计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2021,30(8):186−193 [doi: 10.15888/j.cnki.csa.008057]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于流量摘要的僵尸网络检测①

肖喜生[1,2], 龙　春[1,2], 杜冠瑶[1,2], 魏金侠[1], 赵　静[1,2], 万　巍[1,2]

[1](中国科学院 计算机网络信息中心, 北京 100190)
[2](中国科学院大学 计算机科学与技术学院, 北京 101408)
通讯作者: 龙　春, E-mail: anquanip@cnic.cn

摘　要: 随着僵尸网络的日益进化, 检测和防范僵尸网络攻击成为网络安全研究的重要任务. 现有的研究很少考虑到僵尸网络中的时序模式, 并且在实时僵尸网络检测中效果不佳, 也无法检测未知的僵尸网络. 针对这些问题, 本文提出了基于流量摘要的僵尸网络检测方法, 首先将原始流数据按照源主机地址聚合, 划分适当的时间窗口生成流量摘要记录, 然后构建决策树、随机森林和 XGBoost 机器学习分类模型. 在 CTU-13 数据集上的实验结果表明, 本文提出的方法能够有效检测僵尸流量, 并且能够检测未知僵尸网络, 此外, 借助 Spark 技术也能满足现实应用中快速检测的需要.

关键词: 僵尸网络; 机器学习; Spark; 流量摘要

引用格式: 肖喜生,龙春,杜冠瑶,魏金侠,赵静,万巍.基于流量摘要的僵尸网络检测.计算机系统应用,2021,30(8):186–193. http://www.c-s-a.org.cn/1003-3254/8057.html

Botnet Detection Based on Flow Summary

XIAO Xi-Sheng[1,2], LONG Chun[1,2], DU Guan-Yao[1,2], WEI Jin-Xia[1], ZHAO Jing[1,2], WAN Wei[1,2]

[1](Computer Network Information Center, Chinese Academy of Sciences, Beijing 100190, China)
[2](School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 101408, China)

Abstract: With the development of botnets, detecting and preventing botnet attacks has become an important task of network security research. Existing studies, which rarely consider the timing patterns in botnets, are ineffective in real-time botnet detection and cannot detect unknown botnets. To tackle these problems, this study proposes a flow summary based botnet detection method. First, the network flow data is aggregated according to the source host IPs, and the flow summary records are generated in a given time window. Then, decision tree, random forest, and XGBoost machine-learning classification models are built to validate the performance of our method. The experimental results on the CTU-13 dataset show that the method we propose can effectively detect botnet traffic and detect unknown botnets. With the help of Spark technology, our method can also meet the needs of rapid detection in real applications.

Key words: botnet; machine learning; Spark; flow summary

根据僵尸网络威胁报告 Spamhaus Botnet Threat Update Q1-2020 [1] 调研, 传统的 C&C 相关的僵尸网络数量减少了一半, 但新的恶意软件已大量涌现, 这些新兴恶意软件利用特定的云基础设施进行非法活动. 黑客利用僵尸网络进行点击欺诈、分布式拒绝服务攻击(DDoS)、发送垃圾邮件和窃取个人信息等恶意活动, 僵尸网络中的主机通常在所有者不知情、未授权的情况下被黑客劫持. 然后, 黑客通过控制这些主机一起攻

① 基金项目: 国家重点研发计划网络空间安全重点专项 (2017YFB0801902); 中国科学院“十四五”网信专项先期建设项目 (WX145XQ11)
Foundation item: National Key Research and Development Program on Cyberspace Security (2017YFB0801902); Early Construction Project of the 14th Five-Year Plan Network and Information Technology Special Project, CAS (WX145XQ11)
收稿时间: 2020-11-23; 修改时间: 2020-12-22, 2021-01-08; 采用时间: 2021-01-13; csa 在线出版时间: 2021-07-31

击更多主机. 僵尸网络也可以通过在网络空间中传播恶意软件或勒索软件达到攻击的目的. 因此, 检测僵尸网络并防范其攻击是网络安全研究的重要任务.

随着技术的发展, 用于感染主机和运行僵尸网络的恶意软件为了逃避检测也迅速发展, 从而令许多常用的僵尸网络检测技术失效. 例如, 僵尸网络通过改变其通信协议来逃避检测[2]. 在一开始, 僵尸网络使用 IRC (Internet Relay Chat) 协议来控制其他主机. 此后, 网络空间中逐渐出现采用 P2P 协议通信的僵尸网络, 其中每个主机都充当客户端和服务器; 然后基于 HTTP 协议的僵尸网络开始流行[3]. 2016 年, Methbot 僵尸网络成为有史以来最大的数字广告恶意软件, 该恶意软件通过位于美国的 ISP 获得了数千个 IP 地址. Mirai 僵尸网络[4] 在同年年底席卷了整个互联网, Mirai 发动了几次大规模 DDoS 攻击破坏了大量主流站点. 还有一些更复杂更隐蔽的僵尸网络通过更改其通信模式来长时间隐藏而不被发现. 因此, 僵尸网络检测算法需要与时俱进, 要能迅速适应不断进化的僵尸网络.

