电子科技大学

UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

专业学位硕士学位论文

MASTER THESIS FOR PROFESSIONAL DEGREE

论文题目 面向 Fast-flux 与 Domain-flux 的僵尸网络流量检测方法研究

专业学位类别	工 程 硕 士
学 号	201622060414
作 者 姓 名	熊智鹏
指 导 教 师	杨浩淼 副教授

分类号 _____ 密级 _____

UDC [注1] _____

学 位 论 文

面向 Fast-flux 与 Domain-flux 的僵尸网络流量检测方法研究

（题名和副题名）

熊智鹏

（作者姓名）

指导教师 杨浩淼 副教授

电子科技大学 成 都

（姓名、职称、单位名称）

申请学位级别 硕士 专业学位类别 工 程 硕 士

工程领域名称 计算机技术

提交论文日期 2019.3.27 论文答辩日期 2019.5.21

学位授予单位和日期 电子科技大学 2019 年 6 月

答辩委员会主席 _____

评阅人 _____

注 1：注明《国际十进分类法 UDC》的类号。

Research on Fast-Flux and Domain-flux Botnet Traffic Detection Method

A Master Thesis Submitted to

University of Electronic Science and Technology of China

Discipline: **Master of Engineering**

Author: **Zhipeng Xiong**

Supervisor: **Haomiao Yang**

School: **School of Computer Science & Engineering**

独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得电子科技大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

作者签名：熊智鹏　　日期：2019年 5月27日

论文使用授权

本学位论文作者完全了解电子科技大学有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权电子科技大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后应遵守此规定）

作者签名：熊智鹏　导师签名：杨浩教

日期：2019年 5月27日

摘　要

信息网络的飞速发展，迎来了互联网时代。随之而来的网络安全问题也突显出来，其中僵尸网络随着技术的变革也卷土重来，而采用新技术的新型僵尸网络也逐渐成为研究热点。在僵尸网络建立通信传输的过程中，如今比较主流的是利用到 DNS 协议的相关技术，分离服务器域名与 IP 地址，旨在受感染的网络中实现可靠的通信，同时对恶意内容的灵活和弹性的托管。

这其中应用较为广泛的技术主要包括 Fast-flux、Domain-flux 技术。Fast-flux 技术就是指域名和 IP 地址之间的关联关系会不断发生变化的一种技术，而 Domain-flux，而是根据一定的算法动态生成的、变化的域名，而攻击者只要实现受控主机注册到一个有效域名，即可以实现僵尸网络的控制，目前对于这两种技术类型的研究多关注在其中一种类型，而对于检测特征多集中于流量的局部特征，难以应对不断更新变化的僵尸网络。

针对上述问题，本文通过深入分析僵尸网络的技术原理，进一步研究 Fast-flux 与 Domain-flux 两种类型僵尸网络的检测现状，提出一种面向该两种类型高效新型的检测方法，其主要内容与创新点如下：

1. 研究上述两种类型僵尸网络，基于 DNS 流量，提出一种基于 DNS 关联映射图的新型检测模型，该模型旨在根据 DNS 查询响应，提取其域名与 IP 的映射关系，构建 DNS 映射关联图，且能够同时满足 Fast-flux 与 Domain-flux 两种类型僵尸网络的检测要求。

2. 根据 DNS 关联映射图，提出一种基于图组件元素多维特征的分析方法。包括图的结构特征、节点特征（FQDN 节点与 IP 节点）、连接边特征，同时融合黑名单统计特征，实现图组件的多特征分析，选取 LightGBM 算法完成图组件分类。

3. 分析检测系统在实际网络环境下的应用难题，设计并实现一种满足高速网络下僵尸网络流量检测需求的原型系统。

最后，本文基于 CTU-13 等僵尸网络数据集进行算法评估与测试，结果显示在综合类型的数据集上，对于僵尸网络流量的分类检测相较普遍的单类型检测方法具备更高的准确率，能够达到 92%以上。随后再设计并实现僵尸网络检测原型系统。经过测试结果表明，系统能够满足高速网络中的检测需求，能够实现 Fast-flux 与 Domain-flux 两种类型的僵尸网络流量的检测。

关键词：僵尸网络，DNS 流量，Fast-Flux 技术，Domain-flux 技术

ABSTRACT

The rapid development of information network ushered in the Internet age. The following network security issues are also highlighted, in which the botnet is coming back with the technological changes. And the new technologies used in new botnets have gradually become research hotspots. In the process of establishing communication and transmission in botnet, the mainstream is that useing the related technology of DNS protocol to separate the server domain name and IP address, and to achieve reliable communication in the infected network, and flexible hosting of malicious content.

The most widely used technologies include Fast-flux and Domain-flux technologies. Fast-flux technology refers to the technology that the relationship between domain name and IP address will change constantly. And the Domain-flux, dynamically generates a changed domain name according to a certain algorithm, and an attacker can control the botnet by implementing a controlled host registration to a valid domain name. At present, most of the research on these two types of technology focuses on one type, while most of the detection features focus on the local characteristics of traffic, which makes it difficult to cope with the constantly changing botnet.

In response to the above problems, this thesis further studies the detection status of Fast-flux and Domain-flux botnets by analyzing the technical principles of botnets, and proposes a new and efficient detection method for these two types of botnets. The main contents and innovations are as follows:

1. Research on the above two types of botnets, based on DNS traffic, propose a new detection model based on DNS association map. The model aims to extract the mapping relationship between domain name and IP based on DNS query response, and construct a DNS mapping association graph, which can satisfy the detection requirements of two types of botnets, Fast-flux and Domain-flux.

2. According to the DNS association map, an analysis method based on multi-dimensional features of graph component elements is proposed. Including the structural features of the graph, the node features (FQDN node and IP node), the connection edge feature, and the blacklist statistical features are combined to realize the multi-feature analysis of the graph component, and the LightGBM algorithm is selected to complete the graph component classification.

3. Analyze the application problems of the detection system in the actual network environment, and designing and implementing a prototype system that meets the traffic detection requirements of botnets under high-speed networks.

Finally, the thesis evaluates and tests the algorithm based on the botnet datasets such as CTU-13. The results show that the classification detection of botnet traffic has higher accuracy that reach more than 92% than the common single-type detection method on the comprehensive type of data set. And designing and implementing the botnet detection prototype system. The test results show that the system can meet the detection requirements in high-speed networks, and can detect the botnet traffic of both Fast-flux and Domain-flux.

Keywords: Botnet, DNS traffic, Fast-Flux, Domain-flux

目 录

第一章 绪 论

1.1 研究背景及意义

作为新世纪的信息发布、获取、传播、分享的平台，网络已经在不经意间成为了我们社会生活的必需品[1]。但技术的发展也随之带来了诸多安全问题。如计算机资源（文件、系统、程序等）的盗用、窃取、破坏等不当访问。其中涉及的网络技术包括计算机病毒、蠕虫、木马程序、网络窃听、逻辑炸弹、阻断网络服务等。而网络攻击者大多以僵尸网络为载体，利用各种黑客技术，实施对个人信息的窃取、组织网络钓鱼、勒索软件、虚拟货币挖矿、甚至组织大规模网络攻击等恶意行为。

僵尸网络（botnet）是指攻击者采用一种或多种传播手段，将恶意程序如计算机病毒、网络蠕虫、特洛伊木马和后门工具等植入感染大量控制主机，然后通过一对多的命令与控制信道所组成的网络[2]。这些被黑客控制的主机，如同“僵尸”，也称为“肉鸡”，任由控制者摆布，从而进行各类的非法网络活动。

根据 CNCERT/CC 官方发布的 2018 年 10 月互联网安全威胁报告[4]，截至 2018 年 10 月，境内感染网络病毒的终端数近 64 万个，其中近 41 万个 IP 地址对应的主机被木马或僵尸程序控制，较上月 40 万个相比增长 3.4%。网络病毒在传播过程中，黑客往往利用了注册的大量域名。截至报告发布时，CNCERT 监测发现的放马站点中，通过域名访问的共涉及有 13,538 个，通过 IP 直接访问的共 5,999 个。在放马站点域名中，于境内注册为 5,999 个（约占 44.3%），于境外注册的域名数为 2,539 个（约占 18.8%）。根据行业网络安全公司 Symantec 公布的 2018 年互联网安全威胁报告[5]显示，以 WannaCry 为代表的勒索软件攻击成为安全事件界的明星，2017 年勒索软件日均检出量约为 1242 次，保持高位水平。“Locky”家族主要散布者之一的 Necurs (Backdoor.Necurs) 僵尸网络仍然对 2017 年的网络犯罪威胁态势有着巨大影响，在 Web 威胁中，网关识别的整体僵尸网络活动增加了 62%，而攻击者也正试图将新型设备添加到僵尸网络，影响将波及物联网。挖矿活动可能会持续增加，会使部分云服务器用户遭受财务损失[6]。僵尸网络的形式变化之快与影响范围之广可见一斑。

僵尸网络作为最有效的网络攻击平台之一，给互联网的安全带来了巨大的威胁，而随着互联网的应用与技术的不断发展进步，僵尸网络的也有着新的变化与发展，一些新的技术与机制也被引入到其中，其演进得越发强大与健壮，这给网络安全的防御人员带来了新的挑战[7]。其中最引入关注的技术包括 Fast-flux 与

Domain-flux。Fast-flux 技术[8]主要通过不断改变 IP 到同一域名映射关系的一种技术，而 Domain-flux 则关注于动态生成域名，从而映射到特定 IP 的技术。这两种技术依托于 DNS 协议，实现域名与 IP 的动态变化，从而实现两者的动态映射，这种动态 DNS 技术，提供了灵活弹性的 DNS 策略，极大提到了僵尸网络服务的健壮性，同时也进一步增大了检测防护难度。

全球知名 CDN 服务商 Akamai 公司研究人员发现，恶意攻击者采用了一种巧妙的技术，即“Fast-flux”，使用 1.4 万多个 IP 地址组成的僵尸网络传播恶意软件，并且难以被击垮[9]。而在 2016 年，RiskAnalytics 发布了一份报告[13]，其中详细阐述了 Zbot 僵尸网络使用一项被称为“Fast-flux”的技术，即将多个 IP 地址映射到同一个域名，DNS 查询该域名得到的 IP 地址是被不断更换过的，这样就能绕过检测并保持存活。2018 年 3 月，安全厂商 proofpoint 观察到一个新型恶意的基于 Fast-Flux 技术的恶意软件传播基础架构 SandiFlux，并在最近作为了 GandCrab 勒索软件基础设施的代理[14]。根据 Forcepoint 于 2017 发布的一份安全状况白皮书[15]，其中提到 ZeuS_Gameover 具备在线身份窃取和恶意软件安装功能，该僵尸网络采用 Domain Flux 协议，每天生成 1000 个纯字符域名，TLD 包括 com,biz,org,net,ru 和 info；当年 2 月出现的勒索软件 Locky 以随机数和当前日期作为输入，但一天只生成并访问 6 个域名。

因此可见，在僵尸网络中，由于 Fast-flux 与 Domain-flux 等机制的广泛应用，使其在如今的网络状况下仍然保持着极强的隐蔽性与较高的存活率，这对互联网而言是一个极大的安全隐患。因此，深入了解僵尸网络的运行机理与发展趋势，进一步研究 Fast-flux 与 Domain-flux 技术原理，并提出面向两者的新型的检测防御方法，且设计实现能应用于大型高速网络的原型系统具有深刻意义。

1.2 国内外研究现状

僵尸程序（bot）的起源可追溯到 1993 年，它最早叫“Eggdrop Bot”，是帮助 IRC 网络管理人员更高效的管理网络。而后来随着其不断发展，逐渐被用于以破坏为目的的非法行为。因此，对僵尸网络的研究也逐渐兴起。对僵尸网络领域的研究主要可归纳为检测、追踪、测量、预测和对抗 5 个部分，其中检测就是发现新的僵尸网络，也成为该研究的首要一步与研究热点。

国家互联网应急中心联合国内重要系统单位、基础电信运营商、网络安全公司，软件厂商以及互联网企业，每年针对互联网安全会定期发布《CNCERT 互联网安全威胁报告》，报告中重点关注了木马僵尸网络监测数据分析。国内一线安全公司如 360 成立的网络安全研究院长期监测僵尸网络。国外的通信服务提供商

CenturyLink 的威胁实验室通过记录流量，长期跟踪僵尸网络引起的威胁[16]。

对于使用 Fast-flux 与 Domain-flux 的僵尸网络检测的研究，关注点则主要集中在通过 DNS 流量分析，进一步检测僵尸网络的存在，最新的主要几种研究方法如下述。

1.2.1 基于 DNS 流量统计特征的分类检测方法

基于网络中通信产生的流量，在不同的协议、应用场景中必然会产生某些特定的流量特征，这个是流量被动分析的基本原理，也是做僵尸网络检测的常见方法，同样也针对 Fast-flux 与 Domain-flux 为主的僵尸网络检测。

Perdisci 等人[17]提出构建 FluxBuster 系统，它是一种基于被动 DNS 流量的分析系统，用于检测与跟踪恶意的 Fast-flux 网络。FluxBuster 主要通过监控位于分散于不同地理位置的数百个不同网络中的递归 DNS 服务器生成的流量来做分析检测。不仅使用从垃圾邮件或预生成域中提取的可疑域名，同时也结合监督的机器学习算法识别恶意域名的集群，该方法主要针对 Fast-Flux。

Bilge 等人[18]提出了 Exposure 系统，它通过应用 4 组 15 个唯一的 DNS 特征来做实时大型网络的域名检测。结合恶意软件域列表和垃圾邮件黑名单，同样使用监督的机器学习算法将域名分类为恶意或非恶意，同时还可以给定恶意的程度。该系统的方法旨在覆盖各种类型的恶意 DNS 活动，包括 Fast-flux 与 Domain-flux。

Shi Y 等人[19]提出了一种基于极限学习机（Extreme Learning machine）的恶意域名检测方法，主要依据 ELM 精度高、学习速度快的现代神经网络，从多个资源中提取特征，然后再对域名进行分类，能够有效识别可疑域，对识别 Domain-flux 有一定作用，还能在一定程度上增强 APT 攻击的检测。

1.2.2 基于 DNS 域名特征的检测方法

在 DNS 协议中，域名是其重要的组成部分，尤其在 Domain-flux 技术中，基于域名的特征而提出的检测方法也是其中有效途径之一。

Grill 等人[20]提出了一种利用域生成算法（DGA）检测恶意域的技术。它是使用一种统计方法，对本地网络中每台主机的 DNS 请求和访问 IP 的比例进行建模，并将偏离该模型的部分标记为执行 DGA 的恶意行为，具备快速高效的特点，还能够帮助识别僵尸网络受损客户机。

Pereira 等人[21]提出一种基于 DNS 流量的域生成算法的字典提取词图算法，它在传统的基于域名单词列表（也成为字典）或其他字符特征来分析域的基础上，将域的特征应用于图形中，然后通过字符构成图的特性来过滤 DGA 生成的域名。

1.2.3 基于 DNS 的行为检测方法

在网络应用中，通常需要根据 DNS 协议将网络域名转换为正常的 IP 地址，在这个过程中，必然会有 DNS 解析等各种行为，包括请求响应，连通性等。

Wang K 等人[22]提出一种基于行为的僵尸网络并行检测方法（BBDP），BBDP 采用模糊模式识别方法检测受感染的主机，主要根据 DNS 查询与 TCP 请求中的异常行为来检测。

Sharifnya 等人[23]设计了一种根据 DNS 组活动与查询失败的历史记录的检测系统 DFBotKiller，来检测 Domain-flux 僵尸网络，由于在该类型的僵尸网络中，通常会生成大量解析到相同 IP 地址的 DNS 请求查询，该系统旨在构建一个 DNS 的负声誉系统，根据可疑的群组活动以及 DNS 流量中失败的历史记录，给目标主机分配负声誉评分。

综上分析可知，目前针对包含 Fast-flux 或 Domain-flux 技术的僵尸网络的检测研究大多集中在 DNS 协议的基础上，来分析各个层面的特征，然后结合机器学习中的各类聚类或分类算法实现，从而进一步研究僵尸网络的检测，而这些研究方法主要存在如下几方面的问题：

1. 关注点的局限性问题。对于僵尸网络的检测，大多研究者主要关注于 Fast-flux 或 Domain-flux 某一方面的检测问题，没有能够从更高维度来思考适用性更高的方法，从而导致部分方案检测准确度不够、误报率高、适用范围受限等问题。

2. 特征的全面性问题。目前大多数方法提供的流量检测特征，很多不够全面，例如只包含局部特征 TTL、NS 记录、AS 记录等，而关于域名或 IP 的全局性特性如 DNS 注册信息、Whois 更新信息、IP 地址所属地等。如果关注特征过于片面，容易被攻击者发现而针对性地绕过。

3. 大量数据处理的效率问题。大部分方案中均给出了检测方法的实现方案，但针对的数据集或数据量整体有限，而针对于实际大型网络中高速大量的数据，则难以有效应对处理，会对检测的准确率、漏报率、误报率等指标影响较大。

4. 检测方案的设计问题。对于部分检测方法，其采用的机器学习训练模型多基于离线数据信息，而提取的特征多为离线数据特征，而基于此的验证与检测难以与实际在线情况相符合，且部分方案依赖主机的监测数据，实际中存在较大部署与应用难度。

针对上述问题，本文提出一种基于域名 IP 映射关联的新型检测方法，同时面向 Fast-flux 与 Domain-flux 类型的僵尸网络，并结合 PF_RING 技术，实现能够在大型高速网络环境下对僵尸网络流量检测的原型系统。

1.3 本文主要研究内容

通过分析目前使用 Fast-flux 或 Domain-flux 技术类型的僵尸网络检测方法，从 DNS 流量特征、域名特征、DNS 行为三个层面讨论目前主流的研究方法，经过对比分析，本文主要完成的工作如下：

1. 僵尸网络研究：深入研究目前僵尸网络的基本原理及运行机制，并重点研究僵尸网络中 Fast-flux 技术与 Domain-flux 技术的应用原理，并归纳总结针对该类型目标网络的检测方法。

2. DNS 图映射关联分析：提出一种基于 DNS 流量的域名 IP 映射关系的关联图分析方法，从构建关联图的各项特征基础上，在更高一个层面研究同时适用两种技术的方法。

3. 设计面向 Fast-flux 与 Domain-flux 的受监督的机器学习分类算法；在构建的图关联分析的基础上，结合两者技术的各自的各项特征，完成特征分析，提出使用 LightGBM 的分类算法。

4. 设计实现在大型高速网络下的僵尸网络检测原型系统：结合本文提出的新型检测分析方法，设计实现检测原型系统，在公开的僵尸网络数据集的基础上进行验证，并结合实验室真实网络环境下附加重放流量完成对该原型系统的在线测试与评估。

1.4 本文的组织结构

第一章：绪论。概述了僵尸网络检测的背景及意义，介绍了目前在针对 Fast-flux 或 Domain-flux 类型僵尸网络检测的国内外最新研究现状，同时阐述本文的研究内容，并对本文的组织结构安排做进一步说明。

第二章：Fast-flux 与 Domain-flux 僵尸网络研究。主要介绍僵尸网络的基本原理及运行机制，并着重分析 Fast-flux 技术与 Domain-flux 技术在僵尸网络中的应用情况。

第三章：基于 DNS 流量映射关联的僵尸网络检测方法设计。针对 Fast-flux 与 Domain-flux 类型僵尸网络的诸多特征，提出一种综合性的检测方法，包括 DNS 流量过滤，域名 IP 映射提取，图组件特征提取，结合其他特征的完整性检测方法，并采用 LightGBM 机器学习算法进行分类识别完成检测。

第四章：大型高速网络中僵尸网络的检测系统设计与实现。在前面章节提出的检测方法的基础上，结合实际大型高速网络的运行特点，做针对性的设计优化，实现该网络情况下的僵尸网络检测原型系统，并详细描述该系统的实现步骤。

第五章：算法评估与系统测试。在完成原型系统设计实现的基础上，在公开

的僵尸网络的数据集上进行实验验证，并给出实验的评估结果，并结合实际网络环境完成在线测试。

第六章：总结和展望。最后完成本文的总结，包括论文的研究重点，本论文的一些不足之处，以及后期改进优化的思路与想法。

1.5 本章小结

本章节内容主要是阐述了当前僵尸网络研究的背景和意义，同时总结了前人对 Fast-flux 或 Domain-flux 类型僵尸网络检测问题的研究成果，最后给出本文的主要研究内容以及文章的整体内容安排。

第二章 Fast-flux 与 Domain-flux 僵尸网络相关研究

2.1 僵尸网络概述

僵尸网络是在传统的计算机病毒、特洛伊木马、网络蠕虫、间谍软件等恶意代码基础上发展而来的一种综合性攻击方法的平台。而近些年，新型的僵尸网络程序还能够将 0DAY 漏洞[24]、网络钓鱼、p2p 等技术应用到其传播中，能够使得传统主机、移动设备甚至云设备、路由器感染而成为僵尸网络控制的主机，俗称“肉鸡”。而僵尸网络仍然是当前互联网重大威胁之一，再加上其不断变化发展。对僵尸网络的检测成为了安全领域人员一直以来的重大挑战。因此，本文将针对僵尸网络的定义、类型及逃逸方式等内容做进一步研究。

2.1.1 僵尸网络的定义

僵尸网络可以解释为由 4 元组组成，攻击者通过命令控制信道从而进行控制，而使目标主机进入特定状态的计算机群[12]，李可等[7]给出其形式化定义：

$$Botnet = (Zombie, Botmaster, CMD, CCC) \tag{2-1}$$

1. *Zombie*：指被攻击者入侵而获取的僵尸主机的集合，可记为：

$$Zombie = (BOT, S, Activity) \tag{2-2}$$

其中 *BOT* 表示受控的主机上运行的僵尸程序的集合；*S* 表示僵尸程序状态集合；*Activity* 则表示僵尸程序在接受控制指令后的动作集合[12]。

Botmaster：指僵尸网络的攻击者，也称为控制者，主要是其通过命令控制信道完成对僵尸主机的各种操作，以实现受控主机完成各类恶意活动。

2. *CMD*：指对僵尸主机控制的命令集合，包括更新与攻击两类。

3. *CCC*：指僵尸网络的命令控制信道，它是整个僵尸网络实现的基础结构与核心。可记为：

$$CCC = (Topology, Protocol, Method) \tag{2-3}$$

其中 *Topology* 指僵尸网络 *CCC* 的拓扑结构，而 *Protocol* 则指网络通信协议等，包括 IRC、HTTP、P2P、Fast-flux、Domain-flux。*Resource* 指攻击者所使用的资源的集合，包括域名、代理节点、密钥、P2P 节点等。*Method* 指操作者使用命令控制的过程中所涉及的算法，包括服务器寻址、加密与编码算法、认证机制等。

对于僵尸网络所涵盖的范畴，通常容易与蠕虫、后门，间谍软件及木马[25]相

混淆，而实际上僵尸网络中的也包含了恶意软件，这部分则是属于传统恶意代码所涵盖的内容，但僵尸网络包含其特有 C&C，则是与传统恶意代码最大的区别所在，这其中的关系见图 2-1。

对于传统恶意代码中的计算机病毒，其侧重于感染计算机进行代码传播，而蠕虫则是一种具备传播能力能自我复制的恶意代码，间谍软件则侧重于窃取用户信息，而木马与僵尸网络则较为相似，但在传播控制等方面存在差异。僵尸网络与传统的恶意代码在自主性、传播性、可控性、窃密性、协同性以及危害等级方面的区别详见表 2-1。

