载的 0，1 分布更加平均，目的是隐藏指纹特征和统计分布属性。然而，随机性作为必要条件无法用于直接断定 Obfs4 流，但可以对负载随机性程度较低的流进行剪枝。因此，握手有效载荷的随机性可以被用于过滤 Obfs4 流。

序列随机性检测问题有许多现行标准，本文采用国家标准 GB/T 32915-2016 中的块内频数检测方法作为 Obfs4 有效载荷随机性的检测算法[19]。对于待测数据包有效载荷 0，1 比特序列，先将其划分成若干等长不重叠的子序列，然后检查任意子序列中 1 的个数是否接近子序列长度的一半，最后使用统计方法得到有效载荷的随机性。检查每个握手负载的随机性，需要对最大 8128 字节的 0，1 比特序列进行统计，造成了较大的时间开销。因此，本文对大负载进行固定间隔的采样，抽取其中的一部分字节并重新组合，以进行块内频数的检测。子序列长度按照一般共识设置为 20。握手有效载荷块内频数检测过程的伪代码如算法 1 所示。

算法 1 随机性检测算法

输入：数据包待测负载序列 ε，长度 l，采样率 p

输出：随机性检测识别结果 true 或 false

1. 序列 ε 每 $\frac{1}{p}$ 位取 1 位，重组为 ε_0，长度为 l_0
2. 初始化等长子序列的长度 m
3. if $l_0 < 20$ then
4. $\quad m = l_0$
5. else
6. $\quad m = 20$
7. end if
8. /* 初始化数组 X，计算子序列中 1 的频率 */
9. X=[]
10. for i ← 1 to $\left\lfloor \frac{l_0}{m} \right\rfloor$
11. $\quad X[i] = \frac{1}{m}\sum_{j=1}^{m} \varepsilon_{0(i-1)m+j}$
12. end for
13. /* 计算统计量的值 V */
14. $V = 4m\sum_{i=1}^{\left\lfloor \frac{l_0}{m} \right\rfloor}\left(X[i]-\frac{1}{2}\right)^2$
15. /* 计算 p_value 的值 */
16. $p\_value = igamc\left(\frac{\left\lfloor \frac{l_0}{m} \right\rfloor}{2}, \frac{V}{2}\right)$
17. if p_value $\geqslant \alpha$ then
18. $\quad$ return true
19. else
20. $\quad$ return false
21. end if

算法 1 中，$igamc()$ 是不完全伽马函数，α 是显著性水平参数，通常取值为 0.001～0.01。如果 $p\_value \geqslant \alpha$，则可以判断该序列通过随机性检测。

4.2.2 时序检测

根据 Obfs4 协议，Obfs4 握手认证在建立 TCP 连接后立即执行，且过程中不会传输任何其他类型的通信数据包，严格遵守 3 次通信的时序。任意服务端或客户端每次对另一方的请求或响应进行验证，一旦验证失败就立即放弃连接。同时，还应该考虑不同 IAT 模式下 Obfs4 流的表现，这在先前的研究中常被忽略。基于上述考虑，结合不同 IAT 模式的共性表现，本文将 Obfs4 时序检测规则分为 3 项。

1）握手请求必须由客户端发出：握手阶段第一个数据包一定是由客户端发往服务端的。

2）同一方向连续数据包的长度和不得超过协议规定：

握手请求可能按照不同的 IAT 模式进行划分，但是连续同方向数据包的负载长度之和不应该超过协议规定的最大长度。

3）客户端方向存在明显的长时间静默：根据 Obfs4 协议，服务端在收到完整的客户端请求并校验前不能发送任何回应，客户端在没有得到服务端请求并校验前也不能发送下一步请求。因此，在客户端发送第一次握手的请求到客户端收到服务端传来的回应并校验的这段时间内，客户端没有任何请求操作。无论使用何种 IAT 模式，该段静默时间长度将数倍于正常同向数据包传输时间间隔长度。

