五 邑 大 学

毕业设计说明书

**毕业设计题目：以太网卡监管系统开发**

**学院（部） 智能制造学部**

**专 业 通信工程**

**学 号 3117000096**

**学生姓名 张学勤**

**指导教师 张先勇**

**完成日期 2019年5月20日**

# 摘 要

如今，网络俨然已是生活的必需品，网络高级人才却依然是稀缺品。由于计算机网络知识的零散化和抽象性，使得教学重心自然以理论为重心，缺乏实践探索，逐渐成为为必然。导致学生知识碎片化，知识调度能力薄弱，后续学习力不从心。

因此网络理论与实践结合刻不容缓！从Wireshark中获得灵感，从而开发出此款以太网卡监控系统为，为上述问题提供了解决方案。

本系统核心着力为计算机网络求学者完善知识框架提供实践平台，努力成为网络工程师局域网信息排查和网络排错的得力助手。本系统具有以下特点，GUI界面友好灵活上手门槛几乎为零；网络嗅探功能全面完善，保证功能齐全满足日常教学办公需要；数据归档复用性强，数据包文件保存格式各平台通用，数据导出类型丰富，图表生成精美。

测试结果表明，常规网络场景中该系统能流畅完成各式任务，在高速流量网络场景中，除个别数据包丢失外，其他功能一切正常。

本系统立足实践，务求为初学者提供网络知识实践平台，为网络工程师提供开箱即用的集网络分析、排错和统计一体化工具。

**关键字**： 协议分析，局域网监听，嗅探技术

# Abstract

Today, the Internet is already a necessity for life, however the network of senior engineer is still scarce. Due to the fragmentation and abstraction of computer network knowledge, the teaching is naturally focused on theory, lacking practical exploration, and gradually becoming inevitable. Lead to the fragmentation of students' knowledge, weak knowledge scheduling, and lack of follow-up learning.

Therefore, the combination of network theory and practice is an urgent task! Inspired by Wireshark, the Ethernet card monitoring system was developed to provide a solution to the above problems.

The core of the system is to provide a practical platform for computer network scholars to improve the knowledge framework, and strive to become the network engineer's LAN information troubleshooting and network troubleshooting. The system has the following characteristics: the GUI interface is friendly and flexible, and the threshold is almost zero; the network sniffing function is comprehensive and perfect, and the function is complete to meet the daily teaching office needs; the data archiving reusability is strong, the data packet file saving format is common to all platforms, and the data is exported rich in types and beautifully generated charts.

The test results show that the system can smoothly complete various tasks in the conventional network scenario. In the high-speed traffic network scenario, except for the loss of individual data packets, all other functions are normal.

The system is based on practice, in order to provide beginners with a network knowledge practice platform, providing network engineers with out-of-the-box tools for network analysis, troubleshooting and statistical integration.

**Keywords**: protocol analysis, LAN monitoring, sniffing technology

# 目 录

[摘 要 I](#_Toc8726284)

[Abstract II](#_Toc8726285)

[第1章 绪论 1](#_Toc8726287)

[1.1 课题背景 1](#_Toc8726288)

[1.2 课题目的和意义 2](#_Toc8726289)

[1.3 国内外文献综述 3](#_Toc8726290)

[第2章 相关理论与技术 7](#_Toc8726291)

[2.1 计算机网络知识 7](#_Toc8726292)

[2.1.1 网络层次划分及其功能 7](#_Toc8726293)

[2.1.2 数据包封装与解封 8](#_Toc8726294)

[2.1.3 主机间通信过程 9](#_Toc8726295)

[2.2 地址与端口 10](#_Toc8726296)

[2.2.1 MAC地址 10](#_Toc8726297)

[2.2.2 IP地址 11](#_Toc8726298)

[2.2.3 端口号 14](#_Toc8726299)

[2.3开发涉及知识 15](#_Toc8726300)

[2.3.1 嗅探技术 15](#_Toc8726301)

[2.3.2 ARP攻击技术 15](#_Toc8726302)

[第3章 需求分析 17](#_Toc8726303)

[3.1 需求获取 17](#_Toc8726304)

[3.2 分析需求 18](#_Toc8726305)

[3.2.1 表面需求 18](#_Toc8726306)

[3.2.2 本质需求 18](#_Toc8726307)

[3.2.3 产品需求 19](#_Toc8726308)

[3.2.4 需求排序 19](#_Toc8726309)

[第4章 系统总体设计 21](#_Toc8726310)

[4.1 划分子系统 21](#_Toc8726311)

[4.2 子系统模块结构 22](#_Toc8726312)

[4.3 软件架构图 24](#_Toc8726313)

[4.4 软件流程图 25](#_Toc8726314)

[第5章 系统详细设计与实现 27](#_Toc8726315)

[5.1 界面设计 27](#_Toc8726316)

[5.2界面实现 28](#_Toc8726317)

[5.3模块详细设计 31](#_Toc8726318)

[5.4模块实现 34](#_Toc8726319)

[第6章 性能分析 43](#_Toc8726320)

[6.1 测试环境 43](#_Toc8726321)

[6.1.1测试概述 43](#_Toc8726322)

[6.1.2主机环境 43](#_Toc8726323)

[6.1.3宿主机信息 45](#_Toc8726324)

[6.2软件测试 46](#_Toc8726325)

[总结与展望 49](#_Toc8726326)

[致 谢 51](#_Toc8726327)

[参考文献 52](#_Toc8726328)

# 第1章 绪论

## 1.1 课题背景

近年来，我国的互联网发展呈现出一片欣欣向荣的景象。据《中国互联网发展报告（2019年2月）》指出，截至2018年12月，国内网民规模为8.29亿，互联网的普及率达59.6%，需要特别指出，这是自2013以来连续五年的持续的增长，且保持百分之二以上的稳定攀升[1]。取得如此骄人的成绩，首先是互联网基础设施不断完善，其次是网络战略的不断深化，互联网服务渗透，各行业在互联网下大放异彩。

今天联网的发展让我们的生活产生了前所未有的变化。在计算机和网络通信高速发展，以及政府大力倡导的“互联网+”战略的双重“攻势”下，居民生活方式产生了天翻地覆的巨变。网络购物的出现使得人们从“出门——挑选——结账——运输”繁琐的购物模式，简化为“挑选——结账”，极大简化了的操作流程，提高了消费次数。长途出行中“一票难求”的窘境曾经是阻碍大家远行的障碍，现在实名认证网络购票不仅方便快捷而且十分安全，为顺利出行提供了新的解决方式。崇尚美食是中国人孜孜不倦的追求，因此造就了国内庞大的餐饮市场。互联网结合餐饮业，诞生出的网络订餐便捷应用，解决了无数都市白领餐饮的需求，已广泛得到消费者的认可与青睐[2]。

互联网的发展日新月异，不断刷新我们对生活新的认识。对于互联网的发展我们并不能忽视起着奠基的作用的计算机网络。

但无论是计算机网络的教学还是学习之路都布满坎坷。在教学过程中，由于计算机网络理论知识偏多，造成教学重心自然地向理论教学倾斜，实践教学沦为了辅导甚至直接取消。导致学生即使用心学，却苦于理论知识难以理解，实践机会不足，教学成果收效甚微，更不必说满足新形势下日新月息的需求[3]。而在学习过程中，以学生的视角来看，计算机网络是大二必修的计算机基础学科，该学科融合了通信工程与计算机基础知识，知识间联系紧密度不高，在学习完一个模块后，知识大致有个认知，可几周过后缺乏相应的实践内容辅助记忆，遗忘速度陡增。尤其是网络的分层结构，教师多次强调要理解背诵，苦于没有实践操作，遗忘是轻而易举的事情。这导致在在日后学习网络相关协议时，明显产生力不从心的感觉。例如对于协议的学习，学生只朦胧知道有协议头字段的概念，但是协议字段真实意义如何，同学的大多只是人云亦云，知其然却不知所以然，无法将课本的内容转化为自己的知识。

网络人才的培养发展并不乐观，可对于网络人才尤其是高素质人才的需求却十分急切。据一份针对川渝地区网络公司及政府部门的分析[4]指出，目前计算机网络专业的岗位主要分为五大类，计算机网络组建、网络系统的管理、网站的开发与组建、计算机及网络产品的维护维修与销售。以上岗位均对计算机网络知识能力有要求。网络早已渗透到各行各业，对应于网络相关从业的要求将被提高，对于网络设备的调试、安装，网络的组建、管理、维护工作都将对网络从业人员提出新的挑战。新技术不断涌现，只是表面现象，对于从业人员务必学会戳破现象看本质，做扎实网络基本功知识是成为高素质网络人才的不二法门。

## 1.2 课题目的和意义

本课题的目的可分为以下两点：其一是为网络初学者完善知识框架，其二是为网络从业者提供一款集成式的局域网监管软件。

对于网络初学者，计算机网络学习的痛点有以下方面。其一是理解网络的组成，计算机网络是通信技术与计算机网络的结合学科，这包含硬件与软件的结合，通信子网与资源子网的结合。如何迅速理清这些内容，对后续的学习十分重要。其二是对网络协议的学习，网络协议是该学科的根本，网络沟通是通过协议之间进行联系的。其三部分是网络的分类，对于初学者而言正确区分局域网、城域网、广域网不仅在范围大小上，我们还需要通过网络层面上去理解。其四是理解好TCP/IP协议族，这是将网络学习中零碎的知识进行有机的整合，通过将知识在TCP/IP结构上进行归类，将知识的一个个小点连成先升华为面，最后组成一个立体的思维结构[5]。

本研究的主要目的是为网络初学者攻破上述后三个方面，即网络协议的学习，网络的分类，理解好TCP/IP协议族。对于网络协议的学习，本研究成果能对互联网绝主流应用的数据包进行解析，分为精简信息和详细信息。精简信息，关注点落在数据包的地址和该协议的封装层次的展示，详细信息中数据包将被进行字节的解析，事无巨细呈现出数据包蕴藏的数据信息，让学习过程中原本虚无飘渺的网络协议头字段变的真实。关于网络分类的学习，本研究成果提供了一个对IP地址进行信息的查询的辅助功能，不少初学者对于私有IP和公网IP概念模糊。通过这个工具，可以学习者认识到，公网IP是如“与众不同”的，其便捷的及是反馈机制，有助于对局域网和广域网的区分从地理的理解转变为网络层理解。对于TCP/IP协议族的学习，着力点在于让学习者明晰各个网络层次划分关系的不同，重点学习在网络层（IP）与传输层（TCP/UDP）上面，对于以上主流的协议，本研究成果完善地对其进行了详尽的解析，务求令初学者能够看到数据包中各个字段的准确解析。

对于网络的从业者，他们的需求与网络使用者有所不同，他们日常处理的对象并不是自己的主机而是存在于局域网的主机，所以本课题成果为覆盖更广泛的使用人群。为从业者提供了如下的功能，局域网主机扫描与信息导出，对局域网主机进行数据包的捕获，对于网络协议的过滤功能，对于网络信息的统计绘图，对于计算机术语的查询等综合功能。

本课题实际目的显而易见。为计算机网络的初学者提供了知识框架提供实践平台，使其能对计算机网络拥有一个“可视化”的视图，加深对计算机网络知识的理解与认识，从畏惧网络转变到理解网络。为网络从业者提供一个简易携带，开箱即可使用的网络协议分析工具，使得繁复的网络资源信息统计与记录变得简便而迅速。

## 1.3 国内外文献综述

本设计使用的核心技术为网络嗅探。

嗅探器（Sniffer）国外专业名词为（Packet Analyzer），直译为数据包分析仪（以下使用嗅探器代之），它可以是硬件或是软件构成的系统，主要目的是对流经数字网络或一部分网络的网络流量进行截取和记录[6]。与之成对出现的还有一个名词是（Packet Capture），直译为数据包捕获，其主要强调对网络流量进行截获和记录的动作。需要指出的是，虽然它们的职能相类似，但在国内嗅探器与数据分析仪器二者是有区别的，嗅探器一般为纯软件实现，而数据分析仪器会包含硬件部分。

嗅探器的工作原理可大致分为以下三类，一是中间人攻击的嗅探原理，二是基于ARP欺骗的网络欺骗（Spoof）原理，三是网卡设置为混杂模式的嗅探原理。以下对第三种方式，进行补充说明。网卡的工作方式可分为四种，以下依照收发数据包的“范围”由小到大进行讲解。直接方式，该模式下只有目的网卡才允许接收数据，在以太网帧表现为接收目的地址为本机MAC的数据包；组播模式，在该模式下网卡能接收到组播数据包，在以太网帧表现为接收所有多播数据帧无论自己是否为组内成员；广播模式，在该模式下网卡能接收到网络中的广播数据，在以太网帧表现为接收MAC地址为0xFFFFFF的广播帧；混杂模式，在该模式下网卡能够接收网络中一切通过的数据。[7]网卡的默认工作模式包含直接模式与广播模式。