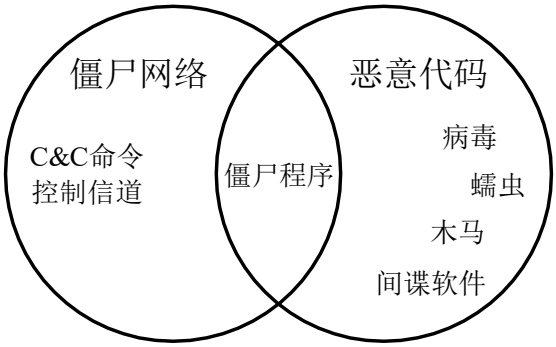

图 2-1 僵尸网络与恶意代码的关系

表 2-1 僵尸网络与恶意代码的差异

类型	自主性	传播性	可控性	窃密性	协同性	危害等级
僵尸网络	是	可控	是	可选	是	完全控制、高
计算机病毒	否	被动	否	否	否	感染文件、中
蠕虫	是	主动	否	否	否	网络流量、高
木马	是	否	是	是	否	信息泄漏、中
间谍软件	是	否	否	是	否	隐私窃取、中

2.1.2 僵尸网络的类型

僵尸网络的本质特性就是“可控”与“协同”，这两者的关键在于命令控制信道的实现，其负责完成攻击者与受控机之间的信息传递与交互。而对于安全人员，理解僵尸网络的拓扑结构与协议是进行防御的前提要求。

2.1.2.1 拓扑结构分类

僵尸网络根据其拓扑结构可以分类为中心式、P2P 式及混合结构类型。

1. 中心式结构僵尸网络：这种类型结构的僵尸网络类似于传统的 C/S 结构，

在这种结构中，受控的僵尸主机通过与特定数量的C&C服务器通信，接受命令或者更新程序，如图 2-2 所示。而按照寻址方式又可以分为静态中心结构与动态中心结构，其中静态结构是指 C&C 服务器的域名或 IP 地址是通过硬编码的方式编写固定的，而动态结构是指服务器的域名或 IP 则是通过算法动态生成，算法种子则可以是当前日期，社交热点话题等[27]。

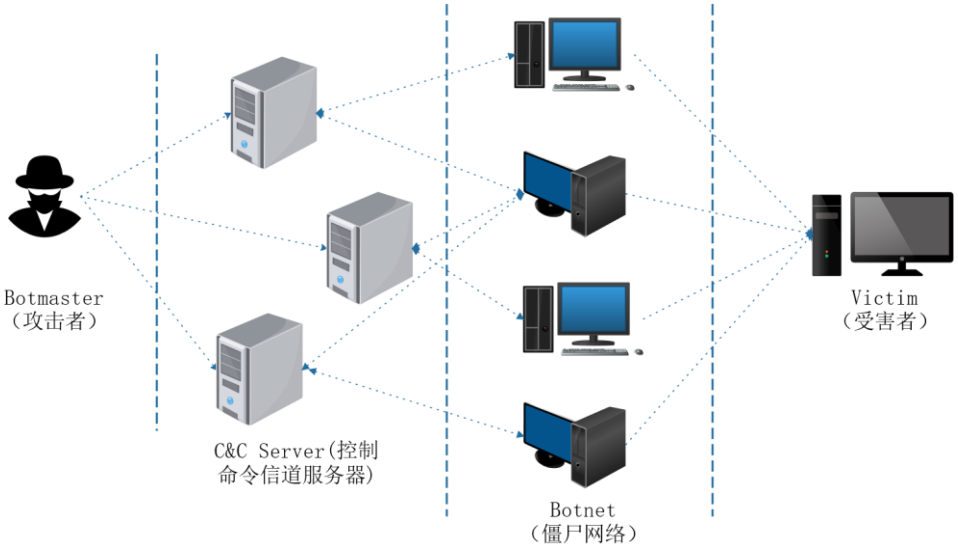

图 2-2 中心式结构僵尸网络

这种结构的僵尸网络反应快、协作性好，但对 C&C 服务器的存活性依赖较高，因此目前的僵尸网络会对通信内容用到加密混淆等方式，而为了克服C&C集中式服务器的缺点，新型的僵尸网络会用到新的逃逸技术，如 Fast-flux、Domain-flux 等技术，这也是本文重点研究的内容。

2. P2P式结构僵尸网络：由于中心式结构的僵尸网络容易出现中心节点失效问题，为了提高灵活性与存活性，一种新型 P2P 结构的僵尸网络出现了，网络中的每个主机既可当服务器，又可当客户端，如图 2-3 所示，这样就很好避免了中心节点的脆弱性问题。

这种结构的僵尸网络的组成包括两个步骤[25]：一、节点成员选择；二、将节点成员加入僵尸网络；而要实现节点的选择一般有 3 种方法：

1）寄生：指寄生在已存在的 P2P 网络上，它会从中选择有漏洞的主机作为节点，僵尸网络的大小与寄生网络的规模与安全措施有关。

2）吸附：指僵尸网络成员加入已存在的 P2P 网络中，利用该网络中的通信协议作为 C&C 通信协议。

3）专用网络：该网络中只有僵尸主机，使用自己的专用通信协议，如 Nugache[28]。

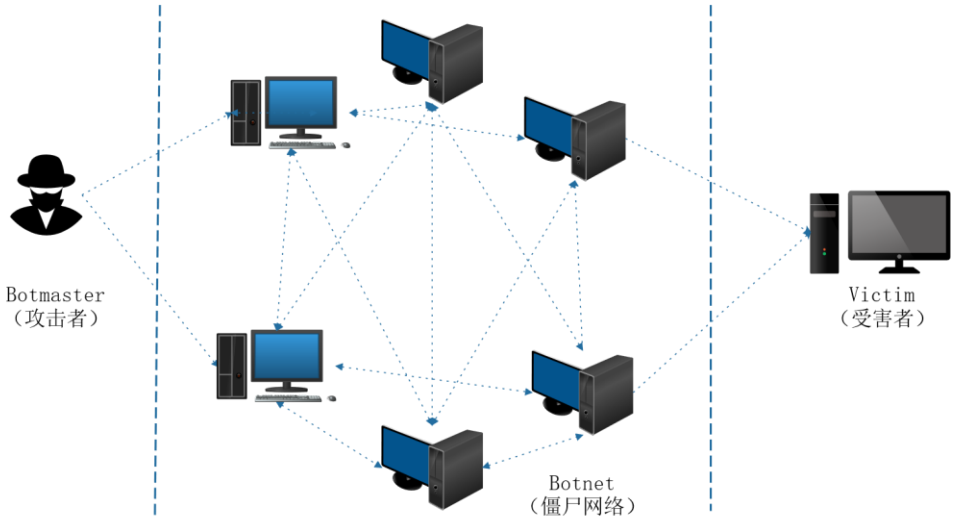

图 2-3 P2P 式结构僵尸网络

3. 混合式结构僵尸网络：

顾名思义，混合式是指同时包含上述两种结构的僵尸网络，而根据其主要结构还可划分为两个类，其结构见图 2-4：

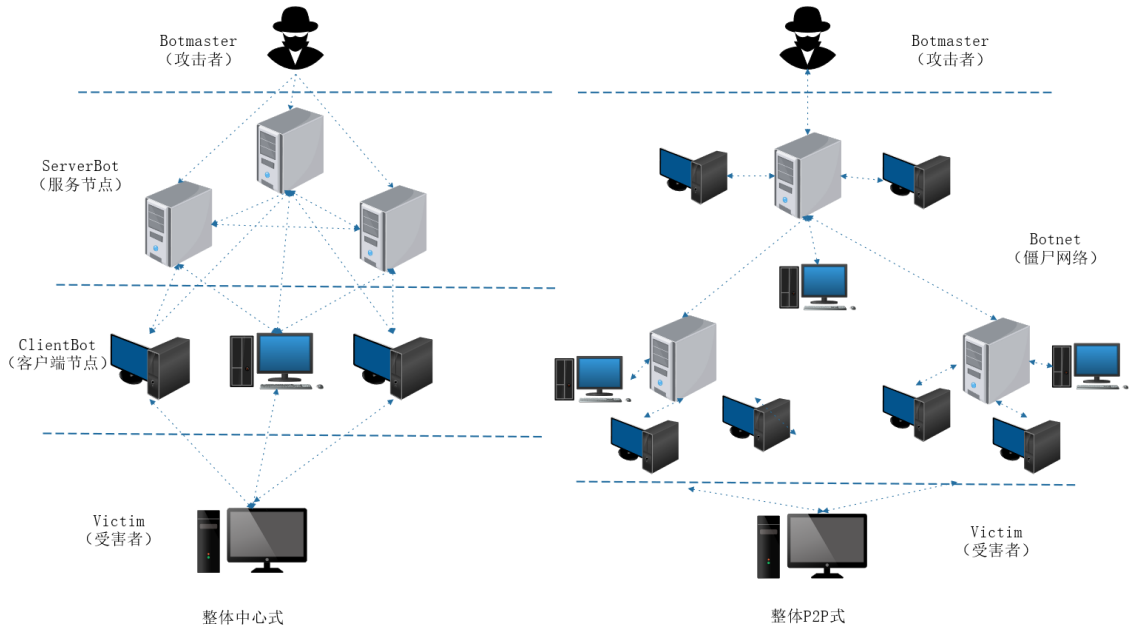

图 2-4 混合式结构僵尸网络

1). 整体中心结构：该结构同时包含局部 P2P 结构，这种结构从整体而言，逻辑上属于中心式，其服务节点则动态构成一个可扩展的 P2P 式结构网络，同时具备静态 IP，可被公网访问等特点，而受控的僵尸主机，则与该服务节点通信，获取命令等。

2). 整体 P2P 结构：这种结构呈现出局部中心式网络的特点，见图 2-4 所示，部分公网可达的节点充当服务节点，则彼此之间连通，而局部则为每一个服务节点构成中心式结构僵尸网络。

2.1.2.2 命令控制协议分类

僵尸网络的命令控制信道是其实现的基础架构，而命令控制信道则包含通信协议，根据命令控制协议，僵尸网络可划分为 IRC 僵尸网络、HTTP 僵尸网络、P2P 僵尸网络，DNS 协议僵尸网络。

1. IRC 僵尸网络：该方式较为为经典简单，它是通过借助 IRC 聊天室来实现命令的发布与接受。

2. HTTP 僵尸网络：该方式是借助 HTTP 协议来实现通信，具备较好的通用性，当其作为 C&C 协议时能够混杂在正常流量中，且还支持 HTTPS，可以轻易绕过防火墙等过滤设备，避免被 IDS[29]检测，具备较好的隐蔽性，。它主要是通过僵尸主机的程序访问特定预先发布的网页获取控制命令，从而完成隐秘通信。该方式也是目前僵尸网络主流的网络信道协议之一。除去传统 HTTP 僵尸网络硬编码的寻址方式，新型发展的访问机制包括 Fast-flxu 与 Domain-flux，即本文重点关于研究的类型。

3. P2P 僵尸网络：由于 IRC 协议和 HTTP 协议采用的均是集中式的命令控制服务器，都存在关键节点失效问题与风险，如果服务器遭受劫持或攻击，则可能致使整个僵尸网络瘫痪，因此采用 P2P 协议也成为许多僵尸网络的选择[25]，如 Nugache、Storm、Waledac、Zeus，TDL4[30]。

4. DNS 僵尸网络：Skoudis 等人[31]在 2012 年的 RSA 大会上提出越来越多的恶意软件使用 DNS 协议作为其命令控制通道。Xu 等人[32]在 2013 年提出使用 DNS 协议作为 C&C 的僵尸网络，因为 DNS 具有分布式存储、更新、传输等特点，更适合作为命令控制信道。本文基于 DNS 流量分析检测 Fast-flux 与 Domain-flux 两种僵尸网络。

2.1.3 僵尸网络的逃逸方式

为了逃避跟踪与检测，攻击者在僵尸网络中会使用各种方式，如 Fast-flux 技术、Domain-flux 技术、二进制文件混淆或加密技术、加密通信技术等。

2.1.3.1 混淆与通信加密技术

部分僵尸网络使用混淆与加密的技术增加僵尸程序的二进制代码分析难度，

还会使用代码的数字签名验证技术，Rustock[33]使用 aPLib 和 UPX 对代码压缩与混淆，使用 RC4 对二进制文件加密。而 P2P Zeus[34]则使用 MD5 算法对明文进行哈希运算，使用 RSA-2048 来计算签名，而在 2013 年的 P2P Zeus 中则使用 RC4 代替原先的循环 XOR 算法。

针对通信内容，僵尸网络也进行加密防止被检测破译。例如 Torpig[35]使用 XOR 混淆机制与 Base64 编码加密窃取的数据[25]。P2P Zeus 中针对不同类型的信息使用不同的签名机制，如在 TCP 请求中，使用接受这的标识符作为密钥。

2.1.3.2 Rootkit 技术

僵尸网络为了达到窃取数据的目的，会将 Rootkit 技术包含在其僵尸程序之中。如使用各种 API 钩子。在 Zeus 中使用了网络相关的 API 钩子。而 TDL4[30]则是将内和模式下的 Rootkit 替换了 MBR，从而在系统重启时启动。Rustock[36]通过伪装成系统驱动程序以及其他文件替换的方式，隐藏其磁盘与网络操作。

2.1.3.3 Fast-flux 技术

僵尸网络攻击者使用 Fast-flux 技术将大量的僵尸主机组成一个动态的代理网络，能够将其背后的控制服务器很好地隐藏起来。由于具备良好的隐蔽性，攻击者可使用固定的服务器，将大量恶意内容存放其中，同时便于管理。如 Storm[28]，Kelihos[37]，Waledac[38]等僵尸网络使用了该技术来逃避检测。针对该技术原理与实现机制，本文后续章节将做详细介绍。

2.1.3.4 Domain-flux 技术

Domain-flux 则是另外一种常见逃逸方式，它主要是指僵尸网络的控制者动态改变域名从而逃避检测。这其中的关键就是域名生成算法（DGA），它利用种子随机生成大量的域名，然后僵尸主机逐个发起 DNS 请求尝试通信连接，而这其中只有部分请求才会被响应。如 Torpig[27]和 P2P Zeus[34]等使用了该技术。关于 Domain-flux 的技术细节，后续章节会有进一步介绍。

2.1.3.5 协议隧道技术

由于防火墙允许 HTTP 流量通过，使用 HTTP 协议隧道可以用来隐藏僵尸网络的命令和控制通道。除此之外，因为部分中间网络设备对 IPv6 的数据识别度不够，僵尸网络则可以通过 IPv6 的隧道技术来传播 C&C 命令。

2.2 DNS 协议概述

DNS 是域名系统(Domain Name System)的缩写，它是现在互联网系统中的一项重要服务，也是一个能够将域名与 IP 地址关联映射的分布式数据库，类似与电话簿，记录域名与 IP 地址的对应关系，用户可以避免记忆复杂的数字串，能够更好地使用互联网。

2.2.1 DNS 技术概述

网络通信大部分是基于 TCP/IP 的，而这两者则是基于 IP 地址，因此计算机在网络上只能识别类似“223.5.5.5”的 IP 地址，而无法识别域名。而我们在使用互联网服务，尤其是网页浏览时，通常需要输入方便记忆且具有特定含义的域名，如“www.example.com”,而不是直接的 IP 地址，那么就需要 DNS 服务器来完成这个翻译过程，这样才能访问到对应的网页。其基本工作原理如图 2-5 所示。

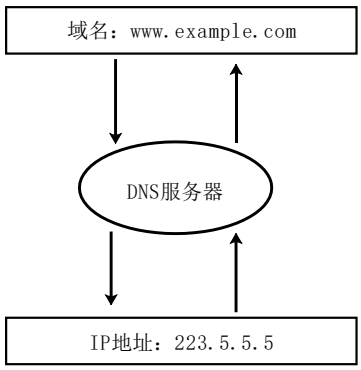

图 2-5 DNS 基本工作原理

为了解析域名，本地客户端通常需要询问本地递归 DNS 服务器（RDNS），递归服务器迭代地负责对每个区域的域名做权威解析，这就产生了一个迭代查询的过程，在域名与 IP 地址之间产生了映射。由于域名具有分层结构，本文将根据约定，将完整域名称为完全限定域名（FQDN），其中，FQDN 的第 n 层表示为 n-LD。例如“www.example.com”中的 1-LD（顶级域，TLD）是“com”，2-LD 是“example”，3-LD 是“www”。其他相关术语约定如表 2-2 所示：

DNS 协议在运输层使用了 TCP 与 UDP 的 53 端口，对于域名长度，每一级长度限制为 63，总长度则不超过 253 个字符。DNS 中查询最常见的资源记录类型是是 A 记录，即主机记录，将特定的主机名映射到对应主机的 IP 地址上。除此之外，还包括 CNAME 记录、AAAA 记录、SRV 记录等，后续将对资源记录类型作详细介绍。

表 2-2 DNS 相关术语解释

名称类型	说明	示例
根域	DNS 域名中，规定由尾部句点(.)指定名称位于根或更高级别的域层次结构	单个句点（.）或句点用于末尾的名称
顶级域	指示国家地区组织的名称	.com
第二层域	个人或组织企业注册的名称	example.com
子域名	注册域名的派生域名，或称网站名	www.example.com
主机名	域名最左侧标识主机的名称	h1.www.example.com

2.2.2 DNS 层次结构

根据 DNS 系统的工作原理，它获取域名的 IP 地址是经过分级查询，而分级查询的关键在于 DNS 的分层结构，最上层为根，从上之下依次为顶级域名、二级域名等。其示意图如图 2-6 所示。

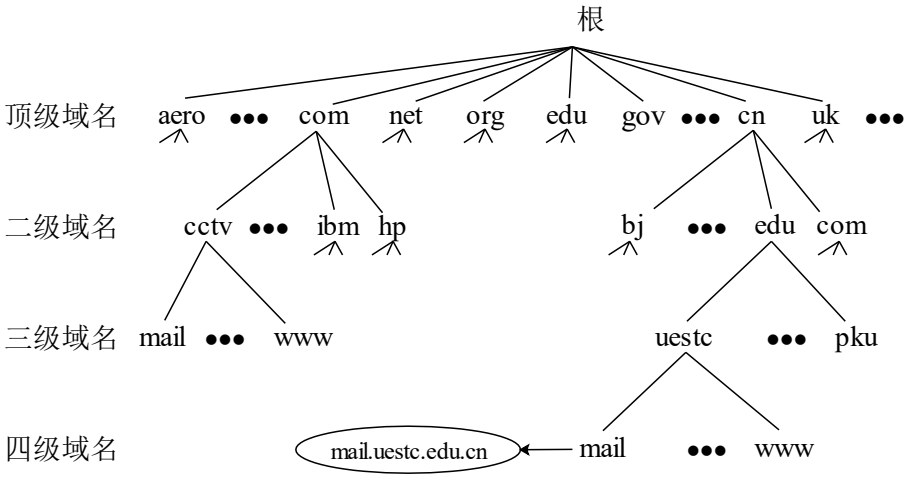

图 2-6 DNS 的层级结构

在域名解析的过程中，首先由本地 DNS 服务器（RDNS）来解析，客户机发送的查询包含 3 条信息，用来指定服务器应答的问题：

1. DNS 域名，即 FQDNs，全限定域名。
2. 查询类型，它根据类型指定资源记录或查询操作的类型。
3. DNS 域名的指定类别。

DNS 查询包含各种方式解析，如客户机可使用之前查询的缓存应答，服务器同样可以使用自身的缓存信息应答查询，服务器也可作为客户端向其他 DNS 服务

器查询解析，从而将得到的解析结果作为应答相应初始的客户机查询，这个称为递归查询。

当本地 DNS 服务器向根服务器查询时，收到的应答报文给出 IP 地址或者告诉下一步查询的域名服务器，然后本地服务器基于该应答进行后续查询，不断进行该过程，直到完成的域名得到解析。这样的过程则称为迭代查询。

一个典型的 DNS 查询过程如图 2-7 所示：

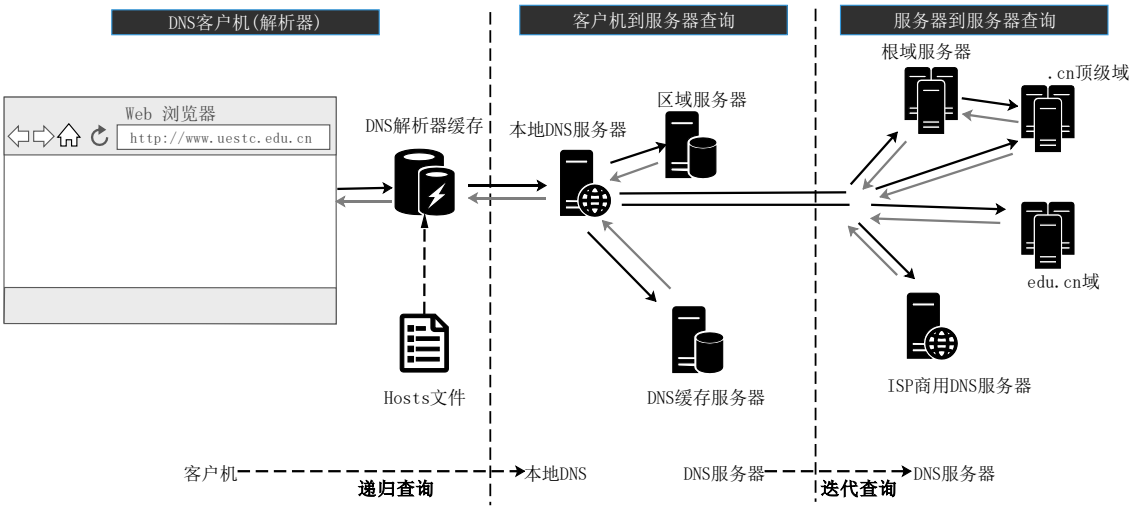

图 2-7　DNS 查询过程

2.2.3 DNS 报文格式

DNS 协议的报文包含固定长度为 12 个字节的头部字段与可变长度的 4 个正文字段，包括查询问题区域、回答问题区域、授权区域、附加区域，其中回答、授权、附加区域只存在于 DNS 响应报文中。其整个报文格式见图 2-8 所示。

图 2-8　DNS 协议报文格式

1. 头部格式

1). 会话标识：它是 DNS 报文的唯一 ID 标识，用来匹配请求报文与应答报文。

2). 标志：包含 DNS 其他说明信息，格式内容见图 2-9 所示。

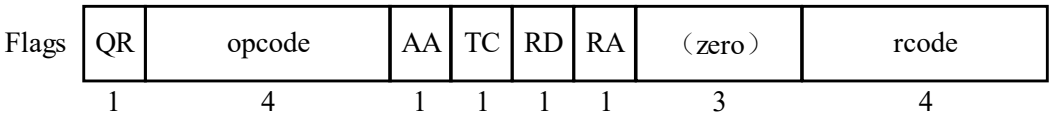

图 2-9　DNS 头部标志

关于该标志的具体含义见表 2-3 所示。

表 2-3 DNS 头部标志含义

QR (1bit)	查询/响应标志，0 为查询，1 为响应
opcode (4bit)	0 为标准查询，1 为反向查询，2 为服务器状态请求
AA (1bit)	授权回答
TC (1bit)	可截断
RD (1bit)	期望递归
RA (1bit)	可用递归
Rcode (4bit)	返回码，0 为无差错，3 为名字差错，2 表示服务器错误

