中图分类号：TP393　　论文编号：183130308
学科分类号：520. 3040　　密　　级：

天津理工大学研究生学位论文

SDN 中基于扫描攻击检测的移动目标防御技术研究

（申请硕士学位）

一级学科：网络空间安全

研究方向：软件定义网络及安全

作者姓名：吕星璇

指导教师：韩俐 副教授

2021 年 3 月

Thesis Submitted to Tianjin University of Technology for the Master's Degree

Research on Moving Target Defense Technology Based on Scanning Attack Detection in SDN

By

Xingxuan Lv

Supervisor

Li Han

March 2021

摘要

传统网络由于其静态，可预知的攻击面，很容易成为入侵者攻击的目标。网络管理员与入侵者的信息不对称也加大了防御难度。扫描攻击是攻击者入侵的首要步骤，因此对扫描流量的检测，分类以及针对性地制定后续移动目标防御(Moving Target Defense，MTD)策略显得极为重要。软件定义网络(Software Define Network，SDN)是一种新型网络架构。SDN 控制器的集中控制和可编程性极大方便了管理员对网络设备的管理，转发策略的下发以及相关功能的扩展。MTD 的目的是生成变化的攻击面以增大攻击者入侵难度，而 SDN 网络的相关特性大大方便了该过程的实施。因此，本文在 SDN 环境下，基于深度神经网络(Deep Neural Networks，DNN)模型对网络中扫描流量进行检测，根据不同检测结果，设计相应的 MTD 技术保护网络设备安全。

本文主要工作如下：

（1）提出了一种 SDN 环境下基于 PCA-DNN 的扫描攻击检测模型。利用 SDN 架构特点，基于流表项进行特征提取。设计了数据采集模块从 OpenFlow 交换机中提取数据，预处理模块进行数据处理，以及 PCA-DNN 模块完成降维和分类的功能。此外，引入额外参数对 ReLU 激活函数进行改进。

（2）提出了一种基于响应数据随机化的 MTD 机制，并结合 SDN 控制器的可编程性进行相关架构设计和开发。为了解决传统 MTD 技术缺乏对网络状态的感知能力，盲目地进行 MTD 机制的选取问题。在第三章检测结果的基础上，分析扫描攻击的相关原理，通过控制器和 OpenFlow 针对性地对响应数据包关键字段进行跳变。增大攻击者对网络中存活主机和开放端口的检测难度，达到混淆攻击者，保护网络设备安全性的目的。

（3）模拟实验仿真。使用 Docker+OVS+Ryu 搭建实验平台，Scapy 生成数据流量，对本文设计进行实验并分析。实验结果表明扫描攻击检测模块对四种扫描流量的精确率和召回率均达到 98%，改进的 ReLU 激活函数改善了模型训练过程中的神经元死亡问题。MTD 功能模块增加了入侵者攻击难度，混淆其扫描结果。初步证明本文研究是有效的。

关键词：软件定义网络，扫描攻击，深度神经网络，移动目标防御，响应数据随机化

Abstract

Traditional networks are easy to be targeted by intruders due to their static and predictable attack surface. The asymmetry of information between network administrators and intruders also increases the difficulty of defense. Scanning attack is the first step for attackers to invade, so it is very important to detect and classify scanning traffic and formulate follow-up MTD strategy. Software Define Network (SDN) is a new network architecture. The centralized control and programmability of SDN controller greatly facilitate administrators to manage network equipment, issue forwarding strategies and expand related functions. The purpose of MTD is to generate changing attack surfaces to increase the difficulty of attackers' intrusion, and the related characteristics of SDN network greatly facilitate the implementation of this process. Therefore, in SDN environment, this paper detects the scanning traffic in the network based on DNN model, and designs corresponding MTD technology to protect the security of network equipment according to different detection results.

The main work of this paper is as follows:

(1) This paper proposes a scanning attack detection model based on PCA-DNN in SDN environment. Utilize the characteristics of the SDN architecture and perform feature extraction based on flow entries. The data acquisition module is designed to extract data from the OpenFlow switch, the preprocessing module performs data processing, and the PCA-DNN module completes the functions of dimensionality reduction and classification. In addition, additional parameters are introduced to improve the ReLU activation function.

(2) An MTD mechanism based on randomization of response data is proposed, and we combine the programmability of the SDN controller to design and develop related architectures. In order to solve the problem that traditional MTD technology lacks the ability to perceive the network status, it blindly selects the MTD mechanism. On the basis of the detection results in Chapter 3, the relevant principles of scanning attacks are analyzed, and key fields of response data packets are hopped through the controller and OpenFlow. Increase the difficulty of the attacker's detection of surviving hosts and open ports in the network, so as to confuse the attacker and protect the security of network equipment.

(3) We use Docker+Ryu+OVS to build experiments for simulation test, scapy

generates data traffic, and conducts experiments and analyses on the design of this paper. The experimental results show that the accuracy and recall rate of the scanning attack detection module for the four scanning flow entries are both 98%, and the improved ReLU activation function improves the dying ReLU problem during model training. The MTD function module increases the difficulty of the intruder's attack and confuses their scanning results. These prove that the design of this paper is effective.

Key words： Software defined network, Scanning attack, Deep neural network, Moving target defense, Response data randomization

目　录

第一章 绪论

1.1 研究背景与意义

MTD 技术是美国国家科学技术委员会近年来提出的新型网络安全技术[1]。与目前工程中的大多数入侵防御方法不同，它并非构建牢不可破的系统来保护网络安全，而是通过生成，实施以及分析多种动态变化的防御手段，通过增大攻击者的攻击面来增加资源消耗，提升攻击难度[2]。传统 MTD 技术大多缺乏对网络状况以及系统安全的动态感知，盲目的进行 MTD 策略的选取，导致了网络状态跳变的低效性，增大了资源的消耗。因此，面对现如今的网络安全威胁，如何制定快捷有效且针对性强的 MTD 策略显得尤其关键。

目前，网络安全形势复杂多变[3]，网络业务正在遭受各式各样的入侵和攻击，如木马上传，网络嗅探，网络扫描，越权访问，中间人攻击等。而网络扫描往往是攻击者后续渗透的首要步骤，攻击者通过扫描攻击获取操作系统指纹以及运行中的相关服务信息，为后续入侵提供思路和指引方向。因此，针对扫描攻击的有效检测，对制定后续 MTD 安全策略十分关键。

Nick McKeown 在 2009 年提出 SDN 的概念[4]，其转控分离的理念方便网络管理者对整个拓扑进行配置等相关操作[5]，控制器的可编程性也极大方便对其功能的拓展，使得网络管理更加自由灵活。近年来，多个国家和企业将 MTD 的思想应用到互联网设备中[6]，同时，研究者将 SDN 技术与 MTD 思想结合进行研究，利用控制器可编程性对网络设备进行功能拓展，降低入侵者成功几率[7]。

深度学习是常见的一种机器学习方式，它来自科研者对神经网络的研究。通过模仿人脑神经元对外部事物的学习机制来对网络进行训练，实现对未知样本的分类预测。近年来，已经有部分研究者将深度学习与 SDN 相结合解决网络安全问题[8]，借用控制器的全局视图快速定位网络中的恶意流量，从而实施对应的防御措施，并取得不错的准确率。

本文在以上所述研究背景下，在 SDN 环境下对异常扫描进行检测，基于 OpenFlow 协议进行相关数据的采集，利用控制器进行二次开发，免去在网络中部署相关采集设备。为改善传统 MTD 技术盲目跳变的现象，在检测结果的基础上，分析不同种类扫描攻击的原理，针对性的设计 MTD 机制。将 SDN 控制器对网络的灵活编排与动态防御相结合，是一个具有广阔前景和价值的研究工作。

1.2 国内外研究现状

如今，国内外众多科研人员将关注点转移到 MTD 技术之上。谢丽霞等提出了一种针对链路泛洪攻击的 MTD 机制[9]，通过拥塞监控和快速重路由等方法，对网络配置做出改变，从而恢复网络状态，并对恶意攻击进行缓解。张连成等人提出了一种基于路径与端址跳变的 MTD 技术[10]，通过路径跳变策略，降低攻击者对网络中某条特定路径的嗅探能力，端址跳变增加攻击者在单一路径上的攻击难度，实验表明该方案具有很好的抗网络截获能力，但当攻击者已知网络真实主机地址时，无法有效做出回应，同时多路径求解的开销较大。基于 MTD 思想，Stephen 等人提出了一种指令随机化技术[11]，通过对程序指令的异或和解异或进行系统防护。实验表明，该技术可以抵御部分溢出攻击以及代码注入攻击，但是需要特殊的硬件设备支持以及资源消耗。Jafarian 等提出了一种基于时空地址的 MTD 机制[12]，在空间随机化的基础上，增加时间参数，二者共同确定主机跳变的瞬时 IP 地址，增大了地址池的不可预测性，然而在现实网络中并没有对该架构进行实现。胡毅勋等人提出了一种基于 Dijkstra 算法的 IP 与端口跳变[13]，实验表明该方案在不影响网络通信的条件下，有效的增加攻击者网络嗅探的难度，但对网络设备的要求过高，同时无法对预先入侵的攻击者做出防御举措。为了增强网络跳变的针对性，景湘评等人提出了一种基于 Hurst 值的 MTD 技术[14]，在对网络流量检测的基础上基于 Web 服务进行后续跳变，实验表明，该方案降低了攻击者获取服务器信息的真实性，但 Hurst 值的确定需多次实验尝试，同时并没有对其他服务进行相关补充说明。

近年来，在关于网络扫描检测等领域，研究者进行了大量相关研究。主要归类总结为基于阈值[15]，基于概率[16]和基于规则[17]这 3 种检测方法。文献[15]实施起来操作十分简便，但阈值的选择较为困难。文献[16]具有较高的检测准确率，但误报率也较高。文献[17]对规则的精细度要求过高，且可移植性较差。

SDN 的集中控制和全局视图使得部分研究者将其与异常检测相结合进行研究。马超等提出了一种云环境下 SDN 的异常检测机制[18]，通过建立“主机连接历史”集以及“首次连接”队列，记录关于主机的相关信息，根据集合运算以及阈值比较分析主机是否异常。这种方法需要维持一个查询表维持后续操作，同时一定程度上影响 SDN 控制器的内存性能。Prabhakar 等人提出了一种混合机器学习与深度学习的检测框架[19]，通过数据平面的粗粒度监控与控制平面的细粒度分类器对 DDoS 攻击进行检测，使用 CICIDS2017 等数据集进行验证，结果表明具有较高的准确率以及较少的内存和时间开销。Kshira 等人提出一种基于信息距离的流量距离检测框架[20]，该框架能够在 SDN 环境下对 Flash 事件中的 DDoS 流量进

行检测，仿真结果表明，能够有效检测边缘交换机中的流量，从而尽早发出警报。王晓瑞等提出了一种基于反向传播（backpropagation, BP）的 DDoS 检测模型[21]，该模型具有较高的准确率，但特征的手工构建较为复杂，并且太深的模型层数会导致梯度消失现象的出现，影响检测结果。李鹤飞等提出了基于支持向量机（Support Vector Machine, SVM）的流量识别模型[22]，对现如今大流量网络数据训练难以实施。李传煌等使用混合深度学习算法来检测 DDoS 攻击[23]，基于 OpenFlow 交换机流表项构建特征。实验结果表明，该算法具有较高的检测精度，对软硬件设备的依赖性小等优点，但高维数据使计算开销过大。

针对以上研究者提出的方法与不足，本文提出了一种 SDN 环境下基于扫描攻击检测的 MTD 机制。基于 PCA-DNN 模型对异常扫描进行检测，并基于检测结果针对性设计 MTD 策略，减少跳变的盲目性。增加攻击者对网络设备状态信息的获取难度，达到降低安全风险的目的。

1.3 论文主要工作

本文以 SDN 架构为基础，提出了一种基于扫描攻击检测的 MTD 机制。设计 PCA-DNN 模型提高对网络扫描的检测精度，实施 MTD 策略增加攻击者难度，降低对系统安全性的危害。主要研究内容为：

（1）提出一种 SDN 中基于 PCA-DNN 的扫描攻击检测模型。该模型充分利用 SDN 架构特点。通过 OpenFlow 协议收集流表数据，避免部署硬件采集设备。基于 OVS 流表项生成模型的输入数据，减少过多人工特征的构建。使用 PCA 算法进行数据降维，确保准确率的同时节省后续计算开销。同时改进 DNN 中常用到的 ReLU 激活函数，增强其泛化能力并改善神经元死亡问题。实现了高效，准确的扫描攻击检测。

（2）提出了一种 SDN 中基于响应数据随机化的 MTD 机制。为解决传统 MTD 技术无先验条件的低效跳变问题，在第三章扫描检测结果驱动的基础上，针对性地对地址，端口，以及状态码进行随机跳变。为避免主机的参与，通过控制器完成响应数据包的生成和 MTD 流表的下发，混淆攻击者对网络节点和端口状态的获取结果，增大其扫描以及后续入侵难度，保护系统中的网络设备和相关信息。

（3）使用 Ryu+OVS+Docker 搭建仿真平台，Scapy 模拟生成扫描流量。设计对比实验对扫描攻击检测模型进行测试，记录结果并进行分析。对四种 MTD 子模块进行测试，验证各个子模块是否有效。

1.4 论文章节安排

本论文共 6 章，其章节安排如下：

第一章为绪论部分。首先介绍有关本研究的背景，引出该研究的意义，价值以及前景。接着概括了有关使用 SDN 进行异常检测，MTD 技术保护网络安全的国内外研究现状，然后对本文的主要工作做出总结，最后分章节对本论文的安排进行说明。

第二章为相关技术介绍。对 SDN 技术以及 OpenFlow 协议进行介绍，介绍面向 SDN 的 MTD 技术，分析了通过 SDN 技术解决安全问题的必要性和长处。总结阐述现有关于网络扫描攻击的检测方法。介绍关于扫描攻击感知的 MTD 技术，分析并说明基于扫描策略感知的 MTD 技术是十分重要的。

第三章说明了 SDN 中基于 PCA-DNN 的扫描攻击检测模型。首先说明了面向 SDN 的 PCA-DNN 的工作流程。然后分别从数据采集，预处理，以及 PCA-DNN 三个模块介绍了该检测模型的具体构成。引入额外参数改进 ReLU 激活函数，以此对模型训练中的神经元死亡现象进行改善。

第四章介绍了 SDN 中基于响应数据随机化的 MTD 机制。首先对本机制的系统架构进行阐述，对 MTD 跳变过程进行详细说明。然后为解决传统 MTD 技术的盲目低效跳变问题，在第三章模型检测结果基础上，对每种不同扫描攻击，针对性地提出了对应的 MTD 机制，介绍了跳变的具体内容以及相应流程。