具体方法如算法 2 所示。

算法 2 时序检测算法

输入：流建立 TCP 连接后按照时序无间隔收集的数据包集合{P_1，P_2，P_3，…}，当前流方向 d

输出：时序检测识别结果 true 或 false

1. 初始化当前流方向 d 为 TCP 第一次握手方向
2. /* Obfs4 握手在 TCP 连接建立后立即进行，第一次握手传输方向与 TCP 第一次握手方向一致 */
3. if P_1. direction ≠ d then
4. return false
5. end if
6. 计算第一次握手最大连续数据包数量 n_1，负载长度和 L_1
7. if L_1 > 8128 then
8. return false
9. end if
10. /* 计算第一次握手平均发送间隔 t */
11. $t = \frac{1}{n_1}(P_{n1}.\text{sendtime} - P_1.\text{sendtime})$
12. 计算第二次握手最大连续数据包负载长度和 L_2
13. if L_2 > 8 096 then
14. return false
15. end if
16. 计算第三次握手最大连续数据包负载长度和 L_3
17. if L_3 > 8128 then
18. return false
19. end if
20. 计算客户端方向最长发送时间间隔 tmax，并初始化间隔时间阈值参数 k
21. if tmax < kt then
22. return false
23. end if
24. return true

间隔时间阈值参数 k 用于描述最长发送时间间隔是否显著长于连续发送时间间隔，需要实验测定。由协议可知携带最大 8128 字节负载的数据包至少需要被划分为 6 个不大于 1 448 字节的数据包，因此 k 的正常取值约为 0～6。k 的值不应该太大，否则容易剪枝目标流量。与随机性检测一样，时序检测算法理论上能够以较高的召回率过滤掉一些非 Obfs4 流，但是不能保证通过时序检测的就是 Obfs4 流。

4.2.3 数据包长度分布检测

根据 Obfs4 协议，Obfs4 客户端握手数据包填充长度为 85～8128 字节，服务端握手数据包填充长度为 45～8096 字节，加上握手数据包固定的 64 字节公钥和校验部分，可以得到 Obfs4 两个方向上握手数据包的最小值分别为 149 字节和 109 字节。因此，可以根据该规则设置最小过滤阈值，将数据包负载长度小于该值的流进行剪枝。在现实互联网流量环境中，占比较高的小负载的 TCP 数据包通常起到探测或保持连接活跃的作用。为满足实时性需求，可以适当提高最小过滤阈值，从而剪枝更多非目标小流，降低误报率，但这将在一定程度上牺牲召回率，需要通过实验来确定合理的最小过滤阈值。

4.3 加密通信阶段识别方法

Obfs4 流在加密通信阶段缺乏用于直接识别的显式特征，需要借助机器学习等分类问题解决方法。结合当前流量识别分类场景和数据集特点，本文选取 XGBoost 分类模型进行精细化识别。

对加密通信阶段 Obfs4 流量的特征选取需要满足真实准确和区分度高两个要求。分析 Obfs4 协议原理可知，随机负载的添加使得 Obfs4 数据包具有显著的高信息熵特征。然而，各种加密流量大多都具有类似的高熵特征，并且通过了握手负载随机性检测的数据流检测出高熵的概率很大。因此，信息熵可以用于区分普通流量和加密流量，但在加密流量之间的分类效果可能并不明显。Obfs4 数据包的不确定随机负载填充长度与当前网络状态有关，对于任何网络流来说，数据包个数特征、长度特征等都是最基本的特征，可以作为特征选取的参考。

此外，考虑到 IAT 技术对数据包进行分割会产生连续的收发序列，必须将此部分作为 Obfs4 流特征的重要参考，赋予其更高的权重，包括连续收发个数、连续收发时间等。最后，本文选择了 23 个特征用于加密通信阶段的 Obfs4 流量识别，详情如表 1 所列。

