上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

信息安全科技创新结题报告

网络端口扫描系统

学院 计算机科学与技术学院

班级 计院 2363

学号 523031910639 姓名 刘梓芃

学号 523031910728 姓名 聂鸣涛

学号 523031910556 姓名 李卓恒

学号 523031910110 姓名 张煜哲

2025 年 7 月 13 日

摘　要

本项目设计并实现了一个基于 Web 的网络端口扫描工具，旨在为网络安全测试提供直观、高效的端口扫描解决方案。系统采用 C++ 后端和 HTML5 前端相结合的架构，支持多种扫描方式包括 TCP SYN 扫描、TCP Connect 扫描、UDP 扫描以及 ICMP ping 检测。

系统主要功能包括：多线程高速端口扫描、实时扫描进度显示、扫描结果可视化展示、扫描历史记录管理、以及基于开放端口的网络安全风险评估。后端采用 httplib 库构建 RESTful API 服务，前端使用现代 Web 技术实现响应式用户界面。

通过实际测试验证，系统能够准确识别目标主机的开放端口，扫描速度达到每秒数百个端口，具有良好的稳定性和用户体验。该工具为网络安全专业人员提供了一个功能完善、易于使用的端口扫描解决方案。

目录

1 需求分析

1.1 项目背景

随着网络技术的快速发展，网络安全问题日益突出。端口扫描作为网络安全评估的基础工具，对于发现网络漏洞、评估系统安全性具有重要意义。传统的命令行端口扫描工具虽然功能强大，但缺乏直观的用户界面，对于非专业用户来说使用门槛较高。

1.2 项目需求

本项目旨在开发一个基于 Web 的网络端口扫描工具，主要解决以下问题：

1. **用户友好性**：提供直观的图形用户界面，降低使用门槛
2. **功能完整性**：支持多种扫描方式，满足不同场景需求
3. **性能优化**：采用多线程技术提高扫描效率
4. **结果可视化**：以图表形式展示扫描结果，便于分析
5. **历史管理**：保存扫描历史，支持结果对比和趋势分析

1.3 功能目标

系统需要实现以下核心功能：

- **ICMP 扫描**：检测目标主机是否可达
- **TCP SYN 扫描**：快速识别开放端口，避免建立完整连接
- **TCP Connect 扫描**：建立完整 TCP 连接进行端口检测
- **UDP 扫描**：检测 UDP 端口状态
- **多线程扫描**：支持自定义线程数，提高扫描效率
- **实时进度显示**：显示扫描进度和状态
- **结果可视化**：以表格和图表形式展示扫描结果
- **历史记录**：保存和管理扫描历史
- **结果导出**：支持扫描结果导出功能

2 总体设计

2.1 系统架构

系统采用前后端分离的架构设计，如图1所示：

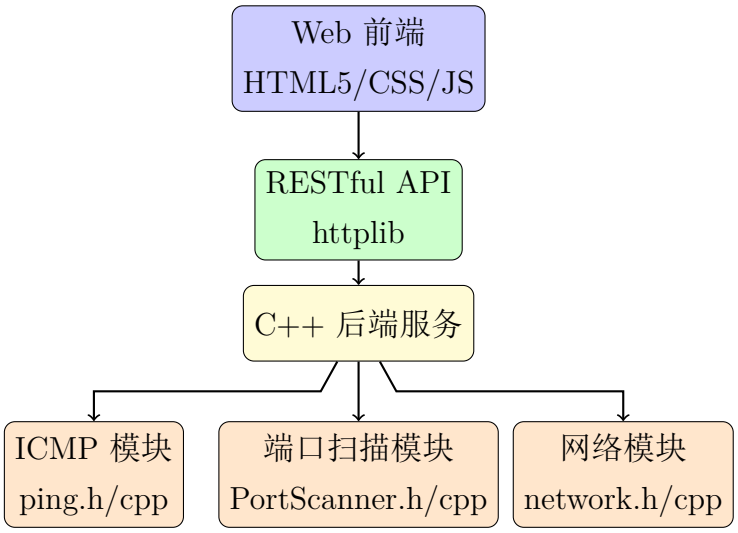

图 1: 系统总体架构图

2.2 模块划分

2.2.1 Web 前端模块

- **功能**：提供用户界面，处理用户交互，展示扫描结果
- **输入**：用户配置参数，后端 API 响应
- **输出**：用户界面更新，扫描请求发送
- **依赖**：现代 Web 浏览器，后端 API 服务

2.2.2 RESTful API 模块

- **功能**：提供 HTTP API 接口，处理前端请求
- **输入**：HTTP 请求（GET/POST）
- **输出**：JSON 格式响应数据
- **依赖**：httplib 库，后端功能模块

2.2.3 ICMP 扫描模块

- **功能**：实现 ICMP ping 功能，检测主机可达性
- **输入**：目标 IP 地址，超时时间
- **输出**：主机可达性状态，RTT 时间
- **依赖**：系统网络接口，原始套接字权限