第五章对基于扫描攻击检测的 MTD 进行效果测试与分析。介绍了本文测试工具的选择，网络拓扑的搭建过程和背景流量的生成。通过扫描攻击检测测试以及 MTD 效果测试对本研究的有效性进行验证分析。

第六章为总结与展望。归纳研究成果，分析现存不足，展望未来方向。

第二章 相关技术介绍

2.1 SDN 中的 MTD 技术

2.1.1 SDN 架构和 OpenFlow

5G、大数据以及云计算等技术的兴起，给传统网络带来了潜在的挑战[24]。SDN 的出现，使得网络管理者能够通过全局视图集中对网络进行控制，极大方便了对系统的操作和维护。

转控分离是 SDN 架构的核心，通过控制平面对数据平面（转发平面）进行管理，其中，控制器对网络拓扑的控制大多借助于南向接口来实现，包括流表下发，链路发现等。此外，网络设备也可通过此接口将反馈信息传递给 SDN 控制器。对上层应用来说，可通过北向接口进行网络业务的编程实现与信息交互。SDN 的架构如图 2.1 所示。

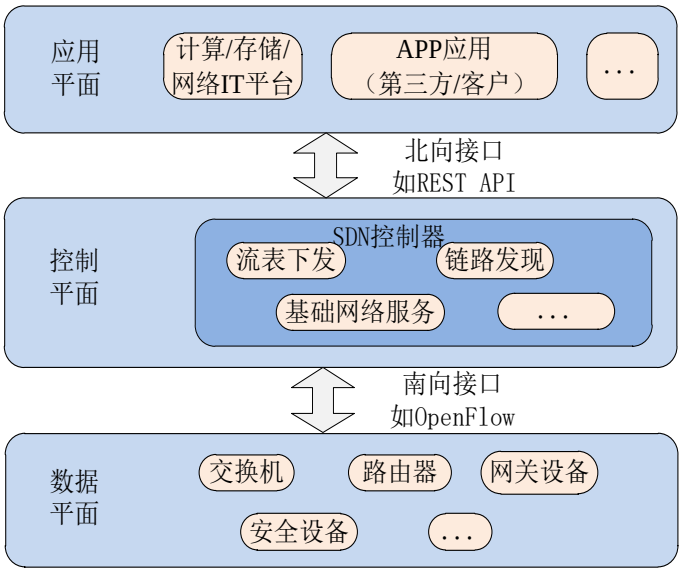

图 2.1 SDN 架构图

在 OpenFlow1.3 协议中，交换机和控制器通过安全通道进行通信。数据包的转发是通过流表匹配来进行的。1.3 版本的 OpenFlow 支持对传输层，网络层，链路层 3 层相关协议类型，端口号以及地址的匹配。OpenFlow 交换机构成如图 2.2 所示。

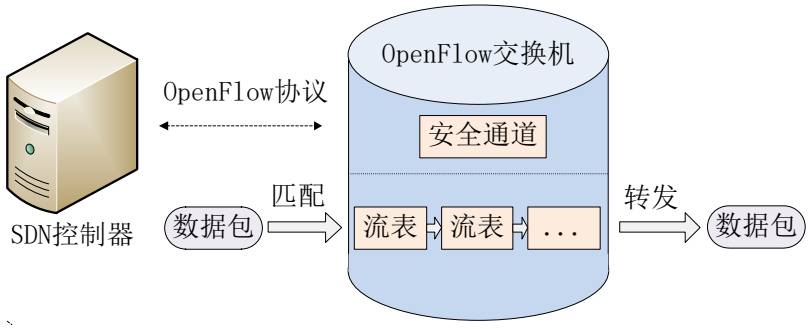

图 2.2 OpenFlow 交换机构成图

2.1.2 面向 SDN 的 MTD 技术

MTD 是 2011 年由美国研究者提出的一项技术。针对传统网络环境的静态性和可预测性，通过动态的，不可预知的攻击面来增加攻击者入侵难度。改善网络安全中易守难攻的现状。研究者将现有的 MTD 技术总结分为数据随机化，网络随机化以及应用随机化三类。

SDN 转控分离的特点为路由动态配置，网络协同管理等带来了新的解决方案。目前，许多研究者结合 SDN 网络结构，进行有关 MTD 技术的研究。SDN 中的 MTD 技术通常分为数据层 MTD，控制层 MTD 以及应用层 MTD 技术。

Jafarian 等人提出了一种 OpenFlow Random Host Mutation（OF-RHM）跳变机制[25]，该过程如图 2.3 所示。当数据包到达交换机时，控制器会进行请求授权并下发流表，对 IP 地址进行修改并完成数据转发，到达目的终端前交换机修改并还原数据包，这些操作对通信双方而言是透明的，并且 SDN 架构使实际网络环境中跳变的实现过程更加灵活便捷。实验表明，该机制能够有效的防御网络中的被动嗅探和中间人攻击，但并没有考虑到交换机的不可信因素以及其他类型的攻击。

端口跳变也是一种常用的 MTD 技术，研究者多关注于其跳变过程中的同步问题。目前的研究方向可分为时间戳同步[26]，ACK 确认同步[27]和严格的时间同步[28]。文献[26]中同步方式实现较为复杂，且部署难度过大。文献[27]需要将同步信息填充在数据报文中，容易被攻击者通过嗅探等方式获取并分析。文献[28]容易受到网络抖动的影响，稳定性较差。张连成等人在 SDN 环境下实现一种透明同步方式[10]，利用控制器在网络传输过程中对数据包进行随机变化，该方案不需发送同步数据包进行严格时间同步。实验结果表明，该机制能够在较低网络开销下保护网络安全，但无先验条件导致部分跳变是多余的。

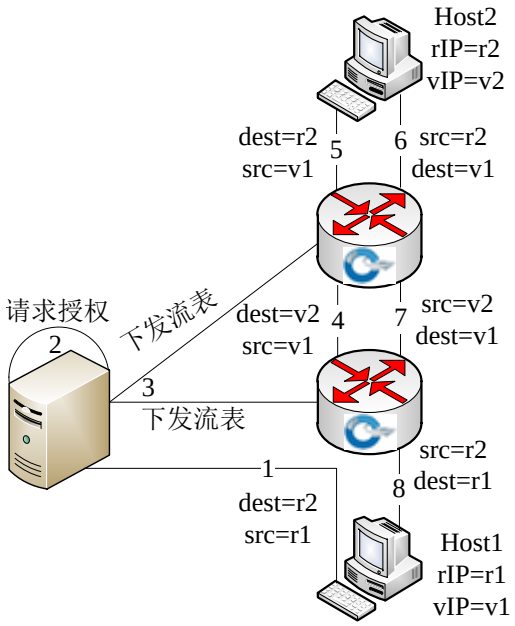

图 2.3 OF-RHM 过程图

为了解决 SDN 网络中的 Crossfire 泛洪攻击，谢丽霞等人提出了一种基于快速重路由的数据层 MTD 技术[9]，系统架构如图 2.4 所示。路由表基于 ICMP 监控而生成。链路监控模块统计负载和带宽，定时处理阻塞和重路由。追踪分析模块对网络中的跟踪路由（Tracert）进行捕获，并将数据传送给其他相关模块，最终选择新的转发路径。仿真实验表明，该方案能有效对网络中攻击流量进行缓解，但重路由本身的计算会导致传输时延的增加，同时频繁的路由策略会降低系统本身的鲁棒性。

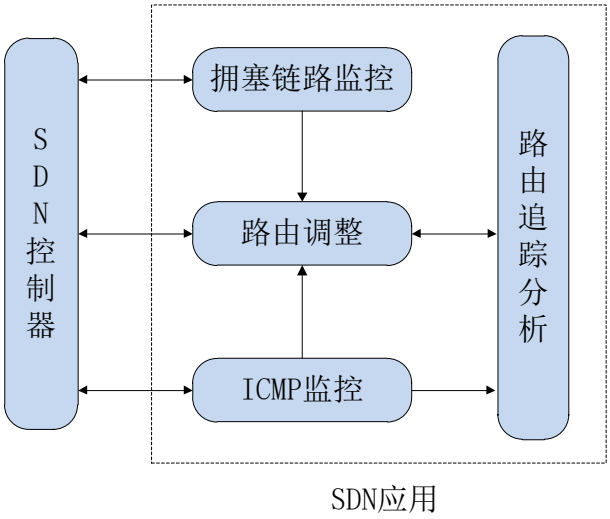

图 2.4 基于快速重路由的 MTD 架构图

SDN 控制器是 SDN 网络的核心，对于针对控制器的攻击，研究者进行了部

分相关研究。王红运等人提出了一种基于动态控制器分配的 MTD 机制[29]，控制器动态分配架构如图 2.5 所示。该架构将控制器分为两种：一种为管理和控制 OpenFlow 交换机的普通控制器，负责处理 Packet_in 消息，收集自身负载和网络资源状态信息等，另一种为控制器管理层的决策控制器，通过获取普通控制器运行状态，监测网络状态，对控制器可能出现的负载不均衡和其他故障进行处理。当普通控制器故障时，对其所通信网络设备重新分配控制器。而当出现问题的是决策控制器时，选择新的普通控制器为决策控制器。实验表明，该方法能够有效处理网络通信过程中的控制器故障事件，但决策控制器周期性的计算加大了网络的时延。此外，还有部分研究者将网络攻防双方抽象为博弈问题，通过纳什均衡等方式求解最佳防御策略。

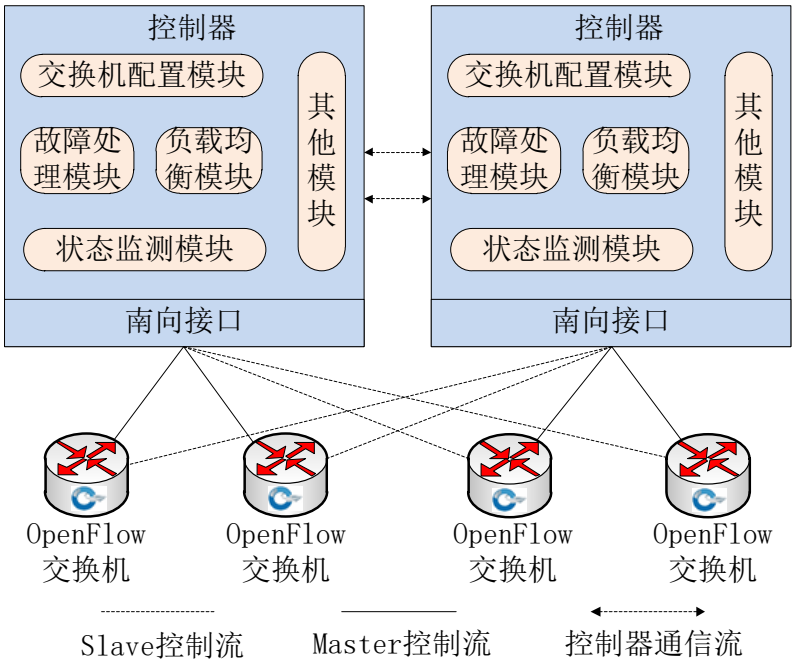

图 2.5 控制器动态分配架构图

目前，有关应用层的 MTD 机制研究尚处于起步阶段，主要分为基于虚拟机迁移的 MTD 技术[30]以及基于动态访问控制的 MTD 技术[31]。文献[30]通过动态地改变虚拟网络来分配资源，实验表明该方案能够有效地抵御 DDoS 攻击，但是关于网络成本与系统功能之间需要很好的权衡。文献[31]通过流量分析和网络状况对访问者进行动态访问控制，但该方案的有效性需进一步验证。

2.2 扫描攻击检测

网络扫描往往是攻击者入侵网络的首要步骤。攻击者通过对网络中主机等设

备的网络扫描，获取存活主机和相关服务信息，为实施下一步入侵提供思路和方向。除了第一章提到的 3 种常用的检测方法外[15-17],目前许多研究者将机器学习与异常检测相结合进行研究。

机器学习在入侵检测上的应用主要分为 3 个大类，即监督学习、无监督学习和半监督学习。监督学习需要标记样本从而进行训练，检测率较高。典型的监督学习算法有朴素贝叶斯分类（Naive Bayes Classifier，NBC）、决策树、SVM、神经网络等算法。无监督学习即聚类，数据不需要标签，缺点是检测率较低。半监督学习则是将二者进行结合研究，通过有标记样本的特征学习无标记样本。目前来说，当半监督学习的模型选择不恰当时，监督学习的性能或许会更高。

基于以上分析，对于一个样本数较大且均有标记的数据集，使用监督学习显然是一个能够获得较高检测率的好方法。

基于 NBC 的异常流量检测算法：NBC 假设特征条件独立，通过训练集对模型学习训练，之后对每个测试样本得到理论上分类结果。该方法理论扎实，算法简单。王玉栋等人提出了一种基于朴素贝叶斯的入侵检测方法[32]，通过受限玻尔兹曼机降低特征间的相关性。通过 KDD99 数据集的模拟实验表明，对分类器的改进是有效的。

基于决策树的异常流量检测算法：决策树是一种类似树形的分类模型，通过节点对数据特征进行判断，向下成新的分支继续判断其他特征，分类结果由最底层的叶节点给出。该算法过程易于理解，并且计算量较小。王远帆等人提出了一种基于决策树的端口扫描方式[33]。在 CICIDS2017 数据集上进行实验仿真，结果表明，该方案能够有效对入侵流量进行检测，但是未剪枝的模型可能发生过拟合现象。

基于 SVM 的异常流量检测算法：SVM 的核心点在于求解分隔超平面，该平面可以对数据进行分类，研究者通过几何间距评价平面选择的优良程度。该方法可以解决小样本下的分类问题，同时可用于高维数据的情况，避免局部最小值。张康宁等人提出了一种基于 SVM 的检测模型[34]。通过引入 Bagging 算法增强 SVM 对大规模数据的处理能力。通过 KDD99 数据集进行了仿真实验，结果表明，该方案对扫描攻击的检测准确率达到 93%，但对于多分类问题仍然存在困难。

卷积神经网络（Convolutional Neural Networks, CNN）是一种通过卷积计算提取数据蕴含信息的网络结构，经常被用于图片分类等领域。近年来，部分研究者将其应用于网络安全等领域。刘月峰等人构建了一种基于多尺度 CNN 的检测模型[35]，主要思想是将网络数据集通过维度填充转换为适合 CNN 的二维数据集。在 KDD99 上进行仿真验证，实验结果表明该模型大大提高了分类性能。但是模型训练时间较长，并且关于数据维度填充并没有给出相关证明解释。LSTM 网络

引入输入、输出和遗忘门解决 RNN 的梯度消失问题，通常用于序列数据的处理问题等领域，如语音识别，语言翻译等。高忠石等人使用 LSTM 网络进行入侵检测，取得较高检测率，但误报率也较高[36]。