嗅探器与数据包分析技术是许多安全软件实施的基础，该技术应用已在业内国内外的应用软件中得到了较为完善的发展，下面我们将回顾利用该技术为核心所研发的一系列软件。

1. Wireshark

Wireshark是一款免费的开源数据包分析工具。它是一款可用于网络故障排除、分析、通信协议研发和教学的软件。该软件啊最初命名为Ethereal，于2006年5月由于项目商标的问题，更名为Wireshark。Wireshark使用的Pcap数据包格式是跨平台的，能运行于各大操作系统平台之上，如Unix-类系统，微软，苹果[8]。下图为Wireshark运行的界面。

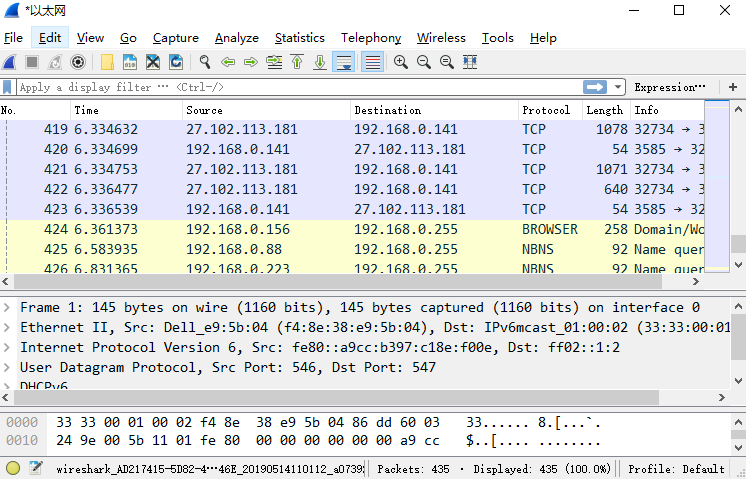


图1.1 Wireshark 运行界面

1. Tcpdump

Tcpdump是一款通用的数据包分析软件，它通过命令行运行。允许用户显示通过计算机所接入网络传输中收发的TCP/IP协议栈等类型的数据包。该软件基于BSD许可证分发，属于免费软件。Tcpdump适用于大多数类Unix操作系统，如Linux、Solaris、FreeBSD、NetBSD、OpenBSD、OpenWrt、macOS、HP-UX、AIX等，在上述系统中Tcpdump利用Libpcap库进行捕获数据包，在Windows系统中则使用WinPcap库，其对应软件名为WinDump。

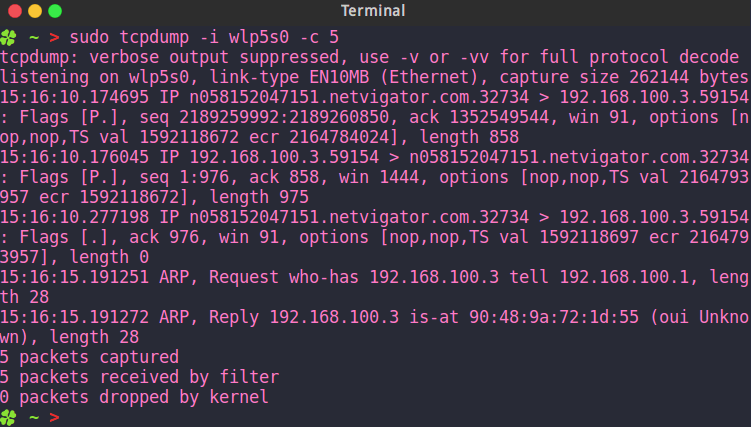


图1.2 Tcpdump 命令行运行

1. Softperfect Network Protocol Analyzer

SoftPerfect网络协议分析器是一款免费的专业软件，用于分析、调试、维护和监控网络和Internet连接。它可用于捕获以太网卡传输的数据，分析这些数据，然后以可读的形式表示出来。对于网络管理人员，安全专家，网络应用程序开发人员，以及需要全面了解网络连接或局域网流量的人而言，这是一款十分有用的工具。

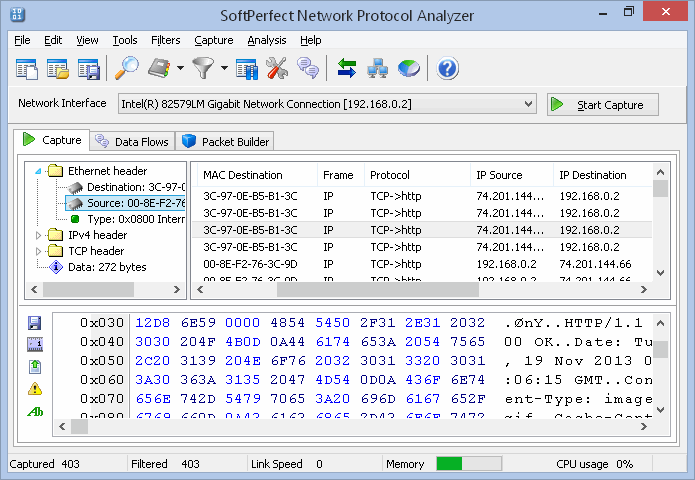


图1.3 Softperfect 图形界面

1. Capsa-free

这是一款国产的网络协议分析仪，来自国内网络分析专业厂商科来，也是一款免费的网络分析软件。它能为用户提供丰富的体验，可以学习如何监控网络活动，查明网络问题，增强网络安全性。Capas-free是Capsa网络分析仪的特殊版，适合于学生、教师以及对计算机协议学习的爱好者。



图1.4 Capsa-free图形界面

# 第2章 相关理论与技术

## 2.1 计算机网络知识

### 2.1.1 网络层次划分及其功能

网络协议(Protocols)的开发通常是依照不同的层次进行的，不同的层次负责通信中不同的部分。对于一个协议族/协议族而言(Protocol Suite)，例如TCP/IP协议族，它由多层次不同的协议协同组合而成的。对于TCP/IP协议族，它被划分为4个层次，如下图所示。

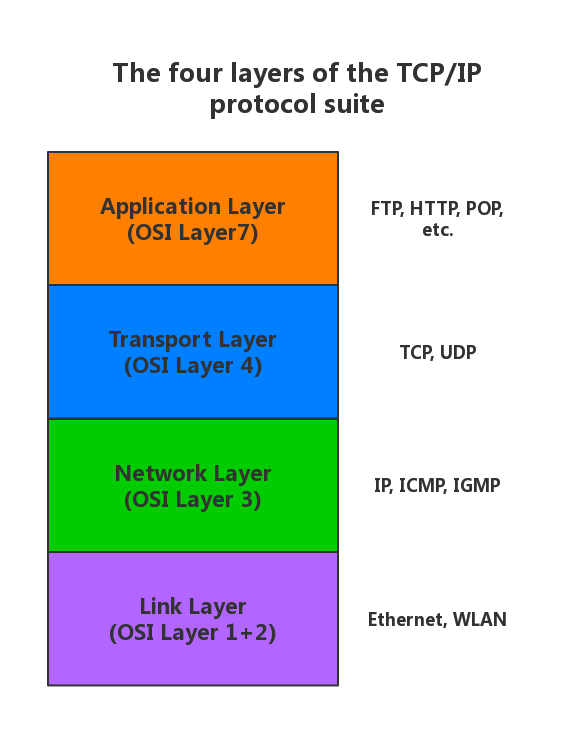


图2.1 TCP/IP分层图

不同层次决定了不同的功能，以下将自底向上进行讲解。

链路路层(Link Layer)，该层也被称为(data-link)数据链路层或网络接口层(Network Interface)。该层处理包括操作系统中的设备驱动以及计算机物理设备的网卡(Network Interface Card)的事务。具体体现为传输前将比特编码为数据包(Packet)，接收端将比特封装为数据帧，比特作为计算机通信中最基本的信息单元，数据包是所有现代计算机网络信息传输的基本单位。该层的职能还包括逻辑链路控制、媒体访问的控制、硬件寻址、差错检测和处理以及物理层定义的物理标准的工作。通过必要的差错控制与流量控制来传输数据包，达到可靠数据传输的目的。

网络层(Network Layer)，该层也称为网际层(Internet Layer)。负责路由功能，解决物理地址与逻辑地址的转换问题，在网络路径中依照给定的算法选择一条合适的路径进行数据包的传输。

传输层(Transport Layer)，盖层负责维持网络上端到端的通信，传输层响应来自上方应用层的服务请求，并向下层的网络层发出服务请求，以此在两台主机之间建立对话（虚连接）。该层主由两个协议组成，一个是面向连接的TCP协议，另外一个是无连接得UDP协议。区别在于TCP发送的每个数据包都要求接收方提供确认反馈，而UDP则不需要任何确认反馈信息。

应用层(Application Layer)，该层提供给人或软件使用网络资源，提供了网络用户接口和支撑服务例如收发电子邮件的POP3与SMTP，远程主机登录的Telnet、SSH、Rlogin，文件下载的FTP，网页浏览的HTTP等。

### 2.1.2 数据包封装与解封

数据包从一台主机发送到另一台主机这一过程，对发送方而言，需要对数据包进行封装(Encapsulation)，对接收方而言需要对数据包解封(Decapsulate)，而数据处于不同的层次用专属名称进行区分，具体如下图所示。

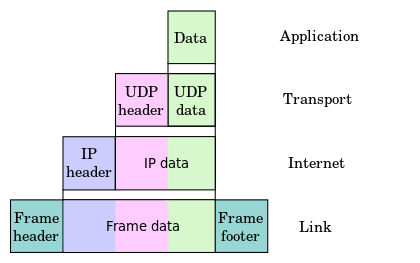


图2.2 数据包层次

在数据包发送过程中，数据由应用层中相关应用产生，应用层将数据递交给传输层，传输层将按照 “需求”使用TCP或UDP对数据进行封装，封装后传输给网络层，网络层进行IP头部的封装，封装完成后，将数据包传递给数据链路层，数据链路层进行以太网头和尾部校验的封装，最终将数据包通过传输介质发送出去。

在数据包接收过程中链路层将接收到整个数据包被称为帧(Frame)，当数据链路层剥离了以太网头部和尾部后数据包被送到网络层，此时将得到包(Packet)，当网络层将IP头部剥离送达传输层后，将得到以UDP或TCP为头部的内容称为报文(datagram)，当传输层将其头部剥离传输给应用层后，将得到真正的数据(data)。

封装与解封是一组互逆的过程，可形象地将其理解为邮政包裹的包装和派送这两个动作。

### 2.1.3 主机间通信过程

主机通信过程以FTP通信作为实例作为讲解，过程如下图所示。

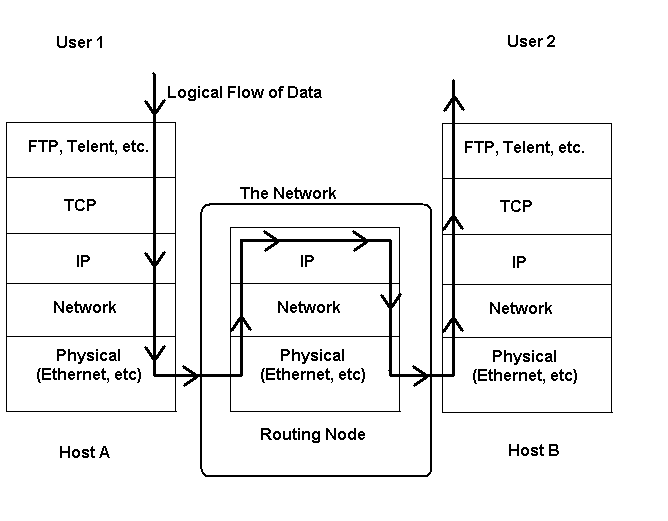


图2.3 通信过程图

假设User1要与User2的FTP相关进程进行通信。User1的FTP进程会先把需要发送的数据交给传输层，由于FTP传输对传输有数据完整有极高的要求，因此FTP应用进程选择将传输任务交给传输层的TCP协议。传输层接收到数据后，为数据包封装TCP头，原本“裸露”的数据变成了报文，报文继续发送到下层网络层，网络层接过报文，为其封装IP头部，将报文变成了包，网络层继续将包递交到下一层数据链路层，数据链路层为包打上以太网头部和尾部，将数据包变成了以太网帧，紧接着网卡以比特流的形式，将以太网帧送出网卡到达传输介质。

当这串比特流离开User1的计算机后，通过传输介质到达一台至多台的路由器(Router)，仅以图例讲解，路由器是一台拥有三个网络层次的设备。比特流在路由器的数据链路层将会被会解封以太网头部和以太网尾部，接着进入路由器的网络层将会被解封IP头部信息，依据相应路由算法计算出下一条的地址，将计算出新的地址重新封装新的网络层信息，封装新的以太网信息，通过路由器网卡的出口，再次以比特流的形式传输到传输介质上。

经过一次或多次路由的地址切换后，数据包最终将抵达User2的主机。此时User2主机的网卡接受该串比特流，比特流通过数据链路层解封以太网头部和以太网尾部，将包移交给网络层，网络层解封IP头部将报文移交给传输层，传输层解封TCP头部，将数据移交给上层的FTP应用进程。至此，User1的数据包顺利到达User2主机上，完成一个数据包的收发过程。

撇开通信的细节而言，网络中的通信就是依照上述的流程进行的。网络通信的流程如此复杂却能有条不紊的进行下去，关键因素在于使用了层次划分的理念。分层所带来的好处是明显的，各层次独立工作、灵活性好、结构上可以分割开、易于实现和维护和能促进标准化工。[9]