现有的僵尸网络检测技术仍存在以下问题:

(1) 大多数检测方法能够很好的拟合训练数据, 然而在测试数据上显现出效果不佳的问题, 普遍存在着模型过拟合的情况;

(2) 检测模型泛化能力差, 针对已知的单一类型的僵尸网络检测效果较好, 但面对未知类型僵尸网络乏力;

(3) 大部分检测方法忽略了网络流量中的时序通信模式, 导致在实际网络环境中检测效果不佳, 应用性不强;

考虑到上述问题, 本文提出了基于流量摘要的僵尸网络机器学习检测方法, 首先将原始流数据按照源主机 IP 地址聚合, 划分适当的时间窗口, 利用 Spark 计算所选原始特征的统计特征生成流量摘要记录, 对时间窗口内该主机的通信行为进行建模, 然后构建机器学习分类模型用于检测僵尸流量. 最后利用 CTU-13 数据集对本文提出的方法进行验证, 实验结果表明本文提出的方法能够有效检测僵尸流量, 并且能够检测未知僵尸网络.

1 相关工作

目前, 僵尸网络检测领域已有大量的国内外学者开展了相关方面的研究, 传统的方法有人工分析或黑白名单过滤, 或通过手动维护相应的签名数据库进行简单的匹配.

Gadelrab 等[5] 通过分析已知僵尸网络的几种恶意软件样本, 确定了一组特征, 这些特征可以帮助区分正常和僵尸网络流量. Gu 等[6] 在他们的工作中研究了僵尸网络恶意软件感染的生命周期. 2008 年, 他们在后续的研究[7] 中提出了一种基于网络的异常检测方法来识别局域网中的 C&C 僵尸网络, 而无需先验签名数据库和 C&C 服务器地址. 他们的方法能够识别网络中的 C&C 服务器和受感染的主机. 他们通过部署蜜罐来验证其方法的有效性. 其他一些文献[2,8,9] 论述了基于蜜罐的僵尸网络检测方法的局限性. 蜜罐在检测多种漏洞利用方面存在局限性, 无法扩展到其他恶意攻击, 也不能实时检测攻击. 此外, 在自己的环境中构建的蜜罐不是开源的, 并且由于公开僵尸网络数据集的缺乏, 无法比较实验的结果. Garcia 等[10] 认为先前僵尸网络检测方法没有进行任何对比, 因此他们采集并开源了一个具有标签的僵尸网络数据集 CTU-13, 其中包括僵尸流量、正常流量和背景流量. 他们为僵尸网络检测方法设计了一种新的评价指标并且比较了 (BClus 和 CAMNEP) 与 BotHunter[6] 的检测结果. 由于一些僵尸程序的预先编程性质, 致使僵尸网络流量表现出重复的行为模式, 仍有大多数研究没有考虑到网络流量中的时序模式[11,12]. 尽管有其他一些论文考虑了时间模式, 但仍然有一定的局限性. 他们只考虑某些特定源 IP 地址中的时间特征, 而没有考虑整体的网络流量, 使得在实时僵尸网络检测中效果不佳[13,14]. 另外, 一些现有的研究仅限于传统的基于 IRC、P2P 和 HTTP 协议的僵尸网络. 因此, 这些方法无法检测各种类型的僵尸网络或未知的僵尸网络.

基于机器学习的方法是最近比较流行的僵尸网络检测方法, 安全管理员可以使用训练好的机器学习模型识别僵尸网络. 基于机器学习的方法可从僵尸网络数据集中自动提取代表性和容易区分的特征, 不需要有关僵尸网络流量的任何先验信息.

