电子科技大学

UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

专业学位硕士学位论文

MASTER THESIS FOR PROFESSIONAL DEGREE

论文题目 工业物联网信息安全防护技术研究

专业学位类别	工 程 硕 士
学 号	**201522070342**
作 者 姓 名	王斌
指 导 教 师	郑宏 副教授

分类号 _____ 密级 _____

UDC [注1] _____

学 位 论 文

工业物联网信息安全防护技术研究

（题名和副题名）

王斌

（作者姓名）

指导教师 郑宏 副教授

电子科技大学 成 都

（姓名、职称、单位名称）

申请学位级别 硕士 专业学位类别 工 程 硕 士

工程领域名称 控制工程

提交论文日期 2018.04.11 论文答辩日期 2018.05.15

学位授予单位和日期 电子科技大学 2018 年 6 月

答辩委员会主席 _____

评阅人 _____

注 1：注明《国际十进分类法 UDC》的类号。

Research on Information Security Protection Technology of Industrial Internet of Things

A Master Thesis Submitted to

University of Electronic Science and Technology of China

Discipline: **Master of Engineering**

Author: **Wang Bin**

Supervisor: **Prof.Zheng Hong**

School: **School of Automation Engineering**

独创性声明

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

作者签名：王建　　日期：2019年 6月 19日

论文使用授权

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

（保密的学位论文在解密后应遵守此规定）

作者签名：王建　　导师签名：

日期：2019年 6月 19日

摘 要

工业物联网（IIoT，Industrial Internet of Things）是工业自动化系统与物联网系统的融合，具有全面感知、互联传输、智能处理以及自组织与自维护等特点，其应用遍及智能交通、智慧工厂、智能电网、智能环境检测等众多领域。随着 IIoT 技术的广泛应用，工业物联网系统所遭受的网络安全威胁与日俱增，信息安全问题成为其发展过程中需要面临的重大挑战。为保障工业物联网系统不受网络攻击的危害，本文旨在研究工业物联网信息安全防护技术，设计并实现了针对工业物联网系统的安全防护策略。本文的主要工作如下：

首先分析了国内外工业物联网系统的安全研究现状，然后研究了工业物联网系统的典型架构，并根据该系统架构的不同层次，详细分析了目前工业物联网系统所面临的网络安全威胁。

对于工业物联网系统所面临的各种安全威胁，结合具体的网络攻击手段，针对性地设计了应用于不同层次结构的安全防护策略，包括针对数据采集层的安全防护策略、数据传输层的安全防护策略以及数据处理层的安全防护策略。

对于上述应用于不同层次结构的安全防护策略，对各个防护方法分别进行了设计实现。对于数据采集层中的物联网终端设备，采用端口扫描检测、暴力破解检测，并与防火墙进行联动防御的方式进行设备防护，防止终端设备遭到 Mirai 等物联网僵尸网络病毒的感染；对于数据传输层，采用 TLS 对传输数据进行加密、X509 证书进行设备身份认证的方式来防御数据窃取和中间人攻击；对于数据处理层，采用基于白名单的深度包检测方法对通信服务器进行访问控制，并使用基于朴素贝叶斯分类器的异常流量检测方法对通信服务器的通信流量进行实时检测。

最后，对上述各个防护方法分别进行实验测试，均能有效防御相关的网络攻击，能够保障工业物联网系统安全、稳定地运行。

关键词：工业物联网（IIoT），物联网僵尸网络病毒，深度包检测，朴素贝叶斯，异常流量检测

ABSTRACT

Industrial Internet of Things (IIoT) is a fusion of industrial automation system and Internet of Things (IoT) system. It features comprehensive sensing, interconnection transmission, intelligent processing, self-organization and self-maintenance, and it is applied to many fields such as intelligent transportation, smart factories, smart grids and environment detection. With the widespread application of IIoT technology, the cybersecurity threats to IIoT systems are increasing day by day, and information security has become a major challenge in the development of IIoT. In order to protect the IIoT system from the cyberattack, this paper makes an in-depth study on the IoT information security technology and design the defense strategy for IIoT. The main work is as follows:

This paper analyzes the current situation of security research of IIoT system at home and abroad, then studies the typical architecture of IIoT system. According to the different levels of the architecture of the system, this paper analyzes the network security threats to the current IIoT system in detail.

For various security threats faced by IIoT system, combining with specific cyberattacks, this paper designs targeted security protection strategies applied to different levels of architecture, including security protection strategies for data acquisition layer, data transmission layer, and data processing layers.

Each protection method mentioned above is designed and implemented separately in this paper. In data acquisition layer, port scan detection, brute force detection, and linkage defense with firewall protection are combined to prevent the terminal devices from being infected by IoT botnet viruses such as Mirai. In data transmission layer, TLS is used to encrypt data and X509 certificates are used for identity authentication to defense against data theft and man-in-the-middle attacks. In data processing layer, deep packet inspection method based on whitelist is used to restrict access to the communication server, and the abnormal traffic detection method based on Naïve Bayes classifier is used to monitor the traffic flow in real time.

After the experimental test, each of the protection methods can effectively prevent related network attacks and ensure the safe and stable operation of IIoT systems.

Keywords: industrial Internet of Things (IIoT), IoT botnet viruses, deep packet inspection, Naive Bayes classifier, abnormal traffic detection

目 录

第一章 绪论

1.1 选题背景与研究意义

物联网（IoT，Internet of Things）是现代科技潮流当中最炙手可热的技术之一，被称为是继计算机和互联网之后的第三次信息技术革命[1]。1999 年，MIT Auto-ID 中心的 Ashton 教授最早提出来了物联网这一概念，其定义是：使用 RFID（射频识别）、GPS（全球定位系统）、红外感应装置、激光扫描装置等信息感知设备[2]，通过某种通信协议就能够将任何物品接入到互联网中，进行数据采集和信息交换，从而完成智能化的识别、定位、跟踪、监控以及管理[3]。除了改变传统的人机交互方式以外，物联网技术也让机器与机器之间的交互方式产生了历史性的变革，很大程度上提高了生产效率和经济利益，为人们的生产生活提供了便利[4]。

物联网发展至今，已经开始应用到各个领域，其中普及速度最快、范围最广的垂直领域就在工业界[5]，能源、医疗保健、汽车和其他相关行业也正开始与工业物联网（IIoT，Industrial Internet of Things）相融合，诸如传感器、机器人、混合罐等越来越多的设备连接在一起[6]。工业物联网可以看作是工业自动化系统与物联网系统的高度结合体，在其发展过程中引入了互联网、高级计算、分析以及传感等技术，并完成了工业生产系统、工业监控系统以及工业管理系统的融合，根据数据中心对工业数据的分析和处理结果，能够大幅提高产品质量和工业生产效率，并有效降低生产管理成本[7]。

根据目前工业物联网技术的发展趋势和应用场景，工业物联网带来的变革主要集中在以下几个方面：

生产设备的监控和管理。通过互联的传感器网络，工业物联网技术能够对设备进行在线检测和实时监控，将整个生产环境的运行情况呈现给管理人员。

制造业的供应链管理。对于整个供应链的管理架构，通过 IIoT 技术对其进行优化和完善，整体运作效率能够得到提升，同时能够降低生产过程中不必要的损耗，提高内部效益并改善外部管理水平[8]。

生产过程的优化和预测性运维。IIoT 技术解决方案包括了互联的传感器、IT 系统、数据和实时分析，能够通过数据挖掘等技术，形成智能诊断和智能决策机制，改善生产工艺和流程，提供更合理的维修计划，实现正常运转时间最大化。

随着工业物联网技术在工业领域的广泛应用，工控系统中设备层的控制装置将通过网络实现信息交互，并且能够将管理层与市场层进行无缝融合。然而，越来越开放的网络化连接使得工控系统、联网设备以及工业云平台容易遭受入侵，给工业环境带来停机、生产中断、资产损失等威胁[9]。

物联网发展至今，国内外基于工业物联网安全方面的事件频繁发生，对于入侵者来说，比起攻击其他行业的物联网系统，攻击工业物联网系统能够引起更大的关注或获得更多的利益。攻击者采取的入侵方式多种多样，如工业核心数据泄露、互联终端遭到非法劫持和操控等[10]，其中影响力较大的针对工业物联网系统或者间接与工业物联网系统相关的安全事件如下：

2016 年 10 月 21 日，美国遭遇了史上最大规模的一次互联网断网，被称为“Mirai”的一款恶意软件所控制的僵尸网络对美国的域名服务器管理服务供应商 Dyn 发起 DDoS 攻击，造成包括 Twitter、Spotify、Netflix、Github、CNN、华尔街日报等上百家网站无法访问[11]。大量的互联网设备是造成此次断网事件的罪魁祸首，黑客利用各类物联网设备存在的安全漏洞，通过 Mirai 软件能够感染了大量联网的摄像头和数字录像机等，这些被感染设备将成为僵尸网络中的“肉鸡设备”，被用来实施大规模 DDoS 攻击[12]。

2017 年 4 月中旬起，国内也出现了由名为“http81”病毒操控的大规模的僵尸网络。http81 僵尸网络在短期内迅速感染了超过 5 万台 IoT 设备，预计可能拥有高达 500Gbit/s 的 DDoS 攻击能力，给国内互联网基础设施带来巨大威胁。

2017 年 4 月 11 日，网络安全服务提供商 Radware 在工业控制系统中发现了一个名为“BrickerBot”的针对 Linux 操作系统的物联网设备的病毒，该病毒一旦感染 IIoT 设备，将会永久删除存储并撤销网络访问，从而使该 IIoT 设备完全失效。

2017 年 8 月，哈尔滨市跨越科技有限公司投放的“橙风单车”遭遇大规模故障，大量单车在扫码后出现无法正常解锁的情况，新用户在注册时也无法接收到验证码等信息，最终导致近 5000 台共享单车无法正常使用。该公司对服务器后台进行检查后发现，单车管理系统遭到黑客发起的持续性攻击。

工业物联网依托于现代成熟的工业自动化系统，融合了大量来自通信、计算机等领域的技术与应用，因此一些传统的网络攻击方法也适用于工业物联网系统，几年来发生的大量攻击事件也逐步将工业物联网在信息安全方面存在的各种安全隐患暴露出来，这对处于上升趋势的工业物联网来说是一大障碍。随着第四次工业革命在全球范围内的兴起，德国提出了工业 4.0 概念，我国也制定了中国制造 2025 战略，工业物联网作为关键技术将会应用到越来越多的工业场景当中，因此工业物联网系统的信息安全是物联网安全目标的重点。

1.2 国内外研究现状

近年来，针对工业物联网系统的网络攻击事件层出不穷，呈现出持续上升的趋势，工业物联网系统的安全问题已经在信息安全行业引起了高度关注，国内外

的官方组织、研究机构以及安全厂商都已经开展了相关的研究。

1.2.1 国外物联网信息安全研究现状

工业物联网系统中逐渐暴露出来的安全问题已经引起了欧美各国的高度重视，官方组织、学术研究机构等都积极投入到工业物联网环境下的安全研究，旨在构建一系列的行业规范和技术。

2016 年 9 月，工业互联网联盟（IIC）发布了《工业物联网安全框架》，目的在于解决工业物联网及全球工业操作运行系统中存在的安全问题，从多个角度解决信息安全、系统可靠性和隐私保护等问题[13]，该框架从安全保障、隐私性、安全性、可靠性以及适应性这几个方面出发，给物联网相关从业人员提供了各种风险、评估、威胁和性能指标作为安全建议。同年 11 月，美国国土安全部（DHS）发布了《物联网安全指导原则》，其中包括了设计安全、漏洞管理和修复、最佳安全实践的采用、利用风险管理聚焦优先事务、供应链透明性以及持续连接的必要性判定等六项指导性原则[14]。以上安全框架和指导原则的主要目的在于给物联网硬件设备生产商、软件系统开发商以及物联网系统管理员提供有关设计、制造和部署物联网设备时应该遵守的安全建议，然而没有形成一套完整的安全技术规范，也没有涉及对具体的网络攻击和安全技术的研究。

物联网安全项目（SITP）是由美国多所大学组成的一个跨学科的研究项目，其研究目标包括两点：一是研究和定义新的密码学计算模型和安全机制以确保物联网终端设备的安全；二是研究安全、开源的软硬件框架来对物联网应用进行设计和构建，使其可以正确使用这些新的机制。该项目偏向于基础安全理论方面的研究，其研究方向主要包括以下几个方面：研究适用于未来的嵌入式 SoC 的密码学原语；研究应用于物联网设备当中的签名算法；研究应用于物联网设备的安全的嵌入式操作系统；为物联网部署制定操作较强的安全实施规范。

除官方组织和学术研究机构外，各大安全厂商也在进行物联网安全技术的研究工作。对于工业物联网系统，知名安全厂商赛门铁克公司提供了两类解决方案，一是帮助管理人员保护工业物联网系统中基础设施的安全，二是帮助物联网设备供应商提高其产品的安全性。赛门铁克联合全球最大的芯片提供商德州仪器公司，以及密码服务提供商 wolfSSL 公司，将认证技术、加密技术以及嵌入式技术结合到一起，为物联网设备提供可靠的信息加密和身份验证[15]。这种基于嵌入式技术的设备保护方法，需要在物联网设备生产的过程中在硬件层面嵌入可信证书和设备秘钥，对于已有的设备则无法应用该方法，因此其使用范围有一定的局限性。

1.2.2 国内物联网信息安全研究现状

近年来，物联网在我国的发展已经进入多行业落地阶段，同时也开始进入物联网安全建设阶段，相关科研单位及安全厂商都在积极探索物联网安全规范与技术[16]。国内知名安全厂商北京匡恩网络科技有限公司在 2016 年发布了《2016 年度物联网安全研究报告》[17]，该报告整体上对物联网及工业物联网的安全现状、安全保护技术及产业发展趋势进行了阐述。在 2017 年，中国电信联合北京神州绿盟信息安全科技有限公司发布了《2017 物联网安全研究报告》，该报告主要分析了 2017 年物联网存在的安全事件及相关恶意软件，提出物联网安全需求和安全体系结构。

杨悦梅、宋执环在论文中分析了工业物联网在工业项目中的具体应用，以及工业物联网在全面感知层、可靠传输层、智能处理层存在安全风险，并提出了业务认证机制、加密机制、隐私保护等技术[18]。在工业物联网系统安全中提供了建设性的意见，然而其中主要是对各个防护技术的整体介绍，并未涉及该类技术具体的实现方案。

朱艳在论文中分析了物联网感知层中的无线传感器网络，提出了一种针对无线传感器网络的身份认证方案，该方案采用了椭圆曲线密码体制来实现以下功能：生成会话密钥、对传感器之间进行双向认证[19]。该方案所提供的认证机制能够保护物联网感知层的数据机密性和完整性，但是该方案只是对物联网系统感知层的数据安全进行防护，并没有对感知层的物联网终端设备进行防护。

郭莉、严波、沈延在论文中分析物联网系统所面临的安全威胁和安全需求，基于物联网系统传统架构提出了一套系统安全加固方案，包括将密码芯片内置在终端设备中，为信息处理中心设置密码机，通过数据安全网关将信息处理中心和互联网进行隔离，通过认证和加密机制保护数据的真实性、完整性和机密性，访问控制机制能够阻断互联网中对物联网内部资源的非法访问[20]。该方案对物联网系统的网络通信安全提供了重要的参考价值，但是该方案需要对物联网终端设备的硬件进行更改，对已有设备实施该方法时刻操作性不强；其次该方案并没有对物联网系统应用层网络中存在的异常行为进行检测，无法检测到高级持续性攻击。

1.3 本论文主要研究内容和结构安排

目前国内外对物联网系统的信息安全研究主要集中在对基础安全理论和技术的研究上，而对实际应用中的物联网信息安全技术的研究则比较少，尤其缺乏针对物联网系统存在的安全漏洞和遭受的具体攻击方式进行安全防护技术研究，具体体现在缺乏对物联网僵尸网络病毒、物联网通信协议以及物联网云服务平台这

三个方面的安全防护技术研究。

对于工业物联网系统的信息安全研究，也存在以上问题，因此本课题针对工业物联网系统中实际存在的安全需求，进行相应的安全防护技术研究。本文对工业物联网系统的数据采集层、数据传输层及数据处理层中存在的安全威胁进行了分析，针对这些安全威胁设计了一套较为通用的工业物联网信息安全防护方案，该方案以工业物联网的三层体系结构为出发点，主要从物联网终端设备防护、数据加密传输及认证、深度包检测和入侵检测等方面建立一套深度防御体系。

1、本文主要研究内容

本文的研究内容包括以下几个方面：

⑴ 设计针对物联网僵尸网络病毒的安全防护方法

本文深入研究了以 Mirai 为代表的物联网僵尸网络病毒对物联网终端设备的感染方法，总结出其感染方法主要包括端口扫描探测和暴力字典破解这两个流程。针对具体的感染流程，设计了基于 snort 和 iptables 防火墙联动的方法来防御端口扫描，以及通过 iptables 的访问控制功能限制远程登录次数的方法来防御暴力字典破解攻击，从而对物联网终端设备进行安全防护。

⑵ 设计适用于 MQTT 协议的通信加密和身份认证方法

针对数据传输层中 MQTT 协议可能遭受的信息窃取和中间人攻击，本文使用互联网应用中广泛采用的 TLS 加密技术来对 MQTT 协议的通信过程进行加密，防止 MQTT 通信数据泄露；在 TLS 加密的基础上，引入了基于 X509 数字证书的身份认证方法，对请求连接的 MQTT 客户端设备进行合法性验证，降低遭到中间人攻击的可能性。

⑶ 设计基于深度包检测的访问控制防护模块

针对数据处理层中 MQTT 服务器可能遭受的伪造数据包攻击，本文对 MQTT 协议的控制报文结构进行分析，以白名单访问控制策略为基础，设计了从 TCP/IP 数据流和应用层 MQTT 协议报文这两方面进行流量检测的深度包防护模块，限制不符合 MQTT 协议规约的数据包对 MQTT 服务器的访问。

⑷ 设计基于朴素贝叶斯的异常流量检测防护模块

对于构造精良、符合协议规约和访问控制规则的欺骗性数据包，上述深度包检测模块有可能无法检测到其攻击行为，因此非常有必要引入实时的流量检测机制。本文将朴素贝叶斯分类算法应用到流量检测，选取适当的 MQTT 协议特征对分类器进行训练，构建基于朴素贝叶斯分类器的 MQTT 异常流量检测模块，对数据处理层中的 MQTT 协议流量进行实时监控，保护 MQTT 服务器的通信安全。

2、本文结构安排

本论文在结构上一共分为七个章节，主要工作如下：

第一章，绪论。简要叙述选题背景和研究意义，主要介绍了工业物联网的应用场景以及近年来工业物联网系统中发生的一系列安全事件，得出对工业物联网信息安全进行深入研究的必要性；对国内外物联网行业的信息安全研究现状进行分析总结；最后对本论文的主要研究内容和结构安排进行了简单叙述。

第二章，工业物联网系统安全威胁分析及安全防护策略设计。介绍了工业物联网系统的典型系统架构和基本功能，并搭建了该系统的模拟环境作为本文研究的实验对象；从系统架构的数据采集层、数据传输层和数据处理层分析了工业物联网系统的安全威胁及安全需求，然后针对上述安全需求，分别对终端设备保护、通信保护和认证、入侵检测和防御这三个方面进行防护策略研究，分别设计出相应的安全解决方案。

第三章，数据采集层防护策略设计实现。首先分析了以 Mirai 恶意软件为代表的物联网僵尸网络病毒对物联网设备的感染方法，主要包括端口扫描和暴力字典破解这两种攻击方式。针对这两种攻击方式的实现原理，设计了相应的防护方法来防御物联网僵尸网络病毒对物联网设备的感染。

第四章，数据采集层防护策略的设计实现。分析了工业物联网系统数据传输层中广泛使用的 MQTT 协议存在的安全隐患，包括缺乏通信数据加密和设备身份认证机制，并针对这两点安全问题提出了基于 TLS 的 MQTT 通信加密和基于 X509 证书的身份认证方法，保障数据传输层提供安全的数据传输服务。

第五章，数据处理层防护策略的设计实现。针对数据处理层的 MQTT 服务器可能遭受到的伪造数据包攻击，设计了基于白名单的深度包检测模块对该服务器进行防护；对于数据处理层通信网络中存在的高级持续性攻击，设计了基于朴素贝叶斯分类器的异常流量检测模块对通信流量进行实时监控，及时发现入侵行为。

第六章，对工业物联网系统安全防护策略进行测试分析，包括僵尸网络病毒攻击防护方法、MQTT 协议加密传输及身份认证、MQTT 协议深度包检测、基于朴素贝叶斯的 MQTT 异常流量检测方法的测试和分析。

第七章，总结与展望。对本文工业物联网系统安全防护策略的研究工作进行了总结，并阐述了后续可以进一步研究的方向。

第二章 工业物联网安全威胁分析及安全防护策略设计

本章主要分析工业物联网系统的体系结构，根据其典型的系统架构搭建出一套工业物联网中常见的数据采集和远程监控系统，并以这套数据采集系统为研究对象，分析工业物联网系统所面临的各种安全威胁，最后针对每一类安全威胁设计出相应的安全防护策略。

2.1 工业物联网系统架构

2.1.1 一般物联网系统整体结构

物联网是一个融合了各种复杂技术的网络，根据信息采集、传输和处理的流程，物联网的层次结构自下而上可以划分为 3 层[21]：最底层是感知层，通过各种信息感知设备来采集数据；中间是网络层，使用各种通信技术来传输感知层采集的数据；最顶层是应用层，对感知层获取的各种信息进行处理。图 2.1 显示了一般物联网系统的整体结构。

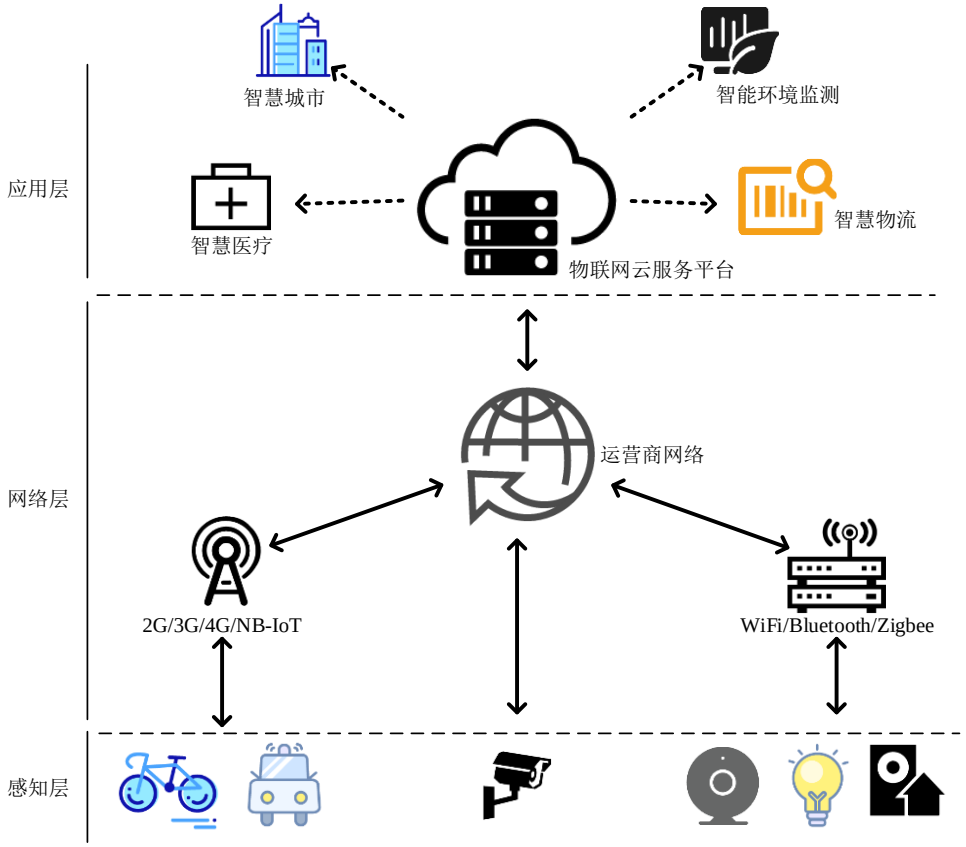

图 2-1 物联网架构示意图

物联网应用中最重要的就是数据，感知层作为信息和数据的来源，是构建物联网系统的基础。感知层中通常包含了以传感器为代表的感知设备、以 RFID（射频识别）为代表的识别设备、GPS（全球定位系统）等定位追踪装置、图像捕捉设备以及激光扫描设备等，并且各个设备节点之间能够组网形成无线传感网络，从而达到对数据全面感知的目的。

作为信息和数据的传输通道，网络层主要是通过移动通信网络、互联网、低功耗广域网等各类通信技术，将感知层采集到的数据传输到应用层，由应用层对感知层数据进行处理。

应用层的主要工作是借助各种数据分析方法对感知层采集的数据进行分析处理，例如通过数据挖掘手段，提取出用于控制和决策的有用信息，为智能电网、智能物流、远程医疗、智能家居等设施提供服务[22]，并且这些信息也可以通过网络层的传输，为感知层的终端设备提供反馈。

2.1.2 工业物联网系统典型架构

工业物联网可以视作是物联网的一个子集，在架构上也可以分为三个逻辑层。工业物联网典型的系统架构如图 2-2 所示。

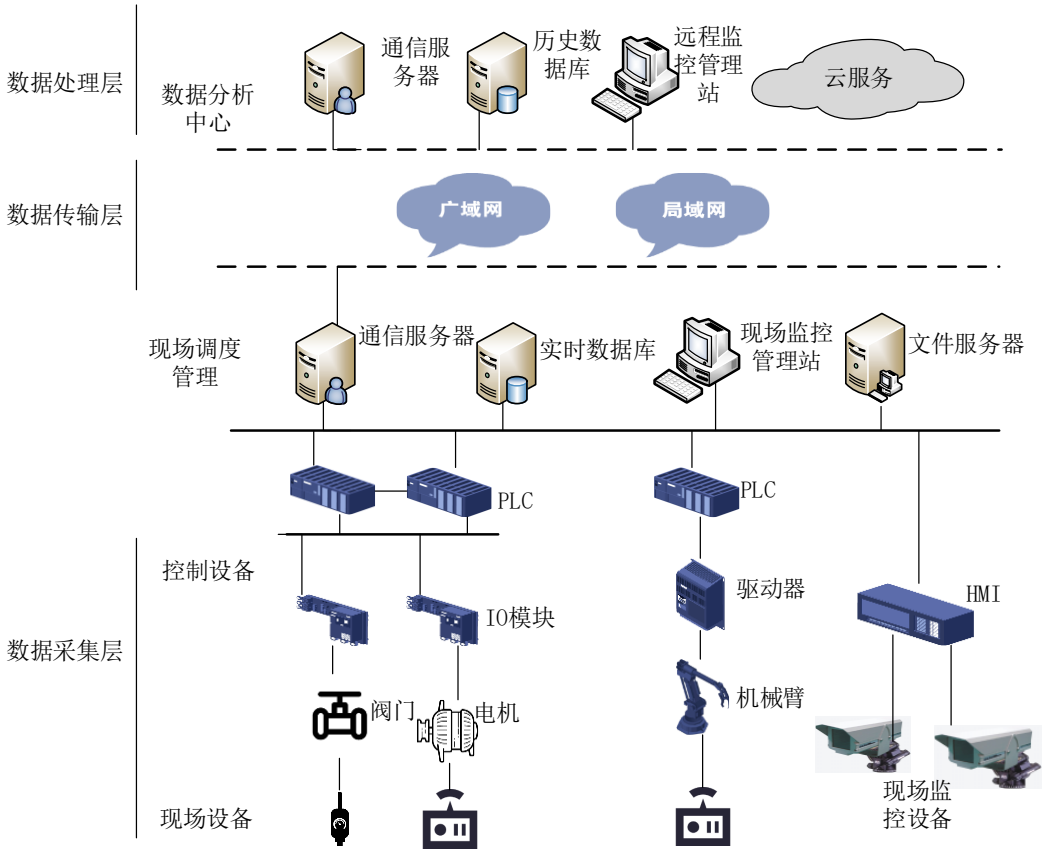

图 2-2 工业物联网系统架构示意图

在工业物联网中，工业控制系统为感知层提供了信息来源，这些信息包括设备运行的参数、监控数据等。对于工业物联网来说，感知层所采集的信息不仅仅是传感器数据，而是还包含了很多其它类型的数据，因此工业物联网中的感知层通常被称为数据采集层[23]。同样地，将工业物联网中的网络传输层称为数据传输层。在工业物联网系统中，一般不直接将处理后的感知数据提供给终端客户使用，而是反馈回工业控制系统当中，对数据的处理占据了主要过程，而数据的应用过程则相对单一，因此工业物联网中的应用层被称为数据处理层。

⑴ 数据采集层

在工业物联网系统的架构中，数据采集层主要包括了控制设备和现场设备。控制设备以 PLC 等控制器为代表，主要负责向各类工控设备下发控制指令，现场设备主要包括各类传感器，这些传感器直接与工控设备连接，负责采集工业现场的各种过程数据[24]。在数据采集层采集的生产数据可以为现场调度管理提供信息反馈，现场监控管理站能够根据数据处理层反馈回来的信息进行控制反馈，及时对生产设备的状态进行调整。

⑵ 数据传输层

工业物联网中的数据传输层与一般物联网的网络传输层是一致的，是一个融合了传感网络、移动网络和互联网的开放性网络，一般采用按国际标准或行业标准搭建其通信网络，如 Wi-Fi、蓝牙、RFID、ZigBee 等短距离的无线通信技术，传统的移动网络、互联网和低功耗广域网等，以及 MQTT 协议为代表的各类物联网通信协议[25]，并且一些专门针对物联网行业的低功耗广域网（LPWAN）也逐步得到应用[26]，为工业现场和远端的数据处理中心搭建起了数据传输通道。

⑶ 数据处理层

工业物联网的数据处理层主要包括通信服务器、历史数据服务器、远程监控端，并且和其他物联网应用领域中的数据处理中心一样，都采用了云计算平台，作为海量感知数据的存储和分析平台[27]。针对工业控制系统，工业物联网的云平台需要一些工业系统专用的处理过程和应用软件，例如需要对工业数据的完整性和机密性进行检查，对现场终端设备的身份进行鉴别。

