

FC-Bot: 基于流点聚合的轻量级物联网僵尸网络检测方法

周纸墨¹, 邹福泰¹, 姜开达², 康学斌³

¹ 上海交通大学网络空间安全学院 上海 中国 200240

² 上海交通大学网络信息中心 上海 中国 200240

³ 深圳安天网络安全技术有限公司 深圳 中国 518000

摘要 随着物联网时代的到来, 物联网设备快速增长, 然而这也为物联网攻击创造了有利条件。其中, 僵尸网络作为一种具有极高威胁和破坏力的攻击形式, 迫切需要一种高效的物联网僵尸网络检测方案。然而, 实时流量的巨大体量给检测带来了巨大挑战。因此, 本文提出了一种名为 FC-Bot 的基于流点聚合的轻量级物联网僵尸网络检测方法, 在保护隐私的同时实现了轻量级的检测。FC-Bot 创新性地引入了流聚合模块, 通过结合 HDBSCAN 算法将网络流转换为事务, 从更高维度检测物联网僵尸网络。首先, 根据源 IP 地址和目标 IP 地址对流数据进行预分组, 然后将每个分组的流数据按照时间窗口切割为流点, 再利用 HDBSCAN 算法将这些流点聚合为事务。通过这种方式, 有效地解决了流数据的时间统一问题, 并能够将聚合后的行为达到集体性级别, 从事务的角度区分正常流量和僵尸网络流量。最后, 我们提取事务特征指纹并将其转化为指纹图像, 使用卷积神经网络进行检测和分类。在收集整理的公开数据集上的评估实验中, FC-Bot 实现了 95.04% 的 F1 值, 证明了该方案的有效性。与目前先进的四种检测方案相比, FC-Bot 的误报率仅为 3.06%, 改善率最高达 4.30%, 具有更大的优势。此外, 在真实场景的评估实验中, FC-Bot 也实现了 92.59% 的 F1 值, 为物联网僵尸网络提供了一种轻量级且有效的检测方案。

关键词 物联网; 僵尸网络; 轻量级; 流聚合; 特征指纹

中图法分类号 TP309 DOI 号 10.19363/J.cnki.cn10-1380/tn.2026.05.11

FC-Bot: A Lightweight IoT Botnet Detection Method Based on Flow Point Clustering

ZHOU Zhimo¹, ZOU Futai¹, JIANG Kaida², KANG Xuebin³

¹ School of Cyber Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China

² Network & Information Center, Shanghai Jiao Tong University, Shanghai 200240, China

³ Shenzhen Antiy Network Security Technology Co., Ltd., Shenzhen 518000, China

Abstract With the advent of the Internet of Things (IoT) era, there has been a rapid proliferation of IoT devices. However, this growth has also created a fertile breeding ground for IoT attacks. Among them, botnets pose a significant risk with their high level of threat and potential for destruction, demanding an efficient detection solution for IoT botnet. However, the immense volume of real-time traffic presents a major challenge to detection. Therefore, we propose a lightweight IoT botnet detection method called FC-Bot based on flow point aggregation, which not only achieves lightweight detection but also ensures privacy preservation. FC-Bot innovatively introduces a flow aggregation module that transforms network flows into transactions by combining the HDBSCAN algorithm, enabling the detection of IoT botnet from a higher-dimensional perspective. Firstly, flow data is pre-grouped based on source and destination IP addresses, and each group is divided into flow points according to a time window. These flow points are then aggregated into transactions using the HDBSCAN algorithm. This approach effectively addresses the issue of time synchronization for flow data and enables the aggregated behavior to reach a collective level, facilitating the differentiation between normal traffic and botnet traffic from a transactional perspective. Finally, we extract feature fingerprints from these transactions and convert them into fingerprint images, which are then subjected to classification using convolutional neural network (CNN) for detection purposes. In the evaluation experiments conducted on publicly available datasets, FC-Bot achieved an impressive F1 value of 95.04%, demonstrating the effectiveness of our proposed solution. Compared to four advanced detection methods, FC-Bot exhibited a low false alarm rate of only 3.06% and achieved a maximum improvement of up to 4.30%, indicating its superior performance. Additionally, in real-world scenario evaluations, FC-Bot achieved an F1 value of 92.59%, providing a lightweight and effective detection solution for the IoT botnet.

通讯作者: 邹福泰, 博士, 高级工程师, 硕士生导师, Email: zoufutai@sjtu.edu.cn。

本课题得到国家重点研发计划项目(No. 2020YFB1807500)的支持资助。

收稿日期: 2023-03-21; 修改日期: 2023-05-29; 定稿日期: 2026-04-09

Key words internet of things; botnet; lightweight; stream aggregation; feature fingerprint

1 引言

物联网(Internet of Things, IoT)已经步入我们的生活^[1]。随着物联网的发展, 通信能力不再仅限于移动设备, 而是扩展到与人类共存的所有事物。全球移动通信系统协会(Global System for Mobile Communications Association, GSMA)预测^[2], 2025 年全球物联网设备(包括蜂窝及非蜂窝)联网数量将达到约 252 亿个。

物联网设备数量的几何级数激增也为物联网攻击提供了有利条件。根据 Straits Research 的报告, 2022 年全球物联网安全支出为 117 亿美元, 预期到 2031 年将达到 1223 亿美元^[3]。

根据美国网络与安全企业 Fortinet 发布的《2022 下半年度全球威胁态势研究报告》^[4], 2022 年 11 月僵尸网络的活动量超过 2.70 亿次, 而 12 月则达到 4.99 亿次, 活动量增长了 85%。攻击者通常利用僵尸网络这种被感染设备构成的网络来进行各种网络攻击^[5]。这些受感染的设备可能是台式机、笔记本电脑、智能手机或物联网设备。僵尸网络为攻击者提供了一个大规模的攻击平台, 一旦大量设备感染了僵尸程序, 攻击者就可以通过控制僵尸网络中的每个僵尸主机, 向它们下发指令并协同进行攻击。这些攻击的危害范围从散布垃圾邮件^[6]到对域名解析服务器进行攻击以使其瘫痪^[7], 危害程度不一。

近年来, 僵尸网络仍是最具破坏性的攻击形式。在 2017 年 5 月 19 日, 黑客发起了一场全球性的僵尸网络攻击, 针对运行微软 Windows 操作系统的机器。尽管攻击在被发现后的几天内得到了阻止, 但它仍然造成了巨大的破坏和中断, 影响了全世界 150 个国家的 30 多万台计算机^[8]。因此, 研究和检测僵尸网络攻击具有重要意义^[9]。

迄今为止, 许多研究^[10-11]提出了基于机器学习的方法来检测僵尸网络, 但这些方法的优异性能仅适用于特定的数据集。一些研究者还结合了深度学习技术。例如, Sriram 等人^[12]使用深度神经网络从流量中提取特征, 并分析整个有效载荷数据以检测僵尸网络。然而, 这种方法的可解释性较弱, 缺乏对细节的描述。Hussain 等人^[13]训练了 ResNet-18 模型来检测僵尸网络的扫描攻击和 DDoS 攻击, 取得了较好的检测效果, 但其检测范围存在局限性。还有一些研究者尝试使用人工神经网络^[14-15]和循环神经网络^[16], 但这些方法的计算复杂度较高。

另一方面, 由于僵尸网络是一个涉及互联网范围的问题, 跨越多个管理域, 因此需要一个覆盖互联网规模的解决方案。然而, 针对如此大规模的数据进行实时处理将面临巨大挑战^[17]。在这种情况下, NetFlow 数据^[18]提供了有价值的思路。NetFlow 数据不仅适用于大型网络, 而且包含丰富的信息, 非常适合于网络性能分析。此外, 它具有轻量级的特性, 有利于对大规模数据进行处理。

本文提出了一种名为 FC-Bot(Flow Point Clustering, FC)的基于流点聚合的轻量级物联网僵尸网络检测方法, 其包括四个组成部分。首先, 对收集到的流量处理获得 NetFlow 数据, 实现兼顾隐私保护和轻量级的考虑。随后, 设计流聚合模块, 先将流数据依据时窗大小切割为流点再将流点聚合, 聚合后形成物联网僵尸网络或正常流量的行为模式, 并命名为事务。这里的事务指僵尸机或正常物联网设备紧密相连的一系列行为动作, 如多台僵尸机对指定 IP 发起 DDoS 攻击, 或无线打印机接收文件并打印。其次对获得的事务提取指纹特征并转化为指纹图像。最后使用卷积神经网络对指纹图像检测分类。

其中, 流聚合与基于事务检测分类的灵感来源于对物联网僵尸网络事务共享特征的观察, 这在 3.2 节有进一步详细的描述。首先, 僵尸网络控制流量为了隐藏和绕过入侵检测系统, 连接往往十分短暂, 数据包更少, 僵尸网络控制事务相比较小。其次, 恶意软件存在代码重用, 僵尸网络事务存在相似性, 而正常事务更为复杂多变。最后, 由于僵尸网络流量中大部分是 DDoS 攻击流量, DDoS 攻击在短时间内突发大量的数据包, 且这些数据包较小, 与正常事务显著不同。

本文的主要贡献如下:

1) 本文提出了一种名为 FC-Bot 的基于流点聚合的轻量级物联网僵尸网络检测方法。NetFlow 具有轻量级的特性, 数据易于收集整理, 且过程中不会访问、读取或记录任何携带 IoT 内容的数据包有效负载。保护隐私的同时, 更适用于海量流量的真实场景。

2) FC-Bot 创新性地设计了流聚合模块。首先将流数据依据时间划分并切割为流点, 有效解决了时间的统一问题; 随后将流点聚合, 使聚合后的行为达到集体性级别, 并将其命名为事务, 通过识别正常事务与僵尸网络事务实现物联网僵尸网络检测。

3) FC-Bot 通过提取事务特征指纹并将其转化为指纹图像, 然后利用卷积神经网络进行分类检测。在

公开数据集上的验证结果表明, FC-Bot 取得了 95.04% 的 F1 值, 证明了其在物联网僵尸网络检测方面的有效性。且与目前先进的检测方案相比, FC-Bot 具有最低的误报率仅为 3.06%。此外, 在真实开放的物联网场景测试中, FC-Bot 能够发现隐藏其中的僵尸网络, 取得了 92.59% 的 F1 值, 进一步验证了 FC-Bot 的检测能力。

2 技术背景

2.1 NetFlow

NetFlow^[18] 是一种最早由思科公司提出和实施的网路协议, 主要应用于路由器和交换机等网络设备上。它的作用是捕获网络流量并将其总结为网络流的集合, 然后将这些信息转发给网络流

量收集器。NetFlow 记录了流的进站和出站信息, 包括流的起始时间、终止时间、数据包数量和字节数等。NetFlow 记录并不涉及访问、读取或记录任何携带数据包内容的有效负载, 因此它不会包含请求和接收双方的具体内容数据, 如文件的上传和下载以及物联网中的交互金额等敏感信息。换句话说, NetFlow 记录的内容是不可知的, 这为保护隐私提供了一定的保障。因此, 在使用 NetFlow 进行物联网僵尸网络检测时, 无须事先了解数据包的具体内容。