支持向量机 (SVM) 由于其出色的泛化性能被广泛用于许多安全应用中[15,16]. Hoang 等[17] 提出了一个基于机器学习的僵尸网络检测模型, 论文使用域名服务查询数据并使用 KNN、决策树、随机森林和朴素贝叶斯算法. 实验结果表明, 随机森林算法的准确率最高, 达到了 90.80%. 但是同时假阳性率也相对较高. Haddadi 等[18] 比较了 5 种不同的僵尸网络检测方法,

其中包括两种基于签名的方法 BotHunter 和 Snort, 其余方法将机器学习算法应用于不同的特征集, 包括基于数据包有效负载和基于网络流的特征. 他们在包含 25 个僵尸网络的数据集上执行多分类和二进制分类测试, 实验结果表明, 基于流的特征是对僵尸网络通信进行建模的最具代表性的特征, C4.5 算法则达到了最高的分类准确性. Drašar 等[19] 详细说明了检测异常的特征选择方法, 他们评估了基于流的特征对检测精度的影响. Stevanovic 等[20] 提出了基于流的僵尸网络检测方法, 选择了 39 个流特征 (例如源端口、目标端口、数据包大小的标准差和流持续时间) 用于对恶意流量进行建模. 他们评估了 8 种监督机器学习算法, 包括朴素贝叶斯、决策树、SVM、贝叶斯网络分类器和逻辑回归. 结果表明, 随机森林算法识别僵尸网络的准确率最高, 达到了 95.7%. Chen 等[21] 通过监督式机器学习算法快速检测复杂网络中的僵尸网络. 他们将基于流和基于会话的特征组合以建立分类模型, 并在 CTU-13 数据集上进行了实验, 同时分析了各种算法的性能, 其中随机森林算法达到了 94% 的精度. Niu 等[22] 使用 XGBoost 算法在 HTTP 流量上检测受感染的主机, 检测准确率达到 98.72%, 假阳性率小于 1%.

分析以上研究发现, 现有的检测方法只针对特定的僵尸网络, 并且会遗漏一些僵尸网络主机之间特定的时序通信模式. 此外, 针对单条流或单个主机的处理会导致较高的时间和计算开销, 此类检测技术也就无法应用于实时僵尸网络检测. 如何更加有效地提取僵尸网络流特征, 如何提高检测方法的泛化性并提高检测效率及精度, 需要进一步研究. 因此, 本文提出基于流量摘要的僵尸网络机器学习检测方法, 以改善上述问题.

2 基于流量摘要的僵尸网络检测方法

2.1 机器学习技术

机器学习是人工智能的一种应用, 它使学习系统能够自动学习并从历史经验中进行改进. 机器学习算法是一类从数据中自动分析获得规律, 并利用规律对未知数据进行预测的算法[23]. 在本文中, 仅考虑了监督学习技术在僵尸网络检测中的性能, 这里本文选择在之前研究中被证明具有较好性能的监督机器学习算法, 包括决策树、随机森林和 XGBoost.

(1) 决策树

决策树 (Decision Tree, DT) 是一种预测模型, 代表的是对象属性与对象值之间的一种映射关系. 决策树通过创建一组决策规则来对对象进行分类, 这些规则是根据训练数据的特征集提取的. 在决策树中, 叶子代表类, 树中的每个子节点及其分支代表导致分类的特征的组合. 因此, 对对象进行分类时首先检查根节点的值, 然后在对应于那些值的树下继续向下. 对每个节点重复执行此过程, 直到遍历到叶节点为止. 决策树通常使用信息增益 (IG) 和基尼系数来选择决策树的特征. 本文的实验使用的决策树算法是优化的 CART 算法.

(2) 随机森林

随机森林 (Random Forest, RF) 是一个包含多个决策树的分类器, 并且其输出的类别是由个别树输出的类别的众数而定. 该算法的核心思想是创建一系列决策树, 这些决策树单独训练并得出独立的结果, 随机森林选择选择的结果即为最多的树预测的结果.

(3) XGBoost

XGBoost 模型是为在不平衡的数据集上运行而构建的, 因为重新采样是在内部进行的, 因此它可以抵抗数据不平衡. XGBoost 被称为极限梯度提升, 是一种顺序决策树. 通过不断地添加树, 不断地进行特征分裂来生长一棵树, 每次添加一个树, 拟合上次预测的残差. XGBoost 方法不同于传统的基于决策树的集成学习方法, 它在损失函数里加入了叶子结点权重和单个决策树模型复杂度等正则项, 这样可以防止决策树模型过于复杂, 进而防止过拟合.

2.2 流量摘要

为了准确地描述特定时间窗口内主机的流量行为, 本文提出了一种基于统计特征的网络流量摘要方法, 利用该方法得到的流量摘要记录包括特定时间窗口内主机发送流量的统计特征.

首先, 根据网络流数据集中的源 IP 地址对流进行分组. 然后, 提取传输层协议的特征, 包括 TCP 和 UDP 协议, 这是上层应用层协议的基础. 本文选择的 4 个流特征为: Dur, TotPkts, TotBytes 和 SrcBytes. 表 1 列出了从网络流量数据中提取的基于 TCP 协议的流特征及其描述. 针对 UDP 协议的流特征提取这里不再赘述.