## 2.2 地址与端口

### 2.2.1 MAC地址

MAC地址(Media Access Control address)直译名称为媒体接入控制地址。该地址是网络接口控制器(Network Interface Card)的唯一标识。同一网段的通信而言，它被用于以太网中进行局域网通信寻址，MAC地址在TCP/IP族中位于链路层。

MAC地址的格式。MAC地址由48位二进制（6个字节）组成，以16位作为一组（2个字节）用16进制对8位二进制进行编码，使用冒号（:）分割。由于MAC地址是NIC的唯一标识符，其地址的分配和使用需要监管，该工作目前由IEEE的注册管理机构RA(Registration Authority)负责。其负责分配MAC地址字段的前24位（3个字节），因此对于MAC地址前24位被称为组织唯一标识码OUI(Organizationally Unique Identifier)，注册厂商设定其为自己得公司标识(Vendor Id)。获得OUI分配号码的厂商，可以自由使用MAC字段的后24位（3个字节），不进行地址保留下可分配的MAC地址数量为2^24个不同的地址。利用以上方法得到的48位地址称为EUI-48 (Extended Unique Identifier)。需要指出的是，不应该依靠24位OUI标识码来对应标示一家公司，因为有可能是几家公司一起购买同一OUI标识码，也有可能是一家公司购买了多个OUI标识码。

除了上面的OUI码划分外，IEEE还作出了以下的规定： MAC地址的第一个字节，最低位为组播(Multicast)和单播(Unicast)的区分位，其为1时表示组播可表示可进行多播，当其为0时，表示单播地址。MAC地址第一字节的倒数第二位为全球管理(Global)和本地管理(Locally)区分位，其为0时表示为全球地址代表这是向IEEE申请购买的OUI，当其为1时表示为本地管理，这时用户可以任意分配网络上的地址。

具体图示如下所示。

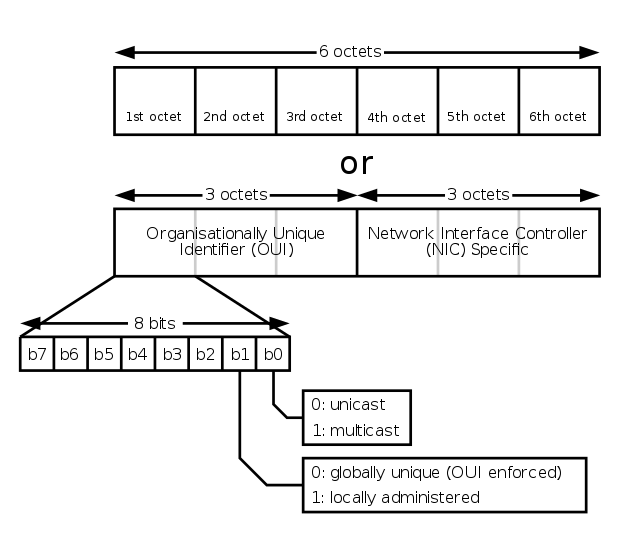


图2.4 MAC地址格式

### 2.2.2 IP地址

目前，IP地址(Internet Protocol Address)可分为两类，IPv4与IPv6，IP地址位于TCP/IP协议族中网络层，本小结只对IPv4相关内容进行介绍。

IPv4依然是我们如今最常用的网络层协议。IPv4地址大小为32位，其理论可分配地址数为2^32。但一些地址被保留用作其他的目的，例如专用网络(Private Network)用去了约1800万个地址，多播寻址用去了约2.7亿个地址。IPv4地址采用点分十进制(Dot-Decimal Notation)的方式表示，由四个十进制数组成，每个十进制数的范围从0到255，用点分隔，每部分均表示一组8位二进制。详情可见下图。



图2.5 IP地址点分十进制

在网络协议开发的早期，网络号使用最高位的前八位作为划分的，最多可分配网络数量为256个。这种分配额方式很快被证实难以应对发展，因此1981年，引入了IP分类网络地址(Classful Network)架构修改了寻址规范。分类网络地址将IP地址划分为，A、B、C、D和E共五类，其中A、B、C类为单播地址，D类为多播地址，E类保留为以后使用。单播地址的划分均由两个字段组成，一个字段为网络号(Net-Id)，另一个字段为主机号(Host-Id)，主机号与网络号二者组合成完整的IP地址，主机号和网络号以及IP地址长度的关系：“网络号长度 + 主机号长度= 32位”。网络号长度确定划分网络的数量，主机号长度决定了一个网络中最多可容纳的主机数。对于A类地址而言，网络号的长度为8位，可分配的网络数256个，每个网络可容纳主机理论值16777216台；B类网络号长度为16位，可分配网络数65536个，每个网络可容纳主机理论值65536台；C类地址网络号长度为24位，可分配网络数6777216个，每个网络可分配的主机数量理论值256台。

随着网络的进一步发展，人们发现A类B类网络的划分存在着利用率低下的问题，一个A类网络中原本可以分配16777216个设备使用，实际生产环境下，却造成了无数地址的浪费。为提高分类网络地址的利用率，人们提出了子网划分 (Subnetwork)，使得原本由两个字段组成的IP地址，变为三个字段组成，保留网络号，主机号拆分成由子网号和主机号两部分构成。其关系如下：“网络号长度 + 子网号长度 + 主机号长度 = 32位”。有了子网号码就必须拥有标识子网号的标记，由此产生了子网掩码(Subnet Mask)，其设计与IP地址格式相类似，由32位二进制组成，其网络号和子网号被全部填充为1，主机号为0。IP地址只要与子网掩码进行异或操作，即可得出子网的网络地址，这对于网络设备路由寻址相当的重要。

在1987年，一份RFC指出在一个划分子网的网络中可同时使用多个不同的子网掩码，这被称为变长子网掩码(Variable Length Subnet Mask)，该技术可更进一步提高IP地址的利用率。在以相同需求下，比较使用VLSM进行地址划分与不使用VLSM进行地址划分二者对IP地址的消耗情况。

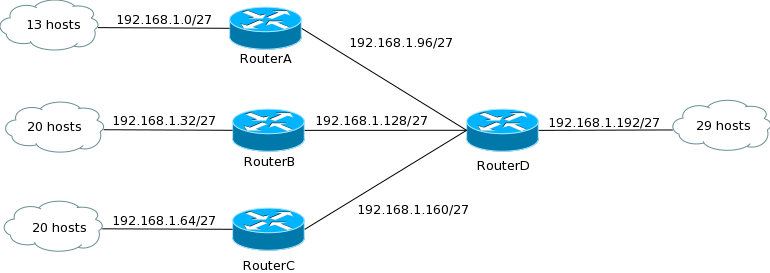


图2.6未使用VLSM进行地址规划

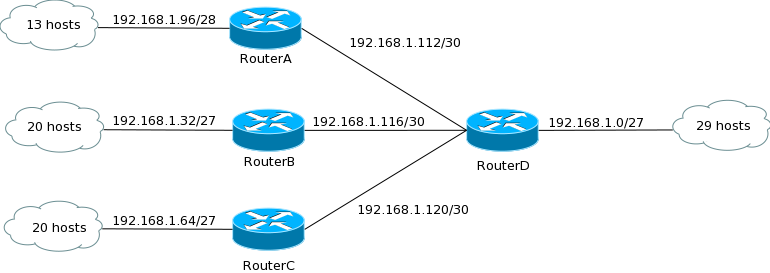


图2.7使用VLSM进行地址规划

在未使用VLSM的图中，可以发现最大的单个子网需要至少容纳29台主机。因此主机号为五位，其子网掩码长度为27位，刚好达到划分的要求。但不难发现，除了与RouterD相连的子网外，其余的子网都存在着或多或少未进行分配的IP地址，这些地址由于都是同属于一个子网，即使该子网不进行分配，其他网络也不能进行使用，这将造成了大量的浪费。在路由器之间相连的接口上，由于这些接口使用了27位子网掩码，本可分配30个IP，现在分配出两个，28个IP地址白白流失。对没有使用VLSM地址划分的拓扑图，统计可得将有118个IP地址在这次分配中被浪费，而分配使用的IP数仅为92个。

在使用了VLSM的图中，比可以看出，其区别在于子网掩码的位数变的可以自由调整，这正是VLSM这个名字的由来。通过自由调整子网掩码的位数，可以实现按需分配，这次分配92个IP地址，仅浪费了22个IP地址，利用率有了明显的提高。

由于IP地址使用越发激烈，在VLSM的基础上又进步研究出了无类编址方法，命名为无分类域间路由选择CIDR(Classless Inter-Domain Routing)，这是一种用于用户分配IP地址以及在互联网上有效路由并对IP数据包的对IP地址进行归类的方法。CIDR消除了传统A、B、C三类地址及划分子网的概念，它把网络前缀都相同的连续IP组织成一个“CIDR”块，通过CIDR块中任意一个地址，我们都可以知道这个地址块的起始地址。CIDR技术用新的手段定义了IP地址，并且在路有表中也能运用上CIDR进行地址块的查询，这种地址聚合被称为路由聚合(Route Aggregation)。

### 2.2.3 端口号

端口号(Port)，具体名称为协议端口号(Protocol Port Number)，其存在于TCP/IP族的传输层。

传输层中通过使用到16位二进制(两个字节)，标识一个端口，该编号具备本地意义，用于标示本计算机的传输层与各个进程间的接口。在互联网上不同的计算机中，相同的端口号之间无任何关联。16位的端口号意味着可以使用的端口数有65536个，对于一台计算机而言是十分充足的。两台计算机要进行通信，除了需要知道双方的IP地址外确认主机身份以外，对于一个应用而言还需要知道双方的端口号，才能确认是哪两个进程之间进行通信。

为了在使用端口的时候，避免操作不当，标准化组织IANA对于端口号的使用进行了规范的制订。该规定以端口号的数值为区分标准，将0-1023定义为熟知端口(Well-Known Port Number)，这些端口指派给最熟悉的应用程序，例如FTP对应21号端口，FTP-Data对应20号端口，SSH对应23号端口，HTTP对应的是80号端口。另外一类被称为注册端口(Registered Ports)，范围为1024-40151，该范围的端口没有熟知的端口号在使用的，但使用这类端口应该向IANA申请办理规定的手续，以防止重复。剩下的从49152-65535被定义为临时端口(Ephemeral Ports)，这些端口可留给客户进程选择暂时使用。通信开始时，客户端进程使用这些端口与服务端进行数据收发，通信结束后客户端释放该端口，循环给本机其他客户进程再次使用。

## 2.3开发涉及知识

### 2.3.1 嗅探技术

嗅探指的是窃听网络中流经的数据包，窃听的对象一般在局域网内。使用嗅探技术的人员有很多，常见的是网络从业人员，他们希望对网络相关数据包进行取样、调查、排错确认网络的情况，但也有像Cracker这类为了盗取别人的银行密码等信息使用嗅探器对数据包进行分析的。该技术是一把双刃剑，关键取决于使用者。

目前以太网俨然成为了局域网的代名词，以太网采用的是CSMA/CD的协议，这个协议在同一时间内只运行一台计算机发送数据，势必造成了所有的主机都必须进行对局域网内的监听，如果A主机在发送数据后产生冲突，需要依照相关的算法进行冲突监测，监测没有冲突后才允许再次进行数据包的发送。既然需要检查，那么局域网势必要进行广播的方式对这些信息进行发送，让每台主机都能收到。正因为广播这个特性，通过对网卡的工作模式进行特定的设置，我们的嗅探器能够轻易地得到局域网内主机见收发的数据。

### 2.3.2 ARP攻击技术

ARP协议(Address Resolution Protocol)，该协议主要在局域网中被使用。其设计的初衷是将IP地址映射到相应的MAC地址上，这是由于局域网内一台主机要与另一台主机通信需要通过MAC地址来寻找主机，ARP协议正是为了完成该功能而创造的。例如在局域网中，A主机希望和B主机进行通信，A主机首先检查IP地址是否在局域网内，接着检查自己的ARP映射表是否有这台主机的MAC地址，如果A主机没有B主机MAC地址的信息，就会向全网发送ARP数据包，包内包含自己的IP、MAC地址以及B主机的IP地址。B主机收到这个ARP请求后，便立即发送ARP回复报文，告知A主机自己的MAC地址，并且记录A主机的MAC地址到IP自己的ARP缓存中。A主机收到B主机的MAC地址，将其记录到自己的ARP缓存中，封装数据包，发送数据给B主机。至此，A和B主机之间就建立起通信连接了。

ARP攻击，是一种利用ARP协议伪装主机，进行欺骗的攻击行为。攻击者一般会将自己伪装成网关，通过大量伪装ARP响应或请求报文，在数据包中将自己的IP映射到网关的IP上，将该数据包大量发送给目标主机，目标主机对于ARP报文由于没有识别能力，一般收到ARP报文后便会自动更新自己的ARP缓存，导致将自己的网管IP映射到了攻击者得MAC地址上。由于局域网中寻址依靠的是MAC地址，因此通过ARP缓存毒害后，目标主机便会将本应发送给网关的数据包误发到攻击者的手上。