DNN 是目前使用非常广泛的深度学习模型。通过模仿人脑神经结构和功能来学习。丁珊等人提出了一种基于 DNN 的入侵检测模型[37]，使用随机梯度下降算法进行训练。在 KDD99 上取得可不错的结果，但当迭代次数过多时，容易发生过拟合现象。吴成智等人提出了一种基于集成多分类器的 DNN 模型[38]。该模型通过 stacking 技术对四种常见分类算法进行集成,增强了检测算法的稳定性。在 KDD99 上实验表明对扫描攻击流量的检测准确率达到 99%,但是太深的 DNN 模型存在梯度消失现象。

2.3 基于异常流量感知的 MTD

由 2.1 节总结分析可知，目前的 MTD 机制大多缺乏对网络攻击的检测以及威胁的动态感知，导致网络系统盲目地进行 MTD 策略的选择。增大网络运行时的整体开销，并且一定程度上使网络防御的有效性降低。已经有部分研究者在攻击面上进行异常流量感知和检测，通过对不同入侵行为地分析，对下一步转移机制的选取和生成进行指引。

景湘评等人提出了一种基于 Web 服务的 MTD 技术[14]，为了解决服务的静态配置和攻防双方信息不对称的劣势，通过对网络 Hurst 值的测量，分析网络情况是否遭受入侵，进而确定合适的跳变频率，同时对 Web 服务的响应信息进行跳变，增加攻击者对系统信息判断的难度。进一步地，设计 IP 跳变子系统，通过 OpenFlow 交换机进行规则下发，SMT 约束求解地址范围。实验结果表明，该方案能够成功检测出网络中的异常情况，Web 响应信息跳变对攻击者获取的响应结果进行混淆，仅有 10%左右使得漏洞利用成功，IP 跳变使得扫描得到存活主机攻击结果无法对后续入侵提供帮助，证明该方法是有效的。

雷程等提出了一种攻击面自适应转换的 MTD 机制[39]，基于熵感知机制，分析扫描攻击，为后续不同 MTD 策略的选取提供指导。提出了基于视图距离的策略生成方法，通过不同的节点选取，增大 MTD 机制的不可预支程度。使用相关约束条件来降低该 MTD 机制的开销。并且通过有关理论对该机制抵御网络入侵的能力和所需运行成本进行分析计算，最后模拟实验也证实了其研究的有效性。

Miyazaki 等基于文献[40]的研究，提出了一种分布式的 MTD 机制[41]，通过启发式算法对网络入侵进行检测，控制器计算下一跳的路由转发地址，最终实现对网络入侵的防御，后续研究将要对大数据网络环境中的防御能力进行验证。

Wang 等人提出了一种 SDN 环境下针对网络侦查的 MTD 机制[42]，设计并实现了嗅探反射器保护网络拓扑。嗅探反射器包括扫描传感器，反射器和影子网络共三个组件。扫描传感器检测出异常流量，不同与传统防御机制，反射器不会丢弃或阻断这些流量，相反，它将反射到影子网络中。由影子网络生成混淆处理的扫描回复，增加攻击者对网络信息的获取难度。最后在小范围网络中对该研究进行测试，实验结果表明，攻击者只能从影子网络接收模糊扫描响应，嗅探器反射器能够有效防御各种网络扫描。

因此，如何实现对网络拓扑的扫描攻击检测，进行相关攻击信息的关联、融合，并且根据感知到的威胁信息，有目的性地选取并制定最优 MTD 策略，提升 MTD 机制的有效性是十分重要的。

本文第四章提出了一种 SDN 中基于响应数据随机化的 MTD 机制，根据第三章的检测结果，针对性地对网络中的扫描攻击进行防御，保护网络安全。

2.4 本章小结

本章主要介绍了研究所涉及的相关技术。首先介绍了 SDN 架构以及基于 OpenFlow 协议的数据包处理流程，对面向 SDN 的 MTD 技术进行总结概括。然后对现阶段关于扫描攻击检测的研究现状进行分类阐述，总结优点与不足。最后介绍了基于扫描攻击检测的 MTD 技术，说明对扫描攻击等网络威胁感知的重要性。

第三章 SDN 中基于 PCA-DNN 的扫描攻击检测模型

本章提出了一种 SDN 环境下的扫描攻击检测模型，该模型使用 OpenFlow 进行数据采集，通过 PCA 进行数据降维，DNN 进行分类训练，改进 ReLU 函数改善神经元死亡问题，从而判断网络是否发生扫描攻击。该模型通过 SDN 控制器集中采集网络数据，基于 OpenFlow 流表项交换机进行特征构建，充分利用 SDN 网络特点。

3.1 面向 SDN 的 PCA-DNN 检测模型

由 2.1 节可知，我们可以直接从 OpenFlow 交换机中提取原始数据。为了便于后续测试进行，本文添加预处理模块，之后连接 PCA-DNN 模型对流表进行分类。面向 SDN 的 PCA-DNN 的工作流程如图 3.1 所示。

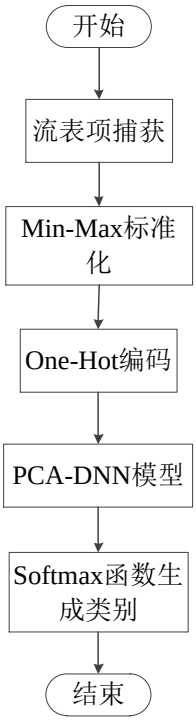

图 3.1 面向 SDN 的 PCA-DNN 工作流程图

为方便获取流表项数据，本文设置流表捕获时间间隔与 OpenFlow 交换机中 idle_timeout 时间一致。之后，控制器预处理模块对数据进行预处理，包括 Min-

Max 标准化与 One-Hot 编码[23]。然后 PCA-DNN 模型进行分类，PCA 算法进行降维，提取含足够信息的流表特征，确保高检测精度的同时节省模型计算成本。训练完成的 DNN 模型对数据特征进行计算，由 Softmax 函数生成类别。

3.2 数据采集模块设计

由 2.1 节可知，SDN 中数据包基于 OpenFlow 交换机流表项进行转发，当网络遭受扫描攻击时，会导致流表项部分字段值的改变，与正常网络情况存在差异，故可通过分析流表项对扫描攻击进行检测[43]。而 SDN 控制器可通过 OpenFlow 与交换机通信，省去额外采集设备的部署开销。因此，采集流表项进行后续分析是方便且有效的。数据采集模块流程图如图 3.2 所示。

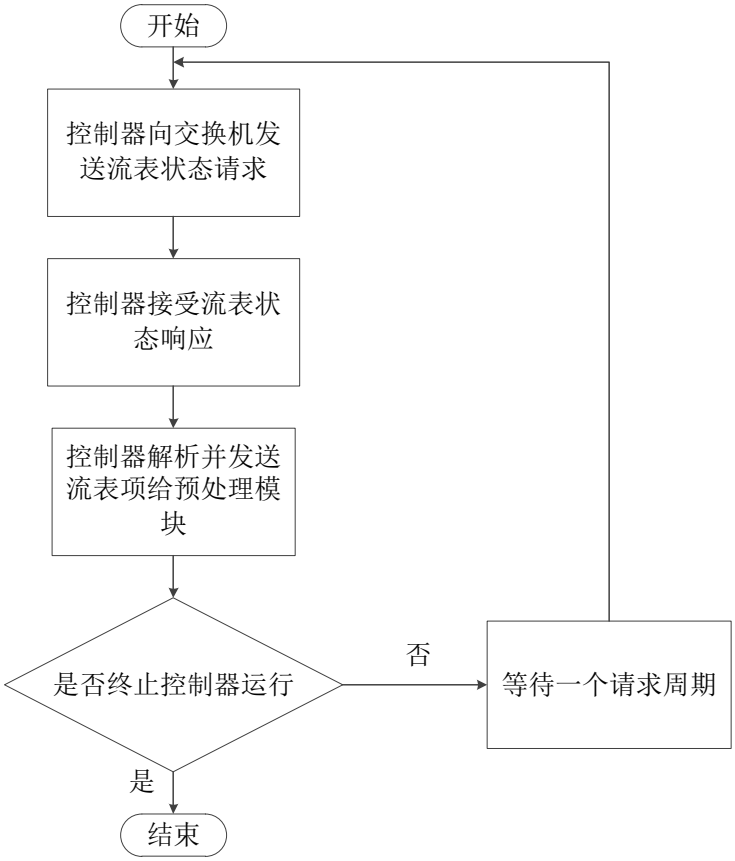

图 3.2 数据采集模块流程图

数据采集模块基于南向接口而设计。SDN 控制器通过发送 OFPFlowStatsRequest 请求获取 OpenFlow 交换机流表信息，交换机回复 OFPFlowStatsReply 响应。触发 EventOFPFlowStatsReply 事件，控制器捕获该事件并解析消息体获取流表项，发送给后续预处理模块。当终止控制器运行时，停

止数据采集流程。否则每经过一个请求周期后，控制器进行一次完整的数据采集流程。

3.3 流表预处理模块设计

数据采集模块收集到的流表数据特征如表 3.1 所示。其中，s 表示字符型特征，d 表示数值型特征。字符型特征并不能直接应用于后续 PCA-DNN 模型，同时，为了达到更好的测试效果，本文采取以下两种预处理方式。

表 3.1 流表特征

流表特征	说明
cookie(d)	流表标识字段
duration(d)	流表项生存时间
table(d)	流表 ID
n_packets(d)	已匹配数据包数
n_bytes(d)	已匹配字节数
idle_timeout(d)	控制器设置的超时时间
idle_age(d)	流表项没有匹配数据包的时间
priority(d)	匹配优先级
protocol(s)	匹配协议类型
in_port(d)	数据包进入交换机的入口号
dl_src(s)	源 MAC 地址
dl_dst(s)	目的 MAC 地址
nw_src(s)	源 IP 地址
nw_dst(s)	目的 IP 地址
tp_src(d)	源端口号
tp_dst(d)	目的端口号
actions(s)	OVS 对匹配数据包的动作

（1）Min-Max 标准化。对于某些分布差异较大的数据，如 duration，不同数量级的数据会影响后续模型分类结果。对其进行 Min-Max 标准化处理，使最终结果在 0 到 1 之间，处理公式为：

$$x^* = \frac{x-min}{max-min} \tag{3.1}$$

其中 *x* 为样本原始数据值，*min* 为样本最小值，*max* 为样本最大值，*x**为标准化后样本数据值。

（2）One-Hot 编码。对于某些特征值如 protocol（其包含 arp，tcp，udp 和 icmp 共 4 种字符型数据特征），模型无法直接处理这些数据。对其进行 One-Hot 编码，编码前后数据对比如表 3.2 所示。

表 3.2 部分编码前后数据

编码前	编码后
arp	0,0,0,1
tcp	0,0,1,0
udp	0,1,0,0
icmp	1,0,0,0

3.4 PCA-DNN 模型设计

3.4.1 PCA 降维算法

从 OpenFlow 交换机中提取出的原始数据，经过 One-Hot 编码后会产生冗余，影响后续 DNN 模型的分类结果。PCA 是一种常用的数据处理算法，基于矩阵变换进行降维。首先对样本数据进行预处理，之后计算处理后数据协方差矩阵的特征向量，并按照对应特征值大小进行排序，选取前目的维数的向量组成新矩阵，最后根据矩阵乘积计算的数学意义完成降维操作。

3.4.2 PCA-DNN 模型构建

DNN 是深度学习中常用到的一种神经网络模型，通过模仿生物体大脑功能和结构来学习。本文使用 DNN 模型进行输出预测，把模型预测结果与样本真实类别的差异（损失函数）作为评价分类准确度的标准，通过梯度下降算法训练模型，最小化损失函数，使模型分类结果接近真实类别，最终实现对未知类别样本的预测功能。

如图 3.3 所示，PCA-DNN 模型将预处理后 *k* 维流表项数据降为 *n* 维。DNN 层输出 y1，y2，y3，y4，y5 以解决五分类问题。实线圆为神经元，非实线圆为 dropout 技术随机失活。DNN 模型通常由输入、隐藏以及输出三层组成，下面从这三个方面对模型构建进行分析。

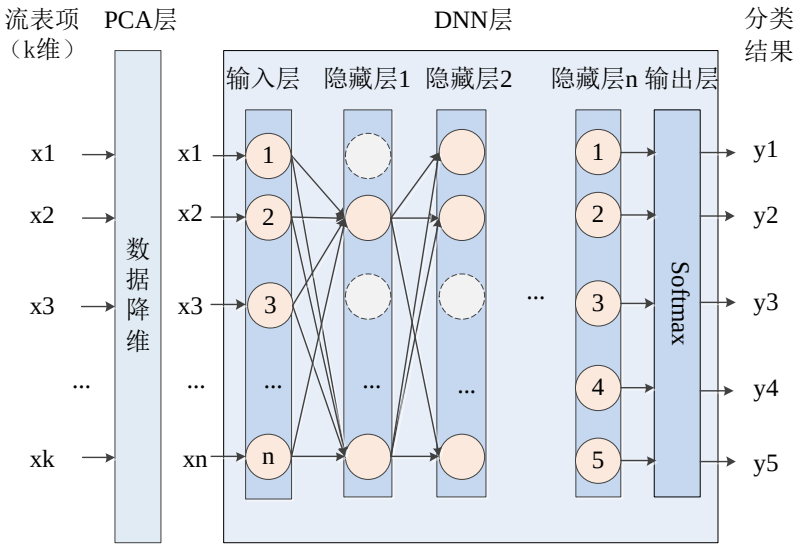

图 3.3 PCA-DNN 模型示意图

首先，输入层神经元个数与输入特征维度有关。在通常的模型设计过程中，输入维度特征的数量与输入层神经元个数相一致。本文模型根据这一点设计输入层神经元个数，既当 PCA 降维到 n 维测试结果最好，神经元数量就设置为 n。后续实验过程中对该数值进行测试选择。

其次，隐藏层参数的选择（层数、神经元数）对本文所提模型性能的影响十分关键。现如今，学术界还没有通用的方法进行确定，本文根据参考文献[44]中方法进行尝试选择。此外，当所用设备算力充沛的情况下，越深的 DNN 模型可以表达的函数越复杂。当处理相同数目的训练数据时，分类性能或许会更优越。然而当模型分类效果已满足我们研究需要时，过分追求隐藏层的深度和模型准确率可能会画蛇添足。因此，在接下来的实验测试过程中，我们将尽可能的减少此类现象的发生。