针对根据源 IP 地址聚合后的网络流, 具体的流量摘要方法如下: 给定一个时间窗口值 t, 计算 t 内基于 TCP 和 UDP 协议选择的上述 4 个特征计算 4 个统计值, 包括最小、最大、均值和标准差值. 对于每个新特征名称, 其前缀反映其统计特征, 然后描述协议类型.

例如, 特征名称 Mean_TCP_Dur 表示在时间窗口 t 内使用 TCP 协议通信的持续时长平均值. 在处理了每个源 IP 地址的每个时间窗口之后, 获得了流量摘要记录的集合, 其中共包含 32 个提取的特征. 图 1 详细描述了流程摘要过程. 通过对给定时间窗口的网络流进行流量摘要, 计算传输层协议统计特征, 以获取时间窗口内该主机的时序通信行为模式, 为之后的僵尸流量检测提供数据支撑.

上述的流量摘要任务涉及到复杂的聚合和统计任务, 使用传统的单机处理方法非常耗时, 一种有效的方法是使用分布式大数据处理技术[24], 将任务分配给不同的计算节点, 然后对结果进行汇总. 因此本文使用流行的大数据处理框架 Apache Spark 来进行流量摘要任务处理.

表 1 基于 TCP 的流特征集描述

特征名称	特征数	描述
TCP_Dur	4	TCP流持续时长
TCP_ TotPkts	4	TCP传输总包数
TCP_ TotBytes	4	TCP传输总字节数
TCP_ SrcBytes	4	TCP传输源字节数

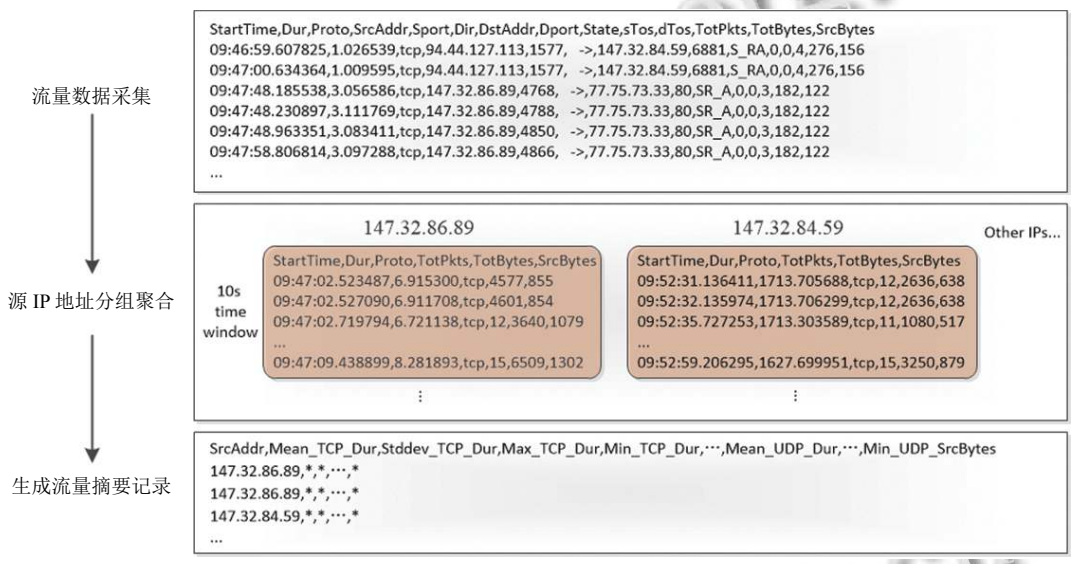

图 1 流量摘要过程描述

2.3 基于流量摘要的僵尸网络检测方法

本文提出了一种基于流量摘要的僵尸网络机器学习检测方法, 如图 2 所示. 本文的方法分 5 个步骤: 流采集、流量摘要、数据预处理、分类和评估. 第 1 步, 流采集可以从某些路由器或其他网络流采集器 (例如 NetFlow 采集设备) 收集网络流数据. 本文使用开源的僵尸网络数据集. 在第 2 步, 本文为流量摘要任务部署了一个 Spark 集群环境, 利用 Spark 技术快速完成流量摘要记录生成. 第 3 步, 将流量摘要记录汇总成新数据集, 并对其进行重新标记, 然后进行特征预处理, 这里由于生成的特征都是数值型特征, 因此进行归一化处理, 特征预处理后将数据集划分为训练集和测试集. 在第 4 步, 基于流量摘要记录数据集训练不同的机器学习分类模型, 所用的分类算法为决策树, 随机森林和 XGBoost, 用于分类僵尸流量和非僵尸流量. 最后一步, 通过实验验证本文提出的检测方法, 利用分类评价指标评估比较不同分类算法在流量摘要数据集的检测性能, 并讨论不同时间窗口值对检测结果的影响.