本章简单阐述了网络的分层知识，通信中每个层次间的不同的标识，并对各个标识进行了简单的梳理。对于开发涉及的技术要点给出了简单的概括，为后续篇章的展开提供知识铺垫。

# 第3章 需求分析

## 3.1 需求获取

要求开发以下系统：捕获网络中传输和发送的数据，分析数据提取有用信息，记录下网络流入流出的流量及网速。在数据包抓取之前，提供有一个筛选的机制只筛选出需要的数据包，重点是能够对不同的应用数据进行过滤，为数据分析提供便利。有了网络数据捕获的功能，系统捕获的网络数据应该不仅是个人主机的，还应该具备到局域网来捕捉数据，系统主要考虑的因素为实用性，这意味着系统必须要能解决特定环境下的网络问题。

需求记录(需求编号，需求类型，需求来源，需求原语，需求内容)：

对于上述的需求资料，必须要进行细致化的整理，理清需求，进行相应的记录。为了表述更为清晰，这里将使用表格进行需求的记录，需求编号的设计上，格式上使用三位数字不满足三位的数字从左侧补零，数值上从一开始相邻序号依次递增。需求类型，按照需求的性质划分，分为显性与隐性，显性需求为客户通过材料得出的信息，隐形需求为通过内容分析得出的信息。需求来源，提出需求的主体。需求原语，当需求来源为资料，标明出自资料的句子。需求内容，明确需求完成的任务是什么。

以下为需求记录表：

**表3.1 需求记录表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 需求序号 | 需求类型 | 需求来源 | 需求原语 | 需求内容 |
| 001 | 显性 | 需求材料 | 捕获网络中数据 | 对网络数据嗅探 |
| 002 | 显性 | 需求材料 | 提取有用信息 | 数据包进行解析 |
| 003 | 显性 | 需求材料 | 记录数据流量 | 网络流量记录 |
| 004 | 显性 | 需求材料 | 筛选出数据包 | 过滤器设置 |
| 005 | 显性 | 需求材料 | 解决网络问题 | 实际应用 |
| 006 | 显形 | 需求材料 | 扩展到局域网 | 全局嗅探功能 |
| 007 | 隐性 | 资料分析 | — | 本机局域网区分 |
| 008 | 隐性 | 资料分析 | — | 过滤字段设置 |
| 009 | 隐性 | 思考联想 | — | 抓取信息保存 |

表3.1(续表)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 需求序号 | 需求类型 | 需求来源 | 需求原语 | 需求内容 |
| 010 | 隐性 | 思考联想 | — | 读取保存文件 |
| 011 | 隐性 | 思考联想 | — | 辅助功能延伸 |

## 3.2 分析需求

### 3.2.1 表面需求

通过上一阶段的需求获取，摘录得出了用户11个需求。现将进行需求的总结，该总结被称为表面需求，这些需求的得出只是根据用户一方的依据，而用户对产品的定位和设计的依据并不充足，他们的建议和需求许多时候并不是产品设计的最终参考。

现有需求分析如下，用户希望能够开发一款可以能应用于本地主机和局域网主机进行网络信息数据包进行过滤和分析，对网络流量进行监视，对汇聚的数据能够实现归档重复利用的应用。

### 3.2.2 本质需求

通过以上表层的分析，不难看出，用户在寻觅的一款集网络嗅探外和网络协议分析与一身的软件，这两个功能在本地主机上实现困难是不大的，唯一比较麻烦的是如何在局域网上执行。为此，在寻找解决方案中，一种方式是使用客户端与服务器端，两端分别启动，每当客户端收送数据时，便向服务器端发送数据进行反馈，原则上实现并不会有问题，但开发的成本和测试的难度提高了不少。另外一种解决方式是通过ARP协议的欺骗攻击，通过对局域网上主机发送ARP报文，伪装自己为网关，通过在本机上开启路由转发(IP Forwarding)的功能，数据包转发给真正的网关，再对网关进行欺骗，让网关误以为本机是发送数据的主机，得到接收的数据包，从而实现局域网的监听功能。这种方式明显更为灵活，因此这里应选择第二种开发方案。

### 3.2.3 产品需求

综合以上的分析，我们得出了开发真正的产品需求。沿用嗅探器与网络协议分析的技术，开发出一款具有混合功能的产品，并为此款产品配备一个有力的过滤器，对于局域网的抓包功能，我们引入ARP攻击模块，以ARP数据包为基础，利用ARP的广播功能探知局域网内存活的主机，以ARP的欺骗，作为监听局域网监听的手段。

### 3.2.4 需求排序

需求的功能是有分不同层次的，根据Kano模型(Kano model)(如下图所示)，用户的需求可分为以下三类，基本型需求(Basic Expectation)，期望型需求(Performance)，以及兴奋型需求(Delighter)。在基本型需求中，所有功能都必须满足，没有用户会选用一个残缺的产品；在期望型需求中，用户会为产品提供更为优秀的功能而得到满足；在兴奋型需求中，挖掘用户潜意识中并没有注意到的需求，这类需求往往能为的是给用户带来惊喜和信赖。

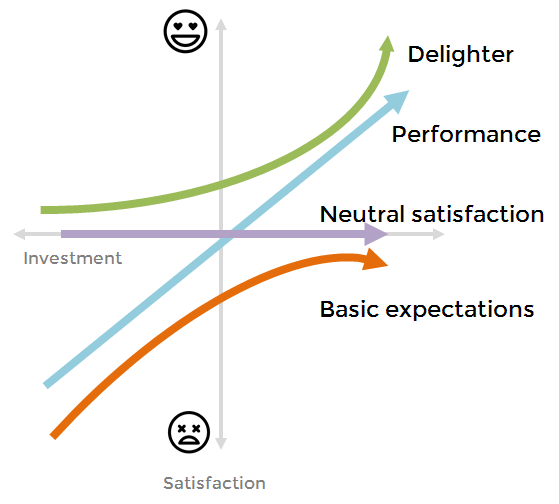


图3.1 Kano模型图

以下将需求按Kano模型进行表格记录：

**表3.2 Kano需求分析表**

|  |  |  |
| --- | --- | --- |
| 需求类型 | 功能 | 实现方式 |
| 基本型需求 | 数据包的捕捉 | 嗅探器原理 |
| 数据包的分析 | 网络协议知识 |
| 网络流量的查看 | 调用相应统计接口 |
| 数据包过滤 | Tcpdump 产生过滤编码 |
| 局域网网络分析 | ARP扫描与ARP欺骗 |
| 期待型需求 | 数据包的保存 | 选用\*.pcap二进制格式 |
| 数据包的读取 | 选用\*.pcap二进制格式 |
| 导出本地主机网络信息 | — |
| 导出局域网所有主机信息 | ARP扫描 |
| 导出数据包概要信息 | — |
| 兴奋型需求 | IP地址信息查询 | 通过网络API |
| 计算机术语查询 | 通过网络API |
| 流量/数据包进出图 | 绘图工具包 |
| 数据包统计图 | 绘图工具包 |
| 数据包长度统计图 | 绘图工具包 |

本章首先阐述了需求资料，通过对资料的细致解读，得出功能的需求记录表。对表格信息进行分析，得到表层分析，通过一步步深入理解本质需求，将本质需求烙印在产品需求中。最后一张Kano表对需求进行排序，为开发顺序提供依据。

# 第4章 系统总体设计

## 4.1 划分子系统

系统总体设计由子系统划分开始，子系统划分依据以下三部分进行阐述：子系统名称、子系统功能以及子系统在系统中担任角色。子系统名称，为子系统提供一个功能标识的名字；子系统功能，对子系统承担的主要业务进行梳理；子系统在系统中担当的据色，揭示出子系统与系统之间的关联。

子系统划分表格如下：

**表4.1 子系统划分表**

|  |  |  |  |
| --- | --- | --- | --- |
| 子系统编号 | 子系统名称 | 子系统功能 | 在系统中担当角色 |
| 001 | 网络信息子系统 | 本模块负责对本机网络信进行处理，以及信息的导出 | 获取基本网络配置信息，为后续进行网络扫描提供数据基础 |
| 002 | 局域网扫描子系统 | 对局域网内存活主机进行扫描，获取存活主机的信息予以显示，以及信息的导出 | 借助本机信息模块的数据，为系统提供存活节点，为后续抓包提供选择 |
| 003 | 数据包过滤器子系统 | 对数据包的捕捉建立筛选机制，提供对感兴趣数据包的捕获 | 过滤出需要的数据包，为系统完善数据包捕获机制 |
| 004 | 数据包捕获子系统 | 对满足条件的网络数据包进行捕获，尝试进行初步解包，简明数据包信息的导出 | 对满足条件的网络数据包进行捕获，尝试进行初步解包，简明数据包信息的导出 |
| 005 | 数据包信息展示子系统 | 对数据包进行解析工作，提供详细的解包信息展示 | 承接数据包捕获模块，对数据包信息进行加工处理分析 |

表4.1(续表)

|  |  |  |  |
| --- | --- | --- | --- |
| 子系统编号 | 子系统名称 | 子系统功能 | 在系统中担当角色 |
| 006 | 统计绘图子系统 | 对网络流量，数据包类型，数据包长度等信息进行统计友好展示 | 优化信息的显示，为系统提供出色的数据统计呈现 |
| 007 | 数据导出子系统 | 对获得的数据进行相关的记录 | 为系统获取的信息进行文档化存档 |
| 008 | 协助支援子系统 | 网络抓包中需要用到的功能，但不属于抓包应用的主要内容 | 辅佐工具箱，为系统提供有里的辅助价值延伸 |

## 4.2 子系统模块结构

子模块结构，将依照以下三部分进行阐述：子系统名称、模块的组成和各模块职能。其中模块的组成，表明子系统中包含的模块具体有那些，这写模块是真正编码的落实内容；各模块职能，具体描述每个小模块之间的工作内容。

子系统设计模块表如下所示：

**表4.2 子系统模块设计表**

|  |  |  |
| --- | --- | --- |
| 子系统名称 | 模块的组成 | 各模块职能 |
| 网络信息子系统 | 1.本地主机网络信息模块  2.网关设备网络信息模块 | 1.对本机的网络信息如IP地址，MAC地址等进行摘录  2.对网关的网络信息如IP地址，MAC地址等进行摘录 |
| 局域网扫描子系统 | 1.扫描方式选择模块  2.ARP数据包发送监听模块  3.扫描信息显示模块 | 1.设置两种扫描方式，IP地址段扫描与掩码扫描  2.发送ARP数据包进行数据收集  3.对扫描得到的信息进行友好显示 |

表4.2(续表)

|  |  |  |
| --- | --- | --- |
| 子系统名称 | 模块的组成 | 各模块职能 |
| 数据包过滤器子系统 | 1.过滤器实现模块  2.过滤器规则设置模块  3.过滤器提示模块 | 1.实现过滤器功能的“引擎”  2.过滤器输入界面的设置  3.过滤器规则的显示 |
| 数据包捕获子系统 | 1.数据包捕捉模块  2.数据包初解包模块  3.数据包文件模块 | 1.实现网络数据包的捕获“引擎”  2.实现数据包基本解包，解析出如抓包时间，抓包协议等内容  3.对数据包文件(\*.pcap)进行保存与读取操作 |
| 数据包信息展示子系统 | 1.数据包分层解包模块  2.数据包原始数据解码模块 | 1.对数据包进行分层解析，得到所需内容  2.将数据包的原始进行解析 |
| 统计绘图子系统 | 1.网络流量图模块  2.协议统计模块  3.数据包长度统计模块 | 1.网络流量上传下载的速度，网络数据包进出个数，统计分析  2.数据包协议类型封装统计，数据包使用协议族统计，数据包地址统计  3.数据包长度统计，TCP数据包长度统计，UDP数据包长度统计 |
| 数据导出子系统 | 1.本地信息导出  2.局域网信息导出  3.数据包摘要信息导出 | 1.本地网络信息的格式导出  2.局域网网络信息的格式化导出  3.数据包摘要信息的格式导出 |

表4.2(续表)

|  |  |  |
| --- | --- | --- |
| 子系统名称 | 模块的组成 | 各模块职能 |
| 协助支援子系统 | 1.协议查找模块  2.IP地址所属查询模块  3.计算机术语查询模块  4.网络流量记录模块 | 1.对捕获的数据包进行筛选查找  2.对IP地址(公网)信息的查询显示  3.对计算机相关术语进行查询显示  4.在一定时间间隔内，对网络流量进行记录与刷新显示 |