3). 数量字段：该区域共计 8 个字节，其中 Questions 对应查询问题区域的数量，Answers 对应回答区域的数量，Authoritative namesversers 对应授权区域的数量，Additional recoreds 对应附加区域的数量。

2. 正文格式

1). Queries 查询问题区域：见图 2-10 所示

图 2-10 DNS 请求区域

其中查询名称表示查询的域名或反向查询中的 IP，而具体的查询类型可以参见表 2-4 所示，查询类通常为 1，表示是 Internet 数据。

表 2-4 DNS 查询类型

类型	助记符	说明
1	A	由域名获得 IPv4 地址
2	NS	查询域名服务器
5	CNAME	查询规范名称
6	SOA	开始授权
11	WKS	熟知服务
12	PTR	将 IP 地址转换为域名
13	HINFO	主机信息
15	MX	邮件交换
28	AAAA	由域名获得 IPv6 地址
252	AXFR	传送整个区的请求
255	ANY	对所有记录的请求

2). 资源记录区域（RR）：包括回答区域、授权区域、附加区域。该三部分区域拥有固定格式，其具体格式见图 2-11 所示。

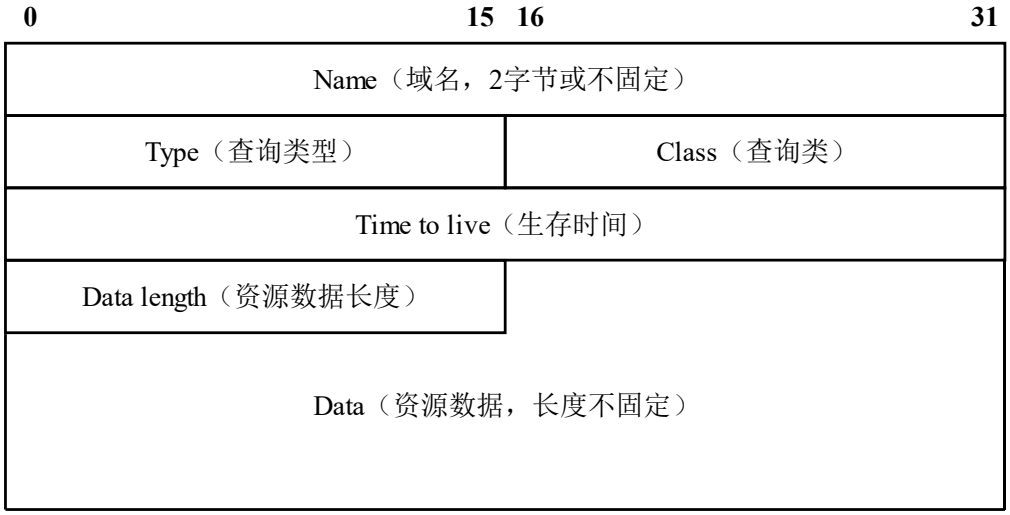

图 2-11 DNS 资源记录格式

其中域名的规范与 Queries 请求区域中的基本一样，只不过在域名字符重复时会有特别处理。查询类型同表 2-4。查询类同 Queries 区域一样。生存时间 TTL 这个表示资源记录的生命周期，在地址解析中获得该资源，该字段表示该资源缓存或使用时间，这个是基于 DNS 流量的僵尸网络分析中一个重要参数。

2.3 Fast-Flux 僵尸网络研究

在传统的僵尸网络中，攻击者会将 C&C 服务器的域名或 IP 地址通过硬编码的方式写到恶意程序中，然后受控主机通过与固定的 IP 与域名进行通信。但是这种固定的方式村活性较差，一旦防御人员通过逆向分析获得 IP，通过黑名单封锁，则会使僵尸网络陷入瘫痪，为了逃避检测，一种新的动态 IP 的技术 Fast-flux 应用到了新的僵尸网络中，给检测带了了新的挑战。

2.3.1 Fast-flux 的技术原理

在正常的 DNS 请求中，服务器对于同一域名的 DNS 查询，在较长时间范围内，多次的查询都会获得相同的结果。而 Fast-flux 技术则是一种通过动态改变域名和 IP 映射关系的一种技术，会使得在短时间的 DNS 查询使用该技术部署的域名，会得到不同的结果。

使用 Fast-flux 技术部署的网络简称为 FFSN（Fast-flux Service Network），FFSN 通过不断改变 DNS 记录，能够为一个合法域名分配多个（甚至上千）IP 地址，保证了域名的较高可用性。

为了实现快速的 DNS 变化，Fast-flux 中通常会将 DNS 查询中的 A 记录的生存时间（TTL）设置成极短的时间，有时候甚至 0 秒。同时为了保证高可用度与负载均衡，对于效率低下的节点会被从 FFSN 中剔除。

FFSN 为保证安全性与可用性，通常还会增加一层：匿名代理重定向。这种机制能够很好绕开对 FFSN 的破坏与防御。匿名代理重定向是指 Fast-flux 中轮询的 IP 地址并非直接访问后端的恶意内容服务器，而是通过广告或者恶意的 URL 等非法手段连接到受控的僵尸主机，这些主机然后通过代理重定向的方式完成真正的内容请求与转发。这项技术原本是为了实现高可用与负载均衡，在正常合法的 Web 服务广泛使用，但也同时被恶意攻击者应用到 FFSN 中。

在 FFSN 中，其背后的“motherships”是控制关键。可类比与传统僵尸网络中的命令与控制（C&C）系统，但是它还有更多的功能。“motherships”隐藏在代理网络之后，由它将恶意内容回传给请求的受控主机，还可以同时托管 DNS 与 HTTP 服务，同时也能够保证较长的存活期。

2.3.2 Fast-flux 的种类

根据 Fast-flux 的结构特点，可以将 FFSN 分为两大类：Single-flux、Double-flux。其中更复杂的 Double-flux 则是在不断改变授权名称服务器的 IP 地址的基础上，增加了一层保护。

2.3.2.1 Single-flux 类型

Single-flux 类型的 FFSN 是指只有一层变化的 Fast-flux，在 Single-flux 中，单个域名对应着一个动态变换的 IP 列表，列表的记录可能成百上千条。如普通访问“http://www.flux.com”网站时，实际上经过 DNS 解析，连接到的是 FFSN 中的代理主机，正常的请求会被重定向攻击者控制的后台“mothership”服务器，然后后台服务器会回传内容给代理服务器，再经过代理服务器返回给普通用户[40]。由于设置的极短 TTL，每次的访问可能会连接到不同的主机，这些代理主机构成的僵尸网络，即使部分主机离线或者关闭，其他代理主机也能够迅速完成取代的任务，整个网络能保持一个较高的可用度。图 2-12 展示了一个 single-flux 网络的访问示例。

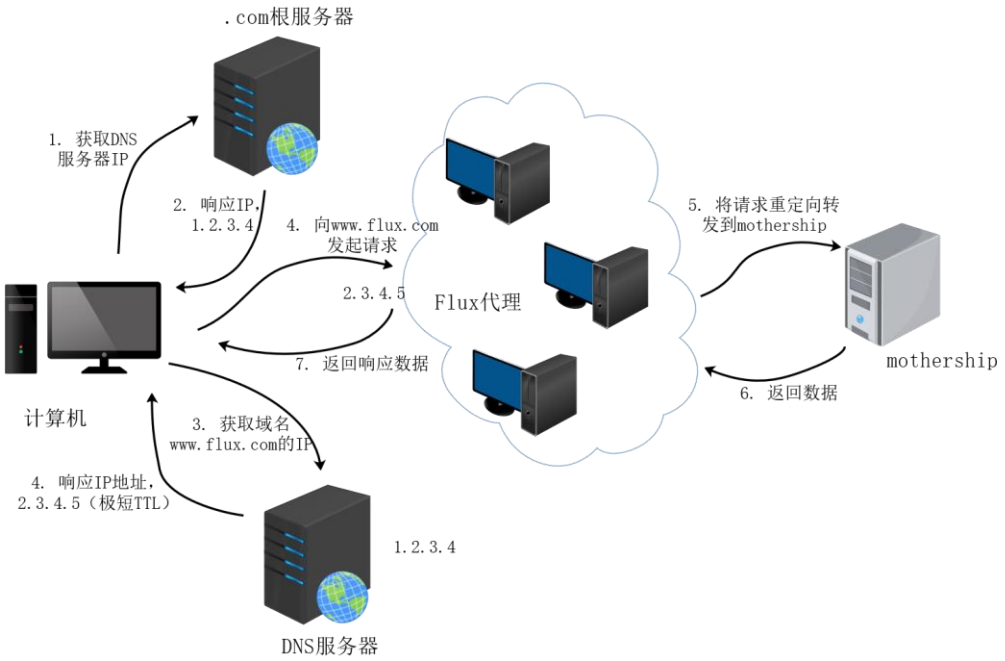

图 2-12　Single-flux 类型服务网络

由于 Fast-flux 利用的是 TCP 和 UDP 的重定向，因此除了常见的 DNS 和 HTTP 外，其他任何单个端口的正常协议都可用于代理 FFSN 中的恶意服务。这样也使得检测的难度大大增加。

2.3.2.1 Double-flux 类型

Double-flux 是一种更复杂的技术，比 Single-flux 多了一层变化。在 Double-flux 中，攻击者通常部署多个底层 DNS 域名服务器，用来解析 C&C 服务器域名，并且在顶层域名服务器中对应的底层 DNS 域名服务器 IP 地址也会随之被修改，

这样会使得每次对于负责解析 C&C 服务器的底层域名服务器 IP 都会随之变化。而攻击者为了僵尸网络的隐蔽性，对于域名服务器的 IP 地址的修改比 C&C 服务器 IP 地址的修改频率降低了很多。

在图 2-13 中，展示了 Double-flux 类型服务器网络的请求示例过程，当用户请求解析域名“www.flux.com”的 IP 地址，首先检索根域名服务器，然后客户端会向顶级域名“.com”的 DNS 服务器请求解析，这一步能够获得解析“flux.com”域名的 DNS 服务器 IP 地址。接下来会向授权服务器“ns.example.com”检索“www.flux.com”域名对应的 IP 地址。

区别于 Single-flux，这里的授权 DNS 服务器也是 Double-flux 中的一部分，也是组成了一个代理网络，使得请求授权服务器的 IP 也是经常发生变化，因为授权服务器也是与后台的 mothership 相关联，授权服务器同时是通过代理转发的方式将请求重定向到 mothership。在获得域名 IP 后发起 HTTP 或其他请求，其过程则与 Single-flux 相同，发起的请求同样会被连接到代理网络，然后通过转发请求的方式再与 mothership 通信，代理网络获得内容后再转发给客户端。

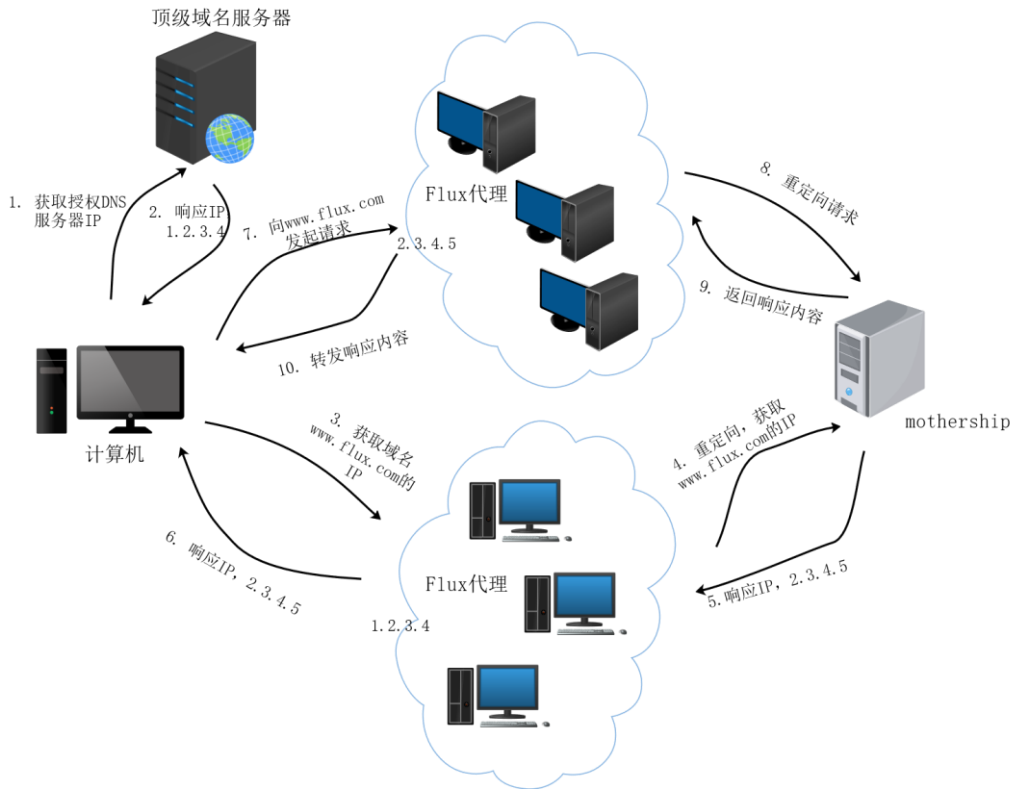

图 2-13 Double-flux 类型服务网络

2.4 Domain-flux 僵尸网络研究

由于传统僵尸网络中 C&C 服务器的域名或 IP 硬编码的特点，如果被安全人员屏蔽，则容易出现节点失效的问题，从而导致整个僵尸网络陷入瘫痪，称为“中心节点失效”。

为了解决该问题，出去 Fast-flux 技术外，还有一种称为 Domain-flux 技术也被应用到僵尸网络中来。在 Domain-flux 中，僵尸主机访问的 C&C 域名是根据一定的算法动态生成，攻击者与受控主机的通信节点也是动态变化，能够很好的逃避检测。

2.4.1 恶意软件域名

在早期的僵尸网络的程序中，通常会使用固定的一个或多个 IP 来完成通信控制。由于硬编码的 IP 容易被封锁，后来开始使用域名解析，即在恶意程序中连接的通信目的地设置为一个或多个域名。而为了大规模、低成本控制僵尸网络，攻击者通常会采用注册费用较低的国家或地区的 TLD，或者使用其他公司组织提供的免费域名。

为了针对恶意软件等非法程序使用的 IP 地址或域名，互联网上已有多个组织会通过长期的分析统计，发布 IP 地址与域名黑名单，如 DNS-BH、MDL、Spamhaus 等。

2.4.2 DGA 算法原理

在 Domain-flux 中，动态生成域名的算法称为 DGA（Domain Generation Algorithm），该算法的输入称为 Seeds，也叫种子。它通常可以是字典、随机数、日期等。大体过程如图 2-14 所示:

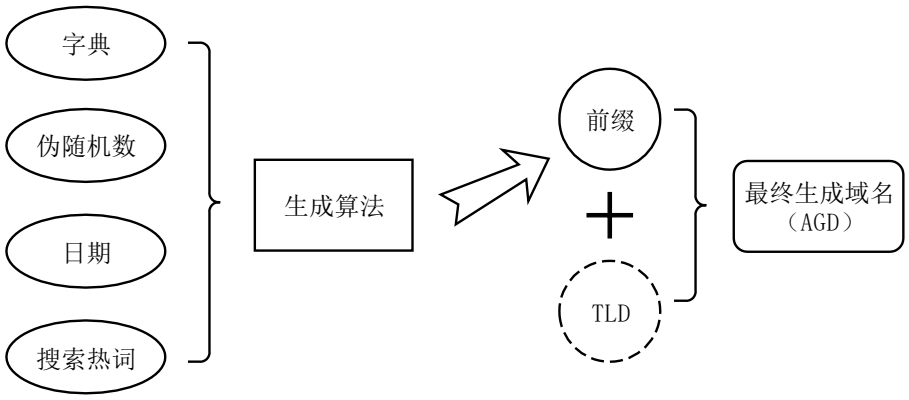

图 2-14 DGA 算法示意图

由图中可知，先由Seeds的输入，经过算法生成一串字符串，然后再增加TLD即可以得到最终域名，称为AGD（Algorithmically Generated Domain），这些生成的域名中，只要存在能够使受控主机通过域名连接到C&C服务器，则可以完成通信控制。而针对 Domain-flux 的防御需要屏蔽大量的域名，成本较高。因此Domain-flux仍然能够在僵尸网络中广泛应用。

2.5 本章小结

本章对僵尸网络的基本定义做了简单说明，从拓扑结构与命令控制协议的两个方面对僵尸网络进行分类，同时还对僵尸网络的几种逃逸方式作了介绍。后续介绍了DNS协议的基本知识，包括层次结构与报文格式。然后针对本文重点关注的Fast-flux与Domain-flux两种技术作深入的原理分析。

第三章 基于 DNS 关联映射图的 Fast-flux 与 Domain-flux 僵尸网络检测方法

结合对僵尸网络的研究，以及在对 Fast-flux 与 Domain-flux 两种技术原理的深入分析的基础上，本章主要阐述一种基于 DNS 关联映射图的 Fast-flux 与 Domain-flux 僵尸网络的检测方法。目前在针对这两个类型的僵尸网络的检测，有多种基于 DNS 流量的检测方法，但大多集中关注在 DNS 流量的局部特征或两种技术中的某一具体技术特征，从而方法缺乏更广泛的应用性与有效性。经过分析对比之前各类检测方法的不足之处，本文设计一种基于 DNS 映射关联图分析的检测方法，涵盖 Fast-flux 与 Domain-flux 两种类型的僵尸网络的检测。该方法根据 DNS 流量构建映射关联图，同时结合两个技术的恶意流量特征以及黑白名单等技术，从整体上实现针对两个类型僵尸网络的通用性检测方案，极大扩展了检测方法的应用范围，同时结合图的多维度特征，提高僵尸网络的检测精确度，有效降低误报率。

3.1 总体检测方法

根据针对 Fast-flux 与 Domain-flux 类型僵尸网络检测的功能分析，结合本文提出的检测方法，其整体功能结构体如图 3-1 所示。

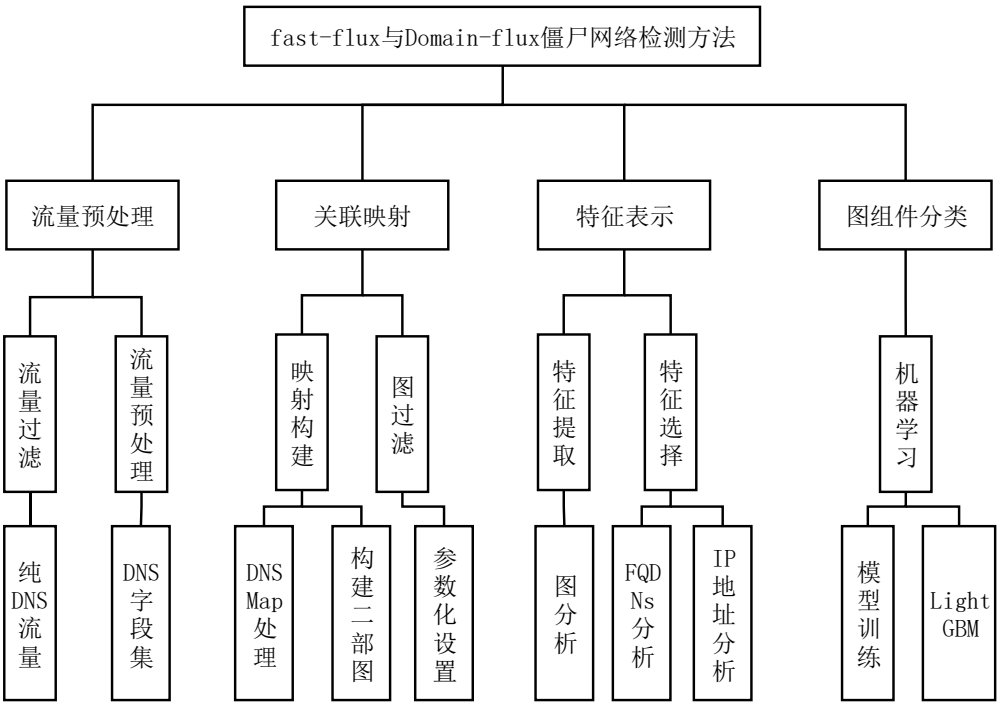

图 3-1 算法功能架构图

面向 Fast-flux 与 Domain-flux 的僵尸网络检测方法可以划分为 4 部分：流量预处理、关联映射、特征表示、图组件分类。流量预处理的过程主要为从大型网络中抓取流量，主要过滤查询响应的 DNS 流量，而为了减少后期处理的数据量，还引入白名单机制，包含流行域名的白名单以及 IP 白名单。关联映射则是利用 DNSMap 工具，根据 DNS 流量提取 FQDNs 与 IP 的关联，然后根据关联映射构建二部图。同时对 DNSMap 参数化设置，设定一定的阈值，能够完成图组件过滤。第三部分为特征表示，包括特征提取，做进一步的图结构分析。图组件特征选择，包括 FQDNs 的特征选择与 IPs 的特征选择。最后一部分而是在前面分析处理的数据上使用机器学习的方法做监督分类，使用 LightGBM 算法模型。

3.1.1 整体流程设计

前述小节设计了基于 DNS 流量的检测基本方法，本小节主要阐述该检测方法的整体设计，包括检测方法示意图与算法总计流程图。

3.1.1.1 检测方法流程

其中检测方法示意图，描述的是该方法的基本流程示意图，具体详见图 3-2 所示。由示意图可知，检测方法的大体流程可以理解为：

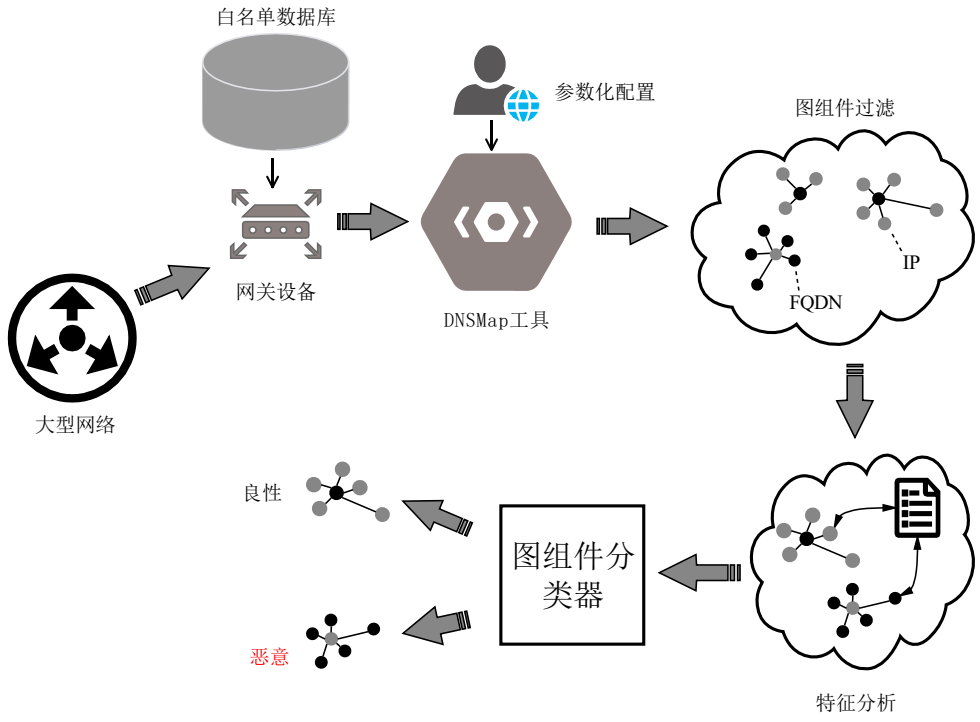

图 3-2 检测方法示意图

1. 该僵尸网络检测方法针对大型网络，经由大型网络的流量数据在网关设备处会经过过滤抓取，以作旁路分析。其中 DNS 流量会被单独过滤出来，而过滤的条件中则会引入白名单数据库，以减少待处理的数据量。