最后，本文通过分析要解决的问题来决定输出层神经元的数量。例如，我们通过划分不同种类型的扫描攻击，比如 ICMP 扫描攻击的为一类，ARP 扫描的为另一类等。使得最终输出层神经元个数与模型的分类数相同，且每个神经元只标识一种类别。本文共分类 4 种扫描攻击和 1 种正常流量，因此是一个五分类的问题，所以最终设定输出层神经元个数为 5。本文采取离线训练方式，对经预处理后数据进行训练，完成预定训练次数后保存模型并结束训练流程。训练流程图如图 3.4 所示。

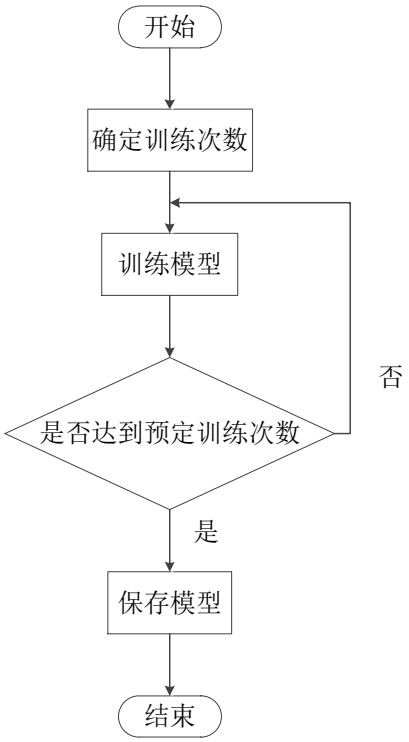

图 3.4 PCA-DNN 训练流程图

3.4.3 关键参数设计

（1）梯度下降算法。梯度下降算法是 DNN 等模型中常用的参数更新算法，但其本身也存在一些缺陷[45]。

缺陷一：选取恰当的学习率比较困难，当学习率设置过小时，模型收敛速度缓慢。而较大的学习率容易使模型训练时在损失函数极值点附近产生震荡，可能最终无法收敛。

缺陷二：在模型训练过程中，其全部参数的学习率都相同，神经网络参数非常多，损失函数对不同参数的梯度不同，根据更新频率高低对不同参数使用不同学习率，能够使得模型参数更新效果更优[46]。本文采用基于 Adam 的方法进行参数更新。通过梯度的一阶、二阶矩估计调整学习率，使得每个参数都享有独自的学习率。同时，学习率的更新和上一次训练状态相关，具有更强的自适应性。

（2）过拟合。过拟合是 DNN 等深度学习模型中经常出现的一种现象。具体表现为模型在训练集上分类效果很好，在测试集上表现一般[47]。通常模型复杂程度越高，该现象越明显。本文采取 dropout 技术改善这一现象。

如 3.4.2 节中图 3.3 所示，实线为原始神经元，虚线为采用 dropout 技术后的神经元。Dropout 技术具体表现为在模型每次训练过程中，使得部分神经元连接权重归零，暂时使其失效，以此降低模型复杂度。

（3）激活函数。激活函数是 DNN 等深度学习模型的核心。分为 Sigmoid 类[48]和 ReLU 类[49]。由于激活函数本身的性质，梯度消失也是 DNN 中十分常见的一种现象。

ReLU 采用分段的函数来解决梯度消失问题，但负区间全部为 0，网络训练过程中不会激活全部的神经元，这种现象称为神经元死亡[50]。研究者提出了 LReLU[51]，PReLU[52]，RReLU[53]等负区间具有导数的函数解决此问题，它们的表达式一般归类为：

$$y=\begin{cases}x & x>0\\ \lambda x & x\le 0\end{cases}\ (\lambda\neq 0) \tag{3.2}$$

为了增强式（3.2）对噪声的鲁棒性和增强模型的收敛速度研究者提出了可以取负值的 ELU[54]。ELU 公式为：

$$y=\begin{cases}x & x>0\\ \alpha(e^{x}-1) & x\le 0\end{cases}\ (\alpha\neq 0) \tag{3.3}$$

3.5 改进的 ReLU 激活函数

为了增强式（3.3）的通用性，本文对其进行了改进，额外引入两个参数β和γ，通过α和β控制曲线的弯曲程度，通过γ控制曲线的平移距离。其公式为：

$$y=\begin{cases}x & x>0\\ \alpha\left(e^{\beta x}-\gamma\right) & x\le 0\end{cases}\ (\alpha\neq 0,\beta\neq 0) \tag{3.4}$$

根据激活函数在 x=0 点处连续的定义，即:

$$\lim_{x\to 0+}=x=0=\lim_{x\to 0-}=\alpha\left(e^{\beta x}-\gamma\right)=\alpha(1-\gamma) \tag{3.5}$$

计算出γ=1，最终改进的 ReLU 激活函数公式为：

$$y=\begin{cases}x & x>0\\ \alpha(e^{\beta x}-1) & x\le 0\end{cases}\ (\alpha\neq 0,\beta\neq 0) \tag{3.6}$$

通过调整α和β的值使改进的 ReLU 函数（以下统称为式（3.6））更加接近其他激活函数。当α设置为一个非常小的数字时，式（3.6）接近为 ReLU。当β接近

1 时，式（3.6）接近为 ELU。对于 LReLU，本实验发现当 LReLU 中λ较小，$\beta \gg \lambda$并且α乘以β接近参数λ，式（3.6）接近为 LReLU。λ的值越小，式（3.6）与 LReLU 越接近。本论文通过画图来证明这一点。

当α乘以β约等于参数λ时，计算式（3.6）与 LReLU 之间的差值：

$$\alpha\left(e^{\beta x}-1\right)-\lambda x=\frac{\lambda}{\beta}\left(e^{\beta x}-1\right)-\lambda x=\frac{\lambda}{\beta}(e^{\beta x}-1-\beta x) \tag{3.7}$$

在不同的λ值下，随着在x和β的变化，LReLU 和式（3.6）的差值如图 3.5,3.6,3.7 所示。当λ和β相同时，x越大，两个函数的差值越大。当x和β相同时，λ越小，两个函数的差值就越小。由此可证，当x和β相同时，λ的值越小，式（3.6）与 LReLU 越接近。

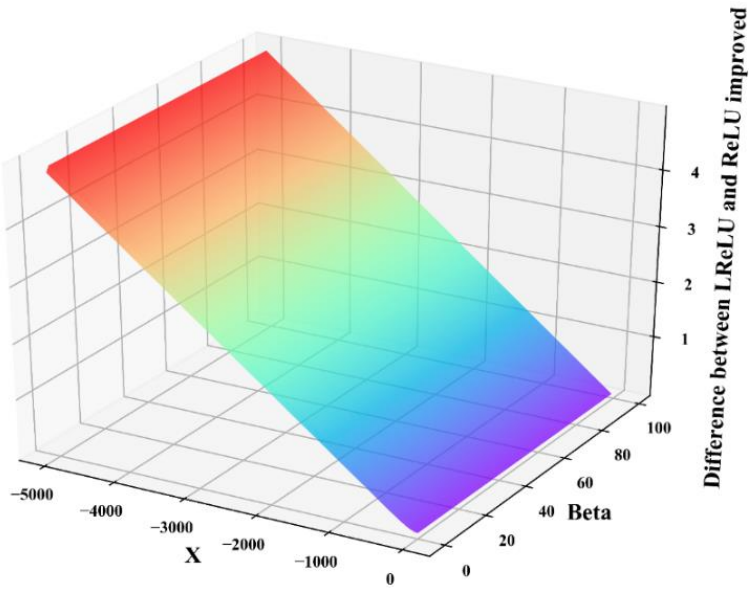

图 3.5 当λ=0.001 时函数的差值

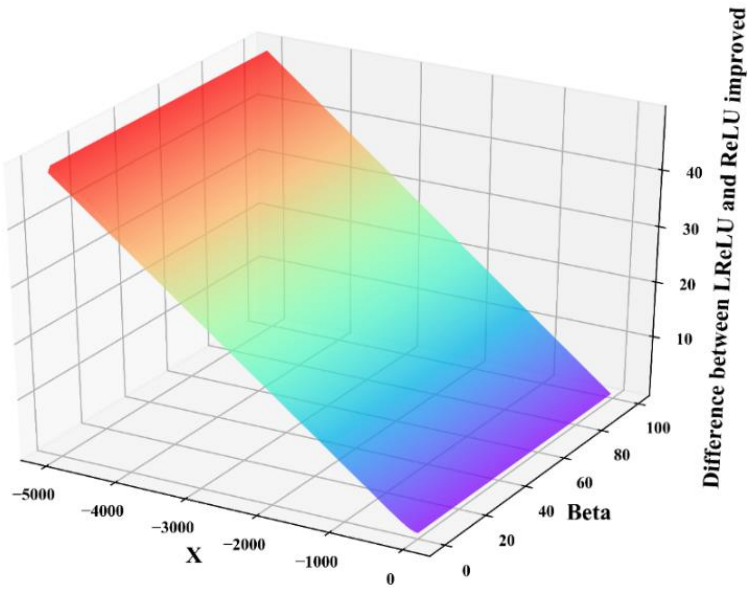

图 3.6 当λ=0.01 时函数的差值

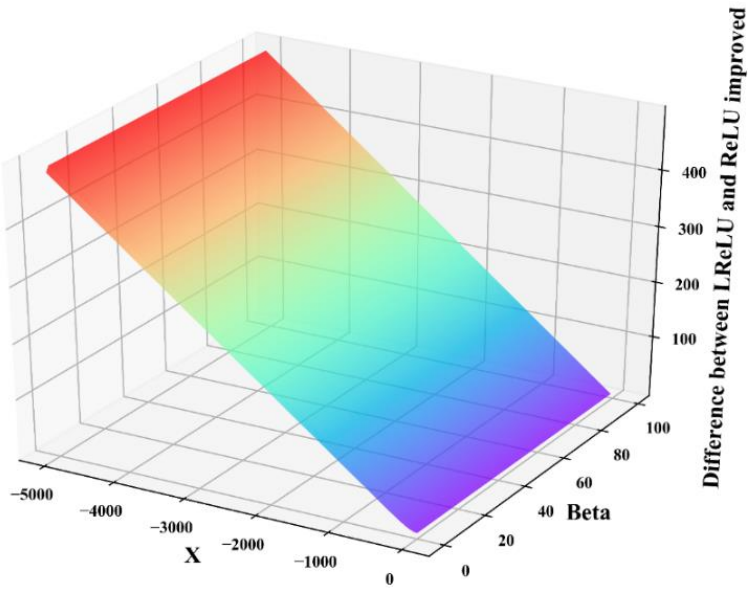

图 3.7 当λ=0.1 时函数的差值

3.6 本章小结

本章主要提出了一种 SDN 中基于 PCA-DNN 的扫描攻击检测模型。首先描述了检测模型的步骤流程，以数据采集，预处理以及 PCA-DNN 为基础，对模型整体架构进行了说明。之后具体说明每个模块的设计细节，包括基于 OpenFlow 协议的数据采集，Min-Max 标准化以及 One-hot 编码在预处理模块中的使用，PCA 算法原理，模型训练流程。同时详细说明了 DNN 模型中学习率，dropout 和

激活函数等关键参数的设计。针对模型训练过程中的神经元死亡现象，对 ReLU 激活函数进行了改进。本模型通过控制器集中进行数据采集，避免部署额外设备。基于流表项生成模型输入数据，充分利用 SDN 架构方便研究进行。

第四章 SDN 中基于扫描攻击检测的 MTD 技术

本章提出了一种 SDN 中的扫描攻击解决方案。为减少传统 MTD 技术的低效性和盲目性，该方案在第三章检测结果基础上，根据不同扫描流量类别，控制器随机化生成响应数据，进行交换机流表下发。该机制不涉及终端主机参与，充分利用 SDN 架构保护网络设备。对响应随机化和网络随机化进行安全性分析。

4.1 SDN 中基于扫描攻击检测的 MTD 技术架构

本文以 SDN 架构为载体，在第三章检测结果基础上，设计并部署相应的 MTD 防御模块。SDN 中基于扫描攻击检测的 MTD 技术架构如图 4.1 所示。

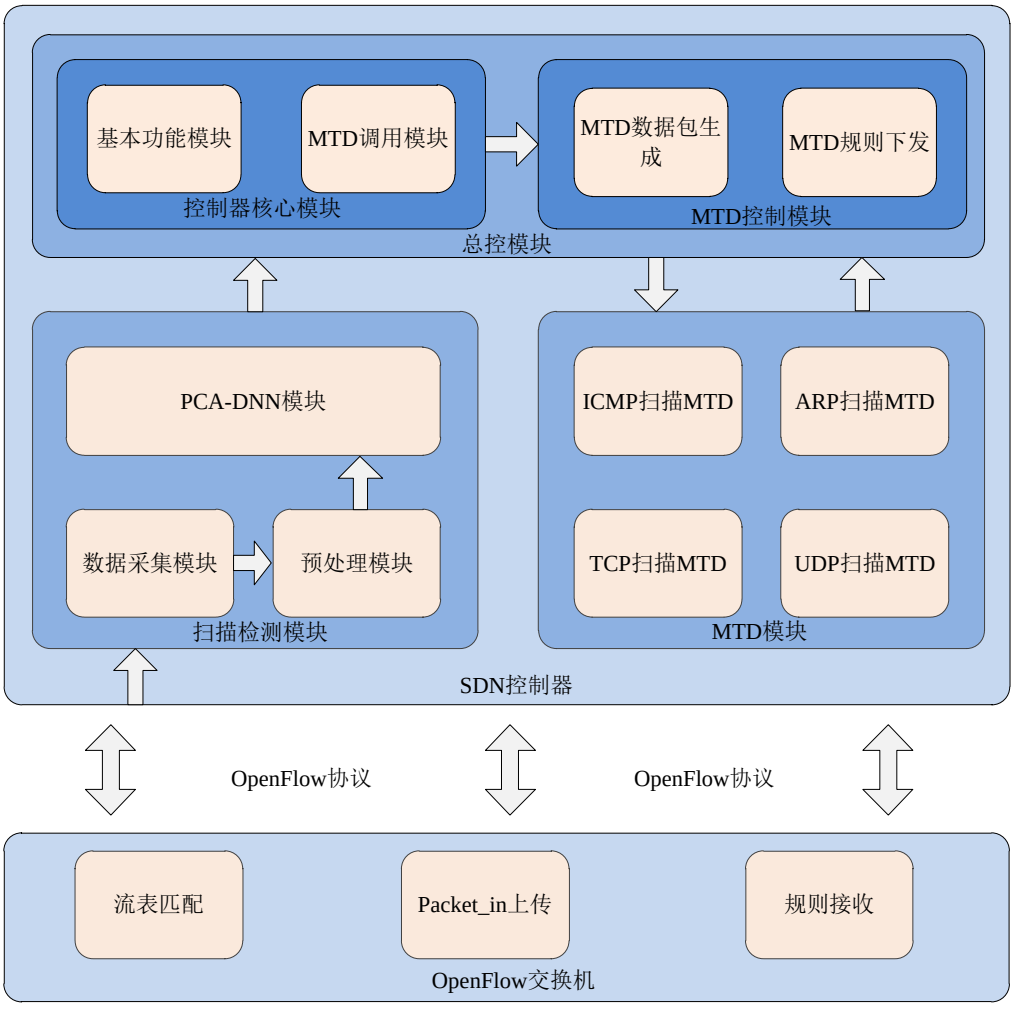

图 4.1 SDN 中基于扫描攻击检测的 MTD 技术架构图

如图 4.1 所示，SDN 中基于扫描攻击检测的 MTD 技术架构主要由扫描检测模块，MTD 模块以及总控模块组成。本节后续内容详细的对各个模块功能及组成进行阐述。