表 1 特征选取及归一化

Table 1 Feature selection and normalization

特征类型	数据包特征描述	特征归一化
长度特征	总长度	最小-最大归一化
	发送总长度	最小-最大归一化
	接收总长度	最小-最大归一化
个数特征	发送总个数	发送总个数/总数
	接收总个数	接收总个数/总数
	携带有效负载的发送总个数	携带有效负载的发送总个数/总数
	携带有效负载的接收总个数	携带有效负载的接收总个数/总数
方差特征	长度方差	最小-最大归一化
	发送长度方差	最小-最大归一化
	接收长度方差	最小-最大归一化
	发送有效负载长度方差	最小-最大归一化
	接收有效负载长度方差	最小-最大归一化
长度熵特征	长度信息熵	最小-最大归一化
	发送长度信息熵	最小-最大归一化
	接收长度信息熵	最小-最大归一化
连续收发序列特征	平均发送时间间隔	最小-最大归一化
	平均接收时间间隔	最小-最大归一化
	最大连续发送个数	×2
	最大连续接收个数	×2
	连续发送个数方差	最小-最大归一化
	连续接收个数方差	最小-最大归一化
	最大连续发送时间	×5
	最大连续接收时间	×5

对于所选特征，除了需要增加权重的部分，基本采用统一的归一化方法：最小-最大归一化方法，用 x_0 表示原始特征

值，用 $x_{\min}$ 和 $x_{\max}$ 表示该特征的最小、最大取值，则经过归一化的特征值表示为 $\frac{x_0-x_{\min}}{x_{\max}-x_{\min}}$。在接下来的实验中，还将根据特征重要性排名进一步选择最有效的特征。

5 实验分析

本章对所提出的基于部分数据的多级剪枝 Obfs4 混淆流量识别方法进行实验评估。首先，提出实验数据集的构造方法；其次，介绍性能评价指标；最后，分别分析并展示了两个阶段的实验结果。

5.1 数据集

实验所使用的数据集包含 Obfs4 混淆流量和背景流量两大类数据流，具体组成与占比参考了现实互联网流量。两阶段实验共准备了 5978 个 Obfs4 流量用例和 46074 个互联网背景流量用例。由于两阶段的识别任务目标不同，所采用数据会根据任务进行相应的调整。

采取以下步骤收集 Obfs4 混淆流量：首先，下载最新的 Tor 客户端，并向官方提供的邮箱发送邮件，获得了 4 个可用的部署了 Obfs4 服务端的网桥。接着，使用 Obfs4 插件与这 4 个网桥建立 Tor 连接。由于部分数据识别方法只需要对流的前若干个数据包进行学习，因此无须关心后续使用 Tor 承载的服务类型（如网页浏览、音频、视频等）。综上所述，流量收集策略是重复进行“建立连接、收集数据、中断连接”的步骤。考虑到 Obfs4 的反检测策略，修改 IAT 的值为 0，1，2，让 4 个网桥每天在同一时段分别运行 2h，以便得到不同的 IAT 策略样本。

为了构建有代表性的互联网背景流量数据集，除了普通 TCP 流量外，数据集还选取了可能对目标流量识别带来干扰的其他类型的加密流量，例如 VPN 加密通信流量，包括从 ISCX Tor 2016 Tor-nonTor 数据集[1)] 中的 nonTor 部分。该部分是由 Draper-Gil 等生成的非 Tor 加密流量公开数据集，其中包含网页流量、视频等 8 种类型的加密流量。对于普通 TCP 流量，一部分来自 USTC-TFC 2016 数据集[2)]，该数据集包含非加密的 TCP 流量，涵盖了各种服务，另一部分是实验室收集的 TCP 流量。数据集的组成如表 2 所列。

表 2 数据集组成

Table 2 Dataset composition

数据类型/承载服务		数量
Obfs4 混淆流量	IAT=0	2000
	IAT=1	2000
	IAT=2	2000
普通 TCP 流量	网页浏览	15500
	聊天	400
	音频	4500
	视频	17800
	邮件	400
	FTP	4500
加密流量	网页浏览	1200
	音频	400
	视频	400
	VOIP	200
	FTP	800