工业物联网系统成为联接互联网和工业控制网络的桥梁，各种通信协议起到了非常关键的作用。工业物联网系统中的通信协议规约分为接入协议和通讯协议这两类，接入协议通常实现工控设备之间的通信、物联网感知设备与工控设备间的通信等，这一类协议已经发展的比较成熟，例如 Modbus、Profinet 等常见的工控协议；通讯协议一般是以互联网 TCP/IP 协议为基础的应用层通讯协议，物联网设备通过这一类协议接入到互联网，例如物联网感知设备与远程监控服务端、数

据库的通信[28]。

不同于传统的物联网通信环境，工业物联网系统要求感知设备与互联网之间的通信时保证数据传输的高可靠性和实时性，因此通讯协议需要满足以下要求：通信的实时性、通信的准确性以及对工业环境的高适应性。在工业物联网发展初期阶段，大部分厂商选择通讯协议时采用了 HTTP 协议。然而 HTTP 协议最初是为 web 浏览器设计的，适合 PC 端、智能手机等移动终端应用，在工业物联网应用中使用 HTTP 协议会有以下缺陷：工业环境中需要主动且频繁地推送数据时，HTTP 在技术实现上成本很高，并且数据传输的实时性也得不到保证；实现 HTTP 对终端设备的硬件资源有一定要求，对于运算能力和存储资源都十分有限的设备来说是一大障碍。

由于物联网系统对于通信协议的特殊需求，IBM 研发了适用于物联网数据传输的 MQTT（Message Queuing Telemetry Transport，消息队列遥测传输）协议。不同于 HTTP 协议的请求/回答模式，MQTT 协议采用了发布/订阅的通信模式，所有的物联网终端设备都通过 TCP 连接到云端服务器，云端通过主题的方式管理每个设备所订阅的通信内容，负责在设备与设备之间转发信息[29]。考虑到计算能力和存储资源受限的物联网设备，MQTT 协议在设计时采用了二进制格式编码，对于低功耗和低速网络也有很好的适应性，易于开发和实现。MQTT 协议还提供了非常完善的 QoS（服务质量）机制，根据业务场景可选择不同的消息送达模式，有效控制数据的可靠性、实时性和生存时间[30]。

基于上述优点，感知设备通过 MQTT 协议可以很方便地把数据传输到物联网云平台，由物联网云服务存储和处理数据，然后应用到各种业务场景中，如图 2-3 所示。目前非常多的工业物联网应用都采用了 MQTT 协议，GE（通用电气）、西门子等工控厂商都提供了支持 MQTT 协议的工业云平台服务，并且现有的基于 MQTT 协议通信的工业物联网设备数量已经达到了千万量级[31]。

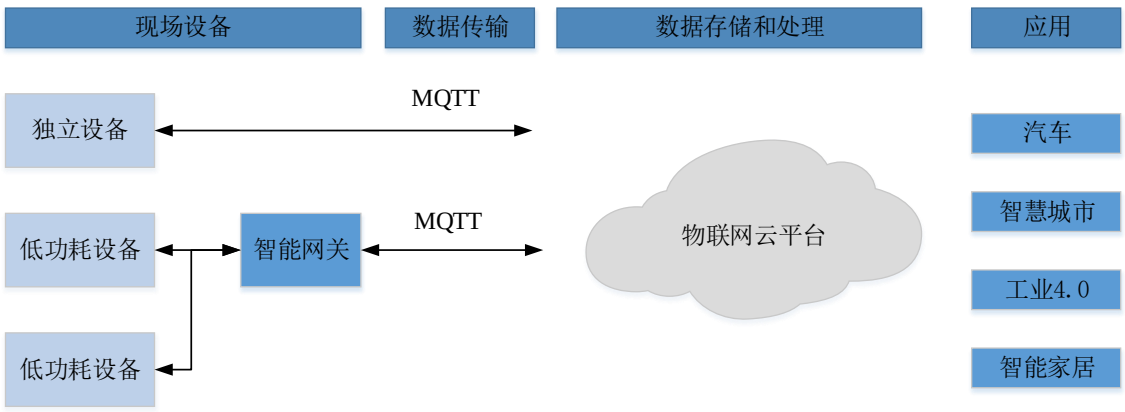

图 2-3 MQTT 协议应用场景

2.2 工业物联网安全威胁分析

2.2.1 安全威胁概述

工业物联网在发展过程中，根据不同的业务需求，融合了 IT（Information Technology，信息技术）、OT（Operation Technology，操作技术）和 CT（Communication Technology，通信技术）中的相关技术[32]，而这些新技术本身的安全防护措施并没有经过安全评估或可靠的验证，给整个系统带来了安全隐患。因此，工业物联网系统存在大多数信息系统中的脆弱性，一些针对互联网、工业控制网络的攻击也逐渐成为工业物联网系统的主要安全威胁[33]。同时，工业物联网系统中的安全漏洞也呈现出逐年增长的趋势。根据 ICS-CERT 的漏洞统计，2014 年至 2015 年工业物联网暴露出的安全漏洞增长率为 49%，而到了 2016 年，仅仅前两个季度工业物联网基础设施漏洞就增长了 60%以上。

为了对工业物联网系统进行比较全面、有效的防护，需要对目前工业物联网系统中已经存在的安全威胁进行深入分析，同时也要尽可能地挖掘出潜在的安全隐患，根据这些安全威胁和安全隐患，设计出适用于工业物联网系统的安全防护方案。由于工业物联网系统没有一个完全统一的标准，在不同的应用场景下其具体实现有所差别，但其系统架构是基本一致的，都是由数据采集层、数据传输层和数据处理层三个层次组成，所以本文以工业物联网中应用最为普遍的数据采集与远程监控系统作为仿真环境，对工业物联网系统进行安全性分析。根据图 2-2 所示的工业物联网系统的典型架构，搭建了符合其体系结构的工业物联网数据采集和远程监控系统模拟环境，如图 2-4 所示。

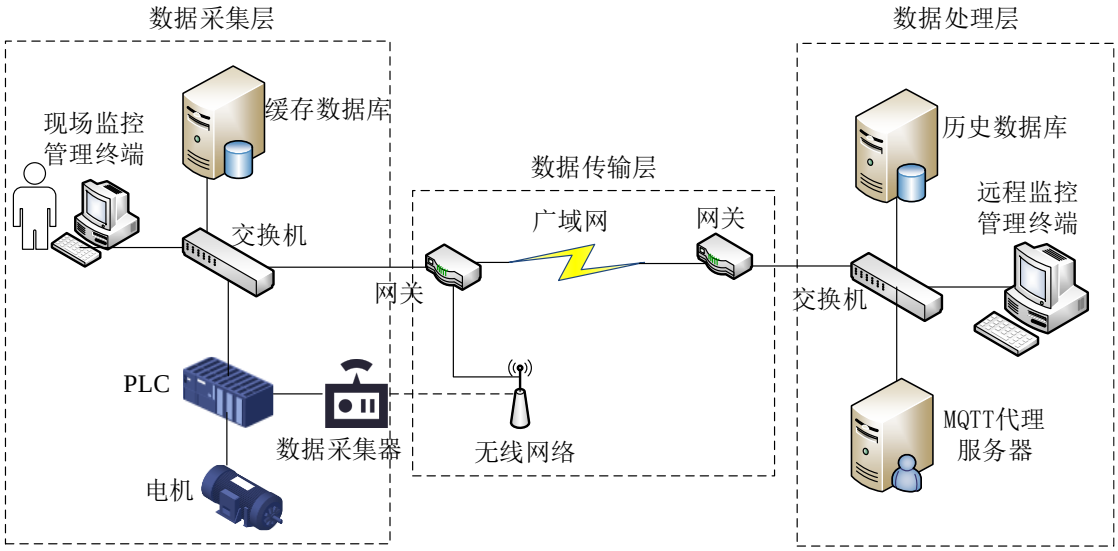

图 2-4 工业物联网系统仿真环境搭建

在以上环境中，数据采集层中的数据采集器相当于物联网终端设备，通过无线网络接入到互联网。该数据采集器负责采集 PLC 的控制数据，通过 MQTT 协议将数据传输到数据处理层中，MQTT 代理服务器将接收到数据存储到历史数据库中。当数据采集器出现网络异常时，会将数据暂时存储到缓存数据库中，网络恢复正常再上传到历史数据库。现场监控管理终端和远程监控管理终端主要负责对数据采集器进行管理，主要包括网络配置、数据采集配置等。

2.2.2 安全威胁分析

工业物联网所面临的信息安全威胁主要源自于引入的其他技术以及自身的结构特点。数据的采集、传输及处理技术作为工业物联网系统底层的核心技术，构成了工业物联网最基本的体系架构，同时这些技术方案中存在的安全漏洞也成为了工业物联网系统中的“安全短板”，可能使整个工业物联网系统暴露在网络风险之中。

根据近年来黑客针对工业物联网系统发起的攻击，并结合图 2-4 所示的工业物联网数据采集和远程监控仿真环境，本文总结了工业物联网系统所面临的一系列安全威胁，如图 2-5 所示。

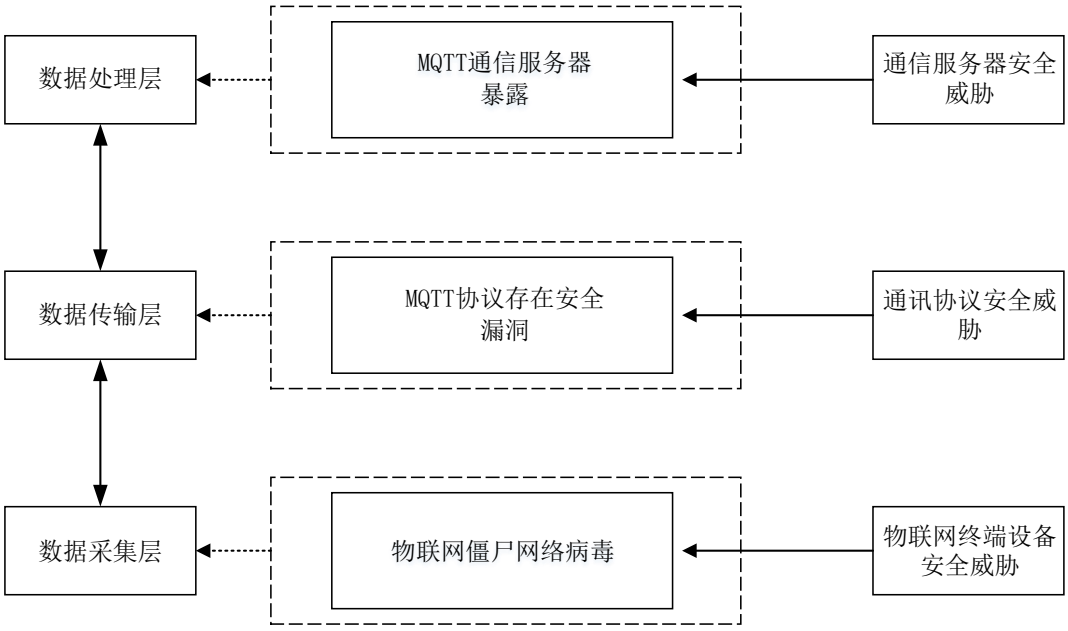

图 2-5 工业物联网系统面临的安全威胁

1、数据采集层的安全威胁

工业物联网数据采集层的主要功能是实现对信息的识别、采集或控制，由各类传感器、RFID 装置、智能终端设备组成，是整个工业物联网系统的基础层[34]。

由于考虑到成本的因素，数据采集层的终端设备功能比较单一，其资源、计算能力受限，攻击者通常采用暴力字典破解攻击的方式来控制物联网终端设备，从而入侵到工业物联网内部网络，进行非法的行为或发起恶意攻击，如对设备数据或指令进行监听、篡改、伪造[35]，或是操纵物联网僵尸网络发起 DDoS 攻击等。

目前工业物联网终端设备所面临的最为危险的安全威胁来自于物联网僵尸网络病毒。通过使用该类病毒，黑客能够利用物联网终端节点本身的脆弱性以及节点之间互联互通的特性，通过自动化脚本组合出可能正确的用户名和密码来破解终端节点的账户名、密码等敏感信息，然后篡改终端设备的软硬件配置，该终端设备就成为了一个僵尸节点。攻击者不断破解更多的终端设备，就可以构建一个庞大的僵尸网络，而这些僵尸网络主要被用来发起 DDoS 攻击，或者向其他物联网设备发起暴力破解攻击，扩大僵尸网络的规模[36]。由于目前的物联网终端节点几乎都不会部署防御僵尸网络病毒的保护机制，因此同一个病毒样本，只要能够与设备硬件平台和操作系统匹配，就可以对设备进行感染。通常，攻击者只需要在远程服务器上准备好一批针对不同设备的恶意代码，并在僵尸网络病毒的入侵程序中加入平台环境的判断就可以轻易地实现大范围、跨平台的攻击。

据安全厂商统计，黑客操纵僵尸网络发起的 DDoS 攻击占据全球 DDoS 攻击的一半以上，美国、中国、德国和俄罗斯由于存在大量缺乏安全性的物联网设备，因此成为遭受物联网僵尸网络病毒的重灾区，这些设备的占比如图 2-6 所示。在 2017 年，全球物联网僵尸网络呈现出井喷式的增长趋势，以 Mirai 为首的物联网僵尸网络病毒衍生出了众多僵尸网络病毒的变种。

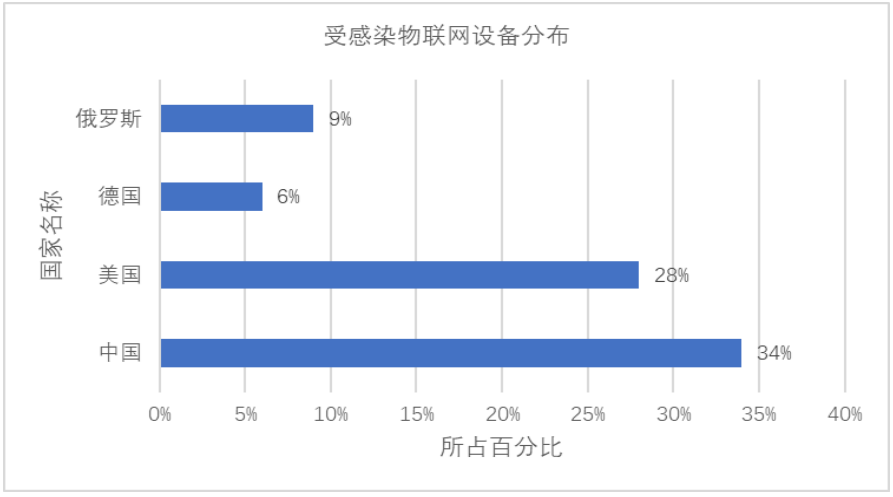

图 2-6 受 Mirai 感染的物联网设备分布

2、数据传输层的安全威胁

物联网通信协议存在的安全漏洞成为工业物联网数据传输层最大的安全隐

患。目前，工业物联网系统所采用的通信协议没有特殊的规定，通常不同的厂家有着各自的标准。不同的物联网通信协议有着不确定的风险，有可能其加密措施比较完备，也可能根本就没有进行任何加密[37]。倘若使用的物联网产品，其采用的通信协议既简单又是明文传输，那么攻击者一旦入侵到物联网系统局域网中，就可以很轻松的采用网络嗅探等方法来拦截通信数据[38]。如图 2-7 是数据传输层遭受数据窃取和篡改的示意图，由于通信协议没有加密，攻击者可以很轻易地检测到数据内容，并可以篡改、伪造数据包内容[39]，或者通过中间人攻击的方式，发送错误的控制指令，可能造成工业设备无法正常运转。

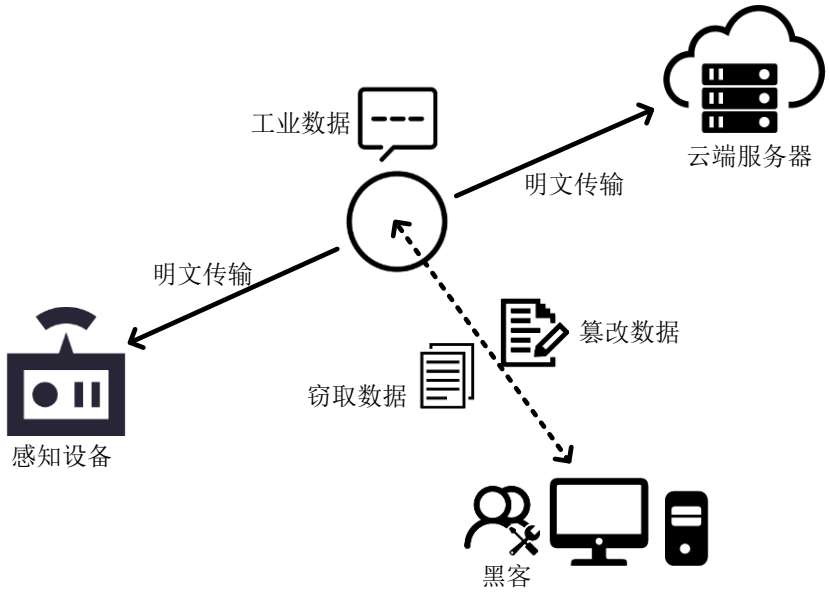

图 2-7 信息窃取和篡改

目前，大量的工业物联网设备接入互联网后，与云端服务器通信所使用的协议有 MQTT、AMQP、XMPP 等，其中 MQTT 协议的应用最为广泛，亚马逊、IBM、阿里云、腾讯云、中移物联等国内外主流的物联网云服务提供商均支持该协议，研华科技、ABB、GE、西门子等工业自动化厂商也都提供了基于该协议的工业物联网开发平台。MQTT 协议完全开放，如果通信过程不采取加密措施，容易发生信息劫持、篡改以及中间人攻击。在图 2-4 所示的工业物联网系统仿真环境中，数据传输层使用 MQTT 协议进行数据传输，既没有对该通信过程进行加密，也没有采取身份认证机制，攻击者入侵到该系统的局域网络以后，可以使用网络嗅探工具抓取到 PLC 的控制信息的明文数据。同时，攻击者只需将伪装的 MQTT 客户端连接到 MQTT 服务器，就能发送伪造的指令给 MQTT 服务器，造成服务器工作异常。

根据以上分析，MQTT 协议的安全漏洞体现在缺乏通信数据加密和设备身份认证这两个方面。MQTT 协议通信过程如果没有加密，采用明文传输数据，攻击者一旦截获到通信数据，就可以轻易地解析出通信内容，为后续的攻击埋下伏笔。MQTT 代理服务器也支持 TCP 协议，意味着攻击者可以使用伪装的 MQTT 客户端进行中间人攻击，发布各种虚假的信息或伪造的控制指令，由此可能会导致工业设备的损坏、系统瘫痪等严重后果，给生产和安全带来严重威胁。因此，没有采取任何安全措施的 MQTT 应用环境都将面临巨大的安全风险。

3、数据处理层的安全威胁

工业物联网数据处理层通常是一个提供了一个计算和存储数据服务的物联网云平台，对数据采集层所收集的数据进行综合、分析、整理等操作[40]，通常由数据库服务器、通信协议代理服务器、远程监控平台组成，其中通信服务器是整个数据处理层的信息中转中心，集中了大量的信息资源。因为工业物联网系统中的终端设备一般都部署在网关后面，无法直接对外提供服务，系统管理员为了在外网实现对设备的控制，需要设备与数据处理层中的云端通信服务器建立长连接[41]，而云端的通信服务器为了保证设备可以随时连接，必须时刻保持开启状态，所以这些云端的通信服务器必然会暴露在互联网上，这些缺乏足够的安全防护措施的通信服务器容易成为受攻击的目标。

据安全厂商统计，目前全球已经有大量运行 MQTT 等面向物联网的通信协议的云服务器暴露在互联网上，并且其暴露数量呈现上升趋势，图 2-8 是暴露 MQTT 服务器的主要国家分布示意图，从中可以看出中国暴露的 MQTT 服务数量最多。

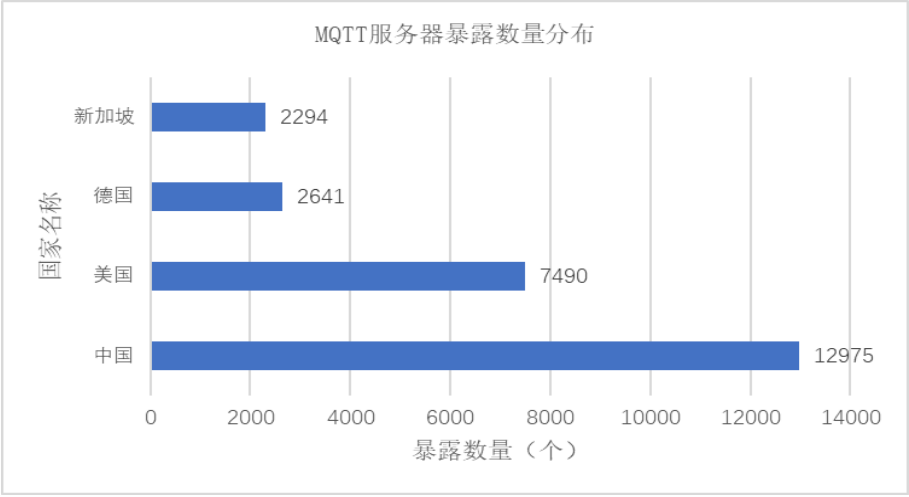

图 2-8 暴露 MQTT 服务器的主要国家分布

在 2017 年的美国黑帽大会（Black Hat USA）上，安全研究人员通过互联网扫描发现中东地区某石油管道物联网系统的 MQTT 云服务器暴露在互联网上，这些

通信服务器中只有两台使用了验证保护措施[42]。安全研究人员指出，攻击者只需要伪造大量非法的 MQTT 控制报文，并将伪造的控制报文发送到这些 MQTT 服务器，就能够对这些服务器的正常运行造成极大的影响，例如扰乱检测仪上的传感器数据，或者随意变更管道中的油流量大小。除石油管道服务器外，研究人员还发现了包括监狱门禁、联网汽车、智能电表和各类工业自动化系统中大量开放的 MQTT 服务器，并且通过相关的黑客工具就能够通过 MQTT 服务器来轻易地控制监狱闸门的打开和关闭。

在图 2-4 所示的工业物联网系统仿真环境中，数据处理层的 MQTT 服务器完全暴露在互联网环境上，如果数据传输层没有对 MQTT 传输过程进行身份认证或者攻击者找到某种方法绕开了身份认证机制，那么攻击者就可以向 MQTT 服务器发送具有异常特征的欺骗性数据包，可能造成服务器无法正常工作。对于这一类基于欺骗性数据包的攻击，通常会使用传统的防火墙或者是简单的访问控制技术来进行防御，但是这一类防护方法只能够检测到具有明显异常特征的伪造数据包攻击，对于符合协议规约和访问控制规则的伪造数据包，能够很容易的绕过防火墙的检测[43]，传统的防火墙不能有效地发现其攻击行为，所以需要采取实时的全流量分析方法，例如深度包检测和异常流量检测措施，来防范此类攻击行为。

2.3 工业物联网安全防护策略设计

根据对工业物联网系统分析可知，工业物联网目前存在比较严重的信息安全隐患。针对其安全威胁，对工业物联网系统各个层次设计出相应的安全防护策略，对数据采集层到数据传输层，最后到数据处理层进行防护，最终构成一套完整的安全防护框架，为工业物联网系统的安全稳定运行提供良好的保障。

2.3.1 物联网安全设计原则

在物联网项目的实施过程中，有许多的安全设计原则和解决方案可以帮助保护物联网系统的安全。关于物联网安全措施设计方面，微软给出了以下几点实践建议：

⑴ 遵循安全的软件开发方法。开发安全类应用软件，需要端到端的安全思考，包括平台的选择、编程语言、开发工具、实现过程、测试方案等。

⑵ 明智的选择开源软件。如果选择开源软件来构建安全解决方案，需要评估该软件是否满足兼容性、扩展性和替代性等需求，其次还需要考虑到该软件是否有比较活跃的社区或商业支持。

⑶ 构建多层次的安全防护方法。多层安全性是设计整体安全解决方案的核

心，在资源允许的范围内应该根据系统自身的架构设计多层安全性，假设其中一个层次的安全防护被攻击者破坏，还有其他层面的安全措施为系统提供保护，同时各个安全层面之间应该具备较低的耦合性，保证对于不同应用场景下的适应性。

⑷ 集成和改进现有的安全技术。为了避免“重复造轮子”，可以引入其他领域中行之有效的安全技术，并结合工业物联网安全问题的实际需要，对一般的网络安全技术进行融合、改进，从而解决工业物联网的安全威胁。比如，互联网环境当中使用的流量分析技术，比如深度包检测和异常流量检测方法，通常只是在网络层和传输层对数据包（TCP/IP 数据包）进行解析[44]，但是对于工业物联网环境而言，这一类流量分析方法还需要解析的物联网应用中的协议，比如广泛使用的 MQTT 等协议。

2.3.2 工业物联网安全防护框架

根据国际数据公司（IDC）发布的《2017 年物联网（IoT）投资支出预测》，在 2020 年之前工业物联网的应用主要集中在制造业、物流和交通运输业、能源和公共电力事业这三大领域，具体集中在生产设备互联及状态监测、远程监控及数据采集、仓储物流管理、能耗自动监测等方面，虽然工业物联网在以上行业中的具体应用场景下有不同的拓扑结构和实施方法，但是其系统架构是一致的，都是由图 2-2 所示的工业物联网典型系统架构中的数据采集层、数据传输层和数据处理层组成。为了设计适用于不同工业物联网应用场景下的安全防护策略，本文以层次化的进行安全威胁分析和安全防护策略设计的思路，构建了如图 2-9 所示的工业物联网安全防护框架，该框架针对各个层次结构分别设计了相应的安全防护措施。

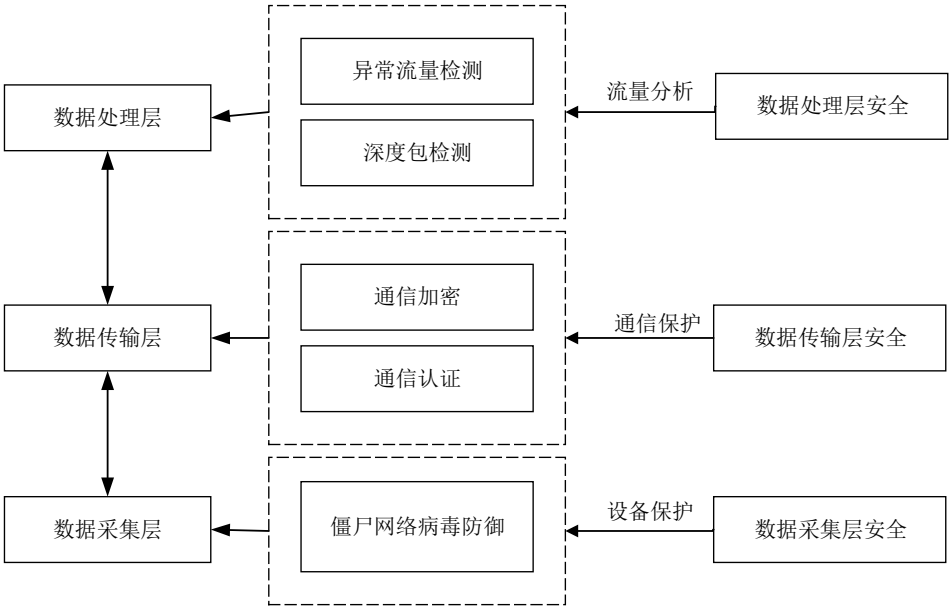

图 2-9 工业物联网安全防护框架示意图

如上图所示，数据采集层安全、数据传输层安全和数据处理层安全构成了工业物联网系统的安全框架。根据对工业物联网系统的安全威胁分析，本文设计了如下针对工业物联网系统不同层次的安全防护措施：

1、数据采集层安全

数据采集层安全要保护的是物联网终端设备的安全，为避免攻击者通过物联网僵尸网络病毒侵入工业物联网内部网络，需要对物联网僵尸网络病毒的攻击原理进行深入分析，比如物联网僵尸网络经常采用的端口扫描、暴力字典破解攻击等，并针对这些攻击手段采取有效的防御措施，保障终端设备不被劫持，提升了终端设备的安全性。

2、数据传输层安全

数据传输层安全主要是保障通信过程中数据传输的安全性，包括通信的终端设备的身份认证以及数据机密性。为保证数据传输层提供安全准确的数据传输服务，防御数据窃取和中间人攻击，对于大部分工业物联网系统采用的 MQTT 通信协议，需要对其通信过程进行加密传输，并且对使用 MQTT 协议接入的终端设备进行可信身份认证。

3、数据处理层安全

数据处理层安全的重点在于通过流量分析的方法保护通信服务器的安全。对于工业物联网系统中广泛使用的 MQTT 代理服务器，为了避免其遭到欺骗性数据包的攻击，采用 MQTT 协议深度包检测防火墙对 MQTT 代理服务器进行防护，根据自定义的访问控制规则过滤掉具有明显异常特征、不符合 MQTT 协议规约的通信流量。同时，对于构造精良的伪造数据包发起的高级持续性攻击，使用针对 MQTT 协议的入侵检测系统，对通信代理服务器与终端设备的网络通信流量进行实时监测，一旦发现异常立即报警，在早期即可发现攻击行为，防止攻击规模的扩大。

该工业物联网安全防护框架贯穿了整个数据采集层、数据传输层和数据处理层三个层次，构建了多层次的安全防护措施。该框架每个层次的安全防护措施互相独立、不具有耦合关系，对于特定的工业物联网应用，可以根据具体应用场景自身的特点，参考该防护框架有针对性地部署相应的安全防护措施。例如，对于工业物联网设备提供商来说，主要关注的是物联网终端设备的安全，所以可以在终端设备引入数据采集层的物联网僵尸网络病毒防御措施来提升终端的安全性；对于工业物联网系统开发平台的提供商来说，关注的重点在于云平台上通信服务器的安全以及终端设备与云端通信服务器的连接是否安全，因此可以引入数据传输层的通信加密技术和身份认证措施来保障数据安全，以及在数据处理层中部署

深度包检测和入侵检测防御措施来保障通信服务器的安全。

2.4 本章小结

本章主要对工业物联网系统进行了安全威胁分析，并设计了适用于工业物联网系统的安全防护框架。首先对工业物联网系统的典型架构进行了分析，研究了该典型架构中各个层次的主要功能以及所使用的通信协议；然后根据工业物联网系统典型架构，搭建了对应的工业物联网仿真环境，并结合该环境对工业物联网系统存在的安全威胁进行了分析，分别指出了数据采集层、数据传输层以及数据处理层中存在的安全隐患；最后针对该系统三个层次所面临的安全威胁，构建了适用于工业物联网系统的安全防护框架，该框架分别对系统架构的数据采集层、数据传输层和数据处理层进行安全防护策略研究，设计出针对具体安全威胁的安全防护措施。