## 4.3 软件架构图

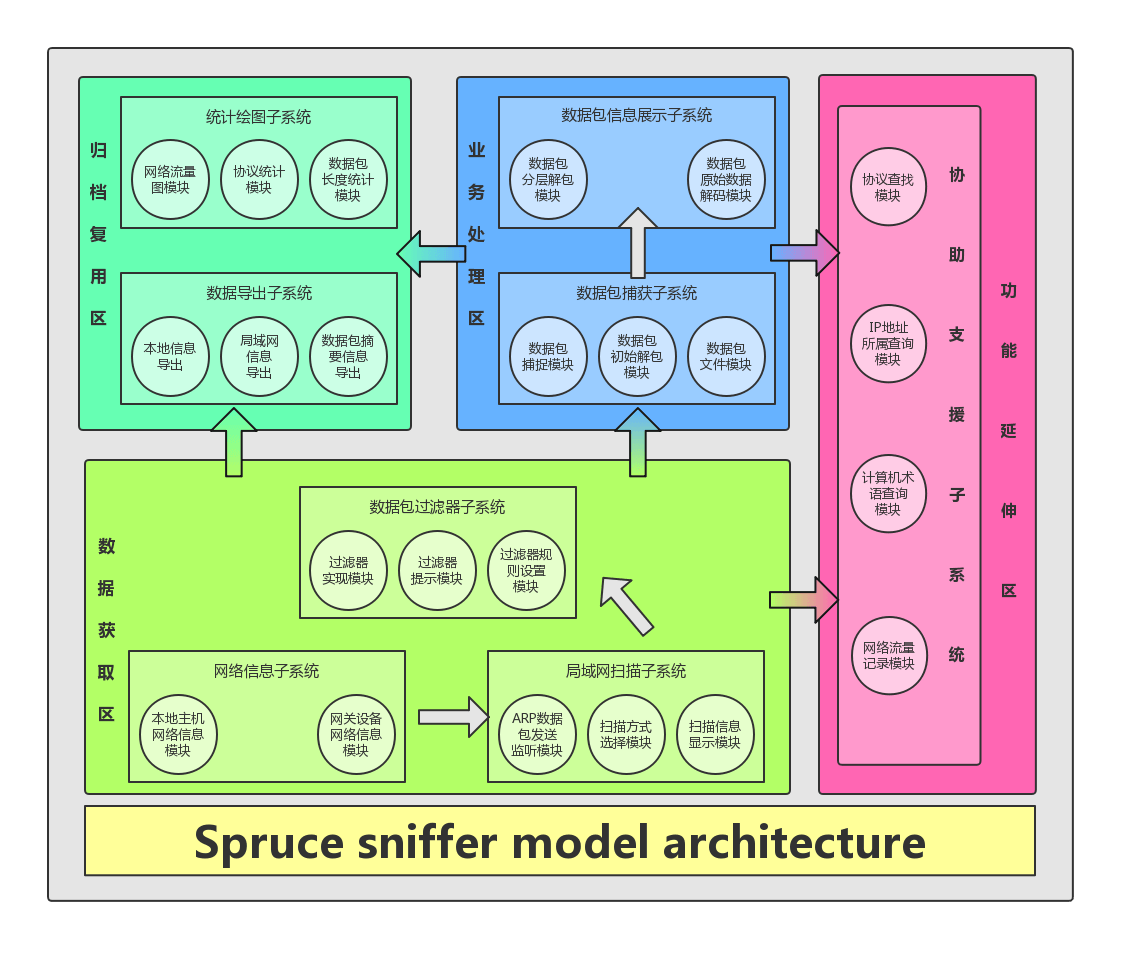


图4.1 软件架构图

该结构图是依据模块设计绘画而成的，其目的是利用图画的特性让逻辑显得更加清晰明了。需要指出的是，这里依照不同的功能特性，将系统划分为四个区域，信息获取区、业务处理区、归档复用区以及功能延伸区。其中信息获取区的作用是为了获取信息，可以理解为系统的初始化过程，通过这个数据获取我们才能进入后续的操作。业务处理区，业务处理是应用的核心功能，包括了该应用最为重要的数据包获得已经数据包分析的功能，这两个子系统是设计系统的“心脏”。无论是初始化的数据获取还是业务处理，均会产生很多有复用价值的数据，因此就产生了归档复用区，其设计目的是为了将所需复用数据进行导出保存。一个良好的应用还需要一定的辅助工具，因此这里还划分了一个功能延伸区，专门处理类似于“插件”这些辅助功能的东西。图中箭头为数据的流动方向，灰白色的箭头表示区域内的数据流动，有彩色过渡的箭头表示区域交互的数据流动。

## 4.4 软件流程图

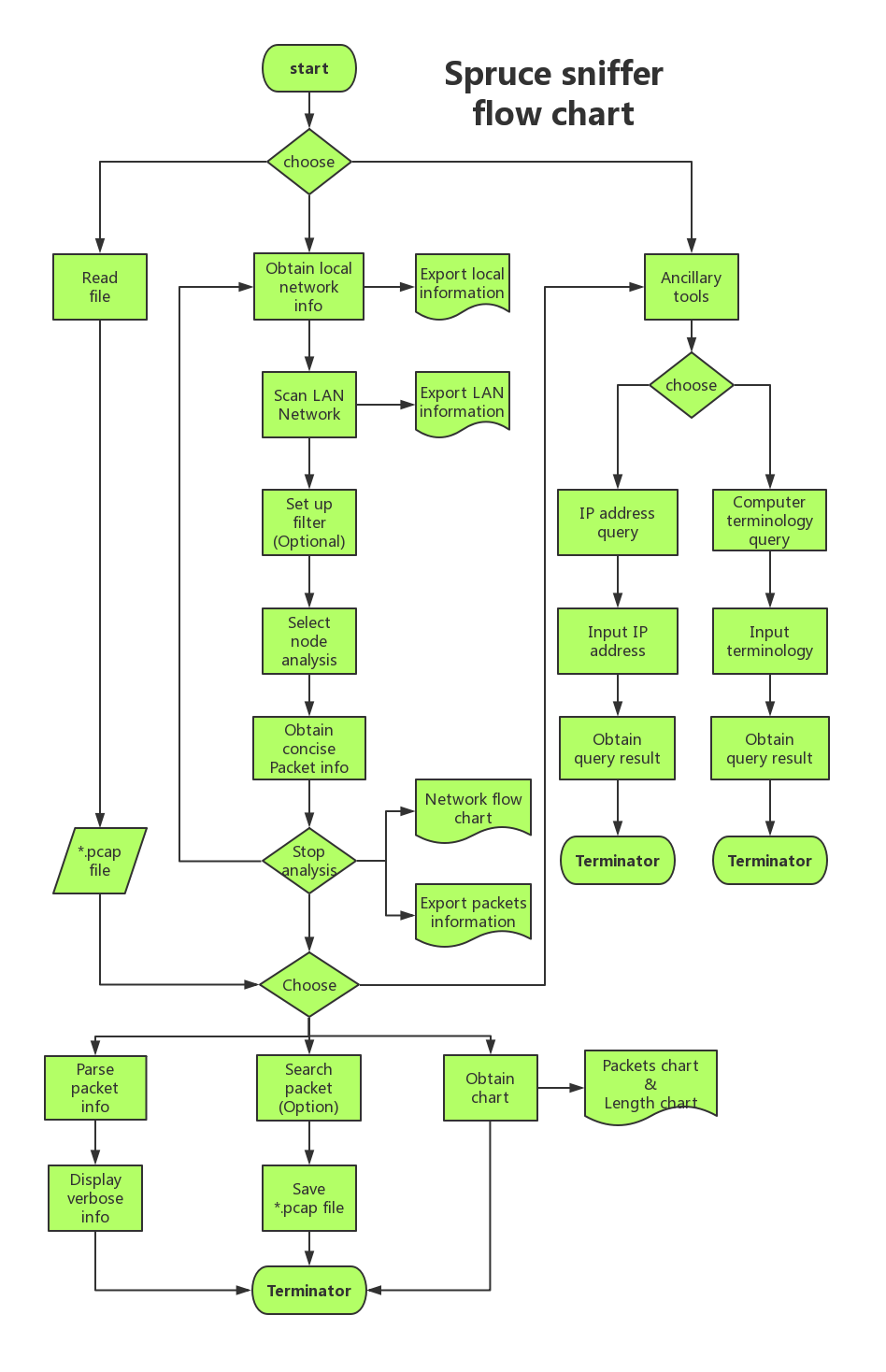


图4.2 软件流程图

上图为软件的功能流程图，设计的目的是规划与展示软件的功能。以下将对该流程图进行详细的解说，以此梳理软件的功能情况。软件在启动后有三个可供选择的功能，第一个是文件读取功能，该功能将读取你在本地上已有的Pcap格式的文件，第二个功能是辅助工具功能，这个功能集成了软件的非核心功能，与软件主业务无关，确是网络人员在数据包分析后经常会调用到的功能，一个是IP地址归属地以及相关信息的查询功能，另外一个是网络专业名词的查询。

第三个功能个人网络信息的获取，这是本应用的核心功能流程第一步。通过对个人网络信息的获取后，可以生成相应的网络信息文档。接着对局域网信息的扫描，扫描完成后，可以对局域网主机信息导出。接着用户设置过滤器，过滤器的设置是一个可选的功能，默认不使用过滤器。然后选择相应需要抓包的节点进行分析，数据包分析会反馈简明的数据包信息。需要指出，在这个过程中，你依然可以选择使用辅助工具。当得到足够的数据包后，可以选择停止数据包的捕捉，停止捕捉后，简明网络数据包信息将可被导出，网络流量的数据图也将可以被使用，在该过程中我们能够重新进行多次抓包。

接着再次进入选择环节，可选择有数据包分析，通过数据包的分析可以得到详细的解包信息，我们可以通过查询功能对数据包进行筛选，我们还能通过数据包的有关信息进行绘图如数据包协议的绘图，以及数据包长度统计数据的绘图。最后可以选择是否保持数据包，保存的文件格式为pcap格式。至此应用的功能流程结束。

本章有宏观到局部介绍了系统的总体设计，逻辑上是一个不断细分逐一击破的过程。将系统划分为多个子系统，在子系统之下根据具体功能划分为不同的模块，模块是日后开发最小的完成单元。接着为模块进行工作区域划分，绘画架构示意图。最后给出软件的流程图，确定软件的执行流程。

# 第5章 系统详细设计与实现

## 5.1 界面设计

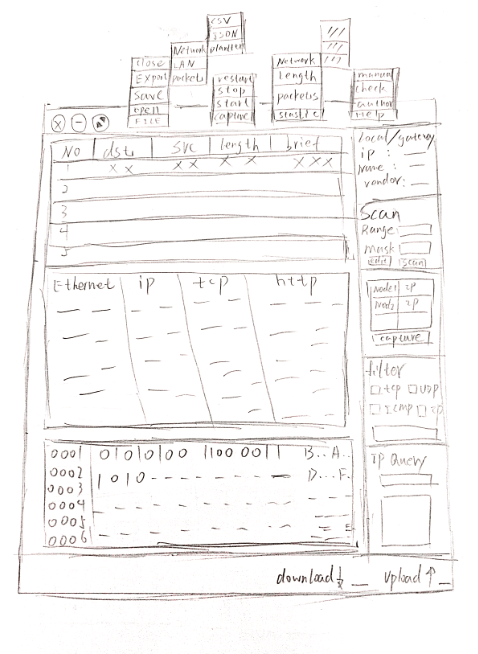


图5.1 界面设计草图

界面设计图如上所示。该设计图为主界面设计图，设计图中所有的功能模块均安放在同一界面下，下面具体介绍各部分控件的功能。

应用的绝大多数功能是通过菜单按钮实现的，依照一般的使用习惯，第一个菜单为文件菜单(File)，负责有关文件读写、数据导出以及应用关闭的功能。第二个菜单为功能菜单(Capture)，负责控制抓包的功能，开始、停止和重启抓包。第三个菜单为统计菜单(Statistics)，负责有关网络流量数据图表生成，协议长度图表的统计生成，数据包相关图表的统计生成。最后一个是帮助菜单(Help)，负责开发者的信息，使用帮助等信息的展示。

应用的主界面分为两个部分，一块是数据呈现区域，另外一块是信息配置区域。

数据呈现区域是应用的核心区域，占应用界面的三分之二。该区内含三个模块，第一部分为数据包基本信息展示，简单地对对数据包解包，可以看到数据包的编号，目的地址，源地址，数据包长度，基本信息展示等相关的条目。接着第二个部分是详细数据包的分析，该部分以TCP/IP族的结构划分为四个大块，数据包信息的头部字段详细解析将在这里完成。第三部分是编码解释区域，这部分将数据包的二进制表示展示出来，使用UTF-8编码格式对其进行解析。

信息配置区域，由五部分组成，主要处理数据获取，辅助工具等内容。对于数据获取，这里第一个模块获得的是本地与网管的网络信息，第二个模块提供了范围IP地址扫描与掩码扫描两种对局域网主机进行发现的方式。第三个模块是对活动节点的展示，为抓包提供节点的选择。第四个模块是过滤器模块，提供了过滤协议的选择，第五个模块是IP地址查询得外界模块，提供了IP地址归属信息的查询，查询结果在文本框显示。