图 1 展示了本文将会用到的 NetFlow 记录的字段, 并提供了一些示例, 包括开始和结束时间戳、使用协议、源 IP 地址、目的 IP 地址、源端口、目的端口以及传输的字节数和数据包数。

Start Time	End Time	Protocol	Src IP	Dst IP	Src Port	Dst Port	bytes	pkts
8871.787858	0.157152	TCP	192.168.150.201	192.74.225.118	56589	6003	128	2
59118.00032	4.990491	TCP	202.120.217.203	120.77.85.106	1935	31344	1377317	1120
43355.29232	4.990341	UDP	10.40.59.175	27.115.94.146	51831	40253	197801	339

图 1 本文使用的 NetFlow 字段
Figure 1 NetFlow fields used

2.2 物联网僵尸网络

物联网可以被定义为实现人与物、物与物之间通信的网络^[19]。随着物联网设备的大规模部署, 越来越多设备的安全漏洞被揭示出来, 为各种类型的攻击敞开了大门。其中, 物联网僵尸网络是一种传播速度快、影响广泛的攻击形式。表 1 对比了传统僵尸网络和物联网僵尸网络的特点。在物联网中, 当一个易受攻击的设备遭到攻击时, 它会成为一个僵尸主机(bot), 并加入到庞大的僵尸网络中。攻击者可以通过命令与控制服务器远程访问和控制这些僵尸主机, 使它们执行恶意活动, 例如发起 DDoS 攻击或进行加密货币挖矿等行为。物联网中的僵尸网络规模庞大, 给网络安全带来了严重的威胁。

物联网设备具备处理能力, 部分设备能够像简化版的专业计算机一样进行工作。然而, 这些设备通常缺乏对计算机安全的意识, 为物联网僵尸网络及其他威胁提供了良好的滋生环境。表 2 对比了传统网络和物联网网络中的设备特点。从表中可以观察到, 物联网拥有庞大的设备数量, 并且设备维护不足, 这进一步增强了物联网僵尸网络的活跃程度, 超过了传统僵尸网络的影响范围。

2.3 HDBSCAN 算法

HDBSCAN(Hierarchical Density Based Spatial Clustering of Applications with Noise)^[20] 是一种基于密度的聚类算法, 适用于具有噪声的应用程序空间聚类任务。

HDBSCAN 算法包括五个步骤, 首先, 通过密度/稀疏度进行空间转换, 从稀疏带噪声的数据中找到密度更高的簇。其次, 构建基于加权距离图的最小生成树, 通过逐步合并距离最近的簇来形成多层的树状图。然后, 构建组件之间的层次簇结构, 将结果表示为树

表 1 传统僵尸网络与物联网僵尸网络的对比
Table 1 Traditional Botnets vs. IoT Botnets

	传统僵尸网络	物联网僵尸网络
结构	C&C 和 P2P	C&C 和 P2P
生命周期	包含四个阶段: 初始感染, 与 C&C 服务器连接, 发起攻击, 攻击后	与传统僵尸网络相同
使用设备	计算机, 服务器	照相机、智能冰箱、智能空调等
检测机制	容易	困难
影响	范围有限, 影响较高	传播速度快, 规模大, 影响极大

表 2 传统网络和物联网网络中的设备对比
Table 2 Comparison of devices in traditional and IoT networks

	传统网络设备	物联网设备
资源能力	非常低的处理能力, 更小的内存	高资源设备可用
僵尸网络检测时间	用户可以容易地监控设备的较少异常行为	物联网设备的图形用户界面有限, 用户与设备的交互更少, 僵尸网络在很长一段时间内不会被注意
设备数量	远少于物联网设备, 一般以千为单位	设备数量以十亿计
构建的安全级别	高	非常低或不部署
传播/影响	低	设备数量庞大, 互联网连接速度快, 损失也更多
用于攻击的设备可用性	取决于网络	全天候可用, 是更稳定的攻击来源

状图的形式。接下来, 使用最小簇大小的压缩层次聚类, 将树状结构压缩为更小的树, 同时使每个节点包含更多的数据点。最后, 利用压缩后的生成树进行分类。

HDBSCAN 算法不受聚类形状和密度的限制, 且在较少超参数的情况下能够获得更鲁棒的聚类结果^[21]。该算法引入的分裂度量方式可以更好地反映数据点的密度特征。

3 FC-Bot

本文提出一种名为 FC-Bot 的基于流点聚合的轻量级物联网僵尸网络检测方案。如图 2 所示, 是 FC-Bot 的框架图, 其包括四个组成部分。1) 流量预处理, 通过 Argus 指令处理 Pcap 数据包获得 NetFlow 数据, 并删除零字节和零持续时间记录等。2) 流聚合, 这也是本文的核心模块, 使用流聚合模块, 先对 NetFlow 数据进行切割, 归一化数据统一时间特性, 获得流点; 紧接着结合 HDBSCAN 算法将多个流点聚集在一起形成僵尸网络事务或正常行为事务。3) 指纹生成, 对获得的事务提取特征指纹并将其转为 RGB 指纹图像。4) 卷积神经网络检测分类, 最后使用卷积神经网络对指纹图像进行训练检测, 得出分类结果。

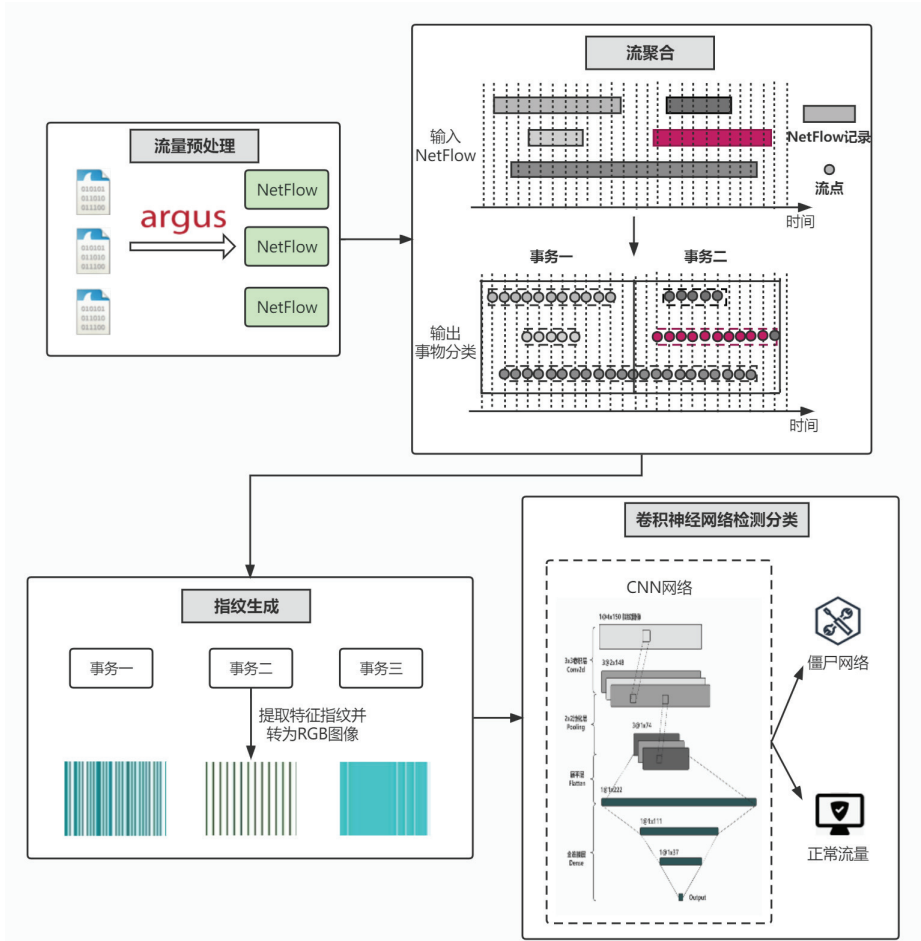


图 2 FC-Bot 的框架图
Figure 2 The frame of the FC-Bot

3.1 流量预处理

本文收集整理了物联网僵尸网络数据集和现实物联网设备良性流量, 随后需要将其处理得到 NetFlow 数据。当然也可以选择直接获取 NetFlow 格式的数据, 但是一般来讲各大学校或实验室提供数据的格式优先还是选择 Pcap, 所以需要对其进行进一步处理。本文选取 Argus 对其处理, Argus 是一个网络流系统, 支持各种流技术, 包括 NetFlow, Jflow, Qflow, Kflow 等, 其可以高效、迅速地处理数据包并生成摘要网络流数据, 一般来说处理 10GB 的数据包耗时小于 5 秒钟。

使用 Argus 指令处理后获得 NetFlow 数据, 进一步删除零字节、零持续时间和无关协议(如 ICMP)的 NetFlow 记录。最后选取其中能体现僵尸网络的字段, 如表 3 所示, 作为后续特征指纹的提取和指纹图像生成的基础。

表 3 从 NetFlow 中选取的字段
Table 3 Fields selected from NetFlow

字段名称	描述
StartTime	流的发送时间
Dur	流持续时间
Proto	流使用的协议
SrcIP	源 IP 地址
DstIP	目的 IP 地址
Sport	源端口
Dport	目的端口
TotBytes	流包含的字节数
TotPkts	流包含的数据包数

3.2 基于流聚合的事务生成

在网络流预处理完成后, 我们意识到单独的流数据往往不足以区分良性用户行为和 IoT 僵尸网络行为, 因为它们都具有集体性质。为了解决这个问题, FC-Bot 引入了流聚合模块, 以检测流的集体行为, 并从集体事务的角度区分它们的不同行为模式。

流聚合模块将由相同事务生成的所有 NetFlow 记录聚集在一起, 以便我们可以比较 IoT 僵尸网络事务和正常事务之间的差异。在本文中, 我们将事务定义为由紧密相邻的僵尸机或真实用户执行的一系列动作。举例来说, 一个任务可以是小米音响查询特定歌曲并开始播放, 或者是一组僵尸机器对同一目标机器发起 DDoS 攻击。

3.2.1 基于事务区分的灵感

从定义和比较事务特征来区分物联网僵尸网络流量和正常流量的灵感来源于以下三点对僵尸网络

事务共享特征的观察。

1) 在僵尸网络中, 存在僵尸网络控制流量。这种流量是指僵尸机与命令与控制服务器(Command and Control server, C&C)之间建立连接通道, 用于发布命令或传递信息。为了规避入侵检测系统的检测并确保隐蔽性, 这些连接通常具有短暂的持续时间, 数据包数量较少且流量较小。过长时间的连接可能会引起检测系统对僵尸机过度的关注^[17]。在我们收集的僵尸网络数据集中也验证了这一点。如表 4 所示, 将本文收集的 Antiy 数据集中的僵尸网络控制流量根据包含数据包数量划分(详细数据集的介绍见 4.1 节)。可以发现绝大多数僵尸网络控制流量的数据包数量小于 10 个, 而大部分的数据包数量不超过 5 个。因此, 僵尸网络控制事务包含的动作应该是小且短暂的。

表 4 僵尸网络控制流量依据包数量划分

Table 4 Divide botnet control traffic according to the number of packets

	包数量大于 10	小于等于 10	大于 5	小于等于 5	总数
流条目数	31	185	3950	4166	
占比	0.74%	4.44%	94.82%	100%	