2. 过滤后的流量则有 DNSMap 工具处理，该工具的主要作用是将 DNS 流量处理形成 FQDNs-to-IPs 的图映射，即域名与 IP 的映射，这个映射的构建是根据 DNS 的查询或响应数据包。同时根据操作人员的参数化配置，还可以提供基本的图过滤，如图中所示，黑色节点表示 FQDN，而灰色的节点表示 IP，边则表示两个之间存在查询映射关系，而其中的图过滤则表示如果黑色节点关联较少灰色节点，即统一域名查询结果对应较少 IP，则为正常 DNS 映射。或者灰色中心节点关联较少黑色节点，则表示有多个域名查询响应同一个 IP，但域名数量较少，则同样为正常查询，而关于该图过滤的节点阈值，后续章节则会作详细介绍。

3. 经过图过滤的图形组件，则是由大量的二部图构成的等待分析的数据集。下一阶段的处理则是特征分析，该阶段主要完成对二部图的特征提取与特征分析。主要包括图结构分析、FQDN 分析、IP 分析、黑名单过滤等。后续章节将针对这 4 个方面做详细特征分析。

4. 经过特征处理的图组件，形成特征向量，下阶段则是由图组件分类器处理，分类器将选用 LightGBM 分类器模型。而分类器的训练过程与上述示意图不同处仅在于输入端为有标签的训练集数据。

3.1.1.2 算法流程设计

而算法总体设计图则描述的是检测方法算法执行层面的流程示意图，其具体内容如图 3-3 所示。

如下图所示，检测方法的算法设计流程可以大体分为以下几个步骤：

1. 流量采集与过滤：这部分算法的核心在于根据域名 IP 黑白名单库，其中使用的是白名单，包括全限定域名白名单即 FQDN 白名单，IP 白名单。经过配置白名单同时配合 DNS 流量过滤条件可以初步的 DNS 流量数据。

2. DNSMap 处理：这部分算法主要根据参数化的配置，使用 DNSMap 工具，将 DNS 流量经过初步处理提取，得到 FQDNs-to-IPs 的映射二部图，同时还根据配置的参数，对所提取的二部图进行过滤，进一步较少后续待处理的数据量，其中如果 经过初步判断为非敏感二部图组件，则会经过处理形成最后的分类结果的一部分数据。

3. 特征分析：在经过图组件初步过滤之后，得到大量的域名 IP 映射关联图组件，特征分析则是对这些二部图组件进行特征提取与分析，大概可以分为 4 大部

分。如图中所示，包括图结构分析，如节点边等结构特征，这个分析是从全局的角度来分析 Fast-flux 与 Domain-flux 技术在 DNS 流量中关联特征，也是本检测方法的核心之一。

其中 FQDNs 分析则是对二部图中的黑色节点即域名进行分析，分析域名的词法特性，域名的长度等各类特征。而 IPs 分析则是对二部图中的灰色节点进行分析，即对 IP 地址特征进行分析，如 IP 的所属区域等。后续部分则是黑名单分析，这部分则分为域名黑名单分析与 IP 黑名单分析，则部分数据分析的来源为域名 IP 黑白名单库，图中也有描述。

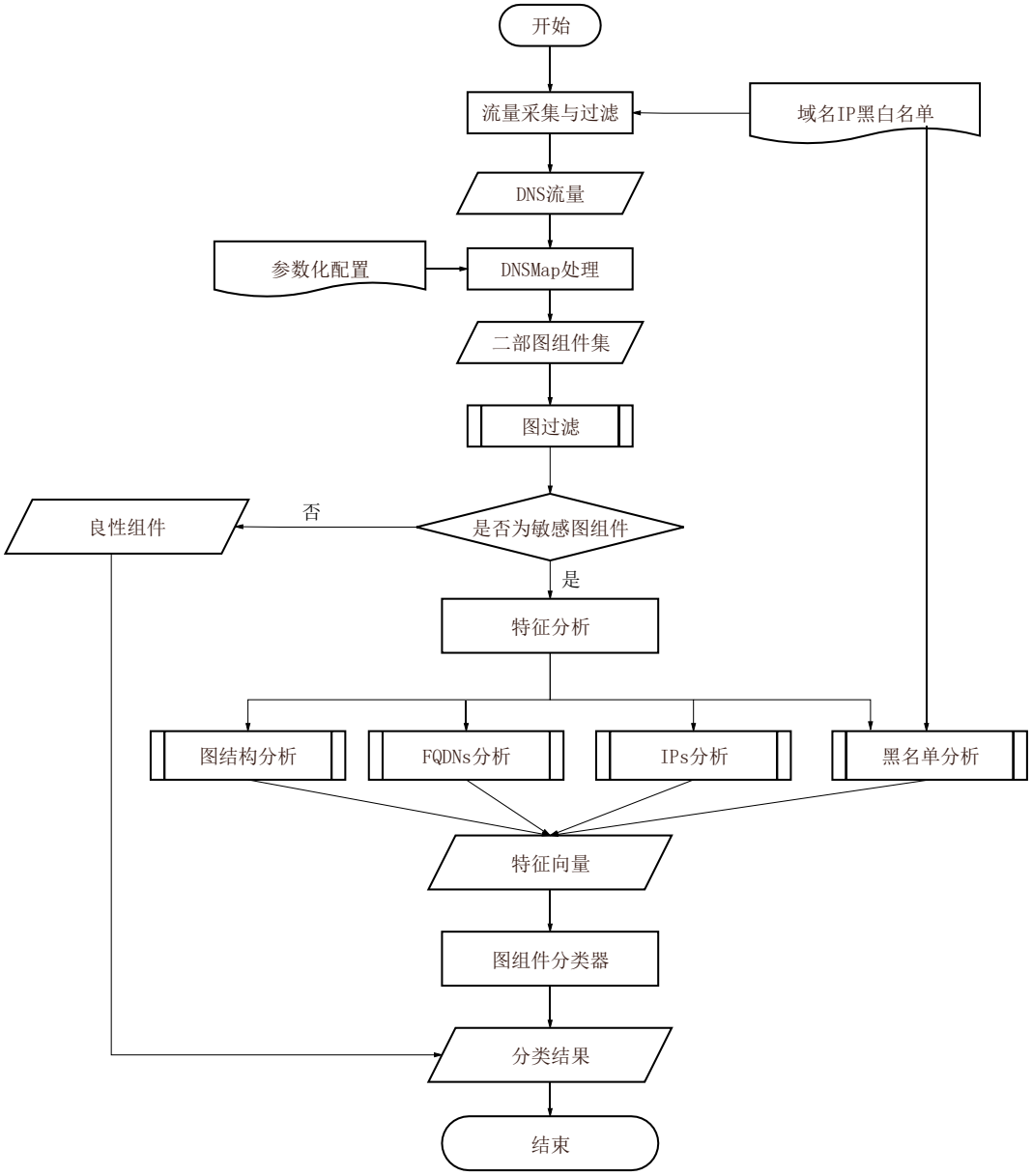

图 3-3 算法总体流程图

4. 图组件分类器：这部分的核心则是对经过特征分析处理后的数据进行良性恶意的分析，经过特征分析的特征向量，是分类器的输入数据。在这里将会应用 LightGBM 分类器对特征向量进行分析，这是对于分类器模型的训练的过程在图中尚未描述，分类器的模型训练过程则是使用有标签的数据集进行学习。

5. 分类结果：经过前序的处理过程，这个步骤则主要是对分类结果的分析，根据分类器结果，可以给出二部图组件的分析结果，即良性或恶意。同时还可以根据组件组成的结构，给出受损或可疑的僵尸网络目标，IP 对应的终端或局域网络。同时给出僵尸网络类型的倾向性分析，即 Fast-flux 或 Domain-flux 僵尸网络。至此算法总体流程结束。

3.2 DNS 映射关联图分析

本文介绍完检测方法的总体设计过程，该小节则是将重点聚焦在基于 DNS 映射关键图分析的详细设计，对检测方法中涉及的关键处理步骤再作进一步的解释，这部分主要有 4 模块组成，分别为流量过滤与预处理、DNSMap 映射处理、图组件特征提取、分类器选取。

3.2.1 流量过滤与预处理

流量预处理与过滤主要完成在大型高速网络下对实时流量的快速处理与初步过滤。而为了克服大型高速网络流量镜像数据量过大与数据包易丢失问题，需要设计响应的解决办法。主要包括 DNS 流量的获取与过滤以及初步的预处理。其中流量过滤则是在采集的全镜像流量过滤出纯 DNS 流量，则预处理则是在纯 DNS 流量的基础上提取其字段信息，包括请求包类型、源 IP、目的 IP 等，为后续特征提取做预处理。

3.2.1.1 流量镜像抓取

在进行网络流量分析，通常需要对流量进行旁路镜像，以便做进一步离线分析，因此作为网络安全人员，经常会用到一些工具，其中较为著名的有 libpcap，wireshark 等工具，其中 libpcap 比较有代表性，其工作原理大体可以理解为：当一个数据包到达网卡时，然后路由器或交换机会镜像转发到其他网络端口，即旁路机制，而在此网络接口上，配置网络抓取程序，如 libpcap。在内核层会根据过滤规则，剩下部分数据会转存到缓冲区，而这些数据则会被 libpcap 使用套接字 PF_PACKET 从链路层驱动程序中获取，然后为用户层提供 API 接口，最后则是根据 API 完成应用程序的开发，其流程示意图如图 3-4 所示。

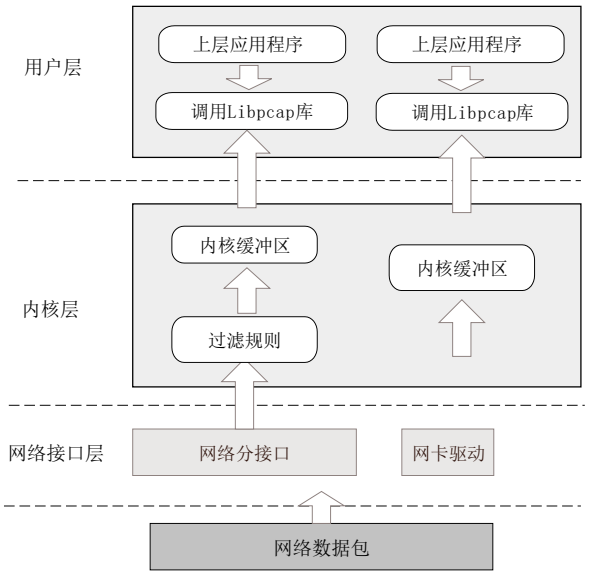

图 3-4 libpcap 流程图

而如果是大型高速网络的应用场景，则基于 libpcap 的方法则存在一定缺陷，在其过程中可以看到，CPU 需要大量时间处理网卡的数据包经过内核的数据结构队列发送到用户空间，即网卡到内核，再到内核到用户空间，这样极大消耗 CPU 计算资源以及降低处理效率。因此需要寻求新的解决方案，一种新的工具引起行业人员的关注，那就是 PF_RING。

PF_RING[41]是 Luca 研发的一种基于 Linux 内核级别的高效数据包捕获技术，拥有完整开发接口的告诉数据包捕捉库，它与 libpcap 十分相似，但性能更优异。同时能够完全满足本设计方案中的目标要求。

3.2.1.2 DNS 流量过滤

在大型高速网络中，网关设备通常是较高的流量负载，甚至可能是 10Gbit+/s, 而如果针对全部流量做镜像分析，则需要专业的高规格底层硬件设备，而本课题的需求点在 DNS 流量，因此在流量的抓取阶段就考虑使用 PF_RING 技术，同时在抓取流量设置好过滤条件，只捕获 DNS 流量，这样可以极大较少后续待处理的数据量，同时也能够适用更多的网络环境。

根据第二章关于 DNS 协议的介绍，可知 DNS 协议是运行在 UDP 协议之上，使用端口号 53。因此在过滤 DNS 流量时，可以根据特定端口过滤流量，通过对 PF_RING 设置参数和运行指定脚本，可以得到纯 DNS 流量，同时结合 DNS 协议分析，根据其报文格式字段，可以解析提取到 DNS 的相关信息，包括查询或响应、请求域名、IP 地址等。

3.2.1.3 白名单过滤

对于恶意的网络行为的检测，通常会使用到黑白名单，这是一种较为简单高效的方式，同时也要认识到这种方式的局限性，才能够更好地使用它。

在安全领域中，服务厂商与安全人员通常会通过部署蜜罐、垃圾邮件陷阱、恶意软件检测等方式获取恶意代码中产生的一些域名或 IP 的黑名单，同时也采用统计、测试等方式获取行业中合理使用的域名或 IP 的白名单。这个名单通常可用于僵尸网络检测的初步过滤或数据标记，其中 CDN 网络中对于 DNS 协议的应用与 Fast-flux 有相似之处，即源站与 CDN 节点位于不同的 IP 节点上，对于同域名资源的请求，会获得不同 IP 地址的响应。而通过 ICP 备案系统搜集各大厂商 CDN 域名，经过整理，可得到 CDN 白名单[42]，使用该名单是一种有效的初步过滤方案。

本研究课题中，旨在通过白名单机制对获取的 DNS 流量实行二次过滤，过滤其中包含正常的域名或 IP 的流量，从而较少后续待处理的数据量。在一般的网络环境下，恶意流量往往只是占据其中极小的一部分，而大多数都是正常的访问流量，白名单过滤是一种极为有效减少数据量的方法，同时由于其实现算法简单，也能够完成更为高效的处理。其过滤的算法设计如图 3-5 所示。

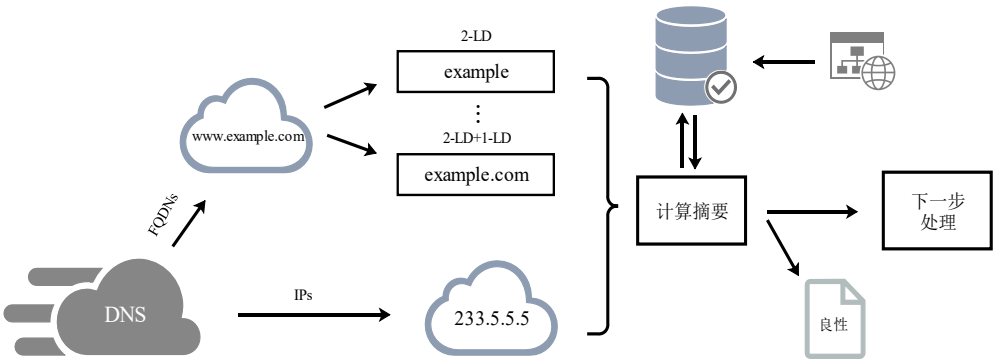

图 3-5 白名单过滤机制

如上图所示，在 DNS 流量的基础上获得其 FQDNs 与 IPs，然后对域名进行连续分拆处理，之所以这样，是因为如果只是对域名与白名单使用字符串匹配算法来比对的话，效率较低，即使采用 KMP 算法也是 O(m+n),仍然不理想。因此这里根据域名的结构特性进行了算法优化，将域名进行分拆，然后采取相邻的串拼接，最后再对齐计算消息摘要，然后再与白名单数据库匹配，获取结果。

同样地对 IP 也是使用计算摘要再匹配。对于白名单数据库，本方案中采取一种动态更新的策略，即根据网络公开的白名单数据及时更新，能够保证检测方法的实时有效性。关于公开的白名单数据库，其引用于目前该领域主流的网站，其

中 Alexa 等网站，对于中国特色的域名则没有效果，因此引入 chinaz 网站排名生成白名单[43]，具体信息可以参见表 3-1。

表 3-1 FQDN 与 IP 白名单源表

网站	包含 FQDN 或 IP	策略
malwaredomains.com	FQDN	前 500 域名
malwaredomainlist.com	FQDN	18 个常见的 CDN 的 2-LD
spamhaus.org	FQND	前 500 个 2-LD
zeustracker.abuse.ch	FQND	前 1000 个域名
alexa.com	FQDN	前 1000 域名
dmoz.org	FQDN	前 3000 个 2-LD
chinaz.com	FQDN	中国特征域名排名前 100
gooddata.com	IP	IP 白名单库

3.2.2 DNSMap 映射处理

在网络流量的初步处理之后，则需要下一阶段的映射处理。而互联网中的 DNS 映射的差异性很大，引入 DNSMap 工具的主要目的则是根据 DNS 流量在时间与空间以及全局关联层面的特征，依据先前实验的最佳参数化，在一定的限定时间周期内，将可疑 DNS 映射分析为二部图，以便后续从图的结构特性的角度做进一步分析与过滤。

3.2.2.1 DNSMap 介绍

Berger A 等人[44]提出构建 DNSMap 工具，它能够基于网络捕获的 DNS 查询响应数据而自适应地构建映射关系，同时也解决了可见性问题与 DNS 通配符问题。同时可根据邻近 IP 承载类似服务时，提供聚合视图，能够很好解决 CDN 与云服务厂商中 IP 灵活复用检测的问题。

DNSMap 实际上是通过将域名映射压缩到特定的 IP 范围，并合并相关的 IP 范围，是一种能够有效存储 DNS 映射的模型，它能够针对整个域名而不仅仅是域名中最重要的后缀。

在假定相同 IP 地址托管的 FQDN 通常是相似的，且相邻 IP 地址托管类似的 FQDN 集合的下，通过 DNSMap 可以得到相同 IP 范围的足够相似的 FQDNs 集群，并将可疑映射形成二部图。

3.2.2.2 映射提取与过滤

在本方法中，会使用到 DNS 的一些术语，如完全限定域名简称为 FQDNs，而主要的映射则是 FQDN 与 IP 的关联映射。采用定期更新的域名列表来识别，如 google.com.hk 中的 T-LD 是 com.hk,而非 hk。映射提取的方法基于一定的假定条件，即 DNS 映射存在一定的“冗余量”，表示发生映射冲突的域名通常是相似的，而相邻的 IP 地址则承载相似的域名集。

首先需对域名处理，我们定义域名差异度(DomainDivergence),简称 DD，它的范围在 0-1 之间，表示两个域名相似度。其计算公式如下：

$$\omega_\lambda = \mathrm{Max}(|X_\lambda|,|Y_\lambda|) \bullet \frac{1}{\alpha+\lambda} \tag{3-1}$$

$$dd_\lambda = \begin{cases} 1 & \text{if } \lambda > \mathrm{Min}(|X|,|Y|) \\ (1\text{-}\mathrm{LR}(X_\lambda,Y_\lambda)) \bullet \omega_\lambda \end{cases} \tag{3-2}$$

$$\Omega = \sum_{\lambda=0}^{\mathrm{Max}(|X|,|Y|)} \omega_\lambda \tag{3-3}$$

$$DD = \frac{\sum dd_\lambda}{\Omega} \quad \in [0,1] \tag{3-4}$$

其中 X、Y 表示两个 FQDN，而 X_λ 表示的是 X 的 λ-LD，$|X_\lambda|$ 表示的是 X_λ 的长度，|X|表示的是 X 的层级数，而 α 则是预设参数，初始化设定为 2。其次，根据上一步骤计算的域名相似度，就可以使用聚类算法对域名进行聚类，将相似的域名聚到同一类，形成一个 FQDN 节点。聚类算法采用 K-means，具体如下：

算法 3-1：域名聚类算法步骤

```
输入：域名 domains
输出：结果集 clusters
    1)  cluster = {}
    2)  m = DomainMedian(domains)
    3)  good = {d ∈ domains if DD(m,d) ≤ θ }
    4)  bad = { d ∈ domains if DD(m,d) > θ }
    5)  if good is empty:
    6)      bad1,bad2 = k-Means(domains)
    7)      clusters += DomainCluster(bad1)
    8)      clusters += DomainCluster(bad2)
    9)  else if bad is empty:
    10)     clusters += (m,good)
    11) else:
    12)     clusters += DomainCluster(good),DomainCluster(bad)
    13) return clusters
```

在这里，采取 k=2 的参数设定，首先能够将符合要求的聚合到一类，然后再处理剩下的另一类别，一直下去直到聚类完成。

而在域名聚类完成后，则还需要构建映射图，在这里称为 IPBlock，它是一些 FQDN 节点与 IP 节点的集合，即这些节点间存在关联映射，在图中反映的则是一个局部的二部图实体，而后续的特征提取的输入则是这样的 IPBlock，而这些图组件最后才经过分类，识别是否为可疑的活动组件，即是否与 Fast-flux 或 Domain-flux 相关。

这其中关于构建图组件的核心算法则是 DNSMap 存储算法，具体如下

算法 3-2：DNS 映射存储算法步骤
输入：(ip,dname)映射集 输出：IPBlock 结果集 res 1) ipb = dnsmap.GetContainingIPBlock(ip) 2) if not ipb: 3) ipb = new IPBlock() 4) if ipb.acceptsDomain(dname): 5) setIpActive(ip); 6) res = (OK,0.0) 7) if not clusters: 8) addCluster([dname]);setIpActive(ip); 9) res = (New,0.0) 10) closestCluster,dd = ipb.findClosestCluster(dname) 11) if dd ≤ 0: 12) closestCluster.add(dname);setIpActive(ip) 13) res = (OK,dd) 14) addCluster(dname);setIpActive(ip) 15) res = (New,dd) 16) return res.