1) https://www.unb.ca/cic/datasets/tor.html

2) https://github.com/yungshenglu/USTC-TK2016

5.2 评价指标

本文将召回率（Recall）、精度（Precision）、准确率（Accuracy）、F1 值和单位流量 CPU 时间作为评估标准。召回率、精度、准确率和 F1 值是二分类问题中常用的评价指标，计算式如下：

$$Recall=\frac{TP}{TP+FN} \tag{1}$$

$$Precision=\frac{TP}{TP+FP} \tag{2}$$

$$Accuracy=\frac{TP+TN}{TP+FP+TN+FN} \tag{3}$$

$$F1=\frac{2Precision\times Recall}{Precision+Recall} \tag{4}$$

其中，TP 表示目标流量且正确分类的数目；TN 表示非目标流量且正确分类的数目；FP 表示非目标流量但错误分类为目标流量的数目；FN 表示目标流量但错误分类为非目标流量的数目。CPU 时间是中央处理器单元用于处理计算机程序指令所需的时间量，通过计算单位流样本识别消耗 CPU 时间可反映解决方案的时间开销。单位流量 CPU 时间消耗越短，流量识别的实时性越高。在握手过滤实验中，本文还使用真阴性率 TN 记录被判定为非 Obfs4 的样本占实际为非 Obfs4 的比例，以表示握手过滤实验的剪枝能力。

本文将对两次实验的综合结果与最近发表的 4 种识别方法进行比较，以验证方案的可行性。方法 1 由文献[20]提出，采用加权特征 KNN 方法（WKNN）对加密网络流进行精细分类；方法 2 由文献[21]提出，使用增量支持向量机（ISVM）对互联网流量进行分类；方法 3 由文献[22]提出，使用随机森林方法（RF）对 Tor 混淆插件进行识别；方法 4 由文献[23]提出，通过改进的卷积神经网络方法（CNN）对加密流量进行识别。本实验使用 3.0 GHz AMD Ryzen 5 4600H 处理器上的 CPU 时间作为实验标准，并在运行时将程序绑定到单独核心，以减少资源占用导致的误差；使用 scikit-learn 版本为 1.1.1，xgboost 版本为 1.5.1，scapy 版本为 2.4.3。

5.3 结果分析

5.3.1 握手阶段过滤实验

握手阶段过滤实验的目标是通过随机性检测、时序检测和数据包长度分布检测对不符合 Obfs4 特征的流量进行多级剪枝，并确定合适的间隔时间阈值参数。实验使用包括全部 3 种 IAT 模式在内的 3978 个 Obfs4 流和 4023 个背景流，两者比例相近，防止背景流数量远大于目标流数量导致参数对实验结果的影响不明显。实验时依次检查数据包，对接下来可能表示 Obfs4 握手阶段的数据包执行随机性检测、时序检测和数据包长度分布检测，然后提取并暂存通过检测的流前 100 个数据包的特征序列。对于随机性检测，抽取数据包中一定百分比数量的字节，然后重新组合子序列，进行块内频数检测。实验测试了不同采样率对识别率的影响，并将显著性水平参数 α 设置为 0.01。对于时序检测，根据实际情况考虑

了 Obfs4 不同 IAT 模式下对大负载数据包进行分割的策略，首先保证不同的 IAT 模式下的召回率。对于数据包长度分布检测，测试了最小过滤长度的阈值对精度的影响。对于间隔时间阈值参数的选取，分别对不同 IAT 模式进行了对比分析。

实验首先将最小过滤长度阈值设定为 Obfs4 协议规定的最小握手数据包长度 149 字节，将间隔时间阈值参数设定为较为保守的 2，以研究采样率对识别率和 CPU 时间的影响，结果如图 3 所示。

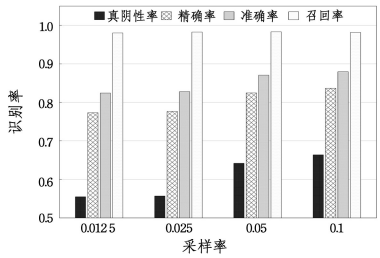


图 3 采样率对识别率的影响

Fig. 3 Effect of sampling rate on recognition rate

图 3 显示，召回率、准确率、精确率和真阴性率随着采样率的提高而提高，符合一般经验。同时，注意到采样率为 0.05 和 0.1 时，各项评价指标之间的差距已经很小。接着，逐步增加最小过滤长度阈值，尽管这与 Obfs4 协议相违背，可能会牺牲一部分召回率，但可以极大地提高真阴性率，从而减少通信阶段仍需检测的流数量，如图 4 所示。