2) 一方面, 由于代码重用, 僵尸机往往共享相似甚至几乎相同的恶意程序; 另一方面, 在典型的 m 命令与控制协议中, 命令的种类是有限的。因此, 僵尸网络流量的大小取值范围是有限的, 这在事务中体现为存在相似性。相比之下, 良性流量由于访问方式和请求内容的多样性, 使得流量的大小取值范围极广, 没有规律可循, 事务更加复杂多变。

3) 僵尸网络流量中 DDoS 攻击流量所占比例极大。DDoS 攻击会在短时间内突发生海量的数据包, 单位时间内的包密度极高, 这与良性流量明显不同。因此, 通过这一特征可以明显区分僵尸网络的 DDoS 攻击事务和良性事务。

3.2.2 FC-Bot 流聚合

有赖于上述灵感, FC-Bot 引入流聚合模块, 首先将流切割成多个流点, 以解决起始和终止时间的统一问题。随后结合 HDBSCAN 算法, 将这些流点聚合到不同事务中。每个事务包含了来自不同流的多个流点, 表示一系列行为动作。通过识别这些事务来检测物联网僵尸网络。图 3 展示了一个流聚合的示例。具体实现如下所述。

1) 流子集划分

针对初始的流数据集 F , 包含了经过预处理的所有 NetFlow 流数据 f 。我们将数据集 F 根据流 f 的

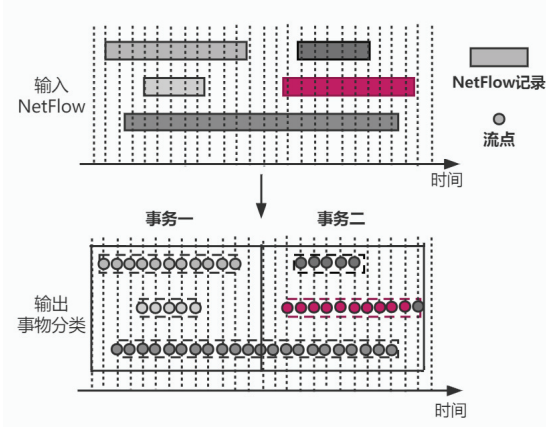


图 3 FC-Bot 流聚合示例

Figure 3 FC-Bot flow clustering example

IP 地址进行划分, 得到不同的子集 F_{set} 。每个子集表示一个源 IP 地址或目标 IP 地址所发送或接收的流量。例如, 具有相同源 IP 地址的 F_{set} 可能表示小米音响搜索某首歌曲并播放的流量, 而具有相同目标 IP 地址的 F_{set} 可能表示该目标 IP 地址同时受到多个僵尸机器发起的 DDoS 攻击。因此, 不同的 F_{set} 可能包含相同的流数据。

2) 流点切割

对于每个 F_{set} 中的流数据, 如图 3 所示, 本文将其持续时间根据预设的切割时窗 T_w 分成多个长度相等的时间段, 并为每个时间段定义一个流点 p 。每个流点拥有四个属性, 分别是上行流和下行流的每秒位数(bit per second, bps)、上行流和下行流的每秒数据包数(packet per second, pps)。

3) 流点空间变换

定义两个流点之间的互可达距离 $d_{\text{mreach-k}}$ 如公式(1)所示, 其中 d 代表欧几里得距离, $\text{core}_k(p)$ 表示流点 p 与第 k 个最近邻样本点的距离, 即核心距离, 核心距离的计算公式如(2)所示。

$$d_{\text{mreach-k}}(p_1, p_2) = \max\{\text{core}_k(p_1), \text{core}_k(p_2), d(p_1, p_2)\} \quad (1)$$

$$\text{core}_k(p) = d(p, N^k(p)) \quad (2)$$

互可达距离表示了流点 p_1 和 p_2 的核心距离与两个样本点之间距离的最大值。通过使用互可达距离来表示流点之间的距离, 我们实现了流点空间的变换。这种变换使得密集区域的样本点之间的距离不受影响, 而稀疏区域的样本点与其他样本点之间的距离被放大。这样一来, 聚类算法在非密集区域具有更好的鲁棒性。

4) 构建最小生成树

为了在流点构成的密集数据集中实现聚类, 即

事务的划分, 我们使用上一步得到的流点的互可达距离作为距离度量。由于密集区域的定义是相对的, 且不同的聚类具有不同的流点密度, 我们将流点构成的密集数据集看作一个加权图。在该图中, 流点作为顶点, 任意两个流点之间的边的权重为它们的互可达距离 $d_{\text{mreach-k}}(p_1, p_2)$ 。现在, 考虑一个阈值 γ , 我们从一个较高的值开始逐步降低 γ , 并删除所有权重超过 γ 的边, 以实现图的分裂。整个过程中, 图的变化为从完全连接到完全断开。图的分裂结果可以简单地理解为使用当前阈值 γ 对加权图进行划分得到聚类。

然而上述操作的时间复杂度为 $O(n^2)$, 为了降低复杂度, 我们可以使用图论中的最小生成树算法来实现上述过程。具体来说, 我们使用 Prim 算法构建互可达距离的最小生成树。该算法逐步从连接当前树节点与树外节点的所有边中选择权重最低的边, 并将其添加到树中。如图 4 所示, 它展示了构建最小生成树的一个示例, 其中边的颜色深浅表示了节点之间的相互可达距离。在最小生成树中删除任何边都会导致树被分割为两个子树, 而且无法找到权重更低的边来连接这断开的两个子树。

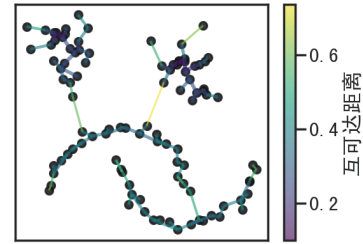


图 4 构建最小生成树的示例

Figure 4 Instance of building the Minimum Spanning Tree

5) 聚类二叉树与剪枝

我们对最小生成树中的所有边进行递增排序, 并依次选择每条边, 将这条边连接的两个子树进行合并, 从而最终得到一个聚类二叉树, 如图 5 所示。在这个二叉树中, 每个节点代表一个流点簇, 最顶

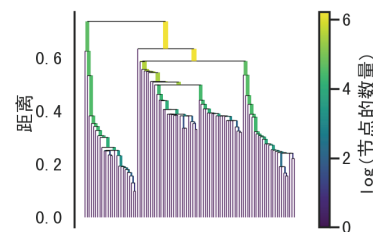


图 5 聚类二叉树的实例

Figure 5 Instance of Clustering Binary Tree

层的根节点表示整个流点数据集。二叉树中每个节点的两条边表示当前子树的分裂,即去掉当前最小生成树中权重最高的边,使得树进一步分裂成两个子树。每次分裂都对应着一个距离,即所去掉的边的权重。

接下来,我们对聚类二叉树进行剪枝压缩,以去除散点,使得聚类树的每个节点都成为适当大小的流点簇。我们定义最小簇大小为 *Minsize*,并自上而下遍历聚类树。如果一个节点的两个子节点中的流点数都大于 *Minsize*,我们将其视为可拆分的流点簇,并从这两个子节点继续向下遍历,同时保留拆分前的父节点,该父节点包含这两个子节点的所有流点。如果只有一个子节点的流点数大于 *Minsize*,我们将该子节点替换当前的父节点,并继续向下遍历。如果两个子节点的流点数都小于 *Minsize*,我们删除这两个子节点。图 6 展示了聚类二叉树经过压缩剪枝后的结果,其中 λ 表示距离的倒数。

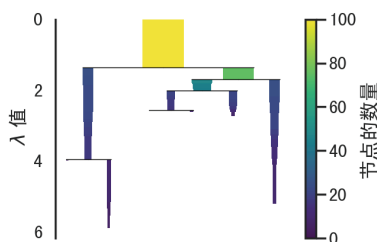


图 6 聚类二叉树压缩剪枝的结果

Figure 6 Results of Clustering Binary Tree compression pruning

6) 提取聚类结果

通过树状图的压缩操作,成功消除了树中的稀疏节点。接下来的目标是进一步合并最相近的节点(即流点簇),以获取最终的聚类结果。为此引入距离的倒数作为参数 λ 。对于每个流点 p ,定义 λ_p 表示当流点 p 从流点簇中分离时,对应断开边的长度倒数。同时,我们定义 λ_{birth} ,代表当前节点分裂时对应断开边的长度倒数,表征该流点簇与兄弟簇之间的紧密度。 λ_{birth} 的值越大,表示簇的稳定性越高,即簇内散点较少。根据这些定义,我们可以得到每个子集的稳定性公式,如公式(3)所示。稳定性值越大,表示簇中的散点越少。

$$S_{\text{cluster}} = \sum_{p \in \text{cluster}} (\lambda_p - \lambda_{\text{birth}}) \quad (3)$$

最后寻找稳定性最大的结果也就是获得聚类的结果。首先初始化簇,令每个叶节点都为初始簇。随后自下而上遍历整棵树,如果当前节点的稳定性小于两个子节点的稳定性总和,那么将该节点的稳定

性设置为其子节点的稳定性之和;如果当前节点的稳定性大于两个子节点的稳定性总和,那么将当前节点定为一个簇,并且删除所有子节点。最终可以获得聚类簇,每个聚类簇表示一个事务。

通过流聚合模块,将 NetFlow 数据转为不同的事务,使其达到集体行为级别,利于进一步区分正常事务与僵尸网络事务。

3.3 事务指纹生成

当从流聚合模块获得后事务,每个事务可能包含不同数量的 NetFlow 流记录,且其信息不统一。为了使不同的事务能够直接相互比较,必须从每个事务的聚合流中定义、提取和规范化特征。

FC-Bot 为每个事务生成一个特征指纹,通过设计一种数据融合方法,从事务中导出一个 $f \times N$ 矩阵,并将这个矩阵作为事务的指纹。这里, N 是指事务的时间窗口内包含的时间段的数量,时间窗即事务包含的所有流中从最早开始时间跨越到最晚结束时间的时长, f 是提取的事务特征的数量。

由于本文将流切割为流点并将流点聚合,聚合的事务可能包括了不同流的流点,且一条流的流点也可能属于不同事务。因此传统的包级别特征(如前向包的平均时间)、流级别特征(流处于活动状态的最长时间)不再适用。表 5 显示了本文提取的 $4 \times N$ 事务特征指纹矩阵。包括上行流和下行流的每秒位数 bps、每秒数据包数 pps 这 4 类特征。由 3.2 节可知,每个事务表示一个源 IP 或目标 IP 所发送或接收的数据,因此上行对应发送,下行对应接收。另外,由于事务的任何时间段都可以有多条流记录,因此该时间段的 bps 和 pps 值应该分别是该时间段内所有流点的 bps 和 pps 值的总和。

表 5 事务指纹提取的特征

Table 5 Features of transaction fingerprint

特征	描述
上行 bps	每秒发送比特数
上行 pps	每秒发送数据包数量
下行 bps	每秒接收比特数
下行 pps	每秒接收数据包数量

事务特征指纹矩阵一旦生成,需要将其规范化。本文使用激活函数 $\tanh$ 将矩阵中的每个值映射到 0 到 255 之间的一个数字上。但是,由于在实际中发现每个事务的 bps 和 pps 值往往过大,前者常常以数万计,后者常常以数千计,当使用常规的 $\tanh$ 函数处理时,我们发现最后的结果分布极其集中,大部分在 0.9 以上,不能体现差异性。因此,如公式(4)所示,