本文提出了基于扫描攻击检测的 MTD 机制，由模型检测结果，进行针对性跳变。用于解决传统 MTD 跳变的低效性问题。扫描检测模块包括数据采集，预处理以及 PCA-DNN 三个模块，数据采集模块通过南向接口定期采集 OpenFlow 交换机中流表项数据，为确保该模块能够采集所有流表项，采集周期的时间选择与流表项中 idle_timeout 一致。预处理模块通过 Min-Max 标准化以及 One-Hot 编码对流表项进行处理，确保输入数据适配训练完成的 PCA-DNN 模型。该模型将流表项分类结果发送给总控模块。该检测模块通过 OpenFlow 进行数据采集，避免部署额外采集设备，基于流表项生成模型输入数据，减少人工特征构建开销，通过控制器进行流表分类，充分利用 SDN 控制器集中控制和全局视图能力。

总控模块由控制器核心模块和 MTD 控制模块两个分模块构成。其中，控制器核心模块包括基本功能模块和 MTD 调用模块，本文基本功能模块包含传统 SDN 控制器基本功能的实现，如流表规则下发和策略生成等。MTD 调用模块接收扫描检测模块分类结果，当检测到网络设备遭受扫描攻击时，调用 MTD 控制模块请求 MTD 策略生成和实现。MTD 控制模块由 MTD 数据包生成和 MTD 规则下发两个子模块构成，该控制模块请求 MTD 模块根据扫描检测结果进行具体策略生成。MTD 数据包生成模块根据返回的 MTD 策略构造数据包并发送给攻击者，同时根据被扫描节点存在与否生成 MTD 规则并下发给 OpenFlow 交换机，之后攻击者发送相同数据包时，由交换机进行后续 MTD 操作，减少控制器负担，否则由控制器对该数据包执行 MTD 操作。该过程完全通过控制器和交换机实现，避免终端主机参与，充分利用 SDN 架构达到对网络设备的透明保护。

MTD 模块是本文架构中的重要模块，根据 MTD 控制模块发送的具体扫描攻击类别生成对应 MTD 策略。该模块由 ICMP 扫描 MTD，ARP 扫描 MTD，TCP 扫描 MTD 以及 UDP 扫描 MTD 四个策略生成子模块构成。本文 4.2 节中对各个子模块的设计进行详细阐述。

该系统架构除了上述 SDN 控制器及所含模块外，交换机完成基本流表匹配，Packet_in 上传以及规则接收等功能。基于扫描攻击检测的 MTD 技术流程如图 4.2 所示。

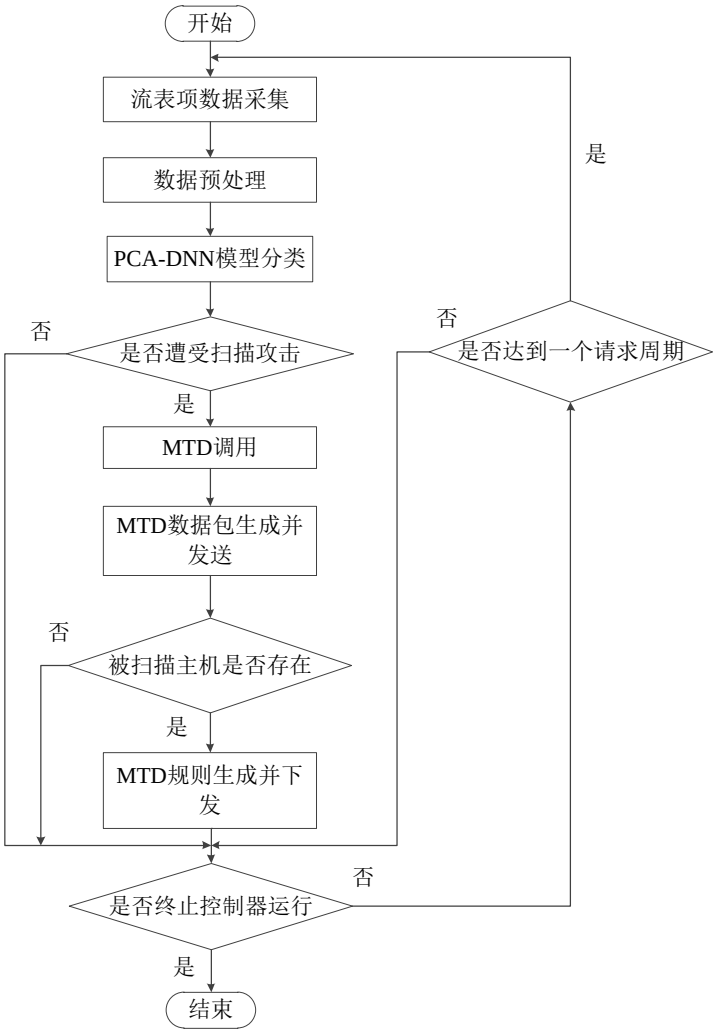

图 4.2 基于扫描攻击检测的 MTD 技术流程图

4.2 MTD 功能模块设计

传统 MTD 技术大多是无状态跳变，缺乏对网络状况的检测和感知，盲目和低效跳变加大了网络中计算资源的消耗。本文在 PCA-DNN 模型的检测结果基础上进行跳变，更加具有针对性。考虑到当网络中扫描攻击发生时，入侵者大概率已经知道了网络中部分节点的 IP 地址以及端口开放情况等其他信息。如果此时选择基于网络随机化的 MTD 技术，并不能防御攻击者对已掌握节点的继续渗透，具体分析见 4.3 节。同时，SDN 中基于应用随机化的 MTD 技术相关研究尚处于初步阶段，部分方法的效率评估和安全保证并没有完整的实验验证，且该类方法跳变开销过大。因此，本文选择基于响应数据随机化的 MTD 技术进行相关研究，针对 PCA-DNN 模型的检测结果，共设计 ICMP，ARP，TCP，UDP 四个扫描 MTD

子模块。下面对有关具体内容进行阐述。

4.2.1 ICMP 扫描 MTD 设计

ICMP 协议用于在网络设备间传递控制消息，而 ICMP 扫描则是网络入侵中经常发生的一种攻击手段，攻击者往往通过主动发送 ICMP 数据包，根据被扫描节点的返回结果对目的主机是否存活进行判断。为了混淆攻击者，本文对 ICMP 返回数据包的类型，状态码进行跳变，增大攻击者对网络节点扫描的复杂度。

本文构造混淆的 ICMP 响应数据包如表 4.1 所示。

表 4.1 ICMP 响应分类

序号	响应数据包分类	响应数据包分类描述	响应状态码	响应状态码描述
1	ICMP_ECHO_REPLY	ICMP 请求响应	ICMP_ECHO_REPLY_CODE	ICMP 请求响应
2	ICMP_DEST_UNREACH	ICMP 目的不可达	ICMP_HOST_UNREACH_CODE	ICMP 主机不可达
3	ICMP_TIME_EXCEEDED	ICMP 超时	ICMP_TTL_EXPIRED_CODE	ICMP 存活周期已满

本文通过对响应数据包类别和状态码组合进行 MTD 操作，当响应数据包类别为 ICMP_ECHO_REPLY 且状态码为 ICMP_ECHO_REPLY_CODE 时，表示被扫描主机在线。当响应数据包类别为 ICMP_DEST_UNREACH 且状态码为 ICMP_HOST_UNREACH_CODE 时，表示被扫描主机不在线。当响应数据包类别为 ICMP_TIME_EXCEEDED 且状态码为 ICMP_TTL_EXPIRED_CODE 时表示传输期间请求超时。

当 ICMP 扫描 MTD 模块工作时，无论请求数据包目的 IP 地址存活与否，都会进行回复。之后根据 random 函数生成一个 0 到 1 之间随机数，根据随机数大小，选择响应数据包的类别和状态码构造返回包发送给攻击者，达到混淆的目的。此外，若请求数据包中目的 IP 地址存活，SDN 控制器根据此次 MTD 操作中选择的参数下发具有一定存活周期的 MTD 规则，对后续通信过程中匹配的 ICMP 返回数据包进行修改。数据包生成算法如表 4.2 所示。

表 4.2 ICMP 响应数据包生成算法

算法：ICMP 响应数据包生成算法
输入：random 随机函数，ICMP 请求目的 IP 地址 IP，响应数据包字典 Response_dic={‘ICMP_ECHO_REPLY’:‘ICMP_ECHO_REPLY’, ‘ICMP_DEST_UNREACH’:‘ICMP_DEST_UNREACH’, ‘ICMP_TIME_EXCEEDED’:‘ICMP_TIME_EXCEEDED’}，参数 P1=1，P2=2，存活主机地址列表 IP_list
输出：ICMP 响应数据包
步骤 1：生成随机数 random_num=random(0,3)
步骤 2：当 random_num 小于 P1 时，转达步骤 3，否则转达步骤 4
步骤 3：选择 ICMP_ECHO_REPLY 类别和 Response_dic[‘ICMP_ECHO_REPLY’]状态码，转达步骤 7
步骤 4：当 random_num 小于 P2 时，转达步骤 5，否则转达步骤 6
步骤 5：选择 ICMP_DEST_UNREACH 类别和 Response_dic[‘ICMP_DEST_UNREACH’]状态码，转达步骤 7
步骤 6：选择 ICMP_TIME_EXCEEDED 类别和 Response_dic[‘ICMP_TIME_EXCEEDED’]状态码
步骤 7：当 IP 在 IP_list 中时，转达步骤 8，否则转达步骤 9
步骤 8：控制器下发 MTD 规则至 OpenFlow 交换机
步骤 9：输出 ICMP 响应数据包

在生成 ICMP 响应数据包时，先生成一个随机数，根据随机数大小选择响应数据包类别和对应状态码，生成该类数据包进行 ICMP 回复。当求目的 IP 地址在存活主机列表内时，需要额外下发 MTD 规则。最后生成 ICMP 响应数据包。攻击者收到的返回数据包是由该 MTD 模块生成的，并不能反映网络主机的真实状态，达到保护网络安全的目的。

4.2.2 ARP 扫描 MTD 设计

ARP 协议是将主机网络 IP 地址与物理 MAC 地址进行转换的协议。频繁的 ICMP 扫描极易被网络管理员发现，此外，网络中部分主机通过防火墙设置屏蔽 ICMP 数据包，使得 ICMP 扫描并不能正确得到主机存活状态。攻击者采用 ARP 扫描，向网络中主机发送 ARP 请求包，存活主机返回 MAC 地址响应。针对 ARP 扫描的 MTD 策略同样重要。因为数据链路层通信基于 MAC 地址，故本文对 ARP 返回数据包的 MAC 地址跳变，便可增大入侵者后续针对链路层的攻击难度。

当 ARP 扫描 MTD 模块工作时，随机生成一个 MAC 地址，之后构造 ARP 响应数据包回复攻击者，对攻击者本地 ARP 缓存表进行混淆。此外，若请求数据包中目的 IP 地址存活，SDN 控制器根据此次 MTD 操作中选择的参数下发具有一定存活周期的 MTD 规则，对后续通信过程中匹配的 ARP 返回数据包进行修改。数据包生成算法如表 4.3 所示。

表 4.3 ARP 响应数据包生成算法

算法：ARP 响应数据包生成算法
输入：ARP 请求目的 IP 地址 IP，存活主机地址列表 IP_list
输出：ARP 响应数据包
步骤 1：随机生成响应 MAC 地址 MAC_address
步骤 2：当 IP 在 IP_list 中时，转达步骤 3，否则转达步骤 4
步骤 3：控制器下发 MTD 规则至 OpenFlow 交换机
步骤 4：输出 ARP 响应数据包

在生成 ARP 响应数据包时，先随机生成响应 MAC 地址，当求目的 IP 地址在存活主机列表内时，需要额外下发 MTD 规则。最后生成 ARP 响应数据包。之后，当攻击者针对链路层发起攻击时，虚假的 MAC 地址使攻击数据包无法传输到目的主机，最终被交换机丢弃，一定程度上增大攻击者扫描难度。

4.2.3 TCP 扫描 MTD 设计

当攻击者获取在线主机的相关信息之后，下一步往往对主机运行服务进行探测，通过向目的主机某个端口发送数据包，根据数据包返回结果，判断相关服务是否正在运行，以便于后续渗透。该过程往往使用 TCP 扫描或 UDP 扫描，为了迷惑攻击者对扫描结果的判断，本文以 TCP SYN 扫描为例，对返回数据包的标示位进行跳变，增加攻击者判断端口开放状态的难度。本文构造混淆的数据包使用到响应标示位如表 4.4 所示。

表 4.4 TCP 响应标示位

序号	响应标示位	响应标示位描述
1	TCP_RST	TCP 复位标志
2	TCP_ACK	TCP 确认标志
3	TCP_SYN	TCP 同步标志
4	无响应	该端口被过滤

本文通过对 TCP 响应数据包的标示位组合进行 MTD 操作，当响应标示位为 TCP_RST+TCP_ACK 时，表示该端口关闭。当响应标示位为 TCP_ACK+TCP_SYN 时，表示该端口开放。当无响应时，表示被扫描主机防火墙对该端口进行了过滤。

当 TCP 扫描 MTD 模块工作时，无论请求数据包目的 IP 地址存活与否，都会进行响应。首先，根据 random 函数生成一个 0 到 3 之间随机数，根据随机数大小，选择响应标示位的类别构造返回包发送给攻击者，达到混淆的目的。此外，若请求数据包中目的 IP 地址存活，SDN 控制器根据此次 MTD 操作中选择的参数下发具有一定存活周期的 MTD 规则，对后续通信过程中匹配的 TCP 响应数据包进行修改。数据包生成算法如表 4.5 所示。