3.2.3 图组件特征选择与分析

该阶段主要是为上阶段处理得到的图形组件选定提取一组判别特征，而这些特征是基于恶意 DNS 流量特征理论知识以及长期研究的经验依据。针对 Fast-flux 与 Domain-flux 两种类型的僵尸网络，选取了 4 个方面的特征集合。

3.2.3.1 图结构特征

该部分则是主要提取二部图节点连接的结构属性特征，以便从图的结构角度来分析 Fast-flux 与 Domain-flux 两种类型的僵尸网络在 DNS 请求响应中的连接模型。提取特征如表 3-2 所示

对于图组件特征的选取是基于 Fast-flux 与 Domain-flux 在连接模型上的经验分

析，能够从一定程度反映其实际特性。

节点数量是图组件结构中的一个特征，在分析恶意 DNS 活动时，更多的节点构成的更大的图形更有可能与恶意的 DNS 映射相关联。

节点度数则是反映某个中心节点关联度的情况，在 Fast-flux 中 IP 的快速变化会产生一个节点，即同一域名在某个时间段内会映射到多个 IP 地址，在图中则是反映出 FQDN 节点的度数偏高，因此度数的特征选择是重要一环，同样度数的特殊变化在 Domain-flux 中也有反映。

表 3-2 图组件结构特表

类别	标号	描述
节点数量	G1	FQDN 节点的数量
	G2	IP 节点的数量
节点度数	G3	FQDN 节点的最大度数
	G4	FQDN 节点平均度数
	G5	IP 节点最大度数
	G6	IP 节点平均度数
节点间性	G7	FQDN 节点间连接性
	G8	IP 节点间连接性

节点间性则是指同类型的聚集图中节点之间的连接性，即关联性。如在 Fast-flux 类型中的会出现多个 FQDN 节点，而这些节点间可能存在映射到同一个 IP，则表示这些主节点则是存在连接，从整体上看，这个特征表现出的则是图节点之间的连接紧密度。

3.2.3.2 节点特征

在构建的二部图中，除了图组件结构会形成等待分析的特征外，构成图的主要元素包含节点，同样也是分析的重要组成部分。在图中，节点分为两类，FQDN 节点与 IP 节点，因此节点特征的提取也分为两部分。

1. FQDN 节点

FQDN 节点即域名节点主要反映的是在 DNS 查询响应中域名体现的特性，而这部分关注的就是这些特性的选取，重点关注的是体现恶意 DNS 活动相关的特性，尤其是 Fast-flux 与 Domain-flux，选取的特征见表 3-3 所示。

Whois 信息是反映特定域名是否被注册或注册商家的详细信息，在这个信息维度上正常与恶意域名也会反映出不同的特征。通常正常的域名注册商家为提高其知名度、可信度，会填写更多更完整的域名信息，并且也会对其信息进行更新，

且那些注册时间较长的域名能够获得更高的可靠度。而恶意域名处出于非法目的，则是会尽量减少暴露的信息，而且注册的域名通常时间较短或无法查询到。这些特征也是作为检测的一个重要维度。

FQDN 层级则是域名在分层上体现的特征，僵尸网络中的恶意域名出发点通常不在推广与便于记忆，因此在域名的设计上会体现出正常域名不具备的特征，尤其是使用 DGA 产生的域名，如域名的层级数量会较多，TLD 与 2-LD 的种类数量可能会较多，这些特征都能够在一定程度反映到 Domain-flux 类型中的域名，同样也是检测不可缺少的方面。

表 3-3 FQDN 节点特征表

类别	标号	描述
Whois 信息	F1	Whois 信息创建时间
	F2	Whois 信息更新次数
	F3	Whois 信息完整度
FQDN 层级	F4	FQDN 节点的最大层数，即 n-LD
	F5	FQDN 节点的平均层数
	F6	FQDN 中 TLD 的种类数量
	F7	FQDN 中 2-LD 种类数量
FQDN 词法	F8	FQDN 中 TLD 的频度
	F9	FQDN 字符的最大长度
	F10	FQDN 字符的平均长度
	F11	2-LD 字符的最大长度
	F12	2-LD 字符的平均长度
	F13	3-LD 字符的最大长度
	F14	3-LD 字符的平均长度
	F15	FQDN 字符中的最多数字数量
	F16	FQDN 字符中平均数字数量
	F17	FQDN 字符中包含单词的平均数量
	F18	2-LD 包含单词的平均数量
	F19	FQDN 节点 2-LD 字符的重复度

FQDN 的词法则是重点关注到域名的字符串在词法上体现的特征，这个通常是恶意域名检测研究的重要方向。通过前面章节对于 DGA 算法的分析，可以了解到，在 Domain-flux 类型的僵尸网络中，产生的恶意域名相较正常的域名在统计学上体现出较大的差异，如字符的随机性更强，字符的长度分布范围更广，同时包含正常的单词字符更少，TLD 方面，恶意服务商则是考虑费用成本等原因，

使用字符的更为不常见，2-LD 也同样体现出不一样的特征，如在同一图组件中，一个 IP 节点连接的多个 FQDN 节点，其 2-LD 字符的重复度较低也是恶意域名的特征。如其他方面，如 FQDN 中数字的数量，在恶意域名中也是有不一样的表现。这些域名词法层面的统计特征是作为检测的重要方面。

2. IP 节点

IP 节点，即图中的灰色节点，它表示的是在 DNS 查询响应中对应的 IP 地址，IP 地址，无论是 IPv4 还是 IPv6，其字符构成不包含可分析的词法含义，但其包含其他特征同样也是分析的重要维度，选取的特征见表 3-4 所示。在基于恶意服务通常依赖更多样化的 IP 地址集的条件下，IP 的多个特征与 Fast-flux 类型的僵尸网络体现出密切的关联性。

IP 地址由全球五大区域因特网注册机构负责管理，这些机构负责 IP 地址、ASN（自治系统号）的分配，如常见的亚太互联网信息中心(APNIC)。这些管理机构查询到 IP 地址的 Whois 信息，包括 IP 地址的状态：是否注册，能否被转移；最后一次更新时间；所属国家等。正常良性的 IP 地址通常管理得较为完善，其相关信息也更为完整、固定。而恶意的行为，如 Fast-flux，其 IP 地址所属国家涉及范围更为广泛，这样是为了逃避检测，所以与正常的 IP 地址有所区别。包括 Whois 的其他信息，恶意的 IP 往往信息量极为缺乏。

表 3-4 IP 节点特征表

类别	标号	描述
Whois	I1	Whois 状态
	I2	Whois 更新时间
	I3	IP 所属国家数
AS	I4	ASN 数量
	I5	ASN 数量与 IP 比值
子网	I6	IP 子网 IP/8 的比率
	I7	IP 子网 IP/16 的比率
	I8	IP 子网 IP/24 的比率
	I9	IP 子网 IP/8 的熵值
	I10	IP 子网 IP/16 的熵值
	I11	IP 子网 IP/24 的熵值

AS 是 Autonomous System（自治系统）的简称，恶意服务其依赖于更广泛，多样化的 IP 地址集，会导致其涉及的 ASN 数量更多，ASN 与 IP 数量 N 的比值

P_{IP} 也会产生响应的变化，其计算公式如下：

$$P_{IP} = \frac{Num_{ASN}}{N} \tag{3-5}$$

IP 地址的使用中需要对子网进行划分，可分为三大类 IP/8、IP/16、IP/24。不同子网的划分比例表示 IP 划分的多样性的程度。而 IP 子网的熵也是是量化其划分的多样性与分散程度，熵值越小则表示划分越有序，以 IP/8 为例，熵值的计算公式如下：

$$IP/8_{entropy} = \frac{-\sum_x p(x)\times log_2 p(x)}{log_2 |P|} \tag{3-6}$$

$$p(x) = \frac{count(x)}{|P|} \tag{3-7}$$

其中，P 表示 IP 地址集，x 表示 IP/8，p(x)则表示 IP/8 在 P 集合中的比例。

3.2.3.3 连接边特征

在关联的二部图中，FQDN 点与 IP 点是通过边连接起来，一条连接边意味着两个点之间产生了 DNS 查询，连接边的特征则是指在 DNS 查询响应中产生的特征，其中重点关注的则是 TTL 值。

TTL（Time To Live）值表示的是域名的缓存时间，这是一个可以通过设置修改的数值，通常正常良性的域名会将 TTL 设置成较长时间，根据文献[43]显示,几乎 80%的良性域名（包括 CDN）会将 TTL 设置大于 600s，而在 Fast-flux 僵尸网络中，超过 90%的域名的 TTL 小于 600s。

而在 Fast-flux 僵尸网络中，由于网络节点中的受控主机彼此之间性能、网络状态差异较大，因此对于 TTL 特征的选择需要考虑的更为全面，选择的特征如表 3-5 所示。

表 3-5 TTL 特征表

类别	标号	描述
TTL	T1	A 记录中 TTL 平均值
	T2	A 记录中 TTL 方差
	T3	A 记录中 TTL 中位数
	T4	NS 记录中 TTL 平均值
	T5	NS 记录中 TTL 的方差
	T6	NS 记录中 TTL 的中位数

3.2.3.4 黑名单特征

对于安全领域的研究方法，黑名单是一种常见且有效的过滤手段，而本文中检测方法并且直接使用黑名单过滤，因为该方法的输入为二部图组件，对于黑名单，可产生节点是否被标记的属性，从而与其他属性特征统一起来，以便更好地做进一步处理与分类。而其中节点则包含 FQDN 节点与 IP 节点，针对这两种情况，选择的特征如表 3-6 所示，黑名单的公开来源如表 3-7 所示。

表 3-6 黑名单特征表

类别	标号	描述
FQDN	B1	图组件中被标记的 FQDN 节点数量
	B2	2-LD+TLD 被标记的数量
	B3	FQDN 节点被标记的最大度数
	B4	图组件中标记的节点与总节点数量比值
IP	B5	图组件中被标记的 IP 节点数量
	B6	IP 节点被标记的最大度数
	B7	图组件中标记的节点与总节点数量比值

表 3-7 黑名单来源表

来源	FQDN	IP
malwaredomains.com	√	
malwaredomainlist.com	√	√
spamhaus.org	√	√
zeustracker.abuse.cn	√	√
kisarbl.or.kr	√	
cyber-ta.org	√	
siteadvisor.com	√	
mywot.com	√	
domaincrawler.com	√	
domains.com	√	
wapawet.cs.ucsb.edu	√	
phishtank.com	√	
safeweb.norton.com	√	
ipvoid.com		√
urlvoid.com	√	

3.2.4 分类器选取

在图组件的多粒度特征选择完成后，会形成特征向量。在此基础上需要应用环境，设计能够满足设计模块的分类方法。本文中需要解决大型高速网络中产生的海量数据，因此更多考虑到分类方法的高效性，同时能够更好利用高性能平台。因此本文选择 LightGBM 方法作为图组件的分类方法。

LightGBM 是一种快速、高性能、分布式的优秀梯度提升框架，它是由微软在 2017 年开源，可用于排序、分类、回归等机器学习任务。它是基于决策树算法，采用最优的叶明智策略分裂叶子节点，在不降低准确率的前提下，相比主流的分类算法速度提升了 10 倍左右，占用的内存反而下降了 3 倍左右。

相比于热门的 XGBoost 算法，之所以选择 LightGBM 算法，是因为其相较于 XGBoost 算法，在树木的生长与划分点搜索上采用了不同的算法，在内存占用与性能优化上能够取得更好地效果，其对照表如下所示。

表 3-8 XGBoost 与 LightGBM 对比表

	XGBoost	LightGBM
树木生长算法	按层生长的方式 利于工程优化，但对学习模型效率不高	选择最大收益的节点展开，在更小的计算代价上选择决策树 控制树的深度及叶子节点的数据量，减少过拟合
划分点搜索算法	对特征预排序的方法	直方图算法：将特征值分成许多小筒，进而在筒上搜索分裂点，减少了计算代价与存储代价，获得更好性能。数据结构也发生变化也对效率有所影响
内存开销	8Byte	1Byte
划分的计算增益	数据特征	容器特征
高速缓存优化	无	在 Higgs 数据集上加速 40%
类别特征处理	无	在 Expo 数据集上速度快了 8 倍

3.3 本章小结

在针对 Fast-flux 与 Domain-flux 两种类型僵尸网络的深入研究的基础上，结合大型高速网络的应用场景，提出了基于 DNS 关联映射的多粒度特征分析僵尸网络检测方法，且同时面向 Fast-flux 与 Domain-flux 两种类型。首先根据需求，设计

整体的检测方法流程与整体架构图，然后针对高速大型网络提出结合使用 PF_RING 工具完整流量的初步过滤，随后根据总体方法设计，详细描述了 DNSMap 映射处理过程设计，最后重点描述了图组件的特征提取与分析过程。

第四章 高速网络下面向 Fast-flux 与 Domain-flux 僵尸网络检测原型系统的设计与实现

第三章中对本文提出基于 DNS 映射关联图的多粒度特征分析方法，可面向 Fast-flux 与 Domain-flux 两种类型的僵尸网络实现检测的方案做了详细介绍，而本章的内容则是根据第三章提出的检测方法，结合实际应用环境，设计其原型系统，并对原型系统的网络部署，总体架构设计以及各个重要模块的设计与实现做出详细论述。

4.1 系统设计目标

4.1.1 设计难点

在目前的网络安全领域，对于僵尸网络的检测研究已经发展到一定阶段，检测方法多关注在对于离线数据集的分析结果，而对于高速网络环境下的僵尸网络检测系统的设计与应用则相对较少。设计符合实际网络环境中的检测要求的系统面临着诸多问题，本课题提出了基于 DNS 映射关联的图分析检测方法，而要根据该设计思路，设计原型系统，其中涉及的难点的问题如下：

1. 高速网络流量的捕获问题

如今的网络技术发展迅速，网络基础设施也在快速地更新换代，百兆、千兆网络甚至更大的网络已经逐渐普及，而在此网络环境下，单位时间内产生的数据量将十分惊人。而对于如此海量的流量数据，如何做到完整的捕获是一个难题。除此之外，设计的系统还必须考虑到不应对待检测的正常系统形成干预与影响。

2. 同时面向两种类型僵尸网络的检测问题

在僵尸网络检测的研究中，大多数课题的研究方向选择 Fast-flux 或者 Domain-flux 两种类型之一，而设计一种新的检测方法与系统同时覆盖这两种类型的僵尸网络则必然会极大增加复杂度，需要涉及的知识面则更为广泛，检测方法的可行性与原型系统的实用效果将是一个巨大问题。

3. 海量数据下检测方法的高效性

本课题的原型系统的应用环境为高速网络，产生的数据也是海量，如何对数据进行预处理与过滤，如何对过滤后的大量数据进行分析操作，在数据量增加之后，检测方法的高效性成为了又一个问题。系统引入新的设计思路显得尤为重要。

4. 复杂系统多模块的集成与控制问题

原型系统采用多模块设计，而这样减小耦合度的方式则会使得系统的复杂度急剧上升，各个模块如何划分、如何单独设计成为首要问题。随之而来的是模块间复杂的数据交互、控制问题。保证系统能够长时间稳定可靠运行同样面临着挑战。

5. 系统运行开销问题

在原型系统的理论设计阶段，在考虑检测方法实现的效率、准确率外，还有一个问题则难以回避，系统实际运行需要的硬件条件以及开销问题。在海量数据情况下，如何确保系统整体的计算资源的消耗在一个可控合理的范围内也是难点。同时减少系统开销则是进一步优化的目标。

4.1.2 系统目标

根据前序章节介绍的相关背景知识可以了解到，如今的互联网形势下，僵尸网络也发生了新的变化，不少新的逃逸技术被应用到其中，如 Fast-flux 与 Domain-flux，本课题深入研究这两个技术的基本原理，在僵尸网络中两者均是对 DNS 的恶意滥用，在提出一种基于 DNS 映射关联图的分析方法，且涵盖两种类型僵尸网络的检测后，需要结合具体的应用场景，设计实现大型高速网络环境下的僵尸网络检测系统，主要完成目标如下：

1. 实现对使用 Fast-flux 技术的该类型僵尸网络的检测

对于 Fast-flux 的技术研究，帮助我们理解其工作原理，本章设计的检测方法能够结合 Fast-flux 技术在 DNS 层面体现的各类特征，结合其他过滤条件，实现对该类型的僵尸网络的有效检测与预警。

2. 实现对使用 Domain-flux 技术的该类型僵尸网络的检测

在另外一种类型的僵尸网络中，使用了 Domain-flux 技术，即根据 DGA 算法生成域名从而逃避域名屏蔽的逃逸方式，而本文设计的检测方法需要在 DNS 流量的基础上，找到 Domain-flux 技术体现的各类可检测特征，同时结合其他方法，完成对该类型僵尸网络的检测。

3. 高效的检测效率与准确率

本文从整体与局部两个角度出发，提出的检测方法，相比于传统的局部特征检测，能够在初步筛选的情况下提高检测效率，同时结合局部特征，在后续详细的分类检测中，使用目前高效的机器学习模型，能够提高准确率。

4. 高度可扩展性与适用性

本章设计的检测方法，在提取图组件局部特征时，结合一些动态更新的内容，对检测模型做更新升级。同时设计的检测方法能从整体考虑，能够更好适用更新

变化的僵尸网络的检测。

4.2 网络部署

本课题原型系统的检测方法是基于 DNS 流量分析的基础上，同时为减少对待检测的网络构成干扰与影响，系统部署则需考虑采用并行旁路的方式。即在目标网络出口处网关或交换机等设备上，将一个或多个 LAN 端口的数据流量转发到某个特定端口，从而实现流量镜像的效果。而这个端口获得的镜像流量则作为本系统的数据输入，作为后续流量预处理与过滤的基础，这个端口也被成为“镜像端口”。则网络结构部署图如 4-1 所示。

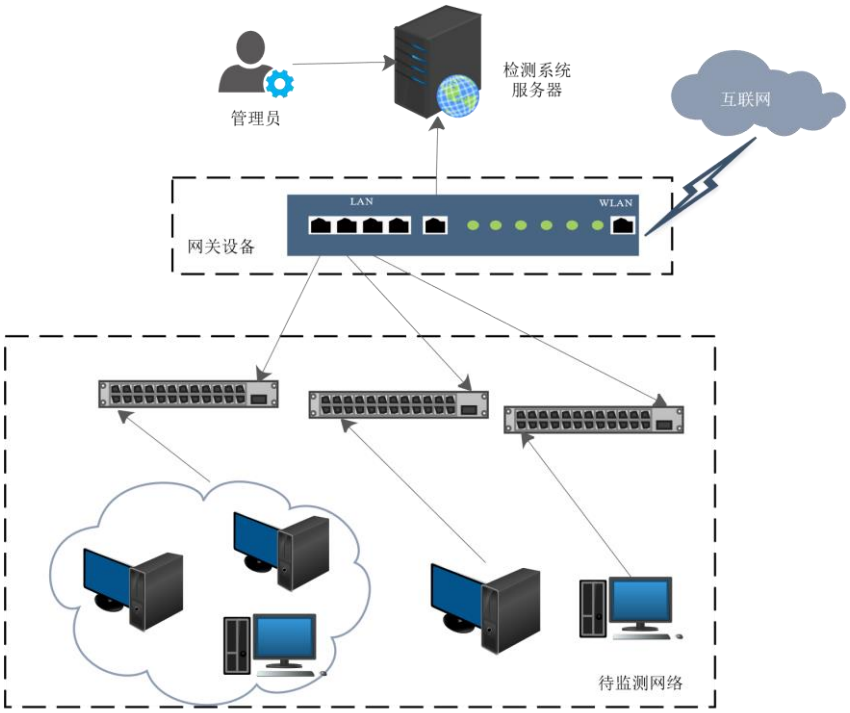

图 4-1 网络结构部署图

4.3 总体架构设计

本章检测方案从数据流传递的纵向角度分析架构，可以分为 4 层，包括数据接入层、数据存储层、处理单元层、用户界面层。具体详见总体方案总体架构图，见图 4-2 所示。

1. 数据接入层：该层为总体结构的底层结构，为输出的初始输入层，即外部数据来源，如图中所示，主要包括网关设备的旁路镜像流量以及在线的黑白名单网站。前者为流量分析的输入数据，后者为下一层需要进行黑白名单过滤的依据。

2. 数据存储层：该层是数据的初始处理层，将初始的输入数据经过过滤或者其他格式化处理，如镜像流量将根据设定的条件过滤其中的 DNS 流量，根据公开的黑白名单数据分类处理后，得到 FQDNs 黑白名单以及 IPs 黑白名单。

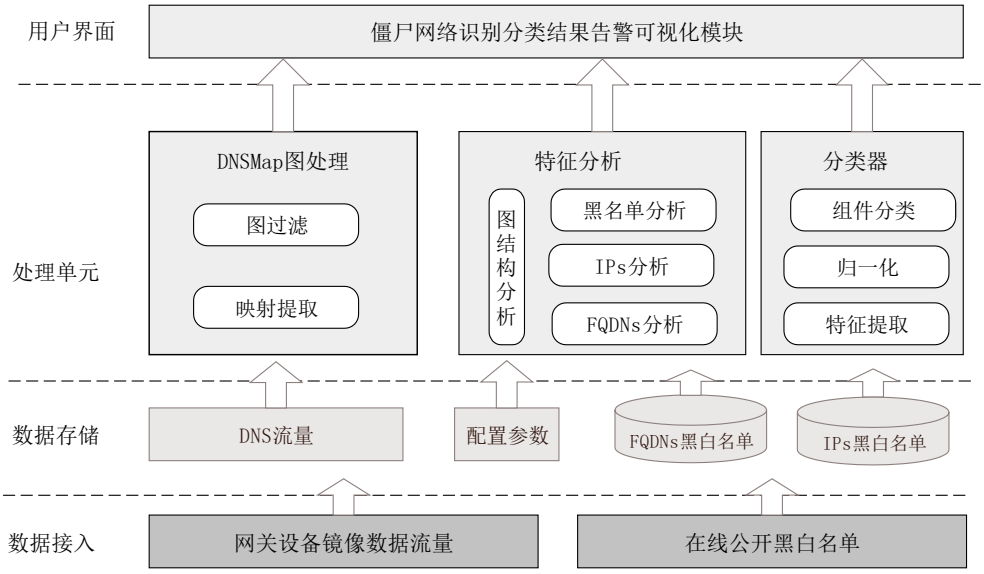

图 4-2 检测方案总体架构

3. 处理单元层：这一层则是将低一层的数据进行处理，是检测方法总结结构中最为关键的一层。这一层可以分为三大模块，分别为 DNSMap 图处理模块、特征分析模块、分类器模块。其中 DNSMap 图组件处理模块的作用是借助 DNSMap 工具对 DNS 流量进行 FQDNs-to-IPs 映射提取，同时还根据配置参数提供一定程度上的过滤效果，即过滤掉一些不敏感的映射，即安全良性的数据。而特征分析则是本检测方法的关键所在，其重点在于对 Fast-flux 技术与 Domain-flux 技术在僵尸网络中体现特征的理解，主要包括 4 部分，分别是图结构分析、FQDNs 分析、IPs 分析、黑名单分析。分类器模块则是完成对特征处理后数据的化为，经过特征提取后，得到特征向量，然后经过归一化处理，组建分类。则得到分类的初始结果。

4. 用户界面层：这一层的主要作用则是对分类结果的进一步处理，以作可视化的展现。包括对结果的分析评估，给出僵尸网络的识别预测与判断，同时还完成对危险的告警。

4.4 系统模块设计与实现

根据总体架构设计，本课题的原型系统可分为 7 大模块，分别是数据库设计模块、在线数据获得模块、系统控制模块、流量预处理模块、图关联映射处理模

块、特征提取模块、分类器模块。本章将针对各个模块的设计实现做详细介绍。

4.4.1 数据库设计模块

在本系统中，从数据流的角度分析，网络数据流等数据是通过文件的方式存储，而处理的中间过程，数据的存在方式则是在内存中。而为保证系统可靠地运行，需要将部分数据做持久化处理，即需要设计数据库保存。

根据设计目标，需要持久存在的数据包括 FQDN 与 IP 的相关的静态信息，包括其对应的黑白名单表、Whois 信息以及图关联映射的数据 Block 块信息表。其整体的数据库表关系图如图 4-3 所示：