本文将 $\tanh$ 函数修改为 $\tanh'$ 函数, 通过添加幂次参数 ω 修正传入参数 x 过大的影响。经过 $\tanh'$ 函数处理后, 数据被映射到 $[0,1]$ 区间, 随后乘以 255 作为归一化的结果。

$$\begin{aligned} \tanh' x &= \frac{e^{\omega x} - e^{-\omega x}}{e^{\omega x} + e^{-\omega x}} \\ x_{\text{out}} &= 255 \times \tanh' x \end{aligned} \quad (4)$$

当传入参数 x 较小时增大 ω , 反之当传入参数 x 较大时减小 ω , 即增大 x 比较小时对归一化结果的影响, 减小 x 比较大时对归一化结果的影响。特别地, 本文通过学习了现实物联网流量 170 GB 数据集和僵尸网络数据集的 bps 和 pps 取值分布后, 对于 bps 值, 选取 ω 为 0.000001, 对于 pps 值, 选取 ω 为 0.001。

现在指纹矩阵中每一个数据均落在 0 到 255 的区间, 随后我们分别处理上行流量和下行流量。对于时长为 N 的事务矩阵, 本文设定其长度为 N 个像素, 对于像素 $I(I=1, \dots, N)$, 其 RGB 值表示为 $(R, G, B)=(\text{bps}, \text{pps}, 0)$, 其余空白像素使用 $(255, 255, 255)$ 填充。这将较小的 bps 或 pps 值映射到较暗的颜色。这样可以获得像素均为 $2 \times N$ 的上行流量和下行流量指纹图像, 对于每一列, 其 RGB 的值是相同的。

图 7 所示是三个指纹图像的示例, 图 7(a)是被感染的 bot 与僵尸网络 C&C 服务器定期联系的上行流量图像, 可能是上传 bot 信息报告状态; 图 7(b)是多台僵尸机对目标 IP 发起 DDoS 攻击产生的流量图像, 大部分僵尸网络图像属于此类; 图 7(c)来源于物联网设备对某域名访问的下行流量, 其连续性较为明显且分布没有规律。从图像中可以较为清楚地看到物联网僵尸网络的事务指纹和真实用户的事务指纹在颜色深度、频率、规律性和密度方面存在不同的模式。

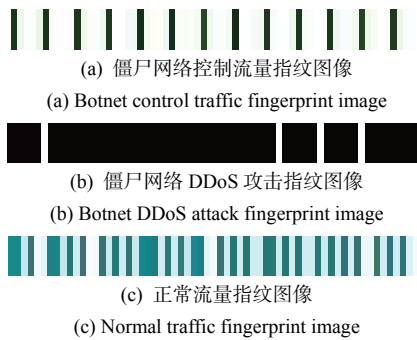


图 7 生成的指纹图像

Figure 7 The generated fingerprint image

3.4 卷积神经网络模型

近年来, 随着机器学习和深度学习的快速发展, 越来越多的研究开始利用人工智能技术来检测物联

网僵尸网络。Alkahtani 等人^[22]构建了一种基于 CNN-LSTM 模型的检测方法, 在九种商业物联网设备中成功检测出僵尸网络, 平均准确率超过 88%。Aydin 等人^[23]提出了一种基于卷积神经网络的在线网络入侵检测系统, 能够快速准确地检测物联网僵尸网络产生的 DDoS 流量。

本文使用 Keras 和 TensorFlow 来实现卷积神经网络(Convolutional Neural Networks, CNN)的架构。如图 8 展示了卷积神经网络的架构。输入获得的 $4 \times N$ 指纹图像, 随后经过 3×3 卷积层和 2×2 池化层后和展平层得到一维向量, 最后经过数个隐藏层得到分类结果。每一个隐藏层后接一个 Dropout 层且丢弃率设为 0.5, 用来防止过拟合。隐藏层使用 ReLU 作为其激活函数, 模型可以快速收敛; 输出层使用 Softmax 函数来输出二分类的结果; 另外, 用随机梯度下降优化算法训练 CNN 模型, 用反向传播算法更新神经网络。

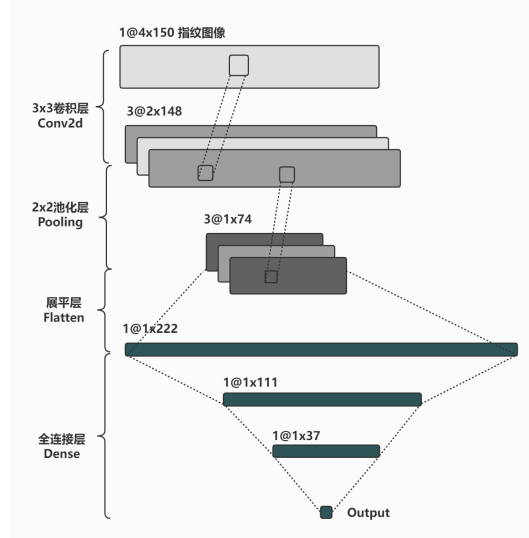


图 8 卷积神经网络架构图

Figure 8 Convolutional neural network architecture

4 实验及分析

4.1 数据集

4.1.1 数据集构成

僵尸网络数据集: 本文收集整理了三部分僵尸网络流量, 既包括公开数据集——IoT-23^[24]数据集和 CIC-IDS2017^[25]数据集, 又包括现实数据集——Antiy 数据集。

IoT-23 数据集为 2018 年至 2019 年捷克 CTU 大学捕获的 23 个不同场景的物联网流量, 其中包括 20 个恶意流量和 3 个正常流量。经过筛选, 本文选择其中与僵尸网络流量相关的恶意流量, 共计 18.5GB。

CIC-IDS2017 数据集是加拿大网络安全研究所在 2017 年 7 月第一周捕获的流量, 其中包含良性流量和最新的常见攻击, 如 Web 攻击、渗透测试、僵尸网络和 DDoS 等。本文选择 Friday 收集到的流量, 其中包含了 Botnet ARES 流量, 可用于补充僵尸网络流量数据, 共计 8.3GB。

Antiy 数据集为安天公司从 2020 年 10 月到 2021 年 12 月通过世界各地部署的蜜罐从现实开放 IoT 场景中捕获到的僵尸网络流量, 共计 7.2GB。其中, 包含了多种 IoT 僵尸网络, 覆盖较广, 数据更为真实。可有效补充僵尸网络流量的不足。

需要注意的是, IoT-23 数据集和 CIC-IDS2017 数据集包含良性流量, 在通过 Argus 对其处理获得 NetFlow 数据后, 需要过滤出僵尸网络相关数据。而 Antiy 数据集中包含的都是僵尸网络相关流量, 无须对获得的 NetFlow 进行过滤。

良性数据集: 本文整理的良性数据集为从现实物联网设备, 如安全系统、智能电器、智能电视和可穿戴健康装置等收集的流量, 总量约为 170GB。

如表 6 所示, 对本文数据集来源、流量大小和 NetFlow 条目数进行了总览。需要强调的是, 对于 IoT-23 数据集和 CIC-IDS2017 数据集获得的 NetFlow 数据还需根据 IP 地址过滤出与僵尸网络相关条目。另外, 对所有 NetFlow 数据需要删除零字节、零持续时间或不相关协议(如 ICMP)的记录。

表 6 数据集总览
Table 6 Dataset overview

	数据集来源	流量大小	NetFlow 条目数
僵尸网络数据集	IoT-23	18.5GB	621845
	CIC-IDS2017	8.3GB	159711
	Antiy 数据集	7.2GB	32389
良性数据集	现实 IoT 设备, 如智能电器、安全系统	170GB	3736462

4.1.2 数据集划分

随后划分训练集、验证集和测试集。本文提取 IoT-23 数据集和 CIC-IDS2017 数据集的僵尸网络流量部分并混合 Antiy 数据集的僵尸网络流量补充恶意流量的不足, 接着按照 8:1:1 的比例划分为训练集、验证集和测试集。现在训练集、验证集和测试集中只有恶意流量, 本文使用从物联网设备收集整理的良性数据集来补充正常流量, 构造模拟环境。其中, 训练集和验证集中恶意流量和良性流量的数量比例为 1:3; 考虑到现实场景恶意流量较少, 为了更真实地模拟现实环境, 测试集恶意流量占比为 10%。

在使用训练集和测试集训练好模型后, 本文又特意收集了另一段时期的 Antiy 现实物联网流量, 其为 2022 年 1 月到 3 月安天公司在物联网设备捕获的流量。除了大部分的正常流量, 其中还包含僵尸网络攻击、渗透测试和 DDoS 攻击等, 直接从现实场景捕获, 更为真实。本文将其作为现实环境直接用于模型测试, 进一步检验 FC-Bot 在现实场景中的检测效果(见 4.7 节)。需要注意的是, 本文的目标是检测僵尸网络, 因此除了僵尸网络流量外其余流量均标记为良性。

4.2 对比模型选择

为了充分评估 FC-Bot 的性能, 我们选取了目前先进的 4 种基于流的僵尸网络检测方法进行比较, 简要描述如下。为了进行公平的比较, 所有的方法都进行了微调, 以在用于评估的数据集上达到较高的精度。

Kitsune^[26]: Kitsune 是一种基于数据包级别的异常检测系统。它利用数据包捕获模块和特征提取模块从 5 个时间跨度对经过网络节点的数据包提取 115 维特征。随后, 这些特征被映射为低维表示, 并提供给自动编码器进行处理。在训练阶段, 自动编码器学习正常行为在子空间中的表示, 并在检测阶段对异常数据进行报警。本文重放收集到的僵尸网络数据包, 使用 Kitsune 来检测异常僵尸网络流量。

Whisper^[27]: Whisper 是一种基于频域分析的恶意流量检测系统。它将数据包特征序列编码为向量, 并将其分割为帧, 然后对每一帧进行离散傅里叶变换, 以提取流量的频域序列信息。在训练阶段, Whisper 利用 K-Means 算法计算良性流量的频域特征聚类中心。在检测阶段, 系统将离群值判定为恶意流量。本文重放收集到的僵尸网络数据包, 使用 Whisper 检测异常僵尸网络流量。

1DCNN^[28]: Kundu 等人提出了一种基于 1DCNN 模型的僵尸网络检测框架。该框架利用特征提取模块从网络流量中提取了 199 个特征, 并生成相应的特征向量。随后, 他们通过对 19 种不同的 1DCNN 架构进行详尽的实验研究, 在确定了最佳模型和参数后, 将提取到的特征输入到最佳的 1DCNN 模型中进行检测和分类。

STM-DT^[29]: Hou 等人提出了一种名为自训练混合决策树(STM-DT)的网络攻击流量分类模型。该模型采用频率编码的方法, 用于提取流量数据集的特征并进行归一化, 以得到频率表示。随后, 他们构建了 STM-DT 模型, 利用少量标记数据对决策树进行训练, 并利用该模型对未标记样本进行预测。接下来,