表 4.5 TCP 响应数据包生成算法

算法：TCP 响应数据包生成算法
输入：random 随机函数，TCP SYN 请求目的 IP 地址 IP，响应标示位列表 Response_flag_list=[‘TCP_RST’, ‘TCP_ACK’, ‘TCP_SYN’]，参数 P1=1，P2=2，存活主机地址列表 IP_list
输出：TCP 响应数据包或无输出
步骤 1：生成随机数 random_num=random(0,3)
步骤 2：当 random_num 小于 P1 时，转达步骤 3，否则转达步骤 4
步骤 3：选择 Response_flag_list[0]和 Response_flag_list[1]构成响应标示位，转达步骤 6
步骤 4：当 random_num 小于 P2 时，转达步骤 5，否则转达步骤 6
步骤 5：选择 Response_flag_list[1]和 Response_flag_list[2]构成响应标示位
步骤 6：当 IP 在 IP_list 中时，转达步骤 7，否则转达步骤 8
步骤 7：控制器下发 MTD 规则至 OpenFlow 交换机
步骤 8：当 random_num 小于 P2 时，转达步骤 9，否则转达步骤 10
步骤 9：构造并输出 TCP 响应数据包，退出算法过程
步骤 10：退出算法过程

在进行 TCP 扫描 MTD 操作时，首先生成一个随机数，根据随机数大小选择响应数据包标示位或者无响应，当请求目的 IP 地址在存活主机列表内时，额外下发 MTD 规则，最后输出 TCP 响应数据包或者本次 MTD 操作无数据包生成。攻击者通过 TCP SYN 返回数据包获取到的端口开放状态并不是真实的，对后续针对特定服务的攻击无法奏效，达到保护网络安全的目的。

4.2.4 UDP 扫描 MTD 设计

本小节介绍针对 UDP 扫描的 MTD 模块。UDP 协议是一种无连接的协议，攻击者通常根据 ICMP 响应包信息进行判断。本文对构造 ICMP 响应包的响应状态码进行跳变，或者不进行回应，增大攻击者对端口状态判断的难度。

本文通过对 ICMP 响应包状态码进行 MTD 操作，当 ICMP 类别为 ICMP_DEST_UNREACH 并且状态码为 ICMP_HOST_UNREACH_CODE 时，表示该 UDP 端口被防火墙过滤。当 ICMP 类别为 ICMP_DEST_UNREACH 并且状态码为 ICMP_PORT_UNREACH_CODE 时，表示该 UDP 端口关闭。当没有收到回复包时，表明该 UDP 端口开放。

当 UDP 扫描 MTD 模块工作时，首先，根据 random 函数生成一个 0 到 3 之间随机数，根据随机数大小，选择 ICMP 类别和状态码构造返回包给攻击者，达到混淆的目的。此外，若请求数据包中目的 IP 地址存活，SDN 控制器根据此次 MTD 操作中选择的参数下发具有一定存活周期的 MTD 规则，对后续通信过程中匹配的 ICMP 返回数据包进行修改。该过程中 ICMP 响应包生成算法如表 4.6 所示。

表 4.6 ICMP 响应数据包生成算法（UDP 扫描 MTD）

算法：ICMP 响应数据包生成算法
输入：random 随机函数，UDP 数据包目的 IP 地址 IP，响应数据包字典 Response_list=[‘ICMP_HOST_UNREACH_CODE’,‘ICMP_PORT_UNREACH_CODE’]，响应类别 ICMP_DEST_UNREACH，参数 P1=1，P2=2，存活主机地址列表 IP_list
输出：ICMP 响应数据包或无输出
步骤 1：生成随机数 random_num=random(0,3)
步骤 2：当 random_num 小于 P1 时，转达步骤 3，否则转达步骤 4
步骤 3：选择 ICMP_DEST_UNREACH 类别和 Response_list[0]状态码，转达步骤 6
步骤 4：当 random_num 小于 P2 时，转达步骤 5，否则转达步骤 6
步骤 5：选择 ICMP_DEST_UNREACH 类别和 Response_list[1]状态码
步骤 6：当 IP 在 IP_list 中时，转达步骤 7，否则转达步骤 8
步骤 7：控制器下发 MTD 规则至 OpenFlow 交换机
步骤 8：当 random_num 小于 P2 时，转达步骤 9，否则转达步骤 10
步骤 9：构造并输出 ICMP 响应数据包，退出算法过程
步骤 10：退出算法过程

在进行 UDP 扫描 MTD 操作时，首先生成一个随机数，根据随机数大小选择

响应数据包类别及状态码或者无响应，当求目的 IP 地址在存活主机列表内时，需要额外下发 MTD 规则。最后输出 ICMP 响应数据包或者本次 MTD 操作无响应数据包生成。攻击者收到 ICMP 返回包并不能确定真实的 UDP 端口状态，达到保护网络安全的目的。

4.3 安全性分析

4.2 节中对基于响应数据随机化的 MTD 操作进行了详细说明，本节对有关网络随机化以及数据随机化的安全性进行分析。

由 4.1 节可知，本文提出了基于扫描攻击检测的 MTD 机制，主要针对扫描检测结果进行响应数据随机化操作。传统网络随机化技术通过对数据包的 IP 地址，端口等信息进行跳变，使攻击者通过抓包等方式对数据包分析后，仅仅得到虚拟的通信地址。但是当网络扫描攻击发生时，攻击者大概率已经获取到网络中部分存活主机的真实 IP 地址，之后便可直接向某个网段进行扫描，通过是否有数据包回应便可直接判断主机存活状态。因此，在攻击者已知网络中一台或多台存活主机的 IP 地址等信息的情况下，传统网络随机化技术并不能抵御攻击者对网络中其他节点信息的获取。

本文在采取网络地址随机化的环境中进行对以上分析进行测试，10.0.0.1，10.0.0.2 为正常通信主机，10.0.0.3 为攻击者，用 wireshark 对通信流程进行抓包，如图 4.3，图 4.4 所示。从图 4.3 可知，网络随机化对 IP 地址进行跳变，10.0.0.1 跳变为 1.1.1.1，10.0.0.2 跳变为 2.2.2.2。但当攻击者已经知道 10.0.0.2 的真实 IP 地址后，可直接根据是否有数据包回应进行存活状态的判断。此时网络通信数据包如图 4.4 所示。因此，在扫描攻击已经发生的情况下，网络随机化并不能保证通信主机的安全性，本文通过响应数据随机化 MTD 机制，可增加攻击者对网络中节点存活以及端口开放的判断难度，是相对安全的。

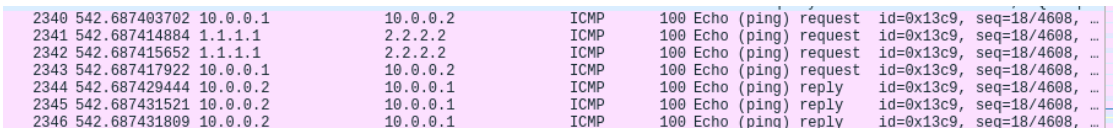

图 4.3 网络随机化过程图

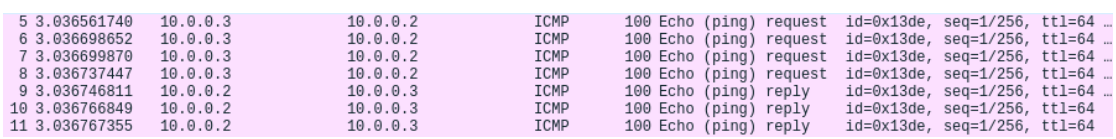

图 4.4 存活主机对攻击者请求回复图

4.4 本章小结

本章主要在第三章基于 PCA-DNN 扫描攻击的检测模型基础上，提出了基于响应数据随机化的 MTD 机制。首先，给出了 SDN 中 MTD 系统架构的整体设计。包括扫描检测模块，MTD 模块以及总控模块，并对各个模块以及本文在 SDN 中的 MTD 流程进行详细说明。之后重点阐述了四个 MTD 子模块的设计，对每个模块响应数据包的生成算法和响应过程进行说明。最后，在攻击者已知网络中某个节点地址信息的情况下，对网络随机化以及响应数据随机化的安全性进行分析和验证，4.3 节测试表明已经发生扫描攻击时，网络随机化是不安全的，第五章对本文 MTD 机制有效性进行验证。

第五章 基于扫描攻击检测的 MTD 效果测试与分析

本章在 SDN 环境下，通过模拟仿真测试，对 PCA-DNN 模型，以及基于响应数据随机化的 MTD 机制进行测试。统计并分析测试结果，对检测模型的分类性能，以及本文 MTD 机制进行评估。

5.1 测试环境搭建

5.1.1 测试工具选择

本文选择 docker 容器，OpenvSwitch（OVS）交换机，Ryu 控制器，Scapy 以及 TensorFlow 框架进行测试环境的搭建。

Docker 是一款应用容器引擎，相比于 VMware 而言，启动更加迅速，使用更加便捷，利用 docker 大大方便了网络中所需服务的部署。OVS 是一款仿真交换机软件，支持 OpenFlow 等协议，支持 ovs-docker 命令进行容器互联。Ryu 控制器是以一款使用 Python 进行代码编写的开源 SDN 控制器，代码简洁，便于二次开发和使用，目前支持 OpenFlow 协议 1.0 到 1.5 版本。Scapy 是一款 Python 编写的数据包生成与接发程序，同时也可以用于对数据包的解析。利用 Scapy，我们可以方便的进行背景流量以及扫描流量的生成等操作。Tensorflow 是一款深度学习框架，支持多种主流编程语言，使研究者不必拘泥于模型实现细节，更加关注于模型整体设计。

5.1.2 网络拓扑的搭建

本文在 3.60GHz 的 Intel Core i7-7700 CPU 的 DELL 上进行此仿真测试，使用 TensorFlow 框架在 PyCharm 中编写模型。选择 Ryu 控制器，OVS 交换机和 docker 来构建 SDN 拓扑。docker 启动了 3 台主机，其中 1 台是攻击者，2 台是服务器。运行 Apache 服务的服务器打开端口 80，DNS 服务开放 53 端口。测试拓扑如图 5.1 所示。

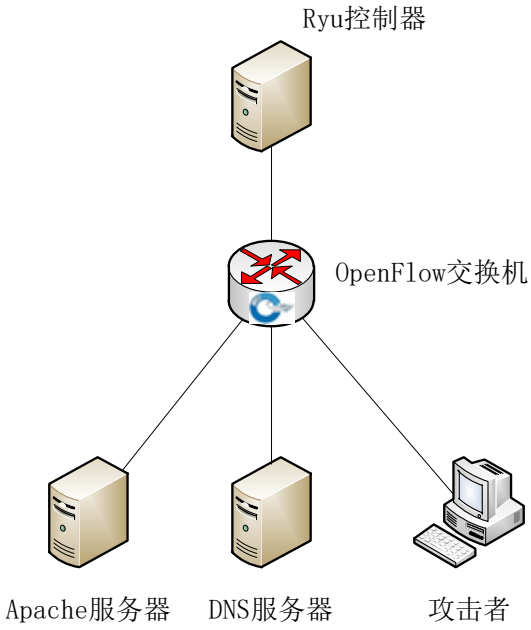

图 5.1 网络拓扑图

5.1.3 背景流量生成

本文参考文献[55]中有关网络中不同协议和端口的比例使用 Scapy 生成背景流量，然后在攻击者主机上生成 PING 扫描，ARP 扫描，TCP SYN 扫描和 UDP 扫描，部分扫描端口与背景流量中端口相同，设置扫描时间间隔都与 idle_timeout 保持一致，均为 5 秒。将收集的流表项记为正常和异常，其中异常涉及四种扫描类型。总共收集了 1178 个正常流表项和 1636 个异常流表项。将其中的 1876 作为训练集，将 938 作为测试集。收集的数据如图 5.2 所示。

```
cookie=0x0, duration=1.116s, table=0, n_packets=0, n_bytes=0, idle_timeout=5,
cookie=0x0, duration=1.158s, table=0, n_packets=1, n_bytes=54, idle_timeout=5,
cookie=0x0, duration=1.154s, table=0, n_packets=1, n_bytes=54, idle_timeout=5,
```

图 5.2 原始流表项数据集图

5.2 扫描攻击检测测试

为验证本文的 PCA-DNN 模型节省了计算成本并改善了神经元死亡问题，本文进行了 5 个测试。首先，通过 2 个预测试估计 DNN 层数及式（3.6）中α和β的值。接下来进行 PCA 测试证明本文的模型可以节省计算成本同时确保准确性。然后验证式（3.6）在解决神经元死亡问题上的有效性。同时对于 4 种具体的扫描

流量，用 3 种不同激活函数进行有关 Pre，Rec 的对比。最后与其他使用 PCA 算法的模型进行 Acc 评估对比。

5.2.1 性能评价指标

本文采用深度学习常用的准确率(Accuracy, Acc)，精确率(Precision, Pre)与召回率(Recall, Rec)3 种评价指标对模型性能进行评估。

Acc 是正确识别正常流量与扫描流量占总流量的比例：

$$Acc = \frac{TP+TN}{TP+TN+FP+FN} \tag{5.1}$$

Pre 是正确识别扫描流量占所有被识别为扫描流量的比例：

$$Pre = \frac{TP}{TP+FP} \tag{5.2}$$

Rec 是正确识别扫描流量占所有扫描流量的比例：

$$Rec = \frac{TP}{TP+FN} \tag{5.3}$$

真阳率(True Postive, TP)是扫描流量被识别出来的比例，真阴率(True Negative, TN)是正常流量被识别出来的比例，假阳率(Flase Postive, FP)是正常流量被识别为扫描流量的比例，假阴率(Flase Negative, FN)是扫描流量被识别为正常流量的比例。

5.2.2 预测试

在确定 DNN 层数的过程中，初步选取了层数分别为 3,4,5 的情况下进行测试。激活函数，batch size， epoch 以及学习率分别为 Tanh,1,30 和 0.0001。测试结果如表 5.1。

表 5.1 预测试 1 参数和结果

维度	层数	Acc(%)
30	3	84.64
30	4	79.10
30	5	87.42

考虑到达到一定检测精度后，尽可能简化模型并节省资源。同时太多的网络层数会引起梯度消失现象。由表 5.1 可知，梯度消失现象导致 Acc 几乎不会随网络层数增加而增加。同时，考虑到 3.4.2 节中的相关分析，本文剩余测试中选择的网络层数均为 5。

预测试 2 中 Batch size, epoch 分别设置为 25,90。剩下部分的参数与预测试 1 相同。测试参数和结果如表 5.2 所示。本文最终选择参数值为：$\alpha = 1, \beta = 0.5$。

表 5.2 预测试 2 参数和结果

参数选择	Acc(%)
$\alpha = 0.5, \beta = 0.5$	68.87
$\alpha = 0.5, \beta = 1$	68.87
$\alpha = 1, \beta = 0.5$	69.92
$\alpha = 1, \beta = 1$	69.30

