学　　号：23620174010
论文密级：公　开
中图分类号：TP393.08
学科分类号：520.1060
学校代码：91037

战略支援部队信息工程大学

硕士学位论文

基于用户实体行为的异常检测技术研究

论文作者：孙剑文
指导教师：武东英 副教授
申请学位：工程硕士专业学位
学科名称：计算机技术
研究方向：入侵预警与主动防御
论文提交日期：2020 年 4 月 30 日
论文答辩日期：2020 年 7 月 1 日

战略支援部队信息工程大学

2020 年 4 月

A Dissertation Submitted to

PLA Strategic Support Force Information Engineering University

for the Degree of Master of Engineering

Research of Anomaly Detection Technology Based on User and Entity Behavior

Candidate: Sun Jianwen

Supervisor: Prof. Wu Dongying

Apr. 2020

摘　要

随着网络技术的不断发展和广泛应用，互联网络所承载的数据流量不断攀升，其中的信息量越来越大。依赖于网络而生的应用正呈多元化趋势发展，而网络攻击也随之越来越多样化和复杂化。入侵检测技术伴随着入侵行为的不断产生而不断完善，针对各类场景制定相应的防御方案，但是一直被飞速发展的相关技术挑战。目前新型的攻击已经开始使用深度学习技术对抗现有的入侵检测设备，并且更多的使用加密和隧道技术，提高了隐蔽性，从而避开现有的安全防御。这使得未知的攻击行为更难以被检测，造成设备系统数据被篡改或是重要信息泄露等一些难以挽回的损失。

为了应对不断涌来的新威胁，本文把 UEBA 的思想应用到网络流量的异常检测领域。以面向行为的流量分析为基础，深入分析正常用户行为和攻击行为产生的流量特征中与时间相关的特征，从而更全面的刻画正常用户的网络流量行为基线，以发现新产生流量中行为的变化，从而检测攻击和其他异常行为。本文提出运用生成对抗网络模型进行基线学习，该方法可以避开该领域在训练时攻击样本量不足的问题，使用充足的正常用户行为数据训练异常检测模型。本文的训练样本易于采集，模型易于部署，可以针对任意用户、系统或者有 IP 地址的设备进行部署，该框架检测迅速，并且对多种攻击的识别具有一定效果。本文完成的主要工作如下：

首先，分析了网络流量粒度的划分，不同划分方式的优势和劣势，确定研究使用流级别的划分方式。针对传统检测漏报率高，难以检测未知攻击的问题，分析基于行为异常检测的研究现状，又针对检测模型难以处理高维数据的问题，分析基于机器学习方法的异常检测研究现状，再针对特征选择片面和模型通常面向特定场景进行检测的问题，分析了 UEBA 的框架思想，最终锁定了擅长分析复杂数据类型的基于深度学习的方法。

其次，分析了网络流量的表示方式，确定研究使用 TCP/IP 底层元数据表示上层交互行为。深入分析了正常用户的会话流量特征，提出了用户行为差异的基本表示。以攻击流程划分，剖析了各类常见攻击行为的流量特征，总结了识别攻击行为的显著特征。分析了现有用于入侵检测的网络数据集，指明现有数据集的不足，从而确定通过采集长时间的真实用户行为数据进行模型训练。结合上述分析，提出了一组基于双向流和子流与时间相关的元数据特征，特征为 54 维，能够较全面刻画并且稳定表示用户的行为。依据特征组合，对采集的正常用户流量和公共数据集中的攻击流量进行提取，形成数据集。最后针对不平衡的训练数据，分析处理此类数据方法的优缺点，最后确定了仅使用一类数据的建模算法。

然后，提出了改进的基于双向生成对抗网络的基线建模方法。该模型通过仅学习正常样本，在过程中不断自动优化和学习，为不同的用户建立基线模型。阐述了模型的原理、训练部分的改进、参数选择、各部分的网络结构和训练的过程，最后将数据中的连续型特

征和离散型特征预处理为符合模型学习的形式，通过实验对比进行了基线模型的有效性测试，并且确定了适合此类用户的建模时长。

最后，设计实现了基于用户实体行为的异常行为检测系统，并且对性能进行了测试。将系统应用于真实采集的实验室用户流量和公共数据集中的 7 类攻击流量，实验结果表明，该系统能够通过构建的不同用户的行为基线，产生不同的检测效果。检测速度很快，对部分攻击产生了极好的检测效果，验证了方法的可行性。

关键词：网络异常检测，行为分析，流量行为特征参数提取，深度学习

Abstract

With the continuous development and wide application of network technology, the amount of data carried by the Internet is rising, especially the amount of information in it. At the same time, applications that rely on the Internet are developing in a diversified trend, and cyberattacks are becoming more and more diverse and novel. Intrusion detection technology is constantly improving with the continuous generation of intrusion behaviors, by formulating corresponding defense solutions for various scenarios. But it has been challenged by the rapid development of related technologies. At present, new types of cyberattacks have begun to use deep learning techniques to combat existing intrusion detection systems, and more use of encryption and tunneling technology improves the concealment, so that the attack can easily avoid the existing security defense. These make unknown attacks more difficult to detect, resulting in some irreparable losses such as device data tampering or important information leakage.

In order to deal with new threats that are constantly coming in, this dissertation brings the idea of UEBA into the field of network traffic anomaly detection. Based on behavior-oriented traffic analysis, in-depth analysis of time-related features in normal user behavior and attack traffic, so as to more fully depict the baseline of normal users' network traffic behavior, to discover changes in newly generated traffic behavior, and thus to detect attacks and other abnormal behaviors. This dissertation proposes to use the generative adversarial network model for baseline learning. This method can avoid the problem of insufficient sample size during training in this field, and use sufficient normal user behavior data to train anomaly detection models. The training samples in this dissertation are easier to collect, and the model is easier to deploy. It can be deployed for any user, system, or device with an IP address. The framework detects quickly and has a certain effect on multiple attacks. The main research work and innovations of this dissertation are as follows.

1. It analyzes the network traffic granularity division, the advantages and disadvantages of different division methods, and determines the flow-level division method used in the study. Aiming at the problem of high false negative rate in traditional detection, which is difficult to detect unknown attacks, the research status based on behavior abnormality detection is analyzed. In addition, for the problem of detection model is difficult to handle high-dimensional data, the current status of anomaly detection based on machine learning methods is analyzed. Then, for the problem of selecting one-sided features and models usually face a specific scenario for detection, it analyzed UEBA's framework idea and finally locked in a deep learning-based method that is good at analyzing complex data types.

2. It analyzes the representation of network traffic, and decided to use TCP / IP underlying metadata to represent upper-layer interactions. Then, in-depth analysis of the flow features of normal user's session, and put forward a basic representation of differences in user behavior. At the same time, according to the attack process, it analyzes the traffic features of various common attack behaviors and summarizes the salient features of identifying attack behaviors. In addition, the existing intrusion detection data sets are analyzed, and their deficiencies are pointed out, so as to determine the model training by collecting long-term real user behavior data. Combining the above analysis, a set of time-based metadata features based on bidirectional streams and substreams are proposed. The feature is 54-dimensional, which can more fully characterize and stably represent user behavior. According to the combination of features, the collected normal user traffic and attack traffic in the public data set are extracted to form a data set. The final step before modeling is to analyze the advantages and disadvantages of methods for processing such data for unbalanced training data. Finally, a modeling algorithm that uses only one-type-of-data is determined.

3. An improved baseline modeling method based on bidirectional Generative Adversarial Network (GAN) is proposed. An improved baseline modeling method based on bidirectional generated confrontation network is proposed. By learning only normal samples, the model continuously optimizes and learns automatically in the process to establish a baseline model for different users. In the study, the principle of the model, the improvement of the training part, the parameter selection, the network structure of each part and the training process are elaborated. Finally, the continuous and discrete features in the data are preprocessed into a form that conforms to the model learning. After the model is trained, the effectiveness of the baseline model is tested through experimental comparison, and the modeling time suitable for such users is obtained.

4. This dissertation designs and implements an abnormal behavior detection system based on user and entity behavior and tests the performance. Furthermore, the system is applied to real-time collected laboratory user traffic and seven types of attack traffic in public data sets. The experimental results show that the system can produce different detection effects through the construction of different user behavior baselines. The detection speed is very fast, which has produced an excellent detection effect for some attacks, and verified the feasibility of the method.

Key Words： Network Anomaly Detection, Behavior Analysis, Behavior Characteristic Extracting, Deep Learning

目　录

图 录

表 录

第一章 绪 论

1.1 研究背景与意义

今时今日已经广泛应用于终端之间的 TCP/IP[1]，已经经过了 46 年的发展，它是 1974 年由美国国国防部的 Robert Eilliot Kahh 与斯坦福大学的 Vinton Cerf 共同提出的通信基础协议，已经无缝联接了全世界数以亿计的互联网终端设备，服务于当代社会的方方面面。截至 2020 年 3 月，全球互联网使用人口占比为 58.7%，近 20 年的增长率为 1167%，亚洲则是增长最快的地域，从图 1 中可以看到全世界各州的增长率[2]。第 44 次《中国互联网络发展状况统计报告》发布，截至 2019 年 8 月底，中国使用互联网的人数已经达到 8.54 亿，其中在线教育、在线政务、市场电商、视频娱乐已经成为互联网使用的主要用途[3]。在中国，互联网已经改变了传统的生活状态，成为必不可少的生活方式。

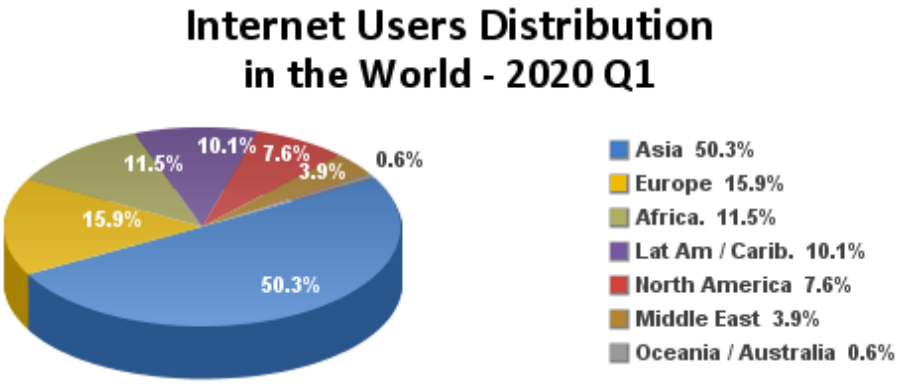

图 1 世界互联网用户分布

然而，在丰富、方便、快捷的背后，也存在着隐患。网络安全问题随着网络应用的纷繁化和业务系统的复杂化日益突出，向政治、经济、文化、社会等诸多方面提出了挑战。于 2019 年 8 月发布的《2019 年上半年我国互联网网络安全态势》[4]显示，仅上半年，我国境内感染恶意程序的计算机约有 240 万台；收录安全漏洞 5859 个；大流量分布式拒绝服务攻击(Distributed Denial of Service, DDoS)平均每月 4300 起左右，峰值超过了 10Gpbs；监测到针对境内的钓鱼网站约 4.6 万个；0day 漏洞数量增加等等。其中，超过六成的 DDoS 攻击由僵尸网络发起。报告明确指出，个人信息和重要数据泄露风险严峻，以及频频报出的漏洞和攻击事件严重威胁着我国重要设施设备安全。纵观全球近几年发生的安全事件，体现了一种趋势：安全事件正在从传统的攻击和防御，从对业务系统或者主机设备的攻击，转向了针对信息泄露和数据篡改的威胁。比如爆发在 2017 年 5 月的“WannaCry”蠕虫式勒索病毒，该病毒将受害者主机内的大量文件数据加密，令其付费解锁[5]。该病毒影响了全球金融、医疗、能源等诸多行业，同时造成约 30 万名用户遭到入侵，损失高达 80 亿美

元。在大数据时代，数据的价值逐渐被大家认可和重视。《中华人民共和国网络安全法》第22条表明，任何人不得入侵他人网络、不得窃取网络数据[6]。

恶意的网络入侵行为严重影响着网络信息安全，小到用户的一台普通主机，大到国家的重要基础设施。保护网络信息安全和重点实体，不但人人有责，利用好有效的预警和防御措施尤为重要。随着的网络设施和计算机技术的发展，网络攻击技术变得越来越多样化，各种流量通信加密手段也应运而生。传统的防御方式可以抵御已知的攻击，但是对于感知网络态势，预判异常行为，检测未知攻击存在着很大挑战。基于网络行为的异常检测模型可以发现正常与异常通信间的行为偏差，成为了异常检测技术的一个重要研究方向。毋庸置疑，实现高效的网络异常检测，是实现网络安全的重要手段之一，是防范恶意攻击大规模传播的前提，是对现有安全检测设备和防火墙的补充，是预测恶意行为的重要工具，是保障重要基础设施、网络信息系统、用户计算机系统安全的重要课题。

1.2 问题与挑战

传统的安全技术一直依赖于检测规则，检测引擎具有许多内置规则，专家经验和人为设置的阈值。随着网络流量加密数据的不断增加，这种基于现有规则的方法会导致较高的误报率和较低的准确性，甚至无法检测到一些未知的威胁行为。基于行为的异常检测技术被广泛应用，这些研究中的算法通常不再基于规则，而是从流量行为的特征入手。虽然可以弥补一些传统方法的不足，但仍存在一些不能回避的问题。概括为以下两点：

1. 流量行为刻画片面

现有的对正常行为刻画从而检测异常的方法，其刻画能力具有局限性。流量行为的特征选取主要面临以下问题：（1）对用户的交互特征锁定于OSI模型的应用层元数据，忽略了特征的时间关系，而且当协议加密时难以提取；（2）只利用包头信息，对应用协议没有通用性，并且在加密协议下更是难以提取；（3）仅选取流的前一部分数据包，对行为的刻画较片面;（4）涉及的应用越来越多，数据量巨大，提取的特征需要考虑到计算的复杂性。

2. 对未知攻击检测能力不足

在现有入侵检测研究中，通常针对某一个数据集设计合适的模型，能实现较好的入侵检测效果。检测大多局限于特定的攻击场景，因此，只要有足够的样本量，现有的基于机器学习等方法的入侵检测系统(Intrusion Detection System, IDS)就可以检测新的攻击。但在实际中，网络空间是瞬息万变的，每时每刻都可能产生未知攻击，训练样本难以采集，更难以及时发布整理为数据集，从而使检测模型对新型和未知威胁的检测能力不佳。

针对上述问题，本文提出的策略是：以正常行为为基准，设计一个不针对应用场景的通用检测模型。选取能够全面描述流量行为的特征，刻画一个较为全面的用户行为基线，从而发现新产生流量与基线的偏差，感知异常行为，进行主动防御。

1.3 论文内容与主要贡献

本文把 UEBA 的思想应用到实际环境的流量异常检测中，以面向行为的流量分析为基础，深入分析正常用户行为和攻击行为产生的流量特征中与时间相关的特征，从而更全面的刻画正常用户的网络流量行为基线，以发现新产生流量行为中的变化，检测攻击和其他异常行为。本文提出运用生成对抗网络模型进行基线学习，使用充足的正常用户行为数据训练异常检测模型，该方法可以避开该领域在训练时攻击样本量不足的问题。本文的训练样本易于采集，模型易于部署，可以针对任意用户、系统或者有 IP 地址的设备进行部署，该框架检测迅速，并且对多种攻击识别具有一定效果。本文的研究内容主要分为以下四个方面：

1. 研究适合高维复杂数据进行异常检测的方法

分析网络流量粒度的划分，不同划分方式的优势和劣势。针对传统检测漏报率高，难以检测未知攻击的问题，分析基于行为异常检测的研究现状。又针对检测模型难以处理高维数据的问题，分析基于机器学习和数据挖掘方法的异常检测研究现状。再针对特征选择片面和模型通常面向特定场景进行检测的问题，分析 UEBA 的框架思想，最后确定了研究使用的方法和工具。

2. 研究流量行为刻画和特征参数提取

分析网络流量的表示方式，深入分析正常用户的会话流量特征，提出用户行为差异的基本表示。以攻击流程划分，剖析各类常见攻击行为的流量特征，总结识别攻击行为的显著特征。分析现有用于入侵检测的网络数据集，统计数据集的属性。基于双向流和子流与时间相关的元数据特征，创建一组能够全面刻画用户正常行为的特征，采集实验室 3 个用户的日常行为流量，按提取的特征组提取出训练数据集。同时，对公共数据集中的攻击流量进行提取，形成数据集。

3. 研究复杂类型数据的建模方法

针对不平衡的训练数据，分析处理这一类数据方法的优缺点，提出基于双向生成对抗网络的基线建模方法。详细分析模型结构原理，训练改进方式和参数选择，包括将复杂的流量数据输入模型的方法。然后，研究了生成对抗网络模型在训练中的问题，进行适当的参数调整。最后，通过实验对比，验证该基线模型的可行性，并且确定适合用户的建模时长。

4. 构建异常检测模型并进行性能验证

设计实现基于用户实体行为的异常行为检测系统，并且对性能进行测试。将系统应用于真实采集的实验室用户流量，采用 7 天正常用户的行为流量作为每个用户的模型训练数据，在公开的数据集中提取 7 类攻击流量样本，在每类攻击中随机抽取 36 个样本，并分别与该用户其余的正常流量进行混合作为测试数据，从而验证方法的有效性。

在上述研究内容的基础上，本文的贡献如下：

1. 纵向分析了从传统异常检测方法到基于重构误差的方法之间的优势与不足

针对传统检测漏报率高，难以检测未知攻击的问题，本文分析了基于行为异常检测的研究现状。又针对检测模型难以处理高维数据的问题，分析了基于机器学习等方法在异常检测研究中的现状。再针对流量特征选择片面和模型通常面向特定的应用场景进行检测的问题，分析了 UEBA 的框架思想，提出了更适合于高维复杂数据进行异常检测的方法和工具。

2. 提出了一组基于用户实体行为画像的流量特征，并且生成了供分析实验的数据集

分析了网络流量的表示方式，深入分析了正常用户的会话流量特征，提出了用户行为差异的基本表示。以攻击的流程划分，剖析了各类常见攻击中较有代表性的攻击行为的流量特征，总结识别攻击行为的显著特征。分析了现有用于入侵检测的网络数据集，统计数据集的属性。基于双向流和子流与时间相关的元数据特征，创建了一组 54 维的能够全面刻画用户正常行为的特征。采集了实验室中为期 3 个月的真实日常行为流量，按提取的特征组提取出数据集，该数据集为 57 维，在 54 维的特征组基础上增加了流标识、时间戳和用户标签属性。

3. 提出了一种复杂类型数据的一类数据基线建模方法

针对不平衡的训练数据，分析了处理此类数据方法的优点和缺点，提出基于双向生成对抗网络的一类数据基线建模方法。本文详细分析了模型结构与原理，充分考虑复杂的流量数据输入模型的方法，分别处理连续型数据和离散型数据，使之适合模型学习。又针对复杂数据类型在生成对抗网络模型的训练中可能遇到的梯度消失等问题，改进模型训练方式。最后，通过实验进行了基线模型的有效性测试，并且确定了适合一般正常用户的 7 天建模时长。

4. 实现了异常检测系统构建

针对以往研究对未知攻击的检测性能不佳的问题，本文在研究了正常用户会话行为特征，各类常见攻击的行为特征以及深度模型算法的基础上，提取了一组刻画正常用户行为的元数据特征，通过对充足的正常行为数据训练建立基线模型，从而检测攻击行为。通过实验验证了该方法的可行性，测试表明系统检测速度快，善于检测未知攻击，并且对多种攻击具有较好的检测效果。

1.4 论文组织结构

本文共分为六章，论文的组织结构如下：

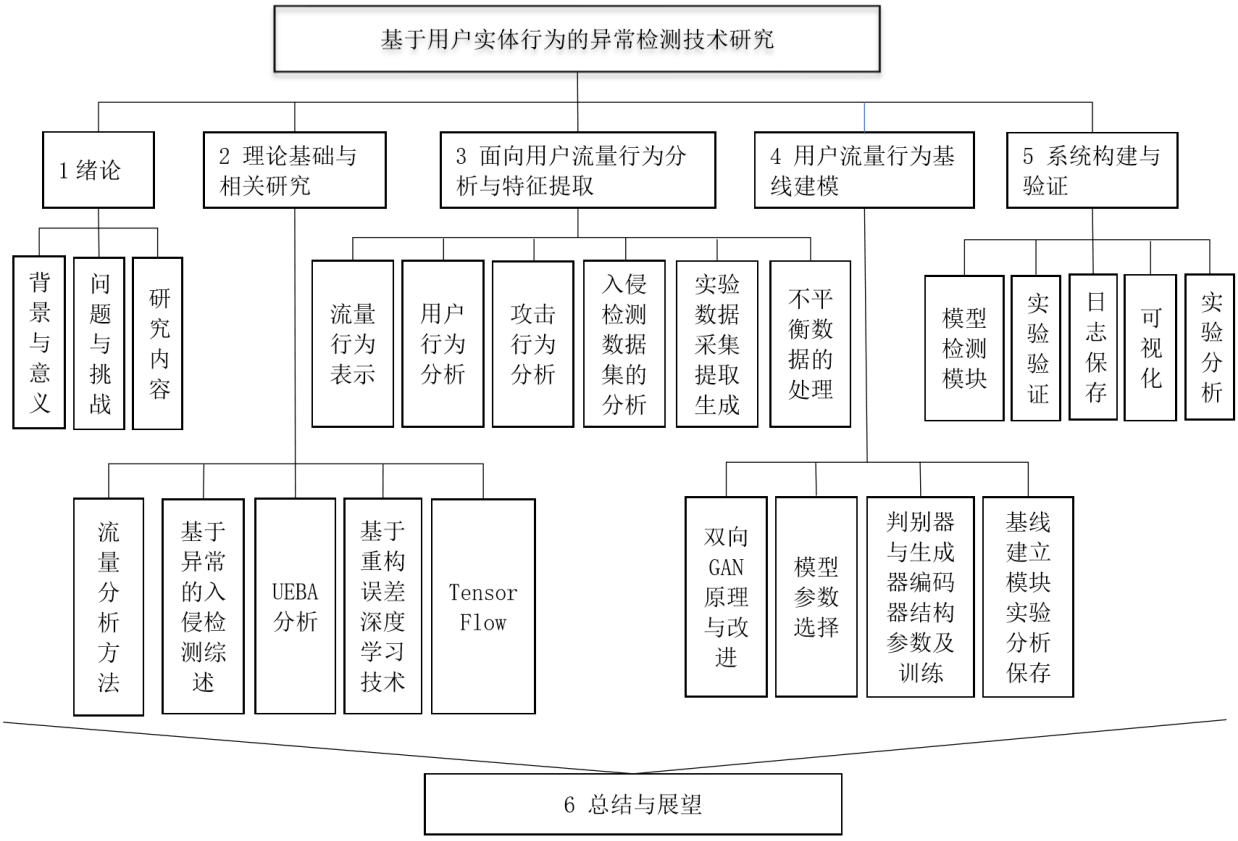

图 2 论文的组织结构

第一章，指出当前网络安全的严峻形势，点明网络异常检测面临的问题和挑战，揭示基于流量行为的异常检测是当下和未来的研究趋势；提出研究思路和方向，并概述本文的研究内容以及主要贡献。

第二章，介绍网络流量的分析方法，综述基于异常的入侵检测方法，分析其中的优势与不足。同时，简要介绍 UEBA 的思想、框架、作用和发展趋势，重点归纳基于重构误差的深度学习异常检测算法，以及上述理论基础与本文研究方法的关联。最后，介绍了本文使用的深度学习工具。

第三章，本章从网络流量行为的表示入手，首先，深入分析用户日常行为的流量特点和攻击的不同阶段中较有代表性的攻击行为的流量特点。其次，对现有的入侵检测数据集做简要概述，归纳分析数据集的基本属性并作出关心属性的对比。然后，主要阐述用户行为数据的生成方式和特征选取的标准，并生成本文的实验数据。最后，介绍不平衡数据的处理方法，明确本文应用的方法。

第四章，提出改进的基于双向生成对抗网络的基线建模方法。模型通过学习用户流量的正常样本，在训练过程中自动更新参数进行优化，使模型快速收敛，从而为不同的用户

建立行为基线模型。本章阐述了模型的原理、参数选择、模型训练的过程，并且对基线模型进行有效性测试和分析。

第五章，构建异常检测系统的原型系统，简述系统的基本架构，详述系统中异常检测模块的工作原理，通过多个用户的流量数据和多种攻击流量数据进行实验验证、结果分析和结果对比。

第六章，综述全文工作，点明研究的意义、创新所在和良好前景，展望进一步深入研究的方向。

第二章 理论基础与相关研究

本章依据研究思路，分节介绍本文涉及的理论知识和相关技术。首先是网络流量的分析方法，其次是现有基于异常的入侵检测方法的优点和缺点，然后对工业界基于用户实体行为分析的框架和产品做简要介绍和趋势分析，最后是本文采用的基于重构的深度学习异常检测技术优缺点分析和本文所使用的深度学习工具和库。

2.1 网络流量分析方法

本节从流量的层级划分和基本分析方法进行阐述。

2.1.1 流量粒度

网络流量在数据表示时可以记为一个 n 维的向量，但在实际情况中是较为复杂的。OSI 参考模型对网络自下而上划分为七层结构，二层以上结构均能在网络流量数据中得以体现[8]。网络流量通常在捕获后以 pcap 格式存储，最小传输单位是数据包，通常数据包由包头和载荷构成。

网络流量的划分有四种层面：字节级别(Bit-level)、包级别(Packet-level)、网络层流级别(Flow-level)、IP 层流级别(Stream-level)[9]。如图 3 所示。将原始流量以不同方式划分，得到的数据在表示和数量级上也会有很大区别。

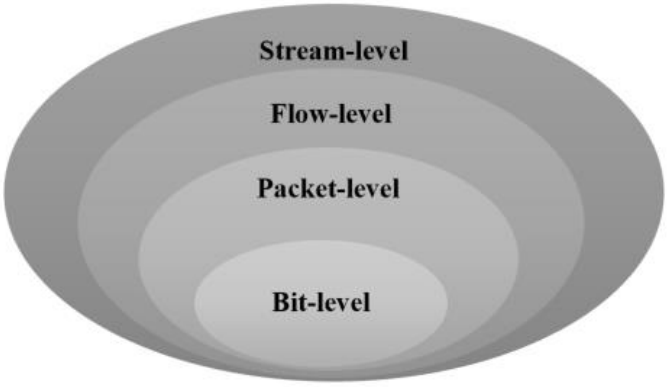

图 3 网络流量划分的层次关系

Packet-level 的网络流元数据提取决定于传输所使用的协议：TCP 的包头包含如序列号、确认号、标志或校验和值等；UDP 的包头只包含源端口、目的端口、长度和校验和[10]。比如，一个超文本传输协议(Hyper Text Transfer Protocol, HTTP)数据包，它依次包含 14 字节的介质访问控制(Medium Access Control, MAC)层包头，20 字节的网际协议(Internet Protocol, IP)层包头，20 字节的 TCP 层包头，而后才是 HTTP 包头和载荷。仅基于包头提取特征的方法有两大缺点：其一，载荷中包含很多重要信息，若将其忽略，则该方法难以适应其他协议，例如提取文件传输协议(File Transfer Protocol, FTP)包头特征的方法不能用

于提取 HTTP 包头；其二，以数据包为检测对象时，包间存在的时间关联将被忽略。实际上一个最基本的 TCP 连接都要经过“三次握手”，从而产生多个在时间上互联的数据包，它们是一个整体。

Stream-level 通常以三元组表示<源 IP、目的 IP、协议类型>；Flow-level 通常以五元组表示<源 IP、目的 IP、源端口、目的端口、协议类型>[11]。数据流是多个相关数据包的组合，基于流的数据格式相比于 Bit-level 和 Packet-level 更简洁和紧密，是时间窗口内共享属性的包聚合起来的流，它的元数据主要是网络连接的信息，可以在一定程度上反应出流的行为特点。数据流的表示可以是单向的也可以是双向的，会话即为双向流组成，其标准五元组表示中的源和目的相关信息可以互换。

2.1.2 流量分析方法

网络流量分析最初的目的是识别网络相关的攻击，这项工作始于 20 世纪的 90 年代[12][13]，研究人员用这种技术来研究问题的不同方面，例如流量优先级划分、网络优化、网络行为分析和内部滥用等。有两种基本方法可用于分析网络流量：

1. 基于 Packet-level 的分析方法

这类方法是与已知威胁的预定义签名进行逐位比较。如果数据包（特别是有效载荷部分）与威胁签名相似性较高，就被检测为攻击。现有的分析工具包括 TCPDump、Wireshark、Xplico、Microsoft Network Monitor 等[14]，可以帮助分析人员解析数据包的信息，从而更好地理解攻击是如何形成的。但是，这些方法在应用中有许多限制，其一就是分析大量数据时非常耗时，另一个最为重要的问题是在网络通信中越来越多地使用加密（如 SSL/ TLS）协议，这阻止了分析人员对有效载荷的分析[15]。虽然，研究人员又提出了不同的方法来处理深度包检测的限制，以加快识别恶意攻击的过程，但他们的解决方案仍然有限[16][17]。