第三章 数据采集层安全防护策略设计及实现

本章针对工业物联网系统数据采集层存在的安全隐患，主要从防御物联网僵尸网络病毒这个方面进行安全防护研究。首先介绍了物联网僵尸网络，主要分析了以 Mirai 为代表的物联网恶意软件的运作原理，并针对物联网僵尸网络病毒对物联网设备进行感染的逻辑，设计了针对端口扫描以及暴力字典破解攻击的防御方法，保障物联网终端设备的安全。

3.1 物联网僵尸网络

物联网僵尸网络是一种结合了病毒、木马以及蠕虫技术为一体的新的信息安全威胁方式，创造和管理僵尸网络的恶意软件不仅可以在物联网终端设备中进行常规的系统驻留、数据窃取、远程劫持[45]，还具有类似于蠕虫病毒的网络感染特性，而物联网的迅猛发展更是加速了这种恶意软件的传播。物联网僵尸网络最大的特点是攻击者不需要直接登录僵尸客户端，就能够在僵尸客户端上执行一系列的攻击操作，大量的僵尸客户端能够协同完成任务，比如对同一个目标发起 DDoS 攻击。图 3-1 是物联网僵尸网络的工作流程。

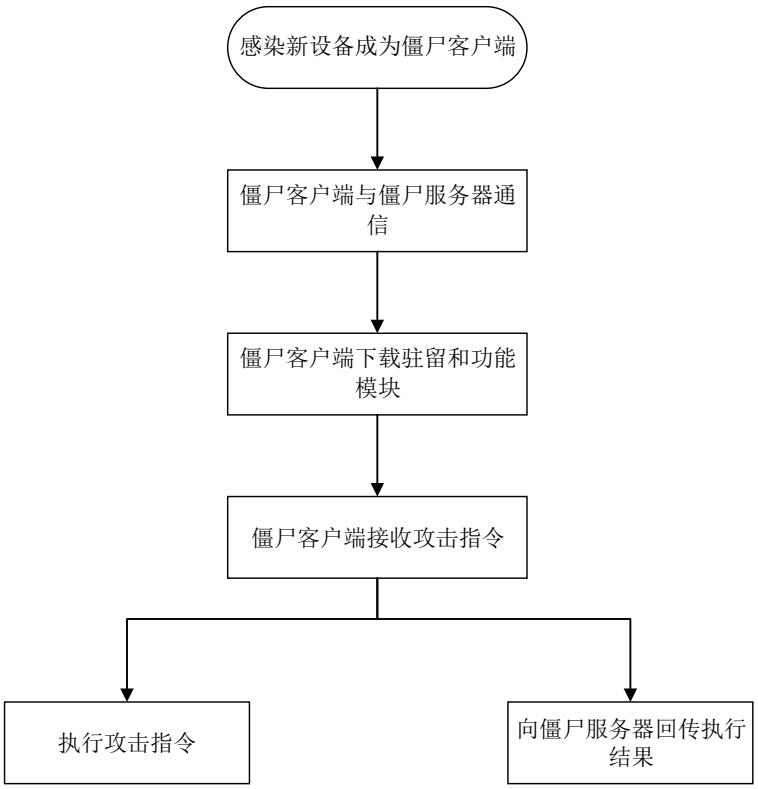

图 3-1 物联网僵尸网络工作流程

一个完整的物联网僵尸网络中主要包含了僵尸服务器和僵尸客户端，并且僵尸客户端的数量十分惊人，即使是一个小型的僵尸网络也可能包含了成百上千的僵尸客户端。从功能结构上来看，僵尸网络一般情况下由扫描模块、驻留模块和功能模块构成。其中，扫描模块用来扫描物联网设备并传播僵尸网络病毒，驻留模块的功能在于隐匿在物联网设备中，功能模块则用来执行攻击者下发的攻击指令。

通常情况下，攻击者会利用物联网系统自身存在的安全漏洞，通过“钓鱼方式”或者自动化的扫描破解工具来对物联网设备进行破解，比如通过常用的登录弱口令来暴力猜解物联网设备的登录用户名和密码，一旦破解成功，便可以从僵尸服务器植入恶意代码，将物联网设备劫持为僵尸客户端。这些僵尸网络主要用来发动 DDoS 攻击，即利用大量的僵尸客户端对其他网络发起泛洪攻击，或者用来感染其他物联网系统成为新的僵尸网络[46]。

物联网僵尸网络的这种入侵攻击方式，一方面是因为物联网终端设备自身存在安全漏洞，攻击者针对这些安全漏洞找到了破解设备的有效方法；另一方面是因为物联网设备厂商和用户没有重视安全问题，在生产或使用物联网设备的过程当中没有强化或更改设备登录口令，致使设备遭到攻击者的破解和控制。表 3-1 是目前物联网设备最常采用的一些登录弱口令。

表 3-1 物联网设备常用登录弱口令

用户名（username）	登录密码（passwords）
root	Admin
Admin	Root
root	123456
admin	123456
test	1234
ubnt	12345
access	Ubnt
pi	Raspberry

3.1.1 Mirai 恶意软件简介

世界上的僵尸网络的规模相当庞大，并且还在不断的扩张，目前已经存在大量针对物联网的僵尸网络病毒，如 QBOT、Luabot、KTN-RM 等等，其中最具代表性、影响力最大的当属 Mirai 恶意软件。2016 年 9 月，黑客 Anna-senpai 在 GitHub 网站上公开发布了 Mirai 恶意软件的源码。同年的 10 月，美国域名解析服务提供

商 Dyn 公司就遭受到了史上最严重的 DDoS 攻击，造成美国东海岸地区大面积网络瘫痪。Dyn 公司经过分析，确认此次攻击流量来源于受 Mirai 病毒所感染的物联网设备。

Mirai 病毒的源码在公开之后，黑客利用该病毒感染了大量的物联网设备，并构建了一大批物联网僵尸网络。网络安全厂商 Imperwa Incapsula 通过调查到的感染设备分析发现，当前全球感染 Mirai 病毒的物联网设备已经超过百万台，感染范围跨越了一百多个国家和地区，其中感染量最多的是俄罗斯、美国和中国大陆[47]。

3.1.2 Mirai 攻击原理分析

目前大多数物联网僵尸网络病毒都是源自于 Mirai 恶意软件的变种，其工作流程都基本相同，因此本节以 Mirai 恶意软件为例，具体分析物联网僵尸网络病毒的攻击原理。

在物联网设备运行时，如果用户没有及时更改设备的默认登录口令，攻击者便可以针对这些设备进行登录破解。Mirai 在端口扫描和感染设备等方面进行了改进，比如采用了高级 SYN 端口扫描，大幅度提升了扫描物联网设备的速度，提高了感染效率，并且 Mirai 还具有跨平台的特性，能够针对几乎所有的主流硬件平台进行攻击，如 x86、ARM、MIPS、PowerPC 等，囊括了大部分主流的处理器架构。为了有效地对 Mirai 病毒的攻击进行防御，需要对 Mirai 的组成及其感染物联网设备的逻辑进行研究和分析。

如图 3-2 所示，Mirai 的源码主要包含了 loader 和 mirai 两个文件夹，其中 loader 文件夹中是加载器，mirai 文件夹中的程序完成主要的入侵行为，包括僵尸客户端连接、执行攻击指令、下载恶意程序等功能。

```
Mirai_Source_Code
├─loader            # 加载器
│  ├─bins           # 一部分二进制文件
│  └─src            # 加载器的源码
│      └─headers
└─mirai             # 病毒本体
    ├─bot           # 攻击、扫描器、域名解析等模块
    ├─cnc           # 使用 go 语言写的服务器程序
    └─tools         # 存活状态检测、加解密、下载文件等功能
```

图 3-2 Mirai 源码目录结构

根据 Mirai 的源码目录结构，可以将其分为以下四个模块组成：

⑴ CNC 模块，主要用来管理遭到感染的设备，向僵尸客户端下发 DDoS 等攻

击指令；

⑵ Bot 模块，运行在受感染设备中的主要程序，主要是对网络设备进行弱口令扫描破解，接收下发自 CNC 模块的攻击指令，对目标发起网络攻击；

⑶ ScanListen 模块，主要用来接收 Bot 模块弱口令扫描得到的设备信息，包括 IP 地址、可用端口、用户名和登录密码，并将这些信息发送给 Load 模块进行处理；

⑷ Load 模块，主要用来接收 ScanListen 模块发送过来的设备信息，并将恶意程序下载到设备当中。

图 3-3 为 Mirai 僵尸网络的结构图，配置服务器运行 Load 和 ScanLoad 模块，CNC 控制服务器主要运行 CNC 模块，而 Mirai 文件服务器主要用来提供 Bot 恶意程序下载，Bot_n 是受到感染的物联网终端设备。

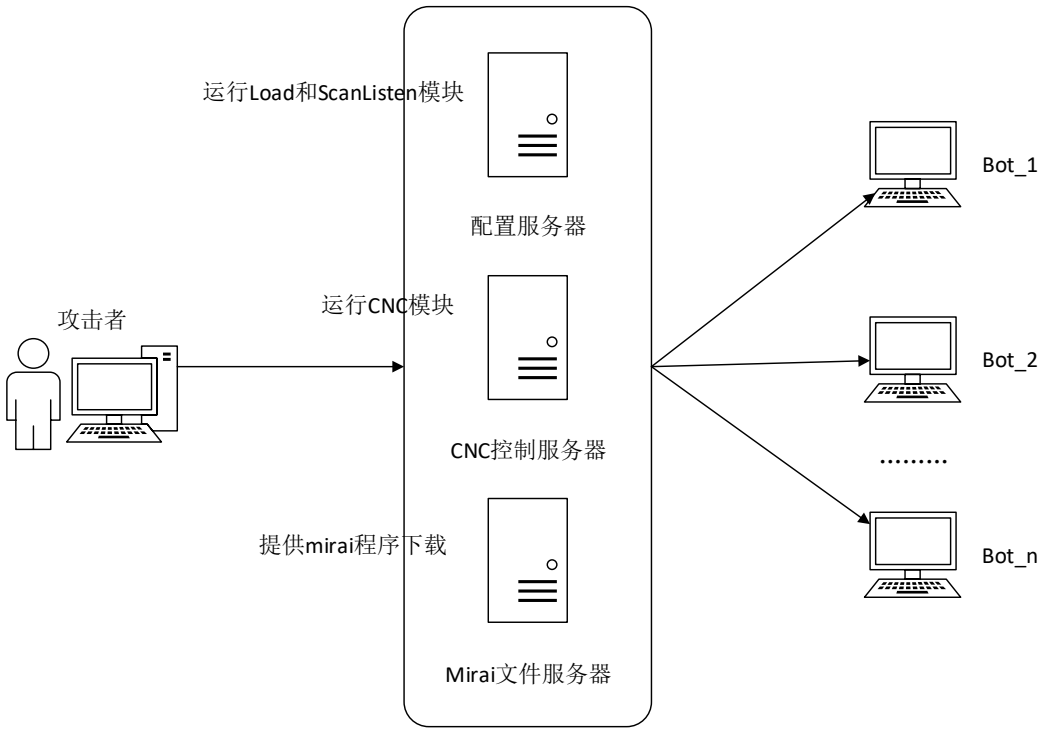

图 3-3 Mirai 僵尸网络结构图

图 3-4 显示了 Mirai 僵尸网络的感染过程。其中 Load 模块采用 Telnet 或 ssh 远程登录的方式对互联网上的物联网设备进行弱口令暴力破解，获取设备远程登录的账号信息，然后将获取到的登录账号信息传递给 ScanListen 模块进行解析，并使用这些信息来登录相关设备，然后通过 echo、wget 或 tftp 三种方式中的其中一种，向目标设备推送一个具有下载功能的 dvrHelper 微型模块，最后由 dvrHelper 模块从 Mirai 文件服务器下载恶意 Bot 程序并执行。

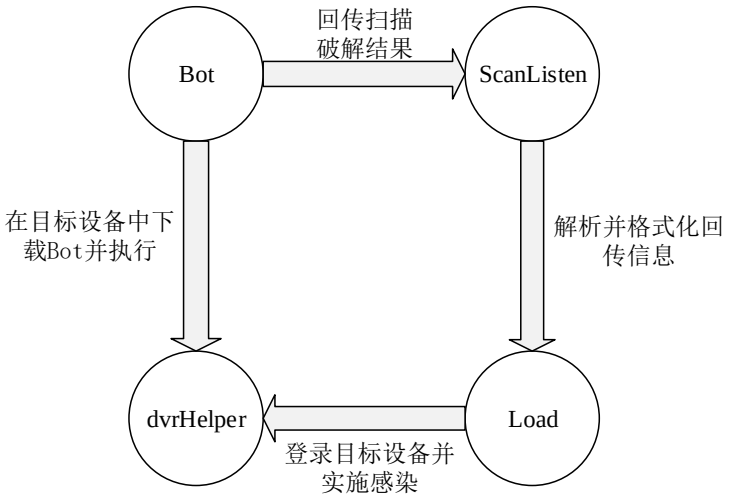

图 3-4 Mirai 感染物联网设备的过程

与普通的僵尸网络不同的是，Mirai 的感染过程不仅可以通过黑客服务端来发起，还可以依靠已经遭到感染的设备来实施。受到 Bot 恶意程序感染的物联网设备成为僵尸客户端后，也会采取一定策略扫描互联网上的其他物联网设备，对开启了 Telnet、ssh 等远程连接服务的设备进行暴力破解，将成功破解的设备 IP 地址、可用端口、用户名和密码等信息发送给配置服务器的 ScanListen 模块，由此周而复始的在网络上进行传播，提高了僵尸网络的扩展速度。

Bot 是 Mirai 的攻击模块，遭到 Bot 模块感染的物联网设备会自动连接到 CNC 服务器，并等待接收 CNC 服务器的控制命令对目标发动攻击，如下图 3-5 所示。

```
[main] We are the only process on this system!
listening tun0
[killer] T[rmyaiing to kni]l lA tptoermtp 23
ting to con[nkeicltl etro]  CFNiCn
ding and killing processes holding port 23
[scanner] Scanner process initialized. Scanning started.
[resolv] Got response from select
[resolv] Found IP address: d70al0ac
Resolved Fconucn.dm iirnaoid.ec o"m4 8t6o6 01"  IfPovr4  paodrdtr e2s3s
es
[main] Resolved domain
[main] Connected to CNC. Local address = 235540652
[killer] Found pid 8439 for port 23
```

图 3-5 受感染设备连接到 CNC 服务器

黑客利用 Mirai 构建物联网僵尸网络的主要目的在于盈利，因此为这类僵尸网络设置了僵尸网络管理员和普通用户两种角色，僵尸网络管理员可以将一定数量的僵尸网络售卖给普通用户，每个普通用户有自己登录僵尸网络 CNC 服务器的用户名和密码。攻击者使用普通用户账号登录 CNC 控制台后，就可以向连接到 CNC

的受感染物联网设备下发攻击指令了，攻击类型包括 UDP flood 攻击、http fload 攻击、syn flood 攻击、ack flood 攻击以及最新型的 GRE IP flood 攻击等。

3.2 终端设备防护策略设计

从上一小节对 Mirai 攻击原理的分析可知，攻击者使用物联网僵尸网络病毒攻破物联网终端设备并发动 DDoS 攻击的过程可分为三个阶段：

第一阶段，攻击者通过端口扫描，发现因业务需要或配置失误而被暴露在互联网上的物联网设备所开启的 SSH（22 端口）、Telnet（23 端口）、HTTP/HTTPS（80/443 端口）等服务，这类服务成为僵尸网络病毒感染设备的突破口；

第二阶段，攻击者对终端设备进行渗透，发现设备存在的弱口令漏洞，由于物联网设备厂商和用户在生产或使用物联网设备时没有对设备登录口令进行安全强化或者更改，使得僵尸网络病毒能够使用暴力字典尝试轻易地对设备实施破解；

第三阶段，终端设备沦为僵尸主机，成为攻击者控制的僵尸网络的一部分，接收 CNC 服务器的指令发动攻击。

因此，针对僵尸网络病毒攻击对物联网终端设备进行安全防护策略设计，关键在于防御上述的第一阶段中的端口扫描以及第二阶段中的暴力字典破解，因此本节将针对端口扫描和暴力破解进行检测和防护措施设计。

3.2.1 端口扫描检测及防护

端口扫描是攻击者收集攻击目标信息的重要步骤，通常会在入侵之前对目标主机的端口进行扫描，以确定哪些端口是开放的。攻击者从开放的端口可以猜测目标主机所开启的服务，进而找到目标主机可能存在的漏洞。常见的端口扫描方法有全 TCP 连接扫描、SYN 扫描、ACK 扫描、FIN 扫描等。

僵尸网络病毒在扫描设备可用端口时采用的是 SYN 扫描方法，该扫描方法最大的特点就在于不需要建立完整的 TCP 连接，其扫描过程如图 3-6 所示。

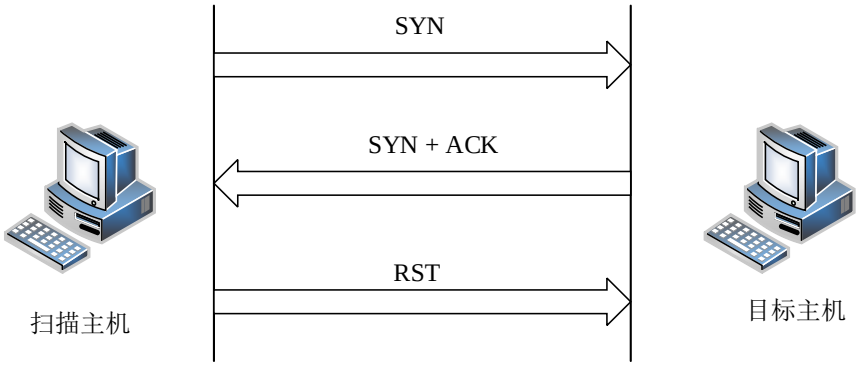

图 3-6 端口扫描检测和防御系统整体架构

扫描主机首先会向目标主机的某个端口发送连接请求的 SYN 包，目标主机如果开启了该端口，就会向扫描主机回复 SYN+ACK 的确认包，扫描主机收到该确认包后，不是发送 ACK 的连接应答，而是发送 RST 包来请求断开连接。SYN 扫描不需要完成 TCP 三次握手也能够探测到目标主机开放的端口[48]，因此其扫描速度比其他端口扫描方法快很多，具备强大的扫描感染能力。

对于端口扫描，目前比较有效的检测方法是部署入侵检测系统（NIDS）。众多的入侵检测系统中，在检测的准确率、响应速度以及实用性等方面表现最好的是开源的轻量级入侵检测系统 snort。snort 能够进行实时流量分析，快速检测到网络攻击并及时发出告警，然而其主要功能在于监测系统的网络运行状态，及时发现各种入侵企图、入侵行为和入侵结果[49]，对于各种入侵活动只能采取报警的方式，不能对攻击活动进行主动响应。

考虑到 snort 不能对攻击行为采取主动防御这一缺陷，本文采取将 snort 和 Linux 系统中应用广泛的 iptables 防火墙联动的方法来检测并防御物联网僵尸网络病毒的端口扫描。iptables 防火墙作为基于安全规则的防火墙，能够实现网络数据包过滤、网络地址转换以及对数据报进行处理的功能，其作用在于根据安全规则来控制不同网络或主机之间的访问控制，然而由于其安全规则是预先设定的静态规则，所以 iptables 不具备对入侵行为进行动态响应的能力。因此，通过将 snort 和 iptables 联动起来的方法可以克服它们各自的缺点，组成一个同时具备入侵检测和攻击防护能力的防御系统。

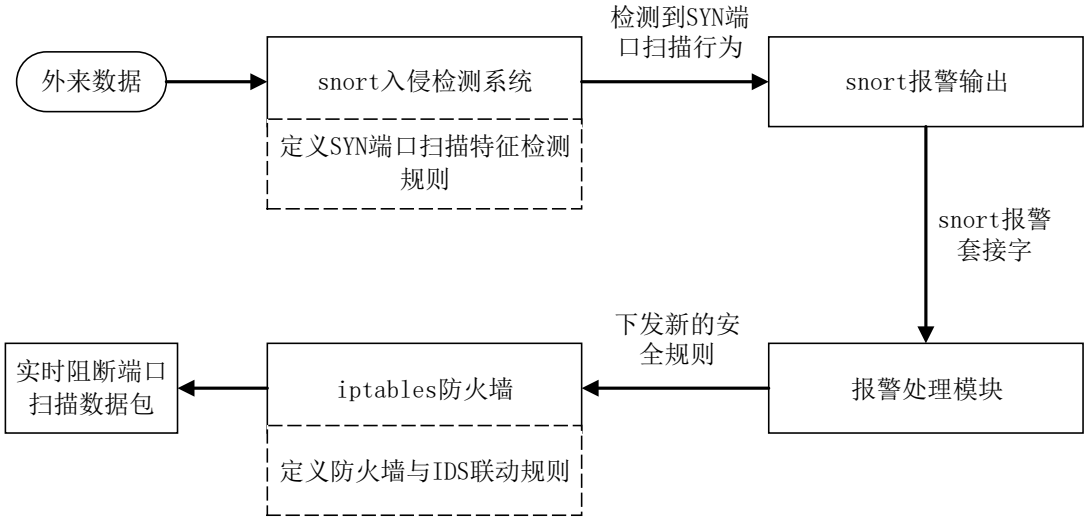

图 3-7 端口扫描检测和防御系统整体工作流程

采用 iptables 和 snort 联动来进行端口扫描检测和防御，其整体工作流程如图 3-7 所示，所有外来数据都会经过 snort 检测引擎进行检测，如果当前数据流量与

已定义好的 SYN 端口扫描特征检测规则是匹配的，则检测到端口扫描入侵行为，snort 会以套接字的方式将入侵报警信息输出到报警处理模块，报警输出模块对报警信息进行分析，并根据分析结果向 iptables 防火墙下发新的安全规则，iptables 防火墙基于此规则实时阻断来自攻击源的端口扫描报文，从而达到实时阻断攻击源的目的。

在上述 snort 和 iptables 联动的方法中，重点就在于为 snort 设计有效地端口扫描特征检测规则以及实现报警处理模块。

1、snort 检测 SYN 端口扫描

snort 的端口扫描功能是通过名为“sfportscan”预处理器插件来实现的，该预处理器的配置模型如下（#表示对该配置选项的注释）：

preprocessor sfportscan ：

proto <protocols> #需要检测的协议类型

scan_type <portscan | portsweep | decoy_portscan | all> #需要检测的扫描类型

sense_level <low | medium | high> #检测端口扫描的敏感度水平

watch_ip <IP> #指定需要检测的扫描主机列表

ignore_scanners <IP list> #指定需要忽略的扫描主机列表

logfile <path and filename> #指定端口扫描检测日志的存储路径

对于检测 SYN 端口扫描，需要将上述配置模型中的 proto 选项设置为 TCP，即只分析 TCP 数据包；scan_type 选项设置为 portscan，即执行端口扫描检测；sense_level 选项设置为 high，即端口检测限制最小、扫描时间窗口最长，获得更加精确的扫描结果；其它选项可以根据具体需要来设置。

该预处理器开始工作后，会尝试在 T 时间段内检测从单个扫描主机发送到目标主机某些端口的 TCP 数据包，然后依据数据包的特殊标志位，在这些数据包中查找不符合 TCP 连接三次握手过程的异常数据包，因为这类异常的数据包要么具有奇怪的 TCP 标志位组合，要么根本就没有设置标志位。一旦检查到 TCP 标志位组合异常的数据包，snort 就会判定发生了端口扫描。对于 SYN 端口扫描，snort 主要是检测数据包的 SYN 标志位和 FIN 标志位，对应的扫描特征检测规则如下：

[portscan-detecting rules]

① alert tcp any any -> $HOME_NET any (msg:"INDICATOR-SCAN ipEye SYN scan"; flow:stateless; flags:S; seq:1958810375; metadata:ruleset community; classtype:attempted-recon; sid:622; rev:11;)

② alert tcp any any -> $HOME_NET any (msg:"INDICATOR-SCAN synscan portscan"; flow:stateless; flags:SF; id:39426; metadata:ruleset community;

classtype:attempted-recon; sid:630; rev:10;)

在上述两条规则中，“alert”表示表示对匹配该项规则的数据包生成报警，“tcp any any -> $HOME_NET any”表示对一切外来的 TCP 数据包进行检测，“flags:S”和“flags:SF”表示对 TCP 数据包的 SYN 和 FIN 标志位进行检查。

2、报警处理模块

报警处理模块是 iptables 和 snort 联动工作的核心所在，承担着连接 iptables 和 snort 的作用，其工作流程如下图 3-8 所示。

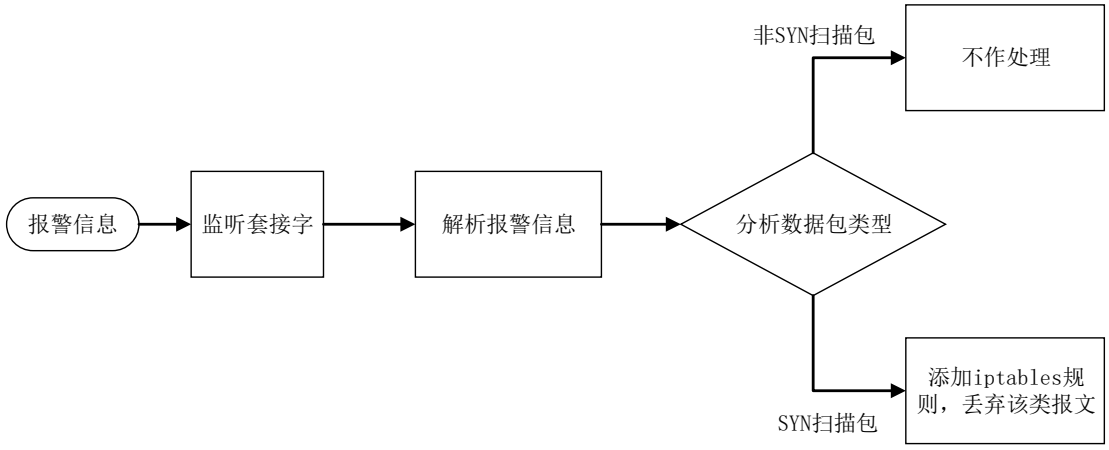

图 3-8 报警处理模块工作流程

snort 将生成的报警信息以数据报的方式发送到指定路径下的套接字（Unix socket），报警处理模块在 snort 指定的该路径上通过监听套接字（Unix socket）来接收报警信息，然后调用 snort 提供的报警信息解析函数对报警信息中的源/目 IP 地址、源/目端口、报警内容和类型、报警事件 ID 等信息进行解析，并根据引发 snort 报警的数据包的类型进行不同响应。对于 SYN 端口扫描报文，报警处理模块向 iptables 防火墙添加新的安全规则，如果后续报文的源 IP 地址以及源/目端口号和安全规则相匹配，则丢弃该类报文。

3.2.2 暴力破解检测及防护

暴力破解攻击是指攻击者系统地组合所有可能的账户名和密码，尝试所有的组合是否能够登录，破解用户的账户、密码等信息，目前最为流行的是 ssh 暴力破解。ssh 暴力破解是自 Linux 平台诞生以来就存在的一种攻击行为，至今已经衍生出针对 telnet、ftp、smtp 等服务的暴力破解方式[50]。以 Mirai 为代表的大部分物联网僵尸网络病毒在感染物联网设备的过程中，采用的就是 telnet 暴力破解方式，当扫描到网络中开启了 telnet 服务的设备时，Mirai 首先会逐一使用内置的约 60 组用户名和密码尝试登陆设备，如果登陆成功则进行下一步的入侵行为，比如开启 shell

等操作。

常用的防御暴力破解攻击的方法及其优缺点分析如下：

⑴ 增强设备的账户和密码复杂性。这种方法能够消耗攻击者更多的计算资源和时间，增加了入侵者的攻击成本，降低设备遭到破解的可能性，但是要求设备厂商和用户具备较高的安全意识，并且管理大量设备的账户和密码也是一项艰巨的挑战；

⑵ 更改或禁用 ssh、telnet 等远程登录服务的端口号。更改端口能够限制小规模的暴力破解攻击，对于 Mirai 这一类僵尸网络发起的大规模、有针对性的暴力破解攻击没有多大防御作用；禁用 ssh、telnet 等端口，虽然能够完全杜绝设备遭到破解，但是会限制设备可以提供的服务，同时也给设备运维带来了不便。

⑶ 使用 fail2ban、denyhosts 等暴力破解防护工具。这一类入侵防御工具，必须使用正则表达式来分析日志，有可能遭到基于日志的攻击方式，并且目前已经有黑客找到了绕开这些防护工具的方法。

基于对上述方法的优缺点分析，以上三种方法都不适合用来防御 Mirai 恶意软件的暴力破解攻击。

本文以 iptables 防火墙为工具，通过限制一定时间段内对 ssh、telnet 等服务的登录尝试次数，来防御 Mirai 的暴力破解攻击。暴力破解攻击是通过大量的尝试次数来获得一定成功率的，频繁的登录尝试会在目标设备上留下登录失败的记录，可以利用这一点作为检测此类暴力破解攻击的依据，通过分析系统的登录记录，当攻击者尝试登录 ssh、telnet 等服务时，只要在一定时间段内达到预设的登录失败次数，iptables 防火墙就会屏蔽登录失败次数达到预设阈值的 IP 地址。对于合法用户或系统管理员来说，如果忘记了或误输入用户名/密码，其主机的 IP 就有可能遭到目标设备屏蔽，这是不希望出现的情况。因此，为了避免这一类误屏蔽合法用户和系统管理员的情况发生，在上述限制错误登录次数这一方法的基础上加入了合法用户或系统管理员白名单，白名单中的 IP 地址可以有无限制的登录尝试次数，而非白名单中的 IP 地址在设定的时间内最多只能有预设的登录尝试次数，达到失败次数上限就会遭到屏蔽。

这种方法的优点是不需要任何额外的第三方软件，几乎所有的 Unix/Linux 操作系统都支持 iptables，因此可以方便快速地部署此解决方案。接下来将对这一暴力破解防御方法的工作流程、对应的 iptables 安全规则脚本模型以及使用该方法防御 Mirai 暴力破解分别进行详细阐述。