5.2.3 PCA 降维对比测试

关于 PCA 的测试中首先对数据集进行特征筛选，计算 PCA 之后特征的可解释性方差[56]，如图 5.3 所示。本文选择了 3、8、25 和 30 共 4 个数据维度。降维到 3 维和 8 维数据中包含的可解释性方差分别达到 99.962%，99.999%，30 维为对照组，batch size 设置为 25。测试结果如表 5.3 所示：

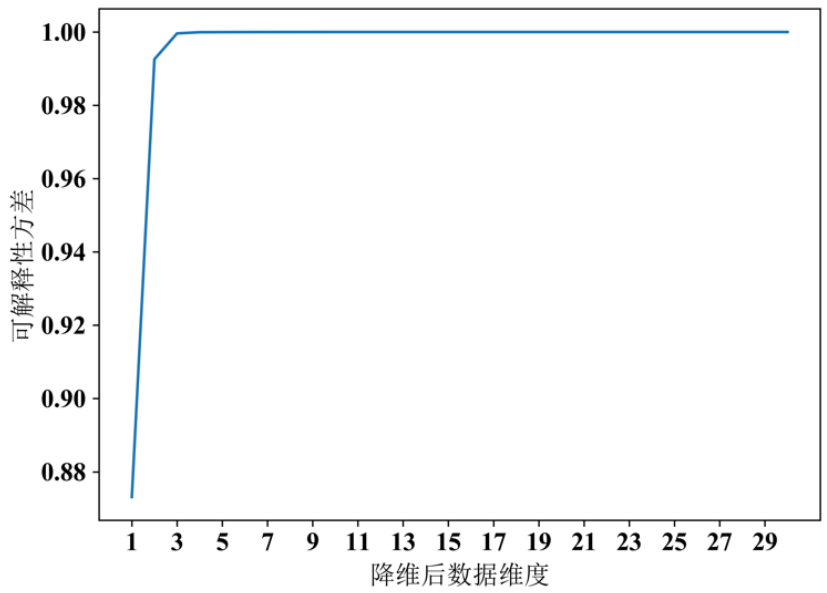

图 5.3 降维后数据维度与可解释性方差的关系

表 5.3 PCA 测试结果

激活函数	Sigmoid				ReLU				改进的 ReLU			
Epoch	90				90				90			
维度	3	8	25	30	3	8	25	30	3	8	25	30
耗时(s)	4.47	4.44	4.54	4.52	4.43	4.43	4.52	4.58	12.98	12.99	13.08	13.10
Acc(%)	42.22	42.22	42.22	42.22	68.87	68.23	70.79	69.83	68.87	68.87	69.10	68.12
Pre(%)	8.44	8.44	8.44	8.44	27.45	27.76	28.03	27.62	27.58	27.58	27.62	27.58
Rec(%)	20	20	20	20	37.33	38.17	38.32	37.61	37.54	37.54	37.70	37.54

由表 5.3 可知，PCA 对高维输入进行降维，3 种激活函数训练时间都有缩短。当维度为 25 时，(除 Sigmoid 函数外)，Acc，Pre 和 Rec 都有增加。需要注意的是，不同数据集的特性差别可能很大，因此关于维度值并不存在一个通用最优解，本文选择 25 维的维度进行后续测试。

5.2.4 改进的 ReLU 函数对比测试

为验证式（3.6）能够有效解决神经元死亡问题，本文又进行了对比测试。当发生神经元死亡时，一个有效的观察指标是随着 epoch 的增加，Acc 不会增加，batch size 为 50，测试参数和结果如表 5.4，图 5.4 所示：

表 5.4 关于神经元死亡问题的测试参数和结果

激活函数	神经元死亡区间	Acc(%)	Pre(%)	Rec(%)
Sigmoid	100-700	72.60	28.61	39.50
ReLU	700-900	90.06	68.80	78.70
改进的 ReLU	无	90.30	87.44	76.02

由表 5.4 和图 5.4 分析可知，当经过同样 epochs 次数的训练时，选择 Sigmoid 函数的模型频繁发生神经元死亡，最终 Acc 较低。当 epoch 次数过多时，ReLU 函数也会出现此问题。而式（3.6）在负区间导数不为 0，没有此现象发生。同时式（3.6）具有更高的 Acc 和 Pre。事实证明，本文关于 ReLU 函数的改进对于某些神经元死亡问题是有效的。

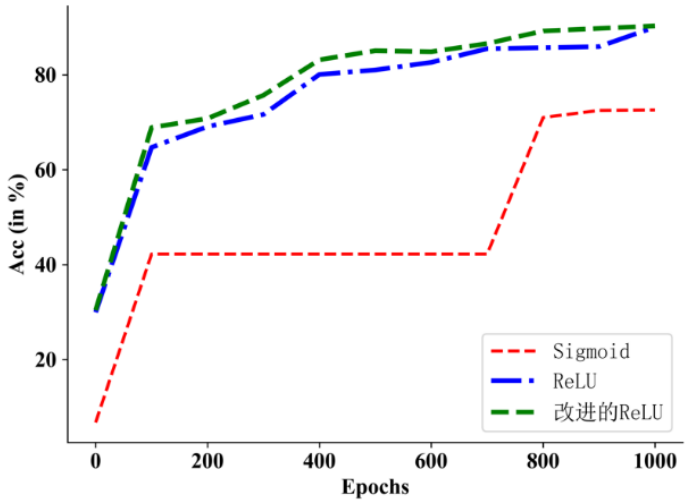

图 5.4 神经元死亡对比测试

5.2.5 扫描流量分类对比测试

为证明式（3.6）对具体扫描流量的识别有效性，本文进行测试。Batch size, epoch 分别为 25,1000。结果如表 5.5,5.6 所示。

表 5.5 4 种扫描流量分类的 Pre

激活函数	ARP	TCP	UDP	ICMP
Sigmoid	0	69.95	0	0
ReLU	83.54	99.48	98.39	69.77
改进的 ReLU	98.44	99.63	98.39	99.23

表 5.6 4 种扫描流量分类的 Rec

激活函数	ARP	TCP	UDP	ICMP
Sigmoid	0	99.41	0	0
ReLU	94.20	99.32	99.54	47.62
改进的 ReLU	89.96	99.26	99.19	98.20

由表 5.5,5.6 可知，对于 ARP，UDP 和 ICMP，Sigmoid 函数的 Pre 和 Rec 非常低，因为其软饱和导致梯度消失使得模型不收敛。在 ICMP 流量上，式（3.6）显著高于 ReLU，表明本文模型在识别 ICMP 流量方面更有效。

5.2.6 不同模型分类对比测试

本文还与 PCA-LSTM 和 PCA-随机森林方法进行比较。结果如表 5.7、图 5.5、图 5.6 所示。由表 5.7 所示，LSTM 对数据序列分布的标准性要求过高，而随机森林训练产生的过拟合现象，二者 Acc 都低于本文模型。同时由图 5.5 可知，PCA-DNN 模型的 AUC 值也高于二者。此外，由图 5.6 可知，在同一坐标轴中，本文模型的 PR 曲线距离右上侧更近。综上，本文模型的分类性能更好。

表 5.7 与 PCA-LSTM 和 PCA-随机森林对比

模型名称	Acc(%)
PCA-LSTM	89.45
PCA-随机森林	88.27
PCA-DNN	91.58

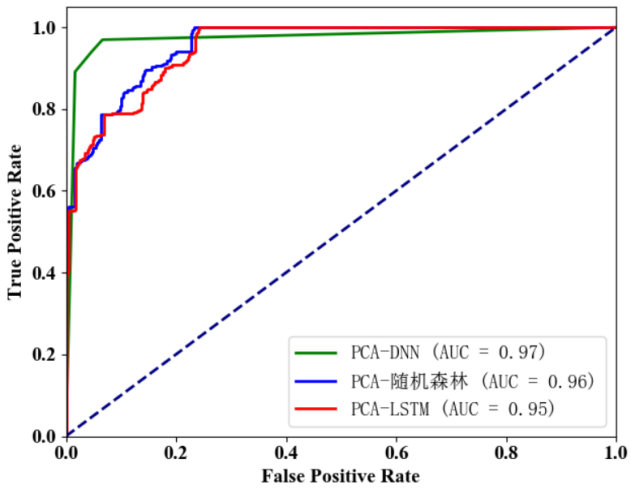

图 5.5 不同模型的 ROC 曲线图

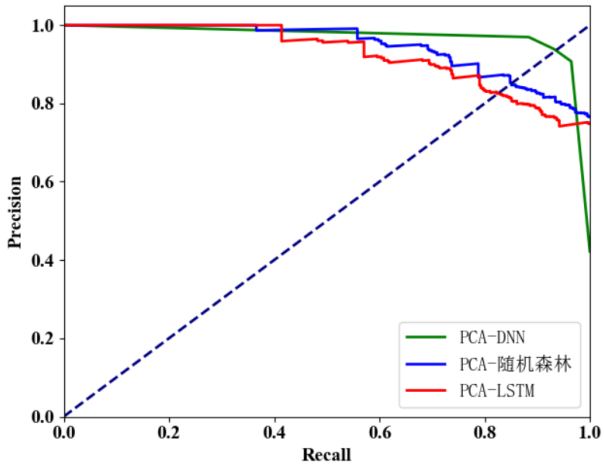

图 5.6 不同模型的 PR 曲线图

5.3 移动目标防御效果测试

5.3.1 ICMP 扫描 MTD 效果测试

本小节测试内容为：在攻击者主机上启动 ICMP 扫描工具，测试当 ICMP 扫描 MTD 模块启动时，攻击者检测到存活主机个数。一共进行 10 次测试，每次对攻击者所在网段进行全网扫描，记录并统计存活主机数与真实存活主机数。测试结果如表 5.8 所示。

表 5.8 ICMP 扫描 MTD 测试

测试次数	攻击者认为存活主机数	真实存活主机数	存活百分比
1	78	1	1%
2	84	1	1%
3	86	2	2%
4	85	0	0%
5	81	0	0%
6	67	0	0%
7	87	0	0%
8	94	1	1%
9	91	1	1%
10	82	1	1%

由表 5.8 可知，除第 3 次测试外，其余测试中，每次攻击者真正检测到存活主机占比最高约为 1%。而第 3 次测试可能因为随机数生成过程中，使得两台存活主机都被攻击者检测到，属于可以接受的情况。而大部分情况下，攻击者通过 ICMP 扫描所得到信息并不能有效的对其接下来入侵提供帮助。因此，可以认为该 MTD 模块是有效的。

5.3.2 ARP 扫描 MTD 效果测试

本小节次测试内容为：在攻击者主机上启动 ARP 扫描工具，测试当 ARP 扫描 MTD 模块启动时，统计并记录攻击者根据 ARP 响应包得到 MAC 地址。测试中，攻击者得到控制器生成的完全随机的 MAC 地址。

5.3.3 TCP 扫描 MTD 效果测试

本小节次测试内容为：在攻击者主机上启动 TCP SYN 扫描工具，测试当 TCP 扫描 MTD 模块启动时，攻击者检测到开放端口个数。一共进行 10 次测试，每次对攻击者对服务器 50-100 端口进行扫描，其中仅 80 端口开放。记录并统计开放数与真实开放端口数，测试结果如表 5.9 所示。

表 5.9 TCP SYN 扫描 MTD 测试

测试次数	攻击者认为开放端口数	真实开放端口数	开放百分比
1	15	1	7%
2	19	1	5%
3	12	0	0%
4	19	0	0%
5	14	0	0%
6	13	0	0%
7	14	1	7%
8	14	0	0%
9	17	0	0%
10	23	0	0%

由表 5.9 可知，在第 1 次，第 7 次测试中，攻击者检测到开放端口占比为 7%。其余测试中，每次攻击者真正检测到开放端口占比最高约为 5%。大多数情况下，攻击者通过 TCP SYN 扫描所得到信息并不能对其接下来入侵提供帮助。因此，可以认为该 MTD 模块是有效的。

5.3.4 UDP 扫描 MTD 效果测试

本小节次测试内容为：在攻击者主机上启动 UDP 扫描工具，测试当 UDP 扫描 MTD 模块启动时，攻击者检测到开放端口个数。一共进行 10 次测试，每次对攻击者对服务器 50-100 端口进行扫描，其中仅 53 端口开放。记录并统计开放端口数与真实开放端口数，测试结果如表 5.10 所示。

由表 5.10 可知，在第 1 次，第 2 次测试中，攻击者检测到开放端口占比为 7%。其余测试中，每次攻击者真正检测到开放端口占比最高约为 5%。大多数情况下，攻击者通过 UDP 扫描所得到信息并不能对其接下来入侵提供帮助。因此，可以认为该 MTD 模块是有效的。

表 5.10 UDP 扫描 MTD 测试

测试次数	攻击者认为开放端口数	真实开放端口数	开放百分比
1	21	0	0%
2	14	1	7%
3	14	1	7%
4	12	0	0%
5	17	0	0%
6	21	0	0%
7	22	1	5%
8	13	0	0%
9	18	0	0%
10	22	1	5%

5.4 本章小结

本章通过模拟仿真对 PCA-DNN 模型以及基于响应数据随机化的 MTD 机制进行了测试和分析。介绍了本研究是如何搭建测试环境，扫描攻击检测测试以及 MTD 效果测试和分析。测试结果表明，PCA-DNN 模型在确保检测准确度的同时能够有效减少训练时间，改进的 ReLU 函数对部分神经元死亡问题是有效的。同时对四种具体的流量分类情况进行测试，结果表明对每种流量的分类准确度都达到 98%以上，相比于 Sigmoid 以及 ReLU 激活函数，改进的 ReLU 函数在 ICMP 流量上分类性能更优。同时相比于其他模型，本文模型也具有更高检测准确率。对 MTD 效果测试结果的分析也表明，基于响应数据随机化的 MTD 机制能够增大攻击者对网络信息的获取难度，表明该机制是有效的。

第六章 总结与展望

6.1 研究工作总结

SDN 是一种新型的网络架构，其转控分离，可编程等特点极大方便了网络管理员对网络的控制和开发，当发现网络遭受入侵时，快速定位并做出响应。而网络扫描则往往是攻击者进行后续渗透的首要步骤。MTD 是一种新型的网络防御技术，通过不断变化攻击面，降低网络环境的可预知性来增加入侵者攻击难度。本文在 SDN 架构下，提出了一种基于扫描攻击检测的 MTD 机制。

本文主要工作总结如下：

（1）介绍了 SDN 网络，扫描攻击检测及 MTD 技术的发展，对相关技术结合研究进行简要说明。介绍了研究背景和意义，阐述目前国内外有关现状，分析目前相关研究存在的不足之处。