2. 基于 Flow-level 的分析方法

这类方法通常分析当前与历史网络流的差异来识别异常。属于特定流的全部数据包均有一组相同属性[18]。与 Packet-level 的方法相比，此类方法的优点是需要分析的数据量大大减少。而且，检测异常可以不依赖于数据包的内容[18]。如今，加密协议的应用在增加，这使 Flow-level 的检测方法日渐突出。Flow-level 的特征通常使用各种字段，例如，时间戳、源 IP、目的 IP、端口号、有效载荷的长度以及协议的类型等。现有成熟的监测分析工具包括 NetFlow[20]，SFlow[21]等。目前，基于 Flow-level 的网络分析仍是领域内常用的方法，并且不断有研究者提出并设计新的工具[22]。

基于 Flow-level 的划分方法能够表达时间窗口内整条流的全局特征，还能够避免错序、丢包、重传的干扰，本文后续对流级别的特征开展研究和分析。

2.2 基于异常的入侵检测方法

入侵检测从上世纪 80 年代被提出至今[23]，经历了由理论框架到深度学习技术的迭代更新，伴随着入侵行为的不断产生而不断完善。与其他领域一样，流量领域也开始依靠数据挖掘、机器学习和深度学习技术来实现网络的异常检测。IDS 一直是安全人员研究的重点，是当代安全体系不可或缺的重要组成部分[24]。网络入侵检测系统(Network Intrusion Detection System, NIDS)用来检测网络中的入侵行为，传统上的 NIDS 一般被认为是一个流量分类系统，即把网络流量分为正常和恶意两类。通常使用基于签名或者基于异常的检测方法。

基于签名的入侵检测系统(Signature-based Intrusion Detection System, SIDS)是一种基于先验知识的检测，又称为误用检测，即检测与现有特征知识库中匹配的操作，这种方法试图以定义一组异常行为规则为基础。优点是能够减少误报，对已知的攻击具有较高的检测精确度[25]。但是在动态变化的计算环境中，此方法需要定期更新知识库以获取预期的正常行为，是一项耗时的工程，因为采集有关正常行为的所有信息非常困难。而且，随着 0day 漏洞增加，SIDS 难以匹配到签名，对未知攻击进行检测。而基于异常的 IDS 克服了 SIDS 的局限性。如表 1 所示。

表 1 入侵检测技术优缺点对比

	基于签名的 IDS	**基于异常的 IDS**
优点	误报率低	漏报率低
	检测快	可检测未知攻击
缺点	漏报率高	误报率高
	难检测未知攻击	行为建模困难
	特征库需要实时更新	

综上，基于本文的目的是提高检测未知攻击的能力，因此本文采用基于异常的入侵检测方法。

基于异常的 IDS 假设恶意行为不同于正常用户的行为，从而被划分为异常行为。在该方法中创建计算机系统行为的正常模型，与模型之间的任何重大偏差都被视为异常，定义为入侵。开发该系统包括两个阶段：训练和测试。在训练阶段，使用正常流量学习正常行为的模型，在测试阶段，使用新的数据集，为了提升系统的泛化能力，检测未见过的入侵。本节将基于异常的 IDS 的训练方法分三类进行分析：基于统计的方法，基于数据挖掘和机器学习的方法，基于深度学习的方法。

2.2.1 基于统计的方法

基于统计的方法是流量异常检测中应用最早的方法，它是根据正常行为的参数建立模型，检测低概率事件并将其标记为潜在的入侵。该方法的理念与本文最为相似，但不同的是它实质上是考虑了统计指标，例如数据包数的中位数、均值、众数和标准差。通过监控每个数据包，从而提取流的指纹，用于识别当前行为与正常行为的差异。该方法通常使用以下模型，如图 4 所示。

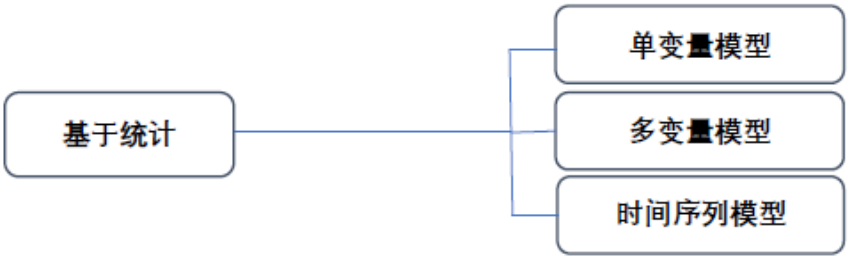

图 4 统计方法分类

1. 单变量模型

在单变量模型中，数据只有一个变量。仅针对计算机系统中的一个度量创建正常统计轮廓，从而在每个单独的指标中寻找异常[26]。

2. 多变量模型

多变量模型是基于两个或多个度量之间的关系，以便分析它们之间的关联。如果实验数据表明对相关度量的组合分析能够比分别进行分析获得更好的分类，则该模型将有价值。例如，通过建立长期正常活动状况来识别入侵的多元控制方法[26]。

3. 时间序列模型

时间序列是在一定时间窗口内的一系列行为。如果新的行为在当时发生的概率极低，则判定是异常。例如，使用时间序列来处理入侵检测警报聚合[27]，通过检查时间序列数据中的突变来检测网络异常[28]等。

基于统计的方法的优缺点分析如表 2 所示。本文采用的方法在理念上相似于第二点与第三点的综合，但是由于刻画用户的网络流量行为相当于处理高维的特征分布，难以选择出合适的统计量用于基于统计的方法，因此该方法不适用于本研究。

表 2 基于统计方法的优缺点

优点	缺点
统计学习性； 易适应各种网络环境； 可检测未知攻击，触发阈值后报警； 计算复杂度低	难以处理好高维数据

2.2.2 基于数据挖掘和机器学习的方法

基于数据挖掘和机器学习的方法可以弥补统计模型的缺憾[29]。机器学习模型由一组规则或复杂的传递函数组成，可以对海量且高维的数据提取出有意义的数据模式和潜在的关系，这些潜在信息无法被直观感受到，需要根据潜在信息来建立对应的模型，之后再根据模型来识别和预测行为。这类方法已经广泛应用于入侵检测领域，本小节的介绍将如下图结构所示。

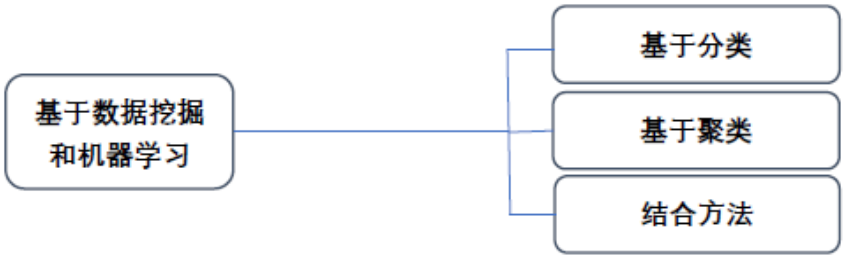

图 5 数据挖掘和机器学习法分类

1. 基于分类的方法

该类方法尝试将所有正常流量和恶意流量分开，首先是从一系列有标签的数据中学习训练分类器，而后通过训练好的分类器模型为新样本分类。合适的分类方法不仅能处理训练数据，还应能够准确的识别未知的新数据，其难点在于如何最小化误报和漏报的数量。

根据训练数据的标记不同，基于分类的方法又可以分为单类别检测和多类别检测。

单类别的方法是为正常数据训练出边界，界外的数据都被判定为异常。多类别的方法则是为多个正常类别数据训练出各自的分类，若无法判定待检测数据属于哪个分类，则划分为异常。

在多种分类的方法中，为正常与异常类别作区分的方法有例如朴素贝叶斯(Naive Bayesian, NB)、支持向量机(Support Vector Machine, SVM)、人工神经网络(Artificial Neural Networks, ANNs)等。

NB 基于贝叶斯原理，在算法上给予了相应简化，目标值的属性之间相互条件独立。NB 依赖于攻击行为和正常行为中具有不同的发生概率的特征，从而进行分类。NB 网络可以被用于构建正常行为模型，从而判别网络是否运行于健康的状态[30]。将其与时间序列结合，可以有效降低误报率[31]。然而，对于 KDD99 这类属性较为复杂的数据，难以发挥作用[32]。

SVM 是由分裂超平面定义的判别分类器，该方法选用核函数将训练数据映射到高维空间进行线性分类。SVM 以其泛化能力而闻名，当属性数量大而数据点少时，SVM 方法效果较好。在 IDS 数据集中，许多特征对于数据点的正确分类影响较小或者多余，因此，在使用 SVM 训练时应考虑特征选择问题。比如，在使用 KDD 99 数据集进行入侵检测研究时，从 41 个特征中选择了 19 个关键特征，提高了分类准确率[33]。

ANNs 方法早在 2000 年就以 96%的检测率成功检测出了 DARPA98 数据集中的攻击[42]，这类方法最常用的技术之一是反向传播(Back-propagation, BP)算法，它通过更新权重来评估网络误差的梯度，适合检测频繁的攻击。而对于不频繁的攻击，由于其训练数据较少，ANNs 难以正确学习这些攻击的属性。在入侵检测领域，若无法检测到低频的攻击，可能会造成巨大的危害。同时，由于 ANNs 方法会经常遇到局部最小值，因此稳定性较低。ANNs 因其全局近似非线性映射，使学习过程变得非常耗时。但是，这种方法仍有它的优势，它能够凭借一个或者多个隐层，构成高度非线性的模型，可以捕捉输入数据与分类标签之间的复杂关系。

2. 基于聚类的方法

聚类的方法优于分类方法的一点是用于训练过程的数据不需要标签，是一种无监督学习。将每条流量数据映射到二维空间中，进行聚类，若实例的距离远离群体，则判定这条数据为异常。在聚类方法中将流量区分为正常和异常的方法，例如离群点检测(Outlier Detection, OD)算法。

OD 旨在查找数据中与预期行为不符的模式。可以基于距离，如 K-means 方法，对检测 DoS 攻击很有效[34]。也可以基于密度，检测局部的离群点，适合在数据分布不均匀的情况下使用。它为每个数据分配一个局部离群值因子，仅考虑每个对象的受限邻域[35]。

该类方法优势与劣势并存，如表 3 所示。当簇本身规模较大或者密度差异较大时，由于聚类算法的多数目标函数是距离和，此时算法表现会极差。潜在特征维度的选择和大量训练数据的获取本身就为基于数据挖掘的方法带来了挑战[36]，又因为网络流量的数据特征维度高，且特征间存在关联，因此聚类的方法仅适合判断单一的网络异常，即针对特定的场景进行模型设计，不适用于本研究。

表 3 基于聚类方法的优缺点

优点	缺点
时间和空间复杂度低； 可以检测未知异常流量； 利用参数间的关系，适合处理海量数据； 适合处理高维数据	聚类中心点的选定对检测效果影响大； 适合仅针对特定场景判断单一的网络异常

3. 结合的方法

分类和聚类的方法都有各自的优势，二者相结合既能提高准确率又能提高效率。使用 K-means 与 ID3 方法结合对计算机地址解析协议(Address Resolution Protocol, ARP)中的正常和异常流量分类，准确率达到了 98%[37]。有研究者在检测 NSL-KDD 数据集时使用结合的方法，对单隐层的前馈神经网络进行训练，输出模糊隶属度向量，再使用模糊向量对未

标记的样本进行分类，该方法大大提高了检测的准确度[43]。还有研究者将 ANNs 与 SVM 结合的方法应用在 KDD CUP 99 数据集中，对 U2R 和 U2L 类型的攻击检测性能很高[44]。

2.2.3 基于深度学习的方法

深度学习方法是机器学习的分支，可以理解为深度的机器学习。大多数传统的机器学习方法的结构是由一个输入层和一个输出层组成的浅层网络，中间没有隐藏层或者是仅有单一隐藏层的神经网络(Neural Networks, NNs)，也称为 ANNs。而三层以上（包含输入和输出层)才可以称为有“深度”[38]。深度学习架构的出现突破了机器学习的可扩展性和数据中泛化的局限性[39]。深度学习原本是模拟动物的生物特征，建立与其类似的深度神经网络(Deep Neural Networks, DNNs)来理解数据。可以无需人工干预，进行自动特征提取，可以发现非结构化数据中的潜在结构，例如图像、文本、语音等。

基于深度学习的异常检测方法可以分为三类，如图 6 所示。

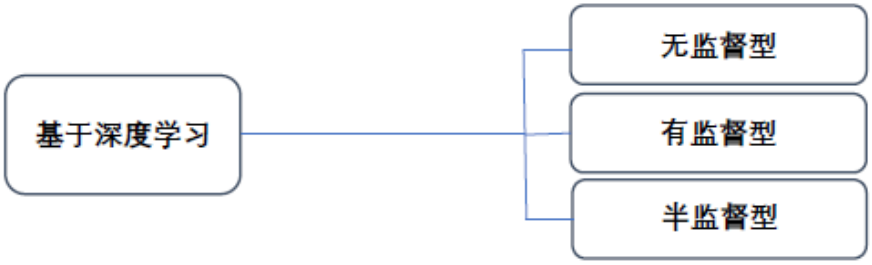

图 6 深度学习方法分类

1. 有监督型方法

这类方法依旧使用正常和异常标签，将任务视为二分类或多分类问题，存在传统方法关于标签和不平衡数据的弊端，尽管提升了性能，但是不如另外两类方法应用广泛。

2. 无监督型方法

这类方法面向数据标签难以获取的情况，基于 NNs 的方法可以从大量训练数据中提取特征，在此过程中可以发现有意义的数据分布，从而识别或预测行为[40]。

3. 半监督型方法

这类方法在正常数据比异常数据的标签更容易获取的情况下应用更广泛。在 DNNs 中，其节点在前一层输出的基础上自动学习识别特征，即反复重构输入样本。随着深度增加，节点能识别出越来越复杂的特征，从而处理大规模高维度的数据。该网络最后的输出层是一个 Softmax 或逻辑分类器，输出所属标签或者某一结果，可以理解为广义上的预测。

后两类方法的目的都是学习精确的数据分布，从而生成相似但不相同的新数据点。常见模型分别是编码器(Autoencoder, AE)和生成对抗网络(Generative Adversarial Network, GAN)。将在 2.4 节中介绍。

基于深度学习的方法仍然存在一些挑战[41]，如表 4 所示。

表 4 基于深度学习方法优缺点

优点	缺点
擅长分析大量且复杂的数据类型，更适合入侵检测领域； 可不需要专家先验知识； 属（离线）训练模型，虽复杂度高训练时长，但检测速度快，不影响实时性	如何确定模型参数的最优数量是挑战； 如何提高计算的实用性是挑战； 开发高维度深度学习算法是挑战； 非平稳数据的增量学习是挑战

2.2.4 基于异常的入侵检测系统的不足

综上所述，基于异常的入侵检测的方法不断进化，迭代发展，解决了诸多实际问题。虽然它具有检测速度快和能够检测未知攻击的优点，但是其检测需要建立正常的流量模型，依赖阈值设置，这可能导致误报率高，因为被划分为异常的行为有可能是新的正常行为，而非入侵。

2.3 用户实体行为分析

本节从框架入手，对用户实体行为分析(User and Entity Behavior Analytics, UEBA)的思想进行阐述，而后概述其应用前景。

2.3.1 UEBA 框架

基于用户行为分析(User Behavior Analytics, UBA)进行异常检测的思想实质就是基于主机的入侵检测，这项研究早在上世纪 90 年代已经开展，研究是根据处理系统的审计日志的统计结果来判断是否出现了异常[45]。而 UEBA 的前身是就是 UBA，不过 UBA 最初应用于购物网站，通过收集用户搜索的关键字为用户需求画像，预测其购买欲并进行推送。而后，这项技术很快应用于安全领域，在业务安全监控系统中，维护用户的个人行为数据库，统计指标旨在检测相关异常，而后将用户会话中触发指标的分数进行融合，计算出最终分数获得重要度排名[46]。

2015 年，Gartner 引入了 UEBA，用于发现企业基础结构中的安全威胁[47]。UEBA 框架通过采集多种数据（文件系统日志、业务处理日志、点击流数据、配置文件、数据库审计日志、WEB 访问日志、Windows 事件日志等)，不仅分析用户的行为，还分析设备、应用等实体对象的行为，使用异常检测和其他机器学习的方法来检测行为偏差。如图 7 所示。该架构需要处理来自多个源的数据，它们格式不同，结构化有差异，而且速率较快。UEBA 需要从如此大量数据中提取有价值的信息进行学习和判断，其检测到的异常不一定是威胁，只是偏离了预期的行为，需要引起关注。

图 7 UEBA 一般架构

在实际应用中，UEBA 的软件层一般使用 HADOOP、STORM、SPARK 这类能够处理大规模集群数据的引擎，而底层设施需要一定的成本[48]。

UEBA 的优势是能通过对正常行为持续性的记录和分析，更容易发现异常的活动。用户的身份能够轻易伪造，但是模仿其行为却相对困难。UEBA 从另一角度发现问题，从一维检测到多维分析，从单点检测到长期分析，从基于规则到关联分析，从行为建模到异常分析，以发现威胁。

2.3.2 UEBA 现状与趋势

企业中使用安全信息事件管理(SIEM)和 UEBA 相结合，检测大量用户和设备。国内外已有多家企业自主研发了基于 UEBA 的产品，Exabeam 通过一系列专家规则和服务日志，预见事件时间表[50]；GURUCUL 侧重检验超出签名、规则和模式能力的威胁，预测风险评分并立即查找和制止威胁[49]；LogRhythm 通过多维行为分析快速显示事件和其优先级[51]。以上产品的共同之处是都用于检测内部威胁、数据泄露、身份盗用这一类正常用户做出的异常行为。在国内，启明星辰[52]用于发现员工泄露数据和账号等异常行为；观安[53]同样用于数据泄露分析。综上所述，从功能上看，UEBA 产品与传统的 SIEM 类似，它们的差异在于方法不同，如表 5 所示。

表 5 UEBA 与 SIEM 区别

UEBA	SIEM
构建员工一段时间内的配置文件，形成基线，并在发现用户行为模式有任何突变时发出警报。	主要依靠分析在防火墙，操作系统和其他系统日志中捕获的事件，该过程需要捕捉大量数据。

在 2018 年 Gartner 发布的应用安全报告中，单个 UEBA 的产品越来越少，它们的发展趋势都是作为一种核心引擎并购到其他产品中以发挥作用[54]。UEBA 在工业界属于一种数

据驱动的安全分析产品，仅是一定程度上使用了机器学习方法，并且多数产品依靠于统计特征信息。它虽然能够弥补传统检测中单点安全分析的不足，但是实际落地对硬件设施要求较高。

在金融领域，有研究基于 UEBA 思想针对“鼠标轨迹分析”和“击键动力学”提出一种“偏好-行为”算法来检测欺诈行为[55]。而在流量领域却难以展开研究，UEBA 的核心能力是接入海量异构和非结构化数据，弥补单点安全分析的不足，但是仅使用纯流量来全面描述正常用户的行为是一个挑战。

本文将应用 UEBA 的全面数据驱动思想，选取合适的学习模型，为正常用户刻画全面的纯流量行为基线，从而发现异常。

2.4 基于重构误差的深度学习异常检测技术

重构是从经过变换的数据中还原原始数据的过程，而重构误差就是重构数据与原始数据之间的差异，如图 8 所示，为误差重构的过程。

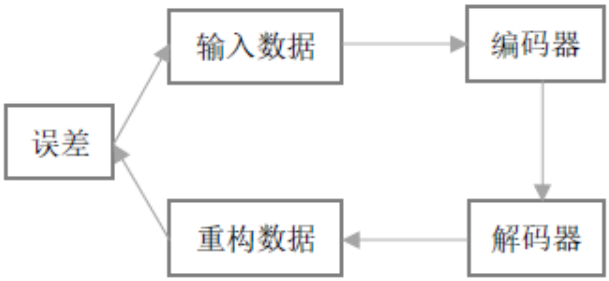

图 8 重构误差产生过程

重构误差分为：基于数值的重构和基于分布的重构。

基于数值的重构，例如均方误差(Mean squared error, MSE)。

MSE 的数学定义为：

$$MSE = \|x - z\| \tag{1}$$

其中，x是原始数据，z是重构数据。MSE 评估的是x与z在数值上的变化程度，均方误差值越小，则重构效果越好。

基于分布的重构，例如交叉熵，用来评估当前训练得到的概率分布与原始分布的差异。

其离散函数形式为：

$$H(p,q) = -\sum_{x} p(x)\log(q(x)) \tag{2}$$

其中，$p(x)$是原始数据的概率分布，$q(x)$是重构数据的概率分布。

交叉熵损失函数，即信息熵，是在结果出来前对可能产生的信息量的期望。相对熵，又叫 KL 散度(Kullback-Leibler divergence)，是概率分布间差异的非对称性度量[56]。在信息论角度理解，等价于两个概率分布的信息熵差值[39]。

KL 散度的数学定义为：

$$D_{KL}(p\|q) = E_{x\sim p}(logp(x) - logq(x)) \quad (3)$$

其中，$logp(x) - logq(x)$是关于$p(x)$的期望。

基于数值的重构和基于分布的重构方法分别对应以下两种模型。

2.4.1 自编码器

AE 是基于数值的重构模型，是深度学习中重要的异常算法的模型之一[57]。AE 网络采用无监督的学习方式，训练的基本原理是最小化重构误差，其目的是学习和表示数据。AE 最主要的部分是编码器(Encoder, E)和解码器(Decoder)。其结构是对称的，如图 9 所示。

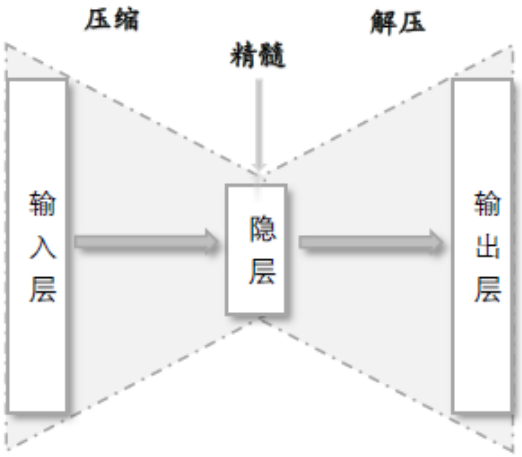

图 9 自编码器结构

在结构中，通常隐层尺寸小于输入层和输出层的尺寸，是原始数据的精髓。可将前半部分看为 E，把输入样本映射到低维空间，负责对数据降维，后半部分看为解码器，负责对数据重构（还原）。

E 的数学表达式可表示为：

$$h = \sigma(w_{xh}x + b_{xh}) \quad (4)$$

Decoder 的数学表达式可表示为：

$$z = \sigma(w_{hx}x + b_{hx}) \quad (5)$$

其中，w是权重，b是其偏置，σ是激活函数，可以使用 Sigmoid、ReLU、LeakyReLU、Tanh 等。E 中原始输入是x，隐层的输出为h；Decoder 中使h尽量还原为原始输入，得到重构变量z。此时，AE 的目标是最小化 MSE。

AE 的特性是使输出数据与输入数据尽可能相同，从而可以构建异常检测模型，将 RE 作为异常分数的判别标准，当分数超出阈值，则判定为异常。通常单个 AE 的学习效果不佳，需要堆叠多个 AE 构成深层的学习模型，以获得数据的最佳表示。

2.4.2 生成对抗网络

GAN 是由 2014 年首次被提出[58]，是一种基于分布的重构模型，也是一种混合的 DNNs 模型。引申到模型中，它的主要思想是有两个竞争的 NNs 模型，一个是生成器(Generator, G)模型，将隐层的随机噪声作为输入并生成样本，目的是学习原始数据的数据分布；另一个是判别器(Discriminator, D)模型，从 G 和原始数据中接收样本，并且必须能够区分样本的源。将 D 比作一个二分类器（0-1 分类器），来判断输入的数据是真是假，也就是判断输出大于 0.5，还是小于 0.5。D 的目的就是估计样本来源于原始数据的概率。而后，损失值传回 D，使 D 进行调整。由此，G 学习越来越多的原始样本，而 D 学习如何更好地区分原始样本和重构样本，从而 G 尽可能的骗过 D，但 D 要尽可能的判别真伪。模型架构如图 10 所示。

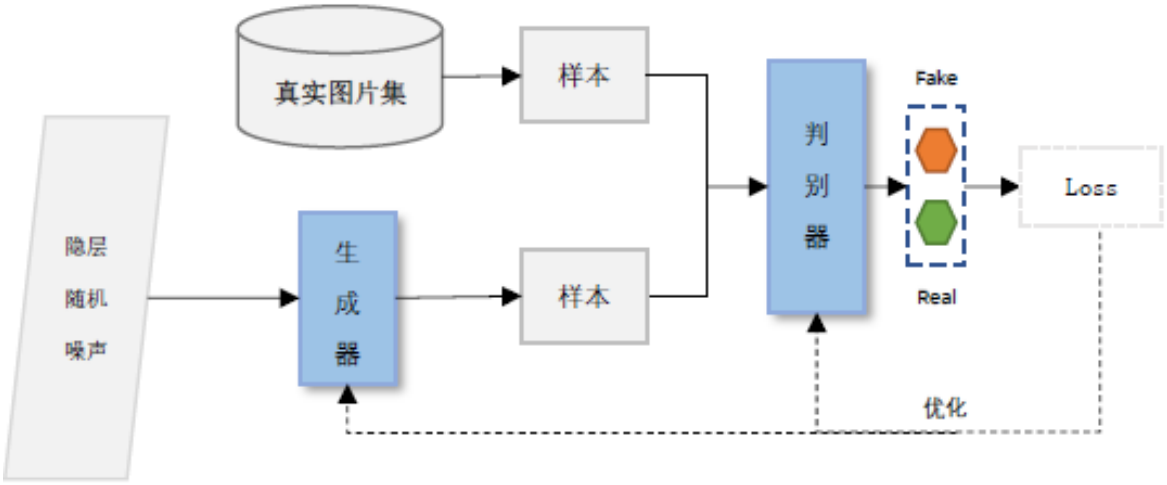

图 10 GAN 模型架构

GAN 的目标函数为：

$$\min_{G}\max_{D} V(D,G) = E_{x\sim p_x}[logD(x)] + E_{z\sim p_z}[\log(1-D(G(z)))] \quad (6)$$

其中，p_x是原始数据的分布，p_z是随机隐变量的分布，G 将p_z映射到数据空间得到p_g，p_g是重构数据的分布，GAN 希望p_g与p_x相似。当p_g=p_x时为最优解，即达到了纳什均衡，D 的准确率达到了 50%。G 的训练过程是最大化 D 判别错误的概率的过程，即希望 D 把重构数据当作原始数据。

GAN 能够对高维复杂模型进行建模，能够充分学习出输入数据的精确分布，这表明它们可能实现有效的异常检测。目前，基于 GAN 的异常检测框架已经在医学[59]、影像[60]等多个领域[61]得到研究。利用 GAN 进行异常检测的核心是异常分数的定义，其原型的残差损失的数学表达式如下：

$$L_R(z) = \Sigma\|x - G(z)\| \quad (7)$$

$L_R(z)$就是原始数据与重构数据的分布差异。在理想情况下，GAN 的重构误差为 0，即重构数据与原始数据相同。

判别损失：是由 D 输出测试样本与原始样本相同与否的概率。

异常分数[59]的数学表达式如下：

$$A(x) = (1-\lambda) * R(x) + \lambda * D(x) \quad (8)$$

其中，$R(x)$是残差损失分数，$D(x)$是判别损失分数，λ是二者的权重。

2.4.3 优势与不足

重构误差是损失函数的一种定义方法，基于分布重构的生成模型是目前异常检测领域的主要研究方法。该类模型的优点是能够学习到数据的固有特征信息，并且根据学习到的内在共性特征检测出异常点。在实际应用中，异常通常无法预知，但是各个领域里的异常点定义却相似。生成模型提供了这样的思路，挖掘出数据本身的特性，当数据样本足够丰富，模型就能快速收敛，达到学习效果。

生成模型的不足也相对明显：其一，该模型对噪声非常敏感，随着训练样本中噪声的增加，模型的性能将逐渐变差，因此，生成模型通常适合在正常数据集下进行训练。其二，高维数据的固有特征复杂，生成模型足够复杂才能够学到好的特征。其三，由于生成模型的基础是 NNs，超参数调参的过程很耗时，其复杂性远大于传统的异常检测模型。其四，通过加入高斯噪声以缓解模型训练的过拟合现象，该方式会导致模型的性能下降。

本文将应用基于分布的重构模型，使模型学习到正常用户流量特征中的共性特征分布，从而检测出攻击行为中流量特征的异常分布，改进模型结构，使其复杂度能够适应高维数据的学习，并且在模型训练的效率上加以改进。

2.5 深度学习工具

在深度学习开源框架不断涌现的今天，TensorFlow 仍然最受关注。如表 6 所示，在 GitHub 上开源框架的关注度和贡献人数等属性的统计（统计信息截止 2020 年 4 月 15 日）。

表 6 GitHub 上部分开源框架统计

框架	所属	支持语言	关注人数	贡献人数
TensorFlow	Google	Python/Java/C++/…	143000	2461
PyTorch[63]	Facebook	Python	37700	1361
Caffe[64]	BVLC	Python/C++	30100	266
MXNET[65]	DMLC	Python/C++/Java/…	18600	785
CNTK[66]	Microsoft	C++	16700	198