3 实验分析

为了验证本文检测方法的效果, 本节在流量摘要生成的新数据集上比较了决策树、随机森林和 XGBoost 算法的分类性能, 然后验证了不同时间窗口值对分类结果的影响.

3.1 数据集

本文使用开源的僵尸网络数据集 CTU-13[10], 该数据集包含 13 种僵尸程序感染场景. 原始流量为 PCAP 格式, 由多个数据包组成, 对 PCAP 文件进行进一步处理可以获得 NetFlow 文件, 这些文件包含标签并可以很好地区分客户端和服务器. 文献 [10] 使用单向 NetFlow

表示流量并标记标签, 但作者认为不应该使用这些单向 NetFlow 文件, 因为使用双向 NetFlow 文件效果更好. 双向 NetFlow 相对于单向 NetFlow 具有几个优点. 双向 NetFlow 文件解决了客户端和服务器的区分问题, 包含了更多的信息, 并且包含了更详细的标签. 因此, 本文也使用双向 NetFlow 文件进行实验.

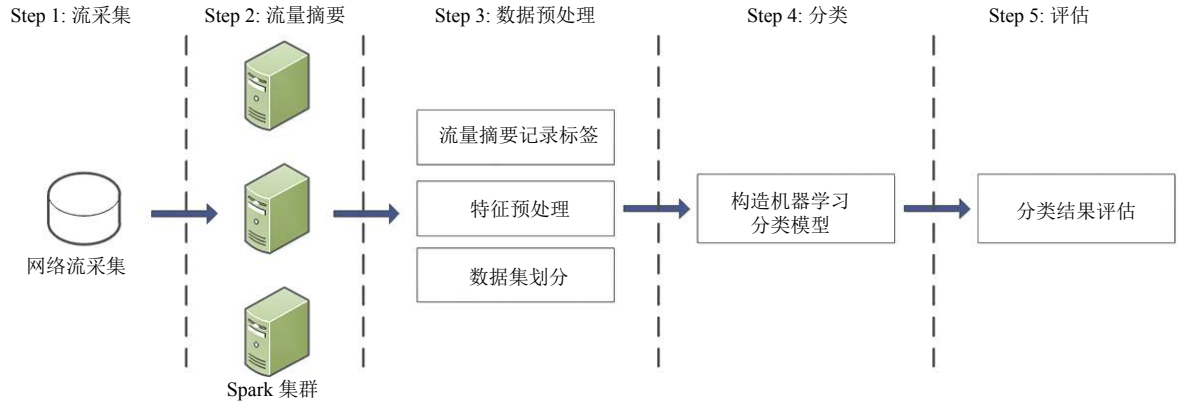

图 2 基于流量摘要的僵尸网络检测流程

3.2 数据预处理

根据前文对流量摘要的描述, 原始流数据首先根据预先给定的时间窗口 (例如 10 s 的时间窗口) 按源 IP 地址进行分组, 此过程还包括计算该时间窗口下不同的协议的统计特征生成流量摘要记录. 分别处理了 13 个 CTU-13 数据集的攻击场景之后, 获得了 13 个流量摘要记录组. 然后根据源 IP 对流量摘要记录重新标记. 例如在场景 1 中, 主机 147.32.84.165 是被感染的僵尸主机, 因此由该 IP 聚合的流量摘要记录将被标记为僵尸流量. 本文选择不同的时间窗口进行聚合, 包括 5 s、10 s、15 s、30 s 和 60 s, 以验证不同时间窗口值对分类结果的影响. 表 2 显示了整个 CTU-13 数据集进行流量摘要后标签分布结果, 其中数字 0 表示正常流量或背景流量, 数字 1 表示僵尸流量.

表 2 流量摘要数据集标签分布

场景	5 s		10 s		15 s		30 s		60 s	
	0	1	0	1	0	1	0	1	0	1
1	1241154	3150	1207177	1691	1189527	1132	1162983	567	1139662	284
2	805841	1930	780651	1151	768308	800	750274	404	734239	204
3	1571101	9373	1405653	5119	1318745	3431	1202631	1725	1104355	870
4	313269	551	290281	365	278501	270	261322	153	247361	90
5	41422	202	38897	111	37610	77	35762	40	34187	21
6	146682	716	136361	570	131209	457	123673	243	117257	123
7	33551	12	31543	9	30469	7	28930	6	27574	5
8	1037274	1870	970113	1769	937989	1568	894147	1281	856238	1094
9	682885	15963	653911	8123	639750	5439	619213	2752	600753	1403
10	367084	1657	341512	1019	329081	775	310745	500	294037	357
11	38496	47	36762	29	35950	22	34630	16	33520	11
12	100260	294	94351	204	91210	160	86404	111	82060	81
13	812936	11387	761651	5813	736204	3891	702126	1953	674040	987

由于流量摘要数据集包含的 32 个特征都是数值型特征, 在数据处理过程中需要处理数值分布区间较大的情形, 因此需要对数据进行归一化处理, 将原本随机分散的数据经过处理压缩到一个较小的分布区间, 避免离散点对检测结果产生较大的影响.