（2）对本文所涉及到相关技术进行介绍。介绍 SDN 架构以及 OpenFlow 技术，对面向 SDN 的 MTD 技术进行详细说明。对现有扫描攻击检测方案以及基于扫描攻击检测的 MTD 技术进行总结概括。

（3）提出了一种基于 PCA-DNN 的扫描攻击检测模型，使用 PCA 算法对数据进行降维，DNN 对处理后数据进行分类。对面向 SDN 的 PCA-DNN 检测模型过程进行说明。设计并说明了数据采集模块，预处理模块以及 PCA-DNN 模块。对 ReLU 激活函数进行改进，改善模型训练过程中的神经元死亡现象。本模型基于 OpenFlow 提取和生成数据，避免额外采集设备部署和人工特征构建，通过 SDN 控制器进行训练，充分利用 SDN 架构。

（4）在第三章检测结果的基础上，提出了响应数据随机化的 MTD 机制并进行相关架构设计。介绍了扫描检测模块，MTD 模块以及总控模块的功能和组成。对 MTD 功能子模块进行详细说明。最后，在网络遭受扫描入侵的情况下，进行网络随机化和本文方法的安全性分析。本文通过 SDN 控制器对响应数据进行跳变以及相应 MTD 流表下发，达到了对终端主机的透明保护，充分利用 SDN 集中控制的特点。

（5）模拟仿真测试，搭建 Ryu+OVS+Docker 的测试环境，通过 Scapy 工具模拟背景流量以及扫描流量。对 PCA-DNN 模型进行多方面实验测试，分析测试结果，进行 MTD 效果测试，统计结果情况。

6.2 未来工作展望

本文提出了 SDN 环境下基于 DNN 的 MTD 机制，设计并实现了基于响应数据随机化的 MTD 模块。仿真测试表明，当网络中发生扫描攻击时，DNN 模型能够有效进行检测，同时 MTD 模块增加了攻击者对网络中存活节点的探测难度。但是在模型设计和机制实现方面还有一些不足之处：

（1）在 PCA-DNN 模型中，本文提取 OpenFlow 交换机流表项作为输入特征。考虑基于流表项和时间等参数手工构造特征，增加模型输入数据对网络流量的表征性也是十分有价值的研究方向。此外，如何降低检测模型对网络通信的影响也是重点研究方向。

（2）在本文所设计的 MTD 机制中，实验中发现 OpenFlow1.3 版本并不支持细粒度的 TCP 扫描 MTD 流表。在实际网络环境中，入侵流量种类繁多，且每类又有多种不同的攻击方式。结合未来 OpenFlow 协议版本的更新，设计更具通用性的 MTD 机制是下一步的研究方向，此外，在真实网络情况下的可用性，时延性，与其他机制的效果对比也需进一步验证。

（3）在相关架构设计中，本文仅仅使用单一控制器进行功能模块的设计和开发。使用分布式控制器，将不同模块分开部署，节省单个控制器节点计算开销，也是未来的研究方向。

参考文献

[1] AYDEGER, ABDULLAH, SAPUTRO, et al. A moving target defense and network forensics framework for ISP networks using SDN and NFV[J]. Future generations computer systems: FGCS, 2019, 94: 496-509.

[2] 张晓玉，李振邦．移动目标防御技术综述[J]．通信技术，2013，46(06)：111-113．

[3] 张博卿，孙舒扬．2019 年中国网络安全发展形势展望[J]．网络空间安全，2019，10(01)：36-41．

[4] 张朝昆，崔勇，唐嚣祎，等．软件定义网络(SDN)研究进展[J]．软件学报，2015，26(1)：62-81．

[5] 谭晶磊，张红旗，雷程，等．面向 SDN 的移动目标防御技术研究进展[J]．网络与信息安全学报，2018，4(07)：1-12．

[6] FETAIS, NOORA, ENOCH, et al. Dynamic security metrics for measuring the effectiveness of moving target defense techniques[J]. Computers & Security, 2018, 79: 33-52.

[7] 赵正．基于软件定义网络的移动目标防御关键技术研究[D]．河南：解放军信息工程大学，2017．

[8] IMAD A, ADLEN K, YASSINE H, et al. Improving traffic forecasting for 5G core network scalability: a machine learning approach[J]. IEEE Network: The Magazine of Computer Communications, 2018, 32(6): 42-49.

[9] 谢丽霞，丁颖．链路洪泛攻击的 SDN 移动目标防御机制[J]．清华大学学报(自然科学版)，2019，59(01)：36-43．

[10]张连成，魏强，唐秀存，等．基于路径与端址跳变的 SDN 网络主动防御技术[J]．计算机研究与发展，2017，54(12)：2761-2771．

[11]唐秀存，许强，史大伟，等．移动目标防御(MTD)关键技术研究[J]．微型机与应用，2016，35(07)：1-5+15．

[12]JAFARIAN J H, AL-SHAER E, DUAN Q. An effective address mutation approach for disrupting reconnaissance attacks[J]. IEEE Transactions on Information Forensics & Security, 2015, 10(12): 2562-2577.

[13]胡毅勋．基于 Openflow 的主动防御关键技术研究[D]．北京：北京邮电大学，2017．

[14]景湘评．基于 Web 服务的移动目标防御技术研究与实现[D]．北京：北京邮电大学，2018．

[15]高亮，郑荣锋．基于 Snort 和 OpenWrt 的网络入侵检测与防御系统[J]．现代

计算机，2019(36)：17-22+75．

[16]JUNG J, PAXSON V, BERGER A W, et al. Fast portscan detection using sequential hypothesis testing[C]// IEEE Symposium on Security & Privacy. Berkeley: IEEE, 2004: 211-225.

[17]JUNGSIK H, MINSOO K. Effective detecting method of Nmap idle scan [J]. Journal of Advanced Information Technology & Convergence, 2019, 9 (1): 1-10.

[18]马超，程力，孔玲玲．云环境下 SDN 的流量异常检测性能分析[J]．计算机与现代化，2015(10)：92-97+102．

[19]PRABHAKAR K, SUBHASRI D, KRISHNASHREE A. VARMAN: Multi-p lane security framework for software defined networks[J]. Computer Comm unications, 2019, 148: 215-239.

[20]KSHIRA S S, SANJAYA K P, SAMPA S, et al. Toward secure software-de fined networks against distributed denial of service attack[J]. The Journal o f Supercomputing, 2019, 75(8): 4829-4874.

[21]王晓瑞，庄雷，胡颖，等．SDN 环境下基于 BP 神经网络的 DDoS 攻击检测方法[J]．计算机应用研究，2018，35(03)：911-915．

[22]李鹤飞，黄新力，郑正奇．基于软件定义网络的 DDoS 攻击检测方法及其应用[J]．计算机工程，2016，42(02)：118-123．

[23]李传煌，吴艳，钱正哲，等．SDN 于深度学习混合模型的 DDoS 攻击检测与防御[J]．通信学报，2018，39(07)：176-187．

[24]BEAKAL G A, ÖZNUR Ö. A survey of energy efficiency in SDN: Softwa re-based methods and optimization models[J]. Journal of Network and Com puter Applications, 2019, 137: 127-143.

[25]JAFARIAN J H, AL-SHAER E, DUAN Q. Openflow random host mutatio n: transparent moving target defense using software defined networking[C]. In Proceedings of the first workshop on Hot topics in software defined net works (HotSDN '12). New York: Association for Computing Machinery, 20 12: 127–132.

[26]周春莲．基于时间戳的数据同步技术实现研究[D]．江西：南昌大学，2009．

[27]BADISHI G, HERZBERG A, KEIDAR I. Keeping Denial-of-Service Attack ers in the Dark[J]. IEEE Transactions on Dependable & Secure Computing, 2007, 4(3): 191-204.

[28]LEE H C J, THING V L L. Port hopping for resilient networks[C]// IEEE Vehicular Technology Conference Washington: IEEE, 2004: 3291-3295.

[29]王红运．基于 SDN 控制器的故障恢复与负载均衡策略研究[D]．安徽：安徽大学，2020．

[30]DEBROY S, CALYAM P, NGUYEN M, et al. Frequency-minimal moving

target defense using software-defined networking[C]// International Conference on Computing. Kauai: IEEE, 2016: 1-6.

[31]JANTILA S, CHAIPAH K. A security analysis of a hybrid mechanism to defend DDoS attacks in SDN[J]. Procedia Computer Science, 2016, 86: 437-440.

[32]王玉栋. 基于朴素贝叶斯的入侵检测关键技术研究[D]. 北京：北京工业大学，2017.

[33]王远帆，施勇，薛质．基于决策树的端口扫描恶意流量检测研究[J]．通信技术，2020，53(08)：2002-2005.

[34]张康宁，廖光忠．基于改善 Bagging-SVM 集成多样性的网络入侵检测方法[J]．东北师大学报(自然科学版)，2020，52(04)：53-59.

[35]刘月峰，王成，张亚斌，等. 用于网络入侵检测的多尺度卷积 CNN 模型[J]. 计算机工程与应用，2019，55(03)：90-95+153.

[36]高忠石，苏旸，柳玉东．基于 PCA-LSTM 的入侵检测研究[J]．计算机科学，2019，46(S2)：473-476+492.

[37]丁珊．基于深度学习的入侵检测关键技术研究[D]．北京：北京交通大学，2018.

[38]吴成智．基于机器学习的网络入侵检测技术研究与实现[D]．广东：广东工业大学，2019.

[39]雷程，马多贺，张红旗，等．基于网络攻击面自适应转换的移动目标防御技术[J]．计算机学报，2018，41(05)：1109-1131.

[40]FINK G A, HAACK J N, MCKINNON A D, et al. Defense on the Move: Ant-Based Cyber Defense[J]. IEEE Security & Privacy, 2014, 12(2): 36-43.

[41]MIYAZAKI R, KAWAMOTO J, MATSUMOTO S, et al. Host independent and distributed detection system of the network attack by using OpenFlow [C]// 2017 International Conference on Information Networking (ICOIN). Da Nang: IEEE, 2017: 236-241.

[42]WANG L, WU D. Moving Target Defense Against Network Reconnaissance with Software Defined Networking[C]// International Conference on Information Security. Springer: Cham, 2016: 203-217.

[43]杨垠坦．基于卷积神经网络的 SDN 入侵检测技术研究[D]．陕西：西安电子科技大学，2019.

[44]夏克文，李昌彪，沈钧毅. 前向神经网络隐含层节点数的一种优化算法[J]. 计算机科学，2005(10)：143-145.

[45]王丹．随机梯度下降算法研究[D]．陕西：西安建筑科技大学，2020.

[46]胡黄水，赵思远，刘清雪，等．基于动量因子优化学习率的 BP 神经网络 PID 参数整定算法[J]．吉林大学学报(理学版)，2020，58(06)：1415-1420.

[47]GERUM R C, ERPENBECK A, KRAUSS P, et al. Sparsity through evolutionary pruning prevents neuronal networks from overfitting[J]. Neural networks : the official journal of the International Neural Network Society, 2020, 128: 305-312.

[48]LI J L, YOU Y X, CHEN K. Applications of An Eddy-Viscosity Eliminator Based on Sigmoid Functions in Reynolds-Averaged Navier-Stokes Simulations of Sloshing Flow[J]. China Ocean Engineering, 2020, 34(04): 463-474.

[49]ZOU D F, CAO Y, ZOU D R, et al. Gradient descent optimizes over-parameterized deep ReLU networks[J]. Machine Learning, 2020, 109(7587): 467-492.

[50]郭敏钢，宫鹤．基于 Tensorflow 对卷积神经网络的优化研究[J]．计算机工程与应用，2020，56(01)：158-164．

[51]曹惠珍．基于改进卷积神经网络的图像分类研究[D]．广西：广西师范大学，2017．

[52]WANG S H, MUHAMMAD K, HONG J, et al. Alcoholism identification via convolutional neural network based on parametric ReLU, dropout, and batch normalization[J]. Neural Computing and Applications, 2020, 32(3): 665-680.

[53]CHAITY B, TATHAGATA M, EDUARDO P J. Feature Representations Using the Reflected Rectified Linear Unit(RReLU) Activation[J]. Big Data Mining and Analytics, 2020, 3(02): 102-120.

[54]CLEVERT D A, UNTERTHINER T, HOCHREITER S. Fast and accurate deep network learning by Exponential Linear Units(ELUs)[J]. Computer Science, 2015, 5(2): 713-716.

[55]曹铮，傅文卿，黄蕊．互联网流量成分及运营策略分析[J]．中国新通信，2006(03)：76-78．

[56]张鑫杰，任午令．基于 Fisher-PCA 和深度学习的入侵检测方法研究[J]．数据采集与处理，2020，35(05)：956-964．

在学期间取得的科研成果和科研情况说明

取得的科研成果：

[1]《SDN 环境下基于 PCA-DNN 的扫描攻击检测模型研究》-天津理工大学学报，已录用；第一作者

参与的科研项目：

[1]本人参与了国家自然科学基金委 青年基金项目《软件定义安全架构下的异常流量检测与控制技术研究》，项目批准号 61802281，项目时间：2019-01-01 至 2021-12-31

致　　谢

在为期两年半的研究生生活中，我得到许多周边老师和同学的帮助和鼓励，在此，对其进行感谢。

首先，我要感谢的是我的导师韩俐副教授。在学校的科研工作中，她认真、求实、负责的学术精神感染了我。耐心给出科研道路上的指引，提供力所能及的帮助，使我受益良多。在日常的校园生活中，她更像是一位大朋友，关注我们的身体健康，解忧心中的烦恼与迷茫，使我度过了充实难忘的学习生涯。在未来的生活中，我将谨记恩师的教诲，认真、细致的完成日常工作。同时，我还要感谢409实验室的林胜、石凯老师。在实验室组会中，他们细心地指出我们学术研究上的不足，给出对应的解决方案和建议参考。在此对其表示衷心的感谢。

其次，我还要感谢同师门的杨志师兄、张昭俊、张博锋、叶和元、英明、宋吉祥几位同学。在平时科研工作中，我们认真研讨相关问题，耐心解答彼此疑惑，互帮互助，共同进步，使我学术研究稳步向前。我还要感谢汪卓越、麻括朗、周浩、付国凯、李闯、陶峰、吕垛斌等实验室同学，提供生活上的帮助和科研上的提点。与他们宿舍夜晚的畅谈，将是我读研生涯难以忘怀的时光。

我还要感谢我的父母，感谢他们对我的抚养和关心。他们无私奉献，供我成长、学习。在未来的日子里，我将用自己所学知识和双手，去努力回馈二老给予的一切。

最后，我还要感谢所有认识和帮助过我的人，感谢天津理工大学为我们创造舒适的学习环境和适宜的生活场所，祝愿理工大学在未来越办越好，吸引更多的学子前来求知生活。