TensorFlow 搭建网络方便，并且有极强的编译处理能力，同时可以搭载 GPU 进行加速训练，适合数据量较大的入侵检测领域[67]。

2.5.1 TensorFlow 框架

TensorFlow[62]是 Google 开源的第二代分布式机器学习框架，计算可以表示为一个有向图，当中的每一个运算操作都作为一个节点，它可以有任意多个输入和输出，连接节点的为边，计算在边中流动的数据，即为 TensorFlow 的含义。主要技术特性如表 7 所示。

表 7 TensorFlow 框架概况

模型	数据流模型
语言	Python、C、C++、Java、Go、Rust 等
部署	一次编写各处运行
计算资源	CPU、GPU、TPU
实现方式	单机式、分布式、集群式
平台支持	Google 云平台、Hadoop 分布式文件系统
数学表达	数学计算图表达、自动微分
优化器	Pipeline、数据并行、模型并行、通信优化、异步核优化、共同子图消除

TensorFlow 的框架包含三个模块：client、master 和 worker。

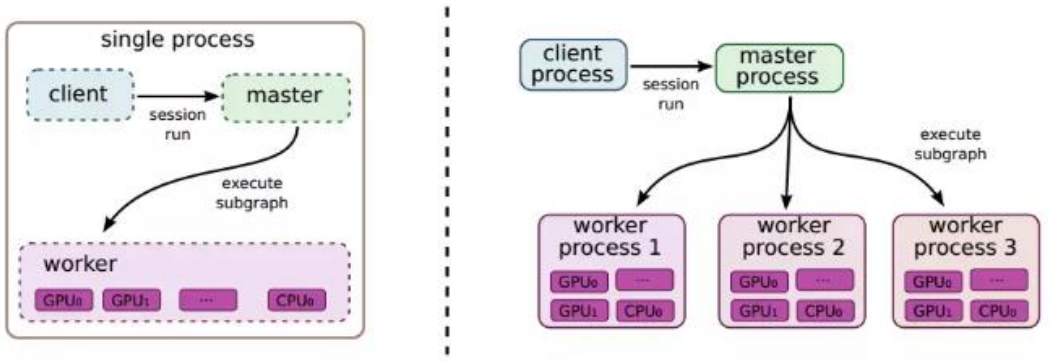

图 11 单机模式和分布式模式

如图 11 所示。左侧为单机模式，右侧为分布式模式。Client 提供给用户编程接口，master 接受用户 client 请求，构造数据流图，分配给 worker 执行。单机模式中，只有一个 worker 负责所有计算任务，而在分布式模式中，有多个 worker 可以同时执行不同的任务。在分布式模式中，master 预先估计数据流图中各节点的数据量，以确保各 worker 的内存相差不大，估计计算时长，以使运算总时间最短。

TensorFlow 的节点执行顺序依赖其计数机制，对于每个节点待输入数据的个数，当个数降至 0，该节点启动。这一执行方式完美的体现了 TensorFlow 以数据驱动的理念。

2.5.2 可视化

TensorBoard 是 TensorFlow 中自带的可视化工具，可以用最直观的方式发现我们所构建的 NNs 是什么样子，有助于发现编程中出现的问题。还能够以折线图、直方图等形式监控程序运行中各标量、张量迭代等变化的趋势，例如 SCALARS，通过使用函数 tf.summary.scalar()记录损失值(loss)随着训练迭代变化的趋势；HISTOGRAMS，通过使用函数 tf.summary.histogram()记录数据的变化趋势；PROJECTOR，可以将模型不同层的数据变化投影到 3D 空间，可以查看数据的高维向量形式。常用工具如表 8 所示。

表 8 TensorBoard 常用工具

工具	生成函数	功能
GRAPHS	默认保存	显示计算图
SCALARS	tf.summary.scalar()	显示张量随迭代变化趋势
DISTRIBUTIONS/ HISTOGRAMS	tf.summary.histogram()	显示直方图
IMAGES	tf.summary.images()	显示训练的图片
AUDIO	tf.summary.audio()	显示训练的音频
TEXT	tf.summary.text()	显示训练的文本
PROJECTOR	tf.train.Supervisor()/ tf.train.saver()	高维空间数据可视化

2.5.3 深度学习库

Keras[78]是一个基于 Python 语言的高级神经网络 API，它能够以 TensorFlow 作为后端运行，并成模块化。表 9 对常用的库进行了汇总。

表 9 Keras 常用库

重要模块	具体内容
优化器	RMSprop、Adagrad、Adam、Adamax、Nadam、Sgd 等
目标函数	mean_squared_error、squared_hinge、categorical_crossentropy、binary_crossentropy 等
激活函数	Sigmoid、Tanh、ReLU、LeakyReLU、PReLU、Softmax 等
参数初始化	uniform、lecun_uniform、normal、orthogonal、zero、glorot_normal、he_normal 等
层	Dense、Convolutional、Dropout、Pooling、Recurrent、Embedding、Model 等

2.6 本章小结

本章分层次介绍了本文涉及的理论知识和相关技术的综述，其中涵盖了网络流量的分析方法，现有基于异常的 IDS 方法，以及本文关注的基于重构的深度学习异常检测技术和分类，并对现有方法的优点与缺点进行分析。同时，简要介绍 UEBA 的架构及其发展趋势，以及上述理论基础与本文研究方法的关联。最后，介绍了本文实验所使用的深度学习工具和库。

第三章 面向用户流量行为分析与特征提取

本章从网络流量行为入手，首先，深入分析用户日常行为流量的特点和不同阶段常见攻击行为的流量特点。其次，对现有入侵检测数据集做简要概述，归纳分析数据集的属性。然后，主要阐述用户行为数据的生成方式和特征选取的标准，并生成本文的实验数据。最后，介绍不平衡数据的处理方法。

3.1 网络流量表示

本节阐述流量行为的表示和流量连接的管理机制。

3.1.1 流量行为表示

用户的日常行为包括（但不限于）浏览网页、娱乐（如观看在线视频）、通讯（如使用QQ）、金融（如网上购物）和办公应用（如使用文档）。事实上，许多传统的桌面应用程序，如 Office，也是基于互联网的服务[104]。试想一下，用户在早上 8:00 登录即时消息应用程序，与朋友聊天，然后 10 分钟后向朋友发送文件。由于使用 SSL 协议或 TLS 协议的应用与日俱增，若对上一过程中用户所使用的每个应用程序的行为进行采集，将非常耗时。

表 10 OSI 参考模型和 TCP/IP 各层职责和常用协议

OSI 结构	职责	TCP/IP 结构	常用协议	职能
应用层	针对应用程序提供服务	应用层	HTTP、SMTP、SNMP、FTP、TFTP、Telnet、DHCP、SSL、SLP、POP3、IMAP、DNS	负责主机间的数据传输
表示层	数据格式转化、数据加密			
会话层	建立、管理、维护会话			
传输层	建立、管理、维护端到端的连接	传输层	TCP、UDP	负责网络互连
网络层	IP 地址选择、路由选择	网络层	IP、ICMP、RIP、IGMP	
数据链路层	互联设备之间传送和识别数据帧	链路层	Ethernet、Wi-Fi	负责介质传输
物理层	连接器、网线等			

由表 10 所示 OSI 结构和 TCP/IP 结构，流量的特征似乎都体现在应用层。例如，浏览网页应用 HTTP 和 DNS 协议，通讯使用 SSL 协议，传送邮件使用 SMTP 协议，系统间的文件交互使用了 FTP 协议等。应用层作为 TCP/IP 的第四层，直接为应用进程服务，那么用户的活动被封装在 TCP/IP 的第三层，即传输层的各种标识中，传输层的信号又封装在 TCP/IP 的第二层，那么通过相应的元数据标识，一样可以识别用户的行为，而且不受数据加密的限制。图 12 中为数据传输的封装过程。

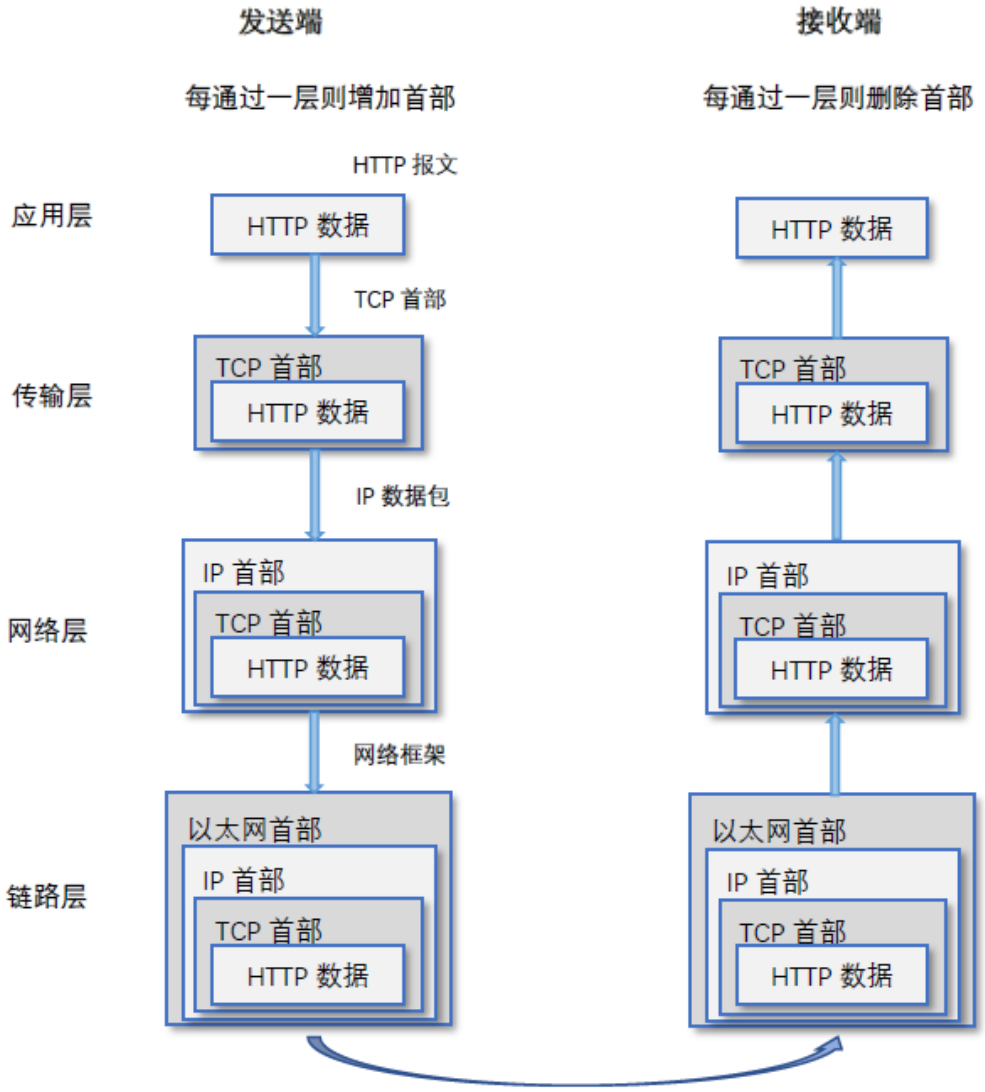

图 12 数据传输封装过程

因此，本文的研究从原始网络流量元数据中提取特征，用以刻画行为。

本章分析流量的行为表示，主要关注两点：其一，需要找到一些可以测量的变量来定义流量中的行为特征；其二，所选用的变量对于描述行为有意义。

3.1.2 流量的连接管理机制

1. TCP 连接

对于正常的 TCP 连接，经过“三次握手”连接，通过“四次挥手”释放连接，也是判断一条完整连接的重要依据。图 13 中是一条完整的流。

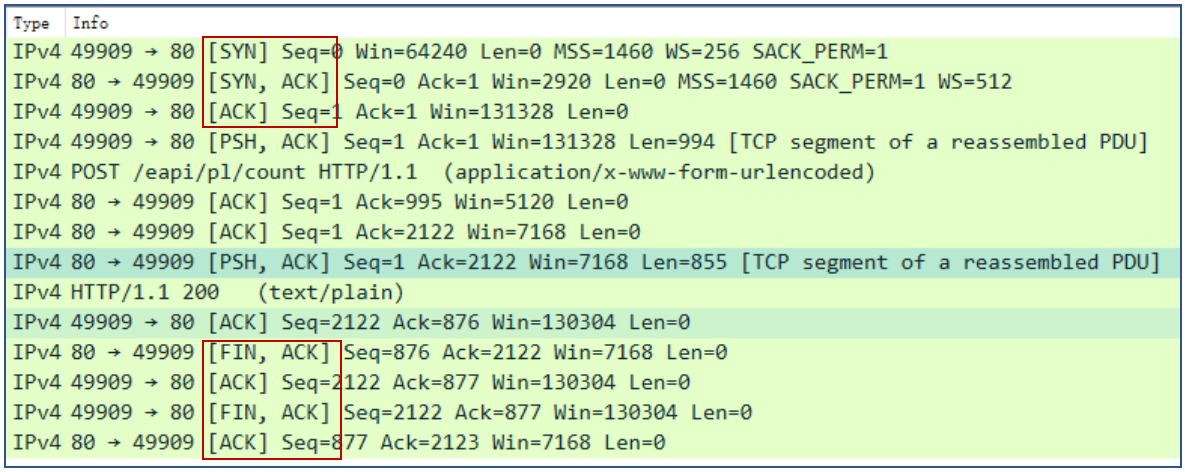

图 13 完整 TCP 流

（1）连接的建立过程

1st: Client 向 Server 发送 SYN 包(seq=x)；

2st: Server 确认 SYN 包(ack=x+1)，向 Client 发送 SYN 包(seq=y);

3st: Client 确认 ACK 包(ack=y+1)，此时连接建立成功。

Client 与 Server 传送 TCP 报文过程中，双方的确认号(ack)与序号(seq)都是在彼此值的基础上加 1 表示的，这样可以确保一旦一方有报文丢失，就无法继续进行此次 TCP 连接。

（2）连接的释放过程

连接的释放必须是一方主动释放，另一方被动释放，释放不一定都是 Client 发起的，图 13 中就是由 Server 发起的释放，但是释放的过程是一样的。

1st:Server 向 Client 发送 FIN 包(seq=x)；

2st: Client 确认 FIN 包(ack=x+1)，向 Server 发送 ACK 包(seq=y)；

3st: Client 向 Server 发送 FIN(seq=z);

4st:Server 确认 FIN 包(ack=z+1)，发送 ACK 包(seq=h)，此时连接释放成功。

与连接的过程相似，ack 和 seq 都是在彼此基础上建立的，一旦丢包，则释放连接不成功。

2. UDP 的连接

与 TCP 相比，UDP 没有连接特性，没有差错控制，即使在传输数据时，通信双方也不建立逻辑连接，也没有会话时长的定义。它是一种面向报文的传输方式。在一次会话的第一个报文起至最后一个报文止，即 UDP 的一次会话。

3.2 用户行为分析

用户的流量行为数据与人类的行为类似，每个用户的身份和习惯之不同，使其有自己独特的行为模式。基于用户行为的规律性，可以对其行为进行建模，得到其行为模式基线，不仅可以区分不同用户，还可以检测行为异常。

本节拟通过用户的行为数据来研究用户流量之间的关系：（1）流量行为的时间特征，即特征在时间上的动态变化；（2）用户流量行为分布的动态变化。

1. 日常行为分析

首先，通过 Wireshark[7]采集用户的几种典型的日常行为流量进行直观分析。试图从底层网络流量元数据中识别和提取上层应用行为。

网络流量的元数据中有一些属性能够直接描述用户的行为。例如，通过 TCP 的 SYN 和 FIN 标志的组合，可以识别通信流的开始和结束，这意味着用户活动的开始和结束。但是，大部分元数据不能直接表现用户的网络行为，却可以分析出用户的网络行为。例如，当用户通过即时消息应用程序聊天时，较长的数据报可能表示其在对话中发送了很多内容。如表 11 所示。

表 11 用户应用交互级别相关的网络元数据

活动定义	网络元数据标识
开始时间	TCP SYN Flag
结束时间	TCP FIN Flag
应用名称	目的 IP
服务类型	目的端口
活动	载荷长度

又比如，当用户浏览网页，Client 向 DNS Server 通信，获取其 IP 地址，服务端口为 53，随后就进入了正常的 TCP 连接。然而，当用户使用即时消息软件时则需要认证用户和安全通信，此时在 TCP 连接之后，Client 与 Server 执行 SSL 握手协议来建立安全的通信，以 443 端口通信，直至活动结束。因此，目的端口通常代表了特定的应用服务，也是本文在特征提取时保留目的端口的原因。

为了更好的理解通过底层元数据来表示应用层的活动，本节汇总了部分浏览网页、即时通讯和文件传送的特征。如表 12 所示。

表 12 用户行为元数据分析

应用	行为	协议	目的端口	载荷长度	包数量	方向
浏览器	浏览网页	TCP	随机	不固定	不固定	Server>Client
浏览器	看视频	TCP	随机	MTU	大量	Server>Client
即时消息软件	聊天	TCP	443	794+	1	Client>Server
FTP	共享文件	UDP	随机	MTU	大量	Client 发送
FTP	共享文件	UDP	随机	固定	大量	Client 接收

2. 用户行为差异

经过上述分析发现，在建立完整流的基础上，仅需要几个基本特征就能定义用户与时间相关的行为，能够区分不同用户的行为习惯，如表 13 所示。

表 13 用户行为的基本表示

用户行为的基本表示
源 IP
目的 IP
源端口
目的端口
协议类型
时间戳
载荷长度
包头长度
Flag

这些信息既可以定义动作的方向，也可以反映出动作的趋势。

3.3 攻击行为分析

攻击行为通常分两个方面特征进行分析，首先是内容特征，分析协议段中的值和字符等；其次是流量层面的网络连接特征和数据流的统计特征。本节主要分析流量层面的特征，流量的特征展示角度为被攻击主机接收到的流量与传回流量的特征。本节按下图所示的一般流程进行阐述。

图 14 一般攻击流程

3.3.1 探测类

在此阶段，攻击者通常试图先发现目标，再确认目标（收集信息），最后发现脆弱点（是否有安全漏洞），通常不会对计算机造成实质性的危害。探测类攻击最常使用端口扫描，属于该类攻击的第二阶段。

端口扫描攻击的目的通常是向大范围的主机发起连接，返回对哪些主机的哪些端口连接成功，以保留相关信息用于攻击。对网络端口进行扫描不仅可以获取计算机开放的服务程序和版本信息，还可以发现系统的漏洞。因此，尽早发现端口扫描行为，可以有效防范实质攻击事件的发生。端口扫描攻击可以按协议分类，如图 15 所示。

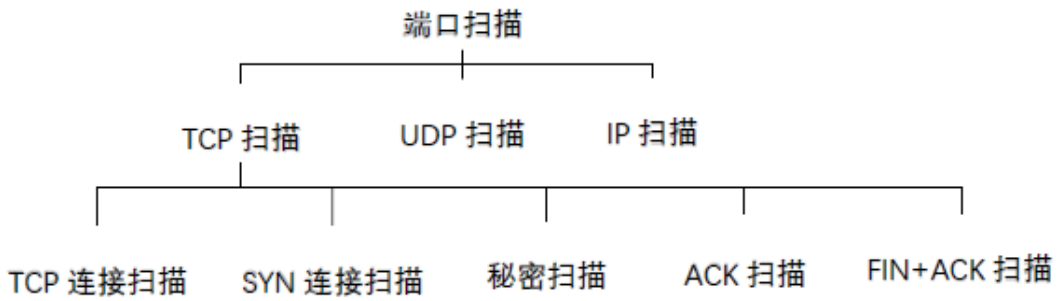

图 15 端口扫描形式

1. TCP 扫描

（1）TCP 连接扫描

该扫描又称为开放扫描，拥有完整的“三次握手”过程。其优势是扫描的可靠性较高，不需要 root 权限；缺点是同时也会产生审计信息，留下痕迹，容易被发现。

从数据分析来看，该攻击的特征为：小包多；会话中包的字节和数量相似的较多；TCP Flag 统计中其同步位发送与复位接收的 Flag 数目较多。

（2）SYN 扫描

该扫描又称半开放扫描，无需建立完整的 TCP 连接，因此不易被发现。攻击的行为是发送一个 SYN 包，若对端回复 SYN-ACK 包，则发现端口。

（3）秘密扫描

即不建立 TCP 连接。攻击者仅发送没有标志位的 TCP 包，若目标主机的对应端口是关闭状态，则返回 RST 包，若主机不响应，则说明该端口是开放的。该方式的隐蔽性优于 SYN 扫描，但是这种数据包容易被丢弃。

（4）ACK 扫描

该扫描属于一种寻找存活主机的扫描。攻击者发送仅有 ACK 标志的包，无论目标主机端口是否开放，都会返回 RST 包，该扫描不能确定端口状态，但是能确定目标主机是否存活，且能够发现防火墙规则。

（5）FIN+ACK 扫描

同属于寻找存活主机的扫描，原理与 ACK 扫描类似。

（2）至（5）中的四种扫描均没有建立完整 TCP 连接，具有相似的特征：小包多，存在大量 RST 数据包，发送的数据包中使用的标志位计数多，扫描包中没有实质内容传送，其数据包 PSH 位标记为 0。如图 16 所示。

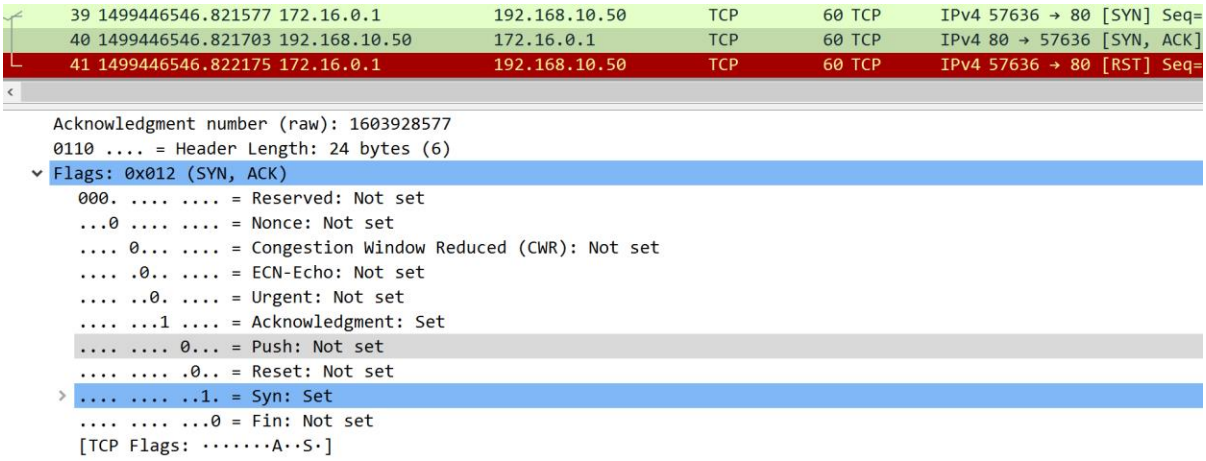

图 16 端口扫描攻击 Push 包状态

扫描发起时通常不限于一种形式，图 17 和图 18 中所示为 TCP 扫描的多种扫描方式发起的端口扫描攻击。攻击来自多个主机，每条流的持续时间极短，同一个攻击类型产生的行为中，包速率和被攻击方接收的数据包大小具有一定特征。

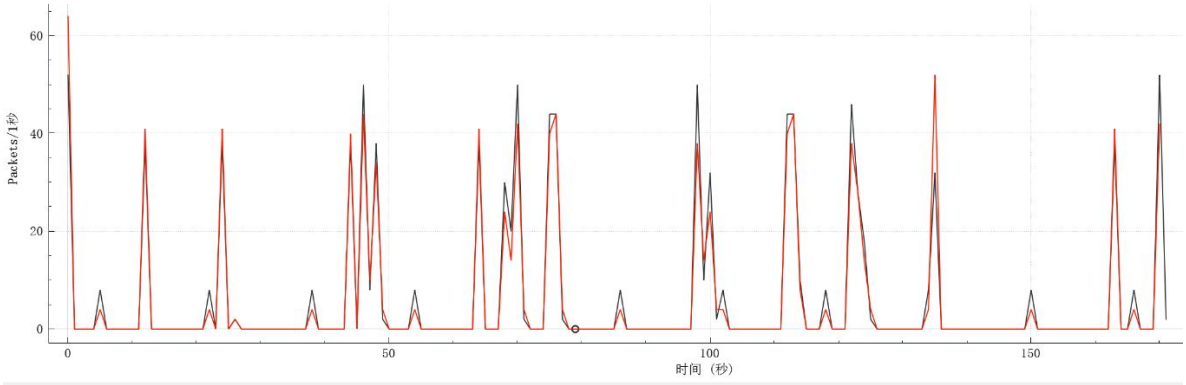

图 17 端口扫描被攻击方接收（红）与发送（黑）整体流量情况

Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration
192.168.10.5	53555	192.168.10.50	21	50	3472	22	1436	28	2036	12.605609	0.1410
192.168.10.5	53556	192.168.10.50	43970	29	22 k	19	21 k	10	612	12.723821	0.0223
192.168.10.5	53560	192.168.10.50	22	82	14 k	38	4716	44	9450	98.683812	1.2655
192.168.10.5	53561	192.168.10.50	22	80	13 k	38	4780	42	9022	99.949829	1.2895
192.168.10.9	7892	192.168.10.50	21	50	3470	22	1436	28	2034	44.795083	0.1581
192.168.10.9	7901	192.168.10.50	22	82	14 k	38	4716	44	9450	68.667040	1.4302
192.168.10.9	7902	192.168.10.50	22	80	13 k	38	4780	42	9022	70.097887	1.4295
192.168.10.12	36282	192.168.10.50	22	90	15 k	44	5584	46	10 k	46.681246	1.3075
192.168.10.12	36284	192.168.10.50	22	88	15 k	44	5648	44	9554	47.992866	1.3078
192.168.10.14	56337	192.168.10.50	22	84	14 k	38	4716	46	9570	170.555132	0.4491
192.168.10.15	55695	192.168.10.50	21	50	3472	22	1436	28	2036	64.331104	0.1599
192.168.10.15	55696	192.168.10.50	38779	29	22 k	19	21 k	10	612	64.456073	0.0339
192.168.10.15	55698	192.168.10.50	22	90	14 k	42	4956	48	9690	122.173365	1.4189
192.168.10.15	55699	192.168.10.50	22	88	14 k	42	5020	46	9262	123.592944	1.4444
192.168.10.16	48318	192.168.10.50	139	4	272	2	132	2	140	26.588280	0.0001
192.168.10.16	34244	192.168.10.50	22	90	15 k	44	5584	46	10 k	75.149167	1.3035
192.168.10.16	34246	192.168.10.50	22	88	15 k	44	5648	44	9554	76.452898	1.3357
192.168.10.16	44134	192.168.10.50	21	50	3944	22	1644	28	2300	163.718533	0.1487
192.168.10.16	35976	192.168.10.50	10014	29	22 k	19	21 k	10	676	163.852018	0.0146
192.168.10.17	44405	192.168.10.50	21	50	3946	22	1644	28	2302	24.555158	0.1280
192.168.10.17	57771	192.168.10.50	16506	29	22 k	19	21 k	10	676	24.672137	0.0105
192.168.10.17	34504	192.168.10.50	22	90	15 k	44	5584	46	10 k	112.205408	1.2511
192.168.10.17	34505	192.168.10.50	22	88	15 k	44	5648	44	9554	113.459815	1.3046
192.168.10.25	52891	192.168.10.50	21	64	4878	36	2576	28	2302	0.000000	0.1540
192.168.10.25	52892	192.168.10.50	49993	52	24 k	28	22 k	24	1648	0.116612	0.0368
192.168.10.25	52992	192.168.10.50	139	84	11 k	52	5944	32	5316	135.280534	0.0899

图 18 端口扫描被攻击方接收和发送数据情况

（2）UDP 扫描

该攻击的原理是给主机发送不含内容的 UDP 包，若主机响应，则说明端口开放，若返回“ICMP 端口不可达”，则表示端口是关闭状态，但是主机是存在的。以端口关闭的情况为例，如下图 19 和图 20 所示。

Protocol	Length	Protocol	Type	Info
UDP	60	UDP	IPv4	46186 → 21 Len=0
ICMP	70	ICMP,UDP	IPv4	Destination unreachable (Port unreachable)
UDP	60	UDP	IPv4	46186 → 22 Len=0
ICMP	70	ICMP,UDP	IPv4	Destination unreachable (Port unreachable)
UDP	60	UDP	IPv4	48806 → 22 Len=0
ICMP	70	ICMP,UDP	IPv4	Destination unreachable (Port unreachable)
UDP	60	UDP	IPv4	48806 → 21 Len=0
ICMP	70	ICMP,UDP	IPv4	Destination unreachable (Port unreachable)
UDP	60	UDP	IPv4	65002 → 22 Len=0
ICMP	70	ICMP,UDP	IPv4	Destination unreachable (Port unreachable)
UDP	60	UDP	IPv4	65002 → 21 Len=0
ICMP	70	ICMP,UDP	IPv4	Destination unreachable (Port unreachable)