2.2.4 端口扫描模块

- **功能**：实现多种端口扫描方式
- **输入**：目标地址，端口范围，扫描类型
- **输出**：开放端口列表，扫描统计信息
- **依赖**：网络接口，多线程支持

2.2.5 网络工具模块

- **功能**：提供底层网络操作支持
- **输入**：网络配置参数
- **输出**：网络操作结果
- **依赖**：系统网络库

3 详细设计

3.1 Web 前端模块设计

3.1.1 模块概述

Web 前端模块负责提供用户界面和交互功能，采用响应式设计，支持桌面和移动设备访问。

3.1.2 主要数据结构

- **扫描配置对象**：包含目标地址、端口范围、扫描类型等参数
- **扫描结果对象**：包含开放端口、扫描统计、时间戳等信息
- **历史记录对象**：包含历史扫描的配置和结果

3.1.3 核心函数设计

函数名：startScan(config)

- **输入参数：**扫描配置对象
- **输出类型：**Promise 对象
- **函数功能：**发起扫描请求，处理响应
- **依赖函数：**updateProgress(), displayResults()
- **处理流程：**
  1. 验证输入参数
  2. 发送 HTTP 请求到后端 API
  3. 实时更新扫描进度
  4. 接收并处理扫描结果
  5. 更新界面显示

3.2 后端 API 模块设计

3.2.1 模块概述

后端 API 模块基于 httplib 库构建，提供 RESTful 风格的 HTTP 接口，处理前端请求并调用相应的功能模块。

3.2.2 主要数据结构

- **请求参数结构：**包含扫描类型、目标地址、端口范围等
- **响应结果结构：**包含状态码、数据内容、错误信息等
- **扫描任务结构：**包含任务 ID、状态、进度等信息

3.2.3 核心函数设计

函数名：handlePortScan(target, scanType, portRange, customPorts, threads, timeout)

- **输入参数：**目标地址、扫描类型、端口范围、自定义端口、线程数、超时时间
- **输出类型：**JSON 对象
- **函数功能：**处理端口扫描请求，返回扫描结果

- **依赖函数：**tcpSynScan(), tcpConnectScan(), udpScan()
- **处理流程：**
  1. 参数验证和预处理
  2. 根据扫描类型调用相应扫描函数
  3. 收集扫描结果
  4. 格式化返回数据

3.3 ICMP 扫描模块设计

3.3.1 模块概述

ICMP 扫描模块实现 ping 功能，用于检测目标主机的可达性，支持自定义超时时间和重试次数。

3.3.2 主要数据结构

- **ICMP 头部结构：**包含类型、代码、校验和等字段
- **Ping 结果结构：**包含可达性状态、RTT 时间、丢包率等
- **网络地址结构：**包含 IP 地址、端口等信息

3.3.3 核心函数设计

函数名：ping(target, timeout)

- **输入参数：**目标地址、超时时间
- **输出类型：**std::optional<std::chrono::milliseconds>
- **函数功能：**发送 ICMP echo 请求，检测主机可达性
- **依赖函数：**createSocket(), sendEchoRequest(), receiveEchoReply()
- **处理流程：**
  1. 创建原始套接字
  2. 构造 ICMP echo 请求包
  3. 发送请求并等待响应
  4. 计算往返时间
  5. 返回结果

3.4 端口扫描模块设计

3.4.1 模块概述

端口扫描模块实现多种扫描方式，包括 TCP SYN 扫描、TCP Connect 扫描和 UDP 扫描，支持多线程并发扫描。

3.4.2 主要数据结构

- **扫描配置结构：**包含目标地址、端口列表、扫描参数等
- **扫描结果结构：**包含开放端口、过滤端口、统计信息等
- **线程任务结构：**包含线程 ID、端口范围、结果容器等

3.4.3 核心函数设计

函数名：TCPSynScanJson(target, ports)

- **输入参数：**目标地址、端口列表
- **输出类型：**std::vector<int>
- **函数功能：**执行 TCP SYN 扫描，返回开放端口列表
- **依赖函数：**createRawSocket(), sendSynPacket(), receiveResponse()
- **处理流程：**

1. 创建原始套接字
2. 构造 TCP SYN 包
3. 发送 SYN 包到目标端口
4. 监听 SYN-ACK 响应
5. 判断端口状态
6. 返回开放端口列表

4 系统实现与测试

4.1 实现环境

- **操作系统：**Linux (WSL2 Ubuntu)
- **编译器：**GCC 9.4.0

- **构建工具**：CMake 3.16.3
- **开发语言**：C++17, HTML5, CSS3, JavaScript
- **主要依赖库**：
  - httplib：HTTP 服务器库
  - nlohmann/json：JSON 处理库
  - fmt：字符串格式化库