1、基于登录次数限制的暴力破解防御方法工作流程

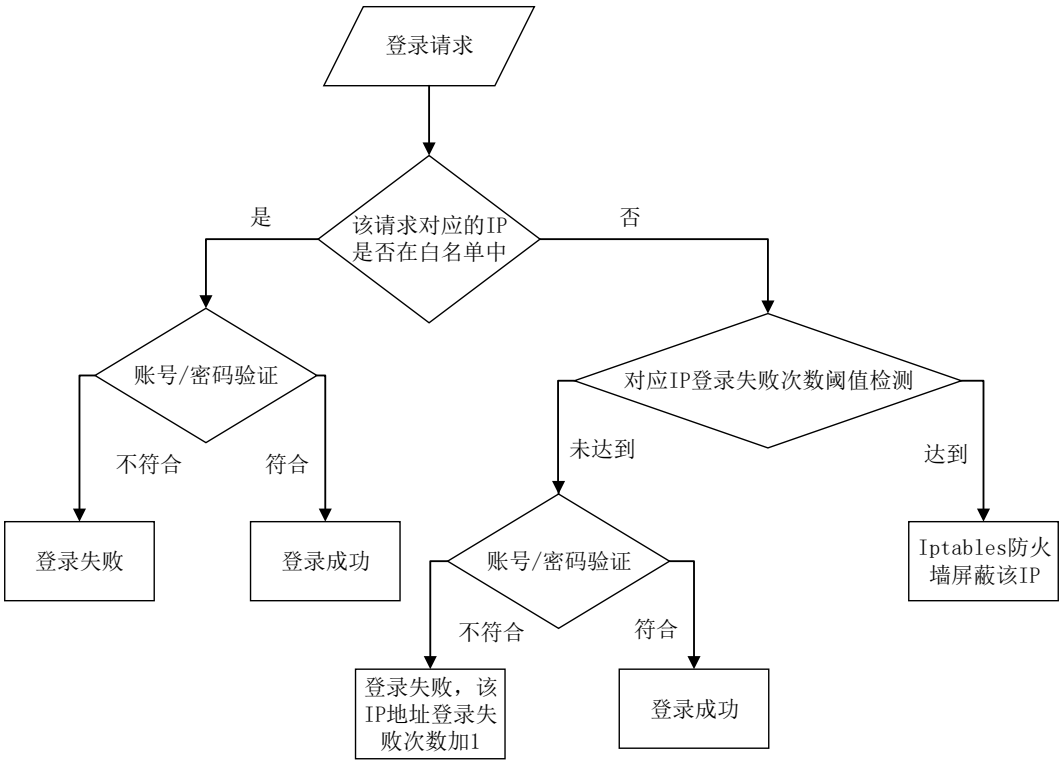

图 3-9 暴力破解防御方法的工作流程

如图 3-9 是上述暴力破解防御方法的具体工作流程，当某个远程登录请求到达目标设备后，首先会将该请求对应的 IP 地址与白名单中的 IP 进行比对，如果该 IP 地址在白名单中，则直接进行账号/密码验证，验证成功则允许登录，验证失败就拒绝其登录，但是不记录其登录失败的次数。如果该 IP 地址不在白名单中，则检查该 IP 在固定的时间周期内是否达到了登录失败次数的阈值，如果达到了登录失败次数的上限，iptables 防火墙就直接屏蔽该 IP 地址，该 IP 地址的主机将无法访问目标设备；如果没有达到登录失败次数的上限，则进行账号/密码验证，验证成功就允许登录，验证失败的话就拒绝其登录，同时将该 IP 地址的登录失败次数加 1 并记录下来。对于后续的登录请求，处理过程同上。

2、对应的 iptables 安全规则模型

要实现上述基于登录次数限制的暴力破解防御方法，关键在于正确、合理地设置 iptables 防火墙的相关安全规则。针对该方法的具体工作流程，本文设计了如下的 iptables 安全规则脚本模型：

[Defense Brute Force]

⑴ 设置默认策略为 ACCEPT：

iptables -P INPUT ACCEPT

⑵ 新建 iptables 链，方便单独管理防御暴力破解的安全规则：

iptables -N [相关链名]

⑶ 匹配 TCP 协议目的端口为[特定端口号]的包并且 state 为 NEW 的包，目的是记录相关 IP 地址的登录请求次数：

iptables -A INPUT -p tcp --dport [特定端口号] -m state --state NEW -j [相关链名]

⑷ 清除链[相关链名]中的所有安全规则，便于添加新规则：

iptables -F [相关链名]

⑸ 将合法用户和系统管理员添加到 IP 白名单，此规则根据需要可以添加多个白名单 IP 地址：

iptables -A [相关链名] -s [合法用户和系统管理员的 IP 地址] -j RETURN

⑹ 非白名单的成员从第一次连接开始，在[限制时间]秒内最多可以尝试连接[连接次数阈值]次，并记录日志：

iptables -A [相关链名] -m recent --rcheck --seconds [限制时间] --hitcount [连接次数阈值] -j DROP

iptables -A [相关链名] -m recent --rcheck --seconds [限制时间] --hitcount [连接次数阈值] -j LOG --log-prefix “DROP BRUTE FORCE ATTEMPTION!”

⑺ 将非白名单 IP 的连接请求次数加 1：

iptables -A [相关链名] -m recent --set -j ACCEPT

3、防御僵尸网络暴力破解

以防御 Mirai 恶意软件的 telnet 暴力破解为例，根据以上 iptables 安全规则模型，需要定义管理 telnet 暴力破解规则的链“TELNET_LIMIT”，并根据 Mirai 进行 telnet 登录尝试的速率，设定 10 秒内最多只允许 3 次 telnet 登录尝试，假设系统管理员主机的 IP 地址为 xxx.xxx.xxx.xxx，具体的 iptables 安全规则脚本如下：

[Defense Mirai-Brute-Force]

① iptables -P INPUT ACCEPT

② iptables -N TELNET_LIMIT

③ iptables -A INPUT -p tcp --dport 23 -m state --state NEW -j TELNET_LIMIT

④ iptables -F TELNET_LIMIT

⑤ iptables -A TELNET_LIMIT -s xxx.xxx.xxx.xxx -j RETURN

⑥ iptables -A TELNET_LIMIT -m recent --rcheck --seconds 10 --hitcount 3 -j DROP

iptables -A TELNET_LIMIT -m recent --rcheck --seconds 10 --hitcount 3 -j LOG --log-prefix “DROP BRUTE FORCE ATTEMPTION!”

⑦ iptables -A TELNET_LIMIT -m recent --set -j ACCEPT

以上安全规则中，规则①将访问设备的默认规则设置为 ACCEPT（即允许访问），规则②新建了一条管理 telnet 访问规则的链“TELNET_LIMIT”，规则③表示链 TELNET_LIMIT 中的安全规则都匹配 TCP 协议、端口号为 23 且连接状态 state 为 NEW 的数据包。假设 Mirai 服务器暴力破解设备 telnet 服务的数据包是在 10 秒内到达本机的，模拟这些连接请求包的处理流程如下：

⑴ 当 Mirai 服务器发出的第 1 个 telnet 连接请求包到达本机，规则⑤检查该请求中的 IP 地址是否在白名单中，如果该 IP 在白名单中，则直接将该请求包转发给本机的 telnet 服务器进行账号/密码验证。显然，由于 Mirai 服务器的 IP 不在白名单中，因此该请求包会转给规则⑥处理；

⑵ 规则⑥检查该请求包对应的 IP 是否达到了连接次数上限（3 次），因为是第 1 个请求包，所以连接次数为 1 次，规则⑥判定不对该请求包执行 DROP（丢弃）操作，而是转给规则⑦处理；

⑶ 规则⑦将这个请求包的 IP 地址所对应的连接次数加 1 并记录下来，然后执行 ACCEPT（允许通过）操作，把该请求包转发给本机的 telnet 服务器进行账号/密码验证（在这里假设没有通过验证）；

⑷ Mirai 服务器后续发出的第 2、3 个 telnet 连接请求包的处理流程同上；

⑸ 当 Mirai 服务器的第 4 个 telnet 连接请求包到达设备，规则⑥检查到该 IP 对应的连接请求失败次数已经达到了 3 次，于是执行 DROP 操作丢弃该请求包并记录日志，并且后续的连接请求包都会被拒绝。

3.3 本章小结

本章针对工业物联网数据采集层安全防护策略进行了设计和实现。首先分析了以 Mirai 为代表的物联网僵尸网络病毒，总结出这类病毒感染物联网终端设备的主要手段是端口扫描和暴力字典破解，然后针对这两种入侵手段设计防御物联网僵尸网络病毒攻击的方法，具体实现为：使用 snort 和 iptables 联动的方法，防御物联网僵尸网络病毒对物联网设备的端口扫描探测；利用 iptables 的访问控制功能，通过基于白名单限制远程登录次数的方法来防御物联网僵尸网络病毒发起的暴力破解攻击。

第四章 数据传输层安全防护策略设计及实现

本章针对工业物联网系统数据传输层存在的安全隐患，从保护通信过程的安全这一方面进行防护研究，包括通信过程中数据的机密性和终端设备之间的身份认证。对于工业物联网系统采用的 MQTT 通信协议，对其通信过程进行加密传输，终端设备之间的通信采用身份认证，保证数据传输层提供安全、准确的数据传输服务。

4.1 数据窃取和中间人攻击

目前，工业物联网系统所采用的通信协议，其安全措施没有统一的规定，特别是在通信加密和身份认证这两方面，如果数据采集层使用的通信协议既是明文传输，同时通信的设备之间又没有采取身份认证，攻击者一旦入侵到物联网系统局域网中，就可以很轻松的采用网络嗅探等方法来拦截通信数据。由于通信协议没有加密，攻击者可以很轻易地检测到数据内容，窃取重要的工业生产数据。

同时，没有经过身份认证的通信过程还容易遭到中间人攻击（MITM，Man-in-the-middle attack）。在密码学和计算机安全中，中间人攻击是一种旨在规避互相认证或者缺乏互相认证的攻击方式，如图 4-1 所示。当攻击者能够按照预期的合法目标冒充通信网络中的某个端点时就能够秘密地将自己作为中继或代理与通信的两端分别建立连接并在它们之间传递消息，从而控制整个通信会话过程，因此攻击者能够拦截受害者之间传递的所有消息，并注入经过篡改或伪造的消息[51]。通过这种攻击方式，入侵者可以在数据传输层随意篡改或者伪造数据包内容，或者发送错误的控制指令，可能造成工业设备无法正常运转。

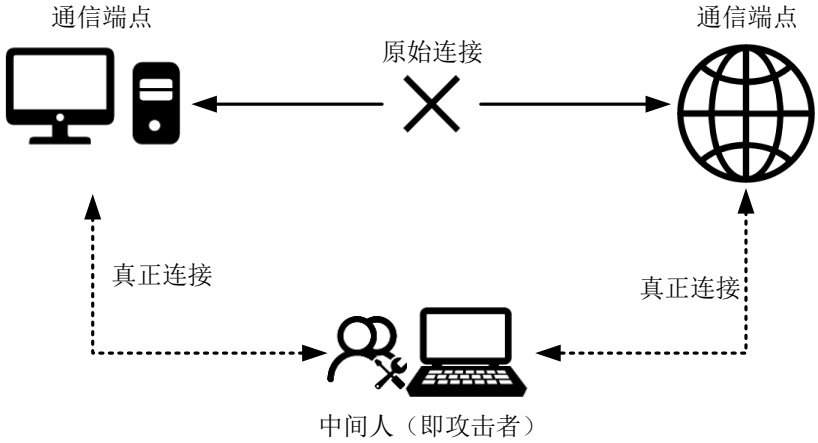

图 4-1 中间人攻击示意图

对于工业物联网系统中广泛使用的 MQTT 协议，在默认情况下其通讯过程既不进行加密也没有身份认证机制，容易遭到信息劫持、篡改和中间人攻击，给工业生产环境造成严重威胁。如果需要使用该协议传输敏感信息或者对设备进行控制，在物联网项目的实施过程中必须考虑到针对该协议的安全设计。本文对通信加密和身份认证方面的主流技术进行研究，并选取合适的技术方案对 MQTT协 议的通信过程进行数据加密和身份认证。

4.2 通信加密及身份认证技术研究

4.2.1 通信加密相关技术

加密技术是信息安全技术的主要组成部分，其主要原理是数据发送端使用某种密码算法将明文信息变换成密文信息，数据接收端再使用规定好的秘钥，通过对应的解密算法将密文信息还原成明文信息。目前应用最为广泛、最具代表性的加密技术是网络层的安全协议 IPSec 以及传输层的安全协议 SSL/TLS。

⑴ 网络层的安全协议 IPSec

IPSec 采用了认证头（AH）、封装安全载荷（ESP）和秘钥管理协议（IKE）等安全技术，在网络层的通信过程中，其认证机制使得数据的发送方和接收方都能确认对方的真实身份，以及通信数据是否遭到篡改；在端到端传输的应用场景中，其加密机制对数据进行了编码，实现了通信数据的机密性和完整性，保障数据在传输过程中的安全，一般用于虚拟专用网络（VPN）通信。

⑵ 传输层的安全协议 SSL/TLS

SSL（安全套接字层）是以传输层（TCP）协议为基础、为应用层协议的通信数据进行加密的标准安全技术，而 TLS（传输层安全协议）是建立在 SSL 规范之上的一种新的协议，相比之下 TLS 协议的规范更加精确、完善和安全，提供了认证用户及服务器、加密数据以及维护数据完整性这三大服务。TLS 在传输层上给原先非安全的应用层协议提供加密保护，如非安全的 HTTP 协议即可被 TLS 加密形成安全的 HTTPS 协议，广泛应用在浏览器、金融科技、即时通讯等方面。

4.2.2 身份认证相关技术

通信过程中的身份认证，是指数据发送方和接收方确认彼此的身份是否合法，其目的是使通信双方建立起信任关系，通过这种互相认证的方式来保证网络的安全性。目前使用很广泛的身份认证技术有以下几种：

⑴ 基于口令的身份认证技术

基于口令的身份认证采取“用户名+用户名密码”的方式来实现，在进入系统时，将输入的用户名和密码与系统事先保存的用户名/密码组合进行比较，如果相同则表示身份合法，这种认证方法通常应用于操作系统。由于所有的用户名/密码组合都是以文件形式存在的静态数据，容易遭到攻击者窃取，同时验证信息在传输过程中也可能被攻击者截获，口令一旦泄露，攻击者就可以冒充合法用户进入系统，因此这种认证方式安全性不高。

⑵ 双因素身份认证技术

双因素身份认证也是一种基于口令的认证方式，但是使用了认证令牌来生成动态口令，进行身份验证是要求同时提供静态口令和动态口令，只有两者都通过验证是才能进入系统，通常应用在电子 、网络游戏等领域。双因素身份认证的动态口令是一个与时间相关且不可预测的随机数字组合，每次生成的口令均不相同，而且只能使用一次，因此攻击者难以假冒合法用户的身份。

⑶ 基于数字证书的身份认证技术

数字证书（Digital Certificate）是一种由权威机构发行的电子文档，包含了身份特征等数据，提供了一种对自身以及对方的身份进行校验的机制，通常应用在浏览器客户端和 web 服务器之间的身份校验。这种认证方式依赖于双方共同信任的第三方，称为 CA（Certification Authority）认证机构，用户凭借数字证书就能访问受该机构信任的服务器。数字证书采用非对称密码体制，在实现上遵循国际电信联盟的 X509 标准，一般也称为 X509 证书，该证书一般用于 HTTPS 等各类应用层协议中的身份认证。

4.3 通信加密和身份认证策略

本章主要针对 MQTT 协议通信过程中可能遭受到的信息泄露、数据篡改和中间人攻击进行安全防护，需要选取适用于 MQTT 协议的通信加密技术以及身份认证方法。

1、MQTT 通信加密

从上一小节可知，IPSec 和 TLS 是目前应用最为广泛的通信协议加密技术，下面将从几个方面来分析这两种加密技术的优缺点：

⑴ 部署和管理成本。IPSec 在部署时需要大量的 IT 技术支持，比如安装复杂的客户端软件、对网络基础设施进行重大改造以及需要长期维护等，因此 IPSec 的管理成本非常高；TLS 则正好相反，在部署时非常简单，能够做到即刻安装、即刻生效，同时对技术支持和运维管理的要求也比较低，因此降低了部署成本。

⑵ 兼容性。传统的 IPSec 因为需要客户端软件的支持，因此对于不同的终端

操作系统就需要安装不同的客户端，并且不同的客户端对操作系统的版本也有很高的要求，而 TLS 因为不需要安装客户端，所以完全不需要这方面的考虑。

⑶ 可扩展性。IPSec 在部署时需要考虑网络的拓扑结构，如果网络中添加新的设备就需要更改网络结构，因此 IPSec 的可扩展性比较差。TLS 在需要重新部署时，可以随时添加受保护的设备，所以比 IPSec 有更好的扩展性。

⑷ 安全性。IPSec 的每一个使用端在网络上都会被当成一个节点并一直处于联机状态，因此黑客可以通过该节点进入另一个端点，比如其他网络内部；TLS 的 session 保护功能可以在会话停止一段时间后自动断开连接，继续访问时需要重新登录，这种机制可以有效避免这种危险发生。

基于以上分析，TLS 加密技术在多方面均优于 IPSec 加密，并且由于 MQTT 是基于 TCP 的应用层协议，符合 TLS 在传输层对网络传输进行加密的技术特点和应用场景，所以本文采用 TLS 对 MQTT 通信过程进行加密。

2、MQTT 身份认证

上一小节介绍了基于口令的身份认证、双因素身份认证以及基于数字证书的身份这三种常用的身份校验技术，基于口令的身份认证主要用于操作系统级别的登录身份校验，并且其安全比较低；双因素身份认证虽然安全性较高，但是主要用于应用程序级别的身份校验。考虑到以上因素，这两种身份认证方法均不适合于 MQTT 协议的身份认证，并且由于加密 MQTT 协议的 TLS 支持基于数字证书的身份校验，因此本文采用适合于应用层协议身份校验的 X509 数字证书对 MQTT 进行身份认证。

4.3.1 基于 TLS 的 MQTT 加密传输

MQTT 依赖于 TCP 作为传输协议，通过 TLS 加密整个 MQTT 通信，MQTT 代理服务器将使用 TLS 协议代替普通 TCP 协议，TLS 在协议栈中的位置如图 4-2 所示。TLS 协议在传输层上封装了应用层的数据，因此可以在不需要修改 MQTT 协议的前提下给不安全的 MQTT 协议提供一定的安全保障。

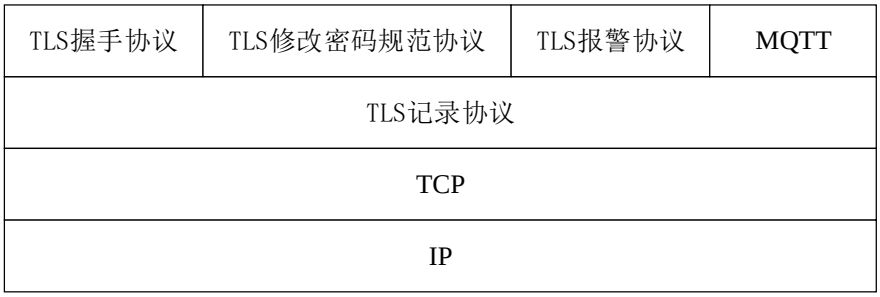

图 4-2 TLS 协议栈

TLS 在 MQTT 客户端和 MQTT 服务器之间建立安全的通信通道，主要包括握手协商阶段和应用数据传输阶段，图 4-3 显示了握手协商阶段的基本过程。

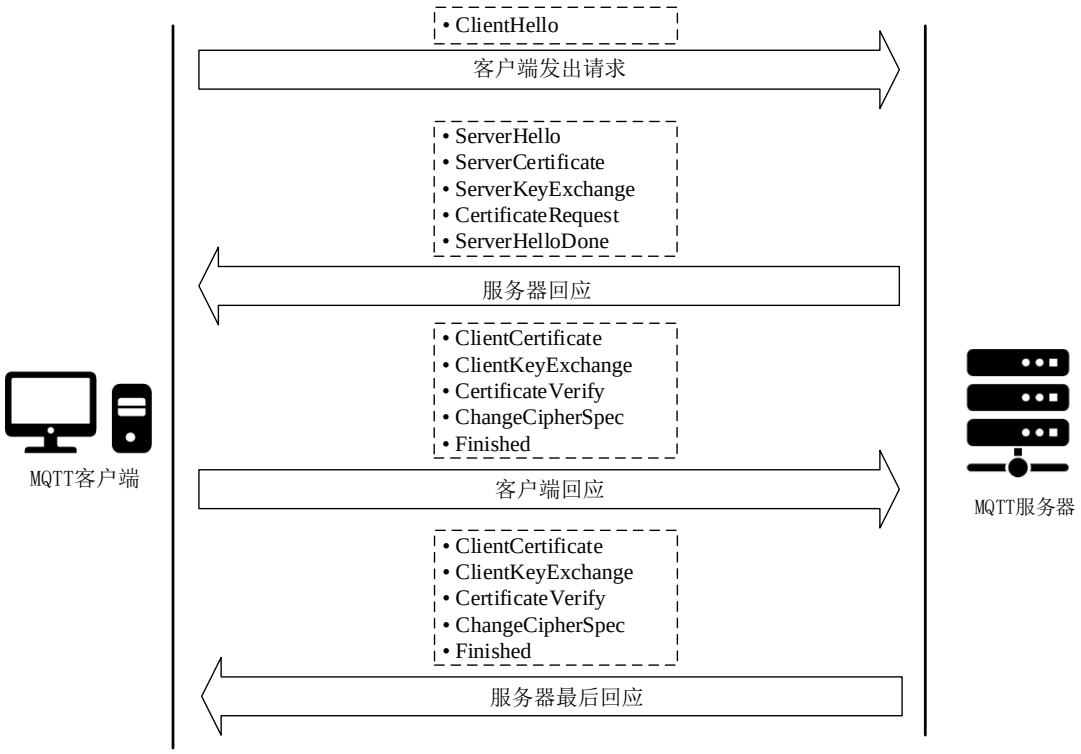

图 4-3 TLS 握手协商阶段

在握手协商阶段，客户端首先向服务器发出连接请求，然后双方会进行一些信息交换，并根据这些数据和一定算法生成相同的会话秘钥，客户端和服务器之间创建安全连接；握手完成后，客户端和服务器之间采用会话秘钥进行加密通信，数据发送方使用对话秘钥来对信息进行加密，数据接收方也用同样的会话秘钥来对密文数据进行解密。TLS 握手协商阶段包括四次通信过程：

⑴ 客户端向服务器发出“ClientHello”请求

首先，客户端会发送一个通信加密的请求给服务器，该请求中包含了客户端能够兼容的加密协议版本、加密算法套件（比如 RSA 公钥加密）以及压缩方法等信息，由服务器来决定具体采用哪种加密协议和算法组合；同时该请求还包含一个由客户端生成的随机数，后续过程中生成会话密钥时会使用到这个随机数。

⑵ 服务器向客户端回应“ServerHello”消息

服务器收到客户端请求后会返回一个回应消息，服务器在该消息中确定了使用的协议版本（应该与客户端的协议版本一致）、加密算法套件以及服务器的数字证书，同时该消息中还包括一个由服务器生成的随机数（用于后续生成会话秘钥）。假如服务器要求对客户端身份进行合法性验证，该消息中就会加入一项需

要客户端发送身份认证数字证书的请求条目。

⑶ 客户端对“ServerHello”消息进行回应

客户端收到服务器的“ServerHello”消息以后，对服务器数字证书进行合法性检查，假如该证书不是合法的，客户端就会直接断开连接或者向用户发送一个服务器不受信任的警告；如果证书通过了合法验证，客户端会再次生成一个随机数（用于后续生成会话秘钥），并使用服务器证书中的服务器公钥对该随机数进行加密，然后向服务器发送消息，该消息包含了加密的随机数、编码改变通知（即应用数据传输阶段将使用双方确定的加密套件）以及客户端握手结束通知（表示客户端已经完成了握手阶段）。

⑷ 服务器回应握手阶段结束

服务器收到上述客户端的回应消息后，使用自己的私钥对该消息中的加密随机数进行解密，然后使用三个随机数（过程⑴、⑵、⑶中各一个）计算应用数据传输阶段所使用的会话秘钥，然后向客户端发送消息，该消息包括编码改变确认（和过程⑶中的客户端编码改变通知相对应）以及服务器握手结束通知（表示服务器也完成了握手阶段）。

经过上述过程，握手协商阶段结束，如果客户端和服务器都校验成功，双方将按照 TLS 协议的规范使用协商生成的会话秘钥来加密通信数据。本文使用开源的安全套接字层密码库 OpenSSL 来实现 TLS 对 MQTT 协议的加密，OpenSSL 包括了主要的密码算法、常用的密钥、证书封装管理功能以及 TLS 协议的实现。根据 OpenSSL 提供的程序开发标准，使用 OpenSSL 对 MQTT 进行 TLS 加密的具体流程如图 4-4 所示。

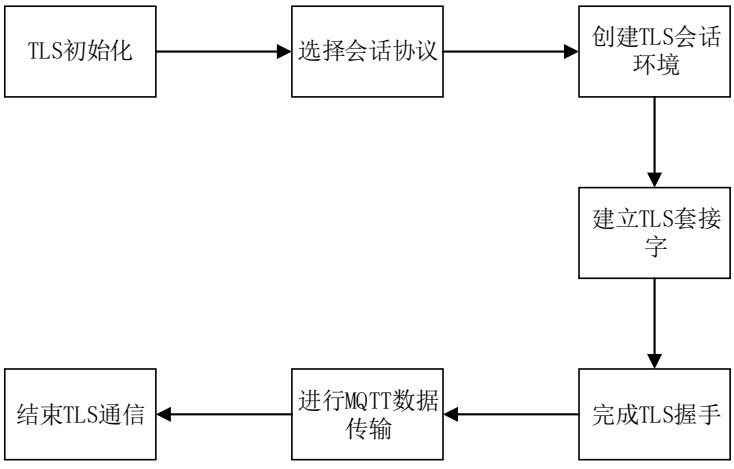

图 4-4 OpenSSL 对 MQTT 进行 TLS 加密的具体流程

由上图可知，使用 OpenSSL 加密 MQTT 通信分为以下步骤：

⑴ TLS 初始化。该阶段主要是加载 OpenSSL 提供的各种加密算法套件以及哈希算法；

⑵ 选择会话协议。在使用 OpenSSL 开始 TLS 会话之前需要为 MQTT 客户端和服务端指定采用的加密协议和版本，应该尽量选择双方都支持的协议的最高版本，本文选择 TLS1.2；

⑶ 创建 TLS 会话环境。该阶段主要是指定 TLS 在握手阶段验证数字证书的方式，并加载 MQTT 客户端和服务端的证书以及 CA 证书（关于证书生成的具体方法会在下一小节叙述）；

⑷ 建立 TLS 套接字。TLS 套接字是建立在 TCP 协议套接字之上的，OpenSSL 提供了类似于建立 TCP 套接字的一套 API 规范；

⑸ 完成 TLS 握手。握手过程完成之后，需要询问 MQTT 客户端和服务端的证书信息，进行相互的身份认证；

⑹ MQTT 数据传输。该阶段将使用握手阶段协商的会话秘钥对 MQTT 报文进行加密和解密；

⑺ 结束 TLS 通信。当 MQTT 客户端和服务端的通信结束后，需要释放相关的 TLS 资源。

4.3.2 基于 X509 证书的 MQTT 身份认证

证书(Certificate，也称为 public-key certificate)是用某种签名算法对某些内容，比如公钥（public-key），进行数字签名后可以用来当成信任关系中介的数字凭证。数字证书的发证机关是证书发行机构（Certificate Authority，也称为 CA），CA 用自己的私钥对用户的身份信息(主要是用户名和该用户的公钥)、证书机构名称、证书有效期等相关消息进行签名，该数字签名和用户的身份信息一起就形成了数字证书，如下图 4-5 所示。

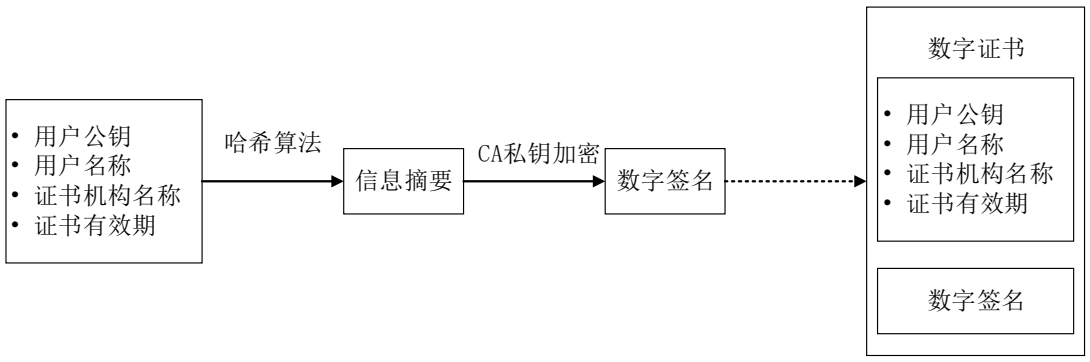

图 4-5 数字证书组成

在前一小节中，使用 TLS 对 MQTT 进行传输加密的过程中，TLS 服务器需要

公钥/私钥对。在 TLS 的握手协商阶段，客户端需要在建立安全连接之前验证服务器端的 X509 证书，客户端需要在建立安全连接之前验证服务器端的 X509 证书，该证书中还包含了服务器的公钥。对于建立安全连接来说，尽管服务器使用私钥/公钥对就足够了，但客户端也可以在 TLS 握手过程中使用唯一的公钥/私钥对来进行验证。在验证服务器的证书后，客户端将其证书（证书中包括了客户端的公钥）作为 TLS 握手的一部分发送给服务器，服务器可以使用该证书验证客户端身份的合法性，一旦客户端证书没有通过验证，则服务器可以终止握手，在传输层级别就能验证 MQTT 客户端的身份，在发送 MQTT CONNECT 消息之前锁定无效的 MQTT 客户端，阻断非法设备的连接，防范中间人攻击。使用数字证书验证 MQTT 客户端身份的工作流程如图 4-6 所示。