图 19 UDP 端口扫描被攻击端接收与发送情况

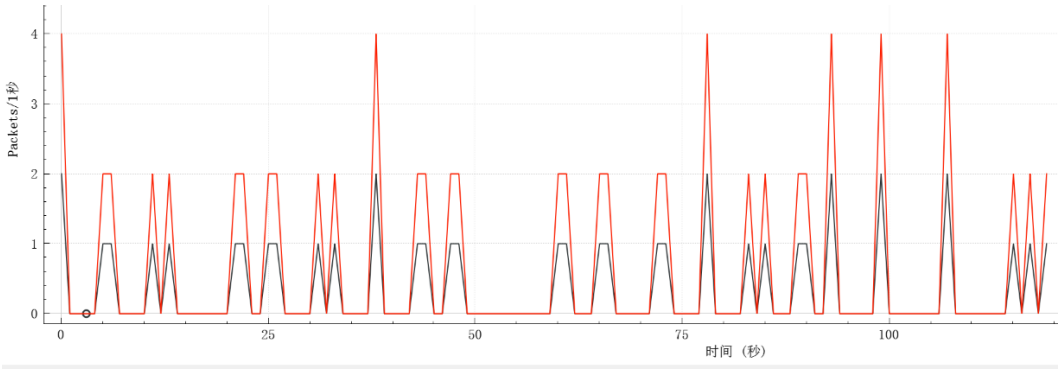

图 20 UDP 端口扫描被攻击端接收（红）与发送（黑）整体流量情况

上图中被扫描主机接收和发送的数据包均为小包，且大小分别是一致的，包速率很有规律。

（3）IP 协议扫描

该攻击仅针对 IP 协议发送数据包，报文不包含内容，若目标主机响应“ICMP 协议不可达”，则说明其 IP 协议是开放的。

该攻击特征为小包多，且 IP 数据包中包的字节和数量相似的较多。

综上三类端口扫描，特征集中于：小包多；含 Flag 的数据包较多；大量数据包具有相似特性。虽然在集中分析时攻击特征明显，但是攻击通常由不同主机发起，以五元组划分后分散在不同的流中，因此需要考虑流之间的时间关系。

体现在元数据上，被攻击方的显著特征归纳为：上行流包速率、下行流初始窗口字节数、PSH 包计数等。

3.3.2 执行类

1. 数据库(SQL)注入

该攻击属于 Web 攻击的一种，目的是非法获取重要服务器和用户的敏感数据[68]。此类攻击会先寻找注入点，当攻击者确定数据库类型，主要特征便体现在载荷上。如图 21 至图 23 所示。

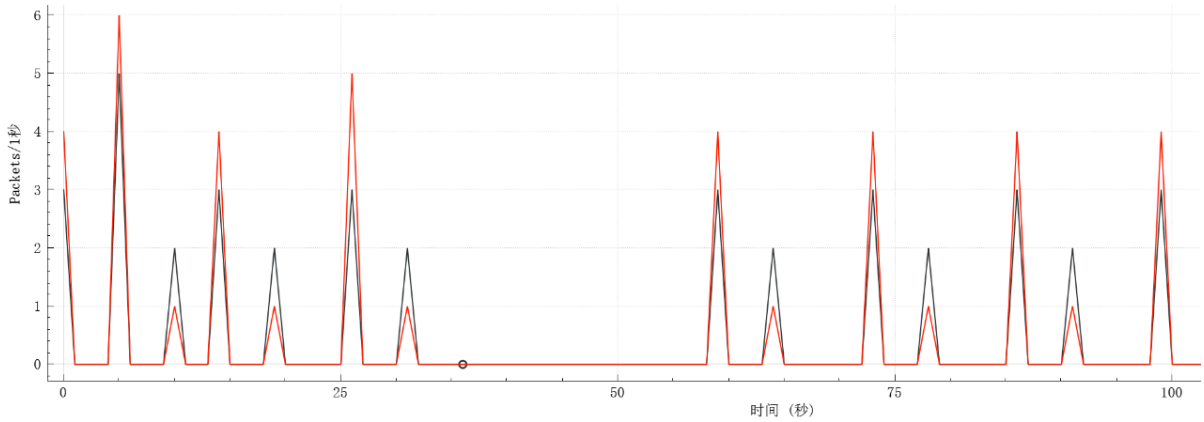

图 21 SQL 注入被攻击方接收（红）与发送（黑）整体流量情况

```
GET /dv/vulnerabilities/sqli/?id=1%27&Submit=Submit HTTP/1.1
Host: 205.174.165.68
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:45.0) Gecko/20100101 Firefox/45.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://205.174.165.68/dv/vulnerabilities/sqli/
Cookie: security=low; PHPSESSID=5dfcuh85kg0vvidf8nrsjtbob5
Connection: keep-alive

HTTP/1.1 200 OK
Date: Thu, 06 Jul 2017 13:40:07 GMT
Server: Apache/2.4.18 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 141
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

<pre>You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version
```

图 22 一次 SQL 注入攻击

```
Topic / Item
HTTP Requests by HTTP Host
  205.174.165.68
    /dv/vulnerabilities/sqli/?id=1%27+and+1%3D1+union+select+user%2C+password+from+users%23&Submit=Submit
    /dv/vulnerabilities/sqli/?id=1%27+and+1%3D1+union+select+null%2C+table_name+from+information_schema.tables%23
    /dv/vulnerabilities/sqli/?id=1%27+and+1%3D1+union+select+database%28%29%2C+user%28%29%23&Submit=Submit
    /dv/vulnerabilities/sqli/?id=1%27+and+1%3D1%23&Submit=Submit
    /dv/vulnerabilities/sqli/?id=1%27&Submit=Submit
```

图 23 SQL 注入攻击请求序列的正则表示

Topic / Item	Count	Average	Min val	Max val	Rate (ms)	Percent	Burst rate	Burst start
Packet Lengths	50	164.22	66	666	0.0004	100%	0.0500	26.046
0-19	0	-	-	-	0.0000	0.00%	-	-
20-39	0	-	-	-	0.0000	0.00%	-	-
40-79	41	67.76	66	74	0.0004	82.00%	0.0400	26.046
80-159	0	-	-	-	0.0000	0.00%	-	-
160-319	0	-	-	-	0.0000	0.00%	-	-
320-639	6	572.67	513	603	0.0001	12.00%	0.0100	0.000
640-1279	3	665.67	665	666	0.0000	6.00%	0.0100	26.047
1280-2559	0	-	-	-	0.0000	0.00%	-	-
2560-5119	0	-	-	-	0.0000	0.00%	-	-
5120 and greater	0	-	-	-	0.0000	0.00%	-	-

图 24 SQL 注入被攻击方接收的数据包大小分布情况

如图 24 所示，包长度具有一定的规律性。有研究对 URI 长度进行了统计，实验表明，注入攻击时的 URI 长度的平均值大多数大于正常流量的，但是就统计的长度标准差而言，正常流量的最大[69]。因此，在考量包长度等特征时，需要考虑其计算统计特征。例如，数值的标准差。

体现在元数据上，被攻击方的显著特征归纳为：下行流初始窗口字节数、上行子流发送字节数、下行子流字节数、下行流报文总大小等。

2. 暴力破解

该攻击的目的是针对有大量数据的设备，攻击者除了破坏正常业务，更可以窃取重要资料并破坏数据库。流程上与 SQL 注入相似，门槛却低很多。攻击发生时，被攻击主机的流量行为会产生异动，反复被攻击者尝试发现系统或服务密码，小包数量增多，通讯时间变短，频率变高。该攻击是基于探测类攻击发生的，攻击者首先知晓目标的开放端口、漏洞等情况，而后实施大量爆破式攻击。

该类攻击的特点为被攻击者前期接收小包多，通讯时间短、频率高；由于针对特定的数据库等，因此攻击的端口相对固定。与时间相关的特征，例如攻击具有连续性，如图 25 至图 29 所示。

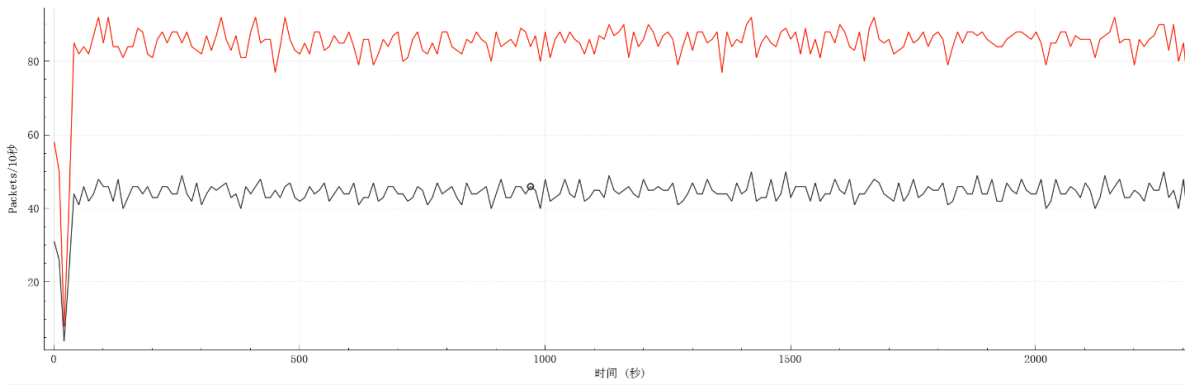

图 25 暴力破解被攻击方接收（红）与发送（黑）整体流量情况

Topic / Item	Count	Average	Min val	Max val	Rate (ms)	Percent	Burst rate	Burst start
Packet Lengths	10248	570.71	66	7306	0.0044	100%	0.0800	0.000
0-19	0	-	-	-	0.0000	0.00%	-	-
20-39	0	-	-	-	0.0000	0.00%	-	-
40-79	3014	69.58	66	74	0.0013	29.41%	0.0500	326.624
80-159	0	-	-	-	0.0000	0.00%	-	-
160-319	0	-	-	-	0.0000	0.00%	-	-
320-639	3616	432.16	428	578	0.0016	35.28%	0.0200	0.137
640-1279	3614	1122.65	807	1200	0.0016	35.27%	0.0200	0.000
1280-2559	1	2200.00	2200	2200	0.0000	0.01%	0.0100	0.000
2560-5119	1	3648.00	3648	3648	0.0000	0.01%	0.0100	35.905
5120 and greater	2	6582.00	5858	7306	0.0000	0.02%	0.0100	0.000

图 26 暴力破解被攻击方发送数据包大小分布情况

Topic / Item	Count	Average	Min val	Max val	Rate (ms)	Percent	Burst rate	Burst start
Packet Lengths	19643	226.84	66	668	0.0085	100%	0.1300	35.852
0-19	0	-	-	-	0.0000	0.00%	-	-
20-39	0	-	-	-	0.0000	0.00%	-	-
40-79	12411	66.87	66	74	0.0054	63.18%	0.0900	35.852
80-159	0	-	-	-	0.0000	0.00%	-	-
160-319	0	-	-	-	0.0000	0.00%	-	-
320-639	5426	445.93	375	467	0.0023	27.62%	0.0400	0.000
640-1279	1806	668.00	668	668	0.0008	9.19%	0.0100	1.113
1280-2559	0	-	-	-	0.0000	0.00%	-	-
2560-5119	0	-	-	-	0.0000	0.00%	-	-
5120 and greater	0	-	-	-	0.0000	0.00%	-	-

图 27 暴力破解被攻击方接收数据包大小分布情况

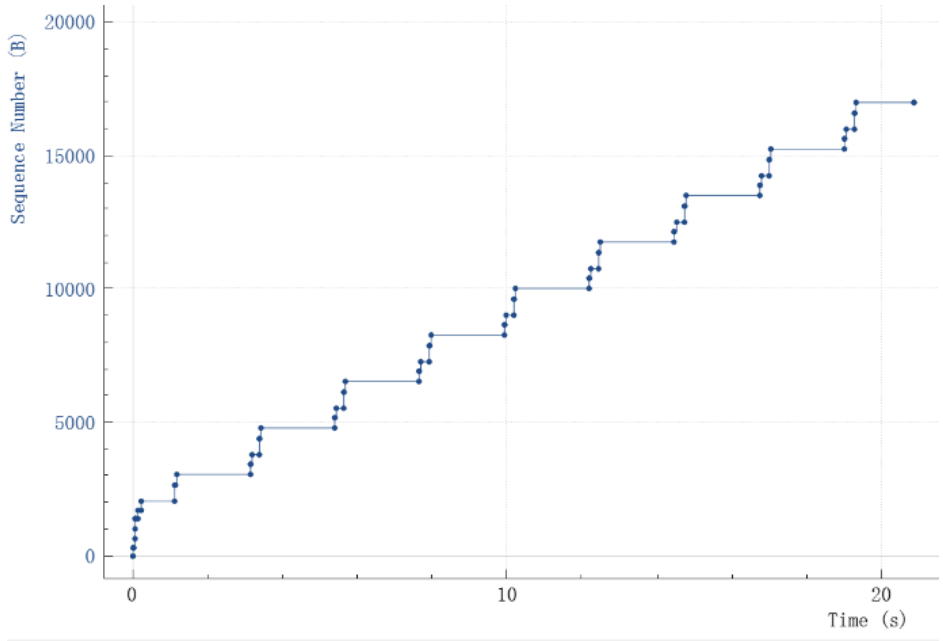

图 28 暴力破解攻击时间序列

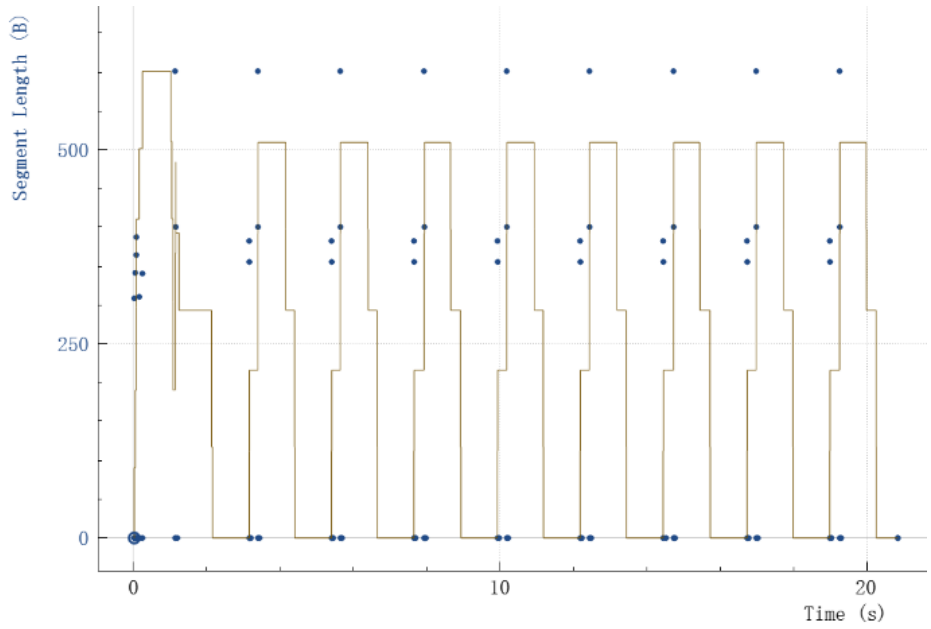

图 29 暴力破解攻击吞吐量

体现在元数据上，被攻击方的显著特征归纳为：下行流初始窗口字节数、下行子流发送字节数、下行流报文总大小、Flag 计数、下行包速率等。

3.3.3 控制类

该类攻击中最常使用隐蔽隧道通信，它构建于各层协议上，但是其中应用层的协议，例如 HTTP、HTTPS、DNS 等更容易实现隧道通信。隐蔽隧道攻击通常是将通信内容封装在协议的数据包中进行传输，当数据包到达后解析通信内容。

DNS 协议用于把域名解析为 IP 地址，是网络通信中必不可少的服务。其流量通常不像 HTTP 或 SMTP 等协议那样受监控，通常很少被防火墙和 IDS 过滤。因此，DNS 隧道作为可以绕过网络中的访问和安全策略的机制，为 DNS 隐蔽隧道通信制造了极大的方便，也被证明是建立僵尸网络或其他恶意通信链接的理想选择。本节以 DNS 隧道攻击为例进行分析。

该攻击的目的是通过 DNS 协议封装，建立相应通信，从而实现远程控制或文件传输等目的。该攻击一般分为 IP 直连型和域名型。IP 直连型由 Client 直接封装数据，速度较快，但隐蔽性弱。域名型则是通过迭代查询的中继方式，隐蔽性较好，但由于需要通过多个节点，速度较慢。

攻击特征：DNS 隧道在传输时通常会将数据切为小单元传送，间隔时间非常短。如图 30 和图 31 所示。

Source	Destination	Protocol	Length	Protocol	Type	Info
140.240.220.64	45.80.170.1	DNS	137	UDP	IPv4	Standard query 0x027a A a2cc1c075805f7aa6e543402140
74.26.155.117	45.80.170.1	DNS	117	UDP	IPv4	Standard query 0x00a0 SOA b1f2d0ce-bdff-42fb-a431-8
128.196.244.125	45.80.170.1	DNS	136	UDP	IPv4	Standard query 0x85d1 A cdb0f846eb91237441b92fcbe1c
112.67.178.97	45.80.170.1	DNS	136	UDP	IPv4	Standard query 0x5ae5 A 902f34ded310ca9619c046bfcec
167.176.145.72	45.80.170.1	DNS	134	UDP	IPv4	Standard query 0xc989 SOA 41b3bb01-7eb9-4d17-92c5-0
193.130.247.122	45.80.170.1	DNS	130	UDP	IPv4	Standard query 0xe6dd A 1953e3a4-ba0b-492e-acd6-7f4
193.130.247.122	45.80.170.1	DNS	130	UDP	IPv4	Standard query 0x95fe A 73287ae1-e7fd-4157-a180-450
63.69.208.223	45.80.170.1	DNS	125	UDP	IPv4	Standard query 0xad28 A 902f34ded310ca9619c046bfcec
40.127.53.125	45.80.170.1	DNS	125	UDP	IPv4	Standard query 0x5f87 AAAA 902f34ded310ca9619c046bf

图 30 DNS 隧道接收数据包大小

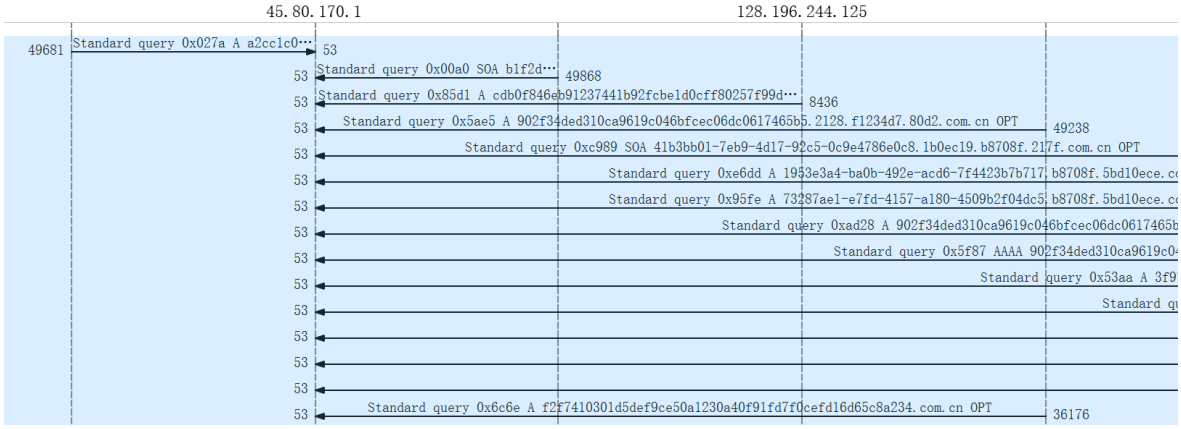

图 31 DNS 隧道传送文件过程

当没有数据交互而保持连接状态时，会保持间隔时间内发送一个数据包。如图 32 所示。

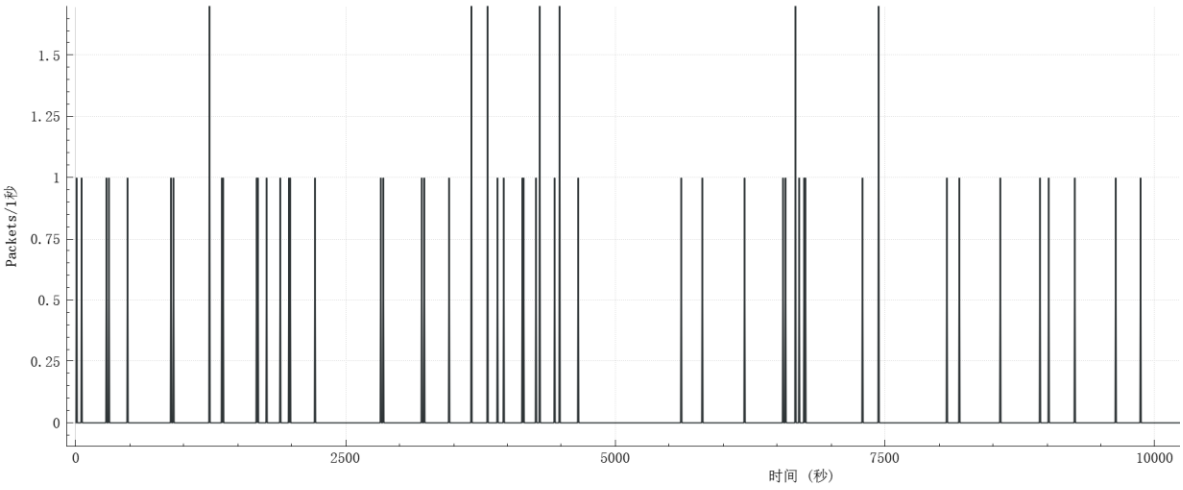

图 32 DNS 隧道攻击接收数据整体流量情况

DNS 隧道攻击会随着木马生命周期结束而终止，在攻击的整个过程中包含了传输信息和保持心跳等流量，因此，应该关注流中数据包的总数分布情况。

体现在元数据上，被攻击方的显著特征归纳为：流持续时间、下行子流接收字节数，以及下行流报文总大小、流活跃时间等的分布变化。

3.3.4 破坏类

该类攻击中最常见的是 DDoS 攻击。DDoS 攻击的发起一般有以下几个步骤：首先，利用安全漏洞植入木马、蠕虫或后门等或者使用僵尸网络，以控制大量用于攻击的主机。然后，可以利用通信协议的脆弱性放大流量起到倍增目的。最后，在攻击者控制下一同发起攻击，造成严重影响。本小节以攻击目的、利用手段和增强方式的顺序举例做出分析。

1.攻击目的

DoS 攻击[71]的目的是阻止或限制用户使用服务或者网络资源，通常是通过过度占用连接到计算机的服务来进行的。该攻击一般分两种形式：资源耗尽和资源过载。DoS 攻击的发起者使用一台主机向目标不间断的发送数据，使目标对象的网络出现拥堵，无法应对过多的数据请求，从而导致正常的服务请求被拒绝。

DDoS 攻击与 DoS 攻击的区别在于，DDoS 的一个或多个攻击者可以通过僵尸网络[70](Botnet)控制大量主机同时发起攻击，导致目标网络更严重的拥堵[72]。DDoS 规模更大，危害更严重，但是隐蔽性较弱。

低速率拒绝服务(Low-rate Denial of Service, LDoS)攻击则与 DDoS 在使用资源上相反，它在应用层上仅用低速率的请求占用目标对象的资源，其构造的行为与合法行为近似，隐蔽性较强，相对难以检测[73]。

以 DDoS 攻击为例，通常是指高速率的攻击。当攻击发生时，在一段时间内会发生突发的流量增加，其峰值较高。其特征体现在：流量具有相似的特性，目标主机的访问流量大小、数据字节和数量均相似；大量 Request 数据包；大量 SYN 数据包；发送数据长度超过 65535 字节等。

体现在元数据上，被攻击方的显著特征归纳为：流持续时间，流活跃时间、上行报文大小、数据包大小的分布等。

2. 攻击手段

DDoS 攻击有两种类型，即洪范攻击和漏洞攻击[74]。洪范攻击是通过 Botnet 进行，以控制被感染主机发动 HTTP 洪范攻击、邮件攻击等。以 Botnet 作为手段为例，其目的是通过入侵网络远程控制若干无关用户终端的攻击行为。如图 33 中，攻击者将指令以报文形式发送给被控主机，使被感染的主机构成僵尸网络。标红色的为攻击者发起的流量，标彩色的为 5 台被感染主机的流量，体现出高度的协同性。

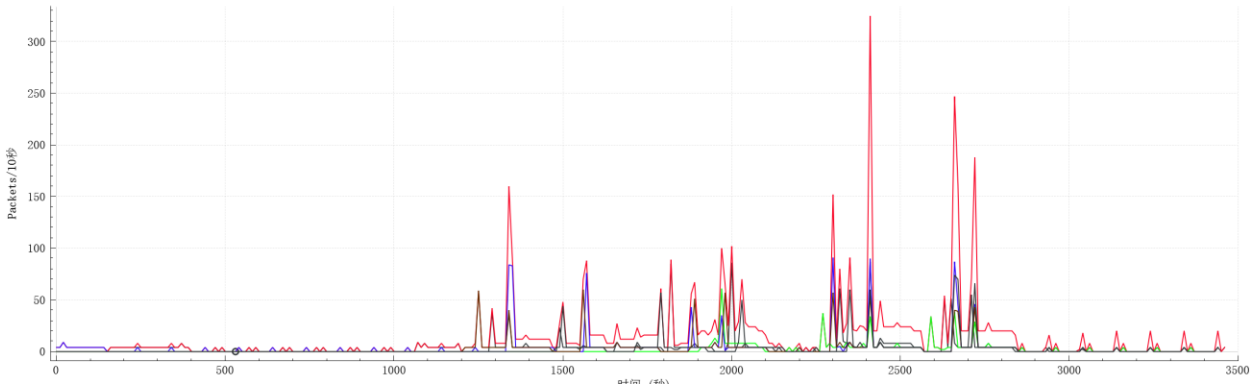

图 33 僵尸网络攻击端（红）与被攻击端（彩色）整体流量情况

该类攻击体现在被攻击方元数据上的显著特征为：下行子流字节数、下行流报文总大小、上行流包速率，以及下行流报文大小的分布等。

3. 增强方式

根据 DDoS 洪范攻击的机制，分为直接攻击和间接（通过反射器）攻击，间接攻击可以导致攻击的规模更大。根据目标协议级别，可以分为传输级别（例如 TCP、UDP、ICMP 等）和应用级别（例如 HTTP、DNS 等）的攻击，它们的共性都是利用了协议本身的脆弱性[75]。

以 DNS 协议的放大反射为例，它基于 UDP 协议。应用该倍增方式的 DDoS 攻击曾在 2016 年轰动一时，攻击者使用当时新型的 Mirai Botnet，将互联网的大部分 DNS 系统引向美国 Dyn 公司的基础设施，形成了当时有史以来最大规模的 DDoS 攻击[76]。该攻击的原理是通过伪造域名，导致递归查询，从而放大流量。主要方式是发送大量带有被攻击者 IP 地址的数据包给成千上万的攻击主机，然后，攻击主机对 IP 地址做出大量回应，形成拒绝服务攻击。因此，又可以称作分布式反射拒绝服务(Distributed Reflection Denial of Service，DRDoS)攻击。

图 34 至图 38 中是两种 DNS 反射放大攻击的表现：通常查询包为 ANY 类型请求，以便接收可能的最大响应；返回的响应包字节量远远大于发送包字节量等。

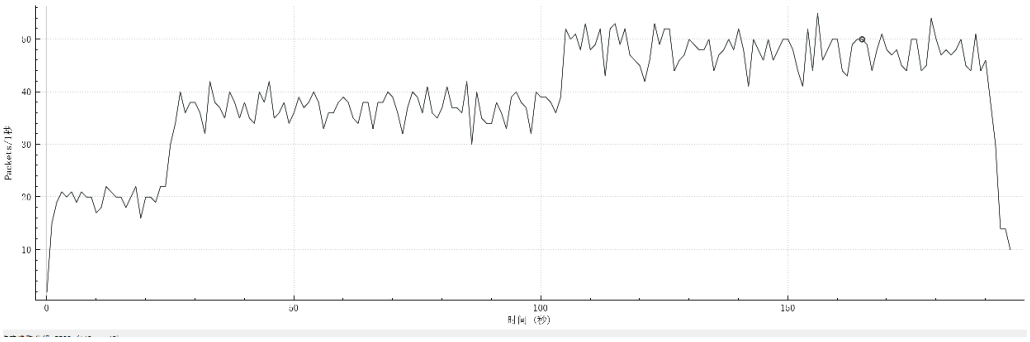

图 34 DNS 反射放大攻击整体流量情况-1

查询包与响应包的数量差距极大：

DNS	78	Standard query 0x67c6 ANY d1a4.cc OPT
DNS	4061	Standard query response 0x67c6 ANY d1a4.cc SOA ns1.c1daa87c667f52.net A 232.5.157.59 MX 10 abelard.c1daa87c667f52…
DNS	78	Standard query 0x67c6 ANY d1a4.cc OPT
DNS	4061	Standard query response 0x67c6 ANY d1a4.cc SOA ns1.c1daa87c667f52.net A 232.5.157.59 MX 10 abelard.c1daa87c667f52…
DNS	78	Standard query 0x67c6 ANY d1a4.cc OPT
DNS	4061	Standard query response 0x67c6 ANY d1a4.cc SOA ns1.c1daa87c667f52.net A 232.5.157.59 MX 10 abelard.c1daa87c667f52…