将预测的样本与标记的样本混合, 使用混合数据来训练新的决策树, 以获得最终的分类器。本文重点关注该模型在僵尸网络检测方面的应用。

4.3 指标选择

本文采用如表 7 所示的指标, 其中, 真阳性(True Positive, TP)表示被识别为正常网络流的正常流量, 真阴性(True Negative, TN)表示被识别为恶意网络流的恶意流量, 假阳性(False Positive, FP)表示被错误识别为正常网络流的恶意流量, 假阴性(False Negative, FN)表示被错误识别为恶意网络流的正常流量。

表 7 评估所用指标

Table 7 Evaluation Metrics

指标	公式
Precision(精度)	$\frac{TP}{TP + FP}$
Recall(召回率)	$\frac{TP}{TP + FN}$
F1(F1 值)	$\frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$
FPR(误报率)	$\frac{FP}{FP + TN}$

准确率统计的是正例和反例判断正确的情况, 当类别不平衡时不能很好地反映模型的情况。而精确率、召回率和 F1 值即使在正例个数和反例个数相差极大的情况下也能反映模型是否有效。因此, 不考虑准确率这一评价指标。

4.4 FC-Bot 系统参数的配置与分析

FC-Bot 的不同模块具有多个系统参数。本文在 4.4.1 节为一些系统参数设置经验值或默认值, 并具体地研究了两个关键参数。4.4.2 节中讨论最小簇大小 Minsize 的取值, 以及 4.4.3 节中讨论指纹生成模块中的时长 N , 这也是图片长度的像素值, 并在 4.4.4 节描述使用最优参数后模型的检测效果。

4.4.1 系统参数默认值

一方面基于前沿的研究方法, 另一方面基于本文整理数据的分布以及经验研究, 设置了如下的默认参数:

1) 对于流点切割, 本文定义每个切割的时间段 T_w 长为 1 秒。这个设定是基于我们发现僵尸网络 C&C 服务器与被感染的 bot 会定时联系或发布指令, 过小的时长会导致指纹矩阵过长, 计算更复杂。该参数也是可调整的, 因为较小的时段长可以以较高的计算成本获得更精确的结果。对于持续时间小于 1 秒的流, 本文分别将其起始时间和终止时间取整, 并选择与该流更为接近的时间整数作为该流点的时间表示。

2) 对于流量聚合模块, 我们仅需对最小簇大小 Minsize 进行选择。根据网络角度的用户行为模型^[30], 用户通常不会在 10 到 20 秒内触发 NetFlow 记录, 此外本文同样从僵尸网络数据集中发现当 C&C 服务器与 bot 较频繁联系时其时间间隔常小于 20 秒。因此事务时间间隔一般在 10~20 秒, 反映在流点上即为最小簇要包含不少于 20 个流点。本文对 Minsize 取 20, 如果 Minsize 过小则一些离群点会被错误判定为事务, 如果 Minsize 过大, 则一些小型事务会被错误判定为离群点。

3) 对于事务指纹生成模块, 本文将 N 设置为 150, 也是指纹图像长度的像素, 此设定基于我们发现 C&C 服务器与 bot 较频繁联系时持续时间极少超过 150 秒。当然, 较大的 N 值将产生更准确的结果, 但需要更多的训练数据和计算。

4.4.2 最小簇大小 Minsize 的讨论

本文研究了最小簇大小 Minsize, 该参数影响模型在测试集上的 F1 值。对于流聚合模块分别选择 Minsize 为 5、10、20、30 和 40, 用训练集训练模型, 并用测试集检测模型效果。如图 9 所示显示了不同 Minsize 对应的测试集 F1 值。从中可以看到, F1 值随着 Minsize 的增大处于先上升后下降的趋势, Minsize 当 Minsize 取 20 时, 卷积神经网络在测试集的 F1 值取得最高为 0.95, 而随着 Minsize 的增大, 逐步降低。当 Minsize 过小时, 一些良性的短小流被错误聚合为事务, 而这些流的短小高速性使得其和 DDoS 攻击的流存在相似性, 造成误判进而降低 F1 值; 而 Minsize 过大时使得一些僵尸网络的控制流量被误认为离群流量, 使得遗漏部分僵尸网络事务, 造成 F1 值下降。本文最终选取 Minsize 为 20。

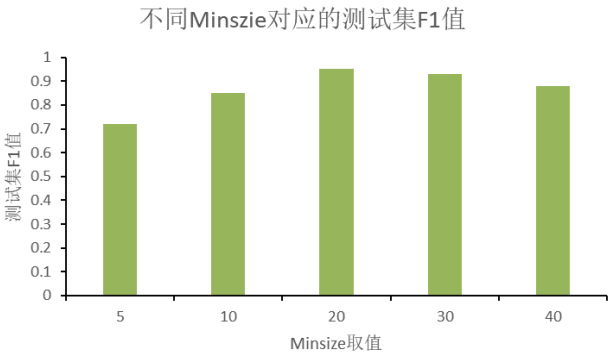


图 9 不同 Minsize 对应的测试集 F1 值

Figure 9 The F1 value of the test set corresponding to different Minsize

4.4.3 时长 N 的讨论

通过观察僵尸网络数据集我们发现僵尸网络中

最常见的 DDoS 流量往往特别紧密连贯, 且特征明显, 其对 N 取值的依赖性不强。但是对于僵尸网络控制流量, 其较为依赖 N 合理的选取。这是因为 C&C 服务器与僵尸主机较频繁联系时持续时间极少超过 150 秒, 但也有一些超过 200 秒的情况, 同样的也有一部分持续时间小于 50 秒。因此, 有必要针对指纹生成模块中的时长 N , 也是图片长度的像素值进一步讨论。

如图 10 所示, 本文针对不同的 N 值分别进行了实验验证。从结果我们可以发现, 随着 N 取值的增大, 模型的各项指标也在不断上升, 这也符合我们的预期。进一步进行分析, 可以看到在 N 由 20 增加到 50 时, 有一个较为明显的增长, 这是因为对于僵尸网络控制流量, 大部分僵尸网络的 C&C 服务器与僵尸主机联系时间要大于 50 秒, 如果 N 取 20 会将一部分事务错误地划分为两部分造成误差; 当 N 取值大于 150 时可以看出增长势头变缓, 这是因为在僵尸网络控制流量中只有极少部分僵尸网络的联系时间要大于 150 秒, 尽管 N 取值越大越有利于完整的还原事务, 但是这也会增大模型的计算负担。因此综合考虑, 本文最终选择 N 取值 150。

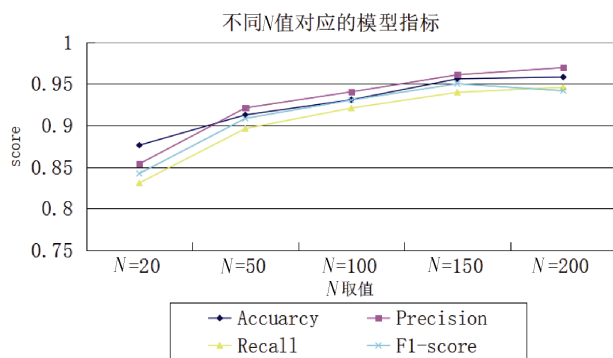


图 10 不同 N 值下模型的检测结果

Figure 10 Results of the model under different N

4.4.4 最优参数下 FC-Bot 检测结果

当使用最优参数时, FC-Bot 在训练集和验证集上的准确度曲线和损失值曲线如图 11 和图 12 所示, 另外在测试集上的结果指标如表 8 所示。可以发现当使用最优参数时, FC-Bot 在测试集取得了 96.11% 的精确率和 94.00% 的召回率, 以及 95.04% 的 F1 值。这表明模型可以识别并正确分类绝大部分僵尸网络流量, 具有较好的表现结果。

误报率在 3% 左右, 进一步分析误报情况有以下几点原因。

(1) 物联网僵尸网络数据集来源的复杂性

一方面 IoT-23 数据集包含了 CTU 大学在 2019

年捕获的 Mirai 僵尸网络流量, 这些流量均是从真实的 IoT 硬件设备中捕获, 是真实的网络行为。另一方面, CIC-IDS2017 数据集来源于 2017 年某一周捕获的流量, 其类型为 ARES 僵尸网络, 但是这个僵尸网络是模拟出的数据, 混杂了部分良性的流量, 数据集只是做到了接近真实而不能说其为真实的流量。本文选择的 Antiy 数据集来源于安天公司通过蜜罐捕获的僵尸网络流量, 其包括较多的僵尸网络类别, 当然也有部分僵尸网络在捕获时已经失活。尽管物

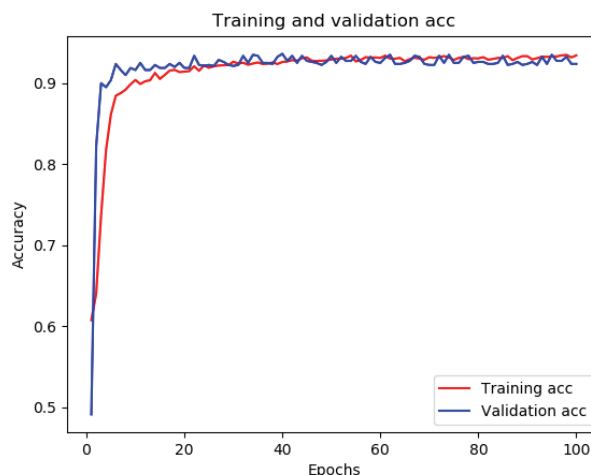


图 11 使用最优参数值时 FC-Bot 训练和验证的准确率

Figure 11 FC-Bot training and validation accuracy when using optimal parameters

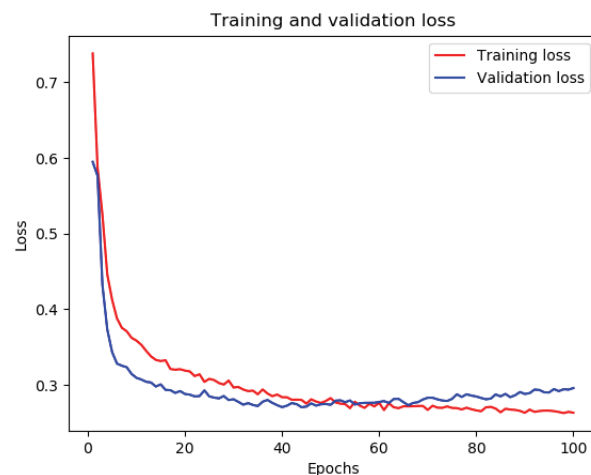


图 12 使用最优参数值时 FC-Bot 训练和验证的损失值

Figure 12 FC-Bot training and validation loss when using optimal parameters

表 8 使用最优参数值时 FC-Bot 在测试集的结果指标

Precision	Recall	F1	FPR
96.11%	94.00%	95.04%	3.06%

联网僵尸网络存在内在的相似性,但是不同类别还是存在差异。在良性流量与僵尸网络流量比例严重失衡时,为了增加恶意样本容量,本文将整理到的僵尸网络流量进行了合并,这也就导致了无论类别、真实性、僵尸网络的活性都可能存在差异,进而使得存在部分错误分类情况。