3.3 数据集过采样

CTU-13 数据集存在严重的类别不平衡问题, 尤其是在流量摘要处理之后, 这种不平衡现象更加严重, 如表 2 所示, 僵尸流量在整个网络流量中占比极少. 为了解决类别不平衡的问题, 本文采用了一种称为 SMOTE

(Synthetic Minority Oversampling TEchnique)[25] 的过采样技术来克服极端的类别不平衡问题. SMOTE 的核心思想是在少数类样本及其邻居之间插入随机生成的新样本, 这可以增加少数类样本的数量并改善类别不平衡的状况.

3.4 结果与分析

本文实验在准确率 (*Acc*)、精确率 (*Pre*)、召回率 (*Rec*) 和 *F*1 值上进行对比, 各性能指标公式如下所示:

$$Acc = (TP+TN)/(TP+TN+FP+FN) \quad (1)$$

$$Pre = TP/(TP+FP) \quad (2)$$

$$Rec = TP/(TP+FN) \quad (3)$$

$$F1 = 2*(Pre*Rec)/(Pre+Rec) \quad (4)$$

其中, *TP* 代表着真实值属于正类, 预测值属于正类的数量; *TN* 代表着真实值属于负类, 预测值属于负类的数量; *FP* 代表着真实值属于负类, 预测值属于正类的数量; *FN* 代表着真实值属于正类, 预测值属于负类的数量.

本文首先分别在数据集的所有场景上进行实验, 时间窗口为 10 s, 3 种机器学习算法的分类结果如表 3 所示. 结果表明, 所有算法均具有较好的分类效果, 其中 XGBoost 在各种指标上具有最稳定的性能.

为了进一步验证本文方法的检测性能, 对比文献 [26] 中使用随机森林对 Neris 僵尸网络进行检测的结果, 本文同样使用随机森林算法进行对比实验, Neris 僵尸网络包括场景 1、2 和 9, 轮流选择 1 个场景用于测试, 剩下两个场景用于训练, 10 s 时间窗口的实验结果如表 4 所示, 其中本文的实验结果用*标记.

从表 4 可见, 本文的方法在所有分类指标上均有提升, 上文的实验在单一类型恶意软件的僵尸网络检测上具有良好的效果, 为了检测未知的僵尸网络, 必须考虑更多类型恶意软件的僵尸网络. 参考文献 [9] 建议, 本文将场景 1、2、6、8、9 组合用于测试, 将其他场景用于训练. 训练集中的僵尸程序包括 Rbot, Virut, Sogou 和 NSIS.ay, 测试集中的僵尸程序 Neris, Menti 和 Murlo. 以 10 s 的时间窗口对整个数据集进行的实验结果如图 3 所示.

从图 3 可以看出, XGBoost 分类器在检测未知僵尸流量上具有最优的检测性能, 其中精确率达到了 96.50%, *F*1 值也达到了 79.55%, 由于使用分类器的默认参数, 因此以后的工作可以研究针对参数进行优化, 进一步提升分类器的检测性能.

为了进一步验证不同时间窗口值对分类结果的影响, 本文选择了 5 s, 10 s, 15 s, 30 s 和 60 s 等不同的时间窗口, 利用分类性能较好的 XGBoost 进行实验. 结果如表 5 所示.