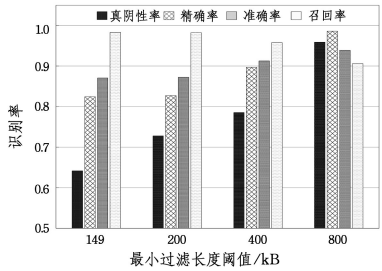


图 4 最小过滤长度阈值对识别率的影响

Fig. 4 Effect of minimum filter length threshold on recognition rate

召回率随着最小过滤长度阈值的增大而降低，同时其他指标有所提升，符合实验预期。在最小过滤长度为 200 字节时，基本保持了 149 字节的指标；当达到 400 字节时，召回率降低至 0.95，真阴性率提高至 0.78；当达到 800 字节时，虽然真阴性率达到 0.95，但召回率偏低，只有 0.90。在 200 字节和 400 字节的效果对比中，前者是更合理的选择，因为必须保证正确地剪枝太多目标 Obfs4 流，下一阶段可以继续进行精细化识别。

图 5 给出了采样率和最小过滤长度对 CPU 时间的影响，每个流的处理时间在毫秒级，在采样率为 0.05、最小过滤长度为 800 时达到最低。为了保证实验的实际意义，提取通过检测流的后续数据包特征序列的时间也被计算在 CPU 时间内，因此尽管较低的采样率可以加快随机检测的速度，但会导致更多的非目标流没能被过滤，从而消耗更多的时间进行特征提取。CPU 时间随着最小过滤长度阈值的增大而减少，这是符合一般逻辑的，因为更多的流将在数据包长度分布检测时被剪枝；但出于识别精度的考虑，本文还是谨慎地选择了 200 字节。

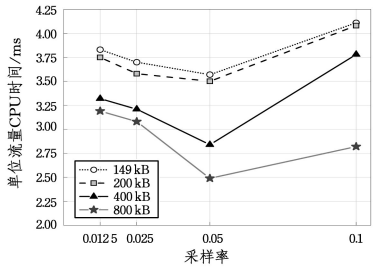


图 5 采样率和最小过滤长度对 CPU 时间的影响

Fig. 5 Effect of sampling rate and minimum filter length threshold on CPU time

使用上述实验得到的最优参数，重新选取等量的 3 种 IAT 模式下的 Obfs4 数据流与等量背景流组成新的实验数据集，测试不同间隔时间阈值参数对识别率的影响。图 6 和图 7 给出了不同 IAT 模式下设置间隔时间阈值参数得到的识别率。IAT 模式为 0 的 Obfs4 流不进行数据包分割，表现出的时间间隔较短，故较小的参数取值下识别率更高。对于 IAT 模式为 1 和 2 的流，数据包分割导致的时间间隔分布较为均匀，由于最大 8128 字节的负载需要最多分成 6 个部分进行传输，因此参数取值为 3～4 时相对而言有最好的准确率和召回率。仍以保证召回率为基本准则，执行保守的识别策略，将间隔时间阈值参数设置为 3，确保在少剪枝目标流的前提下过滤非目标流量。

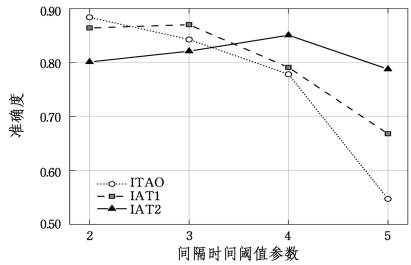


图 6 不同 IAT 模式下间隔时间阈值参数对准确率的影响

Fig. 6 Effect of interval time threshold parameters on accuracy rate in different IAT modes

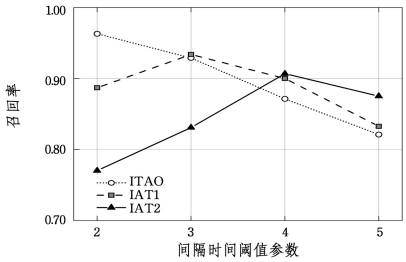


图 7 不同 IAT 模式下间隔时间阈值参数对召回率的影响

Fig. 7 Effect of interval time threshold parameters on recall rate in different IAT modes

5.3.2 通信阶段过滤实验

通信阶段过滤实验采用 XGBoost 方法对输入流样本进行精确识别，使用的数据是通过握手阶段过滤实验并被预测为 Obfs4 的流，包括 4 551 个 Obfs4 流和 12 980 个非 Obfs4