(2) 良性行为存在特殊性

同时,我们发现一部分良性流量存在类似于僵尸网络的行为,其也是在一定时间间隔内对某服务器进行了访问。进一步分析原因,可能是网络或服务器的某些问题使得无法实现访问,于是造成了频繁的请求,且由于响应包较小发送速度较快,故模型将部分僵尸网络流量看作这类行为,造成误报。

(3) 僵尸网络行为的特殊性

本文也进一步检索了僵尸网络被判定为良性流量的情况。我们发现,一些僵尸网络的 C&C 服务器在与被感染的 bot 交互时可能命令其下载或者向其传输一些恶意文件,这个过程并未由专门的恶意程序服务器完成。无论这些文件的体量,传输这些文件的数据包都会远大于用于发送指令的数据包。而这一类传输文件数据包在本文的数据集中体现较少,且在正常环境中传输文件的频率较高,所以错误地

将此类僵尸网络流量误判为良性流量。

4.5 与现有先进模型的比较

如 4.2 节所述,本文将 FC-Bot 和其他基于流的先进的僵尸网络检测方法进行对比,包括 Kitsune^[26]、Whisper^[27]、1DCNN^[28]和 STM-DT^[29]。

如表 9 和图 13 所示是本文方法与 4 种先进检测方法的实验结果对比。可以发现 1DCNN 模型取得了最高的精确率和召回率,分别是 96.74%和 95.32%,但是其拥有较高的误报率 5.15%。我们的方法在精确率、召回率和 F1 值上均取得了第二好的结果,尽管略低于 1DCNN,但是差距较小,且在误报率上有较为明显的优势。

表 9 FC-Bot 和先进模型的实验结果对比

Table 9 Comparison of experimental results between FC-Bot and advanced models

检测方法	Precision	Recall	F1	FPR
FC-Bot	96.11%	94.00%	95.04%	3.06%
Kitsune	84.98%	83.15%	84.06%	7.36%
Whisper	94.96%	92.79%	93.86%	4.17%
1DCNN	96.74%	95.32%	96.02%	5.15%
STM-DT	93.16%	92.78%	92.97%	3.58%

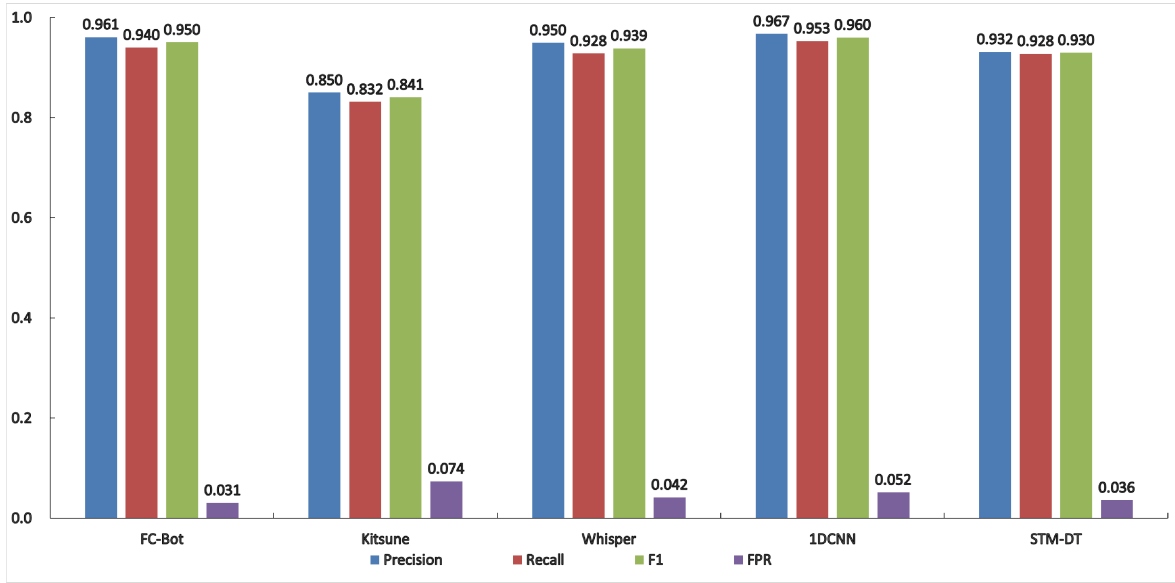


图 13 FC-Bot 和先进模型的实验结果对比

Figure 13 Comparison of experimental results between FC-Bot and advanced models

在实验对比中,数据包级别的分析方法 Kitsune 取得了相对较差的结果,精确率和召回率偏低。分析发现 Kitsune 忽略了僵尸网络的流级别信息,仅基于包特征的异常检测无法检测部分伪装为良性流量的恶意流量。Whisper 提出了一种频域特征的方法,使用频域特征表示流量序列信息,取得了较好的结果。

但是其本质是基于异常的检测,在一种良性流量数据集获得的良性聚类中心存在局限性,导致在本文数据集上部分良性流量因为离群而错误分类为僵尸网络流量。STM-DT 可以从聚合的网络流数据中检测不同类型网络攻击流量,而不仅仅是针对僵尸网络的检测。本文将除了僵尸网络流量外其余流量均标

记为良性, 而 STM-DT 则会对僵尸网络以外的流量进一步区分, 如区分为 XSS 注入攻击, 暴力破解攻击等。由此导致了精确率和召回率的降低。另外 STM-DT 对僵尸网络数据集仅提取了 38 个特征, 可能存在特征提取不全面导致精确率下降的问题。本文方法 FC-Bot 取得了相对较好的结果, 精确率和召回率分别为 96.11% 和 94.00%, 接近 1DCNN 模型, 且取得了最低的 FPR 为 3.06%。

另外, 如图 14 所示是 FC-Bot 和 4 种先进检测方法的 ROC 曲线。可以看出, FC-Bot 具有最高的 AUC 值, 相比其他方法 AUC 值最高提升了 5%, 进一步证明了方法的有效性。

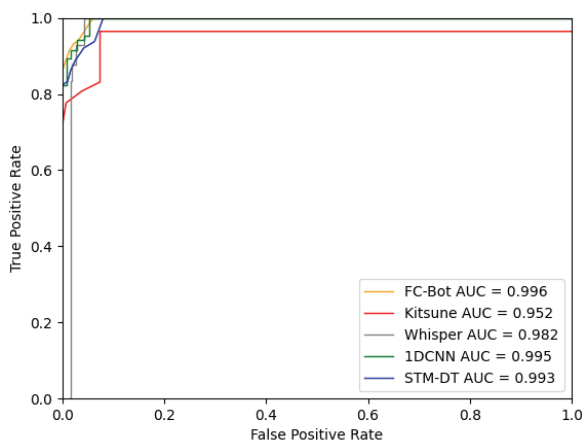


图 14 FC-Bot 和先进模型的 ROC 曲线

Figure 14 ROC for FC-Bot and advanced models

总的来说, FC-Bot 将流聚合模块引入僵尸网络检测, 通过流切割和流聚合, 既有效解决了时间的统一问题, 又使聚合后的行为达到集体性级别, 从事务的角度检测僵尸网络, 取得了良好的结果。相比提取包级别特征的检测方法 Kitsune, FC-Bot 能更好地提取僵尸网络流级别的信息提高检测准确率。频域分析方法 Whisper, 需要获得良性流量聚类中心并将离群流量判定为恶意流量, 极大地依赖于良性流量的覆盖范围, 而 FC-Bot 无需此种顾虑。另外, STM-DT 方法较为依赖于先验知识提取特征, 对僵尸网络的共享特征缺乏有效的观察。1DCNN 模型专注于僵尸网络的时序、上下文特征而忽略了僵尸网络的高维事务特征, 不能从集体性行为的角度进行检测, 造成较高的误报率。

4.6 检测开销讨论

为了评估检测开销, 我们针对 471 个 Pcap 流量文件进行处理, 共计 863MB。实验机器采用的是 DELL R730, 具有 2 个 10 核 20 线程 Intel Xeon E5-2630 v4 的 CPU, 主频为 2.20GHz, 内存 192GB。

本文的检测开销时间包括流量预处理时间、流聚合时间以及模型分类时间, 实验测试得到预处理耗时 0.92s, 流聚合耗时 28.05s, 模型分类时间耗时 2.286s。因此, 检测总开销时间为 31.256s。从流量处理速率来看, 处理 863MB 流量的开销时间为 31.256s, 计算得到每秒可以处理 221Mbps 的流量, 处理速率适用于一般中小企业。另外, 检测阶段的操作是串行进行的, 在预处理阶段还可以并行提取 NetFlow 流量, 对于流聚合的事务也可以并行提取事务指纹, 检测性能可以得到进一步提升。此外, 本文的检测也支持分布式处理, 如在较大流量的高速网络环境中, 就可以通过源或目标 IP 对流量镜像到多个处理机器, 进行分布式协同处理。

4.7 现实场景中 FC-Bot 的检测结果

如 4.1.2 节所述, 本文又专门收集了另一段时期的 Antiy 现实物联网流量, 其为 2022 年 1 月至 3 月安天公司在物联网设备捕获的流量, 完全来自真实场景, 可用于验证 FC-Bot 在现实环境中的检测效果。表 10 所示是 2022 年 3 月第一周 Antiy 现实物联网流量包含的部分僵尸网络, 其中既包括较为经典的 Mirai 僵尸网络、Gh0st 僵尸网络, 也包括近几年出现的 Mozi 僵尸网络、Pink 僵尸网络和 BotenaGo 僵尸网络。

表 10 2022 年 3 月第一周 Antiy 现实物联网流量包含的部分僵尸网络

Table 10 Some of the botnets included in Antiy's real IoT traffic in the first week of March 2022

僵尸网络类别	流数量	数据包大小
DarkComet	142	60K
Gh0st	41	23K
Mirai	196	94K
Mozi	57	181K
Pink	228	398K
FritzFrog	36	12K
BotenaGo	1	56K

如表 11 所示, 是 FC-Bot 在现实场景检测的各指标结果。从表中可以看出, 在实际场景下 FC-Bot 具有较好的检测效果, 取得了 92.91% 的精确率和 92.27% 的召回率, F1 值为 92.59%。

表 11 FC-Bot 在现实场景检测的各指标结果

Table 11 The results of FC-Bot in the real scene

Precision	Recall	F1	FPR
92.91%	92.27%	92.59%	5.28%

FC-Bot 误报率稍高, 具体分析误报情形可分为以下两类。

1) 将部分新型僵尸网络误报。如今物联网僵尸网络不断更新进化, 变体或新型僵尸网络频繁出现。以 2021 年 11 月才披露的 BotenaGo 僵尸网络为例, 根据 VirusTotal^[31]的数据, 时至今日 74 家安全厂商也仅有 27 家能够成功检测。此外 BotenaGo 也尝试利用了 Apache Log4j2 漏洞^[32], 这在之前的僵尸网络中并未出现过。另外, 还有一些新型物联网僵尸网络如 Mozi、Pink 及 FritzFrog 等。在训练阶段这些僵尸网络样本极少或根本没有, 而在现实场景中存在这类僵尸网络, 如表 8 所示, Antiy 现实物联网流量中就包括了 Mozi、Pink、BotenaGo 僵尸网络。表 12 展示了部分僵尸网络、发现时间和 FC-Bot 检测成功率。尽管对于其中的 DDoS 攻击流量 FC-Bot 能完全检测出来, 但对于部分僵尸网络控制流量 FC-Bot 存在误报。例如 BotenaGo 僵尸网络, FC-Bot 的检测成功率为 0, 一方面可能因为其隐秘性较高, 且在训练集中并未出现, FC-Bot 没有学会控制流量的特征; 另一方面在我们整理的现实场景流量中仅有 BotenaGo 控制流量并未发现 DDoS 攻击流量, 而 DDoS 攻击流量才是僵尸网络中具有最显著特征流量。但是对于 Mirai 僵尸网络以及近几年出现的 Mozi 僵尸网络和 Pink 僵尸网络等, FC-Bot 的检测成功率超过 90%, 可以证明我们的模型有较好的泛化适应性。