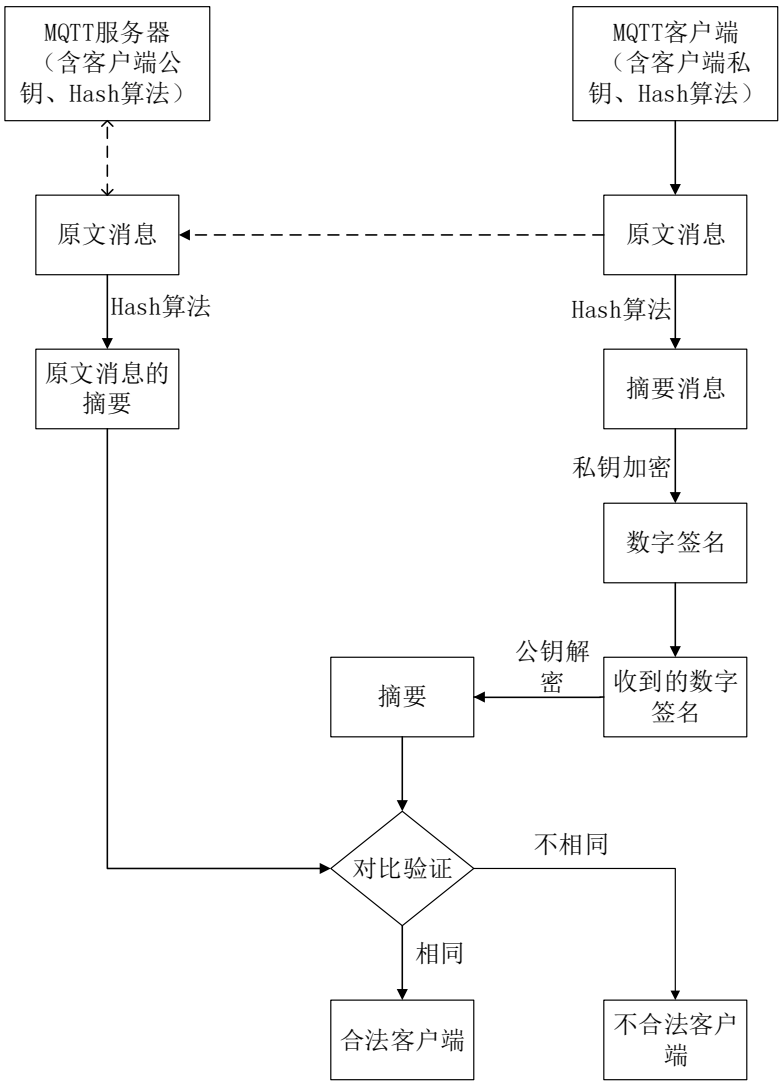

图 4-6 使用数字证书验证 MQTT 客户端身份

MQTT 客户端和服务器在握手成功之后，客户端向服务器继续发送 X509 证书

来认证设备：

⑴ MQTT 客户端对发送给 MQTT 服务器的原文消息使用 Hash 函数，生成摘要消息；

⑵ MQTT 客户端使用自己的私钥，对摘要消息进行加密，生成数字签名；

⑶ MQTT 客户端将数字签名和原文消息一起发送给 MQTT 服务器；

⑷ MQTT 服务器收到消息后，取出其中的数字签名，用客户端的公钥进行解密，得到原文消息的摘要；

⑸ MQTT 服务器再对收到的原文消息本身使用 Hash 函数，将得到的结果与上一步得到的摘要进行对比，如果两者一致，则说明该 MQTT 客户端是合法设备，否则是非法设备。

在本文的实现过程中，使用 OpenSSL 来生成自签名的 CA 证书，然后使用该 CA 证书对 MQTT 客户端和服务端的 X509 证书进行签名，只要保证 MQTT 客户端和服务端的私钥不被泄露，就能保证经 CA 签名的 MQTT 客户端和服务端的 X509 证书是可信、安全的。使用 OpenSSL 生成相关的证书和私钥的方法如下：

⑴ 生成 CA 私钥及 CA 证书

首先生成 CA 的私钥文件 ca.key，然后使用该私钥生成 CA 自签名的证书 ca.crt：

```
openssl -des3 -out ca.key 1024
openssl req -new -x509 -key ca.key -out ca.crt
```

⑵ 生成 MQTT 客户端的私钥及 X509 证书

首先生成 MQTT 客户端的私钥文件 client.key：

```
openssl genrsa -des3 -out client.key 1024
```

使用该私钥生成 MQTT 客户端的证书签名请求文件 client.csr：

```
openssl req -new -key client.key -out client.csr
```

使用 CA 证书为 MQTT 客户端的证书签名请求文件进行签名，生成 MQTT 客户端 X509 证书 client.crt：

```
openssl x509 -req -in client.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out client.crt
```

⑶ 参考以上生成 MQTT 客户端私钥及证书的方法，生成 MQTT 客户端私钥 server.key 及 X509 证书 server.crt：

```
openssl genrsa -des3 -out server.key 1024
openssl req -new -key server.key -out server.csr
openssl x509 -req -in server.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out server.crt
```

至此，已经生成了 MQTT 客户端和服务端进行双向身份认证所需要的密钥文件和证书文件，按照上一小节中使用 OpenSSL 对 MQTT 进行 TLS 加密的流程，在创建 TLS 会话环境时加载 MQTT 客户端、服务端的 X509 证书以及自签名的 CA 证书，TLS 在握手协商阶段，MQTT 客户端和服务端会要求对方发送 X509 证书并验证对方证书的真实性和有效性，从而判断对方是不是可信的合法 MQTT 客户端或服务端。

4.4 本章小结

本章主要针对工业物联网数据传输层可能遭受到的信息泄露和中间人攻击进行防御策略设计，通过对各种主流的通信协议加密技术和身份验证技术的分析、对比，本文选择 TLS 对数据传输层通常使用的 MQTT 协议进行加密传输，以及通过 X509 证书对 MQTT 客户端和服务器端进行身份认证，并使用开源的安全套接字层密码库 OpenSSL 对这两种方法进行了实现，保证数据传输层提供安全的数据传输服务。

第五章 数据处理层安全防护策略设计及实现

本章针对工业物联网系统数据处理层存在的安全威胁，主要对数据处理层的通信代理服务器进行防护研究。本文所研究的工业物联网系统，其通信规约是 MQTT 协议，针对该协议本身存在的安全漏洞，使用基于白名单的深度包检测技术对该协议进行深度解析，防止 MQTT 服务器遭到伪造数据流量的攻击；同时使用针对 MQTT 协议的入侵检测系统，对通信服务器与终端设备的网络通信流量进行实时监测，一旦发现异常立即报警。

5.1 伪造数据包攻击

在工业物联网系统的设计过程中，如果传输层所使用的物联网协议没有采取身份认证措施，那么只要是符合要求的客户端设备都能连接到数据处理层的通信服务器。在实际应用中，非常多的工业物联网环境其传输层使用的是 MQTT 协议，如果使用该协议时没有采取身份认证机制，攻击者就能够通过非法的客户端连接到 MQTT 服务器，进而轻易窃取或篡改重要的工业生产数据，或者伪造具有破坏性的控制指令，可能对生产设备造成严重破坏。

MQTT 协议使用预定义的控制报文来完成数据交换，根据其协议规约，MQTT 控制报文中包含了固定报头、可变报头和有效载荷，其中固定报头是每个控制报文都包含的，而可变报头和有效载荷只有一部分控制报文才会包含。固定报头由两个字节的数据组成，其格式如图 5-1 所示。

Bit	7	6	5	4	3	2	1	0
byte1	MQTT控制报文类型				指定控制报文类型标志位			
byte2	剩余长度							

图 5-1 MQTT 协议固定报头的格式

控制报文的类型以及指定控制报文类型的标志位是固定报头中最重要的数据信息。伪造数据包攻击就是攻击者入侵到物联网系统局域网络后，通过发送伪造的 MQTT 控制报文来发起攻击，这些控制报文在结构上是符合规范的，但是其控制报文的类型以及指定控制报文类型的标志位是攻击者伪造的，任何订阅了这些控制报文的设备都有可能无法正常工作。

如图 5-2 所示，攻击者使用伪装的 MQTT 客户端连接到没有采取身份认证安

全措施的物联网络中的 MQTT 服务器，然后向服务器发布修改了协议内容的控制报文，任何订阅了相关消息主题的合法客户端都有可能遭到攻击者控制。

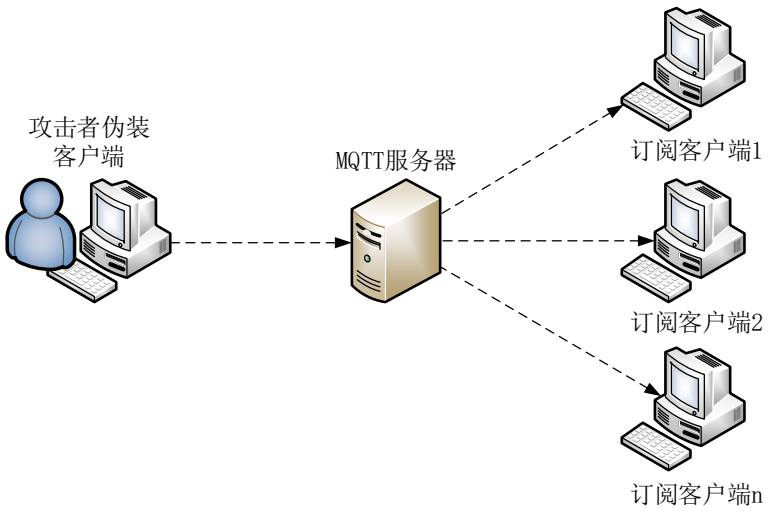

图 5-2 伪造数据包攻击示意图

除了上述攻击方法，攻击者在入侵到物联网络中后，还能将自己的入侵主机同时伪装成 MQTT 客户端和 MQTT 服务器，如图 5-3 所示。对于网络中的合法服务器来说，攻击者的主机相当于 MQTT 客户端，而对于其他合法的客户端来说，攻击者的主机相当于 MQTT 服务器，因此所有发送的数据都会经过攻击者的主机，攻击者能够进行任意的修改再转发出去。同时，攻击者如果发送了经过篡改的控制报文，但是却可以返回正常的信息状态，这种入侵行为将很难被发现。

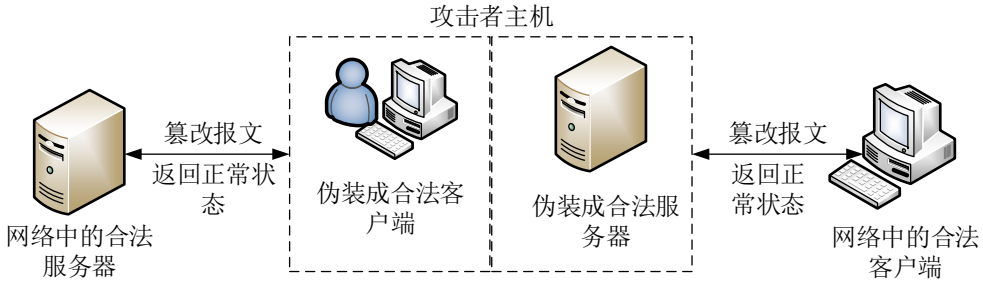

图 5-3 另一类伪造数据包的攻击方式

对于以上两类伪造数据的攻击方式，目前通常采用基于白名单的访问控制方法来进行防护。如图 5-4 所示，MQTT 服务器前设置了白名单防火墙，所有客户端的连接请求都必须经过该防火墙，只有连接请求的源 IP 地址在白名单列表中时，客户端才能通过防火墙连接到服务器，任何不在白名单列表中的 IP 地址，其连接请求都会被拒绝。

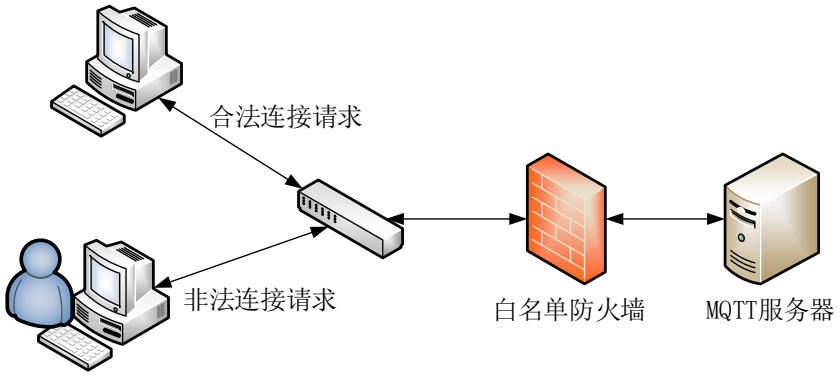

图 5-4 基于白名单的防护模型

使用基于白名单的访问控制技术可以有效过滤掉攻击者的非法连接请求，然而还是有其他方法能够突破该白名单防火墙的限制，比如攻击者可以通过恶意程序等方式来获得对合法客户端设备的控制权，然后控制该合法客户端设备连接到 MQTT 服务器，如图 5-5 所示。攻击者劫持合法的客户端连接到 MQTT 服务器时，由于该客户端设备的 IP 地址在白名单当中，所以其连接求能够正常通过白名单防火墙，之后攻击者依旧可以发布伪造的控制报文。

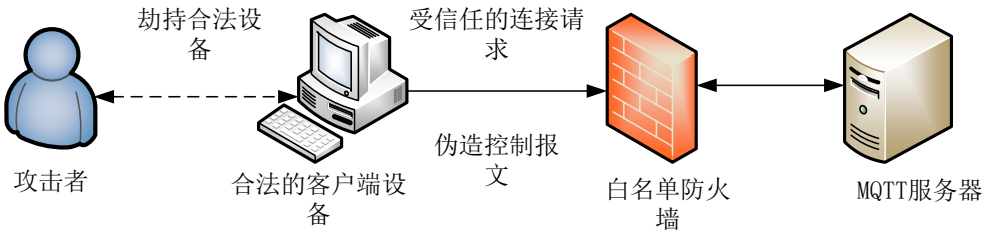

图 5-5 劫持合法设备绕开防火墙检测

5.2 针对伪造数据包攻击的防护策略

大多数情况下，通信协议代理服务器缺乏足够的安全防护措施，只是采用了传统的防火墙来进行一些简单的防护，而这类防火墙只能防御较为简单的攻击，不能提供全面的安全防护效果，因此需要在传统防火墙的基础上引入更加有效的安全防护措施。本文在工业物联网系统的数据处理层使用针对 MQTT 协议的深度包检测防火墙来对应用层协议进行深度检测，同时结合基于朴素贝叶斯的 MQTT 异常流量检测系统来对网络中的流量进实时检测，这两种防护方法的部署如图 5-6 所示。

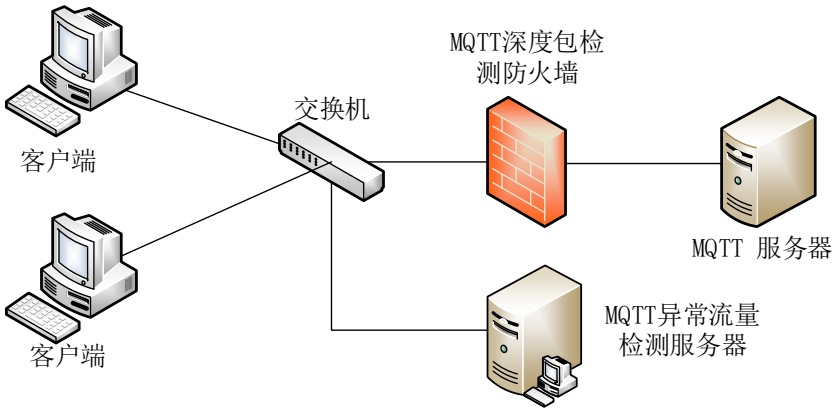

图 5-6 MQTT 服务器防护

深度包检测防火墙和入侵检测系统同时进行检测和防护，不仅能够防御有明显异常特征的伪造数据包攻击，对于符合协议规约和访问控制规则的伪造数据包，也能够实时进行检测，为 MQTT 服务器提供比较全面的防护。两者的工作流程如图 5-7 所示。

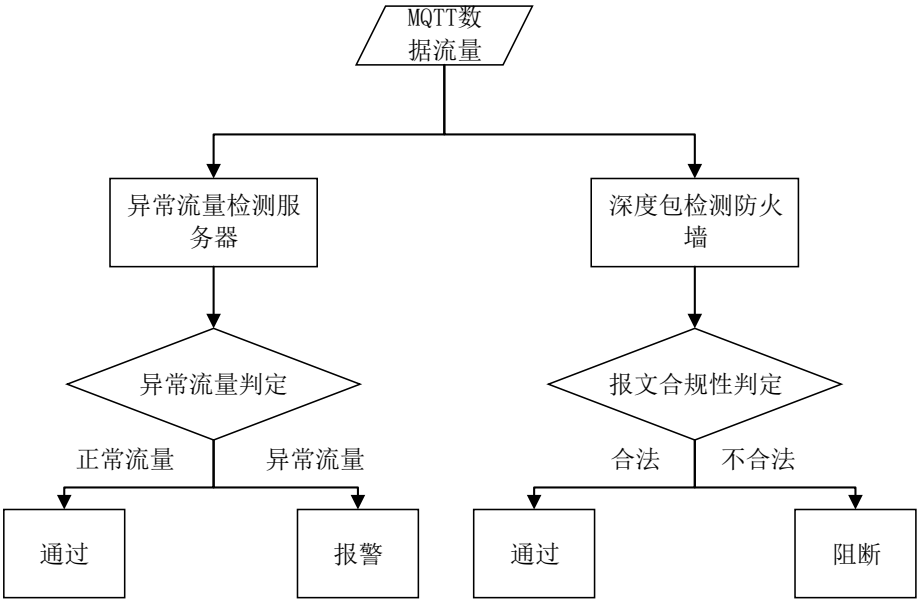

图 5-7 深度包检测和异常流量检测流程

5.3 基于深度包检测的访问控制模块设计

本文设计的深度包检测技术将从 TCP/IP 数据流和应用层 MQTT 协议报文这两方面进行流量检测和访问控制。对于 TCP/IP 数据流，使用基于 IP 地址、端口号的访问控制策略来阻断非法设备连接到服务器；对于应用层的 MQTT 控制报文，采用基于白名单的访问控制策略来阻断伪造的、不符合协议规约的控制报文，以及不满足自定义应用层安全规则的控制报文对 MQTT 服务器的访问。

5.3.1 深度包检测访问控制模型

深度包检测是信息安全中经常使用的一种的流量检测和访问控制技术，当 TCP/IP 数据流或应用层协议数据流通过深度包流量检测系统时，该系统能够深入读取数据包载荷的信息，并对这些信息进行分析和重组，由此解析出整个通信报文的内容，最终根据安全管理员事先配置的安全规则对数据流进行访问控制。本文所设计的 MQTT 深度包检测访问控制模型如图 5-8 所示。

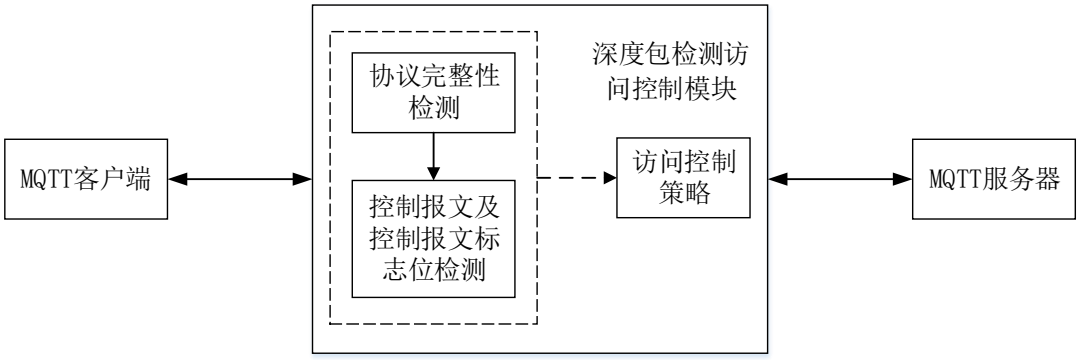

图 5-8 MQTT 协议深度包检测模型

在上图所示的模型中，深度包检测模块部署在 MQTT 服务器和 MQTT 客户端之间，主要包括协议完整性检测、控制报文及控制报文标志位解析、访问控制控制策略。协议完整性检测主要是检测该 MQTT 控制报文是否符合协议规约，能够初步判断该数据包是正常流量还是异常流量；控制报文及控制报文标志位检测主要是对固定报头的内容进行提取和分析，为访问控制策略进一步判定 MQTT 控制报文的合法性提供了匹配依据；访问控制策略将自定义的应用层安全规则与提取到的控制报文内容进行匹配，判断 MQTT 控制报文是否满足访问控制规则，从而实现对 MQTT 服务器的安全访问。

5.3.2 协议完整性检测

如前文 5.1 节中所述，攻击者使用伪装的客户端连接到 MQTT 服务器后，可以下发伪造的控制报文，这些异常的控制报文可能对生产设备造成破坏。因此，需要对 MQTT服务器和客户端之间的通信数据进行完整性检查，结合 MQTT 协议规约，验证控制报文的结构和内容是否合法，丢弃具有明显异常特征的非法报文。对 MQTT 控制报文进行完整性的方法如下：

⑴ 端口号检测

MQTT 协议规约是由 IBM 公司制定的，已经明确指定了使用 1883 号端口，对于访问 MQTT 服务器的目的端口号不是 1883 的控制报文，深度包检测防火墙应

该将其归为异常的数据包。

⑵ 固定报头检测

MQTT 控制报文由固定报头、可变报头和有效载荷这三部分组成，可变报头和有效载荷只有一部分控制报文才会包含，而固定报头是每个控制报文都包含的，位于应用层数据的第 1、2 字节，如果检测到可变报头或有效载荷之前的数据长度不是两个字节，则判定该控制报文异常。

⑶ 控制报文长度检测

MQTT 控制报文的固定报头中，第二个字节指定了该控制报文的剩余数据长度，所以检测固定报头之后的数据的长度，如果和固定报头中的剩余长度不匹配，则判定该控制报文异常。

5.3.3 控制报文类型及控制报文类型标志位检测

对 MQTT 协议的控制报文进行完整性检测，只能够确保控制报文的结构是符合 MQTT 规约的，而不能确保控制报文消息类型的正确性，所以还需要对固定报头中的控制报文类型和指定控制报文类型的标志位进行深度解析。

MQTT 控制报文类型由固定报头第一个字节的二进制位 7-4 表示，取值范围是 0~15，其中 0 和 15 为协议所保留，1~14 按报文流动的方向分为三类：

⑴ 客户端到服务端，包括 CONNECT（值为 1，表示客户端请求连接服务器）、SUBSCRIBE（值为 8，表示客户端订阅请求）、UNSUBSCRIBE（值为 10，表示客户端取消订阅请求）、PINGREQ（值为 12，表示心跳请求）、DISCONNECT（值为 14，表示客户端断开连接）；

⑵ 服务端到客户端，包括 CONNACK（值为 2，表示连接报文确认）、SUBACK（值为 9，表示订阅请求报文确认）、UNSUBACK（值为 11，表示取消订阅报文确认）、PINGRESP（值为 13，表示心跳响应）；

⑶ 两个方向都允许，包括 PUBLISH（值为 3，表示发布消息）、PUBACK（值为 4，表示消息发布收到确认）、PUBREC（值为 5，表示发布收到）、PUBREL（值为 6，表示发布释放）、PUBCOMP（值为 7，表示消息发布完成）。

以上三类控制报文，虽然其报文的流动方向不同，但每一种都需要结合控制报文标志位。控制报文类型的标志位由固定报头第一个字节的二进制位 3-0 表示，其取值由控制报文类型决定。为了对 MQTT 控制报文进行更加全面的安全性检查，需要针对控制报文类型和控制报文类型标志位进行深度包解析。

5.3.4 基于白名单的 MQTT 协议深度包检测访问控制策略

对于采用了白名单策略的安全防护系统来说，其默认的安全规则是阻断所有的通信数据包，只允许完全满足了特定安全规则中所有条件的数据包通过防护系统，只要不满足安全规则中的任何一项条件，都会执行阻断操作。本文设计的 MQTT 协议深度包访问控制模块就采用了白名单策略，包括了 TCP/IP 数据包的 IP 地址、端口的检测，以及对 MQTT 数据包的固定报头、剩余长度、控制报文类型以及控制报文类型标志位的检测，其访问控制安全规则的组成为：

Source_IP + Destination_IP + Destination_Port + Control_Message_Type + Control_Message_Type_Flag + Action

其中 Source_IP/Destination_IP 表示 TCP/IP 数据包的源/目 IP 地址，Destination_Port 表示 TCP/IP 数据包的目的端口，Control_Message_Type 和 Control_Message_Type_Flag 分别表示 MQTT 数据包的控制报文类型和控制报文类型标志位，Action 表示要对数据包执行的操作。

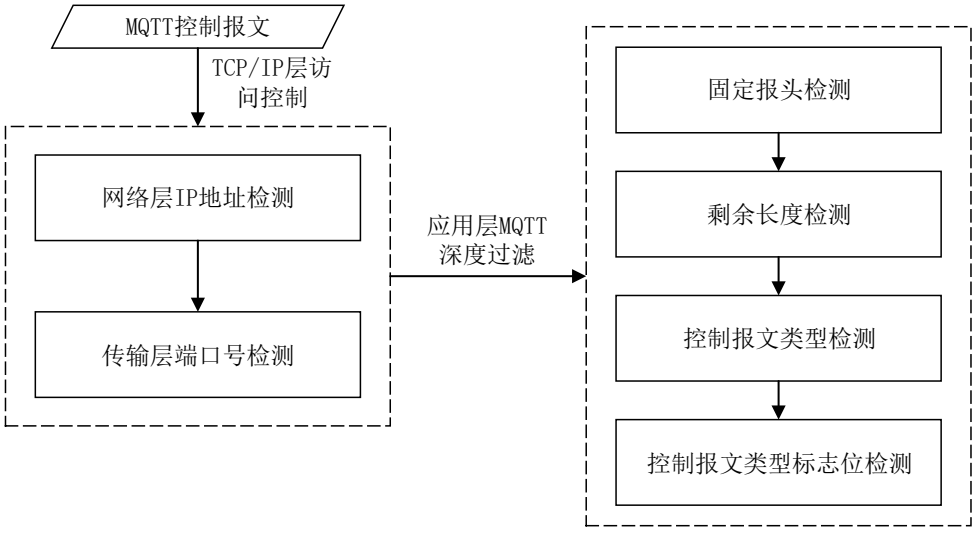

图 5-9 基于白名单的 MQTT 深度包检测流程

图 5-9 显示了基于白名单的 MQTT 深度包检测流程，不仅在网络层和传输层上采取了针对 IP 地址和端口号的访问限制，还在应用层上对 MQTT 控制报文进行了深度过滤，在一定程度上能够保护 MQTT 服务器不受伪造数据包攻击的危害。

5.4 基于朴素贝叶斯的异常流量检测模块设计

基于白名单的 MQTT 深度包检测防火墙能够防御具有明显异常特征的伪造数据包的攻击，但是对于符合协议规约并且满足访问控制规则的伪造数据包，这类

基于包过滤的防火墙不能有效地发现其攻击行为。

在正常情况下，工业物联网系统中的通信流量具有稳定的特征，对于 MQTT 协议来说，通信过程中产生的数据流量具有一定的周期性和稳定性，其数据包中的控制报文类型和控制报文类型标志位也具备统计规律。如果系统遭到精心伪造的数据包的攻击，那么在此期间系统网络中的通信流量会受到一定影响，MQTT 通信流量的周期性和稳定性也会遭到破坏。因此，可以在工业物联网系统正常运行的状态和遭到高级持续性攻击的状态下采集 MQTT 通信流量，并选取某种分类算法构建流量分析模型，根据这些流量数据对模型进行训练，然后使用训练好的流量分析模型对通信流量采取实时的异常行为检测，一旦发现异常流量，则说明系统可能遭受到了攻击，应该立即进行报警。由于工业物联网系统在稳定运行时 MQTT 控制报文的类型有限，只能获得数量有限的训练数据样本，并且训练样本的数据分布可能不平衡，比如一类样本的数量很多而其它类样本的数量很少，因此构建该全流量分析模型的关键在于选取合适的分类算法。

目前常用的分类算法有神经网络、决策树、最近邻算法以及朴素贝叶斯分类算法。神经网络算法在模型训练时很难选择一个合适的拓扑结构，并且需要很长的时间和较多的数据才能有比较好的分类效果；决策树算法在各类别样本数量不一致的情况下，决策树当中信息增益的结果会偏向于具有更多数据的特征属性；最近邻算法因为要存储所有的数据实例并且计算量很大，因此会消耗比较多的内存。基于以上分析，神经网络、决策树和最近邻算法均不适用于构建该 MQTT 流量分析模型。朴素贝叶斯分类是一种运算性能比较高的算法，如果假设的特征条件相互独立性成立的话，那么朴素贝叶斯将比其他的分类模型回归要快，并且只需要较少的训练数据集；即使特征条件相互独立的假设不成立，朴素贝叶斯分类器在实际使用中通常也能表现出良好的分类效果，同时还具有较高的分类效率。因此，本文使用朴素贝叶斯分类算法来设计并实现应用于 MQTT 协议的异常流量检测模块。

5.4.1 朴素贝叶斯分类与异常流量检测

1、朴素贝叶斯分类算法概述

朴素贝叶斯分类算法以贝叶斯定理为基础，其分类原理可以这样简单描述：根据待分类项的各个特征，计算待分类项映射到每一个类别的条件概率，即通过条件概率的大小来预测待分类项属于各个类别的可能性。图 5-10 为朴素贝叶斯分类方法的工作流程示意图：