应用的状态栏上，有下载(Download)与上传(Upload)速度的动态更新。

## 5.2界面实现

系统实现界面由Qt Designer工具辅助完成，完成结果较设计草稿有明显的优化，以下为系统的真实实现效果。

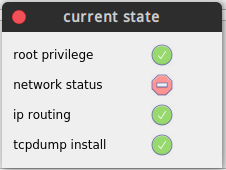


图5.2 环境测试 错误

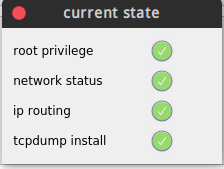


图5.3 环境测试 成功

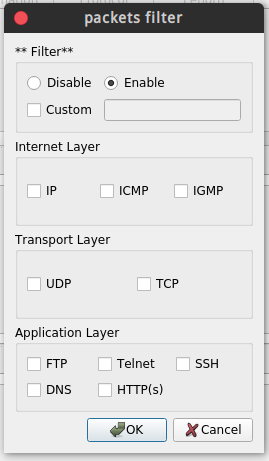


图5.4 数据包过滤对话框

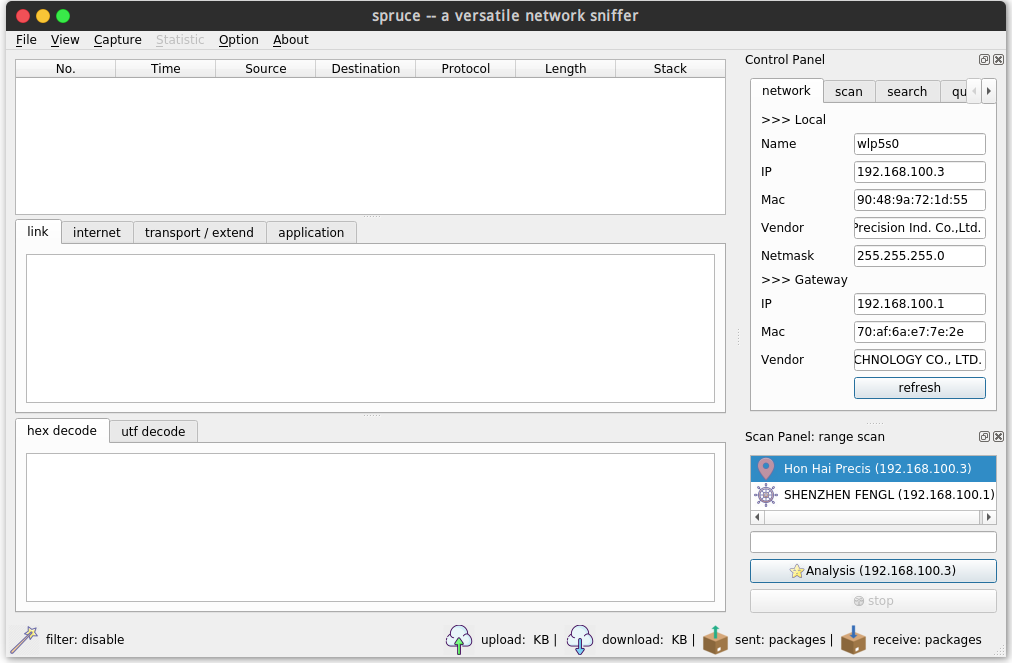


图5.5 软件实现图

上图为系统实际启动图，以下将对实际实现的用户界面和设计草稿之间进行对比，指明提升的地方。

先从界面全局进行分析，明显的发现草稿上拥挤在一起的控件消失了，更换成了内嵌的Tabs格式布局，提升了空间的利用率。在整体设计上，实现的界面增添了不少的图标点缀，为应用带来了不少的生气。

对于菜单界面，新增加了视图菜单(View)这个条目，该条目是两个关于右侧面板的复选框，提供给用户对面板进行显示与隐藏。新增加了选项(Option)这个条目，这个条目下面是过滤器选项，点击该选项将会弹出一个过滤器对话框，提供两种过滤器使用的方法。一种是通过点击的复选框常用协议进行数据包过滤，第二种是自定义过滤字段，可参见图5.4。帮助菜单更名为关于菜单(About)。除了设计草图的功能外，额外增加了Rank功能，用于查看系统环境是否符合应用运行环境的，可参见图5.2和图5.3。为了配合这个rank功能，选项新增了一条更新控件使用权的(Refresh rank)。

对于数据呈现区域，数据包基本信息呈现区域被条目中Brief字段被删除，新增了Protocol与Stack条目。详细数据包分析区域没有变化。编码解析区域从原来的结合状态，改为现在的分离式结构。

信息配置区域，得益于Dock排版，由设计草图中的固定位置，变为如今集拉伸、隐藏、浮动和折叠功能于一身的窗体。为使用的便捷性提供了无可挑剔的优势。由于需要进行ARP活动节点这类较为费事的操作，添加了一条进度条，提供给用户更好的体验。

应用的状态栏上增加了不少的图标点缀，左边增加了一个过滤器提示信息，右边新增了数据包收发数量的统计功能。

## 5.3模块详细设计

**表5.1 网络信息子系统设计表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 子系统名称 | 模块名称 | 输入 | 处理 | 算法描述 | 输出 |
| 网络信息子系统 | 本地主机网络信息模块 | — | 通过系统文件读取获得网络信息当，填写至文本框中 | — | 成功：本机的IP地址、MAC地址、子网掩码和制造商信息  失败：空 |
| 网关设备网络信息模块 | — | 通过对系统文件的读取获得网关信息，填写至文本框中 | — | 成功：网关IP地址、MAC地址和制造商信息  失败：空 |

**表5.2局域网扫描子系统**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 子系统名称 | 模块名称 | 输入 | 处理 | 算法描述 | 输出 |
| 局域网扫描子系统 | 扫描方式选择模块 | 网络相关信息 | 提取IP与掩码进行网络地址计算 | 获得网络地址信息，计算相关格式网络号 | 显示符合规范得扫描IP格式 |
| ARP数据包发送监听模块 | 网络相关信息  扫描范围 | 构建ARP数据包发送接收 | 发送ARP包监听网络，对响应数据包进行记录 | 存储的节点信息 |
| 扫描信息显示模块 | 存储的节点信息 | 扫描的信息进行整理与显示 | 获得MAC地址，根据OUI表查询vendor信息，判断主机节点类型 | 在列表中显示存活节点的IP与制造商信息 |

**表5.3数据包过滤器子系统**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 子系统名称 | 模块名称 | 输入 | 处理 | 算法描述 | 输出 |
| 数据包过滤器子系统 | 过滤器实现模块 | 过滤字符串 | 过滤字符串进行处理，生成可在socket中使用的内容 | 获得过滤器设置字段，将过滤器字段处理成相应的格式代码，应用于socket | Socket 可识别的过滤代码 |

表5.3(续表)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 子系统名称 | 模块名称 | 输入 | 处理 | 算法描述 | 输出 |
| 数据包过滤器子系统 | 过滤器规则设置模块 | — | 将过滤器界面复选框内容映射为过滤字符串 | — | 显示的字符串 |
| 过滤器提示模块 | 过滤器规则设置模块获得的内容 | 对过滤器字符串进行显示 | — | 将过滤器字符串显示在状态栏上 |

**表5.4 数据包捕获子系统**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 子系统名称 | 模块名称 | 输入 | 处理 | 算法描述 | 输出 |
| 数据包捕获子系统 | 数据包捕捉模块 | 过滤器字符串代码 | 开启数据包捕获，设置过滤器，对捕获信息数据包进行保存 | Rawsocet启用，对抓包主机进行判断，设置过滤器，存储数据包 | 保存的数据包 |
| 数据包初解包模块 | 捕获的数据包 | 对捕获的数据包进行特定字段的解析，将字段显示到表格当中 | — | 保存的数据包长度，地址信息，最高层协议等信息 |
| 数据包文件模块 | 数据包存储信息 | 写入pcap头部信息，依照其格式进行数据包的写入 | — | 数据包pcap文件 |

**表5.5 数据包信息展示子系统**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 子系统名称 | 模块名称 | 输入 | 处理 | 算法描述 | 输出 |
| 数据包信息展示子系统 | 数据包分层解包模块 | 读入存储的数据包 | 对数据包进行分层解析 | 依据相关网络协议头字段进行解码 | 在Tabs中显示各层的解析内容 |
| 数据包原始数据解码模块 | 读入存储的数据包 | 对数据包进行编码解析 | 依照utf-8格式进行数据包解码 | 在Tabs中显示原始二进制和解码内容 |

**表5.6统计绘图子系统**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 子系统名称 | 模块名称 | 输入 | 处理 | 算法描述 | 输出 |
| 统计绘图子系统 | 网络流量图模块 | 网络流量信息 | 网络流量为x轴，流量为y | — | 网络流量的时间关系图 |
| 协议统计模块 | 存储的网络数据包 | 数据包类型为x,其个数为y | — | 网络数据包统计关系图 |
| 数据包长度统计模块 | 存储的网络数据包 | 数据包个数为x，其长度为y | — | 网络数据包长度统计图 |

**表5.7 数据导出子系统**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 子系统名称 | 模块名称 | 输入 | 处理 | 算法描述 | 输出 |
| 数据导出子系统 | 本地信息导出 | 网络信息 | 对网络信息进行格式化 | — | 输出csv、Json和纯文本格式 |
| 局域网信息导出 | 局域网信息 | 对局域网记录信息格式化 | — | 输出csv、Json和纯文本格式 |
| 数据包摘要信息导出 | 存储的数据包信息 | 对存储得数据包基本解包信息整理格式化 | — | 输出csv、Json和纯文本格式 |

**表5.8 协助支援子系统**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 子系统名称 | 模块名称 | 输入 | 处理 | 算法描述 | 输出 |
| 协助支援子系统 | 协议查找模块 | 规定下的协议格式 | 对输入的协议信息进行处理，搜索出符合的数据包 | 输入字符串与协议栈字段信息查询比较 | 在列表中加载出符合规则的数据包 |
| IP地址所属查询模块 | IP地址 | 通过网络接口API获取IP相关信息 | — | 数据IP所属信息Json格式 |
| 计算机术语查询模块 | 计算机术语  （英文） | 通过网络接口API获取该术语的信息 | — | 输出该术语的相关解析 |
| 网络流量记录模块 | — | — | 以间隔1秒的密度对网络流量的上传下载速度已经数据包的收发进行获取 | 存储网路流量的速度值和数据包收发情况 |

## 5.4模块实现

网络信息子系统：

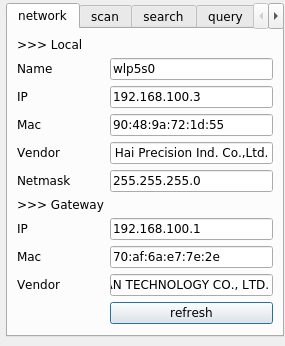


图5.6 网络信息

该控件包含了本地计算机信息以及网关信息，通过对刷新按钮(refresh)的单击可以刷新当前网络信息。

局域网扫描子系统：

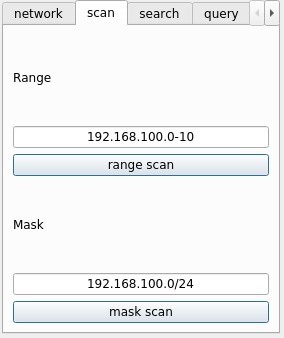


图5.7 扫描选择

局域网的扫描方式有两种，一种是范围扫描，使用“-”来表示范围。另外一种是子网掩码的扫描，使用“/”来表示后面的子网掩码。以上两个信息均是由网络信息自动计算而来，如有需要也能通过手动进行修改。

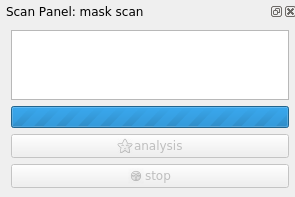


图5.8 扫描存活节点

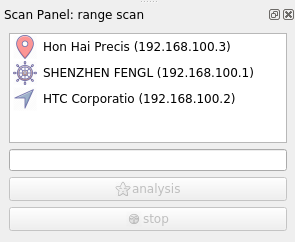


图5.9 存活节点显示

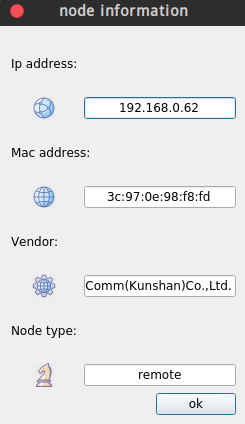


图5.10 节点详细信息

点击扫描按键(Scan)后，扫描控件将会被激活，在扫描控件中将会显示扫描所选用的方式，其进度条将以走马灯的形式告诉用户，正在进行扫描操作。扫描完成后，将会在列表空间中看到相应的活跃节点，三种不同的图标表示节点不同的类型，其中红色的“坐标”图标表示这为当前用户的主机节点，“罗盘”图标表示的是网关节点，蓝色“箭头”图标表示的是其他存活主机节点，图标后续依次是硬件厂家信息和IP地址。更方便的方式是双击需要查看的主机，这将显示出相关的节点详细信息。

数据包过滤器子系统：

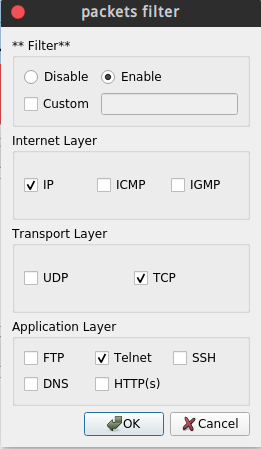


图5.11 过滤器设置

C:\Users\leslie\Desktop\forMs\figure\c6\realize\xfig7_filter_str.png

图5.12 过滤字段显示

这是过滤器操作界面，如需启动过滤器需要先对过滤器进行其用，默认情况下过滤器是关闭的(Disable)。启动过滤器后，过滤器字段有两个选择，第一个通过复选框，进行常用协议的过滤，另一种是自己手工设置过滤字段。第一方式适合初学者，第二种方式适合进阶使用。无论是那种方式对过滤器进行设置，过滤器字段只要被确定，将会在状态栏中进行显示，如图5.11。