表 12 部分僵尸网络和 FC-Bot 的检测成功率

Table 12 Some botnets and detection success rate of FC-Bot

僵尸网络	发现时间	检测成功率
Mirai	2016 年 8 月	96.15%
Mozi	2019 年 9 月	92.86%
Pink	2019 年 12 月	90.91%
BotenaGo	2021 年 11 月	0.00%

2) 将僵尸网络部分行为误报为良性行为。在现实场景的检测中, FC-Bot 误报了两例 Dark_nexus 僵尸网络事务。进一步分析发现, 这是 Dark_nexus 僵尸网络的一个特点, 有时会伪装攻击。如图 15(a)展示了其中一次事务, 这是 Dark_nexus 发起的一次攻击, 其试图将流量伪装成由浏览器生成的无害流量, 生成的事务指纹图像是连续且稳定的。这和图 15(b)僵尸网络一般行为的指纹图像存在显著差异, 一定程度上更接近图 15(c)的正常流量指纹图像, 因此 FC-Bot 产生误报。此外, 我们还发现 FC-Bot 误报了 11 例恶意文件下载的事务。这是被感染的僵尸机尝

试去感染更多机器, 并向其传输恶意文件。由于在 FC-Bot 训练阶段, 本文使用的 Iot-23 数据集、CIC-IDS2017 数据集和 Antiy 数据集里大部分都是成型的僵尸网络, 有关感染扩大僵尸网络的流量较少。因此 FC-Bot 将其误报为良性流量的文件下载事务。未来可以通过在训练集中补充恶意文件传输流量来减少此类误报。

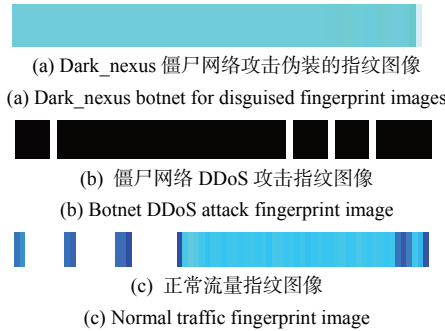


图 15 事务指纹图像的对比

Figure 15 Comparison of transaction fingerprint images

5 相关工作

僵尸网络根据其网络结构可以分为两种类型, 即集中命令和控制(Command and Control, C&C)和分散对等(Point to Point, P2P)^[33]。在 C&C 僵尸网络中, 被感染的机器(bot)连接到一个被称为命令和控制服务器的特定服务器, 以交换信息并接收来自僵尸主机的恶意命令。P2P 僵尸网络具有分散的结构, 受到感染的机器直接相互交换消息, 而无须经过中央服务器。

针对僵尸网络传播的前几步以及 bot 机器与 C&C 服务器的通信, Ozcelik 等人^[34]提出了一种面向边缘的检测方案, 专注于物联网中的 DDoS 攻击, 并以 Mirai 僵尸网络为例进行了研究。Summerville 等人^[35]开发了一种超轻量级的深度数据包异常检测方法, 可以在资源受限的物联网设备上运行。然而, 考虑到僵尸网络攻击每天都在不断变异并变得越来越复杂^[36], 上述面向僵尸网络特定阶段的方法存在局限。

Dange 等人^[37]将物联网僵尸网络检测技术分为基于主机的检测和基于网络的检测。基于网络的检测又进一步分为基于签名、基于 DNS、基于流量、基于异常和基于挖掘的检测方法。

基于通信签名检测的方法通过预先配置特征匹配规则, 发现僵尸网络活动。Gu 等人^[38]提出了检测引擎 BotHunter, 它通过行为匹配来检测恶意软件感染过程的特定阶段, 包括入站扫描、漏洞利用、出站

bot 协调对话和出站攻击传播等。Babu 等人^[39]提出了一种基于混合的入侵检测系统,它建立了基于签名和基于异常的检测器模型,专注于 DDoS 攻击,并使用基于机器学习的集成技术来检测僵尸网络和未知攻击。然而,上述方法需要不断维护和更新签名知识库,这增加了检测成本。

Singh 等人^[40]对基于 DNS 的僵尸网络检测技术的逐年演变进行了总结,并评估了现有方法的优缺点。然而,他们的研究仅对基于人工智能的僵尸网络检测做出了展望,并未提供具体的解决方案。

基于异常的检测是僵尸网络检测中的一个重要研究领域,其基本思想是基于主机行为或网络流量异常来识别僵尸网络,例如高网络延迟、大量流量、异常端口上的流量和异常系统行为^[41]。在过去的几十年中,研究人员使用了各种传统的机器学习方法来检测僵尸网络,并解决了经典方法的一些缺点^[42-43]。Azab 等人^[44]提出了一种网络信息流分析框架,利用丰富的网络流量特征来捕获 C&C 通信渠道和恶意流量的模式。Li 等人^[45]在高速网络环境下动态提取流量特征,并使用随机森林模型在 CTU-13 数据集上验证了该方法的有效性。然而,上述方法必须定期更新僵尸网络流量日志,否则无法检测到最新的僵尸网络变体。Nguyen 等人^[46]提出了基于全时序列数据的协作学习机器模型,可以有效地自动检测物联网僵尸网络。但是该检测框架需要建立在 V-Sandbox 执行环境和协作机器学习模型之上,时间资源消耗较大,不符合轻量级的要求。Alani 等人^[47]设计了 BotStop 系统用于检测物联网僵尸网络,但该系统仅提取了数据包级别的特征,存在局限性。

随着深度学习的快速发展,有大量研究基于深度学习模型检测物联网僵尸网络。例如卷积神经网络模型^[48],前馈人工神经网络模型^[49],时序卷积神经网络模型^[50]等。Meidan 等人^[51]提取网络行为的快照,并使用深度自动编码器来检测受感染的物联网设备中的异常网络流量。然而,该方法是基于受感染设备的检测,存在滞后性。

Bertino 等人^[52]采用了数据链架构来提高物联网僵尸网络检测的性能。他们提出了一种基于经验评估的网络方法,用于识别来自物联网僵尸网络的攻击。Rahmantyo 等人^[53]使用深度残差一维卷积神经网络模型进行僵尸网络流量检测,并取得了超过 88% 的验证准确率。Joshi 等人^[54]提出了一种特征工程方法,使用 ANN 模型进行僵尸网络的分类。然而该方法只在 CTU-13 数据集上进行评估,存在局限性。Popoola 等人^[55]提出了用于零日僵尸网络攻击检测

的联邦深度学习方法,以避免物联网边缘设备中的数据隐私泄露。上述方法需要模型参数服务器与物联网边缘设备进行较多通信轮数才能收敛,模型训练时间较长不符合轻量级的需求。

Nguyen 等人^[56]则从恶意软件检测的角度设计了 PSI-Graph,利用控制流图和图神经网络来检测物联网僵尸网络。此外,文献[57]在 PSI-Graph 的基础上引入了动态字符串特征,将静态分析和动态分析相结合,以提高检测效率。然而,这些方法的本质仍然是恶意软件检测,忽略了物联网僵尸网络的特性。

另外,由于僵尸网络攻击模式多种多样,在一种数据集上取得良好效果的特征在其他数据集上表现未必可靠。且当前流量数据规模庞大,常以 TB 计量,检测效果提升的同时必然伴随着时间的大量损耗。因此,本文尝试使用 NetFlow 这一轻量级数据进行处理,实现了检测效果与检测速度的兼顾,并进一步通过流聚合模块,以及使用聚类算法归纳僵尸主机的行为。

6 结束语

本文提出了一种轻量级的基于 NetFlow 的物联网僵尸网络检测方法 FC-Bot,在保护隐私的同时实现了无须知晓数据包的内容而能够对 IoT 僵尸网络进行检测,为物联网安全提供了行之有效的安全保障方案。FC-Bot 引入流聚合模块,结合 HDBSCAN 算法,先将 NetFlow 数据切割为流点再将流点聚合,使聚合后的行为达到事务级别,进而从事务的角度区分正常流量与僵尸网络流量。随后提取事物特征指纹并转化为 RGB 指纹图像,利用搭建的卷积神经网络进行检测分类。在模拟环境中的测试结果表明,FC-Bot 取得了 95.04% 的 F1 值,证明了方案的有效性。相比目前先进的检测方案,FC-Bot 具有最低的误报率 3.06%,最高相比 Kitsune 改善了 4.30%,更有优势。在真实场景中,FC-Bot 也取得了 92.59% 的 F1 值,能够检测出大部分僵尸网络,提供了一种可行的物联网僵尸网络检测方案。

参考文献

- [1] Yan Z, Niemi V, Yang L T. Key Technologies for 5G, the Next Generation of Mobile Networks and Services[J]. *International Journal of Communication Systems*, 2016, 29(16): 2328-2329.
- [2] GSMA. The Internet of Things by 2025. <https://www.gsma.com/iot/wp-content/uploads/2018/08/GSMA-IoT-Infographic-2019.pdf>. Aug. 2018.
- [3] Straits Research. IoT Security Market. <https://straitsresearch.com/report/iot-security-market/segmentation>. Feb. 2023.