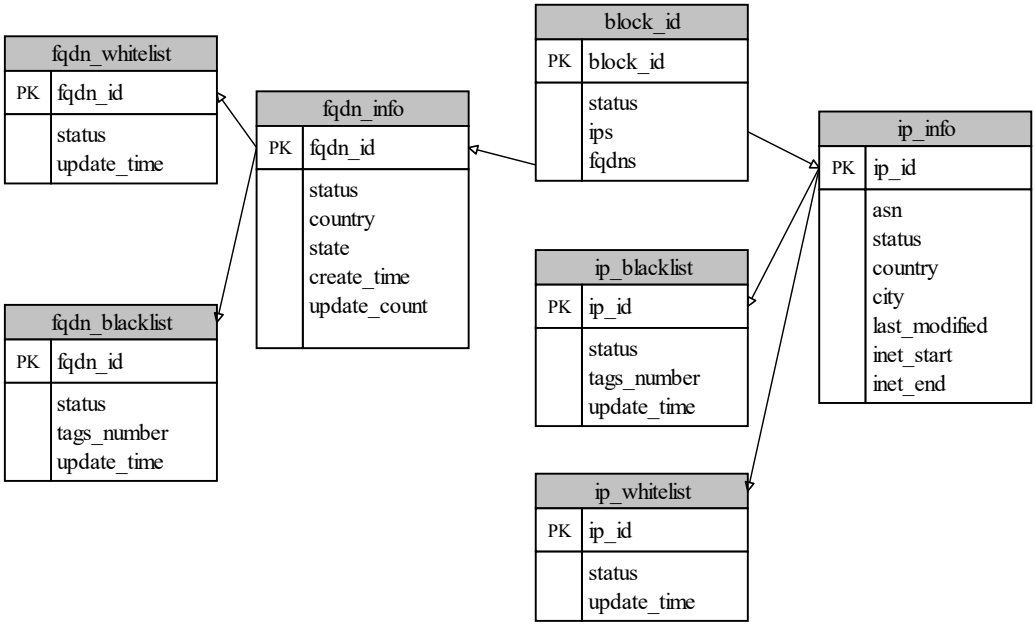

图 4-3 核心数据库表关系图

4.4.2 在线数据获取模块

在本系统中，部分模块功能的实现需要借助外部在线数据源，如 FQDN 与 IP 的黑白名单表，FQDN 与 IP 的 Whois 注册信息等。同时为实现系统的高适用性，这些数据则需要不定时更新，从而提高系统的检测识别的效果。而对于在线公开数据的管理则是由该模块负责。

关于在线公开的数据源，前序章节已作介绍。本模块则是根据数据源提供的查询接口获取响应数据，而对于未提供编程接口则需要根据网页内容编写 Python 爬虫分析。该模块处理的大体流程如图 4-4 所示：

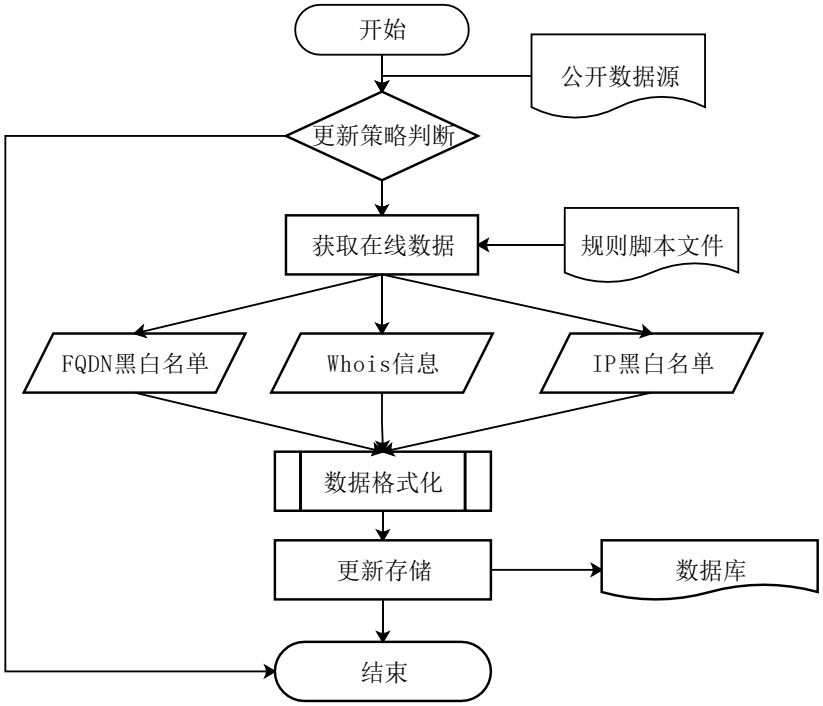

图 4-4 在线数据获取流程图

对于 FQDN 与 IP 的白名单过滤机制的核心算法在前序章节已有介绍，根据 FQDN 与 IP 的特点计算摘要，而进行比对的数据库则是由本模块功能完成。而对于黑白名单的数据获取，针对不同的网站来源，需要编写不同的规则脚本。如在 Alexa 网站获取域名列表白名单数据，在 Malwaredomainlist 站点获取黑名单的核心脚本代码如代码 4-1 所示：

代码 4-1 黑白名单匹配规则代码

Alexa 匹配规则： pattern=re.compile(r'"<divclass="count">(.*?) .*?(.*?).*?   '",re.S)
Malwaredomainlist 匹配规则： pattern=re.compile(r'"<trbgcolor=".*?"onmouseover=".*?"onmouseout=".*?"><td><nobr>(.*?)_\ d{2}:\d{2}</nobr></td><td>(.*?)</td><td>(.*?)</td><td>(.*?)</td><td>(.*?)</td><td>(.*?)</td ><td>(.*?)</td><td align="center">'", re.S)

4.4.3 系统控制模块

在系统设计之初，为降低耦合度，提高可拓展性。通过功能点划分的方式，系统划分成多个模块。各个模块完成特定的功能，模块之间则通过数据通信与控

制等方式关联起来，达到协同工作的效果。而随着模块功能愈趋近复杂性与模块数量的增加，如何更好地协同控制各个模块则是一个新挑战。

系统控制模块的设计初衷就是各功能模块的协同枢纽。用户可通过编写配置文件、设定参数、传入指令等方式控制各功能模块之间的数据流通，从而达到整个系统的协调运作，同时通过该模块还可以实时了解整个系统的运行状态。系统控制模块与其他模块之间的数据通信关系如图 4-5 所示：

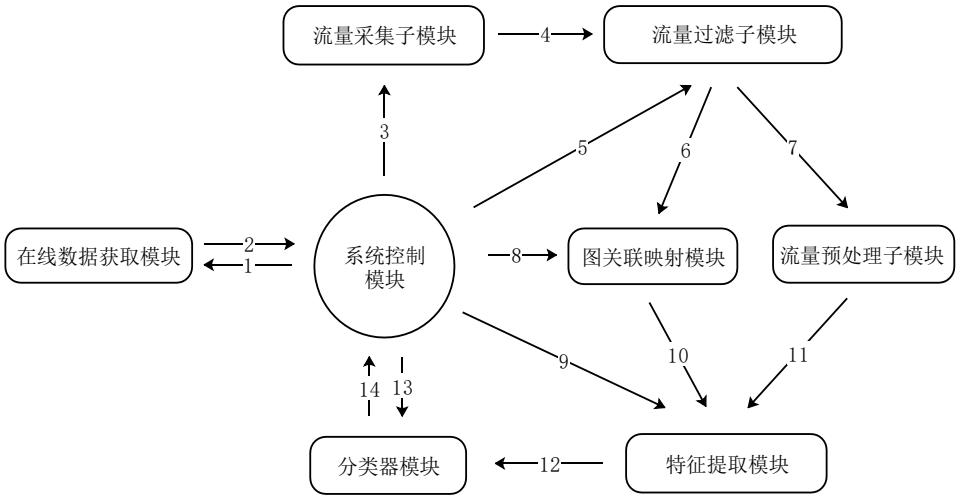

图 4-5 模块间通信关系图

通信 1：在线数据更新策略、不同网站数据获取接口及执行脚本。

通信 2：FQDN、IP 等黑白名单以及 Whois 的最新信息等。

通信 3；PF_RING 配置参数，包括抓取模式、缓存大小等，以及待检测网络的配置参数。

通信 4：端口镜像抓取的网络流量数据。

通信 5：流量过滤规则、白名单信息，DNS 流量的处理方式。

通信 6、7：过滤后的 DNS 流量数据。

通信 8：图关联模块算法的设定参数。

通信 9：特征提取参数以及黑名单信息

通信 10：DNS 流量映射关联后的图信息。

通信 11：DNS 流量字段数据集。

通信 12：特征提取后的特征向量集。

通信 13：分类算法的执行参数等。

通信 14：图组件分类结果。

4.4.4 流量采集与预处理模块

4.4.4.1 流量采集子模块

为实现在大型高速网络中实现实时高效地流量抓取，本方案选择结合 PF_RING 工具。是因为在 libpcap 中存在的相同数据结构的多次拷贝以及涉及用户空间与内核空间反复调用而限制接受报文的效率等问题，这种问题对小报文的影响尤为明显。而 PF_RING 针对该问题的解决方案是减少报文在传输过程中的拷贝次数。

PR_RING 是一种新型的基于环形缓冲区的包捕获套接字模型，其每次创建一个 PF_RING 套接字便分配一个环形缓冲区，当套接字结束时则释放。PF_RING 通过 Linux NAPI 轮询来自 NIC 的数据包。这意味着 NAPI 将数据包从 NIC 复制到 PF_RING 的环形循环缓冲区，如果缓冲区满则丢失该数据包。然后用户程序可以直接从这个缓冲区中读取数据，即这个过程存在两个轮询过程，应用程序与 NAPI，这样可以更好利用 CPU 周期，同时 PF_RING 还可以同时将输出数据包分配到多个环形缓冲区，提供更好的并发性。其原理示意图如图 4-6 所示。

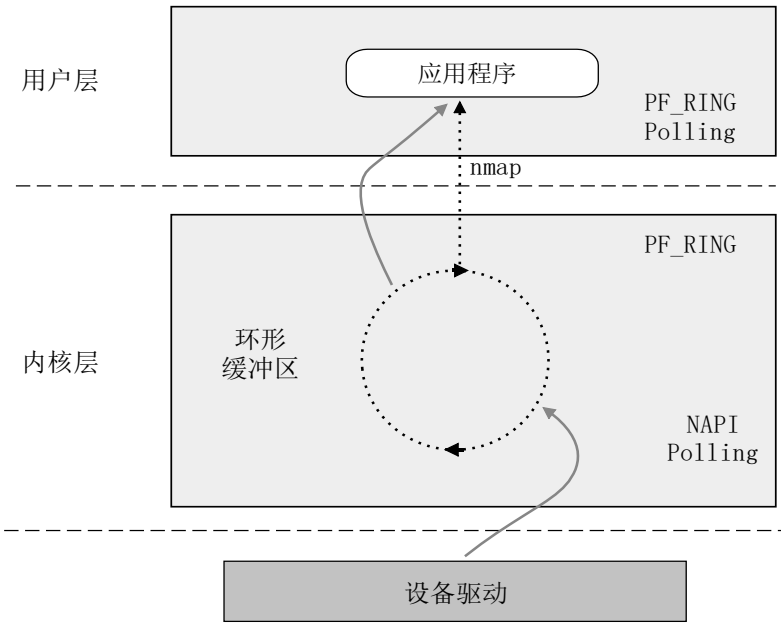

图 4-6 PR_RING 原理

PF_RING 还采用了模块化结构，可以使用标准 PF_RING 内核模块的其他组件，如 ZC 模块、基于 FPGA 的卡模块，堆栈模块、时间线模块、Sysdig 模块。其中重点关注的 ZC 模块，支持使用用户空间 ZC 驱动程序，这是一种英特尔的新型的 DNA，直接 NIC 访问，可用于高速的数据包捕获与传输，这是因为 NIC NPU

（网络处理单元）在可以没有任何内核干预的情况下向用户层发送数据包或获取数据包，最高可支持 10Gbit 带宽发送或接受数据，根据官方文档说明，硬件要求为基于 Intel 82599（例如 Intel X520）或 Silicom Director 10 Gbit NIC 以及最新的 Linux 内核（≥2.6.31）。其工作模式与传统的 libpcap 和 PF_RING 普通模式的区别如图 4-7 所示。

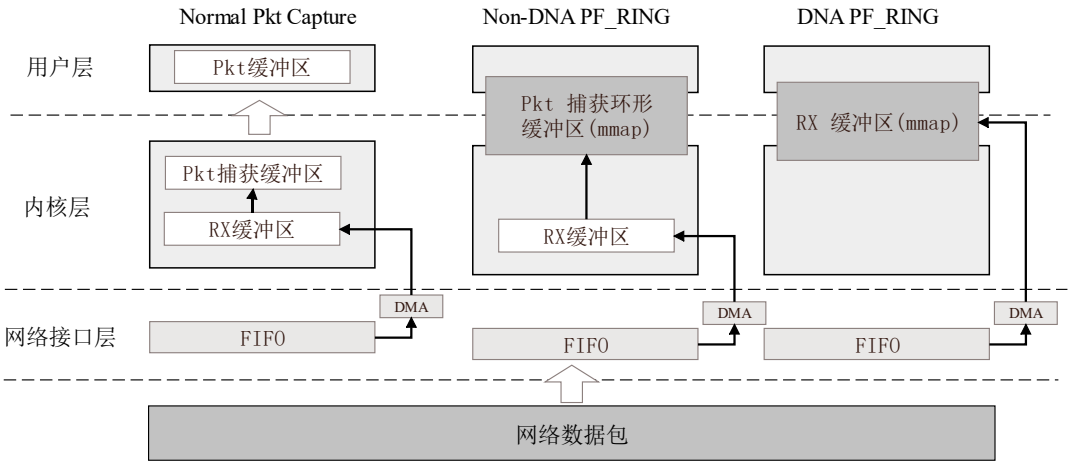

图 4-7 PF_RING 等方案对比

本方案中将采用最新 PF_RING 方案+ZC（也称为新一代的 DNA）驱动程序的方式，它允许数据包直接从网络接口同时以 Zero Copy（零拷贝）方式绕过 Linux 内核和 PF_RING 模块，从而完成对大型高速网络的实时流量抓取，这样能够依托于最新的硬件支持，即 NIC NPU，从而获得更高效的处理结果。

对于 ZC 模块的使用，需要注意的是在加载驱动程序（由“-zc”后缀标识）之前，需要先加载 PF_RING 内核模块，且使用前缀“zc:”打开网络接口设备，该设备将无法用于标准网络，因为它是通过内核旁路以零拷贝的方式访问，如同 DNA 一样。其核心配置代码如下：

代码 4-2　PF_RING ZC 模块核心代码

```
zc = pfring_zc_create_cluster(ID, MTU, MAX_BUFFERS, NULL); //创建结果集
for (i = 0; i < num_devices; i++)   //打开设备
   inzq[i] = pfring_zc_open_device(zc, devices[i], rx_only);
for (i = 0; i < num_slaves; i++)   //创建队列
   outzq[i] = pfring_zc_create_queue(zc, QUEUE_LEN);
zw = pfring_zc_run_balancer(inzq, outzq, num_devices, num_slaves, NULL,
NULL, !wait_for_packet, core_id);
```

4.4.4.2 流量过滤子模块

在正常网络流量中，DNS 流量大小往往占比不到 1%，这是流量分析重要依据，也为本系统在大型高速网络的应用性提供可能。同时对于流量的过滤预处理则必不可少。

在前序章节的背景知识中可以了解到，DNS 协议在区域传送时会使用 TCP 协议，而在域名解析时使用 UDP 协议。但同时占用 53 端口，则是 DNS 流量关键过滤标准，可以通过配置 BPF 过滤器过滤 DNS 流量。

BPF[46]也称柏克利封包过滤器，是类 Unix 系统上数据链路层接口，而 PF_RING 支持 BPF 过滤器，且默认开启，且自带的两种过滤器（Wildcard 与 Preicise）。如目标网络流量数据较大，则使用 PF_RING+ZC 方案采集全流量，需要注意的是，在这种情况下 PF_RING 不再支持 BPF 过滤器，需要配合其他的工具完成 DNS 流量的过滤，如 Tcpdump、Tshark 等，即在全镜像流量的基础上再设定 BPF 过滤器规则，通过配置过滤器，则可以达到过滤只包含 DNS 域名解析的数据流量的效果。

4.4.5 图关联映射处理模块

系统在完成流量预处理与过滤后，则得到纯 DNS 流量，而这部分数据则是图关联映射模块的输入。根据前序章节的介绍，图关联映射是在 DNSMap 工具的基础上完成，依据映射提取关联算法，可以得到可疑映射列表与关联映射二部图组件。

其中对于可疑映射的定义是通过在配置文件（config.py）的参数设定完成,核心配置如代码 4-3 所示。通过 DNSMap 对图组件的初步分析，可以完成对图组件的初步过滤，从而减少后续分析的数据量。

代码 4-3 图过滤参数配置核心代码

```
clusteringThreshold=0.35       #域名相似度阈值
domainCountThreshold=0.5    #IPBlock 集群相似度
maxClusterSize=30              #集合中 FQDN 节点最大数
maxNumClusters=50              #IPBlock 集群最大数量
timebinSizeMerge=3600*6     #IPBlocks 合并的时间间隔
filterTimeThreshold=3600*3   #关联映射输出间隔
```

通过初步的参数设定完成图过滤，可以得到可疑的图组件，其中 Fast-flux 与 Domain-flux 两种可疑的恶意映射的典型示例如图 4-8 所示。

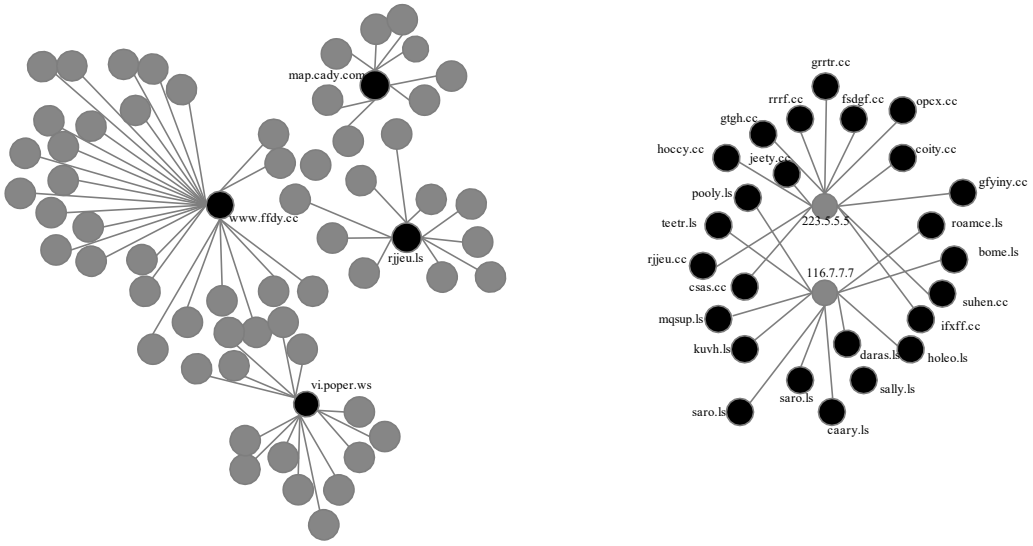

图 4-8 Fast-flux(a)与 Domain-flux(b)示例图

4.4.6 特征提取模块

本课题的特征提取工程是基于 DNS 映射的图组件结果，将图组件的原始数据转化成分类算法可识别的数值特征。主要包括以下方面，分别是图结构特征、节点特征、连接边特征、黑名单特征。该模块工程实现的流程如图 4-9 所示：

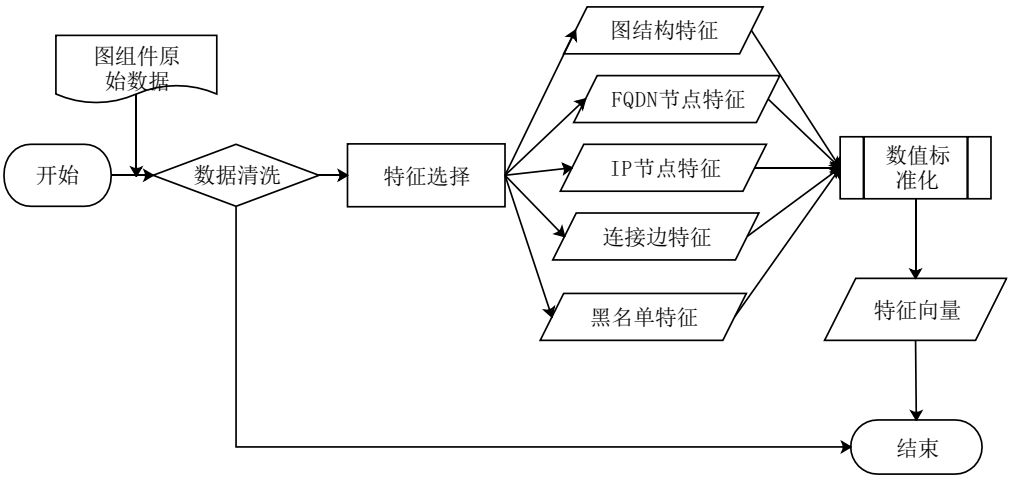

图 4-9 特征提取流程图

在特征提取的流程图中，每一个步骤都有其详细的处理含义，具体如下：

数据清洗：这是在图组件的原始数据输入后的首先处理步骤，它完成对数据的一个过滤，对于图组件中的脏数据，如域名缺失，IP 不符合规范等进行丢弃。

特征选择：针对每一个图组件结构，分别提取其图结构特征、节点特征、边特证、黑名单特征。根据每一特征项的定义，分别计算其数值，这些计算过程还会涉及其他模块的输入数据，如黑名单数据。特征选择为机器学习分类的前序步骤，不针对特定的分类算法。

数值标准化：对于计算后的特征数值，为进行后续的分类处理，还需要进行标准化处理，约束其数值范围与类型，同时对于特征缺失字段，将根据其含义赋予缺省值。同时对于字段缺省值过多的数据，也将进行丢弃处理。经过数值标准化的处理，将得到后续机器学习分类的输入数据——特征向量。

4.4.7 分类器模块

分类器模块主要完成的任务是：根据前序模块数据处理得到的特征向量，选择合适高效的机器学习分类算法，完成可疑的恶意图形组件的检测。

本课题经过需求分析，选用 LightGBM 分类算法。本模块将针对该分类算法在指定数据集上的训练过程与检测实施过程作详细介绍。

4.4.7.1 分类模型训练

为实现准确的检测算法，模型的训练则是首要步骤。LightGBM 对于训练文件的数据格式要求是 CSV、TSV 与 LibSVM 三者之一。而本课题中则选择 CSV 格式，对于二分类问题，LightGBM 要求的是第一列数据为 Label 标签，数值为 0 或 1。同时文件中是不包含 Header（标题）的。同时对于数据量的要求是建议 10000+行，否则会过拟合。

使用 LightGBM 做二分类机器学习，需要完成训练集与测试集的划分，其核心配置如代码 4-4 所示：

代码 4-4 LightGBM 数据集划分代码

```
# 导入数据
train_x, train_y, test_x = load_data()
# 使用 sklearn.cross_validation 进行训练数据集划分
X, val_X, y, val_y = train_test_split(
    train_x,
    train_y,
    test_size=0.05,
    random_state=1,
    stratify=train_y # 这里保证分割后 y 的比例分布与原数据一致
)
# 后续处理
```

对数据集进行划分之后，则需要设定二分类的诸多关键参数，其核心代码如代码 4-5 所示：

代码 4-5 LightGBM 二分类参数配置代码

```
# 使用字典格式保存配置参数
params = {
      'task':'train'
      'boosting_type': 'gbdt',
      'objective': 'binary',
      'metric': {'binary_logloss', 'auc'},  #二进制对数损失
      'num_leaves': 5,
      'max_depth': 6,
      'min_data_in_leaf': 450,
      'learning_rate': 0.1,
      'feature_fraction': 0.9,
      'bagging_fraction': 0.95,
      'bagging_freq': 5,
    …
}
```