图 35 DNS 反射放大攻击查询包数与相应包数比例-1

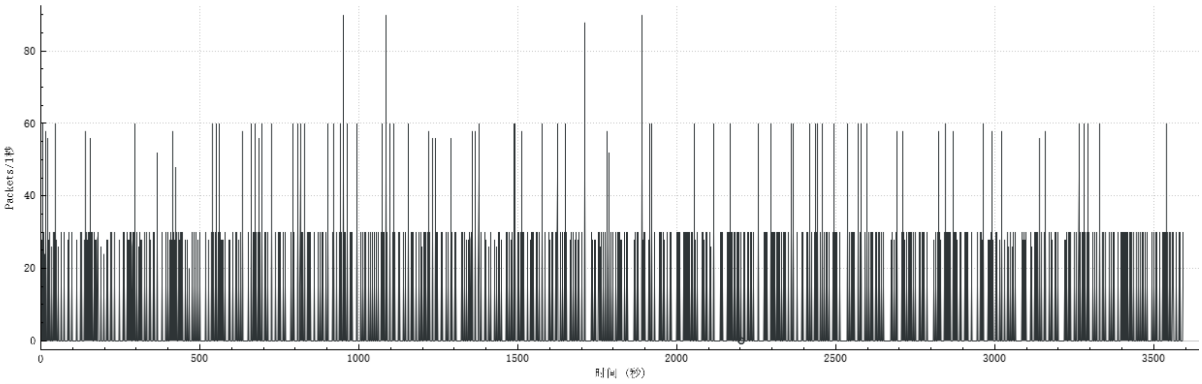

图 36 DNS 反射放大攻击整体流量情况-2

DNS	80	UDP	IPv4 Standard query 0xda35 ANY 734a5.gov OPT
DNS	80	UDP	IPv4 Standard query 0xda35 ANY 734a5.gov OPT
DNS	80	UDP	IPv4 Standard query 0xda35 ANY 734a5.gov OPT
DNS	80	UDP	IPv4 Standard query 0xda35 ANY 734a5.gov OPT
DNS	80	UDP	IPv4 Standard query 0xda35 ANY 734a5.gov OPT
DNS	80	UDP	IPv4 Standard query 0xda35 ANY 734a5.gov OPT
DNS	80	UDP	IPv4 Standard query 0xda35 ANY 734a5.gov OPT
DNS	3969	UDP	IPv4 Standard query response 0xda35 ANY 734a5.gov MX 0 usadf-gov.ma
DNS	3969	UDP	IPv4 Standard query response 0xda35 ANY 734a5.gov MX 0 usadf-gov.ma
DNS	3969	UDP	IPv4 Standard query response 0xda35 ANY 734a5.gov MX 0 usadf-gov.ma
DNS	3969	UDP	IPv4 Standard query response 0xda35 ANY 734a5.gov MX 0 usadf-gov.ma
DNS	3969	UDP	IPv4 Standard query response 0xda35 ANY 734a5.gov MX 0 usadf-gov.ma
DNS	3969	UDP	IPv4 Standard query response 0xda35 ANY 734a5.gov MX 0 usadf-gov.ma
DNS	3969	UDP	IPv4 Standard query response 0xda35 ANY 734a5.gov MX 0 usadf-gov.ma
DNS	3969	UDP	IPv4 Standard query response 0xda35 ANY 734a5.gov MX 0 usadf-gov.ma

图 37 DNS 反射放大攻击查询包数与相应包数比例-2

Topic / Item	Count	Average	Min val	Max val	Rate (ms)	Percent	Burst rate	Burst start
Packet Lengths	11868	3969.00	3969	3969	0.0033	100%	0.3000	660.922
0-19	0	-	-	-	0.0000	0.00%	-	-
20-39	0	-	-	-	0.0000	0.00%	-	-
40-79	0	-	-	-	0.0000	0.00%	-	-
80-159	0	-	-	-	0.0000	0.00%	-	-
160-319	0	-	-	-	0.0000	0.00%	-	-
320-639	0	-	-	-	0.0000	0.00%	-	-
640-1279	0	-	-	-	0.0000	0.00%	-	-
1280-2559	0	-	-	-	0.0000	0.00%	-	-
2560-5119	11868	3969.00	3969	3969	0.0033	100.00%	0.3000	660.922
5120 and greater	0	-	-	-	0.0000	0.00%	-	-

图 38 DNS 反射放大被攻击方接收数据包的分布情况

3.3.5 本节小节

将上述分析的常见的攻击行为归纳为四种行为：探测行为，以发现目标的脆弱点；感染行为，以获取目标的操作权限；窃密行为，以泄露目标的数据；破坏行为，以使目标无法正常工作。它们之间大多存在紧密的关联，越早发现行为异常，越能够及时避免遭受攻击。攻击行为的基本特征总结如下表所示。

表 14 攻击行为基本特征

攻击行为基本特征		
流量持续时间	数据包数量	数据包中的标志位计数
上行包与下行包数比	数据最小分段大小	带有效载荷包数
两条流的间隔时间	流的活跃时间	数据包大小
每秒传输数据包数	每秒传输字节	子流传输字节数

3.4 基于网络的数据集概述

入侵检测数据集对 IDS 的验证起着至关重要的作用。用于 IDS 研究的数据集通常分三类：合成的数据集、真实的数据集、基准数据集[77]。获取方式有两种：自行采集数据和使用公开基准数据集。由于隐私问题，真实产生的数据通常不容易获取。研究者通常使用公开的数据集进行研究和验证。

本节的讨论关注这些广泛应用的公开数据集的属性，包括创建年份、正常用户行为的采集时长、攻击流量种类、攻击样本数量、元数据特征、流量的来源、是否为完整网络、是否包含原始流量，以及它们的局限性。本节介绍四个数据集。

3.4.1 KDDCUP99 概述

讨论到入侵检测数据集，不得不提起鼻祖 KDDCUP99 数据集[79]。该数据集由 DARPA 在 MIT Lincoln 实验室于 1998 年生成的数据整理简化而来。数据采集自 9 个星期的 TCPDump 网络连接和系统审计数据，正常数据和异常数据都是流量生成器合成产生的，是一种附加了日志等丰富信息的数据集。数据集包含 1 种正常数据，4 类共 39 种攻击标识，其中训练集中有 22 种攻击标识。数据向量包含 41 个特征，其中 9 个为离散型特征。

该数据集为 IDS 研究做出了不可磨灭的贡献。但是，由于其生成的时间较早，仿真数据的真实性和多样性都已经过时。虽然此数据集仍旧会被一些研究人员用作基准对比，但是由于数据集没有发布原始 pcap 格式数据，不能用于纯流量的入侵检测领域。

表 15 中能更直观的了解 KDDCUP99 数据集。

表 15 KDDCUP99 数据集详情

标识		特征内容	数据类型	样本数量
正常	攻击类型	单个 TCP 连接的基本特征	5 个连续型；4 个离散型	训练数据 5000000
Normal	DOS	一次连接中包含的内容特征	8 个连续型；5 个离散型	测试数据 2000000
	Probing（监视和其他探测活动）	基于时间（2 秒时间窗口）计算的统计特征	9 个连续型	
	R2L（来自远程机器的非法访问）	基于主机的统计特征	10 个连续型	
	U2R（普通用户对本地超级用户特权的非法访问）			

由于数据集的采集对研究也具有时效性，本节在之后讨论的数据集全部创建于 2010 年之后，而且是仅基于数据包和（或）数据流的流量数据，即不包括来自于主机日志等设备的附加属性。

3.4.2 ISCXIDS2012 概述

该数据集于 2012 年由加拿大网络安全研究所生成[80]。生成的过程是使用实体设备模拟用户的行为，之后设计并执行攻击方案。数据对应用程序、协议以及底层网络实体进行了描述，其中大部分是 HTTP 协议，也包含少量 FTP、STMP 和 IMAP。数据集包含了完整的网络环境，给出了原始流量文件，没有为数据集提取特征。流量数据一共 7 天，其中包括 1 天全正常流量和 6 天包含有攻击的流量。与 KDDCUP99 数据集相比，该数据集的攻击与现实更接近，但持续时间缩短了很多。攻击流量包含 4 种类型的攻击：Infiltration、DDoS、HttpDoS、BruteForce SSH。但是，该数据集没有为攻击类别给出明确的标签，仅标记了“正常”或“攻击”，作为实验数据难以解释结果。

3.4.3 UNSW-NB15 概述

该数据集是澳大利亚网络安全中心 Cyber Range 实验室于 2015 年生成的一个综合数据集。使用了当时最新的技术 IXIA 流量生成器，使仿真的数据更接近真实网络[81]。数据采集了 2 天，时长共 31 个小时，包含 9 种攻击，提取了 47 个特征，并将特征划分为 6 类（流标识、基本特征、基于内容的特征、基于时间的特征和 2 类附加生成时间特征）。表 16 中给出了详细的样本数量以及特征详情。

表 16 UNSW-NB15 数据集详情

标记	样本数量	特征组成	特征详情
Normal	2218761	5 个流标识	源 IP；源端口；目的 IP；目的端口；
Fuzzers	24246	13 个基本特征	协议状态及其相关协议；总持续时间；源到目的字节数；目的到源的字节数；源到目的地的生存时间；目的到源的生存时间；源数据包转发或丢弃；目的数据包转发或丢弃；服务类型；源每秒字节数；目的每秒字节数；源到目的包数；目的到源的包数
Analysis	2677	8 个内容特征	源 TCP 窗口通告；目的 TCP 窗口通告；源 TCP 序列号；目的 TCP 序列号；源发出的流数据包大小的平均值；目的发出的流的包大小的平均值；进入 http 请求/响应事务的连接深度；从服务器的 http 服务传输的数据的内容大小
Backdoors	2329	9 个基于时间的特征（毫秒）	源抖动；目的抖动；记录开始时间；记录上次时间；源包间到达时间；目的包间到达时间；TCP 的“ syn”和“ ack”计数；TCP 的 SYN 和 SYN_ACK 数据包之间的时间；SYN_ACK 与 TCP 的 ACK 数据包之间的时间
DoS	16353	附加生成特征——5 个通用特征	如果源 IP 与目的 IP 相同，且端口号相同，变量值取 1，否则取 0; 根据源/目的生存时间的值的特定范围的状态；具有 http 服务中的 Get 和 Post 之类操作的流数；如果通过用户和密码访问 ftp 会话，取 1，否则取 0；ftp 会话中没有命令的流数
Exploits	44525	附加生成时间——7 个连接特征（每 100 个连接）	包含相同服务类型和源 IP 的连接数；包含相同服务类型和目的 IP 的连接数；相同目的 IP 的连接数；相同源 IP 的连接数；相同源 IP 和目的端口的连接数；相同目的 IP 和源端口的连接数相同源 IP 和目的 IP 的连接数。
Generic	215481	2 个标记特征	每个攻击类别名称
Reconnaissance	13987		标签
Shellcode	1511		
Worms	174		
训练数据	175341		
测试数据	82332		

3.4.4 CICIDS2017 概述

该数据集是目前比较新的 IDS 数据集，与 ISCXIDS2012 同出于一家，是由 CSE 与 CIC 两个机构合作生成[82]。数据包括 5 天的流量数据，星期一是正常的流量数据，其余 4 天的流量里包含 7 种攻击：Infiltration、Dos、DDoS、Web-Attack、Port-Scan、Patator 和 Botnet。数据集包含 1 种正常标识以及 14 种攻击标识，包含原始流量文件和 84 个特征的特征集。

该数据集是目前领域内比较新的数据集，这些用户的正常流量和攻击流量是在完整的网络环境下仿真生成的。

3.4.5 现有数据集的不足

表 17 数据集参数对比

参数	KDDCUP99	ISCXIDS2012	UNSW-NB2015	CICIDS2017
是否仿真	是	是	是	是
网络环境	小网络	小网络	小网络	小网络
数据采集时长	9 周	7 天	31 小时	5 天
是否有原始 pcap	否	是	是	是
攻击类型数量	4	4	9	7
是否有标签	有	无	有	有
特征提取数量	41	无	49	84

表 17 中汇总了本文关心的数据集属性，根据研究需求，现有基于网络的数据集存在以下三点不足：

1. 数据陈旧。随着网络设备以及应用程序的飞速更新，它们所产生的行为数据也在发生变化，而数据集的生成难以赶上更新迭代的脚步，这导致大部分数据集较为陈旧。

2. 正常流量数据采集时间不足。ISCX2012 和 CICIDS2017 数据集虽然公开了原始流量文件，但是总体采集的时长都在 7 天以内，而且这些数据集的重心是攻击行为，仅有少量的正常数据。

3. 正常流量数据不能反应用户行为。通过真实的网络环境采集数据，会涉及隐私问题，这导致更适合研究的数据往往不能进行共享。而通过仿真软件生成的网络流量数据类型非常有限，不能完全反应真实的网络状态趋势。而且，仿真中的正常流量数据不能保证同一 IP 地址就是同一用户的行为。

因此，本研究需要自行采集长时间的正常用户的行为流量。

3.5 数据生成

在本节的工作中有两个贡献：其一，是生成了一个长时间连续的完全真实的用户行为数据集；其二是为该数据集提取了 57 个特征（包含流标识、时间戳和用户标签）。将在后文详细阐述。

3.5.1 数据采集

本文的数据采集方式是最原始的主机采集方式，不需要借助第三方设备，直接使用 Wireshark 底层的 dumpcap 程序，对用户网卡关口的出入数据进行全流量抓取。为了确保不会因 DHCP 分配或更改 IP 产生对实验数据的影响，在采集之前需要进行 MAC/IP 绑定。采集流程如图 39 所示。

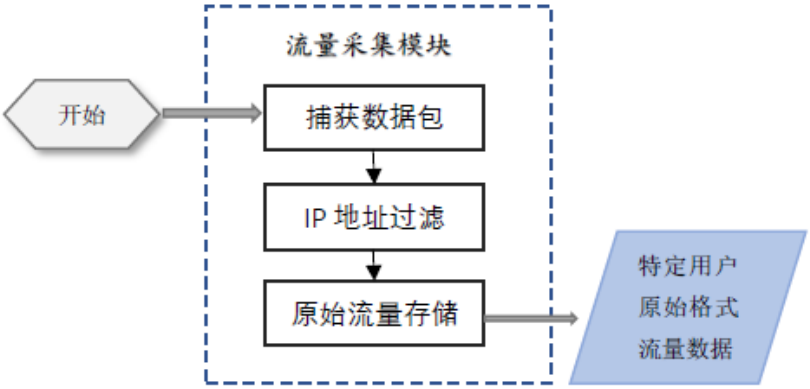

图 39 数据采集模块

本文的数据采集应持续较长时间，且 24 小时无间断，以捕捉周期性的影响（白天、夜晚或者工作日、休息日）。在数据采集期间，为了保证流量的真实性和多样性，没有对接受采集的用户限制操作行为和作息时间。流量采集的用户行为包括但不限于浏览网站、下载文献资料、文件传输、登录系统、即时通信、观看视频和游戏娱乐等。

为了避免非人为中断导致的数据丢失，将文件按 500000KB 分割，以 pcap 格式进行存储。

流量抓取命令：

```
dumpcap.exe -i \Device\NPF_{网卡地址} -w {存储目录}\packets.pcap -b filesize:500000 -F -pcap
```

存储效果如图 40 所示。

packets_00001_20190903184346.pcap	2019/9/3 20:22	Wireshark capture file	488,283 KB
packets_00002_20190903202256.pcap	2019/9/4 7:32	Wireshark capture file	488,283 KB
packets_00003_20190904073226.pcap	2019/9/4 14:39	Wireshark capture file	488,283 KB
packets_00004_20190904143930.pcap	2019/9/4 15:09	Wireshark capture file	488,282 KB
packets_00005_20190904150956.pcap	2019/9/4 18:35	Wireshark capture file	488,282 KB
packets_00006_20190904183551.pcap	2019/9/4 19:58	Wireshark capture file	488,283 KB
packets_00007_20190904195832.pcap	2019/9/4 21:16	Wireshark capture file	488,282 KB
packets_00008_20190904211617.pcap	2019/9/4 21:32	Wireshark capture file	488,283 KB
packets_00009_20190904213250.pcap	2019/9/4 22:53	Wireshark capture file	488,283 KB
packets_00010_20190904225334.pcap	2019/9/5 0:31	Wireshark capture file	488,282 KB

图 40 pcap 原始文件存储

3.5.2 特征构建与提取

特征构建与提取是模型性能好坏的决定性因素，是将采集的原始数据转换为模型可用数据的过程，包括特征构建和提取两个部分。

1. 特征构建

本文的构建特征依据网络通信行为和具有稳定性的行为。

（1）网络通信行为

依据第三章对攻击行为和用户行为的分析，选取能够具体反映用户日常交互行为和各种攻击行为的特征作为模型的学习数据。

（2）具有稳定性的行为

在正常的网络行为下，各个特征的属性能够具有相对稳定的分布。当行为发生异常，这种分布会有较大的幅度变化。

特征选择的目的是去掉无关和冗余的数据，保留能够代表给定问题的最优特征子集。原因主要有两点：首先，不必要的特征会增加计算的时间复杂度，使训练成本上升；其次，不必要的特征可能会使分类器过拟合或者精确度降低。

与以往在特征构建阶段仅提取尽量少的特征方法不同，本文提取的特征尽量涵盖广泛，将特征送入模型，自动学习特征中的重要表示，该方法将在下一章中进行详细说明。本文提取的特征中包含离散型的类别特征，也包含连续性的数值特征。其中，类别特征包含端口号、协议类型和数据标识；数值特征相对较多，提取符合上述（1）（2）两个特性的元数据信息，并对统计特征的计算特征取其数据的标准差。通常，在数据量较大的情况下，标准差更能反映出数据的离散程度。

本文提取的流级别特征为 54 维，如表 18 所示。

表 18 流级别特征

连续型特征		离散型特征
原始统计特征	计算统计特征	
流持续时间	两条流时间间隔标准差	协议类型
上行总包数	上行发送两个包间隔标准差	目的端口类别
下行总包数	下行发送两个包间隔标准差	
上行包总长度	上行数据包长度标准差	
下行包总长度	下行数据包长度标准差	
上行包头总长	数据包长度标准差	
下行包头总长	流处于空闲状态的时间标准差	
每秒上行包数	流处于活跃状态的时间标准差	

每秒下行包数	数据包平均大小
每秒传输字节数	上行数据分段平均大小
每秒传输包数量	下行数据分段平均大小
上行发送两个包间总时间	上行子流平均包数
下行发送两个包间总时间	下行子流平均包数
上行传输包中 PSH 标志次数	上行子流平均字节数
下行传输包中 PSH 标志次数	下行子流平均字节数
上行传输包中 URG 标志次数	上行平均字节数块速率
下行传输包中 URG 标志次数	下行平均字节数块速率
上行带有效载荷包数	上行平均包块速率
上行数据最小分段大小	下行平均包块速率
带 FIN 标志的报文数	上行平均块速率
带 SYN 标志的报文数	下行平均块速率
带 RST 标志的报文数	上传/下载比率
带 PUSH 标志的报文数	
带 ACK 标志的报文数	
带 URG 标志的报文数	
带 CWR 标志的报文数	
带 ECE 标志的报文数	
上行初始窗口发送字节数	
下行初始窗口发送字节数	
数据包最小到达时间	

2. 特征提取

（1）流的表示

本文特征提取以双向流量数据流进行记录，流的方向由第一个数据包的方向标定，使用标准五元组（源 IP、目的 IP、源端口、目的端口、协议类型）作为每一条流的 ID。流的基本信息表示如表 19 所示。

表 19 流的基本信息

从数据包生成流的基本信息	表示
ID	id
源 IP	srcIP
目的 IP	dstIP
源端口	srcPort
目的端口	dstPort
协议类型	protocol
时间戳	timeStamp
载荷长度	payloadBytes
flowID	
其他基本信息	**表示**
Flag 标识	FlagPSH
	FlagURG
	FlagECE
	FlagSYN
	FlagACK
	FlagCWR
	FlagRST
时间窗口	TCPWindow
包头长度	headerBytes

流的正/负方向表示为：

```
if(forward){
        this.flowId = this.getsrcIP() + "-" + this.getdstIP() + "-" + this.srcPort + "-" + this.dstPort + "-" +
        this.protocol;
}else{
        this.flowId = this.getdstIP() + "-" + this.getsrcIP() + "-" + this.dstPort + "-" + this.srcPort + "-" +
        this.protocol;
}
```

（2）特征提取

会话设置为 120 秒超时终止。其中，正常的 TCP 会话是由客户端向服务器发送 SYN 为起始，任意端发送 FIN 且数量等于 2 为终止。特征提取机制如图 41 所示。

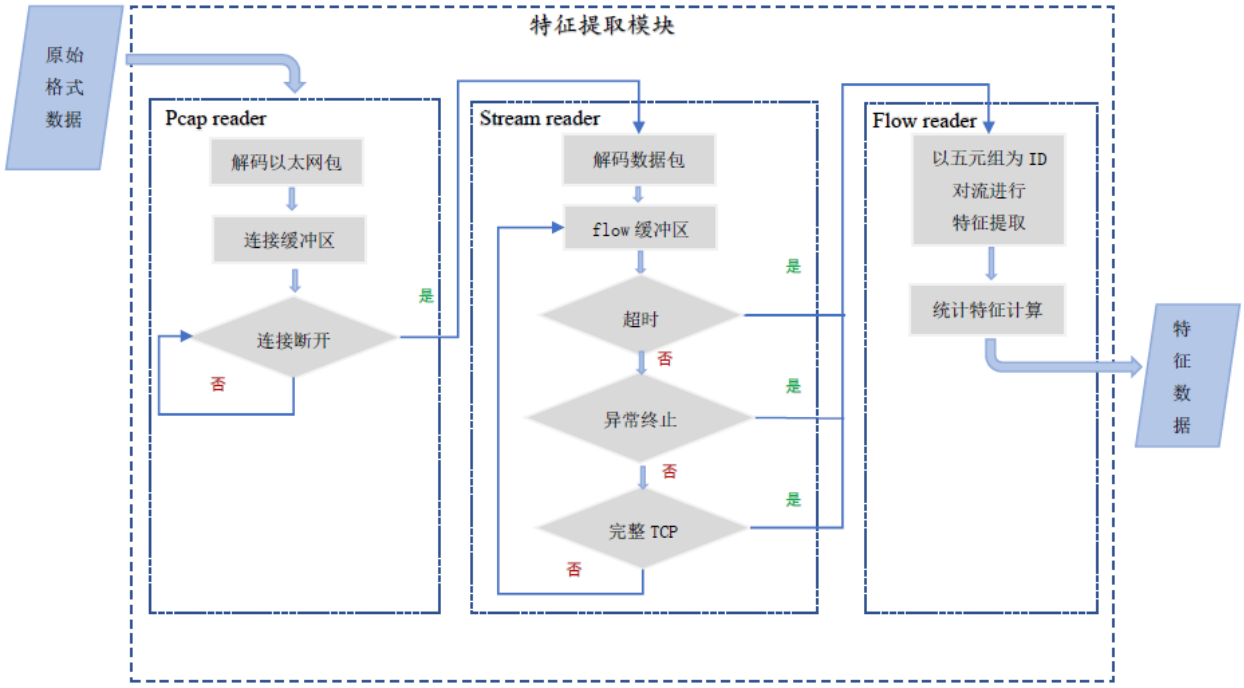

图 41 特征提取模块

当网络数据包到达时，网络流不会直接用于分析处理，而是依据规则形成会话流。如果会话因超时而结束，则将该会话流移至完成列表，从当前缓冲列表中删除。如果以 RST 标志异常终止，则将其移至完成列表，从当前缓冲列表中删除。如果 TCP 会话流以 FIN 标志结束，亦将该流移至完成列表，从当前缓冲区列表中删除。而后为会话流进行特征提取与计算。

3.5.3 生成实验数据

通过上述数据采集、特征提取和特征预处理过程，本章对 3 位用户为期 1 个月的日常流量进行了提取，流量数据按周进行整合，为下一节选取建模时长做数据准备。详细数据如表 20 所示。

表 20 正常用户样本数量

	用户 A	用户 B	用户 C
1 星期流数目	302903	748855	145283
2 星期流数目	534104	1010954	317509
3 星期流数目	746212	1312049	468843
4 星期流数目	920209	1722366	680151

攻击数据采用 CICIDS2017 数据集中的攻击数据，对其原始流量 pcap 文件进行提取，此数据集是目前比较新的入侵检测数据集，涉及的攻击类型有 Infiltration、Dos、DDoS、Web-attack、Port-scan、Patator 和 Botnet，每类攻击中又有不同的攻击方式。攻击数据详情如表 21 所示。

表 21 攻击样本数量

攻击类型	攻击详情	攻击流数	攻击总流数
Patator	FTP-Patator	7935	13832
	SSH-Patator	5897	
DDoS	DDoS	128025	128025
Web-attack	Sql Injection	21	2180
	Brute force	1507	
	XSS	652	
Infiltration	Infiltration	36	36
Botnet	Botnet	1956	1956
DoS	Hulk	230124	251723
	GoldenEye	10293	
	Slowloris	5796	
	Slowhttptest	5499	
	Heartbleed	11	
Port-scan	Port-scan	158804	158804

3.6 不平衡数据处理方法

不平衡数据意味着数据分布差异较大，目标类被充分采样，而异常类被严重欠采样。来自异常类的例子数量很少，可能是因为获得这类的大量训练数据太困难或成本太高。数据不平衡在入侵检测领域是普遍存在的现象，研究者们在研究过程中不断在数据与算法上进行改进，以解决这个困难。目前，针对不平衡数据的解决方法主要是两类：首先是在源头处解决问题，在数据采样时调节类别的数量，使各类数据相对平衡；其次是在过程中解决问题，改进现有算法，使其能适应不平衡数据的分类问题。本节分析三种处理方法。

3.6.1 采样算法

属于上述归纳的第一类解决方法。常用的采样方法有三类：欠采样、过采样和混合采样[83]。

1. 欠采样

该方法的原理是删除一些被充分采样的数据，而保持异常类数据的数量，从而使两类数据的分布差异缩小。欠采样方法有随机删除，有根据两个数据实例间的距离判断（例如 Tomek links 方法[84]），还有根据三个样本的邻近样本判断（例如 neighborhod cleaning rule,

NCL 方法[85])。但综其原理，都是删除一些数据点，这种做法可能使数据丢失一些重要信息，比如与时间相关的数据，如果采用此方法则会缺失数据的连贯性。

2. 过采样

与欠采样的理念类似，是使用方法对少量类别的数据进行处理，使其数据量增大，而不对数据量充足方做处理。常用方法包括简单的随机过采样方法、SMOTE 方法[86]等。

3. 混合采样

此方法是将前两种方法结合，使用欠采样中的方法处理充足方数据，使用过采样中的方法处理不充足方的数据，从而使两方数据分布差异缩小。

真实的网络流量领域不同于图像等领域，随机的对数据进行处理会导致重要的信息丢失，这一类相关方法并不适用于本文。

3.6.2 集成学习算法

属于本节起始归纳的两类方法的后者。集成学习是通过结合多个算法来构建学习任务，即通过针对不同问题训练多个分类器，多数情况下比一般的单一学习器具有更加优越的性能[87]。但也有研究表明，虽然集成方法的整体性能更好，但是单一的 NNs 可以更好的识别潜在攻击[88]。根据结合方式，可以将该算法分为两类：一类是学习器之间不存在依赖关系，使用并行化方式进行学习，其中的代表是随机森林算法；另一类是学习器之间存在较强的依赖关系，使用串行的序列化方式进行学习。集成学习算法的优点是即使高质量的训练样本量不足，即使分类器较弱，但是多个有针对性的分类器结合，也能够提高整体检测的精度和降低误报率。

集成学习的过程有三个阶段：集成生成、集成选择和集成整合。

集成生成：如果使用的分类器具有同一种算法，则属于同质的集成生成，反之是异质的，会产生一个不同的基分类池。

集成选择：针对不同的特征集从不同的基分类池中选择分类器。

集成整合：将集成选择阶段涉及的分类器整合为一组分类器。此阶段是集成学习算法的重点，每个分类器都对应不同的预测范围，有各自的优势。周志华实验室提出的 EasyEnsemble[89]是将充足方的数据划分为若干个集合，供不同的学习器进行学习，这样对于每个学习器，该数据都是进行了欠采样，而数据本身的信息并没有丢失。

对于异常检测领域，这一类方法仍然需要有部分异常数据，而异常数据通常依靠仿真软件生成，真实的攻击行为数据往往难以获取，高质量的异常数据获取仍然是挑战。

3.6.3 一类学习算法

一类分类，也称为离群值检测。该算法在没有反例的情况下仍然可以进行学习[90]。一类分类的目标是围绕目标对象（即网络异常检测中的正常流量）进行学习，以便位于此决策表面内的模式被分类为目标（即正常流量），而位于外部的模式被分类为异常值（即异常流量）。一类分类方法致力于得到“是”一类，不同于二分类得到“是”与“不是”两类。

对于网络流量，一般来说正常流量一定远多于异常流量，训练集数据的比例差距太大，异常识别效果可能差异会非常大。对于这类数据绝对不平衡的问题，一类分类方法在理论上适用。一类学习算法可分为三类，其中最具代表性的是一类 SVM(One-class SVM)[91]属于基于支持域的方法，通过找到一个超平面，最大化训练数据到特征空间原点之间的间隔，而后通过核函数对高维特征数据进行映射，但是这一过程需要较高的计算成本；还有一类是基于密度函数阈值的方法：指定一个密度阈值，对当前的数据进行评估，如果所确定的密度函数产生的阈值小于指定阈值，则判定为异常，但是密度函数在复杂数据的情况下难以确定；最后一类则是基于重构误差的方法。在本文的 2.4 节中已经讨论过基于重构误差的深度学习异常检测技术，它分为基于数值的重构和基于分布的重构方法。其中，基于重构分布误差的方法既适合不平衡数据集学习，又能学习高维数据的固有特征信息，并根据学习到的内在共性特征检测出异常点，适用于检测未知的攻击。正是适合本文思路的方法。