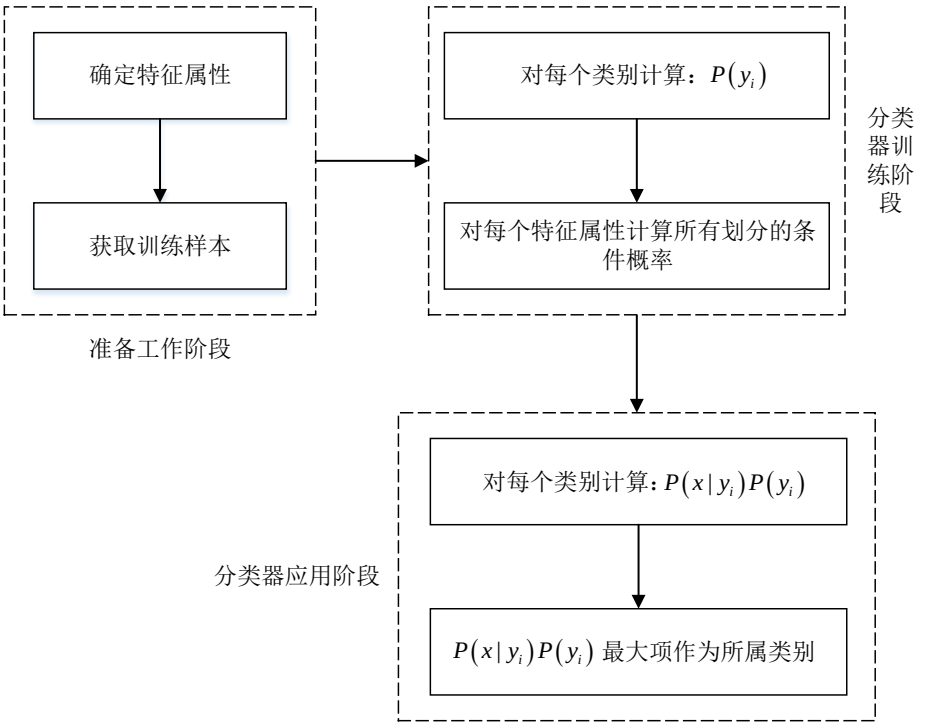

图 5-10 朴素贝叶斯分类流程

根据上图可知，朴素贝叶斯分类的工作流程包括以下阶段：

1、准备工作阶段。首先，需要依据实际情况来为分类项选取合适的特征属性，并且对已经确定的特征属性进行划分，然后将一部分待分类项进行分类，分类完成的这一部分数据将作为训练样本对分类器进行训练，特征属性的选取、划分以及训练样本的质量将对分类器的分类正确率产生很大影响。该阶段可以描述为以下两个步骤：

⑴ 假设特征属性个数为 m，$x=\{a_1,a_2,...,a_m\}$ 是一个待分类项，a_i 是 x 的一个特征属性，其中 $i=1,2,3,...,m$；

⑵ 划分出类别集合，设为 $C=\{y_1,y_2,...,y_n\}$。

2、分类器训练阶段。根据准备工作阶段提供的特征属性划分和训练样本，计算每个类别在训练样本中出现的概率，以及每一个特征属性划分在每一个类别下的条件概率。该阶段可以描述为以下步骤：

⑴ 计算样本集合中每个类别的概率 $P(y_i)$，$i=1,2,3,...,n$；

⑵ 计算在各个类别下各个特征属性的条件概率，即 $P(a_1|y_i),P(a_2|y_i),...,P(a_m|y_i)$，$i=1,2,3,...,n$；

3、分类器应用阶段。根据分类器训练阶段得到的计算结果，对待分类项进行分类预测，待分类项的特征属性作为输入，输出是待分类项对每一个类别的识别率（即待分类项属于每一个类别的条件概率）。该阶段可以描述为以下步骤：

⑴ 计算条件概率：$P(y_1|x)$，$P(y_2|x)$，…，$P(y_n|x)$。假设各个特征属性是条件相互独立的，根据贝叶斯定理可知，待分类项 x 属于类别 y_i 的概率为 $P(y_i|x)=\frac{P(x|y_i)P(y_i)}{P(x)}$，因为分母对于所有的类别来说均为常数，所以计算 $P(y_i|x)$ 时可以把分母省略，只需要计算分子，且因为各个特征属性相互独立的，所以有：$P(x|y_i)P(y_i)=P(a_1|y_i)P(a_2|y_i)...P(a_m|y_i)P(y_i)=P(y_i)\prod_{j=1}^{m}P(a_j|y_i)$；

⑵ 如果 $P(y_k|x)=\max\{P(y_1|x)P(y_2|x),...,P(y_n|x)\}$，则 x 属于类别 y_k。

2、基于朴素贝叶斯分类器的异常流量检测

根据以上对朴素贝叶斯分类工作流程的分析，使用朴素贝叶斯分类器进行异常流量检测的工作流程也可以分为准备工作阶段、分类器训练阶段以及分类器应用阶段：

⑴ 在准备工作阶段，类别集合中包括正常流量和异常流量这两个类别，特征属性划分需要对通信协议进行协议特征属性选取，其中关键在于所选取的协议特征属性要能够尽量包含异常流量的攻击特征；

⑵ 在分类器训练阶段，主要是对训练样本进行数据预处理，用于训练的样本中要同时包含正常和异常的通信流量，然后计算正常流量和异常流量在训练样本中出现的概率，以及所选取的每个协议特征属性划分在正常流量和异常流量下的条件概率，得到朴素贝叶斯分类器；

⑶ 在应用阶段，对于待检测的通信数据包，提取相应的特征属性作为待分类项，然后将待分类项输入到训练好的分类器中就能得到检测结果。

5.4.2 MQTT 协议特征属性选取及数据预处理

由上一小节可知，在准备工作阶段，特征属性的选取、划分以及训练样本的质量都将对分类器的分类正确率产生很大影响。因此，构建应用于 MQTT 协议的朴素贝叶斯分类器，关键就在于准备工作阶段中对 MQTT 协议提取能够有效描述控制报文类型的特征属性，以及在分类器训练阶段对 MQTT 训练样本进行数据预处理，得到构建朴素贝叶斯分类器的各项参数。

1、MQTT 协议特征属性选取

MQTT 控制报文中，最重要的协议特征是固定报头中的控制报文类型和控制报文类型标志位，这两个特征决定了控制报文的类型，因此本文选取 MQTT 协议的控制报文类型和控制报文类型标志位作为特征属性，构建基于朴素贝叶斯的异常流量分类检测模型，对 MQTT 协议通信流量进行异常检测。提取以上特征属性的过程如下：

⑴ 按照时间顺序，每间隔时间 T（通常为 1 分钟）采集 MQTT 的通信流量，得到若干组 MQTT 正常流量和异常流量的数据包作为训练样本，该训练样本的大小记为 N；

⑵ 对于每个 MQTT 数据包，分别提取固定报头中控制报文类型（记为特征属性 x_1）和指定控制报文类型的标志位（记为特征属性 x_2），从而得到每个数据包的特征属性向量 $X_i=\left[x_1^i,x_2^i\right]$，其中 $i=1,2,3,.....,N$，x_1^i 和 x_2^i 分别表示第 i 个数据包的特征属性 x_1 和 x_2 的取值。

2、MQTT 训练样本数据预处理

提取出 MQTT 数据流的特征属向量后，可以得到特征属性向量的样本集合 $\{X_n\}$，$n=1,2,3,.....,N$。在建立朴素贝叶斯分类检测模型前还需要对特征属性向量进行数据预处理，得到朴素贝叶斯分类器的各项参数。数据预处理的过程如下：

⑴ 定义集合 $M=\{(X_1,Y_1),(X_2,Y_2),...,(X_n,Y_n)\}$，其中 $n=1,2,3,.....,N$，$X_i\in\{X_n\}$，$Y_i\in\{0,1\}$。对于每个特征属性向量 ，如果对应的数据包属于正常流量，令其标识 $Y_i=1$；如果对应的数据包属于异常流量，则令其标识 $Y_i=0$。

⑵ 统计集合 M 中正常特征属性向量（即标识 $Y_i=1$ 的特征向量）与异常特征属性向量（标识 $Y_i=0$ 的特征向量）出现的概率，正常向量的概率记为 $P(Y=1)$，异常向量的概率记为 $P(Y=0)$。

⑶ 对于集合 M 中的正常特征属性向量，按照公式（5-1）计算特征属性 x_1 不同取值的条件概率：

$$P(x_1=j\,|\,Y=1)=\frac{\sum_{i=1}^{N}I(x_1=j,Y_i=1)+\lambda}{\sum_{i=1}^{N}I(Y_i=1)+s_1\lambda} \tag{5-1}$$

其中，λ 取值通常为 1，j 表示特征属性 x_1 的具体取值，$j=1,2,3,...,s_1$，s_1 表示控制报文类型的数量；

按照公式（5-2）计算特征属性 x_2 不同取值的条件概率：

$$P(x_2=k\,|\,Y=1)=\frac{\sum_{i=1}^{N}I(x_2=k,Y_i=1)+\lambda}{\sum_{i=1}^{N}I(Y_i=1)+s_2\lambda} \tag{5-2}$$

其中，k 表示特征属性 x_2 的具体取值，$k=1,2,3,...,s_2-1$，s_2 表示控制报文类型标志位取值的数量。

⑷ 对于集合 M 中的异常特征属性向量，按照公式（5-3）计算特征属性 x_1 不同取值的条件概率：

$$P(x_1 = j \mid Y = 0) = \frac{\sum_{i=1}^{N} I(x_1 = j, Y_i = 0) + \lambda}{\sum_{i=1}^{N} I(Y_i = 0) + s_1\lambda} \quad (5\text{-}3)$$

按照公式（5-4）计算特征属性 x_2 不同取值的条件概率：

$$P(x_2 = k \mid Y = 0) = \frac{\sum_{i=1}^{N} I(x_2 = k, Y_i = 0) + \lambda}{\sum_{i=1}^{N} I(Y_i = 0) + s_2\lambda} \quad (5\text{-}4)$$

通过以上数据预处理方法，分别在正常流量和异常流量这两个类别条件下，计算出了 MQTT 控制报文类型和控制报文类型标志位这两个特征属性划分的频率，为建立朴素贝叶斯分类模型提供了所需要的各项参数。

5.4.3 基于朴素贝叶斯的 MQTT 协议异常流量检测模型

图 5-11 是基于朴素贝叶斯的 MQTT 异常流量检测模型示意图。在分类器的训练阶段需要正常流量和异常流量这两类数据，正常 MQTT 数据包标记为 1，异常 MQTT 数据包标记为 0。分类器训练结束后会得到一个基于 MQTT 协议的朴素贝叶斯分类器，该分类器将对待检测的 MQTT 控制报文进行概率意义上的分类估计。每一个待检测项包含通信数据包中提取的特征属性，类别集合中包含正常流量和异常流量这两个类别。假设待检测项中的两个特征属性互相独立，使用样本数据预处理阶段得到的各项参数计算出待检测项对正常流量和异常流量的识别率（即待检测项属于正常流量类别和属于异常流量类别的概率，分别记为 P(正常)和 P(异常)，识别率最高的即为当前待检测项所属的流量类别。

对于待检测的 MQTT 数据包，提取固定报头中控制报文类型和指定控制报文类型的标志位，得到待分类项 $X = [x_1, x_2]$，该待分类项属于正常流量和异常流量的概率分别记为 $P(Y = 1 \mid X)$ 和 $P(Y = 0 \mid X)$。根据贝叶斯分类器的各项参数，分别进行计算以下相对条件概率：

$$P(Y = 1 \mid X) = P(x_1 \mid Y = 1)P(x_2 \mid Y = 1)P(Y = 1) \quad (5\text{-}5)$$

$$P(Y = 0 \mid X) = P(x_1 \mid Y = 0)P(x_2 \mid Y = 0)P(Y = 0) \quad (5\text{-}6)$$

然后，对以上相对条件概率 $P(Y = 1 \mid X)$ 和 $P(Y = 0 \mid X)$ 进行比较，如果 $P(Y = 1 \mid X) > P(Y = 0 \mid X)$，则说明该 MQTT 控制报文属于正常流量的概率要大于属于异常流量的概率，判定该数据包属于正常流量；如果 $P(Y = 1 \mid X) < P(Y = 0 \mid X)$，则说明该 MQTT 控制报文属于异常流量的概率大于属于正常流量的概率，判定该数据包属于异常流量

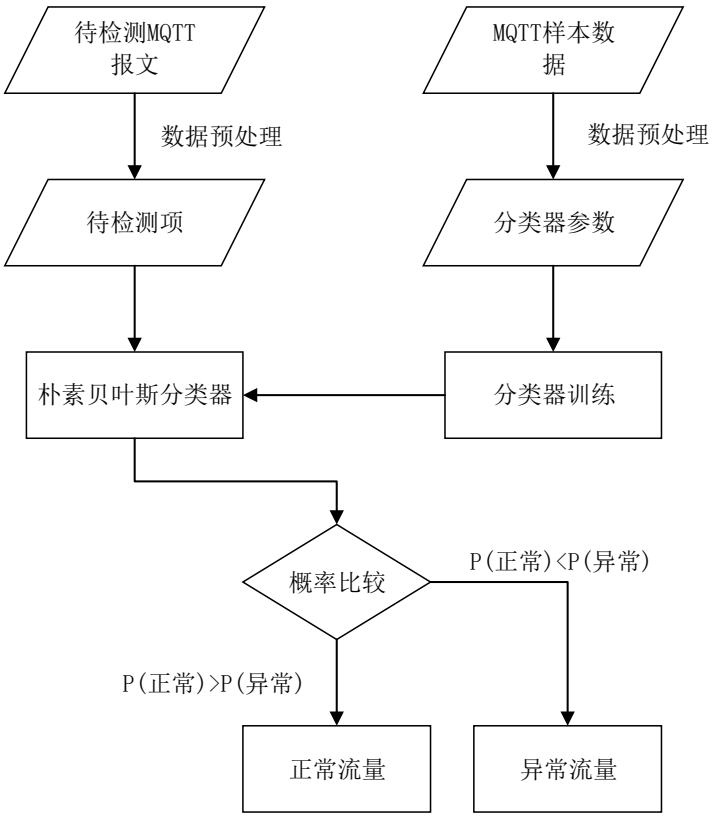

图 5-11 基于朴素贝叶斯的 MQTT 异常流量检测模型

基于以上异常流量检测模型，本文设计了完整的基于朴素贝叶斯分类器的 MQTT 协议异常流量检测模块，该模块的工作流程如图 5-12 所示，其流程可以概括为以下步骤：

⑴ 检测开始后，对 MQTT 服务器与 MQTT 客户端之间的通信流量进行采集，并对采集到的数据包进行目的地址和端口号检测，筛选出 MQTT 协议控制报文，对其他类型的数据包不作任何处理；

⑵ 对采集到的 MQTT 控制报文进行特征属性提取，主要在控制报文的固定报头中提取控制报文类型和控制报文类型标志位，并将提取到的数据进行保存；

⑶ 对提取到的 MQTT 特征属性进行数据预处理，得到符合朴素贝叶斯分类模型输入的待分类项；

⑷ 将待分类项输入到已经完成训练的朴素贝叶斯分类模型，得到待分类项的分类预测结果，如果 P(正常)> P(异常)，判定该 MQTT 报文是正常流量，不做任何处理；如果 P(异常)> P(正常)，判定该 MQTT 报文是异常流量并发出报警。

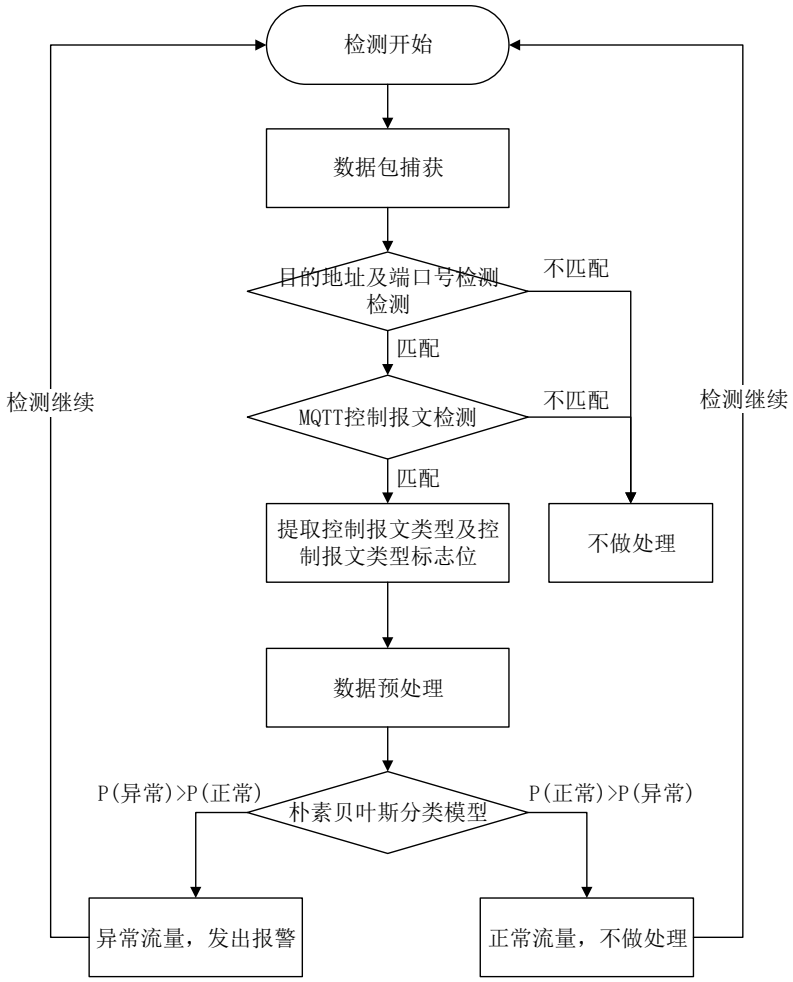

图 5-12 MQTT 协议异常流量检测流程

5.5 本章小结

本章针对数据处理层的 MQTT 服务器进行防护策略设计，主要从深度包检测和异常流量检测两个方面进行安全防护。基于白名单的深度包检测模块能够对 MQTT 协议进行深度解析和访问控制，包括对 MQTT 控制报文进行完整性和准确性检测，拒绝具有明显异常特征和不符合协议规约的数据包；同时，基于朴素贝叶斯的异常流量检测系统，能够对数据处理层 MQTT 服务器的通信流量进行实时监控，可以检测出深度包检测系统无法检测出来异常数据包，并及时进行报警。两种防护措施配合使用，对数据处理层的通信服务器进行比较全面的安全防护。

第六章 工业物联网安全防护模块测试及分析

6.1 总体测试环境搭建

本文设计的各个工业物联网安全防护模块将在如图 6-1 所示的工业信息安全攻防演练平台上进行测试。

图 6-1 工业信息安全攻防演练平台

图 6-1 所示的工业信息安全攻防演练平台由 PLC、电机、工业 PC、防火墙硬件设备、现场/远程监控终端、数据采集器、网络性能测试设备以及各种网络外设组成。为了对本文设计的各个安全模块进行测试，使用上述平台中的各类设备搭建了类似于本文第二章中图 2.4 所示的工业物联网数据采集及远程监控系统，并对该系统所有安全模块的测试都将在该系统中进行。将各个安全防护模块部署到该系统中后，其拓扑结构如图 6-2 所示，针对物联网僵尸网络病毒的防御方法直接部署到数据采集终端当中；MQTT 深度包检测防火墙部署在 MQTT 代理服务器前面，对访问 MQTT 服务器的数据包进行深度过滤；而 MQTT 异常流量检测服务器通过旁路接入到 MQTT 服务器所在网络，对该网络中的 MQTT 流量进行实时监测；模拟攻击源中包括 Mirai 病毒攻击源、MQTT 异常数据包攻击源以及网络嗅探客户端。

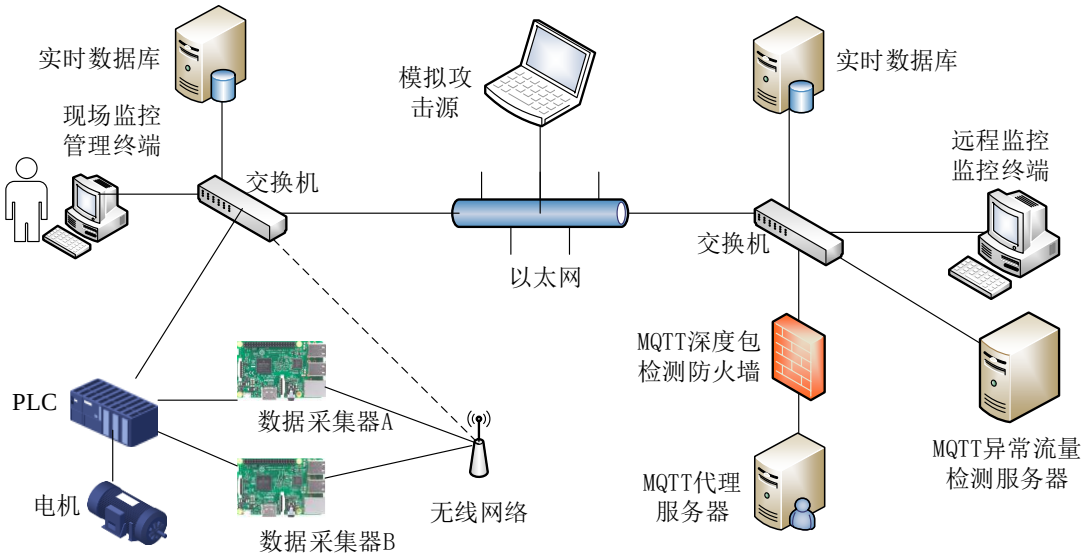

图 6-2 工业物联网安全防护技术整体测试模拟环境

6.2 物联网僵尸网络病毒攻击的防护方法测试

为了便于描述和分析物联网僵尸网络病毒攻击的防护测试过程，将图 6-2 所示的整体测试环境的网络结构进行简化，得到如图 6-3 所示的物联网僵尸网络病毒攻击防护测试实验环境。本实验以防御 Mirai 病毒为例，实验环境中将 Mirai 配置服务器、文件服务器和 CNC 控制服务器都集成到同一台主机上作为攻击源，物联网数据采集终端设备使用树莓派，现场监控管理终端负责对数据采集终端配置安全防护规则。本实验在数据采集终端不采取任何防护措施和采取安全防护措施这两种情况下，使用 Mirai 攻击源对其发起感染，从而验证采取的安全措施对 Mirai 攻击的防御效果。

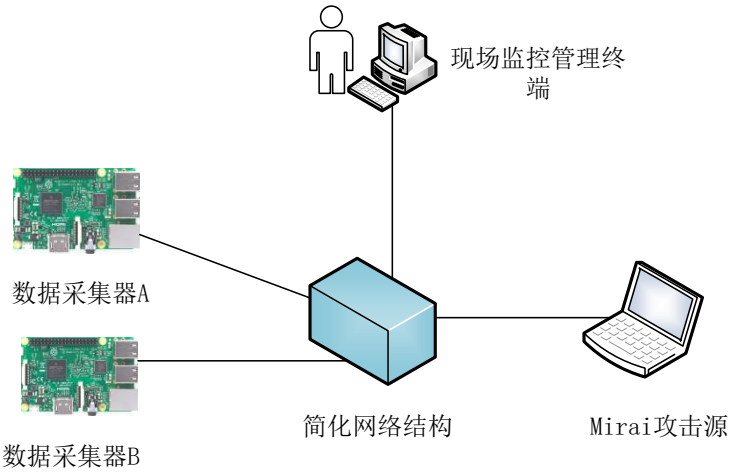

图 6-3 Mirai 攻击防护测试环境

1、数据采集终端 A 接入网络，且不采取任何安全措施

使用搭建 Mirai 攻击源时设置的用户名和密码登录 CNC 控制服务器，如图 6-4 所示的状态表示已经成功登录 CNC 服务器。

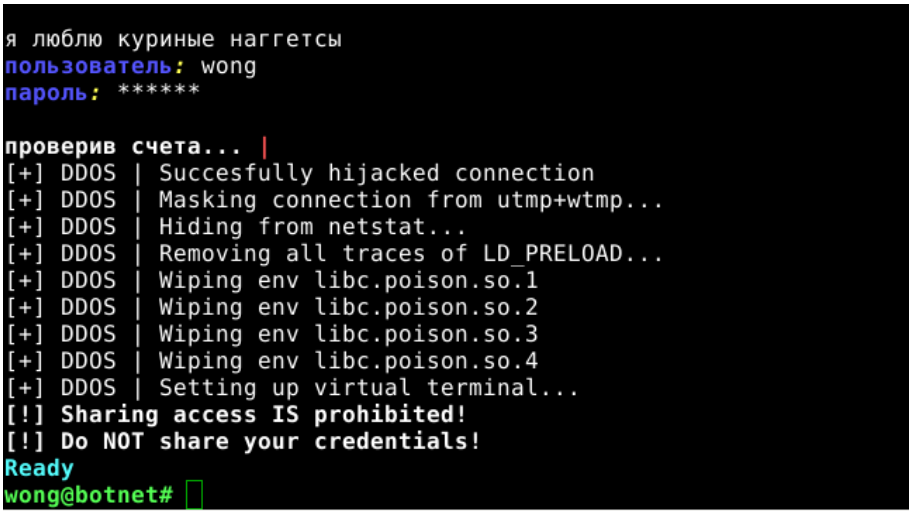

图 6-4 成功登录到 CNC 服务器

登录 CNC 控制服务器后，对 Bot 模块下发攻击指令，对该局域网络中的设备进行弱口令扫描。如下图 6-5 所示，已经扫描到了 IP 为 172.16.10.4 的数据采集终端设备 A，并使用 root:123456 的用户名/密码组合进行 Telnet 爆破。

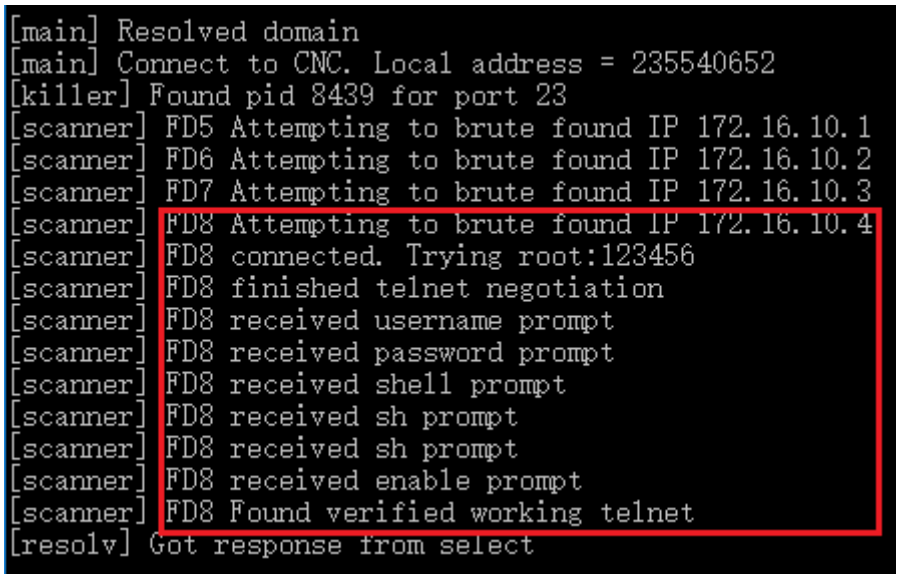

图 6-5 对局域网内设备进行弱口令扫描

爆破成功后，会将得到的结果发送给 ScanListen 模块，接下来进行一系列感染操作。最终，数据采集终端设备完全遭到恶意程序的控制后，会连接到 CNC 控制服务器，如图 6-6 所示。

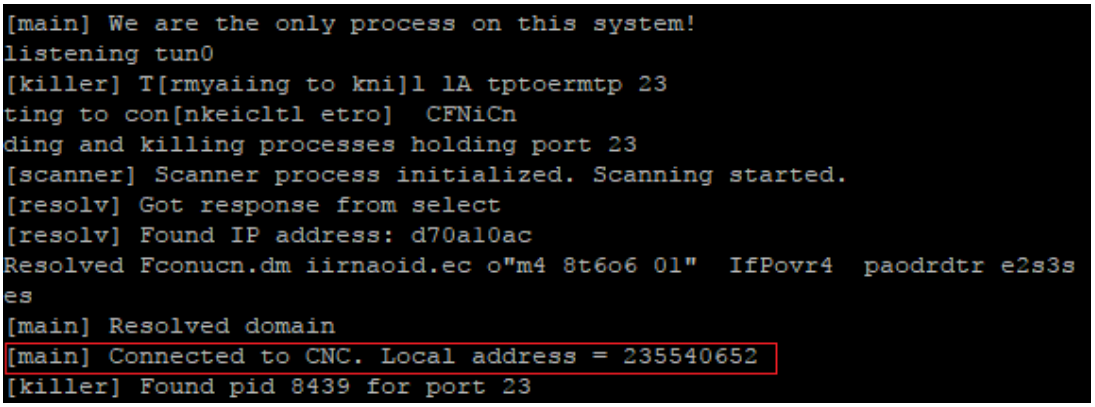

图 6-6 数据采集终端设备彻底遭到 Mirai 感染

2、将数据采集终端 B 接入网络，并部署端口扫描和暴力破击防护措施

⑴ 端口扫描检测及防护测试

对数据采集终端 B 进行安全配置：首先开启 snort 中的 sfportscan 端口扫描检测预处理器，然后向 snort 配置 3.2.1 节所述的 syn 端口扫描检测规则，并添加如下的防火墙和 snort 联动的 iptables 规则：

iptables -I INPUT -j NFQUEUE –queue-num 1 –queue-bypass

完成端口扫描检测防护规则配置后，再次使用 Mirai 攻击源对测试环境中的数据采集终端设备进行扫描和感染。查看 snort 相关日志，发现 snort 检测到了 Mirai 对 IP 地址为 172.16.10.105 的数据采集设备进行的 syn 端口扫描行为，如图 6-7 所示。

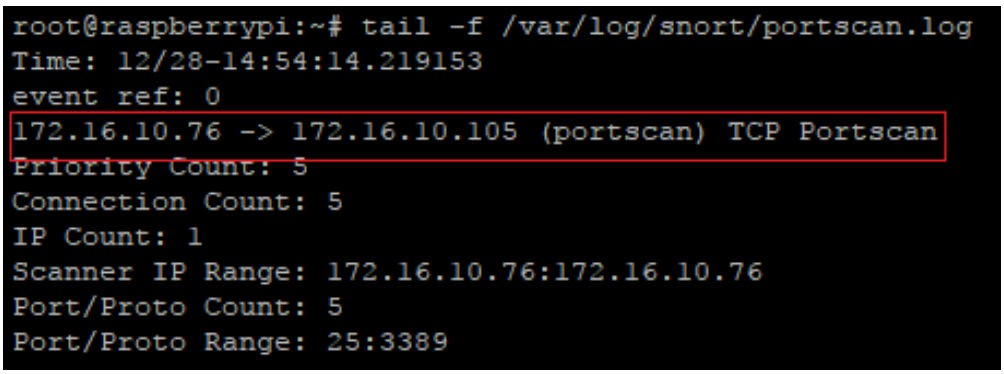

图 6-7 syn 端口扫描检测日志

查看 iptables 防火墙信息，发现防火墙中已经添加了新的安全规则，限制了 Mirai 攻击源（IP 地址为 172.16.10.76）对数据采集设备的访问，如图 6-8 所示。