表 3 各个场景下的分类结果

场景	算法	*Acc* (%)	*Pre* (%)	*Rec* (%)	*F*1 (%)
1	DT	99.92	95.71	96.81	96.26
	RF	99.97	99.04	98.11	98.57
	XGB	**99.97**	**99.10**	**98.63**	**98.87**
2	DT	99.91	95.04	96.19	95.61
	RF	99.95	98.31	96.64	97.47
	XGB	**99.97**	**98.93**	**97.98**	**98.46**
3	DT	99.96	98.21	98.26	98.24
	RF	99.98	99.07	98.48	**98.77**
	XGB	**99.98**	**99.25**	**98.58**	98.52
4	DT	99.60	87.02	67.00	75.71
	RF	99.61	87.93	**67.46**	76.35
	XGB	**99.62**	**88.84**	67.09	**76.45**
5	DT	99.77	86.16	90.13	88.10
	RF	99.90	**97.22**	92.11	94.60
	XGB	**99.91**	96.00	**94.74**	**95.36**
6	DT	99.75	82.27	96.84	88.96
	RF	99.79	84.76	97.54	90.70
	XGB	**99.80**	**84.85**	**98.25**	**98.06**
7	DT	99.80	93.04	86.29	89.54
	RF	99.89	**100.00**	88.71	94.02
	XGB	**99.94**	99.15	**94.36**	**96.69**
8	DT	99.71	78.02	98.33	87.00
	RF	99.72	**78.95**	98.13	87.50
	XGB	**99.72**	78.89	**98.36**	**87.55**
9	DT	99.71	94.63	95.44	95.03
	RF	99.83	97.72	96.55	97.13
	XGB	**99.90**	**98.65**	**98.03**	**98.34**
10	DT	97.35	64.51	98.14	77.85
	RF	97.37	64.64	98.25	77.98
	XGB	**97.37**	**64.66**	**98.46**	**78.05**
11	DT	97.58	67.12	98.94	79.98
	RF	97.64	67.64	98.81	80.30
	XGB	**97.65**	**67.66**	**99.20**	**80.45**
12	DT	99.65	81.74	81.74	81.74
	RF	99.76	**97.60**	77.38	86.32
	XGB	**99.81**	96.22	**83.11**	**89.18**
13	DT	99.91	95.61	95.30	95.46
	RF	99.96	99.19	96.19	97.66
	XGB	**99.97**	**99.30**	**97.80**	**98.54**

表 4 Neris 僵尸网络分类结果

训练场景	测试场景	*Pre*	*Rec*	*F*1
2, 9	1	0.84	0.98	0.91
		0.92*	0.99*	0.95*
1,9	2	0.96	0.96	0.96
		0.98*	0.99*	0.98*
1,2	9	1	0.92	0.96
		0.99*	0.98*	0.98*

从表 5 可以看出, 10 秒的时间窗口下进行流量摘要得到的综合结果最好, 其中准确率 (*Acc*) 达到了 82.60%, 召回率 (*Rec*) 达到了 67.66%, *F*1 值达到了 79.55%, 这 3 项指标为所有时间窗口实验下的最大值. 综上所述, 由于时间窗口较小, 无法捕获僵尸网络流量的时序特征, 时间窗口较大则无法满足现实应用中近实时检测的需要, 导致检测结果不佳, 在实际应用中可划分更加细粒度的时间区间进行实验以确定最佳的时间窗口值.

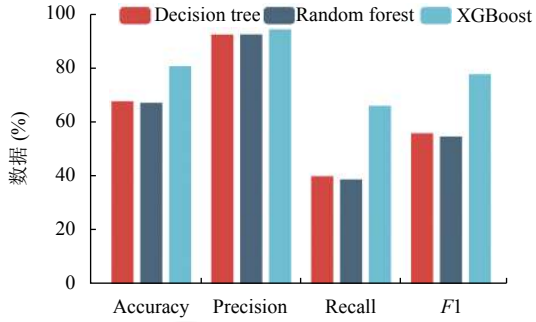

图 3 全数据集分类结果 (10 s 时间窗口)

表 5 不同时间窗口下的全数据集分类结果

窗口(s)	*Acc* (%)	*Pre* (%)	*Rec* (%)	*F*1 (%)
5	80.65	96.72	63.81	76.89
10	**82.60**	96.50	**67.66**	**79.55**
15	80.92	96.77	63.99	77.03
30	75.03	95.91	52.30	67.69
60	78.44	**97.02**	58.69	73.14

4 总结

本文提出了基于流量摘要的僵尸网络机器学习检测方法, 首先将原始流数据按照源主机地址聚合, 划分适当的时间窗口生成流量摘要记录, 然后构建决策树、随机森林和 XGBoost 机器学习分类模型. 在 CTU-13 数据集上的实验结果表明, 本文提出的方法能够有效检测僵尸流量, 并且能够检测未知僵尸网络, 此外, 借助 Spark 大数据技术也能满足现实应用中快速检测的需要. 未来的工作将研究针对分类器参数的优化方法, 以进一步提高检测未知僵尸网络的检测能力.

参考文献

1 The Spamhaus Project. Botnet threat update Q1-2020. https://www.spamhaus.org/news/images/botnet-report-2020-q1/2020-q1-botnet-threat-report.pdf. [2020-09-11].

2 Mahmoud M, Nir M, Matrawy A. A survey on botnet architectures, detection and defences. International Journal of Network Security, 2015, 17(3): 272–289.