完成分类模型的参数配置后，后续步骤则是模型的训练过程与结果的输出，其核心代码如代码 4-6 所示：

代码 4-6 LightGBM 训练与结果输出代码

```
# 创建数据集
lgb_train = lgb.Dataset(X_train, y_train)
lgb_eval = lgb.Dataset(X_test, y_test, reference=lgb_train)
# 训练过程
gbm = lgb.train(params,
                lgb_train,
                num_boost_round=10000,
                valid_sets=lgb_eval,
                early_stopping_rounds=500)
# 导出预测模型
importance = gbm.feature_importance()
names = gbm.feature_name()
with open('./feature_importance.txt', 'w+') as file:
      for index, im in enumerate(importance):
            string = names[index] + ', ' + str(im) + '\n'
            file.write(string)
```

4.4.7.2 在线分类检测

模型训练阶段是在对离线数据集构建的特征向量的基础上，进行 LightGBM 分类算法的训练、预测，不断调整参数，完善分类模型的过程。而在本阶段，则是根据训练的分类模型，以待监测的网络抓取的 DNS 流量作为原型系统数据输入，完成流量过滤、图关联映射、特征提取等流程，得到特征向量。并以此特征向量作为分类模型的测试数据集，以分类结果作为检测结果，即系统的输出。

原型系统中 LightGBM 分类预测同样是 Python 编写，其核心代码如代码 4-7 所示：

代码 4-7 LightGBM 分类预测代码

```
# 导入数据
test_x = load_data()
# 导入模型
with open('./ feature_importance.txt', 'w+') as input_model
preds = gbm.predict(test_x, num_iteration,input_model)   # 输出的是概率结果
# 导出预测结果
for pred in preds:
      result = 1 if pred > threshold else 0
```

4.5 本章小结

本章重点介绍了课题对应的原型系统的设计与实现，内容为系统的设计目标、网络部署、总体架构以及各模块的详细实现。模块化的设计与实现包括系统数据持久化的数据库设计模块、系统外部依赖的在线数据获取实现模块、协调各模块之间数据交互的控制中心模块、实现流量采集与预处理的模块、基于 DNS 流量完成关联映射的模块、在图组件上分析特征与提取特征的模块以及基于 LightGBM 分类器设计实现模块。上述模块完成了高速网络下面向 Fast-flux 与 Domain-flux 僵尸网络检测的原型系统的设计与实现。

第五章 算法评估与系统测试

前面章节介绍了本课题中提出面向 Fast-flux 与 Domain-flux 类型僵尸网络检测方法的详细设计以及原型系统的设计与实现。本章节将针对提出的检测方法，在特定的离线数据集上做实验分析，并评估检测算法指标。同时在构建的待检测网络条件下，选取部分僵尸网络流量数据进行流量重放，再与测试网络正常流量混合，在此网络情况下再部署原型系统，并对系统各模块功能与性能进行在线测试，从而完成本课题检测算法评估与系统测试。

5.1 实验环境

本课题设计原型系统应用场景的设计目标为高速大型网络，面临海量数据问题。因此，实验的测试依托高性能的硬件平台做支撑，其详细的硬件规则如表 5-1 所示：

表 5-1 硬件环境配置表

名称	配置
服务器	戴尔 PowerEdge R730XD
CPU	英特尔至强 Xeon E5-2600 v3 * 2
内存	8G DDR4 2400Hz * 4
硬盘	2TB * 4
网卡	千兆以太网卡 * 4

实验的测试环境也包括运行在硬件平台之上的软件环境，其详细的软件类型与版本如表 5-2 所示：

表 5-2 软件环境配置表

名称	配置
OS	CentOS 7.6.1810
PF_RING	7.4.0 release
Wireshark	3.0.0
Python	2.7.5
JDK	8u201
LightGBM	v2.2.3
TCPReplay	4.3.1

5.2 实验数据集

本课题的实验可以划分为两个阶段：一、离线数据集上算法测试；二、在线网络下原型系统测试。对于第一阶段，数据集将基于网络公开的僵尸网络数据集构建，其中在公开的数据集上筛选本课题面向的两种类型僵尸网络流量。第二阶段，重点在于测试原型系统的可行性以及是否达到系统设计目标，同时将第一阶段中的恶意样本流量混合与正常流量中，然后验证系统能否完成检测目标。

对于僵尸网络的检测，一直是安全领域的热门课题，行业中已有基于僵尸网络程序构建的公开数据集，如 Garcia S 等人[47]构建的 CTU-13 数据集，该数据集包含 13 个捕获场景的不同僵尸网络样本，涉及了不同类型的僵尸网络，如 Neris、Rbot、Virut 等，而其中部分样本包含本课题相关的 Fast-flux 流量。Beigi E B 等人[48]构建的 ISCX-Bot 数据集，其研究人员综合了 ISOT 中部分数据集、ISCX 2012 IDS 中用户真实流量数据以及构建 9 个僵尸网络形成的数据集，通过合并这些数据形成一个统一的评估数据集，而其中包含了 Storm 与 Zeus 等僵尸网络流量，符合本课题需求。Saad S 等人通过 honeynet project[49]构建了 ISOT Botnet 数据集,其中重点包含了 Storm 与 Waledac 两种僵尸网络，也整合了 Hungary[50]和 Lawrence Berkeley National Lab (LBNL)[51]两个流量实验室的数据，其形成的标签完善数据集也是研究 Fast-flux 的重要样本数据。Alenazi A 等人[52]通过 Atom export kit 构建的 ISOT HTTP Botnet 数据集,该数据重点过滤了 DNS 数据，包括正常流量与恶意流量，而恶意流量则是 Zeus 生成，符合 Domain-flux 数据集要求。

5.2.1 Fast-flux 数据集

检测 Fast-flux 类型的僵尸网络，则是需要应用了 Fast-flux 技术的僵尸网络的流量数据，根据第二章的僵尸网络的背景知识可以了解到 Storm、Waledoc 等僵尸网络中使用了 Fast-flux 技术，同时由于分类器的二分类中的训练与测试过程需要完备的带有标签的数据集，因此对于包含 Fast-flux 技术的僵尸网络流量样本分为两部分处理，对于标签完善的 ISOT 数据集作为分类器的训练与测试过程，对于纯 Fast-flux 恶意流量的 CTU-13 部分数据集，则可以作为后续原型系统的测试与验证数据。

关于 ISOT 数据集中恶意与非恶意流量的占比以及标签情况如表 5-3 所示：
从表中可以看到数据集中包含了 Storm、Waledoc 以及 Zeus 三种僵尸网络的流量，且具备 Mac 地址标签用以区分。而其中关于 Zeus 部分的数据流量则可应用于 Domain-flux 类型僵尸网络检测的数据集。

表 5-3 ISOT 数据集流量占比表

IP 地址	恶意流量	正常流量	标签	总计
172.16.2.11	Storm		MAC:BB:BB:BB:BB:BB:BB	55904(3.33%)
172.16.0.12			MAC:AA:AA:AA:AA:AA:AA	
172.16.0.2	Waledac		MAC:AA:AA:AA:AA:AA:AA	
172.16.0.11				
176.16.2.12	Zeus		MAC:CC:CC:CC:CC:CC:CC	
	Zeus(C&C)		MAC:CC:CC:CC:DD:DD:DD	
176.16.2.0/24		√	Others	1619520(96.66%)

5.2.2 Domain-flux 数据集

目前应用到 Domain-flux 技术的僵尸网络包括 Zeus、Torpig。而对于该类型僵尸网络的检测，目前很多方法都是依据 DGA 算法恶意域名，从而自行构建数据流量用以检测，但这样缺乏一定的权威性与可验证性。本课题的数据集则是在公开数据集的基础上构建待检测的数据集。对于 Domain-flux 部分的数据集则是由两部分组成，其中一部分则是表 5-3 中的 Zeus 部分数据，而另外一部分数据则是来源于 Alenazi A 等人构建的 ISOT HTTP Botnet 数据集，但该数据集中正常流量与恶意流量已经分割开来，为完成分类器的训练，则需要构建虚拟环境，将两部分数据进行重发合并，并完成数据的标签标记过程，从而形成可用的数据集。关于该数据集的原始标记情况如表 5-4 所示：

表 5-4 ISOT HTTP 数据集恶意流量部分标记表

序号	IP 地址	域名
1	192.168.50.14	zyklon.botnet.isot
2	192.168.50.15	blue.botnet.isot
3	192.168.50.16	liphyar.botnet.isot
4	192.168.50.17	gaudox.botnet.isot gdox.botnet.isot dox.botnet.isot **
5	192.168.50.18	blackout.botnet.isot
6	192.168.50.30	citadel.botnet.isot ***
7	192.168.50.31	citadel.botnet.isot ***
8	192.168.50.32	be.botnet.isot black ennery
9	192.168.50.34	zeus.botnet.isot

5.3 基于 DNS 图映射关联的检测方法实验与分析

本课题中的检测方法是基于 DNS 流量做关联分析，而实验数据集仅为原始数据，后续还需要经过一系列复杂处理，包括 DNS 流量过滤、DNSMap 图关联处理、图组件结构特征提取、特征数值标准化等，才能够得到机器学习分类处理的特征向量。本小节则是在特征向量的基础上，分析评估检测方法。

本文检测方法中需要进行二分类，对于分类算法，本文将结合方法需求，考虑机器学习中目前较为主流的 4 种有监督式分类算法加入对比，包括决策树（Decision Tree）、支持向量机（Support Vector Machine，SVM）、极端梯度提升树（eXtreme Gradient Boosting，XGBoost）、LightGBM（Light Gradient Boosting Machine）。

5.3.1 评估指标

算法的评估标准将选用机器学习分类方法中的精确率（Precision）、准确率（Accuracy）、召回率（Recall）以及 F1 评分（Precison 与 Recall 的相关函数）。对于二分类问题，根据真实情况与分类器预测情况组合，将样例分为真正例（True Positive，TP）、假正例（False Positive，FP）、正反例（True Negative，TN）、假反例（False Negative，FN），其关系如表 5-5 所示：

表 5-5 二分类结果组合表

真实情况	分类结果	
	正例	反例
正例	TP（真正例）	FN（假反例）
反例	FP（假正例）	TN（真反例）

准确率（Accuracy）的定义如公式 5-1 所示：

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{5-1}$$

精确率（Precision）的定义如公式 5-2 所示：

$$Precision = \frac{TP}{TP + FP} \tag{5-2}$$

召回率（Recall）的定义如公式 5-3 所示：

$$\mathrm{Re}call = \frac{TP}{TP + FN} \tag{5-3}$$

F1-评分（F1-Measure 或 F1-Score）的定义如公式 5-4 所示：

$$F1\text{-}Measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{5-4}$$

5.3.2 实验结果分析

根据评价指标的定义，准确率反映出分类器对样本的预测判断能力，即将正例样本判定为正例，反例样本判定为反例。而精确率则是反映分类器在预测结果上的精度，即分类器预测为正的正例在所有真正例中所占的比例。

在本实验中，采用十倍交叉验证与百分比分割两种方式来评估与验证实验中选用的各个分类器算法的准确率与精确率。

十倍交叉验证指的是将数据集随机划分 10 个较小的子集，然后选取其中 9 个作为训练集，剩下的 1 个作为训练集，并重复该过程 10 次，也称为 10 倍 CV，而实验的结果则是 10 次实验测量值的平均值。该实验结果中的准确率与精确率如图 5-1 所示：

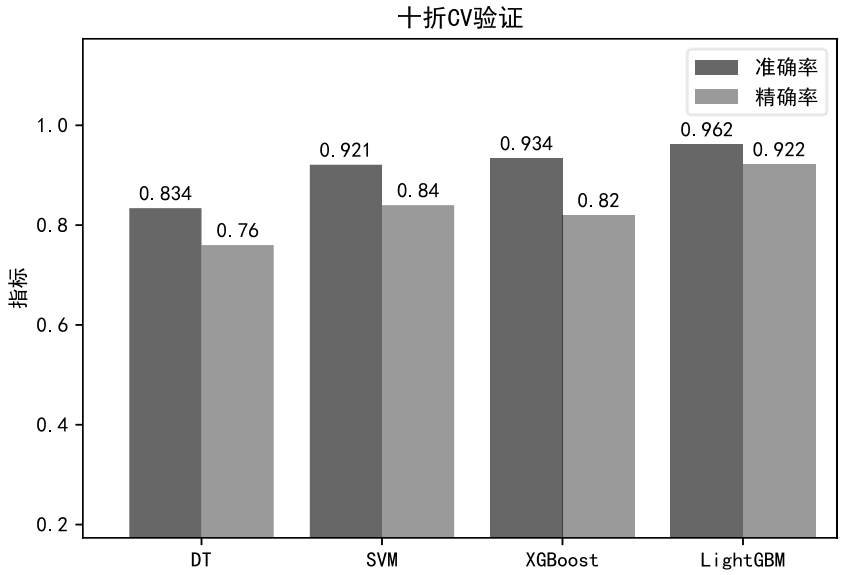

图 5-1 十折 CV 验证情况下的准确率和精确率

百分比切割验证是指在分类器的训练阶段，将数据按不同百分比分为两部分，如 75%的数据用于训练，而 25%的数据则用于测试。使用 75%百分比切割的实验情况下，分类器的准确率与精确率如图 5-2 所示：

根据图中可以看到，XGBoost 和 LightGBM 算法的有着较高的检测准确率与精率，而 DT 和 SVM 的检测精度和准确率则相对较低。XGBoost 和 LightGBM 的检测的准确率都在 93%以上，其中 LightGBM 分类器的检测率最高，准确率在 96.2％左右，精度在 92.2％左右。

除去准确率与精确率以外，评价一个分类算法的优劣，还同样需要其他指标，本文中则是额外增加了召回率（Recall）和 F1-Measure 指标来评估算法的性能。

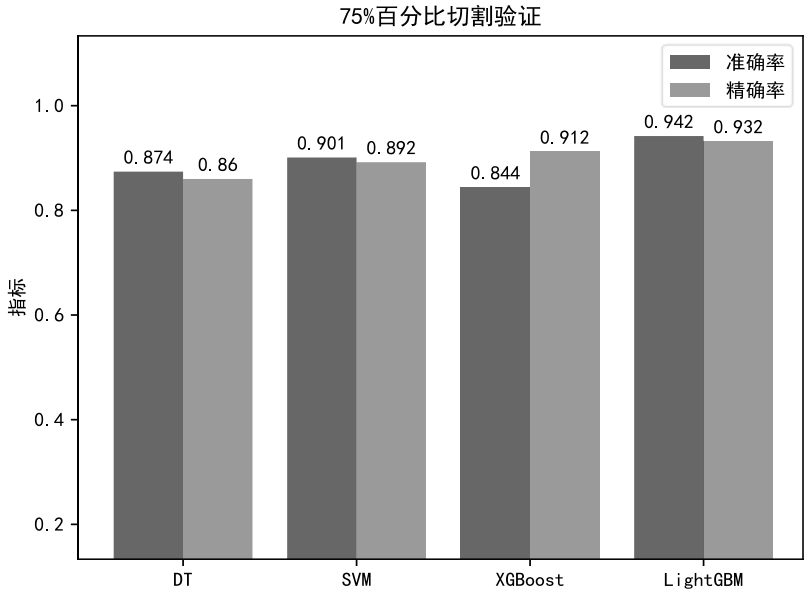

图 5-2 75%分割验证情况下的准确率与精确率

根据公式 5-3 所知召回率反映的是分类器预测的正例占样本中所有正例的的比例，它衡量的是分类算法的查全率，所以也称查全率。而根据公式 5-4 所示，F-Measure 是精确率（Precision）与召回率（Recall）的加权调和平均值，它也是分类领域采用的一个评价指标，因为在实验中经常会出现精确率与召回率相矛盾的情况，如实验结果中分类的恶意组件在数量较少的精确率较高，但在召回率的指标却很低，所以 F1-Measure 综合了两者的结果，当 F1 的数值较高时则更能反应分类算法的效果更理想。

我们在十折 CV 的验证情况下分别计算了四种分类算法实验结果的召回率与 F1-Measure 指标，其具体情况见图 5-3 所示。

根据图 5-3 中可以看到 LightGBM 在召回率与 F1-Measure 指标上表现优异，召回率达到 0.78，F1-score 则是达到了 0.832，与目前流行的 XGBoost 算法相比具备较明显的优势。

除上述指标外，在数据量更大的数据集上，LightGBM 能够支持更好的并行计算的能力，还可以考虑结合目前流行的各类并行计算平台，则与其他算法相比将有更好的性能优势。而针对大数据量的应用情况则是更符合本课题的设计要求，因此在综合对比各方面指标的情况下，本课题选用 LightGBM 作为分类器算法模型则为最优方案。

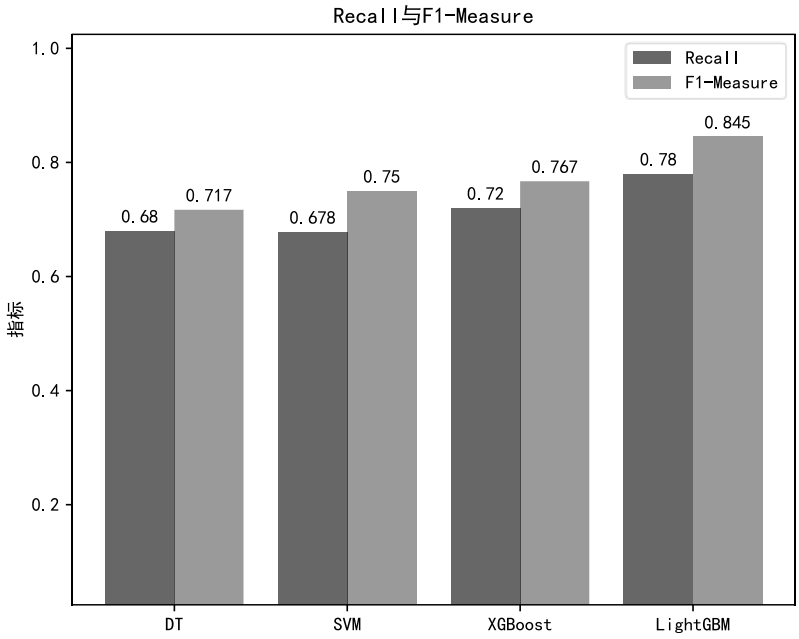

图 5-3 十折 CV 验证情况下 Recall 与 F1-Measure

5.4 高速网络原型系统测试

第四章设计高速网络下面向 Fast-flux 与 Domain-flux 两种僵尸网络检测的原型系统，前序章节已评估验证检测算法的有效性与准确性，本节内容则是在原型系统的各模块整合实现基础上，测试系统在高速网络的应用环境中的实际检测效果，并且按照设计思路，分功能模块测试。

5.4.1 流量采集模块测试

根据设计要求，高速网络的流量采集是基于 PF_RING 工具实现，配置 Zero Copy（零拷贝）方式，并通过在待检测网络的出口设备上的端口镜像功能，将多个端口的流量复制到单一出口，然后连接到原型系统部署的高性能服务器。

需要注意的是使用 PF_RING 的 ZC 模块，依赖于 Intel 的 10G 级别的高性能网卡，该级别系列网卡是由 ixgbe 驱动，如在 Linux 发行版中未作为默认选项加载，则需要在 SourceForge 网站下载编译安装。

本模块主要测试在高速网络下，即大量数据流量下原型系统对于流量采集抓取的可行性、完整性、高效性。原型系统部署测试的情况是：在目标网络中增加特定主机，模拟受感染机器，在主机上通过 TCPReplay 工具，将数据集中的纯恶意流量进行流量重放，与网络中正常流量混合。然后在网关等设备处完成流量采集。

使用 PF_RING 中的 pfcount 工具通过参数指定网卡，可以统计该网卡流量抓取的总体状况，需要特别注意的是在使用 zc 模块时，在网卡前需要添加“zc:”前缀。在 100Mbps 实验室测试网络中的流量采集效果如图 5-4 所示。根据测试结果可以看到 PF_RING 的流量抓取的效率是极高的，丢包率极低甚至无丢包情况。

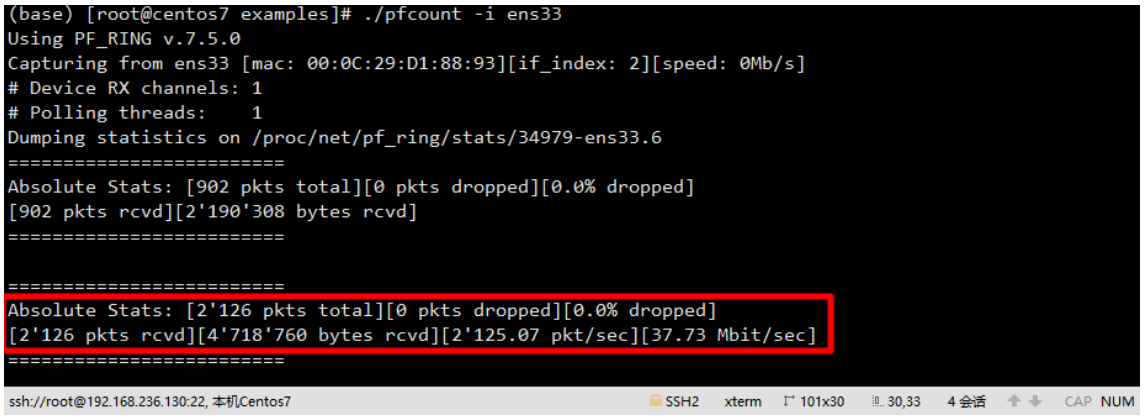

图 5-4 100Mbps 网络下流量采集测试

5.4.2 流量过滤与预处理模块测试

系统在完成流量的旁路镜像抓取之后，得到的流量数据则需要进行过滤与预处理。该模块主要测试对于 DNS 流量的过滤效果以及预处理的结果。

根据 DNS 协议的特点，配置 BPF 过滤器，根据 DNS 查询特点，可以设置 dns.flag.response==1,可以得到纯 DNS 响应包流量，这样可以极大较少后续处理的数据量。经过实验室条件下 12h、24h、48h 的周期测试，DNS 流量的过滤情况如表 5-6 所示：

表 5-6 DNS 流量过滤情况

时间(h) / 流量(MB) / 类别	12	24	48
总流量	88064	202548	445605
DNS 流量	4.6	10.12	21.25
占比统计	5.22‰	4.99‰	4.76‰

从表中可以看到，在正常网络流量中，DNS 流量占比不到 1%，经过 DNS 流量过滤，待处理的数据量得到极大地减少，过滤效果极为明显。

经过过滤得到的 DNS 的 Pcap 文件，可作为 DNSMap 处理的输入文件，但对过滤后的 DNS 流量，还需要一个预处理过程：即提取 DNS 流量中的信息。包括请求类型、TTL、源 IP 与目的 IP 以及各字段信息。预处理得到的信息是后续图组

件中特征提取计算的关键数据来源，如图组件中的 IP 节点、FQDN 节点、连接边等相关信息。对 DNS 流量的预处理得到相应的文本信息，其结果如图 5-5 所示：