本文意图利用 GAN 能够对高维数据建模的优势，并且利用 GAN 的博弈思想，使模型在学习中自动调整阈值，充分学习出正常流量特征的精确分布。应用的具体模型将在下一章作出详细阐述。

3.7 本章小结

本章从网络流量行为入手，深入分析了用户日常行为的流量特点和不同阶段常见攻击行为的流量特点。其次，对现有入侵检测数据集做了简要概述，归纳分析了数据集的属性，指明了现有数据集的不足，主要阐述了从真实环境中采集的用户行为数据的生成方式和特征选取的标准，并且生成了本文的实验数据。最后，介绍不平衡数据的处理方法，选定本实验的建模方法。

第四章 用户流量行为基线建模

为了使检测模型能够易于进行未知攻击的检测，本章提出基于双向生成对抗网络的基线建模方法。通过学习正常样本，在过程中不断自动优化和学习，从而为不同的用户建立基线模型，使模型快速收敛。本章阐述了模型的原理、参数选择、训练的过程和基线模型的有效性测试，并且确定了适合此类用户的建模时长。

4.1 模型原理

本节的阐述从双向 GAN 模型的框架入手，然后是改进模型训练的方法，最后整合为本章建模所用的模型。

4.1.1 双向 GAN 框架

本节采用的模型基础是 2.4 节中介绍过的 GAN 模型，本文的模型是双向的 GAN[92]。在 GAN 的基础上增加了一个编码器(E)，用于将原始数据映射到隐层空间，使得模型既具有编码器的能力，又具有普通 GAN 的生成能力。同时，该模型对判别器(D)进行了改进，D 由两对数据作为输入$(\mathrm{G}(\mathrm{z}),\mathrm{z})$和$(\mathrm{x},\mathrm{E}(\mathrm{x}))$。其中，$\mathrm{E}(\mathrm{x})$是由原始数据$\mathrm{x}$经过编码得到，$\mathrm{G}(\mathrm{z})$是由某个任意分布采样的随机噪声$\mathrm{z}$经过解码（生成）得到。通俗来讲，隐变量$\mathrm{E}(\mathrm{x})$学习已知噪声$\mathrm{z}$，重构变量$\mathrm{G}(\mathrm{z})$学习已知原始数据$\mathrm{x}$，构成了一个双向学习的过程。由此，使 GAN 具备了一定的表征学习能力。模型强调 E 和 G 一起学习的重要性。

其中，D、E、G 是三个独立的 NNs 结构，它们之间的博弈可以理解为，E 和 G 希望能够欺骗 D，使 D 无法区分两个数据对的来源。如果 D 认为该数据对来自 G，则输出 1，如果来自 E，则输出 0。目标函数与 GAN 类似：

$$\min_{G,E}\max_{D}V(D,E,G) \tag{9}$$

其中$V(D,E,G)$数学表达如下：

$$V(D,E,G)=E_{x\sim px}[E_{z\sim pE(.|x)}[logD(x,z)]]\ +E_{z\sim pz}[E_{x\sim pG(.|x)}[log(1-D(x,z))]] \tag{10}$$

式中，$E_{z\sim pE(.|x)}[logD(x,z)]$相当于 $logD(x,E(x))$，$E_{x\sim pG(.|x)}[\log(1-D(x,z))]$相当于$\log(1-D(G(z),z))$。目标函数中使用交叉熵损失函数，用于测量重构样本和原始样本两个分布之间的差异。

E 的概率分布表示为：

$$q(x,z)=q(z|x)q(x) \tag{11}$$

G 的概率分布表示为：

$$p(x,z)=p(x|z)q(z) \tag{12}$$

原型 GAN 中最终衡量两个分布之间的距离等价于 JS 散度[93]：

$$\min_G \max_D V = -2log2 + \min_G [2JSD(P_{data}||P_G)] \quad (13)$$

通常在机器学习中，训练数据的分布是固定的，NNs 训练的目的就是使一个分布拟合另一个分布。但是，在网络流量领域不同于图像领域，流量数据不仅包含连续特征，还存在部分离散特征，离散特征在训练时需要经过编码处理，从而使维度扩展，当低维数据映射到高维空间，重构样本和原始样本之间几乎没有重叠。那么此时 JS 散度为常数$log2$，则梯度为 0，无法再使用梯度下降法使重构样本分布拟合原始样本分布。

4.1.2 训练改进

WGAN(Wasserstein GAN)[94]使用 EM(Earth-Mover)距离，使得两个分布在无重叠或重叠部分可忽略的情况下，仍然能够反映他们的远近。虽然研究者对其叫做 WGAN，但实际上和 GAN 的关系不大，只是解决了原型 GAN 训练时的梯度消失的问题。原理如下：

通常我们需要模型对扰动不敏感，说明模型的泛化能力强。针对于 W 距离，我们希望$\|x-y\|$很小时，$\|f_w(x)-f_w(y)\|$也能够很小。根据 Lipschitz 约束条件[95]，若限制 NNs 中f_w域内的每个点的梯度不超过[-c, c]这个范围，则下式恒成立：

$$\|f_w(x)-f_w(y)\| \le C(w)\|x-y\| \quad (14)$$

WGAN 可以训练生成该模型，其本质来自于线性规划与对偶。在规划与优化问题中，“对偶”是指某种变换，能将原问题转化为一个等价但是看起来似乎不同的新问题。当满足连续条件时，W 的距离可以转化为以下形式，即损失函数：

$$\max_{f,\|f\|_L \le 1} \{-E_{x\sim p_{2(x)}}[f_w(x)] + E_{x\sim p_{1(x)}}[f_w(x)]\} \quad (15)$$

由此，WGAN 的训练过程为：

$$\min_G \max_{f,\|f\|_L \le 1} \{-E_{x\sim p_x}[f_w(x)] + E_{z\sim p_z}[f_w(G(z))]\} \quad (16)$$

但是由于 WGAN 采用权重修剪(Weight clipping)强行满足了 Lipschitz 约束，限制着每个网络参数的取值范围，因此在训练过程中很容易导致梯度爆炸。

Gulrajani 等人[96]将上述 WGAN 的约束改进为梯度剪切(gradient penalty, GP)，将其更换为一个惩罚项来限制梯度的值不超过 K，GP 仅在测量重构样本和原始样本两个分布之间进行随机采样处理，应用在目标函数的最后一项。

WGAN-GP 的表达式如下：

$$L(D) = -E_{x\sim p_x}[D(x)] + E_{\hat{x}\sim p_z}[D(\hat{x})] + \lambda E_{\hat{x}\sim P_{\hat{x}}}[\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1]^2 \quad (17)$$

相当于 WGAN 增加了梯度约束正则项，K 恒定为 1。该方法同时也提高了模型训练的收敛速度。

4.1.3 模型整合

本文中的模型是把 WGAN-GP 应用在双向 GAN 框架中，代替了原有模型中 D 的损失函数，使模型训练和检测保持平稳高效。

目标函数的数学表达式如下：

$$L(D) = -E_{x\sim p_x}[D(x,E(x))] + E_{z\sim pz}[D(G(z),z)] + \lambda E_{\hat{x}\sim P_{\hat{x}}}[\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1]^2 \quad (18)$$

其中，损失函数有三个：

原始数据与隐变量的分布差异：

$$Loss_e = -E_{x\sim px}[D(x,E(x))] \quad (19)$$

已知噪声与重构变量的分布差异：

$$Loss_g = E_{z\sim pz}[D(G(z),z)] \quad (20)$$

重构样本与原始样本的分布差异：

$$Loss_d = Loss_e + Loss_g \quad (21)$$

模型框架如图 42 所示。

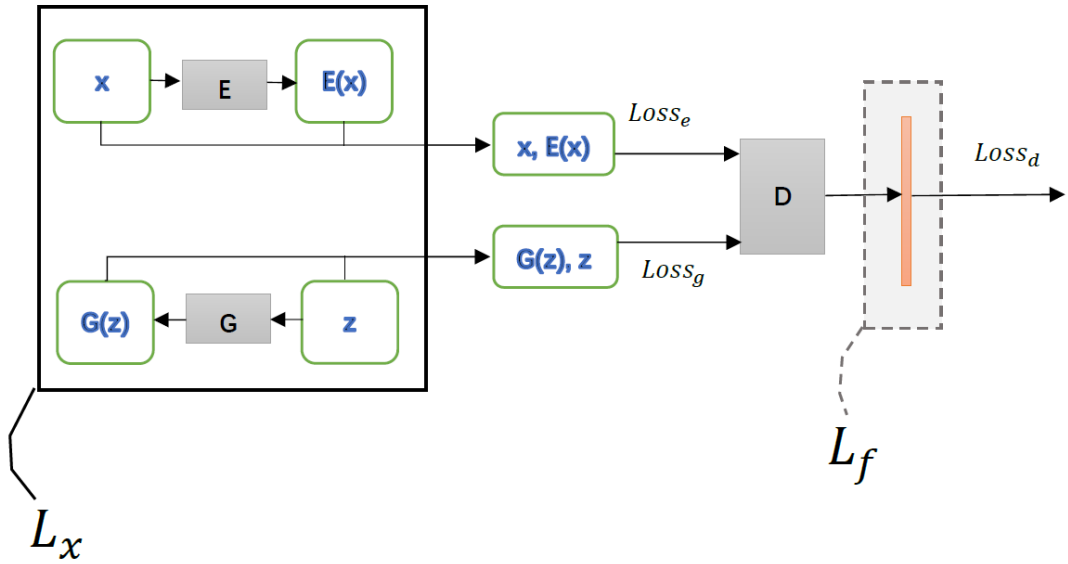

图 42 模型框架与参数图

目标函数部分的算法如下所示：

```
loss_d = loss_dis_enc + loss_dis_gen
loss_g = -tf.reduce_mean(l_generator)
loss_e = tf.reduce_mean(l_encoder)
```

```
alpha = tf.random_uniform(shape=[batch_size, 1],minval=0.,maxval=1.)
differences = x_gen - input_pl
interpolates = input_pl + (alpha * differences)
gradients = (tf.gradients(dis(z,interpolates,reuse=tf.AUTO_REUSE), [interpolates])[0] +
        tf.gradients(dis(z_gen,interpolates,reuse=tf.AUTO_REUSE),[interpolates])[0] ) / 2
slopes = tf.sqrt(tf.reduce_sum(tf.square(gradients), reduction_indices=[1]))
gradient_penalty = tf.reduce_mean((slopes - 1.) ** 2)
loss_discriminator += LAMBDA * gradient_penalty
```

4.2 参数选择

4.2.1 激活函数

为了使模型学习更复杂的目标函数，NNs 的隐层通常选用经过非线性函数变化后得出输出。本节模型所使用的激活函数是修正线性单元(Rectified linear unit, ReLU)函数及其变体，如图 43 所示。

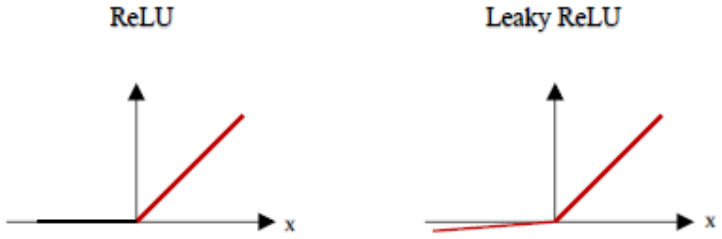

图 43 ReLU 和 Leaky ReLU 激活函数

1. ReLU

只有超出阈值时神经元才激活。神经元的输出为：

$$\max(0, w * x + b) \tag{22}$$

其中，x为输入，w为权重向量，b为偏置。

从图像上看，函数为：

$$f(x) = \begin{cases} 0, (x \leq 0) \\ x, (x > 0) \end{cases} \tag{23}$$

当$x > 0$时，其导数为常数，意味着在该区间内后向传播的算法梯度更新不为 0，即不会梯度消失，使得模型可以保持稳定速度收敛。由于 ReLU 函数不存在复杂的数学运算，计算高效简单，它的收敛速度比 Sigmoid 和 Tanh 快[97]。当$x \leq 0$时，函数值为 0，即在该区间内随着训练过程进行，权重不更新，学习速度缓慢，甚至使神经元无效从而一直保持静默。

在本文模型中，G 训练使用 ReLU。

2. Leaky ReLU

是 ReLU 的带泄露版本，为了解决x取负时参数不更新的问题，引入$leak$（取值为小数），使在取值小于 0 的区间有一个很小的梯度。但也因此使其非线性程度弱于 ReLU，使用时需要因势而异。

数学表达式为：

$$f'(x)=\begin{cases}leak*x,(x\leq 0)\\ x,(x>0)\end{cases} \tag{24}$$

在本文模型中，E 和 D 训练使用 Leaky ReLU。

4.2.2 损失函数

损失函数的目的是度量 NNs 输出的预测值与实际值之间的差异。从传统的判别式网络的思路出发，选取合适的损失函数可以使生成分布和原始分布之间的距离尽可能近。上一节中已经阐述 GAN 原型中使用交叉熵损失函数，等价于 JS 散度。但是由于 GAN 的训练特殊性，本文使用 W(Wasserstrein)距离作为 D 的损失函数。

W 距离是计算两个概率分布的最优传输距离。采样形式如下：

$$W(P_1,P_2)=\inf_{\gamma\sim\pi(P_1,P_2)}E_{(x,y)\sim\gamma}[\|x-y\|] \tag{25}$$

其中，$\gamma\sim\pi(P_1,P_2)$表示 γ的一个联合分布，其边缘是P_1和P_2，可以理解为γ代表了一种运动方案，从原始分布P_1到目标分布P_2；$\|x-y\|$是一个成本函数，代表从 x 运输到 y 的成本；inf是确定下界的符号，简单来说就是取最小。因此，W 距离是在求满足约束的最省力的“推土”距离，又称作推土机距离(Earth Mover’s Distance)。

4.2.3 Dropout

数据的训练误差越小说明拟合效果越好，但是在训练过程中，会发生随着训练集误差越来越低，测试集误差却越来越高，这意味着产生了过拟合。Hinton 提出了 Dropout[98]技术，在 NNs 训练过程中按着一定概率使隐层部分神经元停止工作，从而减少隐层节点间的依赖，使模型泛化性能提升。

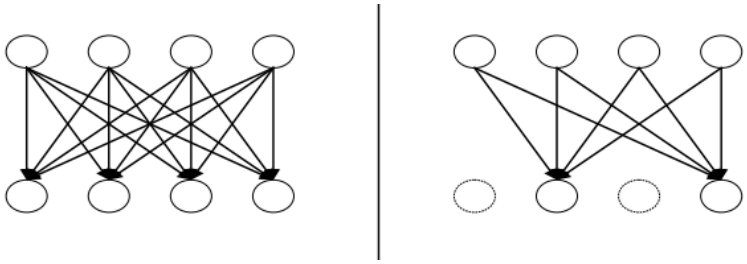

图 44 Dropout 工作原理

图 44 中所示训练模型，左图没有 Dropout，右图是添加 Dropout。

标准的 NNs 传播公式：

$$y_i^{l+1} = w_i^{l+1}x^l + b_i^{l+1}$$
$$x_i^{l+1} = f(y_i^{l+1}) \quad (26)$$

输入x输出是y，先把y通过网络前向传播，而后把误差反向传播决定如何更新参数。加入 Dropout 后的传播公式：

$$r_j^l \sim Bernoulli(p)$$
$$\widetilde{x^l} = r^l * x^l$$
$$y_i^{l+1} = w_i^{l+1}\widetilde{x^l} + b_i^{l+1} \quad (27)$$
$$x_i^{l+1} = f(y_i^{l+1})$$

首先，经过$Bernoulli$函数，生成概率向量（0 和 1），即随机丢弃了部分隐层节点，输入和输出保持不变。而后，x通过改变后的网络前向传播，再把误差反向传播，完成这一过程。恢复被删掉的神经元，按着梯度下降法更新参数w和b。

Dropout 之所以可以解决过拟合问题，因为每次随机丢弃神经元都产生了不同的网络结构，权值更新不再依赖于固有的隐层节点共同作用，阻止了某些特征仅仅在特定情况下才有效果，因而迫使网络学习更加鲁棒。

本模型在训练 D 时采用 dropout 方式。

4.2.4 优化函数

本模型选用 Adam(Adaptive Moment Estimatio)[99]作为优化函数，与传统的随机梯度下降优化算法不同，Adam 不是保持单一学习率，而是自适应学习率。该算法计算了梯度的指数移动均值(exponential moving average, EMA)，通过学习率(alpha)、一阶矩阵估计的指数衰减率(beta1)和二阶矩阵估计的指数衰减率(beta2)来控制 EMA 的衰减率。其中，beta 推荐值为接近于 1。因此，矩估计的偏差接近于 0，首先计算带偏差的估计，而后计算偏差修正后的估计而得到优化。

该方法计算效率高，所需内存小，由于其算法梯度的对角缩放不变性，因此非常适合解决大规模数据和参数的优化问题。

4.3 模型训练

本文模型中应用最多的层是全连接层(fully connected layers, FC)，它是最基本的 NNs 层。FC 是将网络中的神经元与邻层的每个神经元连接，其输入可视为网络从数据中提取到的特征，从几何角度理解，一个特征对应多维空间中的一个点。

本节模型先训练 D，后训练 G 和 E，G 和 E 是同步训练的，而 D 和它们是交替训练的。

4.3.1 训练判别器

D 采用 NNs 结构，它的训练过程可理解为，z和$E(x)$在隐层空间中，$G(z)$和x在真实空间中，训练过程开始后，初始隐变量z和训练样本（原始数据）x分别映射为$G(z)$和$E(x)$，然后分别各自合成对(G(z)，z)和$(x,E(x))$送入 D 中，最后将梯度更新传回以优化 G 和 E。训练过程如图 45 所示。

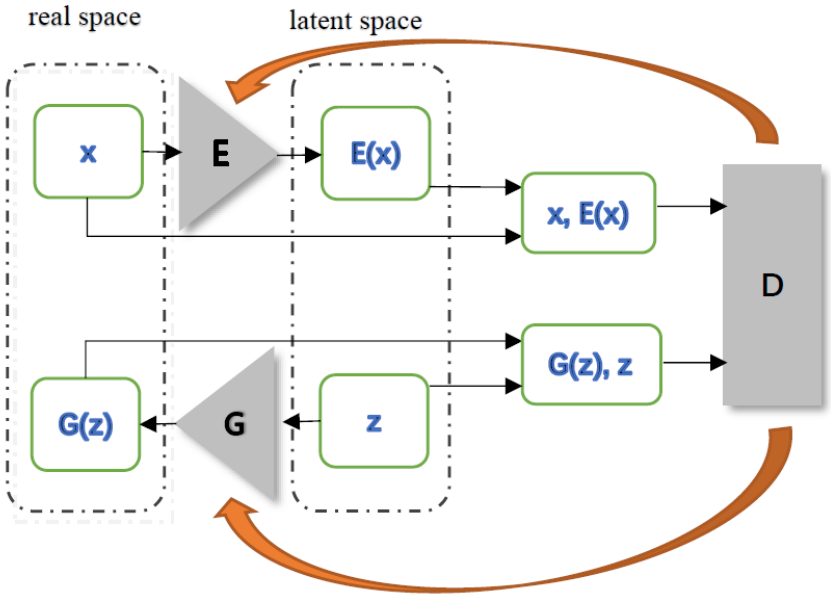

图 45 D 的训练过程

D 的输入来自三个部分：

（1）来自真实空间，训练样本输入 57 维数据（由 54 维特征组经过预处理得来，数据预处理过程将在 4.4.1 节中作出详细阐述），经过第一层 FC 输出 64 维数据，采用 Leaky ReLU 激活，Dropout 设置为 0.2，输出为 64 维原始数据特征；

（2）来自隐层空间，输入为 16 维数据，经过第一层 FC 输出 64 维数据，采用 Leaky ReLU 激活，Dropout 设置为 0.2，输出为 64 维来自隐层变量的特征；

（3）判别部分，（1）中x与其隐层映射拼接，（2）中 z 与其真实空间特征映射拼接，两对数据输入 D，经过第一层 FC 输出 64 维数据，采用 Leaky ReLU 激活，Dropout 设置为 0.2，再经过 1 层 FC，输出 1 维数据，即最后一层 FC 将每一个神经元连接到每一个输出神经元。此过程利用局部相关性对特征进行抽取，减少数据处理量，同时保留有用的特征。最后，输出得到隐层的特征匹配和$D(E，G)$的损失值。

D 的模型结构和参数如表 22 所示。

表 22 D 的模型结构和参数

Layer (type)	Output Shape	Param
Input (None, 57)		
x_layer_1 (Dense)	(None, 64)	3712
LeakyReLU_1 (Activation)	(None, 64)	0
dropout_1 (Dropout)	(None, 64)	0
Input (None, 16)		
z_layer_1 (Dense)	(None, 64)	1088
LeakyReLU_2 (Activation)	(None, 64)	0
dropout_2 (Dropout)	(None, 64)	0
Concat [x, z]		
y_layer_1 (Dense)	(None, 64)	4160
LeakyReLU_3 (Activation)	(None, 64)	0
dropout_3 (Dropout)	(None, 64)	0
y_layer_2 (Dense)	(None, 1)	65
Total params: 9025		
Trainable params: 9025		
Non-trainable params: 0		

判别器部分算法如下所示：

```
with tf.variable_scope('discriminator', reuse=reuse, custom_getter=getter):
    name_x = 'x_layer_1'
    with tf.variable_scope(name_x):
        x = tf.layers.dense(x_inp,
                            units=64,
                            kernel_initializer=init_kernel,
                            name='fc')
        x = leakyReLu(x)
        x = tf.layers.dropout(x, rate=0.2, name='dropout', training=is_training)

    name_z = 'z_layer_1'
    with tf.variable_scope(name_z):
        z = tf.layers.dense(z_inp, 64, kernel_initializer=init_kernel)
        z = leakyReLu(z)
        z = tf.layers.dropout(z, rate=0.2, name='dropout', training=is_training)
```

```
        y = tf.concat([x, z], axis=1)
        name_y = 'y_layer_1'
        with tf.variable_scope(name_y):
            y = tf.layers.dense(y,
                                dis_inter_layer_dim,
                                kernel_initializer=init_kernel)
            y = leakyReLu(y)
            y = tf.layers.dropout(y, rate=0.2, name='dropout', training=is_training)
        inter_layer = y

        name_y = 'y_layer_2'
        with tf.variable_scope(name_y):
            y_2 = tf.layers.dense(y,
                                  1,
                                  kernel_initializer=init_kernel)
    return y_2, inter_layer
```

4.3.2 训练生成器和编码器

GAN 的 G 和 E 均使用 DNNs 结构，它们的同步训练可以理解为，当 G 学习从隐层空间到真实空间的映射时，E 会同时学习从真实空间到隐层空间的逆映射。

G 采用 4 层 NNs。输入是 16 维的隐空间变量，经过第一层 FC 输出 32 维数据，采用 ReLU 激活，经过第二层 FC 输出 64 维数据，再次采用 ReLU 激活，经过第三层 FC 输出 57 维，得到真实空间特征表示，尝试重构原始输入数据，即$G(E(x))$。

G 的模型结构和参数如表 23 所示。

表 23 G 的模型结构和参数

Layer (type)	Output Shape	Param
Input (None, 16)		
layer_1 (Dense)	(None, 32)	544
ReLU_1 (Activation)	(None, 32)	0
layer_2 (Dense)	(None, 64)	2112
ReLU_2 (Activation)	(None, 64)	0
layer_3 (Dense)	(None, 57)	3705
Total params: 6,361		
Trainable params: 6,361		
Non-trainable params: 0		

E 采用 3 层 NNs。输入是 57 维原始数据，经过第一层 FC 输出 32 维数据，采用 Leaky ReLU 激活，经过第二层 FC 输出 16 维数据，输出隐层特征表示，即$E(x)$。

E 的模型结构和参数如表 24 所示。

表 24 E 的模型结构和参数

Layer (type)	Output Shape	Param
Input (None, 57)		
layer_1 (Dense)	(None, 32)	1856
LeakyReLU_1 (Activation)	(None, 32)	0
layer_2 (Dense)	(None, 16)	528
Total params: 2,384		
Trainable params: 2,384		
Non-trainable params: 0		

4.4 基线建立实验

本节先将数据中的连续型特征和离散型特征预处理为符合模型学习的形式，然后通过实验进行基线模型的有效性测试，并分析实验结果，从而判定基线建模方法的可行性，并选定适合此类用户的建模时长，最后将不同用户的基线模型保存，用于检测异常。

4.4.1 数据预处理

在网络流量领域，数据的预处理包括删除随机偏离的数据、对数据进行规范化。数据预处理对于深度学习来说是个非常重要的步骤，模型训练的好坏与训练数据的质量是成正比的。数据预处理的流程如图 46 所示。

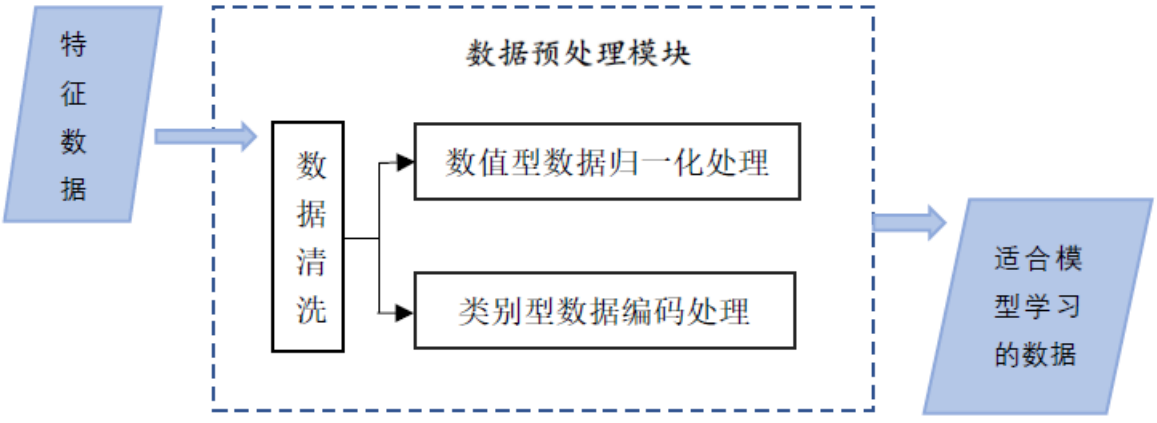

图 46 数据预处理模快

1. 数据清洗

在数据可识别的情况下，该步骤是处理错误的最后一道程序，包括处理数据的格式错误、缺失值和无效值等。其原理是利用数理统计、数据挖掘、预定义清理规则等相关技术，把影响模型学习的数据转化为符合要求的数据。

处理的方法主要是估算和删除，当某个极少数的变量出现缺失和无效值时，可以使用均值、中位数等统计计算值替换，确保不会因个别值影响整体数据质量；当某些变量、列或是整行出现缺失值和无效值时，可以视情况做出删除处理。

2. 数据预处理

此步骤针对连续型的特征做归一化处理，针对离散型特征做编码处理，通常会使模型具有较强的非线性能力。

（1）归一化

数据的归一化是将每一列数值缩放到相同范围，每一列即每一维特征。对数值特征进行归一化处理，可以避免由于某个特征的数值太大，造成其占据学习算法中的主导位置。归一化之后，不同维度之间的特征在数值上可以具有比较性，同时还可以加快模型的学习速度。

本文采用 Min-Max 归一化(Min-Max Normalization)，是默认将每一维特征线性映射为[0,1]的数值，变换不会改变数据的排序。

数学表达是为：

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (28)$$

这种缩放仅与最大值和最小值相关，实际上 min 和 max 都是可以指定的。在某些零值进行转换后有可能变为非零值，归一化的上下界可以调整到指定范围。

数值型数据归一化方法：

```
scaler = preprocessing.MinMaxScaler()
```

数据归一化后部分数据效果如图 47 所示：

```
0.28571429 0.33333333 0.66666667 0.13615561
0.         0.66666667 0.5        0.
0.14285714 1.         0.16666667 0.43020595
1.         0.66666667 0.         1.
```

图 47 归一化效果

在 NNs 中，归一化和标准化都能够解决数值问题，区别在于标准化的缩放和每一个点都有关系，通过方差体现出来，与归一化相比，所有数据点都有贡献，且标准化的输出范围是$(-\infty,+\infty)$。本文采用归一化而非标准化是由于本文的数值特征已经选用标准差，数据较为稳定，而且在本实验中归一化方法的效果优于标准化方法。

（2）哑变量(Dummy)编码

数据中的类别特征需要进行预处理，比如本文中的目的端口类型和协议类型。编码是将离散型每一种特征任都看成一种状态。本文采用 Dummy 编码方式，是任意去除一个状态，即如果有四种状态，则需要三个状态位，其中三种状态在激活时状态位值为 1，而第四种状态可以用[0,0,0]表示。独热(One-hot)编码方式与 Dummy 在理念上相似，但不同之处在于：One-hot 采用 N 位状态寄存器来对 N 个状态编码，每个状态有其独立的寄存器位，任意时候只有 1 位有效，这使得采用 One-hot 处理的数据总是比 Dummy 处理的数据在维度上多出 N 个类别的维度。在 NNs 学习中维度越高对模型的要求便越高，加之本文的类别属于无序的分类，更适合使用 Dummy 编码。