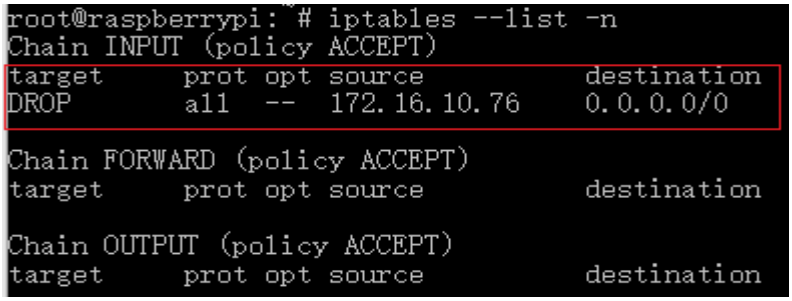

图 6-8 iptables 防火墙防护规则

⑵ telnet 暴力破解检测及防护测试

为了单独测试 telnet 暴力破解检测及防护方法的有效性，需要关闭上一个测试中的 snort 服务，然后为 telnet 服务配置防暴力破解的 iptables 防火墙规则，相关的配置脚本如 3.2.2 小节所述。完成暴力破解的防护规则配置后，再次使用 Mirai 攻击源对测试环境中的数据采集终端设备进行扫描和感染。查看 telnet 服务的登录日志，发现 Mirai 攻击源（IP 地址为 172.16.10.76）对数据采集终端设备 B 进行了两次登录尝试，如图 6-9 所示。

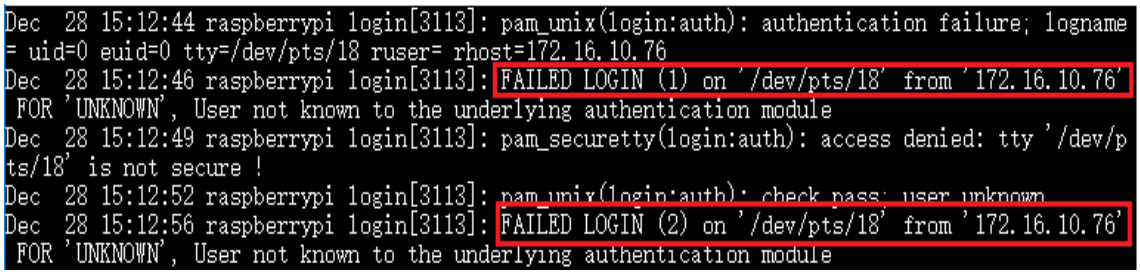

图 6-9 telnet 登录日志

查看 iptables 防火墙的日志信息，如图 6-10 所示，发现防火墙已经阻断了 Mirai 攻击源的第 3 次 telnet 登录尝试，说明成功检测到了 Mirai 攻击源的前两次失败登录，于是判定 Mirai 攻击源正在对设备发起暴力破解，然后对 Mirai 攻击源的 IP 地址进行了屏蔽。

图 6-10 iptables 限制 Mirai 攻击源登录日志

根据以上测试效果，在数据采集终端不采取任何防护措施的情况下很容易被 Mirai 病毒感染，而采取端口扫描和暴力破解防御措施后，能够有效检测到 Mirai 攻击源在对数据采集终端进行感染的过程中发起的端口扫描和暴力破解这两种入

侵行为，然后 iptables 防火墙对 Mirai 攻击源的 IP 地址进行了屏蔽，限制该 IP 地址对终端设备的访问，确保了终端设备的安全。

6.3 MQTT 协议通信加密及身份认证测试

为了便于描述和分析 MQTT 通信加密及身份认证的测试过程，将图 6-2 所示的整体测试环境的网络结构进行简化，得到如图 6-11 的 MQTT 通信加密和身份认证测试环境，其中网络嗅探客户端主机用于抓取服务器和客户端通信数据。

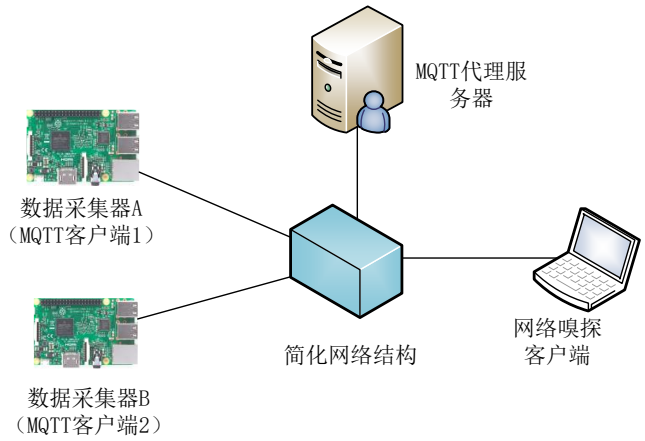

图 6-11 MQTT 加密和认证测试环境

本实验中，使用 Eclipse 提供的开源类库 Eclipse Paho 来实现 MQTT 协议的客户端和服务器程序；同时，对 MQTT 进行 TLS 加密和 X509 认证，使用开源框架 OpenSSL 提供的密码算法库和 TLS/SSL 协议库来实现。应用程序的开发过程不是本文讨论的重点，所以不在此赘述其具体实现过程。

对于 MQTT 的通信加密测试，采用 TLS 对 MQTT 客户端 1 和 MQTT 服务器之间的通信进行加密，使用抓包客户端来抓取两者之间的通信流量，验证其加密效果。对于 MQTT 的身份认证测试，仅对 MQTT 客户端 1 进行 X509 身份认证，然后分别使用两个 MQTT 客户端连接 MQTT 服务器，验证身份认证的结果。

6.3.1 基于 TLS 的 MQTT 通信加密测试

为测试 MQTT 通信加密，本实验在 MQTT 客户端 1 不加密和采取 TLS 加密这两种情况下抓取客户端和服务器的通信数据，通过对比分析来验证 MQTT 采取 TLS 加密的效果。

1、不采取加密

在 MQTT 客户端和服务器不进行通信加密的情况下，MQTT 客户端 1 连接到服务器，并订阅主题为“encryption test”的消息。其中，服务器 IP 地址为

172.16.10.215，通信不加密的情况下默认使用的端口号为 1883。然后服务器向客户端发送主题为“encryption test”、内容为“uncrypted message”的消息，客户端收到该消息内容如图 6-12 所示。

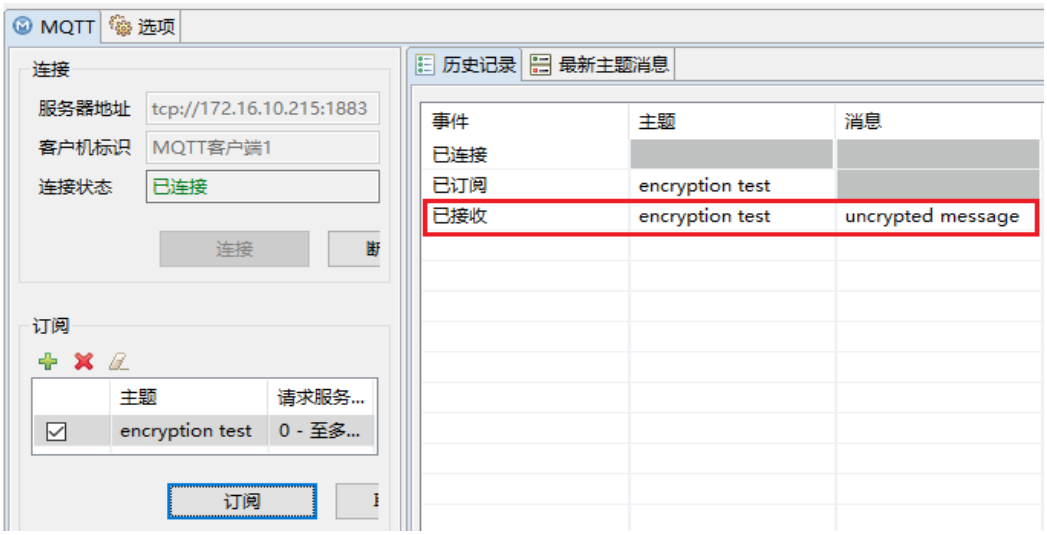

图 6-12 MQTT 客户端收到未加密消息

在此期间，使用抓包客户端抓取 MQTT 客户端 1 和服务器间的通信数据，其结果如图 6-13 所示。

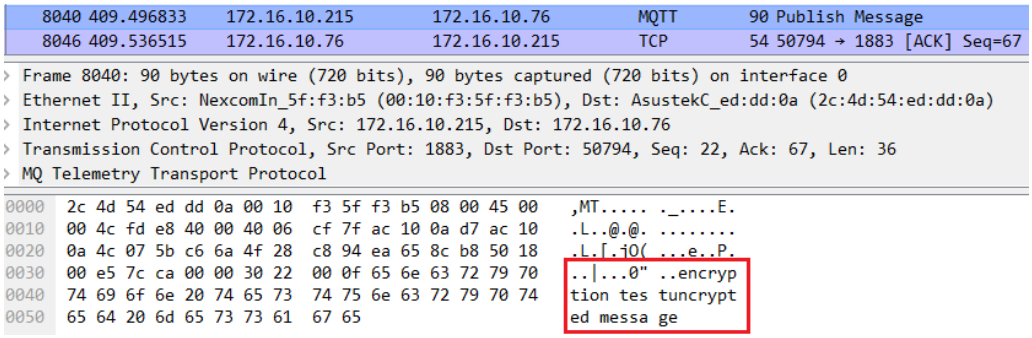

图 6-13 MQTT 客户端和服务器间未加密数据抓包

由图 6-11 中可以看出，在不进行加密的情况下，MQTT 客户端 1 和服务器之间的通信数据是明文传输的。

2、采取 TLS 加密

使用 4.3.1 小节所述的 TLS 对 MQTT 通信进行加密的脚本来生成 CA 安全证书（mosq-ca.crt）、服务器私钥（mosq-serv.key）和服务器安全证书（mosq-serv.crt），然后把以上证书和私钥的存储路径添加到服务器配置文件中，并将 MQTT 默认端口改为 8883 端口（方便和未加密情况下使用 1883 端口进行比较）。至此，MQTT 客户端 1 和服务器间的 TLS 通信加密设置完成， MQTT 客户端 1 连接到服务器，

使用抓包客户端抓取 MQTT 客户端 1 和服务器间的握手协商阶段的通信数据，如图 6-14 所示。

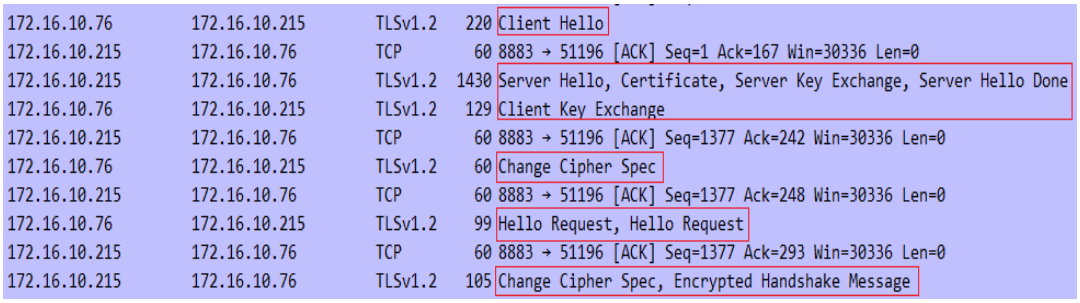

图 6-14 握手协商阶段通信数据

握手协商阶段完成后，MQTT 客户端 1 向服务器发送主题为“encryption test”、内容为“crypted message”的消息，并使用抓包客户端抓取 MQTT 客户端 1 和服务器间的通信数据，如图 6-15 所示。

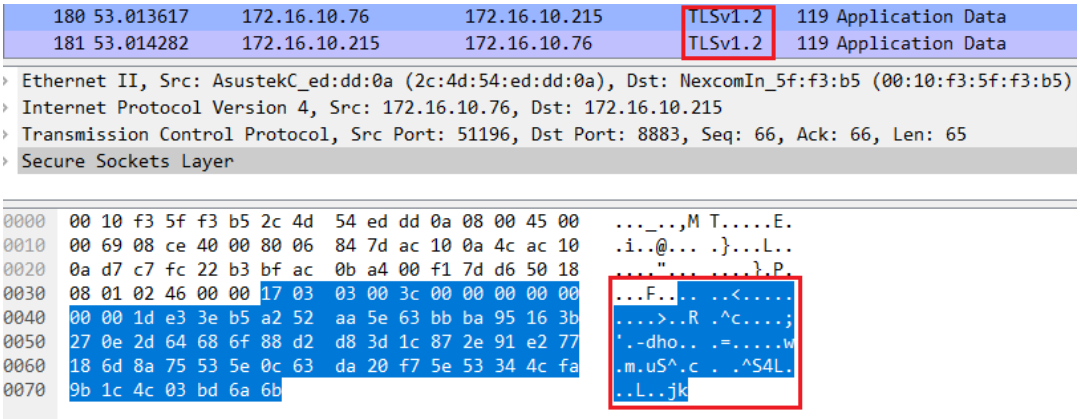

图 6-15 加密通信数据抓包

由上图可以看出，MQTT 客户端 1 和服务器间的通信按照 TLS 协议的规范，使用握手协商阶段生成的会话秘钥来加密通信数据。相比于不采取加密措施的传输过程，使用 TLS 加密 MQTT 控制报文提高了数据传输层的安全性。

6.3.2 基于 X509 的 MQTT 身份认证测试

为测试 MQTT 身份认证，本实验在 MQTT 服务器端开启对客户端的 X509 身份认证，MQTT 客户端 1 使用 X509 证书连接服务器，MQTT 客户端 2 不使用证书连接服务器，通过对比这两种情况来验证 MQTT 采取 X509 身份认证的效果。

首先，使用 4.3.2 小节所述的 OpenSSL 来对 MQTT 客户端进行身份认证的脚本来为 MQTT 客户端 1 生成 X509 证书。MQTT 客户端 1 使用 X509 证书（即 client.crt）连接到到服务器端，其结果如图 6-16 所示。

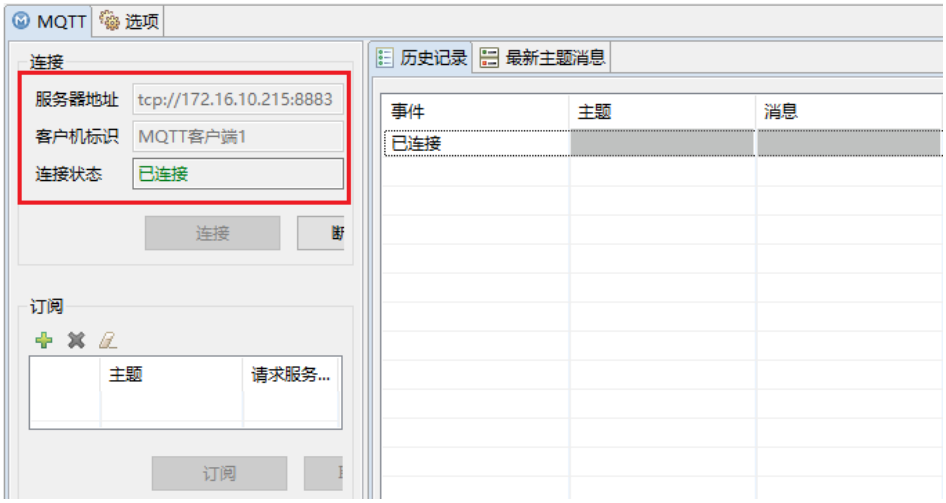

图 6-16 MQTT 客户端 1 使用 X509 证书连接到服务器

由上图可知，服务器端开启身份认证机制后，MQTT 客户端 1 使用 X509 安全证书能够通过服务器端的身份验证，成功连接到服务器。

MQTT 客户端 2 不使用 X509 安全证书连接服务器，其结果如图 6-17 所示。

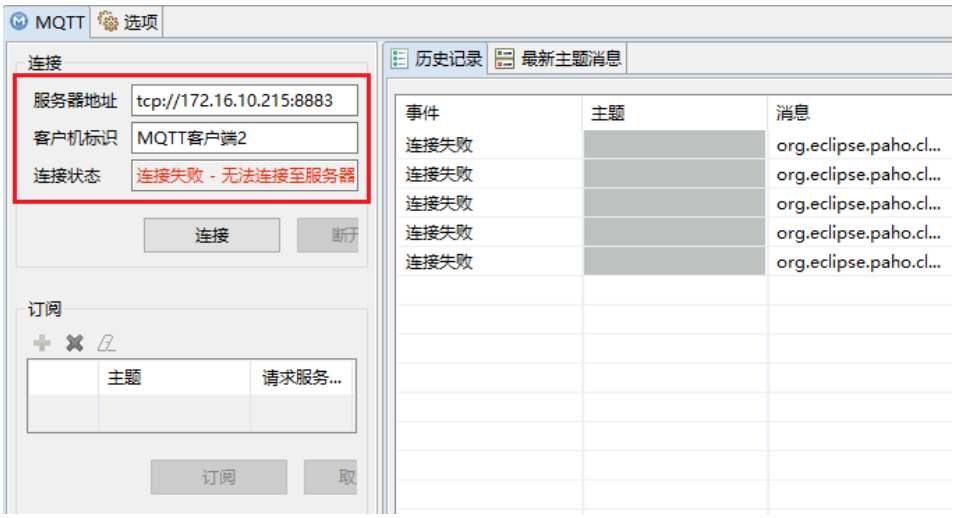

图 6-17 MQTT 客户端 2 不使用 X509 安全证书连接服务器

由上图可知，MQTT 客户端 2 不使用 X509 安全证书连接服务器时，经过多次连接尝试都没有通过服务器端的身份验证，连接服务器失败。

从以上测试结果可以看出，MQTT 服务器端开启基于 X509 证书的身份认证后，只有具备合法 X509 证书的客户端设备能够成功通过服务端的身份验证并连接到服务器，提高了 MQTT 通信过程的安全性。

6.4 基于深度包检测的访问控制模块测试

为了便于描述和分析 MQTT 深度包检测模块的测试过程，将图 6-2 所示的整体测试环境的网络结构进行简化，得到如图 6-18 所示的 MQTT 深度包检测试验环境。

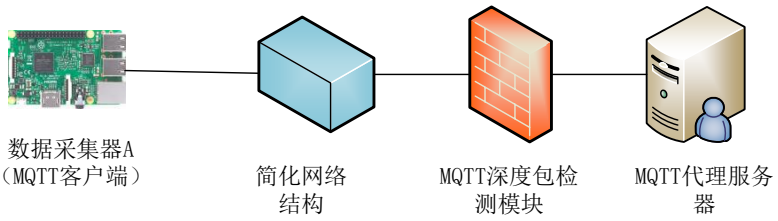

图 6-18 MQTT 深度包检测实验环境

6.4.1 深度包检测模块功能验证

1、控制报文类型检测

MQTT 客户端连接到服务器后，首先发送控制报文类型正确（这里以 SUBSCRIBE 控制报文为例）的数据包。然后将控制报文类型的值改为 0000，向服务器发送该数据包，检测结果如图 6-19 所示。对于修改后的控制报文，控制报文类型检测的结果为异常，深度包检测防火墙拒绝其通过。

```
[1120236.521465]==============start==============
[1120236.521466]app_data len=12,total_data_len=14
[1120236.521467]Packet for source address:172.16.10.100
[1120236.521468]Packet for destination address:172.16.10.120
[1120236.521469]app_data:00 7f 00 00 00 06 01 02 00 19 00 01
[1120236.521470]
[1120236.521471]**********MQTT DPI**********
[1120236.521563]Fixed header check OK
[1120236.521474]Client to Server
[1120236.521475]Message Type name:SUBSCRIBE, value:1000
[1120236.521476]Flags value:0010
[1120236.521478]Message Type value check OK
[1120236.521479]type and flags value check OK
[1120236.521480]app_data len check OK
[1120236.521481]MQTT decode:Legal Packet!! Success!!
[1120236.521483]==============end==============
[1120236.521484]
[1120236.521485]
[1120236.521486]==============start==============
[1120236.521488]app_data len=12,total_data_len=14
[1120236.521489]Packet for source address:172.16.10.100
[1120236.521490]Packet for destination address:172.16.10.120
[1120236.521491]app_data:00 7f 00 11 20 06 01 02 00 19 10 01
[1120236.521492]
[1120236.521493]**********MQTT DPI**********
[1120236.521563]Fixed header check OK
[1120236.521495]Client to Server
[1120236.521496]Message Type name:Reserved, value:0000
[1120236.521497]Flags value:0010
[1120236.521498]Message Type value check FAIL
[1120236.521499]type and flags value check FAIL
[1120236.521500]MQTT decode:Illegal Packet!! Failed!!
[1120236.521502]==============end==============
```

图 6-19 控制报文类型检测记录

2、控制报文类型标志位检测

将 SUBSCRIBE 控制报文的控制报文类型标志位的值改为 0000，向服务器发

送该伪造的数据包。深度包检测的结果如图 6-20 所示，控制报文类型标志位检测为异常，深度包检测防火墙拒绝该数据包通过。

```
[1120236.521507]==============start==============
[1120236.521508]app_data len=12,total_data_len=14
[1120236.521509]Packet for source address:172.16.10.100
[1120236.521511]Packet for destination address:172.16.10.120
[1120236.521513]app_data:00 7f 00 00 00 06 01 02 00 19 00 01
[1120236.521514]
[1120236.521516]**********MQTT DPI**********
[1120236.521563]Fixed header check OK
[1120236.521518]Client to Server
[1120236.521520]Message Type name:SUBSCRIBE, value:1000
[1120236.521522]Flags value:0010
[1120236.521524]Message Type value check OK
[1120236.521525]type and flags value check OK
[1120236.521526]app_data len check OK
[1120236.521528]MQTT decode:Legal Packet!! Success!!
[1120236.521529]==============end==============
[1120236.521530]
[1120236.521532]
[1120236.521533]==============start==============
[1120236.521534]app_data len=12,total_data_len=14
[1120236.521536]Packet for source address:172.16.10.100
[1120236.521537]Packet for destination address:172.16.10.120
[1120236.521538]app_data:2e 7b 00 11 24 26 01 02 00 19 15 37
[1120236.521539]
[1120236.521540]**********MQTT DPI**********
[1120236.521563]Fixed header check OK
[1120236.521543]Client to Server
[1120236.521544]Message Type name:SUBSCRIBE, value:1000
[1120236.521546]Flags value:0000
[1120236.521547]Message type value check FAIL
[1120236.521548]MQTT decode:Illegal Packet!! Failed!!
[1120236.521549]==============end==============
```

图 6-20 控制报文类型标志位检测记录

3、剩余长度检测

在正常的控制报文后面增加两个字节的数据，使得实际剩余长度与指定的剩余长度不匹配，检测结果如图 6-21 所示，有效载荷的长度检测为异常。

```
[1120236.521554]==============start==============
[1120236.521555]app_data len=12,total_data_len=14
[1120236.521557]Packet for source address:172.16.10.100
[1120236.521559]Packet for destination address:172.16.10.120
[1120236.521560]app_data:00 7f 00 00 00 06 01 02 00 19 00 01
[1120236.521561]
[1120236.521562]**********MQTT DPI**********
[1120236.521563]Fixed header check OK
[1120236.521564]Client to Server
[1120236.521566]Message Type name:SUBSCRIBE, value:1000
[1120236.521567]Flags value:0010
[1120236.521568]Message Type value check OK
[1120236.521569]type and flags value check OK
[1120236.521570]app_data len check OK
[1120236.521571]MQTT decode:Legal Packet!! Success!!
[1120236.521572]==============end==============
[1120236.521573]
[1120236.521575]
[1120236.521576]==============start==============
[1120236.521577]app_data len=14,total_data_len=16
[1120236.521578]Packet for source address:172.16.10.100
[1120236.521579]Packet for destination address:172.16.10.120
[1120236.521580]app_data:2e 7b 00 11 24 26 01 02 00 19 15 37 03 18
[1120236.521582]
[1120236.521583]**********MQTT DPI**********
[1120236.521563]Fixed header check OK
[1120236.521586]Client to Server
[1120236.521587]Message Type name:SUBSCRIBE, value:1000
[1120236.521588]Flags value:0010
[1120236.521590]Message type value check OK
[1120236.521591]type and flags value check OK
[1120236.521592]app_data len check Fail
[1120236.521593]MQTT decode:Illegal Packet!! Failed!!
[1120236.521594]==============end==============
```

图 6-21 剩余长度检测记录

4、固定报头检测

将正常的 SUBSCRIBE 控制报文该控制报文的固定报头两个字节的内容去掉，在有效载荷前面加上一个字节的可变报头。检测结果如图 6-22 所示，深度包检测模块检测到该报文缺失了固定报头，因此直接判断该报文是异常数据包，并拒绝该数据包通过。

```
[1120236.602358]==============start==============
[1120236.602360]app_data len=12,total_data_len=14
[1120236.602362]Packet for source address:172.16.10.100
[1120236.602364]Packet for destination address:172.16.10.120
[1120236.602366]app_data:00 7f 00 00 00 06 01 02 00 19 00 01
[1120236.602368]
[1120236.602370]**********MQTT DPI**********
[1120236.602372]Fixed header check OK
[1120236.602374]Client to Server
[1120236.602376]Message Type name:SUBSCRIBE, value:1000
[1120236.602378]Flags value:0010
[1120236.602380]Message Type value check OK
[1120236.602382]type and flags value check OK
[1120236.602384]app_data len check OK
[1120236.602386]MQTT decode:Legal Packet!! Success!!
[1120236.602388]==============end==============
[1120236.602390]
[1120236.602392]
[1120236.602394]==============start==============
[1120236.602396]app_data len=12,total_data_len=13
[1120236.602398]Packet for source address:172.16.10.100
[1120236.602400]Packet for destination address:172.16.10.120
[1120236.602402]app_data:2e 7b 00 11 24 26 01 02 00 19 15 37
[1120236.602404]
[1120236.602406]**********MQTT DPI**********
[1120236.602408]Fixed header check FAIL
[1120236.602410]MQTT decode:Illegal Packet!! Failed!!
[1120236.602412]==============end==============
```

图 6-22 固定报头检测记录

通过以上深度包检测功能测试可知，该深度包防护模块能够对 MQTT 协议进行较为全面的深度数据包过滤，为 MQTT 服务器提供了有效的访问控制，提高了服务器的安全性。

6.4.2 深度包检测模块性能分析

除了对深度包检测防护模块进行功能性测试外，还需要进行稳定性测试，即比较 MQTT 直接传输与进行深度包检测这两种情况下的吞吐量和响应时间有没有比较大的波动。本实验使用 IxChariot 吞吐量测试工具对 MQTT 深度包检测模块进行性能测试，该测试工具支持多种协议的吞吐量与响应时间测试，也支持对自定义的应用层数据进行测试。

1、吞吐量测试

未添加深度包检测功能时，该防护模块的吞吐量如图 6-23 所示。

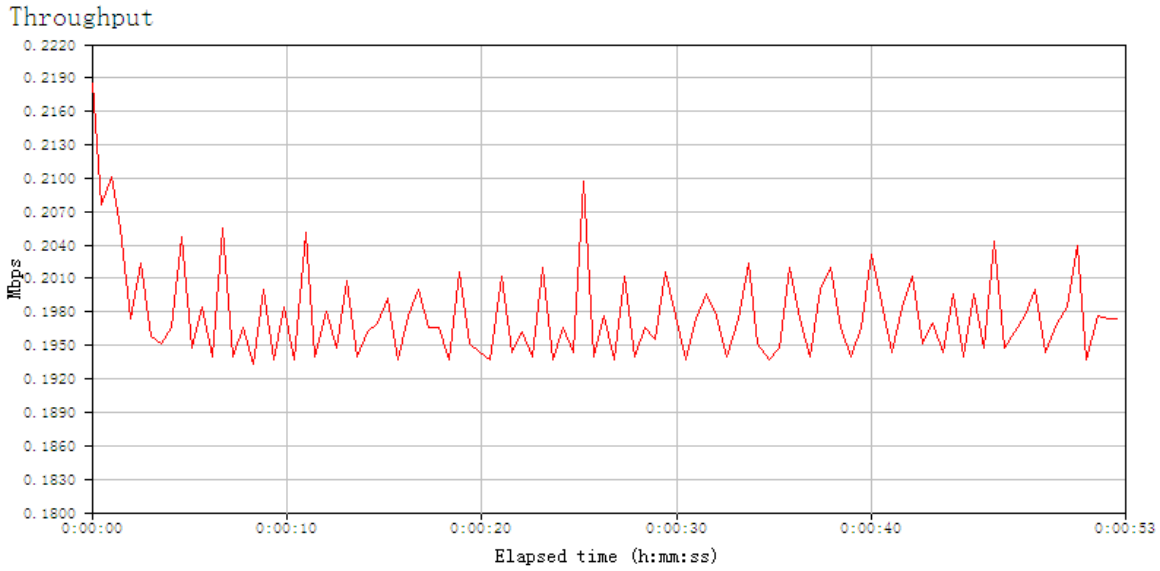

图 6-23 未执行深度包检测时防护模块的吞吐量

防护模块开启深度包检测功能时，其吞吐量如图 6-24 所示。

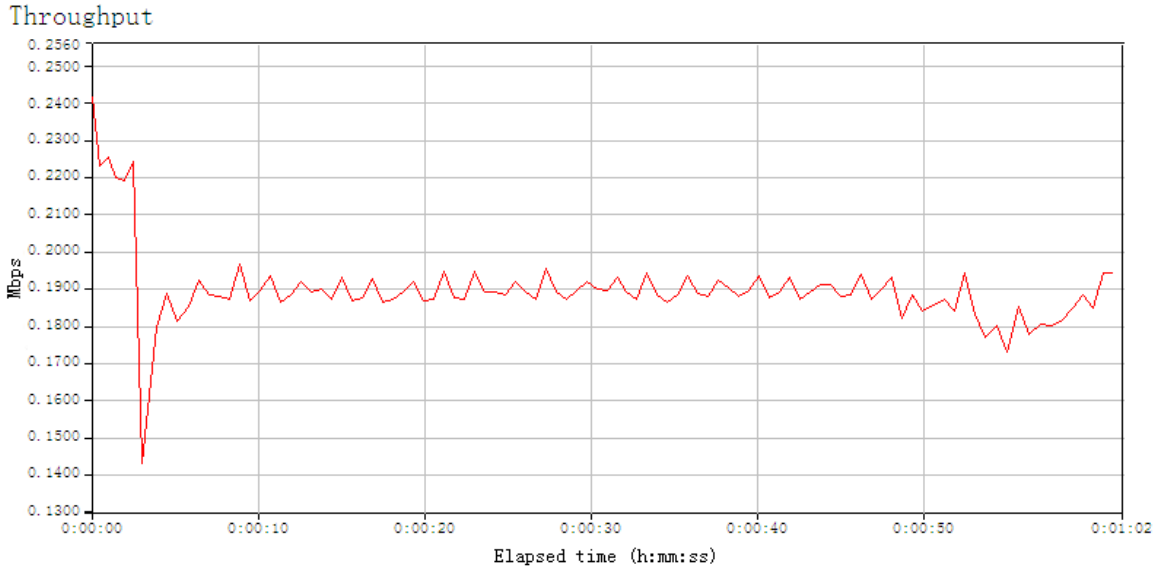

图 6-24 执行深度包检测时防护模块的吞吐量

防护模块不执行深度包检测功能时，其平均吞吐量为 0.198Mbps；防护模块执行 MQTT 深度包检测功能后，其吞吐量在极短时间内会出现下降，之后又回归到稳定状态，平均吞吐量约为 0.190Mbps。对以上结果进行对比发现，相比于不执行深度包检测功能，执行深度包检测功能时防护模块的吞吐量下降了约 4.04%。