数据包捕获子系统：

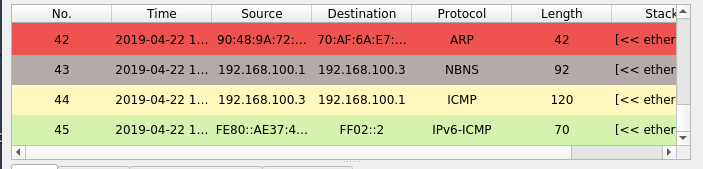


图5.13 数据包信息简要显示

数据包被捕获后，得到的数据包信息将存储到内存中，对基本信息解码后，将会呈现到表格界面当中。为了方便用户使用，对于不同的数据包进行了必要的颜色标识，以作区分。

数据包信息展示子系统：

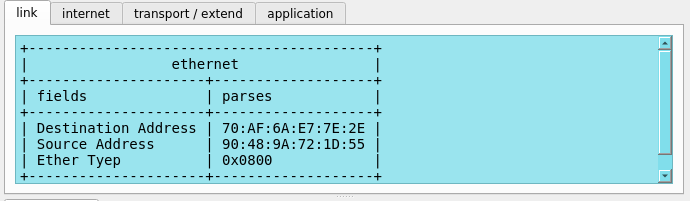


图5.14 分层数据包解析

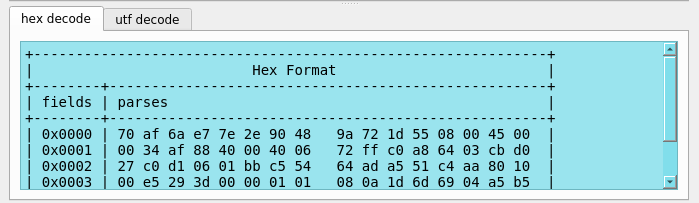


图5.15 数据包编码解析

对于数据包的解析，使用的是TCP/IP族的结构，该结构可以很好地对数据包的层次进行展示。对于二进制的解码，使用的是两个界面，独立查看。

统计绘图子系统：

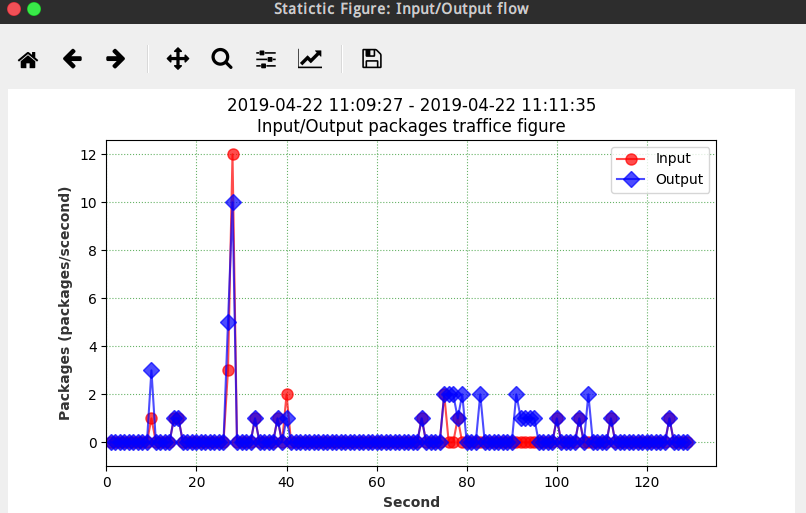


图5.16 网络流量统计图

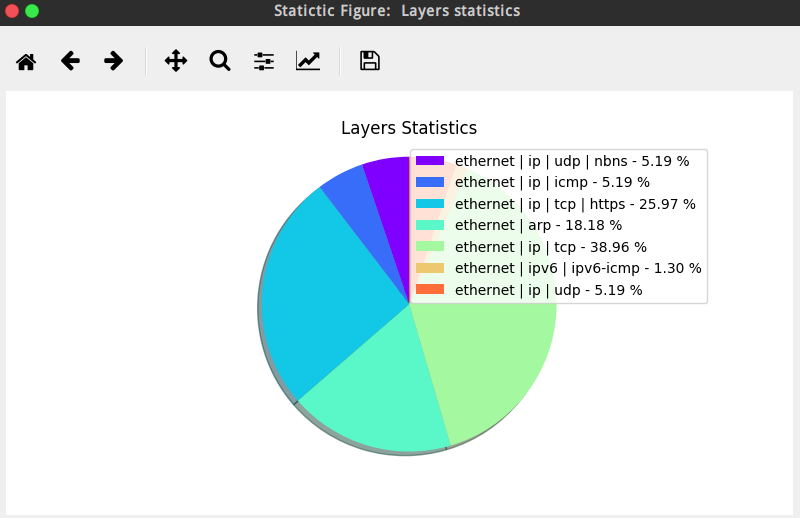


图5.17 数据包结构统计图

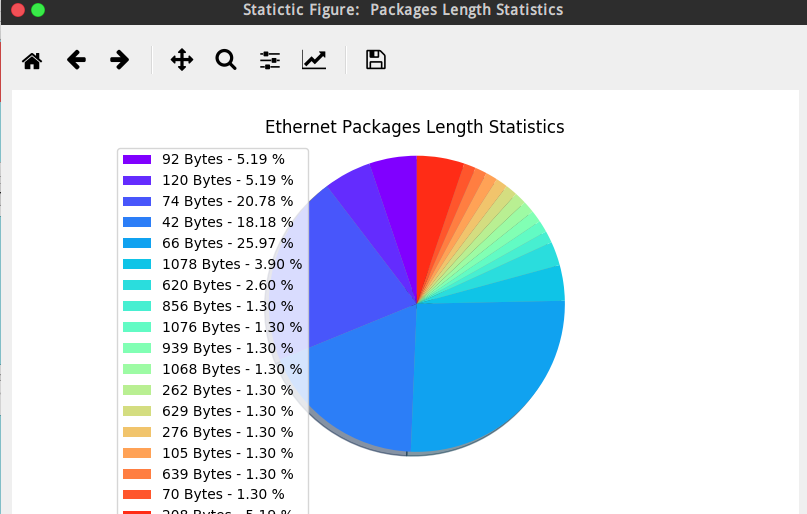


图5.18 数据包长度信息统计

统计数据图有9张，这里不过多展示，只列举了每一个大类别下的一张图。

数据导出子系统：

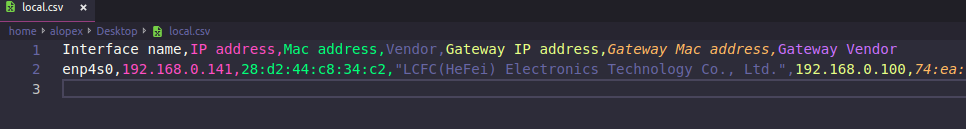


图5.19 本地网络信息csv导出

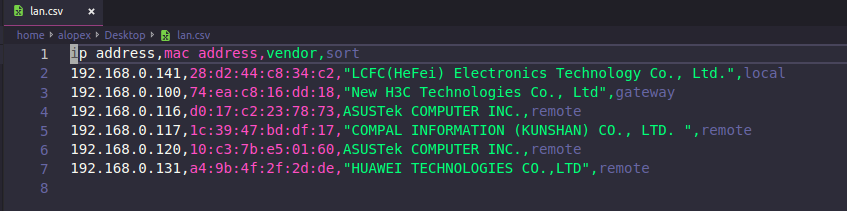


图5.20 局域网信息csv导出

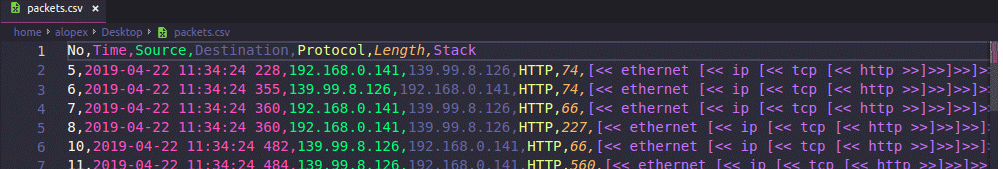


图5.21 数据包简要信息csv导出

对于数据的导出，每个数据类型有三种不同的导出格式csv、Json和纯文本格式，这里为了方便统计三种方式均使用了csv格式进行了展示。

协助支援子系统：

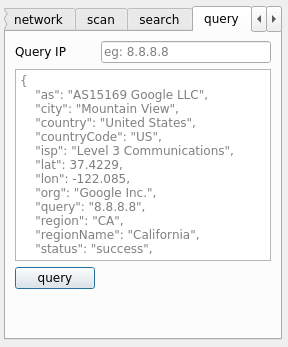


图5.22 未进行IP查询时

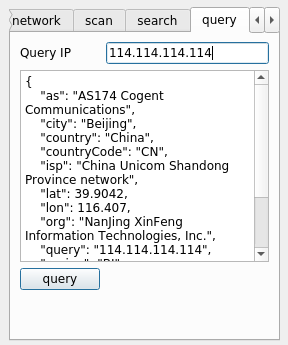


图5.23进行IP查询

对于IP地址信息归属的查询，这里提示的效果。在没有进行IP地址查询的时候，使用了PlaceHolder对将要展示的信息给出了一个demo，使用户在第一次使用时就对获得的信息有一个大致的了解。

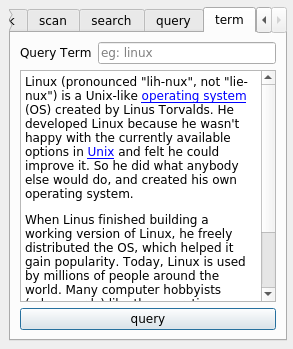


图5.24未进行专业名词查询

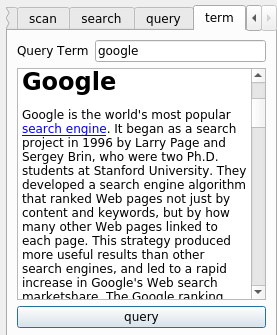


图5.25未进行专业名词查询

对于计算机专业词汇的查询，也是如此，使用上操作简单，上手门槛极低。

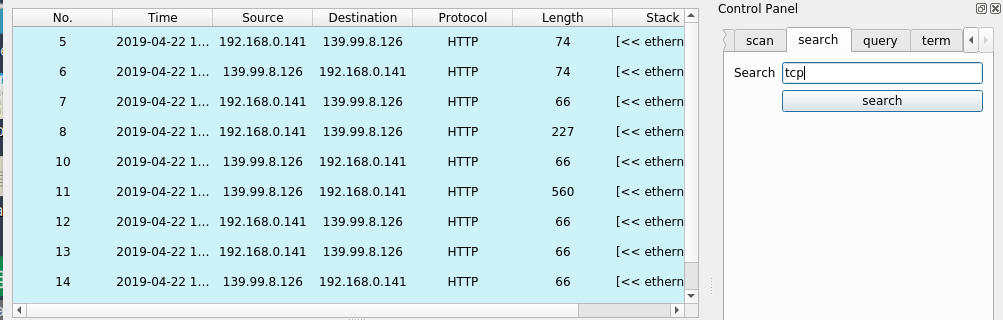


图5.26 数据包查找功能

数据包显示后，还有一个对于数据包查找的功能，通过数据包的查找可以在表格中筛选出符合条件的数据包。值得一提的是，数据包的查找有两个规则可以应用，一个是"."的嵌套协议查找相当于AND操作，一个是","的多个协议查找相当于OR操作。

C:\Users\leslie\Desktop\forMs\figure\c6\realize\xfig_23packet_status.png

图5.27网络状态查看

状态栏的网络相关信息，仅仅在抓包的时候才会启动。这些信息动态更新，通过图标的可读性来替代文字是一个很不错的选择。

本章讲述了系统的详细设计与实现。详细设计中，对系统用户接口设计草图进行了分析并在界面设计过程中不断进行完善和改进，对各个模块的输入、输出、处理等信息的表格化显示，阐述清楚每一个模块处理的工作以及其使用的算法。在模块实现中，分解了软件中每个模块，进行独立的讲解，与子系统设计进行呼应。

# 第6章 性能分析

## 6.1 测试环境

### 6.1.1测试概述

本次系统测试主机系统为Linux(Ubuntu 16.04)，宿主机系统也是(Ubuntu16.04)下进行，虚拟机管理器(Virtual Machine Monitor)选择的是VMM(Virtual Machine Manager)，其版本号为1.4.2。

### 6.1.2主机环境

主机的基本情况：

**表6.1 主机基本信息表**

|  |  |
| --- | --- |
| 条目名称 | 条目信息 |
| Hostname | alopex@alpha |
| OS | Ubuntu 16.04 xenial |
| Kernel | x86\_64 Linux 4.15.0-47-generic |
| Shell | bash 4.3.48 |
| Resolution | 1366x768 |
| DE | Unity 7.4.5 |
| WM | Compiz |
| Font | WenQuanYi Micro Hei 11 |
| CPU | Intel Core i5-4210M CPU @ 3.2GHz |
| GPU | GeForce 840M |
| RAM | 1762MiB / 6823MiB |