本文将目的端口号按照其基本分类划分为动态端口（0-1023）和静态端口（1024-65535），而协议类别分为 TCP、UDP 和其他协议。编码处理后的数据由 54 维扩展至 57 维。离散特征的编码效果如图 48 所示。

Protocol-0	Protocol-6	Protocol-17	Dst Port Type-Dynamic	Dst Port Type-Well-Known
0	1	0	1	0
0	1	0	0	1
0	1	0	0	1
0	1	0	0	1
0	1	0	1	0

图 48 Dummy 编码效果

C. 数据划分

数据划分，将正常数据标记为 0，攻击数据标记为 1。在建立基线模型时仅使用用户的正常流量数据，而在测试异常检测性能的时候需要区分正常和异常数据。

数据的划分方式如下：

```
labels[labels != 'Normal'] = 1
labels[labels == 'Normal'] = 0
```

4.4.2 基线时长的选取实验

根据上述训练过程，采用 4.3 节所述生成的正常流量数据，先为用户 A 建立基线，用以确定相对较好的建立基线模型的时长。该实验的训练集使用前 28 天的流量数据，分别选取 7 天、14 天、21 天和 28 天的流量数据作为训练数据。测试基线性能的流量数据由两部分组成，分别为从 CICIDS2017 数据集中提取的攻击流量和该用户其余时间的正常流量。从正常数据中随机抽取 144 条，从攻击数据的 7 类攻击中分别抽取 36 条，抽取的每类攻击都包含数据集中的所有攻击方式。

测试所用攻击样本数目详情如表 25 所示。

表 25 测试攻击样本

攻击类型	攻击详情	攻击流数	攻击总流数
Patator	FTP-Patator	20	36
	SSH-Patator	16	
DDoS	DDoS	36	36
Web-attack	Sql Injection	28	36
	Brute force	7	
	XSS	1	
Infiltration	Infiltration	36	36
Botnet	Botnet	36	36
DoS	Hulk	30	36
	GoldenEye	3	
	Slowloris	1	
	Slowhttptest	1	
	Heartbleed	1	
Port-scan	Port-scan	36	36

4.4.3 实验结果与分析

实验得到了四种不同量级建立的基线模型对每种少量攻击的检测效果。对于用户 A，7 天的行为数据学习效果好过 14 天、21 天和 28 天，这与我们预期的结果有些差异。不同时长对不同攻击检测效果如表 26 至表 32 所示。

表 26 对 DoS 检测效果

	Precision(%)	Recall(%)	F1(%)	Accuracy(%)	建模时长(s)
7 天	100	100	100	100	533
14 天	72.22	72.22	72.22	88.89	1085
21 天	72.22	72.22	72.22	88.89	1569
28 天	69.44	69.44	69.44	87.78	1798

表 27 对 DDoS 检测效果

	Precision(%)	Recall(%)	F1(%)	Accuracy(%)	建模时长(s)
7 天	100	100	100	100	533
14 天	83.33	83.33	83.33	93.33	1085
21 天	83.33	83.33	83.33	93.33	1569
28 天	63.89	63.89	63.89	85.56	1798

表 28 对 Web-attack 检测效果

	Precision(%)	Recall(%)	F1(%)	Accuracy(%)	建模时长(s)
7 天	91.67	91.67	91.67	96.67	533
14 天	89.19	91.67	90.41	96.11	1085
21 天	91.67	91.67	91.67	96.67	1569
28 天	63.89	63.89	63.89	85.56	1798

表 29 对 Infiltration 检测效果

	Precision(%)	Recall(%)	F1(%)	Accuracy(%)	建模时长(s)
7 天	100	100	100	100	533
14 天	91.67	91.67	91.67	96.67	1085
21 天	88.89	88.89	88.89	95.56	1569
28 天	88.89	88.89	88.89	95.56	1798

表 30 对 Bot 检测效果

	Precision(%)	Recall(%)	F1(%)	Accuracy(%)	建模时长(s)
7 天	58.33	58.33	58.33	83.33	533
14 天	58.33	58.33	58.33	83.33	1085
21 天	52.78	52.78	52.78	81.11	1569
28 天	30.56	30.56	30.56	72.22	1798

表 31 对 Port-scan 检测效果

	Precision(%)	Recall(%)	F1(%)	Accuracy(%)	建模时长(s)
7 天	80.56	80.56	80.56	92.22	533
14 天	61.11	61.11	61.11	84.44	1085
21 天	22.22	22.22	22.22	68.89	1569
28 天	0	0	0	0	1798

表 32 对 Patator 检测效果

	Precision(%)	Recall(%)	F1(%)	Accuracy(%)	建模时长(s)
7 天	80.56	80.56	80.56	92.22	533
14 天	52.78	52.78	52.78	81.11	1085
21 天	80.56	80.56	80.56	92.22	1569
28 天	52.78	52.78	52.78	81.11	1798

针对建模时长选取实验，出现以上结果应该是由于用户在四周这段较长时间内的行为出现一定程度的改变，流量采集的时间跨越了学期的常规作息到假期的加班作息。本实验的最初设想是保护相对重要的实体设备，这类设备一般会有较稳定的操作或被访问的行为流量。对于本实验中的用户，一周时间是能够捕获较全面且较稳定行为流量的最佳选择。

本章的实验以选取建模时长为目的，没有对模型进行过多次训练和调参，统一进行了 10 轮迭代。由于数据量非常大，模型学习不充分，导致在 Port-scan 检测中，28 天建立的基线模型未达到检测效果。但是继续学习，在进行了 20 轮迭代后，模型效果才得以显现。综上实验结果，在下一章的实验中均选择由 7 天流量建立基线，即星期一至星期日的正常流量。

4.4.4 基线模型保存

在实际网络环境下，用户的基线模型在一定时期内是有效的，训练完毕后保存在相应的位置，在检测或进行模型更新时，方便重新加载模型的网络结构和相关参数。如图 49 所示，模型保存为四类文件：checkpoint 文件中存储了模型的文件目录的结构；包含“ckpt”的文件存储了模型的结构信息和参数；“.pbtxt”存储了目标检测与实例分割的配置文件；而多个“events. out. tfevents+编号”文件是模型的可视化信息，可以使用 TensorBoard 进行查看。查看命令如下：

```
程序根目录> tensorboard --logdir=model.training_logdir() --port=number
```

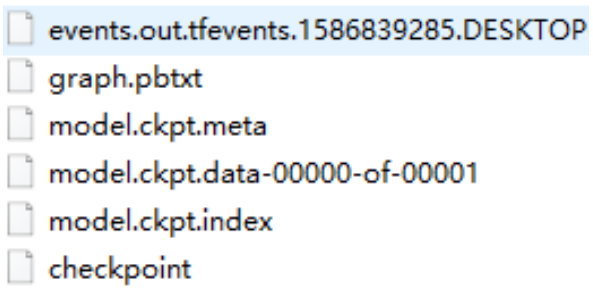

图 49 基线模型存储文件

4.5 本章小结

本章提出了基于双向生成对抗网络的基线建模方法。模型通过学习正常样本，能够在过程中不断自动优化和学习，从而为不同的用户建立了基线模型，经验证，模型收敛速度较快。本章主要阐述了模型的原理、参数选择、训练的过程和基线模型的有效性测试，并且确定了适合此类用户的建模时长。

第五章 系统构建与验证

本章对检测系统进行构建，首先，将各模块进行整合，介绍系统架构，按照流量进入系统的顺序，依次为流量采集、特征提取、数据预处理，基线建模和检测进行阐述，在 5.1 节中详细阐述了检测算法的原理。然后，通过实验验证系统的性能。最后，对构建的系统进行评价。

5.1 检测系统架构

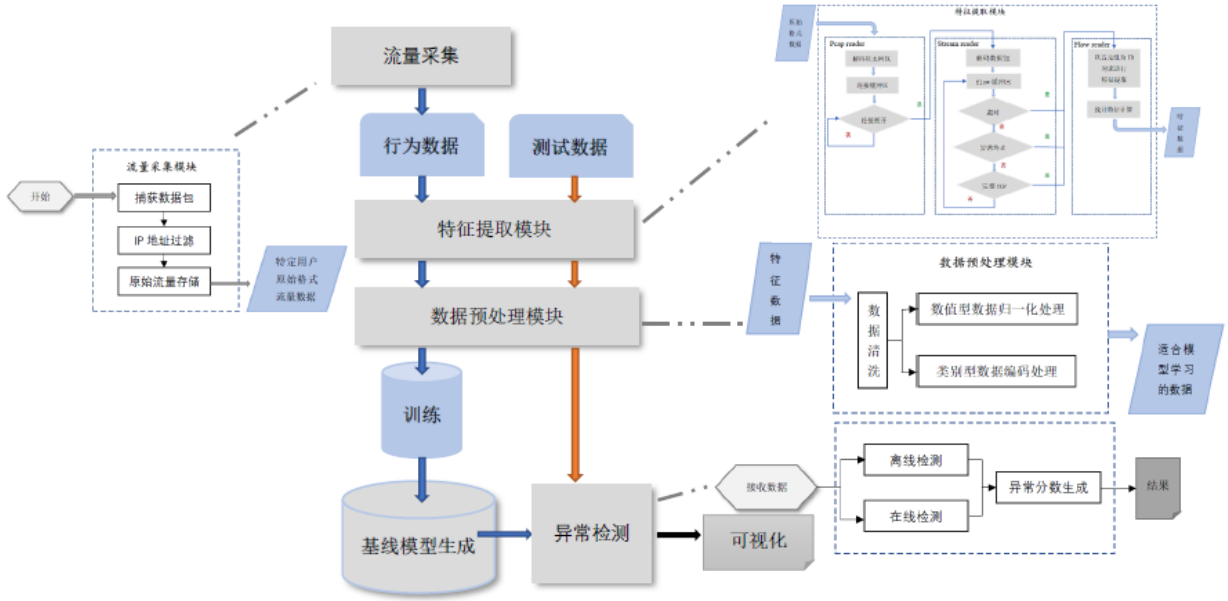

图 50 检测系统架构

如图 50 所示，本文设计的基于用户实体行为分析的异常检测系统包含五个主要模块：数据采集模块、特征提取模块、数据预处理模块、基线学习模块和异常检测模块。

1. 数据采集模块

该模块是检测模型的基础，通过预定的方式采集特定用户的网络流量数据包，并且对采集的数据进行 IP 地址过滤，过滤掉与该用户没有交互的 IP 地址流量，然后以原始数据的 pcap 格式存储。

2. 特征提取模块

该模块对过滤后的原始流量进行分析和检查。首先，以五元组进行会话的划分，丢弃流量中的乱序和重传数据包，提取完整的 TCP 会话，按超时时间划分其他协议会话，以及被异常终止的会话。然后，根据预设的特征字段对元数据进行提取。最后，计算特征组中的统计特征，并送入特征预处理模块。

3. 数据处理模块

在该模块中，对输入的数据进行预处理，包含对计算的缺失值等数据进行清洗。然后，将特征提取模块输出的 54 维特征组，按照数据的连续或离散类型进行归一化或编码处理，使其适合模型学习。

4. 基线模型学习模块

该模块将处理后的特定用户行为数据全部送入模型进行学习，学习时长依据数据量大小而定，对于普通用户，一星期数据的建模时长在 8 分钟左右。

以上四个模块均在第四章中做出了详细介绍。本章主要介绍异常检测模块。

系统整体的工作流程概括如图 51 所示。

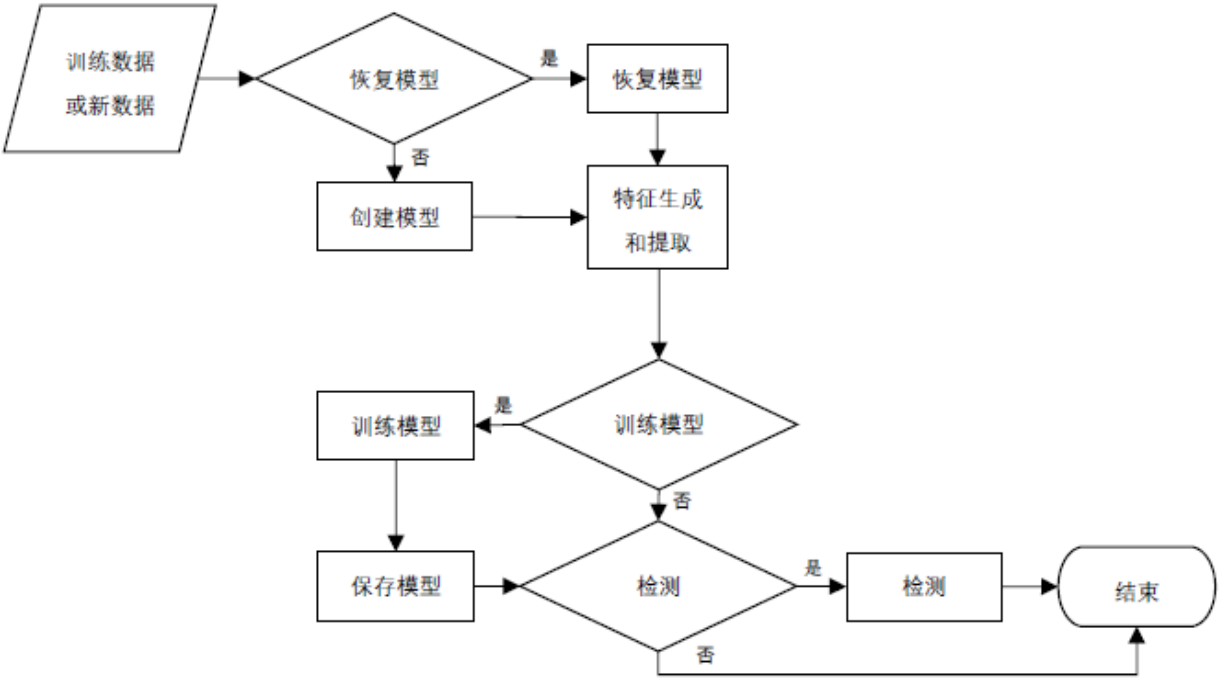

图 51 系统工作流程

5. 异常检测模块

该模块中汇入训练好的基线模型和处理好的检测数据，对新入样本进行异常检测。模型的基线学习是离线进行的，而检测可以分为离线与在线两种，将在下一小节中详细阐述。图 52 为异常检测模块的构造。

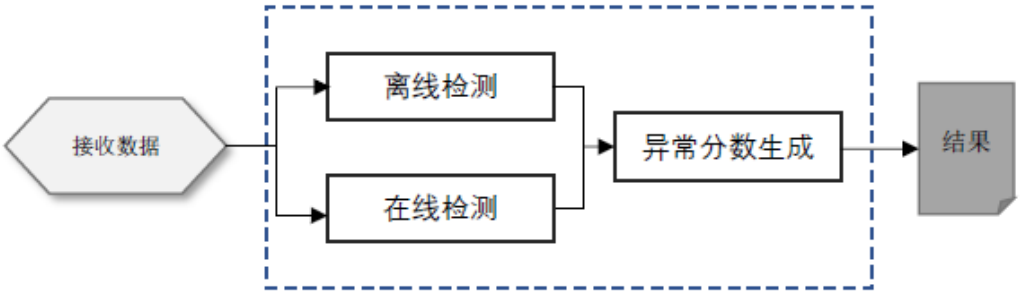

图 52 异常检测模块

5.1.1 检测算法

1. 离线检测

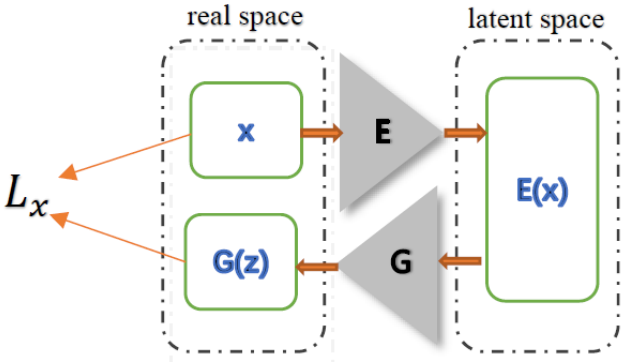

图 53 L_x的由来

在 4.3 节中所述，模型的训练过程得到两个损失函数，其一是 G 和 E 部分的损失函数，计算x经过编码再经过解码得到$G(E(x))$的 loss，用来评估重构样本是否相似，反应了异常部分和异常程度，记为L_x，重构过程如图 53 所示。另一个是 D 部分的损失函数，计算这一过程中隐层特征匹配的 loss，用来评估重构样本的特征是否相似，反应了置信水平，记为L_f。详见公式(29)和(30)：

$$L_x = |\mathrm{x} - \mathrm{G}(\mathrm{E}(\mathrm{x}))| \tag{29}$$

$$L_f = |fD(x, E(x)) - fD(E(x), G(E(x)))| \tag{30}$$

分别计算L_x和L_f的 L1 范数，即损失得分S_x和S_f。相较于 L2 范数，L1 范数具有稀疏性，有内置的特征自动选择性质，计算效率高[100]。

异常分数为：

$$Score = (1 - weight) * S_x + weight * S_f \tag{31}$$

Score 越小，表示样本间越是近似同分布。实验中攻击样本的恶意率占测试集的 20%，使用百分位公式计算分位值得到样本分布的位置分数。当测试样本大于这个值，则为被判定为异常。

2. 在线检测

区别仅在于定义 D 部分的损失函数计算，此处计算$D\left(E(x), G\left(E(x)\right)\right)$的交叉熵损失值作为置信度。在此选择交叉熵作为损失函数的好处是，使用 Sigmoid 函数可以避免在梯度下降过程中，均方误差损失函数的学习速率降低问题，学习速率可以被输出的误差所控制。

计算公式如下：

$$L_D = \sigma(D\left(E(x), G\left(E(x)\right)\right), 1) \tag{32}$$

交叉熵损失函数$\sigma(0,1)$，0 代表为正，1 代表为负。当 D$\left(E(x), G\left(E(x)\right)\right)$相似，即测试样本越靠近原始样本。

在线检测方法的其他部分与离线检测方法相同，使用(29)(32)(31)公式组合。通过学习大量的用户正常流量在训练中得到异常分数，获取合适的阈值。在检测中根据得到的异常评分，小于阈值的判定为正常样本，否则判定为异常。

检测算法部分代码实现：

```
    with tf.variable_scope('l_x'):
        delta = input_pl - reconstruct_ema
        s_x = tf.norm(delta_flat, ord=degree, axis=1, keep_dims=False, name='loss_g')
    with tf.variable_scope('l_d'):
        if method == "cross-e":                    #online
            s_f =tf.nn.sigmoid_cross_entropy_with_logits
                (labels=tf.ones_like(l_generator_ema), logits=l_generator_ema)
        elif method == "fm":                       #offline
            fm = inter_layer_inp_ema - inter_layer_rct_ema
            fm = tf.contrib.layers.flatten(fm)
            s_f = tf.norm(fm, ord=degree, axis=1, keep_dims=False, name='loss_d')
……
    with tf.variable_scope('Score'):
            list_scores = (1 - weight) * s_x + weight * s_f
```

5.1.2 检测日志

检测日志可以日后用来进行模型和数据分析，模型检测时的参数自动保存的间隔设置为 120s 一次，存储的目录以测试时选取的权重、检测方法和随机数创建，即<weight, method, random_seed>。

5.2 实验验证

本节从实验环境、实验方案、评价指标、结果分析、可视化和实验对比几个方面，对验证实验进行全面阐述。

5.2.1 实验环境

本章的实验部署在 Windows 10 64 位操作系统笔记本上，i7-8650U CPU 处理器，16GB 内存，TensorFlow 版本为 v1，Python 版本为 3.7。

5.2.2 实验方案

4.4 节中验证得出，7 天是比较适合用户建模的时长，本节使用 7 天的流量数据分别为用户 A、用户 B 和用户 C 建立行为基线模型。而后，使用与 4.4 节中相同的攻击数据组成测试数据，测试数据中的正常数据同样是从用户 A、B、C 建模所用数据以外的数据中抽取

144 条，与从攻击数据中抽取的 7 类各 36 条一起作为测试样本。本节实验的目的是验证不同用户的基线模型对攻击检测的有效性。

5.2.3 评价指标

IDS 中的分类指标很多，并且这些指标在研究中有很多种名称。本文研究的二分类评估指标，通常使用表 33 所示的混淆矩阵。矩阵中的行代表预测中的实例，列代表实际中的实例。

表 33 IDS 的混淆矩阵表

	实际正例	**实际负例**	
预测正例	预测正实际为正 TP	预测正实际为负 FP	预测正例总数 TP+FP
预测负例	预测负实际为正 FN	预测负实际为负 TN	预测负例总数 FN+TN
	实际正例总数 TP+FN	实际负例总数 FP+TN	

真正率(True Positive Rate, TPR)，又可以称为检出率(Detection Rate, DR)、召回率(Recall)或灵敏度：代表正确预测的异常次数与异常总数之比。即，如果所有异常都被检测到，则 TPR=1。但是，这种情况比较少见。数学表示为：

$$Recall = TPR = \frac{TP}{TP + FN} \quad (33)$$

假正率(False Positive Rate, FPR)，又称为误报率：代表误划分为异常的正常实例与正常实例总数之比。数学表示为：

$$FPR = 1 - TPR = \frac{FP}{FP + TN} \quad (34)$$

准确率(Accuracy)：是所有正确预测的实例占所有实例的百分比。虽然在异常检测中，由于异常样本相对于正常样本的数据极少，数据不平衡，此指标通常不能客观表示性能的好坏。但是在设计测试集的异常样本占比 20%的情况下，也有参考价值。数学表示为：

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (35)$$

精确率(Precision)数学表示为：

$$Precision = \frac{TP}{TP + FP} \quad (36)$$

F 加权调和均值(F-score)，通常 Precision 和 Recall 同时使用，取二者权重相等，即 F1-score。数学表示为：

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \quad (37)$$

本文第五章和第六章的实验均由 Precision、Recall、Accuracy、F1-score 这四个指标进行评估。

5.2.4 实验结果与分析

如表 34 至 40 所示，研究发现用户 A 和用户 C 的基线模型对 DoS、DDoS、Infiltrainon 和 Web-attack，都有较好的检测效果，其中对 DoS、DDoS 和 Infiltrainon 检测率达到了 100%。这表明，框架不仅能有效检测外部网络攻击，对渗透成功后的内部攻击行为也能有更强的洞察力。

回顾 3 位用户的日常行为，其中 A 与 C 的学习和工作操作较为单一，使用主机的作息较有规律。但用户 B 与他们的行为差异很大，有较多娱乐的行为和新颖的操作，日常行为变化较大。由实验得出，本文的模型适合用于对行为较为规律的用户或设备进行异常检测。

实验中检测率最低的是 Botnet、Patator 和 Port-scan 攻击，其中 Botnet 攻击检测效果没有随用户不同和建模数据多少而有所改变。本文在 3.3 节中详细分析了几种攻击的行为特征，其中 Botnet、Patator 和 Port-scan 攻击在发起时都是有大量性质相似的流出现，具体到每一条流所包含的信息并不多。本文以标准五元组分流，当这 3 种攻击发起时，相当于形成多个性质相似的流，加之测试集是在每种攻击中随机抽取的少量流，使之大量相似的特征被打破，这是检测率低的重要原因之一。本研究的目的是设计易于检测未知攻击的框架，因此，在各类攻击样本中仅选取同等少量样本以检测灵敏度，实验中所检测出的攻击均为未知攻击，已基本达到实验目的。

表 34 DoS 攻击检测结果

	Precision(%)	**Recall(%)**	**F1(%)**	**Accuracy(%)**	**平均检测时长(s)**	**建模时长（s）**
用户 A	100	100	100	100	0.09	533
用户 B	72.22	72.22	72.22	88.89	0.15	1495
用户 C	100	100	100	100	0.13	263

表 35 DDoS 攻击检测结果

	Precision(%)	**Recall(%)**	**F1(%)**	**Accuracy(%)**	**平均检测时长(s)**
用户 A	100	100	100	100	0.16
用户 B	83.33	83.33	83.33	93.33	0.17
用户 C	100	100	100	100	0.14

表 36 Web attack 攻击检测结果

	Precision(%)	**Recall(%)**	**F1(%)**	**Accuracy(%)**	**平均检测时长(s)**
用户 A	91.67	91.67	91.67	96.67	0.14
用户 B	89.19	91.67	90.41	96.11	0.24
用户 C	91.67	91.67	91.67	96.67	0.15

表 37 Infiltration 攻击检测结果

	Precision(%)	**Recall(%)**	**F1(%)**	**Accuracy(%)**	**平均检测时长(s)**
用户 A	100	100	100	100	0.12
用户 B	91.67	91.67	91.67	96.67	0.15
用户 C	100	100	100	100	0.15

表 38 Bot 攻击检测结果

	Precision(%)	**Recall(%)**	**F1(%)**	**Accuracy(%)**	**平均检测时长(s)**
用户 A	58.33	58.33	58.33	58.33	0.13
用户 B	58.33	58.33	58.33	58.33	0.15
用户 C	58.33	58.33	58.33	58.33	0.13

表 39 Port Scan 攻击检测结果

	Precision(%)	**Recall(%)**	**F1(%)**	**Accuracy(%)**	平均检测时长(s)
用户 A	80.56	80.56	80.56	92.22	0.15
用户 B	61.11	61.11	61.11	84.44	0.15
用户 C	80.56	80.56	80.56	92.22	0.15

表 40 Patator 攻击检测结果

	Precision(%)	**Recall(%)**	**F1(%)**	**Accuracy(%)**	平均检测时长(s)
用户 A	80.56	80.56	80.56	92.22	0.15
用户 B	52.78	52.78	52.78	81.11	0.13
用户 C	80.56	80.56	80.56	92.22	0.13

5.2.5 可视化

1. 模型计算图

图 54 为本文系统所构造模型的计算图，由 TensorBoard 中的 GRAPHS 自动保存生成，启动 TensorBoard 可以直接显示现有模型的计算图表示，可以比较直观的审视模型运行流程。

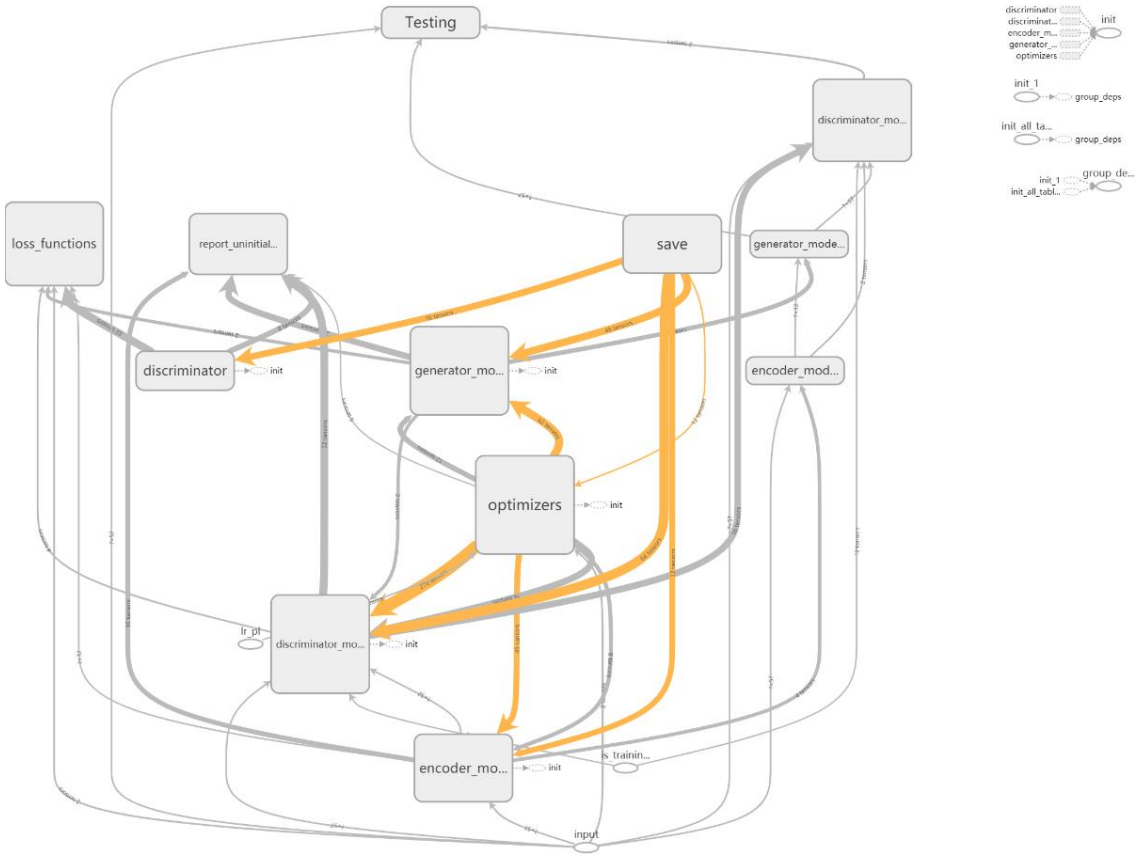

图 54 模型学习与检测结构计算图

2. 检测模型张量随迭代的变化趋势

本小节展示的结果是用户 A 检测 DDoS 攻击的可视化结果，其他的检测过程效果类似。每一个模型学习需要经过 10 次迭代，图 55 中的 D 已“被迷惑”达到震荡效果。