流，其中的非 Obfs4 流在握手过滤阶段被误标记为阳性，实际上应该被标记为阴性。实验将 30%的样本用于学习，将 70%的样本用于测试。XGBoost 的最大树深度设置为 3，学习率设置为 0.01，树的数量设置为 50，并选择对数损失函数作为损失函数。学习时发现，并不是所有特征与目标都非常相关，因此将特征被选中作为 XGBoost 分裂特征的次数作为评判标准，将平均次数小于 10 次的特征去除，留下 17 个有效特征，去除了总个数、发送总个数、接收总个数、数据包长度方差、发送数据包长度方差、接收数据包长度方差共 6 个特征。而对于次数大于 200 的最大连续收发次数和最大连续收发时间，则增大初始权重。实验时分别对除去建立 TCP 连接的 3 个数据包之外的前 10，20，30，50 和 100 个数据包抽取上文提到的 17 个特征。在上述实验设置下，进行近似最优的 XGBoost 训练，并在 70%的样本上进行测试，最后得到了表 3 所列的指标。

表 3 选择数据包个数对识别率的影响

Table 3 Effect of the packets number on recognition rate

选择数据包个数	精确度	召回率	准确度	F1 值
10	0.9873	0.9713	0.9751	0.9732
20	0.9888	0.9801	0.9729	0.9765
30	0.9955	0.9961	0.9852	0.9906
50	0.9957	0.9953	0.9866	0.9910
100	0.9979	0.9971	0.9943	0.9957

随着所选数据包个数的逐渐增加，精确度、召回率、准确度和 F1 值基本都有所提高。选择前 30 个数据包，可以得到 0.99 的召回率、0.99 的精确度和 0.98 的准确度。再增加数据包个数，性能提升并不明显，因此通过前 30～50 个数据包进行识别是最佳选择。需要注意的是，最终的评价指标必须结合握手阶段过滤实验才能得到。我们从原始数据集中随机抽取了 878 个包含全部 IAT 模式的 Obfs4 流和 7921 个背景流，比例接近 1∶10，更能反映现实网络环境 Obfs4 占比小的实际情况。通过设置了最佳参数的两阶段过滤，最终得到 0.98 的召回率、0.99 的精确度和 0.98 的准确度。然后，将 XGBoost 学习算法与最先进方法中使用的机器学习算法在我们收集的数据集上进行比较，得到的结果如表 4 所列。

表 4 不同分类算法准确度的比较

Table 4 Comparison of different classification methods

分类算法	精确度	召回率	准确度	F1 值
WKNN	0.8224	0.6143	0.6283	0.6213
ISVM	0.9656	0.9741	0.9656	0.9698
RF	0.9968	0.9929	0.9939	0.9934
CNN	0.9696	0.9903	0.9903	0.9799
XGBoost	0.9889	0.9803	0.9833	0.9906

实验结果基于对流的前 30 个数据包进行训练，并使用经过筛选的特征，其中 ISVM 方法进行 5 轮增量训练。实验结果显示，对于 WKNN 方法来说，仅使用 30 个数据包还不足以满足识别需求；而 ISVM 和 XGBoost 方法的表现非常出色，其中 XGBoost 方法略优于 ISVM；RF 和 CNN 方法的召回率和准确度均非常高，较 XGBoost 方法高出约 1 个百分点。总体而言，XGBoost，RF 和 CNN 方法在精度上都表现良好。

为了验证本文方法的实时性，需要对时间相关指标进行测量，本文根据是否统计模型训练时间将实验分成实际意义较强的两部分。实验 1 测量模型训练的时间消耗；实验 2 测量模型训练完成后预测算法正常工作的时间消耗。对于本文方法，正常工作时间消耗包括数据包读取（握手阶段过滤）、特征提取和模型预测（通信阶段过滤）；而对于其他对比方法，正常工作时间消耗则包括数据包读取（不过滤）、特征提取和模型预测；CNN 方法不需要特征提取。两者的区别在于，本文方法在握手数据包检测阶段会剪枝掉一些预测为非目标流量的流，这些流不会被抽取特征以及进行下一步预测。实验仍基于每个流的前 30 个数据包，并以单位流量 CPU 时间为评价指标。模型训练实验中，对完全相同的 5013 条数据使用不同方法进行实验，并且保持了与分类算法准确度实验相同的参数设置。实验结果如表 5 所列。