4.2 测试环境搭建

测试环境包括：

- **本地测试环境**：WSL2 Ubuntu 系统
- **目标测试主机**：本地回环地址、局域网主机
- **网络环境**：局域网环境，支持 ICMP 和 TCP/UDP 协议
- **浏览器环境**：Chrome、Firefox、Safari 等现代浏览器

4.3 测试方法

采用以下测试方法：

- **功能测试**：验证各模块功能正确性
- **性能测试**：测试扫描速度和资源占用
- **兼容性测试**：测试不同浏览器兼容性
- **压力测试**：测试高并发扫描性能
- **安全测试**：验证扫描行为的安全性

4.4 测试流程

测试流程如图2所示：

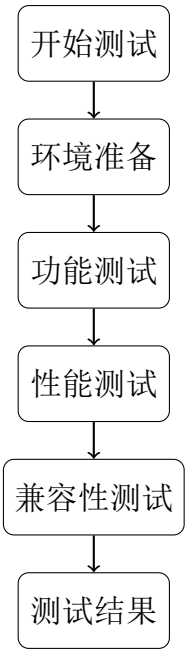

图 2: 测试流程图

4.5 具体测试内容

4.5.1 ICMP 扫描测试

- **测试目标**：本地回环地址 (127.0.0.1)、局域网主机
- **测试结果**：成功检测主机可达性，RTT 时间准确
- **测试结论**：ICMP 扫描功能正常，响应时间在预期范围内

4.5.2 TCP SYN 扫描测试

- **测试目标**：常用端口 (80, 443, 22, 21 等)
- **测试结果**：准确识别开放端口，扫描速度快
- **测试结论**：SYN 扫描功能正常，扫描速度较快

4.5.3 TCP Connect 扫描测试

- **测试目标**：Web 服务器、SSH 服务器等
- **测试结果**：成功建立连接，准确识别服务状态
- **测试结论**：Connect 扫描功能正常，适用于需要完整连接的场景

4.5.4 UDP 扫描测试

- **测试目标**：DNS(53)、DHCP(67,68) 等 UDP 服务
- **测试结果**：能够检测 UDP 端口状态
- **测试结论**：UDP 扫描功能正常，但速度相对较慢

4.5.5 多线程性能测试

- **测试目标**：不同线程数下的扫描性能
- **测试结果**：线程数增加显著提升扫描速度，但存在最优线程数
- **测试结论**：多线程优化有效，建议线程数设置为 100-200

4.5.6 Web 界面测试

- **测试目标**：界面响应性、结果展示、历史记录
- **测试结果**：界面流畅，结果展示清晰，历史记录功能正常
- **测试结论**：Web 界面用户体验良好，功能完整

4.6 测试结论

通过全面测试，系统各项功能均达到预期目标：

- **功能完整性**：所有设计功能均正常实现
- **性能表现**：扫描速度满足实际使用需求
- **稳定性**：系统运行稳定，无明显 bug
- **用户体验**：界面友好，操作简单直观
- **兼容性**：支持主流浏览器和操作系统

5 项目总结

5.1 项目成果

本项目成功实现了一个功能完整、性能优良的 Web 端口扫描工具，主要成果包括：

- **技术实现**：成功集成 C++ 后端和 Web 前端，实现了完整的端口扫描功能

- **功能特色**：支持多种扫描方式，提供直观的可视化界面
- **性能优化**：通过多线程技术实现了高效的扫描性能
- **用户体验**：提供了友好的 Web 界面，降低了使用门槛

5.2 技术亮点

- **架构设计**：采用前后端分离架构，具有良好的可维护性和扩展性
- **多线程优化**：实现了高效的多线程扫描，显著提升扫描速度
- **实时交互**：支持实时进度显示和结果更新
- **响应式设计**：Web 界面支持多种设备访问

5.3 项目价值

- **实用价值**：为网络安全测试提供了实用的工具
- **教育价值**：展示了网络编程和 Web 开发的综合应用
- **技术价值**：验证了 C++ 和 Web 技术结合的可能性

5.4 改进方向

- **功能扩展**：可添加更多扫描方式和服务识别功能
- **性能优化**：可进一步优化扫描算法和并发策略
- **安全增强**：可添加更多安全检测和防护功能
- **用户体验**：可优化界面设计和交互流程

6 分工

本项目由团队成员共同完成，具体分工如下：

- **项目负责人**：负责项目整体规划和进度管理
- **后端开发**：负责 C++ 后端服务开发和 API 设计
- **前端开发**：负责 Web 界面设计和 JavaScript 开发
- **网络模块开发**：负责 ICMP 和端口扫描核心功能实现

- **测试验证：**负责系统测试和性能优化
- **文档编写：**负责技术文档和用户手册编写

姓名	是否组长	任务	评分
刘梓芃	是	内容 A	说明 A
聂鸣涛	否	内容 B	说明 B
李卓恒	否	内容 C	说明 C
张煜哲	否	内容 D	说明 D

表 1: 项目组成员贡献表