捕获模型训练时 G、E、D 迭代趋势的部分代码实现：

```
with tf.name_scope('dis_summary'):
    tf.summary.scalar('loss_discriminator', loss_discriminator, ['dis'])
    tf.summary.scalar('loss_dis_encoder', loss_dis_enc, ['dis'])
    tf.summary.scalar('loss_dis_gen', loss_dis_gen, ['dis'])

with tf.name_scope('gen_summary'):
    tf.summary.scalar('loss_generator', loss_generator, ['gen'])
    tf.summary.scalar('loss_encoder', loss_encoder, ['gen'])

sum_op_dis = tf.summary.merge_all('dis')
sum_op_gen = tf.summary.merge_all('gen')
```

查看可视化效果同样使用 TensorBoard，直接读取训练时建立的文件路径。

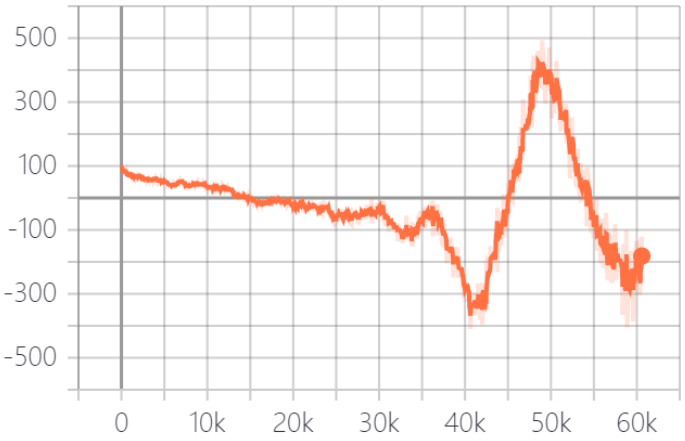

图 55 D 的损失函数训练变化趋势

D 的输出是损失值，它的前一层则是隐层的特征匹配结果，如图 56 所示。高维数据空间的可视化效果实现代码如下：

```
save = tf.train.Supervisor ( logdir= logdir, save_summaries_secs = None, save_model_secs=120)
```

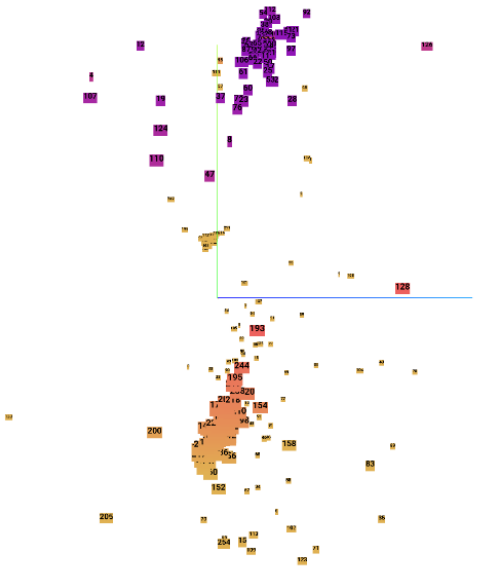

图 56 特征匹配结果

图中自动识别出正常数据和异常数据的重要特征分布，分两种颜色进行表示，可以看出本文的系统对 DDoS 攻击的检测效果较好。

5.2.6 实验对比

目前针对 CICIDS2017 数据集作测试集的研究较少，有少量研究针对数据集中的个别攻击类型作出了检测[101][102][103]，也有部分研究未标注抽取的攻击类型，本节仅与作出标注的研究结果进行对比。检测结果和使用参数对比如表 41 所示。

可以看出，本节的方法在 CICIDS2017 数据集 DoS 和 DDoS 检测中均优于其他方法，且本节实验仅随机使用了少量样本用于检测。

表 41 检测效果对比

方法	数据	样本数量	Acc.(%)	Pre.(%)	Rec.(%)	F1(%)
基于直觉模糊的集成方法（IFSE-AD)[101]	CICIDS2017-DoS	24357	97.35	93.75	100	96.8
基于随机森林和递归特征消除等的集成学习[102]	CICIDS2017-DoS	未知	未知	99.357	98.731	99.7
基于深度信任网络和多层集成 SVM[103]	CICIDS2017-DDoS	51210	96.3	90.04	95.65	93
本文的方法	CICIDS2017-DDoS	36	100	100	100	100
本文的方法	CICIDS2017-DoS	36	100	100	100	100

5.3 本章小结

本章对检测系统进行了构建，按照流量进入系统的顺序，分别是数据采集模块、特征提取模块、数据预处理模块、基线学习模块和异常检测模块。对系统的各个模块的功能进行了阐述，也对检测模块的算法原理进行了阐述。之后介绍了实验开展的环境、采取的方案、选取的评价指标、训练数据和测试数据的采集方式。最后，通过实验验证得出结论，本系统能够为不同的用户建模，并且具有在现实流量中检测攻击的能力。

第六章　总结与展望

6.1 全文总结

基于流量行为的异常检测是当下和未来的研究趋势。为了提高对未知攻击的检测能力，本文将 UEBA 的思想应用到了网络流量的异常检测领域，提出为用户刻画全面的流量基线行为，避免由于行为刻画片面导致对未知攻击检测能力低的问题。同时提出使用改进的双向 GAN 模型对固有特征分布进行自动学习，计算出基线阈值，从而避免依赖专家阈值设定而造成的误报率高等问题。

本文以面向行为的流量分析为基础，深入分析了正常用户行为和攻击行为产生的流量特征中与时间相关的特征，从而更全面的刻画了正常用户的网络流量行为基线，以发现新产生流量行为中的变化，检测攻击和其他异常行为。本文提出运用 GAN 模型进行基线学习，该方法可以避开该领域因攻击样本量不足导致的模型训练不充分问题，可以仅使用充足的正常用户行为数据来训练异常检测模型。本文的训练样本易于采集，模型易于部署，可以针对任意用户、系统或者有 IP 地址的设备进行部署，该框架检测迅速，并且对多种攻击识别具有一定效果。本文完成的主要工作如下：

1. 分析了网络流量粒度的划分，不同划分方式的优势和劣势，确定研究使用流级别的划分方式。针对传统检测漏报率高，难以检测未知攻击的问题，分析了基于行为的异常检测研究现状，又针对检测模型难以处理高维数据的问题，分析了基于机器学习方法的异常检测研究现状，再针对特征选择片面和模型通常面向特定场景进行检测的问题，分析了 UEBA 的框架思想，最终锁定了擅长分析复杂数据类型的基于深度学习的方法。

2. 分析了网络流量的表示方式，确定了研究使用 TCP/IP 底层数据表示上层交互行为。深入分析了正常用户的会话流量特征，提出了用户行为差异的基本表示。以攻击流程划分，剖析了各类常见攻击行为的流量特征，总结了识别攻击行为的显著特征。分析了现有用于入侵检测的网络数据集，指明现有数据集的不足，从而确定自行采集训练数据进行模型训练。提出了一组基于双向流和子流与时间相关的元数据特征，特征为 54 维，能够较全面刻画并且稳定表示用户的行为。依据特征组合，对采集的正常用户流量和公共数据集中的攻击流量进行提取，形成数据集。最后针对不平衡的训练数据，分析处理此类数据方法的优缺点，最后确定了仅使用一类数据的建模算法。

3. 提出了基于双向 GAN 的基线建模方法。该模型通过仅学习正常样本，在过程中不断自动优化和学习，从为不同的用户建立基线模型。阐述了模型的原理、训练部分的改进、参数选择、各部分的网络结构和训练的过程，最后将数据中的连续型特征和离散型特征预处理为符合模型学习的形式，通过实验对比进行了基线模型的有效性测试，并且确定了适合此类用户的建模时长。

4. 设计实现了基于用户实体行为的异常行为检测系统，并且对性能进行了测试。将系统应用于真实采集的实验室用户流量和公共数据集中的 7 类攻击流量，实验结果表明，该系统能够通过构建的不同用户的行为基线，产生不同的检测效果。检测速度较快，对部分攻击产生了较好的检测效果，验证了方法的可行性。

本文主要贡献：

（1）提出一种基于用户实体行为画像的特征选择方法。基于双向流和子流与时间相关的元数据特征，全面刻画用户行为，行为的刻画不必分场景。元数据特征来自 TCP/IP 协议的下层，提取方便且快速，不受加密协议的影响。

（2）提出了一种改进的基于双向 GAN 的一类复杂数据基线建模方法。该模型通过仅学习正常样本，在训练过程中自动优化参数和学习数据分布的特性，为不同的用户建立基线模型。

（3）实现了基于用户实体行为的异常检测原型系统。该系统检测不依赖专家先验知识，不针对特定场景，不受异常样本数量的限制，部署简单，检测速度快，对测试中的未知攻击有较好的检测效果。

6.2 展望

本文针对当今流量异常检测领域的不足之处提出了全新的检测框架，训练数据易于采集、检测速度快、能够检测未知攻击。本研究未来还可以针对以下方面进行深入研究：

1. 进一步研究深度学习相关理论与技术，不局限于特定学习模型，通过对模型进行深入研究，进一步优化基于用户实体行为的异常检测系统架构。

2. 进一步研究提取能够更全面刻画用户行为的流量特征，使特征组能够刻画出正常用户间的更多差异，更好的进行内部威胁的检测。

致　谢

毕业将至，原以为重返校园的三年将如时光飞逝，可以轻松自在，全心投入新的研究领域。然而，人生路上有荆棘。三年间，我收获了成长，不仅是在知识和能力上得到充盈与提高，更是经历了人生中的蜕变。

感谢导师武东英副教授一直以来对我课题研究的信任。更要感谢导师刘胜利教授对我课题的指导，为我指引了研究方向，在关键时刻点开我的思路。同时提供我诸多拓展眼界的机会，在外出交流学习和实验设备上给予全力支持。刘教授在科研上涉猎广泛、工作高效、引导有方，生活中平易近人，热爱运动，感染着我们，使我受益良多，在做科研的同时也要关注健康和生活。

感谢刘龙副教授，给予我支持与委婉的关注，感恩在心。感谢实验室的所有老师，在工作中相互照应，托起了实验室的和谐氛围。

感谢实验室同组唯一的师兄赵幸博士，虽然年纪轻，不多于言辞，却扛起了组里的项目重任，同时对我的课题提供了一些中肯的意见。在一同参加比赛的日子里，他废寝忘食的钻研精神令我佩服。感谢邻组的肖睿卿博士，一位科研与生活兼顾的暖男，跨专业的我初到实验室时，时常问一些让人无法理解的问题，他耐心倾听并提供思路，帮助我进步。感谢郭路路，提供了一些在科研工作中的经验。感谢同级的刘秉楠和王金柱同学，在平日里大小事务上给予的帮助与照顾，尤其在我不能返校的情况下，协助办理毕业事宜。还要感谢实验室里的各位师弟师妹，都发挥着各自的优势，让实验室充满了活力。

能进入这个实验室，这个集体，让我感到非常幸运。

感谢原宿舍里先毕业的小朋友们，“花儿”、“轩儿”、“妞儿”她们个性迥异，聚在一起总能擦出“火花”。从不能适应太活跃的“95 后”，到一同娱乐、健身、享美食，还有稳重的“凡儿”，与她们在一起，让我时而忽略了年龄差异，倍感轻松。

感谢学校里一直和曾经帮助、指导过我的领导和老师，也感谢三年来和谐相处的同学们，祝愿大家都能平安顺意。

最后，感谢给予我深造机会的单位领导和同事。

感谢家人的陪伴、支持与付出。

愿留下一份成果，放下一些执念，带上情谊与祝福，再出发。

孙剑文

2020 年 4 月　于北京

参考文献

[1] Stevens, Richard W. The protocols[M]. Addison-Wesley Pub. Co. 1994.

[2] Internet World Stats. INTERNET USAGE STATISTICS THE Internet Big Picture[EB/OL]. (2020-3-3). [2020-3-31]. https://www.internetworldstats.com/stats.htm.

[3] 中国互联网络信息中心. 第 44 次中国互联网络发展状况统计报告[R]. 北京：国家互联网信息办公室. 2019.

[4] 国家计算机网络应急技术处理协调中心. 2019 年上半年我国互联网网络安全态势[R]. 北京: 国家互联网信息办公室, 2019.

[5] 国家计算机网络应急技术处理协调中心. 2017 年中国互联网网络安全报告[M]. 北京: 人民邮电出版社, 2018.

[6] 中华人民共和国网络安全法[J]. 新疆农垦科技, 2017, 040(001):80-82.

[7] Wireshark. Wireshark[OL]. [2020-3-31]. https://www.wireshark.org.

[8] Zimmermann H. OSI Reference Model--The ISO Model of Architecture for Open Systems Interconnection[J]. IEEE Transactions on Communications, 1980, 28(4):425-432.

[9] Tao H, Hui Z, Zhichun L. A methodology for analyzing backbone network traffic at stream-level[C]. Communication Technology Proceedings, 2003. ICCT 2003. International Conference on. IEEE, 2003, 1: 98–102.

[10] 马丽云,马浩,孙辨华. 计算机网络基础教程[M]. 北京: 清华大学出版社, 2003.

[11] Dainotti A, Pescape A, Claffy K C. Issues and future directions in traffic classification[J]. Network, IEEE, 2012, 26(1):35-40.

[12] Baehr G G, Danielson W, Lyon T L, et al. System for packet filtering of data packets at a computer network interface: US, 5884025 [P/OL]. 1999-03-16[2020-3-31]. http://www.freepatentsonline.com/5884025.html.

[13] Claffy K C, Braun H W, Polyzos G C. A parameterizable methodology for Internet traffic flow profiling[J]. IEEE Journal on Selected Areas in Communications, 1995, 13(8):1481-1494.

[14] Awodele O, Oluwabukola O, Ogbonna C, et al. Packet Sniffer - A Comparative Characteristic Evaluation Study[C]// Insite: Informing Science + It Education Conferences: Usa. 2015.

[15] Cisco. Cisco Visual Networking Index: Forecast and Methodology 2013–2018[OL]. (2014-12-20). [2020-3-31]. https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-481360.html.

[16] Yu F, Chen Z, Diao Y, et al. Fast and memory-efficient regular expression matching for deep packet inspection[C]// IEEE Symposium on Architecture for Networking & Communications Systems. IEEE, 2006.

[17] Ahmed I, Lhee K S. Classification of packet contents for malware detection[J]. Journal in Computer Virology, 2011, 7(4): 279-295.

[18] Camacho J, Perez-Villegas A, Garcia-Teodoro P, et al. PCA-based multivariate statistical network monitoring for anomaly detection[J]. Computers & Security, 2016, 59(Jun.):118-137.

[19] RFC 5101. Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information[EB/OL]. (2008-1). [2020-3-31]. https://datatracker.ietf.org/doc/rfc5101

[20] Cisco. NetFlow[OL]. [2020-3-31]. http://www.cisco.com/c/zh_cn/tech/quality-of-service-qos/netflow/index.html.

[21] SFlow. SFlow[OL]. [2020-3-31]. http://www.sflow.org.

[22] Li B, Springer J, Bebis G, et al. A survey of network flow applications[J]. Journal of Network & Computer Applications, 2013, 36(2):567-581.

[23] Anderson J P. Computer security threat monitoring and surveillance[J]. Technical Report, James P. Anderson Company, 1980: 1.

[24] Sharma S, Gupta R. Intrusion detection system: A review[J]. International Journal of Security and Its Applications, 2015, 9(5): 69-76.

[25] Khraisat A, Gondal I, Vamplew P. An Anomaly Intrusion Detection System Using C5 Decision Tree Classifier[C]// Pacific-asia Conference on Knowledge Discovery & Data Mining. Springer, Cham, 2018.

[26] Ye N, Emran SM, Chen Q, Vilbert S (2002) Multivariate statistical analysis of audit trails for host-based intrusion detection[J]. IEEE Trans Comput 51(7):810–820.

[27] Viinikka J, Hervé Debar, Ludovic Mé, et al. Processing intrusion detection alert aggregates with time series modeling[J]. Information Fusion, 2009, 10(4):312-324.

[28] Wu Q, Shao Z. Network Anomaly Detection Using Time Series Analysis[C]// Autonomic and Autonomous Systems and International Conference on Networking and Services, 2005. ICAS-ICNS 2005. Joint International Conference on. IEEE Computer Society, 2005.

[29] Duffield N, Lund C, Thorup M. Estimating flow distributions from sampled flow statistics[J]. IEEE/ACM Transactions on Networking, 2005, 13(5): 933-946.

[30] Hood C S, Ji C. Proactive Network Fault Detection[C]// INFOCOM '97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE. IEEE, 1997.

[31] Kline J, Nam S, Barford P, et al. Traffic Anomaly Detection at Fine Time Scales with Bayes Nets[C]// International Conference on Internet Monitoring & Protection. IEEE, 2008.

[32] Koc L, Mazzuchi T A, Sarkani S. A network intrusion detection system based on a Hidden Naïve Bayes multiclass classifier[J]. Expert Systems with Applications, 2012, 39(18):13492–13500.

[33] Li Y, Xia J, Zhang S, et al. An efficient intrusion detection system based on support vector machines and gradually feature removal method[J]. Expert systems with applications, 2012, 39(1): 424-430.

[34] Syarif I, Prugel-Bennett A, Wills G. Data mining approaches for network intrusion detection from dimensionality reduction to misuse and anomaly detection. Journal of Information Technology Review, 2012, 3(2): 70-83.

[35] Dokas P, Ertoz L, Kumar V, Lazarevic A, Srivastava J, Tan P N. Data mining for network intrusion detection. In Proceedings of NSF Workshop on Next Generation Data Mining, 2002, 38(7): 21-30.

[36] 许晓东,杨燕,李刚. 基于 K-means 聚类的网络流量异常检测[J]. 无线通信技术, 2013(04): 25-30.

[37] Yasami Y, Mozaffari R P. A novel unsupervised classification approach for network anomaly detection by k-Means clustering and ID3 decision tree learning methods[J]. Journal of supercomputing, 2010, 53(1): 231-245.

[38] Schmidhuber, Jürgen. Deep learning in neural networks: An overview[J]. Neural Networks, 2015, 61:85-117.

[39] LeCun Y, Bengio Y, Hinton G. Deep learning[J]. Nature, 2015, 521(7553):436.

[40] Sumeet Dua and Xian Du. Data mining and machine learning in cybersecurity[M]. USA: Auerbach Publications, 2011.

[41] Lin W C, Ke S W, Tsai C F. CANN: An intrusion detection system based on combining cluster centers and nearest neighbors[J]. Knowledge Based Systems, 2015, 78:13-21.

[42] McHugh, John. Testing Intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory[J]. Acm Transactions on Information & Systems Security, 2000, 3(4):262-294.

[43] Ashfaq R A R, Wang X Z, Huang J Z, et al. Fuzziness based semi-supervised learning approach for intrusion detection system[J]. INFORMATION ENCES, 2017.

[44] Tang D H, Cao Z. Machine Learning-based Intrusion Detection Algorithms[J]. Journal of Computational Information Systems, 2009, 5(6): 1825-1831.

[45] Dorothy E. Denning. An Intrusion-Detection Model[J]. IEEE Transactions on Software Engineering, 1987, 13(2): 222-232.

[46] Kumar R S S, Vu N S K, DiPlacido M, et al. Lateral movement detection: US, 9591006[P/OL]. 2017-03-07[2020-3-31]. http://www.freepatentsonline.com/9591006.html

[47] Litan A. Market Guide for User and Entity Behavior Analytics (G00276088)[EB/OL]. (2015-9-22). [2020-3-30]. https://www.gartner.com/en/documents/3134524.

[48] UEBA 设计之路 1：UEBA 框架[OL]. (2019-1-23). [2020-4-1]. http://www.secpulse.com/archives/-95668.html

[49] Gurucul. UEBA[OL]. [2020-3-30]. https://gurucul.com/products/user-entity-behavior-analytics-ueba.

[50] Exabeam. User and Entity Behavior Analytics [OL]. [2020-3-30]. https://www.exabeam.com/siem-guide/ueba.

[51] Logrhythm. User and Entity Behavior Analytics (UEBA)[OL]. [2020-3-30]. http://logrhythm.com/-solutions/security/user-and-entity-behavior-analytics.

[52] 司德睿, 华程, 杨红光,等. 一种基于机器学习的安全威胁分析系统[J]. 信息技术与网络安全, 2019(4).

[53] 胡绍勇. 基于 UEBA 的数据泄漏分析[J]. 信息安全与通信保密, 2018, 000(008):26-28.

[54] Litan A, Sadowski G, Bussa T, et al. Market Guide for User and Entity Behavior Analytics (G00349450)[EB/OL]. (2018-4-23). [2020-3-30]. https://www.gartner.com/en/documents/- 3872885.

[55] Slipenchuk P, Epishkina A. Practical User and Entity Behavior Analytics Methods for Fraud Detection Systems in Online Banking: A Survey[J]. 2019.

[56] Kullback S, Leibler R A. On Information and Sufficiency[J]. Annals of Mathematical Statistics, 22(1):79-86.

[57] Kasun L L C, Yang Y, Huang G B, et al. Dimension Reduction With Extreme Learning Machine[J]. IEEE Transactions on Image Processing, 2016, 25(8):3906-3918.

[58] Goodfellow I J, et al. Generative Adversarial Nets[C]// Proceedings of the 27th International Conference on Neural Information Processing Systems- Volume 2. Montreal, Canada: MIT Press, 2014: 2672–2680.

[59] Schlegl T, Seebck P, Waldstein S M, et al. Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery[J]. 2017.

[60] Ravanbakhsh M, Nabi M, Sangineto E, et al. Abnormal Event Detection in Videos using Generative Adversarial Nets[J]. 2017.

[61] Creswell A, White T, Dumoulin V, et al. Generative Adversarial Networks: An Overview[J]. IEEE Signal Processing Magazine, 2017, 35(1):53-65.

[62] TensorFlow. TensorFlow[OL]. [2020-4-1]. https://github.com/tensorflow/tensorflow.

[63] PyTorch. PyTorch[OL]. [2020-4-15]. https://github.com/pytorch/pytorch.

[64] Caffe. Caffe[OL]. [2020-4-15]. https://github.com/BVLC/caffe.

[65] Mxnet. Mxnet[OL]. [2020-4-15]. https://github.com/apache/incubator-mxnet.

[66] CNTK. CNTK[OL]. [2020-4-15]. https://github.com/microsoft/CNTK.

[67] TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems[J]. 2016.

[68] Appelt D, Nguyen C D, Briand L. Behind an Application Firewall, Are We Safe from SQL Injection Attacks?[C]// IEEE International Conference on Software Testing, Verification & Validation. IEEE, 2015.

[69] 赵宇飞,熊刚,贺龙涛,等. 面向网络环境的 SQL 注入行为检测方法[J]. 通信学报, 2016, 037(002):88-97.

[70] 方滨兴,崔翔,王威. 僵尸网络综述[J]. 计算机研究与发展, 2011, 048(008):1315-1331.

[71] Mantas G, Stakhanova N, Gonzalez H, et al. Application-layer denial of service attacks: taxonomy and survey[J]. International Journal of Information and Computer Security, 2015, 7(2/3/4):216.

[72] Zargar S T, Joshi J, Tipper D. A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks[J]. IEEE Communications Surveys & Tutorials, 2013, 15(4):2046-2069.

[73] Kuzmanovic A, Knightly E W. Low-rate TCP-targeted denial of service attacks and counter strategies[J]. IEEE/ACM Transactions on Networking, 2006, 14(4):683-696.

[74] Prasad K M, Mohan A R, Rao K V. Dos and DDoS attacks: Defense, detection and traceback mechanisms-A survey[J]. Global Journal of Computer Science and Technology, 2014, 14(7).

[75] Zargar S T, Joshi J, Tipper D. A survey of defense mechanisms against distributed denial of service (DDOS) flooding attacks[J]. IEEE Communications Surveys & Tutorials, 2013, 15(4): 2046-2069.

[76] FreeBuf. 年末了，盘点 2016 年最严重的 7 起 DDoS 攻击事件[OL]. (2018-02-09). [2020-4-1]. https://cloud.tencent.com/developer/article/1041124.

[77] Bhuyan M H, Bhattacharyya D K, Kalita J K. Network Anomaly Detection: Methods, Systems and Tools[J]. IEEE Communications Surveys & Tutorials, 2014, 16(1):303-336.

[78] Keras. Keras[OL]. [2020-4-1].https://github.com/keras-team/keras.

[79] Stolfo S J, Fan W, Lee W, et al. Costbased modeling for fraud and intrusion detection: Results from the JAM project[J]. DISCEX ,2002, 1(2): 130–144.

[80] Shiravi A, Shiravi H, Tavallaee M, et al. Toward developing a systematic approach to generate benchmark datasets for intrusion detection[J]. Computers & Security, 2012, 31(3): 357-374.

[81] Moustafa N, Slay J. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)[C]// Military Communications & Information Systems Conference. IEEE, 2015: 1-6.

[82] Sharafaldin I, Lashkari A H, Ghorbani A A. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization[C]// International Conference on Information Systems Security & Privacy. 2018.

[83] 古平,欧阳源遊. 基于混合采样的非平衡数据集分类研究[J]. 计算机应用研究, 000(2): 379-381,418.

[84] Zeng M, Zou B, Wei F, et al. Effective prediction of three common diseases by combining SMOTE with Tomek links technique for imbalanced medical data[C]// IEEE International Conference of Online Analysis & Computing Science. IEEE, 2016: 225-228.

[85] Laurikkala J. Instance-based data reduction for improved identification of difficult small classes[J]. Intelligent Data Analysis, 2002, 6(4):311-322.

[86] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, et al. SMOTE: Synthetic Minority Over-sampling Technique[J]. Journal of Artificial Intelligence Research, 2002, 16(1):321-357.

[87] Sabhnani M, Gürsel Serpen. Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context[C]// Proceedings of the International Conference on Machine

Learning; Models, Technologies and Applications. MLMTA'03, June 23 - 26, 2003, Las Vegas, Nevada, USA. DBLP, 2003.

[88] Hu W, Maybank S. AdaBoost-Based Algorithm for Network Intrusion Detection[J]. IEEE Transactions on Systems Man & Cybernetics Part B Cybernetics A Publication of the IEEE Systems Man & Cybernetics Society, 38(2): 577-583.

[89] Liu X Y, Wu J, Zhou Z H. Exploratory Undersampling for Class-Imbalance Learning[J]. IEEE TRANSACTIONS ON CYBERNETICS, 2009, 39(2): 539-550.

[90] Sebastiani F. Machine Learning in Automated Text Categorization[J]. ACM Computing Surveys, 2002, 34(1): 1-47.

[91] Schlkopf B, Williamson R C, Smola A J, et al. Support Vector Method for Novelty Detection[C]// Advances in Neural Information Processing Systems 12, NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999. MIT Press, 1999.

[92] Donahue J, Krhenbühl, Philipp, Darrell T. Adversarial Feature Learning[J]. 2016.

[93] Javaid A Y, Niyaz Q, Sun W, et al. A Deep Learning Approach for Network Intrusion Detection System[C]// 9th EAI International Conference on Bio-inspired Information and Communications Technologies. ICST, 2015.

[94] Arjovsky M, Chintala S, Bottou, Léon. Wasserstein GAN[J]. 2017.

[95] Kirszbraun M D. Über die zusammenziehende und Lipschitzsche Transformationen[J]. Fundamenta Mathematicae, 1934, 22:77-108.

[96] Gulrajani I, Ahmed F, Arjovsky M, et al. Improved Training of Wasserstein GANs[J]. 2017.

[97] Jarrett K, Kavukcuoglu K, Ranzato M, et al. What is the Best Multi-Stage Architecture for Object Recognition?[C]// Proc. International Conference on Computer Vision (ICCV'09). IEEE, 2009.

[98] Hinton G E, Srivastava N, Krizhevsky A, et al. Improving neural networks by preventing co-adaptation of feature, detectors[J]. Computer ence, 2012, 3(4): p. 212-223.

[99] Kingma D, Ba J. Adam: A Method for Stochastic Optimization[J]. Computer ence, 2014.

[100] Nie F, Huang H, Cai X, et al. Efficient and Robust Feature Selection via Joint ℓ2, 1-Norms Minimization[C]// Advances in Neural Information Processing Systems 23: Conference on Neural Information Processing Systems A Meeting Held December. Curran Associates Inc. 2010.

[101] Wang J, Zhao H, Xu J, et al. Using Intuitionistic Fuzzy Set for Anomaly Detection of Network Traffic from Flow Interaction[J]. IEEE Access, 2018, 6: 64801-64816.

[102] 马泽文,刘洋,徐洪平,等. 基于集成学习的 DoS 攻击流量检测技术[J]. 信息网络安全, 2019.

[103] Marir N, Wang H, Feng G, et al. Distributed Abnormal Behavior Detection Approach based on Deep Belief Network and Ensemble SVM using Spark[J]. IEEE Access, 2018:1-1.

[104] Microsoft. Office 365 for business FAQ[OL]. [2020-4-1] https://products.office.com/en-us/business/microsoft-office-365-frequently-asked-questions.