表 5 不同分类算法模型训练时间的比较

Table 5 Comparison of training time of different classification methods

选择的机器学习算法	单位流量 CPU 时间/ms
WKNN	0.168
ISVM	0.301
RF	0.171
CNN	67.020
XGBoost	0.251

除了 CNN 方法，其余各方法在模型训练上的时间开销差距基本不大，当数据集大小为 5013 时，模型训练总时间开销在 1s 左右。考虑到正常工作时定期更新模型的频率不会很高，只有 CNN 方法的时间开销太大。接着，我们逐阶段测量了不同分类算法正常工作的时间开销，在这里使用了全部的 5978 个 Obfs4 流和 46074 个背景流，得到的结果如表 6 所列。

表 6 不同分类算法正常工作时间的比较

Table 6 Comparison of normal uptime of different classification methods

选择的机器学习算法	单位流量 CPU 时间/ms
WKNN	6.878
ISVM	6.926
RF	6.512
CNN	9.669
XGBoost	4.423

通过握手阶段的剪枝，本文方法能有效地剔除接近 70%的背景流量，并节省对这些流量进行特征抽取的时间。实际上，由于现实互联网中 Obfs4 流的占比非常小，过滤掉 70%的背景流量几乎等同于过滤掉 70%的流量，从而大大减少了机器学习需要分类的目标流数量。多级剪枝策略先使用简单方法进行过滤，减少进入下一级过滤步骤的流基数，因此尽管检测会产生额外的开销，但总体的时间消耗远低于其他对比方法。WKNN，ISVM 和 RF 方法的时间开销接近，这是因为数据包读取和特征抽取占据了主要时间开销，而这两部分除了本文方法和 CNN 方法外是相同的。

结束语 本文提出了一种基于部分数据的多级剪枝 Obfs4 混淆流量识别方法，该方法能够对应用了 IAT 的 Obfs4 混淆流量进行快速、准确的识别。具体而言，该方法首先根据 Obfs4 流量握手数据包的随机性、时序、长度分布特征，在握手阶段进行多级剪枝，逐次去除干扰流量。然后，对

剩余的流量进行数据包特征序列收集，并抽取最先到达的数据包的特征，最后使用 XGBoost 方法进行机器学习。实际监管的网络是一个动态的、不稳定的环境，训练好的模型在长时间后可能会发生概念漂移现象，弱化识别效果。未来可以设计根据需要监控网络状态、检测到漂移现象后自动更新并优化模型参数的自动化方法，补充一些模型自我更新的策略，增强识别方法的可移植性和鲁棒性。

参 考 文 献

[1] DINGLEDINE R, MATHEWSON N, SYVERSON P F. Tor: The second-generation onion router[C]// Proceedings of the 13th USENIX Security Symposium. 2004, 4: 303-320.

[2] The Tor Project. Tor Metrics[EB/OL]. (2023-10-07) [2023-10-07]. https://metrics.torproject.org/networksize.html.

[3] FAERØY A. Meek [EB/OL]. (2020-06-15) [2023-10-07]. https://gitlab.torproject.org/legacy/trac/-/wikis/doc/meek.

[4] KADIANAKIS G, WILEY B, ANGEL Y, et al. Obfs2 (The Twobfuscator)[EB/OL]. (2013-02-08) [2023-10-07]. https://github.com/Null-Hypothesis/obfsproxy/blob/master/doc/obfs2/obfs2-protocol-spec.txt.

[5] KADIANAKIS G, WILEY B, ANGEL Y, et al. Obfs3 (The Threebfuscator) [EB/OL]. (2013-01-23) [2023-10-07]. https://github.com/Null-Hypothesis/obfsproxy/blob/master/doc/obfs3/obfs3-protocol-spec.txt.

[6] ANGEL Y, MARTÍ D. Obfs4 — The Obfourscator[EB/OL]. (2023-10-05) [2023-10-07]. https://github.com/Yawning/obfs4.