- [4] Fortinet. 2022 Global Threat Landscape Report. <https://global.fortinet.com/lp-en-ap-2022globalthreatlandscape-H2>. Aug. 2022.
- [5] Vinayakumar R, Alazab M, Srinivasan S, et al. A Visualized Botnet Detection System Based Deep Learning for the Internet of Things Networks of Smart Cities[J]. *IEEE Transactions on Industry Applications*, 2020, 56(4): 4436-4456.
- [6] Gao H Y, Hu J, Wilson C, et al. Detecting and Characterizing Social Spam Campaigns[C]. *The 10th ACM SIGCOMM Conference on Internet Measurement*, 2010: 35-47.
- [7] Doshi R, Apthorpe N, Feamster N. Machine Learning DDoS Detection for Consumer Internet of Things Devices[C]. *2018 IEEE Security and Privacy Workshops*, 2018: 29-35.
- [8] Hussain F, Abbas S G, Fayyaz U U, et al. Towards a Universal Features Set for IoT Botnet Attacks Detection[C]. *2020 IEEE 23rd International Multitopic Conference*, 2020: 1-6.
- [9] Wazzan M, Algazzawi D, Bamasaq O, et al. Internet of Things Botnet Detection Approaches: Analysis and Recommendations for Future Research[J]. *Applied Sciences*, 2021, 11(12): 5713.
- [10] Al Shorman A, Faris H, Aljarah I. Unsupervised Intelligent System Based on One Class Support Vector Machine and Grey Wolf Optimization for IoT Botnet Detection[J]. *Journal of Ambient Intelligence and Humanized Computing*, 2020, 11(7): 2809-2825.
- [11] Soe Y N, Feng Y K, Santosa P I, et al. Machine Learning-Based IoT-Botnet Attack Detection with Sequential Architecture[J]. *Sensors*, 2020, 20(16): 4372.
- [12] Sriram S, Vinayakumar R, Alazab M, et al. Network Flow Based IoT Botnet Attack Detection Using Deep Learning[C]. *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops*, 2020: 189-194.
- [13] Hussain F, Abbas S G, Pires I M, et al. A Two-Fold Machine Learning Approach to Prevent and Detect IoT Botnet Attacks[J]. *IEEE Access*, 2021, 9: 163412-163430.
- [14] Gandhi H, Mehra M, Ribeiro V. BOND: Efficient and Frugal DL Model Co-Design for Botnet Detection on IoT Gateways[C]. *The First International Conference on AI-ML Systems*, 2021: 1-7.
- [15] Alzahrani H, Abulkhair M, Alkayal E. A Multi-Class Neural Network Model for Rapid Detection of IoT Botnet Attacks[J]. *International Journal of Advanced Computer Science and Applications*, 2020, 11(7): 344-352.
- [16] Bojarajulu B, Tanwar S, Singh T P. Intelligent IoT-BOTNET Attack Detection Model with Optimized Hybrid Classification Model[J]. *Computers & Security*, 2023, 126: 103064.
- [17] Bilge L, Balzarotti D, Robertson W, et al. Disclosure: Detecting Botnet Command and Control Servers through Large-Scale NetFlow Analysis[C]. *The 28th Annual Computer Security Applications Conference*, 2012: 129-138.
- [18] Claise B. Cisco Systems NetFlow Services Export Version 9[J]. *RFC*, 2004, 3954: 1-33.
- [19] Singh D, Tripathi G, Jara A J. A Survey of Internet-of-Things: Future Vision, Architecture, Challenges and Services[C]. *2014 IEEE World Forum on Internet of Things*, 2014: 287-292.
- [20] McInnes L, Healy J, Astels S. Hdbscan: Hierarchical Density Based Clustering[J]. *The Journal of Open Source Software*, 2017, 2(11): 205.
- [21] Asyaky M S, Mandala R. Improving the Performance of HDBSCAN on Short Text Clustering by Using Word Embedding and UMAP[C]. *2021 8th International Conference on Advanced Informatics: Concepts, Theory and Applications*, 2021: 1-6.
- [22] Alkahtani H, Aldhyani T H H. Botnet Attack Detection by Using CNN-LSTM Model for Internet of Things Applications[J]. *Security and Communication Networks*, 2021, 2021: 3806459.
- [23] Aydın E, Bahtiyar Ş. OCIDS: An Online CNN-Based Network Intrusion Detection System for DDoS Attacks with IoT Botnets[C]. *2021 14th International Conference on Security of Information and Networks*, 2021: 1-8.
- [24] Stratosphere Laboratory. IoT-23: A labeled dataset with malicious and benign IoT network traffic. <https://www.stratosphereips.org/datasets-iot23>. March, 2020.
- [25] Sharafaldin I, Habibi Lashkari A, Ghorbani A A. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization[C]. *The 4th International Conference on Information Systems Security and Privacy*, 2018: 108-116.
- [26] Mirsky Y, Doitshman T, Elovici Y, et al. Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection[EB/OL]. 2018: arXiv: 1802.09089. <https://arxiv.org/abs/1802.09089>.
- [27] Fu C P, Li Q, Shen M, et al. Realtime Robust Malicious Traffic Detection via Frequency Domain Analysis[C]. *The 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021: 3431-3446.
- [28] Kundu P P, Truong-Huu T, Chen L, et al. Detection and Classification of Botnet Traffic Using Deep Learning with Model Explanation[J]. *IEEE Transactions on Dependable and Secure Computing*, 2022. DOI:10.1109/TDSC.2022.3183361.
- [29] Hou Y B, Teo S G, Chen Z H, et al. Handling Labeled Data Insufficiency: Semi-Supervised Learning with Self-Training Mixup Decision Tree for Classification of Network Attacking Traffic[J]. *IEEE Transactions on Dependable and Secure Computing*, 2022. DOI:10.1109/TDSC.2022.3195534.
- [30] Schneider F, Feldmann A, Krishnamurthy B, et al. Understanding Online Social Network Usage from a Network Perspective[C]. *The 9th ACM SIGCOMM Conference on Internet Measurement*, 2009: 35-48.
- [31] VirusTotal. VIRUSTOTAL. <https://www.virustotal.com/>. June. 2004.
- [32] CVE. CVE-2021-44228. <http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-44228>. Nov. 2021.
- [33] Wang C Y, Ou C L, Zhang Y E, et al. BotCluster: A Session-Based P2P Botnet Clustering System on NetFlow[J]. *Computer Networks*, 2018, 145: 175-189.
- [34] Özçelik M, Chalabianloo N, Gür G. Software-Defined Edge Defense Against IoT-Based DDoS[C]. *2017 IEEE International Conference on Computer and Information Technology*, 2017: 308-313.
- [35] Summerville D H, Zach K M, Chen Y. Ultra-Lightweight Deep Packet Anomaly Detection for Internet of Things Devices[C]. *2015 IEEE 34th International Performance Computing and Communications Conference*, 2015: 1-8.
- [36] Kolias C, Kambourakis G, Stavrou A, et al. DDoS in the IoT: Mirai and Other Botnets[J]. *Computer*, 2017, 50(7): 80-84.

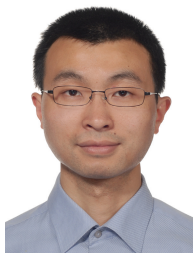
- [37] Dange S, Chatterjee M. IoT Botnet: The Largest Threat to the IoT Network[M]. Data Communication and Networks. Singapore: Springer Singapore 2019: 137-157.
- [38] Gu G, Porras P A, Yegneswaran V, et al. Bothunter: Detecting malware infection through ids-driven dialog correlation[C]. *USENIX Security Symposium*. 2007, 7: 1-16.
- [39] Babu E S, Rao M S, Pemula R, et al. A Hybrid Intrusion Detection System against Botnet Attack in IoT using Light Weight Signature and Ensemble Learning Technique[J]. *ResearchSquare*, 2022.
- [40] Singh M, Singh M, Kaur S. Issues and Challenges in DNS Based Botnet Detection: A Survey[J]. *Computers & Security*, 2019, 86: 28-52.
- [41] Xing Y, Shu H, Zhao H, et al. Survey on Botnet Detection Techniques: Classification, Methods, and Evaluation[J]. *Mathematical Problems in Engineering*, 2021, 2021: 6640499.
- [42] Lashkari A H, Gil G D, Keenan J E, et al. A Survey Leading to a New Evaluation Framework for Network-Based Botnet Detection[C]. *The 2017 7th International Conference on Communication and Network Security*, 2017: 59-66.
- [43] Koroniotis N, Moustafa N, Sitnikova E, et al. Towards the Development of Realistic Botnet Dataset in the Internet of Things for Network Forensic Analytics: Bot-IoT Dataset[J]. *Future Generation Computer Systems*, 2019, 100: 779-796.
- [44] Azab A, Alazab M, Aiash M. Machine Learning Based Botnet Identification Traffic[C]. *2016 IEEE Trustcom/BigDataSE/ISPA*, 2016: 1788-1794.
- [45] Li L X, Ota K, Zhang Z H, et al. Security and Privacy Protection of Social Networks in Big Data Era[J]. *Mathematical Problems in Engineering*, 2018, 2018: 6872587.
- [46] Nguyen G L, Dumba B, Ngo Q D, et al. A Collaborative Approach to Early Detection of IoT Botnet[J]. *Computers and Electrical Engineering*, 2022, 97: 107525.
- [47] Alani M M. BotStop: Packet-Based Efficient and Explainable IoT Botnet Detection Using Machine Learning[J]. *Computer Communications*, 2022, 193: 53-62.
- [48] Wei C B, Xie G G, Diao Z L. A Lightweight Deep Learning Framework for Botnet Detecting at the IoT Edge[J]. *Computers & Security*, 2023, 129: 103195.
- [49] Chen S C, Chen Y R, Tzeng W G. Effective Botnet Detection through Neural Networks on Convolutional Features[C]. 2018 17th IEEE International Conference on Trust, Security and Privacy In Computing and Communications/ 12th IEEE International Conference on Big Data Science and Engineering, 2018: 372-378.
- [50] Sadique F, Sengupta S. Modeling and Analyzing Attacker Behavior in IoT Botnet Using Temporal Convolution Network (TCN)[J]. *Computers & Security*, 2022, 117: 102714.
- [51] Meidan Y, Bohadana M, Mathov Y, et al. N-BaIoT— Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders[J]. *IEEE Pervasive Computing*, 2018, 17(3): 12-22.
- [52] Bertino E, Islam N. Botnets and Internet of Things Security[J]. *Computer*, 2017, 50(2): 76-79.
- [53] Rahmantyo D T, Erfianto B, Satrya G B. Deep Residual CNN for Preventing Botnet Attacks on the Internet of Things[C]. *2021 4th International Conference of Computer and Informatics Engineering*, 2021: 462-466.
- [54] Joshi C, Ranjan R K, Bharti V. A Fuzzy Logic Based Feature Engineering Approach for Botnet Detection Using ANN[J]. *Journal of King Saud University - Computer and Information Sciences*, 2022, 34(9): 6872-6882.
- [55] Popoola S I, Ande R, Adebisi B, et al. Federated Deep Learning for Zero-Day Botnet Attack Detection in IoT-Edge Devices[J]. *IEEE Internet of Things Journal*, 2022, 9(5): 3930-3944.
- [56] Nguyen H T, Ngo Q D, Le V H. A Novel Graph-Based Approach for IoT Botnet Detection[J]. *International Journal of Information Security*, 2020, 19(5): 567-577.
- [57] Nguyen T N, Ngo Q D, Nguyen H T, et al. An Advanced Computing Approach for IoT-Botnet Detection in Industrial Internet of Things[J]. *IEEE Transactions on Industrial Informatics*, 2022, 18(11): 8298-8306.



周纸墨 于 2021 年在上海交通大学信息安全专业获得学士学位。现在上海交通大学信息安全专业攻读硕士学位。研究方向为网络安全、入侵检测。Email: dy-zzm@sjtu.edu.cn



邹福泰 博士, 上海交通大学高级工程师, 硕士生导师。主要研究方向为 DNS 安全、流量安全、网络安全。Email: zoufuta@sjtu.edu.cn



姜开达 博士, 上海交通大学网络信息中心副主任, 硕士生导师。主要研究方向为网络安全、流量安全、漏洞挖掘。Email: kaida@sjtu.edu.cn



康学斌 硕士, 安天实验室研究员, 主要研究方向为恶意代码追踪溯源和欺骗防御技术。Email: dy-zzm@sjtu.edu.cn