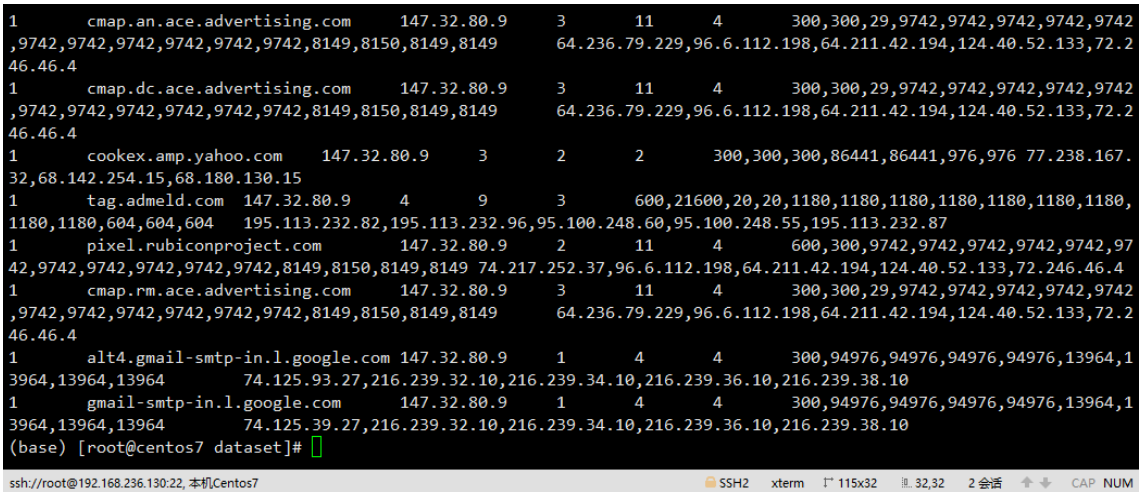

图 5-7 DNS 流量预处理结果

5.4.3 图关联处理模块测试

系统在完成流量的过滤与预处理之后，其中过滤得到的是 DNS 中响应包流量，在响应包中包含请求查询的域名以及返回的 IP 地址，包含这些信息的 Pcap 包则是下阶段 DNSMap 处理模块的输入文件，Pcap 包会经过 pylibpcap 解析，得到流量包中的 IP 地址与请求域名，经过处理得到 IPBlocks，即关联映射图组件。DNSMap 处理的中间结果可疑映射 Suspicious.txt 如图 5-8 所示:

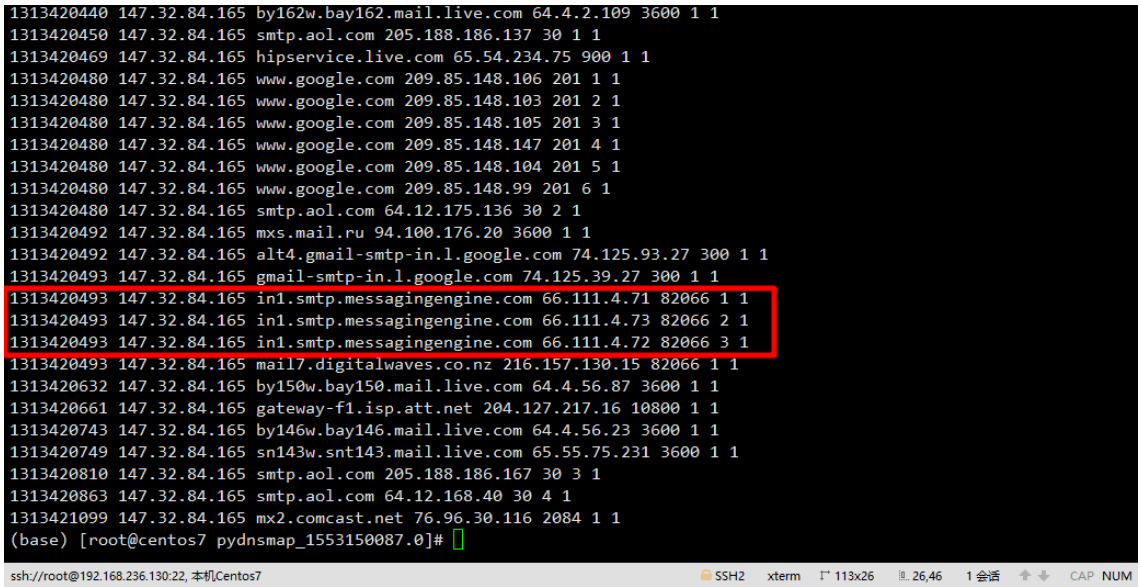

图 5-8 DNSMap 处理中间结果

5.4.4 在线分类检测模块测试

前序模块在完成 DNS 流量过滤与预处理之后，得到的中间数据则交由后台其他模块处理，依次经过 DNSMap 图映射关联、特征提取、LightGBM 算法分类等步骤，然后得到分类结果。

表 5-7 检测结果 XML 文件部分属性说明

序号	属性	说明
1	graph	图组件
2	client	受感染目标主机或网络源 IP 地址
3	ip_lists	组件中的 IP 地址集
4	fqdn_lists	组件中的域名集
5	prediction	预测结果，0 表示正常，1 表示恶意
6	types	预测类型，0 表示正常，1 表示 Fast-flux，2 表示 Domain-flux

流量数据经过图映射关联，数据结构则会发生变化，DNSMap 处理之后得到的是多个二部图组件，每一个组件是由 FQDN 点与 IP 点相连构成。而特征提取则是针对每一个这样的图组件，因此特征向量的每一行数据，则是代表着一个图组件，分类算法最后完成的分类则是对图组件的分类。分类结果的信息包含每一个组件的基本 FQDN 集与 IP 集信息以及预测结果等，以 XML 文件的形式呈现。检测结果文件的部分属性说明如表 5-7 所示。原型系统对目标网络进行持续监测，并基于 1h 时间间隔策略，给出预测结果，其中一次实验的预测结果如图 5-9 所示。

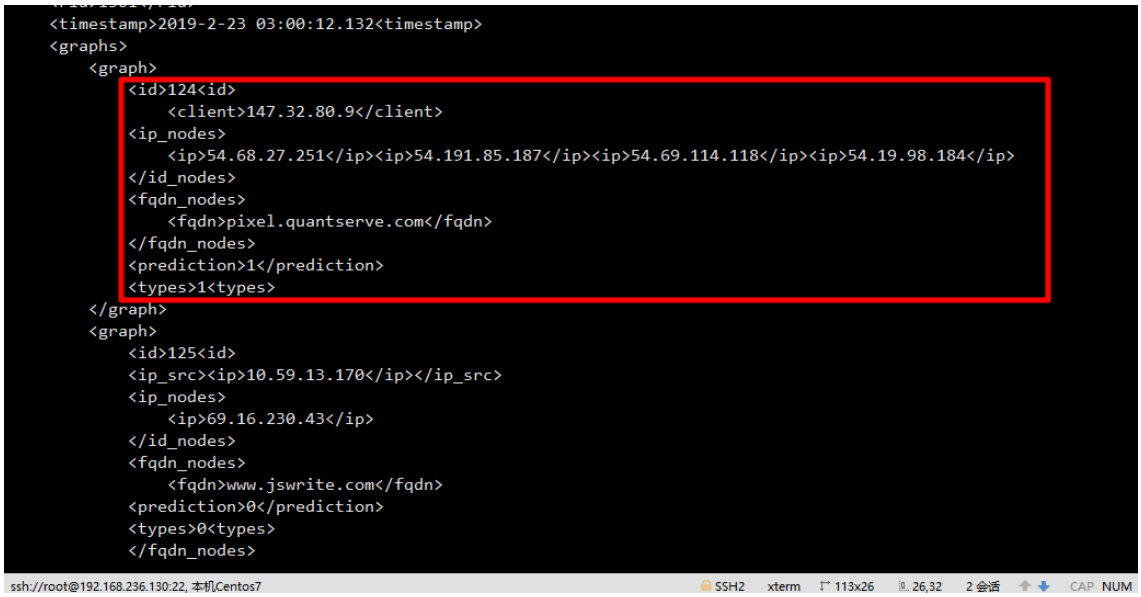

图 5-9 在线分类检测结果

5.5 本章小结

本章节主要在第三章提出的基于 DNS 映射关联的 Fast-flux 与 Domain-flux 僵尸网络检测方法的基础上进行对比实验，综合对比了 4 种不同分类算法在不同的验证情况下分类结果，结果显示本文选择的基于 LightGBM 的分类算法具有相对较高的分类准确率与精确率。

随后再根据第四章设计的高速网络下的原型检测系统，测试原型流量抓取采集的效率性与可行性，测试原型系统对于 DNS 流量的过滤效果，然后测试系统最后的分类检测效果。实验结果证明，原型系统能够高效地完成对目标网络流量的采集及过滤，并能够较为准确地分类预测恶意图组件，即预测疑似僵尸网络流量的 IP 集合与域名集合，并给出可疑僵尸网络流量的类型，即 Fast-flux 类型或 Domain-flux 类型。

第六章 总结与展望

6.1 工作总结

互联网技术的飞速发展，对人们的日常生活产生了深刻影响。如今互联网时代，各行各业已开始“互联网+”[53]式的转变，因此对互联网的依赖程度不断加深，随之而来的安全问题也愈发凸显。近些年网络安全的攻击事件也逐渐进入大众的视野，引发行业人员对于网络安全的再度深思。而这其中僵尸网络再度成为研究热点，随着网络技术的进步，僵尸网络也在发生着新的变化，越来越多新的技术被应用到新型僵尸网络中，其中 Fast-flux 与 Domain-flux 技术较为突出。

本文则是针对 Fast-flux 与 Domain-flux 两种类型的僵尸网络做深入研究，提出一种基于 DNS 流量映射关联图分析的检测方法，并基于该检测方法，设计并实现应用于高速网络的原型系统，完成僵尸网络的检测。本文主要的研究工作包括以下部分：

1. 僵尸网络的原理研究：本文针对僵尸网络的定义进行研究总结，根据拓扑结构与命令控制协议两种方式对僵尸网络进行分类。并针对目前僵尸网络中常见的逃逸检测的方式做了分析与总结。

2. DNS 协议研究：重点研究 DNS 协议的技术规范，分析其层次结构与报文格式，成为后续对 DNS 协议流量的过滤基础。

3. Fast-flux 技术原理研究：本文研究 Fast-flux 的技术原理，并分析其在僵尸网络中应用的两种类型的 Fast-flux 技术特点。

4. Domain-flux 技术原理研究：该部分则是分析 Domain-flux 的含义以及其应用场景，并研究该技术背后的重点算法——DGA 算法。

5. 基于 DNS 映射关联图分析的检测方法研究：本文根据 Fast-flux 与 Domain-flux 两种技术的原理以及其在 DNS 流量上的重要特点，提取基于 DNS 映射关联图分析的检测方法。并针对该检测方法，详细描述了该方法总体的流程设计以及算法设计。然后根据方法的流程划分，详细设计各流量的具体实现，包括流量的采集与预处理、DNSMap 映射关联处理、图组件的特征分析与提取，以及针对特征选择合适的分类算法。

6. 高速网络下僵尸网络检测原型系统设计与实现研究：在具体的检测方法的基础之上，本文根据其实际应用场景，设计并实现符合高速网络的检测原型系统。针对该系统，设计其网络部署结构以及总体架构。对于流量的抓取，该系统选择 PF_RING+Zero Copy 的方式完成流量的采集抓取，能够极大减少丢包率。随后针

对流量分别做 DNS 过滤，预处理。然后再利用 DNSMap 做图映射关联分析、特征提取以及机器学习二分类。

6.2 工作展望

本课题设计实现了一种基于 DNS 映射关联图分析的 Fast-flux 与 Domian-flux 僵尸网络检测方法，实验结果显示该检测算法具备较高的准确率与精确率，符合设计要求。本文还基于该检测方法设计并实现高速网络下的检测原型系统，并测试系统各功能模块，达到预期效果。但整个检测方法的设计与原型系统的设计仍有诸多不足之处，还需后期深入研究改进。

1. DNS 流量局部特征：本文的检测方法是从 DNS 映射关联图的角度出发，然后结合图中各元素的属性特征，对于边的特征，即 DNS 查询响应，本文选取了典型特征 TTL，而在实际的 DNS 查询中，还有很多其他字段未考虑进来，这些字段在僵尸网络中也有着特殊的统计特性，对于 DNS 流量的局部特征分析仍存在不足。

2. DGA 算法生成域名的特征分析：对于 Domain-flux 类型的僵尸网络的检测，DGA 算法是绕不开的一个话题，本文是基于域名的层级结构，选择域名字符的部分统计特征，但对于 DGA 产生的域名，其实还包含中很多不明显可见的字符特征，这部分特征还可借助深度学习等方式来做特征挖掘，在这个方面本文的检测方法稍显不足。

3. 图关联处理效率：对于借助开源工具 DNSMap 来完成 DNS 的映射关联，但阅读其源码，然后结合本检测方法，其实对于输入文件的预处理部分，其实还可以做到提起的预过滤，而本文还未做到其代码深入优化，对于关联处理效率存在一定的提升空间。

4. 分类算法优化：考虑到系统设计时对于分类算法要求的高效性，所以通过对比目前主流的分类算法，最后选择了 LightGBM，而对于该分类算法，对于其算法核心优化能力笔者稍感欠缺，后续可调参以达到更好地优化效果。

5. 检测系统实时性：对于检测结果，原型系统是基于时间间隔的策略来给出检测结果，这样的策略是基于系统可长时间持续分析处理的可行性而考虑，对于高实时性，存在不足，后续可通过优化系统流程，通过异步处理等方式来解决。

6. 检测结果可视化：原型系统是通过设计图组件的数据结构，选取其中关键数据来呈现检测结果，且检测结果以文件的方式给出，对于交互友好型有所欠缺，后续还有很多的改进空间。

致　谢

转眼间，三年的研究生生涯已接近尾声，如今毕业之际，不禁会拾起三年教研室生活的记忆碎片，有欢声笑语，有奋笔疾书，更有那指尖在键盘上迷人舞步以及那噼里啪啦的协奏曲，一切是那样地令人遐想与回味。三年中，是欢笑，有挫折，更有进步与收获。借此撰写论文之际，向三年来陪我一路走过的家人、老师、同学致以最真诚的感谢。

首先，我要感谢我的导师杨浩淼老师、张小松教授，研究生生涯，有他们对我学习科研的孜孜教导，其次衷心感谢牛伟纳对于我在论文研究上极具建设性的指导，尤其她那样忘我的科研精神更为令人感动。还有感谢陈瑞东老师对于项目工程上的交流帮助，还有感谢科研大神的室友何总、无所不知的游 god、时刻敦促我学习的王总以及热心可爱的范范同学。

最后，还要感谢我的妻子小花花。三年的求学之路，是她的陪伴让科研不觉枯燥乏味，是她的支持让未来更觉阳光灿烂。

参考文献

[1] 刘光强.论网络信息安全的重要性[J].城市地理,2015(3X):249-250

[2] 诸葛建伟,韩心慧,周勇林等.僵尸网络研究[J].软件学报,2008(03):702-715

[3] 方滨兴,崔翔,王威.僵尸网络综述[J].计算机研究与发展,2011,48(08):1315-1331

[4] 国家互联网应急中心. CNCERT 互联网安全威胁报告-2018 年 10 月[R].北京：国家互联网应急中心.2018

[5] 赛门铁克公司.2018 年互联网安全威胁报告[EB/OL]. https://www.symantec.com/zh/cn/security-center/threat-report,2018

[6] 阮斌.《2017 年中国网络安全报告》发布[J].计算机与网络,2018,44(05):58-59

[7] 李可,方滨兴,崔翔等.僵尸网络发展研究[J].计算机研究与发展,2016,53(10):2189-2206

[8] 靳冲.Fast-Flux 网络检测与分析技术研究[D].北京:中国科学院研究生院,2011

[9] Katz O, Perets R, Matzliach G. Digging deeper-an in-depth analysis of a fast flux network [EB/OL]. https://www.akamai.com/us/en/multimedia/documents/white-paper/digging-deeper-in-depth-analysis-of-fast-flux-network.pdf,2017

[10] 苑朋朋,王常青.DNS 安全威胁及应对措施研究[J].网络空间安全,2018,9(05):50-54

[11] 赵相男.基于模糊聚类的僵尸网络反规避技术研究[D].北京交通大学,2018

[12] 吴迪,崔翔,刘奇旭等.泛在僵尸网络发展研究[J].信息网络安全,2018(07):16-28

[13] 安全牛.世界最成功的僵尸网络使用 Fast Flux 技术躲避检测 [EB/OL]. http://www.aqniu.com/hack-geek/16971.html,2016

[14] Sandiflux.Another Fast Flux infrastructure used in malware distribution emerges [EB/OL]. https://www.proofpoint.com/us/threat-insight/post/sandiflux-another-fast-flux-infrastructure-used-malware-distribution-emerges,2018

[15] FORCEPOINT.2017 年网络安全状况[EB/OL]. https://www.forcepoint.com/zh-hans/resources/whitepapers/2017-state-cybersecurity,2017

[16] CenturyLink.Century Link 2018 Thread Report[EB/OL].http://lookbook.centurylink.com/threat-report,2018

[17] Perdisci R, Corona I, Giacinto G. Early detection of malicious flux networks via large-scale passive DNS traffic analysis[J]. IEEE Transactions on Dependable and Secure Computing, 2012, 9(5): 714-726

[18] Bilge L, Sen S, Balzarotti D, et al. Exposure: A passive dns analysis service to detect and report malicious domains[J]. ACM Transactions on Information and System Security (TISSEC), 2014, 16(4): 14

[19] Shi Y, Chen G, Li J. Malicious Domain Name Detection Based on Extreme Machine Learning[J]. Neural Processing Letters, 2017: 1-11

[20] Grill M, Nikolaev I, Valeros V, et al. Detecting DGA malware using NetFlow[C].Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on. IEEE, 2015: 1304-1309

[21] Pereira M, Coleman S, Yu B, et al. Dictionary extraction and detection of algorithmically generated domain names in passive DNS traffic[C].International Symposium on Research in Attacks, Intrusions, and Defenses. Springer, Cham, 2018: 295-314

[22] Wang K, Huang C Y, Tsai L Y, et al. Behavior-based botnet detection in parallel[J]. Security and Communication Networks, 2014, 7(11): 1849-1859

[23] Sharifnya R, Abadi M. DFBotKiller: domain-flux botnet detection based on the history of group activities and failures in DNS traffic[J]. Digital Investigation, 2015, 12: 15-26

[24] 百度百科.0DAY 漏洞[EB/OL].https://baike.baidu.com/item/0DAY%E6%BC%8F%E6%B4%9E,2018

[25] 刘莹.计算机病毒分析及应对措施[J].科技信息(科学教研),2008(23):397-398

[26] 崔丽娟,马卫国,赵巍,景秋实.僵尸网络综述[J].信息安全研究,2017,3(07):589-600

[27] Stone-Gross B, Cova M, Cavallaro L, et al. Your botnet is my botnet: analysis of a botnet takeover[C].Proceedings of the 16th ACM conference on Computer and communications security. ACM, 2009: 635-647

[28] Stover S, Dittrich D, Hernandez J, et al. Analysis of the Storm and Nugache Trojans: P2P is here[J]. USENIX; login, 2007, 32(6): 18-27

[29] Hogben G, Plohmann D, Gerhards-Padilla E, et al. Botnets: Detection, measurement, disinfection and defence[J]. European Network and Information Security Agency, 2011

[30] Greengard S. The war against botnets[J]. Communications of the ACM, 2012, 55(2): 16-18.

[31] Skoudis E. The six most dangerous new attack techniques and what's coming next[C].RSA Conference (RSA'12). 2012

[32] Xu K, Butler P, Saha S, et al. DNS for massive-scale command and control[J]. IEEE Transactions on Dependable and Secure Computing, 2013: 1

[33] Quarterman J, Sayin S, Whinston A. Rustock botnet and asns[J]. TPRC ,2011

[34] Andriesse D, Rossow C, Stone-Gross B, et al. Highly resilient peer-to-peer botnets are here: An analysis of gameover zeus[C].2013 8th International Conference on Malicious and Unwanted Software. IEEE, 2013: 116-123

[35] Stone-Gross B, Cova M, Gilbert B, et al. Analysis of a botnet takeover[J]. IEEE Security & Privacy, 2011, 9(1): 64-72

[36] Anselmi D, Boscovich R, Campana T J. Special edition security intelligence report: Battling the restock threat.[EB/OL]. https://www.microsoft.com/security/portal/mmpc/research/research papers.aspx,2016

[37] Kerkers M, Santanna J J, Sperotto A. Characterisation of the kelihos. b botnet[C].IFIP International Conference on Autonomous Infrastructure, Management and Security. Springer, Berlin, Heidelberg, 2014: 79-91

[38] Jang D, Kim M, Jung H, et al. Analysis of HTTP2P botnet: case study waledac[C]. Communications (MICC), 2009 IEEE 9th Malaysia International conference on. IEEE, 2009: 409-412

[39] Josep Albors. Fast Flux networks: What are they and how do they work?[EB/OL]. https://www.welivesecurity.com/2017/01/12/fast-flux-networks-work,2017

[40] Salusky W, Danford R. Know your enemy: Fast-flux service networks[J]. The Honeynet Project, 2007: 1-24

[41] Ntop. High-speed packet capture, filtering and analysis[EB/OL]. https://www.ntop.org/ products/packet-capture/pf_ring,2018

[42] mawenjian. china-cdn-domain-whitelist[EB/OL].https://github.com/mawenjian/china-cdn-domain-whitelist,2017

[43] anonymoustian. white-domains[EB/OL].https://github.com/anonymoustian/white-domains, 2016

[44] Berger A, Gansterer W N. Modeling DNS agility with DNSMap[C].Computer Communications Workshops (INFOCOM WKSHPS), 2013 IEEE Conference on. IEEE, 2013: 387-392

[45] TRUONG DINH TU. HTTP-Based Botnet Detection Using Network Traffic Traces[D].东南大学,2015

[46] Wikipedia. BPF.[EB/OL].https://zh.wikipedia.org/wiki/BPF,2017

[47] Garcia S, Grill M, Stiborek J, et al. An empirical comparison of botnet detection methods[J]. computers & security, 2014, 45: 100-123

[48] Beigi E B, Jazi H H, Stakhanova N, et al. Towards effective feature selection in machine learning-based botnet detection approaches[C].2014 IEEE Conference on Communications and Network Security. IEEE, 2014: 247-255

[49] Saad S, Traore I, Ghorbani A, et al. Detecting P2P botnets through network behavior analysis and machine learning[C].2011 Ninth Annual International Conference on Privacy, Security and Trust. IEEE, 2011: 174-180

[50] Szabó G, Orincsay D, Malomsoky S, et al. On the validation of traffic classification algorithms [C].International Conference on Passive and Active Network Measurement. Springer, Berlin, Heidelberg, 2008: 72-81

[51] LBNL Enterprise Trace Repository.[EB/OL].http://www.icir.org/enterprise-tracing,2005

[52] Alenazi A, Traore I, Ganame K, et al. Holistic Model for HTTP Botnet Detection Based on DNS Traffic Analysis[C].International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments. Springer, Cham, 2017: 1-18

[53] 雷丽萍.新矛盾视域下加快建设网络强国探究[J].陕西行政学院学报,2019(01):70-74

攻读硕士学位期间取得的成果

发明专利

[1] 张小松,熊智鹏等.一种面向 dalvik 字节码控制流混淆方法[P].中国,CN107632832A,2017 年 9 月 27 日

[2] 杨浩淼,熊智鹏等. 一种隐私保护的超声波通信方法[P].中国,CN107147449A,2017 年 7 月 17 日

主研/参研项目

[1] 2016QY04X000. 基于****的****平台

[2] 2016QY04W0803. *******研究

获奖情况

[1] 2018 年：电子科技大学研究生一等奖学金、优秀研究生称号