[7] FIFIELD D, BOCOVICH C, BREAULT A, et al. SnowFlake. [EB/OL]. (2021-11-04) [2023-10-07]. https://gitlab.torproject.org/tpo/anti-censorship/pluggable-transports/snowflake/-/wikis/Technical%20Overview.

[8] BERNSTEIN D J, HAMBURG M, KRASNOVA A, et al. Elligator: elliptic-curve points indistinguish-able from uniform random strings[C]// Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security. 2013: 967-980.

[9] LIANG D, HE Y Z. Obfs4 Traffic Identification Based on Multiple-feature Fusion[C]// 2020 IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS). IEEE, 2020: 323-327.

[10] HE Y, HU L P, GAO R. Detection of tor traffic hiding under obfs4 protocol based on two-level filtering[C]// 2019 2nd International Conference on Data Intelligence and Security (ICDIS). IEEE, 2019: 195-200.

[11] YAO Z J, GE J G, WU Y L, et al. Encrypted traffic classification based on Gaussian mixture models and Hidden Markov Models [J]. Journal of Network and Computer Applications, 2020, 166: 102711.

[12] WANG X B, LI Z Y, HUANG W T, et al. Towards Comprehensive Analysis of Tor Hidden Service Access Behavior Identification Under Obfs4 Scenario[C]// Proceedings of the 2021 ACM International Conference on Intelligent Computing and its Emerging Applications. 2021: 205-210.

[13] HE G F, YANG M, LUO J Z, et al. Online Identification of Tor Anonymous Communication Traffic[J]. Journal of Software, 2013, 24(3): 540-556.

[14] GAO R. Research on Anonymous Network Traffic Identification for Obfs4[D]. Beijing: Beijing Jiaotong University, 2018.

[15] CHEN T Q, GUESTRIN C. Xgboost: A scalable tree boosting system[C]// Proceedings of the 22nd ACM Sigkdd International Conference on Knowledge Discovery and Data Mining. 2016: 785-794.

[16] SMADIA S, ALMOMANIB O, MOHAMMADC A, et al. VPN Encrypted Traffic classification using XGBoost[J]. International Journal, 2021, 9(7): 960-966.

[17] MANJU N, HARISH B S, PRAJWAL V. Ensemble feature selection and classification of internet traffic using XGBoost classifier[J]. International Journal of Computer Network and Information Security, 2019, 11(7): 37-44.

[18] XU W L, ZOU F T. Obfuscated tor traffic identification based on sliding window[J]. Security and Communication Networks, 2021, 2021: 1-11.

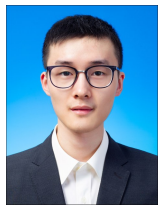
[19] 国家标准化管理委员会. 信息安全技术 二元序列随机性检测方法 [EB/OL]. (2016-08-29) [2023-10-07]. https://openstd.samr.gov.cn/bzgk/gb/newGbInfo?hcno=46D7E3E9C4B81DF460052FFEB706CAB0.

[20] MA C C, DU X H, CAO L F. Improved KNN algorithm for fine-grained classification of encrypted network flow[J]. Electronics, 2020, 9(2): 324.

[21] SUN G L, CHEN T, SU Y Y, et al. Internet traffic classification based on incremental support vector machines[J]. Mobile Networks and Applications, 2018, 23: 789-796.

[22] SOLEIMANI M H M, MANSOORIZADEH M, NASSIRI M. Real-time identification of three Tor pluggable transports using machine learning techniques[J]. The Journal of Supercomputing, 2018, 74(10): 4910-4927.

[23] LOTFOLLAHI M, ZADE R S H, SIAVOSHANI J M, et al. Deep packet: A novel approach for encrypted traffic classification using deep learning[J]. Soft Computing, 2020, 24(3): 1999-2012.



XU Chenhan, born in 2000, postgraduate, is a student member of CCF (No. T4099G). His main research interest is network traffic measurement.

SUN Yu'e, born in 1983, Ph.D supervisor, is a member of CCF (No. 28402M). Her main research interests include traffic measurement, Internet of Things, privacy preserving and crowd sensing.

（责任编辑：喻藜）