3 AsSadhan B, Bashaiwth A, Al-Muhtadi J, *et al*. Analysis of P2P, IRC and HTTP traffic for botnets detection. Peer-to-Peer Networking and Applications, 2018, 11(5): 848–861. [doi: 10.1007/s12083-017-0586-0]

4 Antonakakis M, April T, Bailey M, *et al*. Understanding the mirai botnet. Proceedings of the 26th USENIX Conference on Security Symposium. Vancouver, BC, Canada. 2017. 1093−1110.

5 Gadelrab MS, Elsheikh M, Ghoneim M, *et al*. BotCap: Machine learning approach for botnet detection based on statistical features. International Journal of Communication Networks and Information Security (IJCNIS), 2018, 10(3): 563–579.

6 Gu GF, Porras P, Yegneswaran V, *et al*. BotHunter: Detecting malware infection through IDSdriven dialog correlation. USENIX Security Symposium. Boston, MA, USA. 2007. 7.

7 Gu GF, Zhang JJ, Lee W. BotSniffer: Detecting botnet command and control channels in network traffic. Proceedings of the 15th Annual Network and Distributed System Security Symposium. San Diego, CA, USA. 2008.

8 Feily M, Shahrestani A, Ramadass S. A survey of botnet and botnet detection. 2009 Third International Conference on Emerging Security Information, Systems and Technologies. Athens, Greece. 2009. 268−273.

9 Amini P, Araghizadeh MA, Azmi R. A survey on Botnet: Classification, detection and defense. 2015 International Electronics Symposium (IES). Surabaya, Indonesia. 2015. 233−238.

10 García S, Grill M, Stiborek J, *et al*. An empirical comparison of botnet detection methods. Computers & Security, 2014, 45: 100–123.

11 Zhao D, Traore I, Sayed B, *et al*. Botnet detection based on traffic behavior analysis and flow intervals. Computers & Security, 2013, 39: 2–16.

12 AsSadhan B, Moura JMF, Lapsley D, *et al*. Detecting botnets using command and control traffic. 2009 Eighth IEEE International Symposium on Network Computing and Applications. Cambridge, MA, USA. 2009. 156−162.

13 Torres P, Catania C, Garcia S, *et al*. An analysis of recurrent neural networks for botnet detection behavior. 2016 IEEE biennial congress of Argentina (ARGENCON). Buenos

Aires, Argentina. 2016. 1−6.
14 Sinha K, Viswanathan A, Bunn J. Tracking temporal evolution of network activity for botnet detection. arXiv preprint arXiv: 1908.03443, 2019.
15 Kondo S, Sato N. Botnet traffic detection techniques by C&C session classification using SVM. Second International Workshop on Security. Nara, Japan. 2007. 91−104.
16 Bai XF, Zhang TJ, Wang CJ, *et al*. A fully automatic player detection method based on one-class SVM. IEICE Transactions on Information and Systems, 2013, E96. D(2): 387–391. [doi: 10.1587/transinf.E96.D.387]
17 Hoang XD, Nguyen QC. Botnet detection based on machine learning techniques using DNS query data. Future Internet, 2018, 10(5): 43. [doi: 10.3390/fi10050043]
18 Haddadi F, Zincir-Heywood AN. Botnet behaviour analysis: How would a data analytics-based system with minimum *a priori* information perform? International Journal of Network Management, 2017, 27(4): e1977. [doi: 10.1002/nem.1977]
19 Drašar M, Vizváry M, Vykopal J. Similarity as a central approach to flow-based anomaly detection. International Journal of Network Management, 2014, 24(4): 318–336. [doi: 10.1002/nem.1867]
20 Stevanovic M, Pedersen JM. An efficient flow-based botnet detection using supervised machine learning. 2014 International Conference on Computing, Networking and Communications (ICNC). Honolulu, HI, USA. 2014. 797−801.
21 Chen RD, Niu WN, Zhang XS, *et al*. An effective conversation-based botnet detection method. Mathematical Problems in Engineering, 2017, 2017: 4934082.
22 Niu WN, Li T, Zhang XS, *et al*. Using XGBoost to discover infected hosts based on HTTP traffic. Security and Communication Networks, 2019, 2019: 2182615.
23 孙哲南, 张兆翔, 王威, 等. 2019 年人工智能新态势与新进展. 数据与计算发展前沿, 2019, 1(2): 1–16.
24 华强胜, 郑志高, 胡振宇, 等. 大数据基础理论与系统关键技术浅析. 数据与计算发展前沿, 2019, 1(1): 22–34.
25 Chawla NV, Bowyer KW, Hall LO, *et al*. SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 2002, 16: 321–357. [doi: 10.1613/jair.953]
26 Ongun T, Sakharaov T, Boboila S, *et al*. On designing machine learning models for malicious network traffic classification. arXiv preprint arXiv: 1907.04846, 2019.