2、响应时间测试

未添加深度包检测功能时，该防护模块的响应时间如图 6-25 所示。

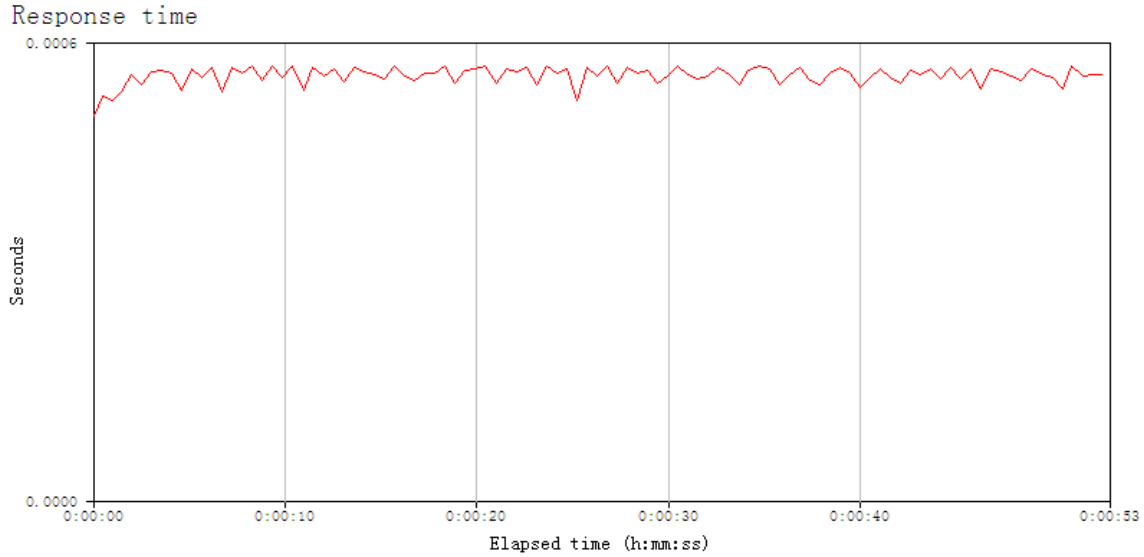

图 6-25 未执行深度包检测时防护模块的响应时间

防护模块开启深度包检测功能时，其响应时间如图 6-26 所示。

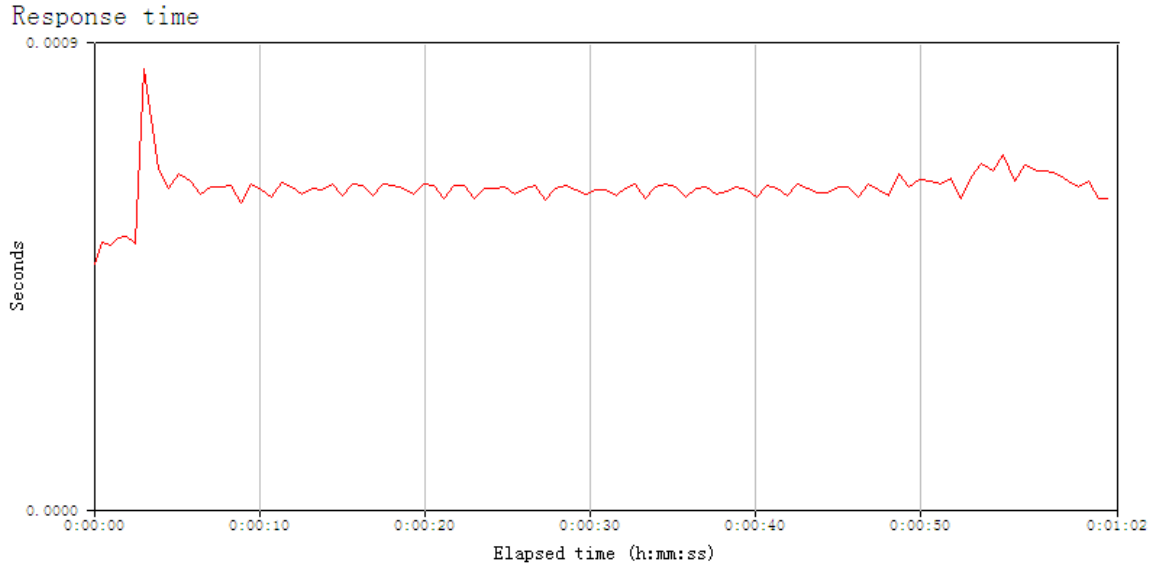

图 6-26 执行深度包检测时防护模块的响应时间

防护模块不执行深度包检测功能时，其平均响应时间为 0.00054Seconds；防护模块执行 MQTT 深度包检测功能后，其吞吐量在极短时间内会出现上升，之后又回归到稳定状态，平均响应时间为 0.00057Seconds。对以上结果进行对比发现，相比于不执行深度包检测功能，执行深度包检测功能时防护模块的响应时间增加了了约 5.56%。

综上所述，防护模块在执行深度包检测功能后，其吞吐量和响应时间会出现短期的波动，之后又会回归到稳定状态，并且吞吐量和响应时间的变化幅度均在

4%到 6%之间，可见深度包检测功能对该防护模块的网络性能影响不大，有着良好的过滤效果。

6.5 基于朴素贝叶斯的异常流量检测模块测试

为了便于描述和分析 MQTT 异常流量检测模块的测试过程，将图 6-2 所示的整体测试环境的网络结构进行简化，得到如图 6-27 所示的 MQTT 协议异常流量的实验环境。

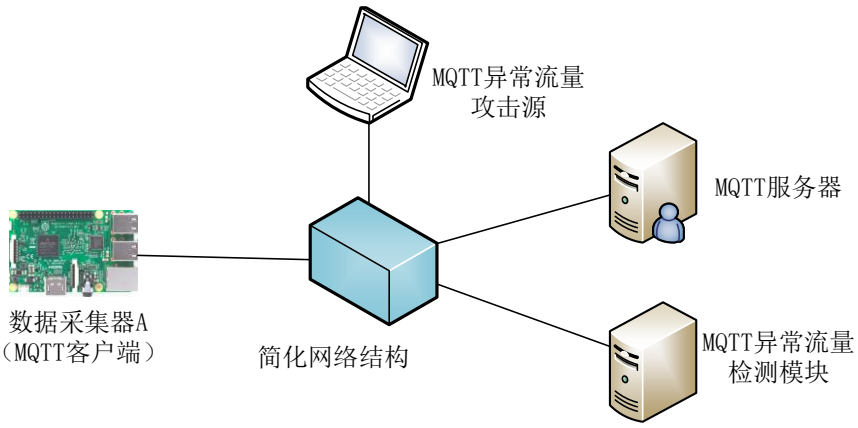

图 6-27 MQTT 异常流量检测测试环境

6.5.1 数据采集及数据分类

本实验中，MQTT 客户端和 MQTT 服务器之间进行正常的通信，MQTT 异常流量攻击源向 MQTT 服务器发送任意构造的 MQTT 控制报文，异常流量检测模块所在的服务器使用 Libpcap（Packet Capture Library）提供的库函数截取交换机中的通信流量，将正常流量和异常流量都复制一份给异常检测模块进行处理，使用这种旁路抓取数据的方法可以避免对 MQTT 客户端和服务器之间的通信造成影响。

通过上述方式，本次实验一共选取了 538 个 MQTT 控制报文样本，其中正常的控制报文样本 282 个，异常的控制报文样本 256 个，并使用这些样本来训练出朴素贝叶斯分类模型。之后再次按照以上数据采集的方式获取了 420 个待检测的控制报文，其中包括 219 个正常控制报文和 201 个异常控制报文，使用训练好的贝叶斯分类模型对以上待检测数据进行测试。

6.5.2 异常流量检测结果分析

评价朴素贝叶斯分类器的质量，主要参考分类正确率、误报率和漏报率三个指标。在本实验的贝叶斯分类检测模型中，分类器的正确率是指分类器正确分类的样本占所有被分类样本的比率，误报率是指分类器将正常样本错误归类为异常

的样本占所有被分类样本的比率，漏报率是指分类器将异常样本错误归类为正常的样本占所有被分类样本的比率。

朴素贝叶斯分类器的质量很大程度上受到特征属性划分的影响。本实验分别在只选取 MQTT 控制报文类型作为特征属性，只选取 MQTT 控制报文类型标志位作为特征属性，以及同时选取 MQTT 控制报文类型和控制报文类型标志位作为特征属性这三种情况下，分别使用 538 个训练样本对朴素贝叶斯分类模型进行训练，再使用这些训练好的模型来对另外 420 个待检测数据进行分类，并对这些分类模型进行分类正确数量、分类误报数量和分类漏报数量统计，得到的结果如表 6-1 所示。

表 6-1 选取不同特征属性的检测结果

特征属性	待检测数据总量	分类正确数量	分类误报数量	分类漏报数量
控制报文类型	420	354	43	23
控制报文类型标志位	420	299	75	46
控制报文类型和控制报文类型标志位	420	387	19	14

根据表 6-1 的统计结果，能够得到选取不同特征属性进行分类器训练时，不同分类器的分类正确率、误报率和漏报率，如图 6-28、6-29、6-30 所示。

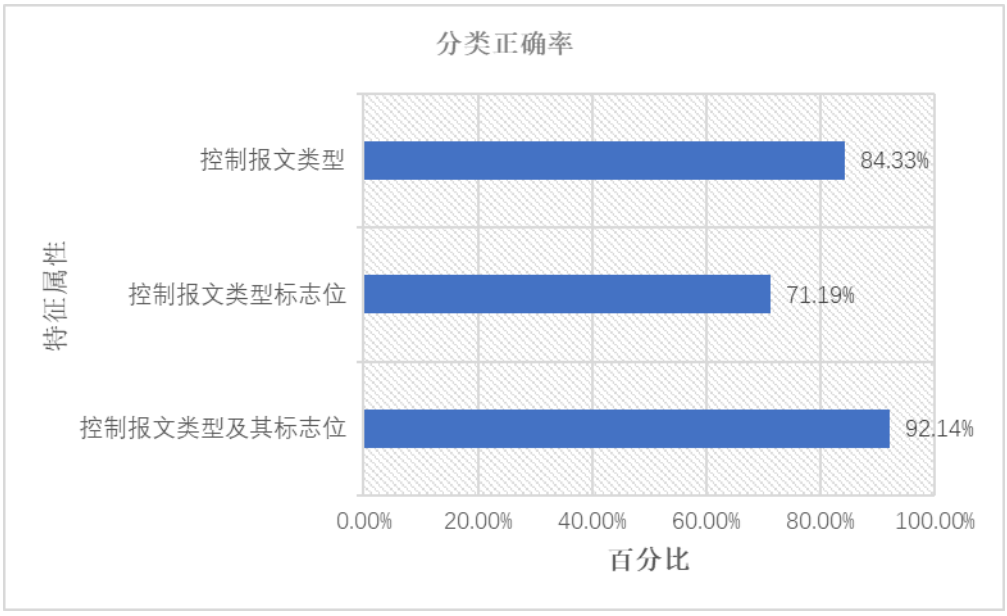

图 6-28 选取不同特征属性时分类器的分类正确率

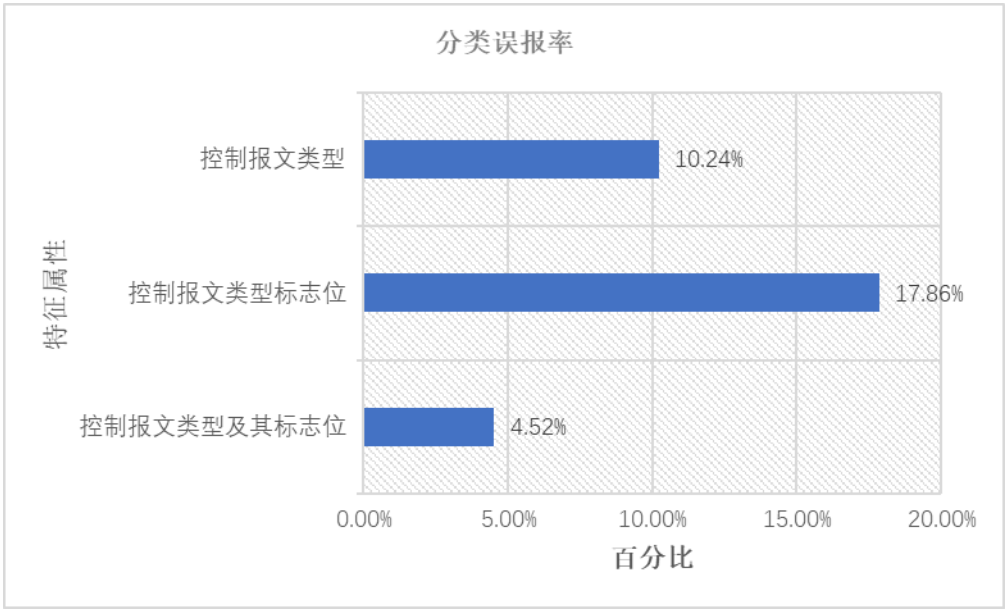

图 6-29 选取不同特征属性时分类器的分类误报率

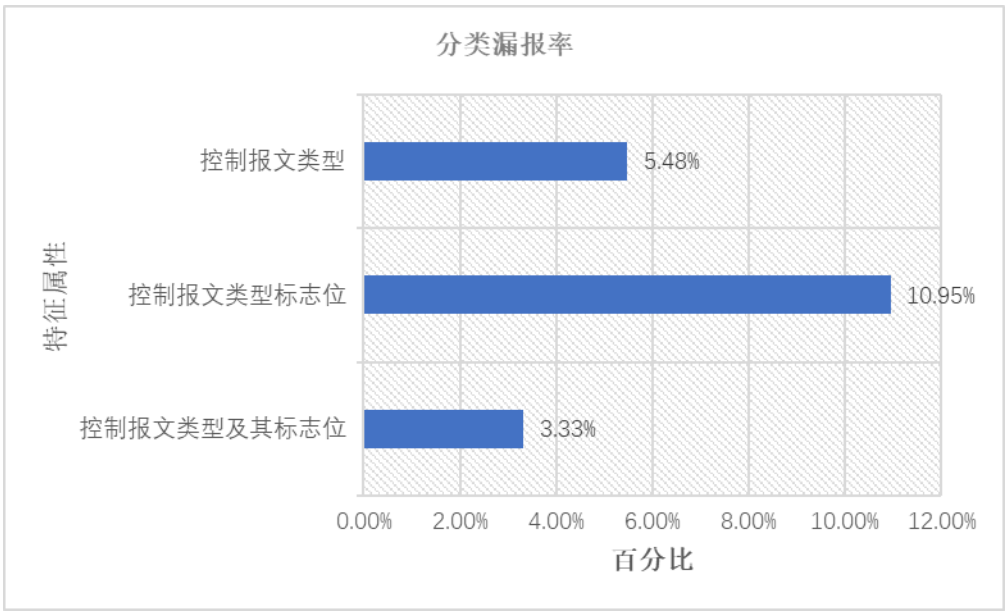

图 6-30 选取不同特征属性时分类器的分类漏报率

根据以上的正确率、误报率和漏报率图形可知：当只选取 MQTT 控制报文类型作为特征属性或只选取控制报文类型标志位作为特征属性，分类正确率不够理想，分别为 84.33%和 71.19%，并且误报率、漏报率和比较高；当选取 MQTT 控制报文类型和控制报文类型标志位作为特征属性时，朴素贝叶斯分类器的分类正确率提高到了 92.14%，并且分类误报率、漏报率也大幅降低。根据以上测试结果，采用该基于朴素贝叶斯的异常流量检测模型，能够对 MQTT 通信的数据流量进行有效地实时检测防护。

6.6 本章小结

本章主要对工业物联网系统的安全防护模块进行了测试和分析，主要对物联网僵尸网络病毒攻击的防护方法、MQTT 协议通信加密和身份认证、深度包检测模块以及基于朴素贝叶斯的异常流量检测模块，分别进行了测试和分析。从各部分的测试结果可知，针对数据采集层中物联网终端设备的端口扫描检测和暴力字典破解防护方法能够有效防御物联网僵尸网络病毒的攻击；基于 TLS 的 MQTT 协议通信加密措施能够防范数据窃取，基于 X509 数字证书的身份认证措施能够防御中间人攻击；MQTT 协议深度包检测模块能够有效检测出不符合协议规范的数据包；基于朴素贝叶斯的 MQTT 协议异常流量检测模块能够有效检测出高级持续性的欺骗性数据包攻击。因此，本文所设计的工业物联网安全防御体系有着良好的防护效果，能够快速的检测到所遭受的攻击并做出防护反应，保障工业物联网系统的网络安全。

第七章 总结及展望

7.1 全文总结

随着工业物联网技术在各个领域的规范应用，其面临的安全威胁也与日俱增，信息安全成为工业物联网发展过程中需要面临的重大挑战。本文针对工业物联网系统进行信息安全防护技术研究，首先对其基本体系结构和功能进行了解，并详细分析了工业物联网系统中存在的安全威胁；然后根据这些安全威胁，结合系统可能遭受的攻击方式，设计了适用于工业物联网系统的安全防护框架，该安全框架包含数据采集层安全、数据传输层安全以及数据处理层安全这三个层次，并对各个安全层次设计和实现了相应的安全防护策略。数据采集层安全主要针对 Mirai 等僵尸网络病毒攻击的攻击原理，设计了对应的端口扫描检测及暴力字典破解防护方法；数据传输层安全主要针对信息泄露和中间人攻击这两种安全威胁，设计了基于 TLS 的 MQTT 协议通信加密和基于 X509 数字证书的身份认证措施；数据处理层主要针对以伪造数据包攻击为代表的入侵方式，设计了 MQTT 协议深度包检测模块以及基于朴素贝叶斯的 MQTT 异常流量检测模块。最后，对上述各个安全防护方法进行了测试，各方法均能够有效地防御系统遭受的网络攻击，保障工业物联网系统安全稳定地运行。

7.2 后续工作展望

本文设计的工业物联网系统安全防护方法，在一定程度上能够防御各种网络攻击，但是在某些方面依然有待改进和进一步完善，具体包括以下几点：

首先，在对 Mirai 等僵尸网络病毒攻击的防护策略中，本文只研究了如何防御这一类病毒对物联网设备的感染，对于已经受到僵尸网络病毒感染的物联网终端设备，缺乏有效清除受到感染设备中的恶意程序的方法。因此，需要进一步研究如何修复受到僵尸网络病毒感染的物联网设备。

其次，对于 MQTT 协议深度包检测方法，在访问控制规则中只能配置固定报头中的控制报文类型和控制报文类型标志位，无法检测可变报头和有效载荷中的内容，还需要对 MQTT 协议进行更深入的分析，设计出更多的访问控制策略。

最后，在基于朴素贝叶斯的 MQTT 异常流量检测方法中，特征属性的选取不应该局限在控制报文类型和控制报文类型标志位这两项，可以对特征属性的划分进行更多的选取，进一步提高分类正确率。

致谢

硕士研究生生涯即将结束，首先要感谢我的导师郑宏教授，郑老师学识渊博，教学严谨有耐心，在工作上兢兢业业，使我受益良多，感谢郑老师在科研、生活等各方面对我的关心和帮助。然后要感谢在科研过程中给予我直接帮助的辛晓帅老师，在诸多难题面前，辛老师总能提出非常好的解决方案。同时还要感谢邹见效老师、何建老师、于力老师、李立英老师、彭超老师，您们对工作认真负责的态度和对科研的热情深深地影响了我，感谢您们对我学习和生活中的帮助。

感谢教研室的邓力师兄、刘灿成师兄在科研、项目上给予我的帮助和指导，感谢郭娅雯同学、杜松同学、刘仁杰同学、朱晨同学、于仁飞师弟、李财师弟等等，感谢你们对我的帮助和支持。

最最要感谢的是我的家人，他们总是默默地支持着我，在我面临压力时，陪伴在我身边，给我信心和鼓励。

最后衷心感谢各位评委老师，感谢您们付出时间和精力对我的毕业论文进行审阅，在此致以崇高的敬意。

研究生的三年生活马上要结束了，但是人生的又一个新起点才刚刚开始，在以后的工作和生活中，我会继续努力，不忘初心。

参考文献

[1] 刘克一. 物联网概念的基本定位[J]. 数字技术与应用, 2015(04):221.

[2] 应桂芬. 物联网的概念、构建和关键技术分析[J]. 电子技术与软件工程, 2015(18):14-15.

[3] 刘博. 物联网及其核心技术解析[J]. 电子世界, 2016(16):12.

[4] 秦志远. 物联网技术及其发展前景展望[J]. 山东工业技术, 2016(21):154.

[5] 王玉峰, 戴伟. 工业物联网:未来制造业生态入场券[J]. 中国工业评论, 2017(04):28-34.

[6] 康世龙, 杜中一, 雷咏梅, 等. 工业物联网研究概述[J]. 物联网技术, 2013(06):80-82.

[7] Kiel D, Arnold C, Voigt K. The influence of the Industrial Internet of Things on business models of established manufacturing companies – A business level perspective[J]. Technovation, 2017,68.

[8] Popescu G H. The economic value of the industrial internet of things[J]. 2015.

[9] TANG G. Research on the Security of Internet of Things Based on Cloud Computing, 中国上海, 2015[C].

[10] 毕然, 李小刚, 姜建. 物联网安全事件案例和问题分析[J]. 电信网技术, 2014(12):10-13.

[11] 舒文琼. 美国断网事件敲响物联网安全警钟[J]. 通信世界, 2016(30):58.

[12] 李柏松, 常安琪, 张家兴. 物联网僵尸网络严重威胁网络基础设施安全——对Dyn公司遭僵尸网络攻击的分析[J]. 信息安全研究, 2016(11):1042-1048.

[13] Liu J. Security and Privacy Problems and Countermeasures of Internet of Things Applications: Security and Privacy Problems and Countermeasures of Internet of Things Applications, 中国重庆, 2017[C].

[14] Ge M, Hong J B, Guttmann W, et al. A framework for automating security analysis of the internet of things[J]. Journal of Network and Computer Applications, 2017,83.

[15] Jan M A, Nanda P, He X, et al. A Robust Authentication Scheme for Observing Resources in the Internet of Things Environment[J]. 2014.

[16] 袁琦. 我国物联网安全发展现状和建议[J]. 现代电信科技, 2014(10):36-39.

[17] 周丽莎, 孔勇平, 陆钢. 物联网安全政策解读及技术标准综述[J]. 广东通信技术, 2017(12):39-41.

[18] 杨悦梅, 宋执环. 工业物联网安全及防护技术研究[J]. 物联网技术, 2015(03):64-66.

[19] 朱艳. 面向物联网的身份认证和访问控制的研究[D]. 南京邮电大学, 2014.

[20] 郭莉, 严波, 沈延. 物联网安全系统架构研究[J]. 信息安全与通信保密, 2010(12):73-75.

[21] 王乐燕, 袁莉青. 物联网技术及其发展前景[J]. 内蒙古科技与经济, 2013(12):66-67.

[22] 刘雪，王文斌．物联网云计算平台测评体系的建立[J]．计算机光盘软件与应用，2014(01):35-36.

[23] Joy Weiss, Ross Yu. 面向工业物联网的无线传感器网络[J]. 电源世界, 2016(08):50-52.

[24] Hisashi S, Toru I, Hisanori H. Future IIOT in Process Automation Latest trends of standardization in industrial automation,IEC/TC65, 中国浙江杭州, 2015[C].

[25] Shi Z, Liao K, Yin S, et al. Design and implementation of the mobile internet of things based on td-scdma network: IEEE International Conference on Information Theory and Information Security, 2011[C].

[26] 黄明. 轻量级工业物联网控制网络协议研究[J]. 信息技术与标准化, 2017(09):59-62.

[27] 韩丽，李孟良，卓兰，等．《工业物联网白皮书(2017版)》解读[J]．信息技术与标准化，2017(12):30-34.

[28] 高爱玲. 物联网中安全通信协议的形式化分析[J]. 信息通信, 2015(06):210.

[29]马跃，孙翱，贾军营，等. MQTT协议在移动互联网即时通信中的应用[J]．计算机系统应用，2016(03):170-176.

[30] 龚永罡，付俊英，汪昕宇，等．MQTT协议在物联网中的应用研究[J]．电脑与电信，2017(11):89-91.

[31] 王培元，杜玉胜．一种基于MQTT协议的物联网智能监控系统[J]．信息技术与信息化，2018(01):107-110.

[32] 张巍，高汝熹，车春鹂．工业物联网技术链、产业链、价值链互动机理研究[J]．上海管理科学, 2010(06):51-57.

[33] 张猛．工业物联网安全风险分析及对策研究[J]．中国工业评论, 2017(04):42-50.

[34] Wang S P, Computer D O. Review on security research of the Internet of things perception layer[J]. Transducer & Microsystem Technologies, 2015.

[35] Lin-Mei D A. Research on Information Security Technology of the Internet of Things at the Perceived Layers of RFID and WSN[J]. Journal of Nanjing Institute of Industry Technology, 2014.

[36] Said N B, Biondi F, Bontchev V, et al. Detection of Mirai by Syntactic and Semantic Analysis[J]. 2017.

[37] 张俊松．物联网环境下的安全与隐私保护关键问题研究[D]．北京邮电大学, 2014.

[38] Niu C C, Zou K C, Yang Y L O, et al. Security and Privacy Issues of the Internet of Things[J]. Applied Mechanics and Materials, 2013(416):1429-1433.

[39] Lu X, Qu Z, Li Q, et al. Privacy information security classification for internet of things based on internet data[J]. International Journal of Distributed Sensor Networks, 2015,2015(6):23.

[40] Yang G, Jian X U, Chen W, et al. Security Characteristic and Technology in the Internet of Things[J]. Journal of Nanjing University of Posts & Telecommunications, 2010,30(4):20-29.

[41] Yoon S, Kim J. Remote security management server for IoT devices: International Conference on Information and Communication Technology Convergence, 2017[C].

[42] Perrone G, Vecchio M, Pecori R, et al. The Day After Mirai: A Survey on MQTT Security Solutions After the Largest Cyber-attack Carried Out through an Army of IoT Devices: International Conference on Internet of Things, Big Data and Security, 2017[C].

[43] Xin X, Liu C, Wang B. Real-Time Intrusion Detection Method Based on Bidirectional Access of Modbus/TCP Protocol: International Conference on Cryptography, Security and Privacy, 2017[C].

[44] 路琪, 黄芝平, 鲁佳琪. 基于深度包检测的防火墙系统设计[J]. 计算机科学, 2017(S2):334-337.

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

[46] 梁春丽. 新型IoT僵尸网络来袭[J]. 金融科技时代, 2017(11):95.

[47] 陈亚亮, 戴沁芸, 吴海燕, 等. Mirai僵尸网络恶意程序分析和监测数据研究[J]. 网络与信息安全学报, 2017(08):39-47.

[48] Korczynski M, Janowski L, Duda A. An Accurate Sampling Scheme for Detecting SYN Flooding Attacks and Portscans: IEEE International Conference on Communications, 2011[C].

[49] Patel S K, Sonker A. Internet Protocol Identification Number Based Ideal Stealth Port Scan Detection Using Snort: International Conference on Computational Intelligence and Communication Networks, 2016[C].

[50] Sperotto A, Sadre R, Boer P T, et al. Hidden Markov Model Modeling of SSH Brute-Force Attacks: Integrated Management of Systems, Services, Processes and People in It, Ifip/ieee International Workshop on Distributed Systems: Operations and Management, Dsom 2009, Venice, Italy, October 27-28, 2009. Proceedings, 2009[C].

[51] Chen Q A, Osterweil E, Thomas M, et al. MitM Attack by Name Collision: Cause Analysis and Vulnerability Assessment in the New gTLD Era: Security and Privacy, 2016[C].

硕士期间主要研究成果

个人简历

2011 年 9 月-2015 年 6 月，电子科技大学自动化工程学院，自动化专业，本科
2015 年 9 月-至今，电子科技大学自动化工程学院，控制工程专业，硕士研究生

主要科研工作

[1] 工业控制系统信息安全防护技术研究，2015年9月~2017年9月
[2] 工业物联网数据采集系统研究，2016年12月~2017年5月

已发表的论文

[1] Xiaoshuai Xin, Cancheng Liu, Bin Wang. Real-time intrusion detection method based on bidirectional access of ModbusTCP protocol. 2017 International Conference on Cryptography, Security and Privacy.

已申请的专利

[1] 邹见效，王斌，辛晓帅. 一种 IEC104 通信访问控制方法[P]. 四川：CN 106982219 A，2017.07.25
[2] 郑宏，王斌，邹见效. 一种基于朴素贝叶斯的MQTT异常流量检测方法. 201810087318.6，2018.01.30