主机详细情况：

**表6.2 主机详细信息表**

|  |  |  |
| --- | --- | --- |
| 硬件/系统 | 条目 | 信息 |
| System | Host | alpha |
| Kernel | 4.15.0-47-generic x86\_64 (64 bit) |
| Desktop | Unity 7.4.5 |
| Distro | Ubuntu 16.04 xenial |
| Machine | System | LENOVO (portable) |
| product | 20C5A081CD |
| v | ThinkPad Edge E440 |
| Mobo | LENOVO |
| model | 20C5A081CD |
| CPU | - | Dual core Intel Core i5-4210M |
| cache | 3072 KB |
| clock speeds | max: 3200 MHz |
| Graphics | Card-1 | Intel 4th Gen Core Processor Integrated Graphics Controller |
| Card-2 | NVIDIA GM108M [GeForce 840M] |
| Network | Card-1 | Realtek RTL8111/8168/8411 PCI Express Gigabit Ethernet Controller |
| Driver | r8169 |
| Card-2 | Realtek RTL8723BE PCIe Wireless Network Adapter driver: rtl8723be |
| Driver | rtl8723be |
| Drives | HDD Total Size: | 1000.2GB(11.1% used) |

### 6.1.3宿主机信息

宿主机为新安装主机软件包安装如下：

**表6.3 软件包安装表**

|  |  |
| --- | --- |
| 包名 | 安装目的 |
| inxi | 产生主机的硬件信息 |
| openssh-server | 便于对命令结果拷贝工作 |
| screenfetch | 产生基本的系统信息 |
| python3-pip | Python软件包管理 |
| python3-tk | Tk图形接口开发工具 |
| python3-venv | 产生虚拟环境 |

宿主机的基本情况：

**表6.4 宿主机基本信息表**

|  |  |
| --- | --- |
| 条目名称 | 条目信息 |
| Hostname | sprucetest@sprucetest |
| OS | Ubuntu 16.04 xenial |
| Kernel | x86\_64 Linux 4.8.0-36-generic |
| Shell | bash 4.3.46 |
| Resolution | 1366x768 |
| WM | Compiz |
| Font | WenQuanYi Micro Hei 11 |
| CPU | 2x Westmere E56xx/L56xx/X56xx (IBRS update) @ 2.594GHz |
| RAM | 762MiB / 1999MiB |

宿主机详细情况：

**表6.5 宿主机详细信息表**

|  |  |  |
| --- | --- | --- |
| 硬件/系统 | 条目 | 信息 |
| System | Host | sprucetest |
| Kernel | 4.8.0-36-generic x86\_64 (64 bit) |
| Desktop | N/A |
| Distro | Ubuntu 16.04 xenial |
| Machine | System | QEMU |
| product | Standard PC (i440FX + PIIX 1996) |
| v | pc-i440fx-xenial |
| Mobo | N/A |
| model | N/A |
| CPU | - | 2 Single core Westmere E56xx/L56xx/X56xx (IBRS update)s |
| cache | 8192 KB |
| clock speeds | max: 2593 MHz |
| Graphics | Card | Red Hat QXL paravirtual graphic card |
| Network | Card | Realtek RTL-8100/8101L/8139 PCI Fast Ethernet Adapter driver |
| Driver | 8139cp |
| Drives | HDD Total Size: | 21.5GB (29.2% used) |

## 6.2软件测试

本次测试使用的Python解析器为CPython3，版本为3.5.2。CPython编译器为GCC，其版本为5.4.0 20160609。spruce安装环境将会在虚拟环境下进行，虚拟环境提供工具为venv。安装方式通过pip工具直接下载位于pypi上的软件包spruce-sniffer，进行本地安装。

安装过程：

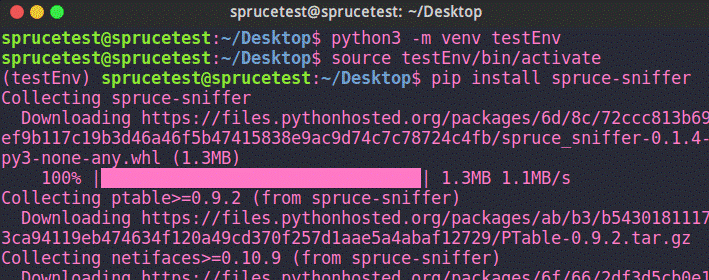


图6.1 软件安装进行图

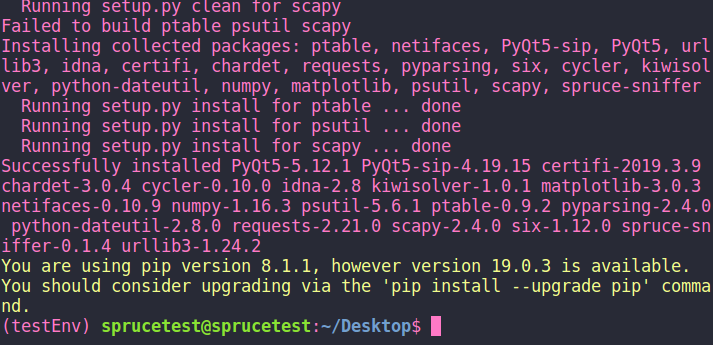


图6.2 软件安装完成图

软件运行方式与展示：

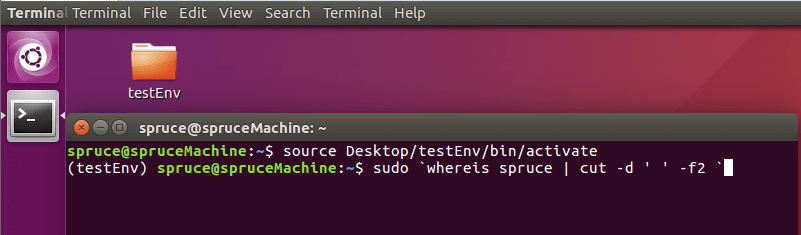


图6.3 软件执行命令

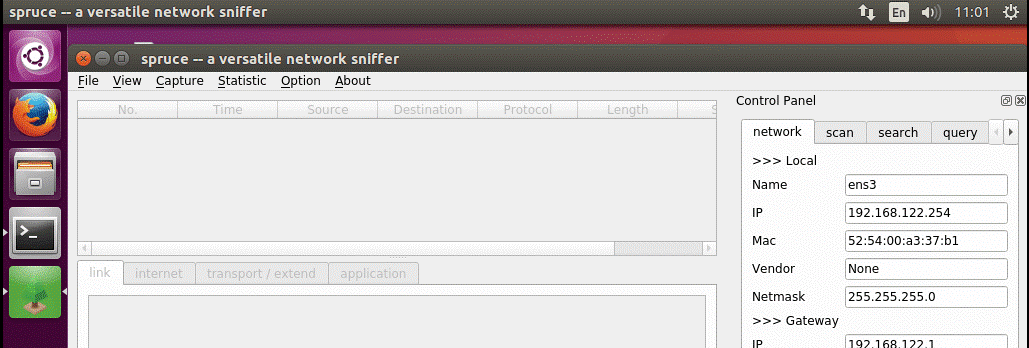


图6.4 软件运行界面

以下将对软件三种状态下进行分析。空转为软件启动但不做任何操作的情况；常规为软件在普通场景经下进行抓包的情况如浏览网页；高负载是在极端环境下如下载大文件时的表现情况。

**表6.6 性能分析表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参考信息 | 尚未启动 | 空转 | 常规 | 高负载 |
| CPU① | 0.23,0.20,0.19 | 0.32,0.20,0.19 | 0.96,0.90,0.59 | 1.20,0.98,0.76 |
| 内存② | 563104(27.5%) | 679160(33.2%) | 737076(36.0%) | 900268(44.0%) |
| 网速峰值 | 0.0KB/s | 0.0KB/s | 392KB/s | 885KB/s |
| 软件流畅度③ | N/A | +++ | +++ | + |
| 数据包捕获个数 | N/A | - | 650 | 26664 |
| 数据包捕获比值 | N/A | - | 99.54% | 66.67% |
| 是否出现无响应 | N/A | 无 | 无 | 无 |

①CPU数据来自uptime 命令

②内存数据来自free 命令

③“+”越多表示越流畅

以上表格可以看出，软件的启动对内存的消耗为百分之五，对CPU消耗并不高。当软件在常规情况下运行时，总体表现良好，内存的占用有所提升，但丢包率并不会太严重，软件使用的流程度依然可观。当软件在高负载情况下运行时，可明显看出，由于数据包存储于内存中，此时对存储的消耗大幅提升，软件的流畅度有了明显的下滑，数据包捕获工作依然能完成，但是其捕获比例呈现出明显的丢包情况。

本章对开发的嗅探器软件进行了性能测试工作。在性能测试前对信系统和与测试环境进行了了硬件信息的详细叙述，对嗅探器软件安装前必要的依赖包进行了解释，务求能在最少改变系统环境的情况下对软件进行安装于测试。对于得出的测试数据，进行了概括行的分析。

# 总结与展望

该应用的优势有以下三个方面，良好的图形交互、安装门槛低以及可扩展性高。

图形界面是这个应用的一大亮点，对于嗅探器这样的应用，良好的图形界面能够让使用着更加得心应手，减少疲劳感。图形界面的设计上，开发者对排版进行了数次的修改，为求令用户有一个自由度更高的界面，在控件选用上下了一番功夫。使得整个应用界面的各个模块之间可以自由活动，每个用户都能依照自己的兴趣习惯来调整界面的布局，无需配置文件只需要简单的鼠标拉伸点击即可。在界面的颜色搭配上，为了令用户在使用是保持一致性，在图标的选择上特意选择了同一位设计师的同一系列图标库，做到在图标的风格上保持一致。

软件的安装是每个软件分发中必须经历的。应用提供了两套软件安装方法，一个方法是利用源代码，进行冻结代码安装，另一个方法是使用CPython的包管理工具pip进行安装。这两种方式适用于有不同需求的用户，但相同的是使用方式都极为简单。该应用对于系统的硬件要求很低，开发者使用的电脑于2014年的购置，其配置是当时常见的i5芯片和4G内存。即使在今天2019年，运行起该软件甚至是分配虚拟机来运行该软件，也没有感觉到明显的卡顿现象。

软件的主要编码由Python提供，软件在对数据包的解析中，没有选用时下Python流行的Scapy库，而是自己编写了解析的文件。在软件的版权上，遵照MIT license执行，任何人都有权利对软件进行修改与重分发。赋予了后续开发者，学习、交流以及完善功能的自由。

该应用的不足主要表现在以下两个方面，平台的兼容性和功能有待完善。

如今的个人计算机(Personal Computer)的主要系统为Windows，对于软件无法跨平台这件事，无疑是软件的短板，严重削减了软件的受众群体。然而因为在开发过程中，其中一些数据的获取功能依赖于Linux某些文件，因此想要做到跨平台需要不少的路要走，对于平台的无法兼容，毫无疑问是一个不小的遗憾。

在高并发中，可以发现软件对于网速过高时，其抓包率并不理想，这一方面是因为软件在数据结构设计方面经验不足，导致数据量过大，对内存资源的消耗过高，另外一个方面是因为选定了使用CPython解释器，其中的GIL(Global Interpreter Lock)也是其中的阻碍因素。另外对于数据包的解释上，网络协议是一个非常庞大的家族，因此该应用只针对常用协议进行了解析，某些协议的解析颗粒度并未达到工业水平。

软件的开发，到了分发只能算完成了一半，后面的一半需要由维护与不断的升级来体现软件后续的生命力。

在软件的使用上，我们的应用交付目前已经完成。但是正如上文所描述的，该软件依然存在这不小的问题，这里将提供一些可以参照的解决思路。

在跨平台功能上，其解决思路是改变目前数据的收集方式，让数据的收集去系统化，让其尽量和系统无关，要做到这点可能刚开始有点困难，但是为了后续的长远发展，这是必须要实现的。在高并发中，依然希望能够稳定抓包，一个简单的解决方案是选择使用无GIL的python解释器，例如pypy，这是Python的一个未来，但是由于一些主要开发包都在pypi上，并为移植到pypy的上，这使得这个简单的方案成无限延期的等待。另外一种方式，是对如今数据存储中得数据结构进行优化，或者是对GIL进行有力的解锁。

# 致 谢

毕业论文是大学生涯必经的一个阶段，是必不可少得锻炼与磨练。从前，总渴望着大学的毕业典礼尽快来临，那里有鲜花有掌声，有脸上洋溢着青春的娇气，有欢欣鼓舞的喝彩。

如今这一刻慢慢逼近，倒有一种不舍油然而生。

毕业设计完成之际，将真诚的感谢送给我得导师张先勇教授。在毕设制作期间，由于缺乏开发经验，我遇到不少棘手的问题，它们的产生使得项目的开发变得困难重重。得益于张老师无意间的点拨，问题随之便能迎刃而解。假若说毕业论文是挫折是一道道坎，那么张老师的点拨就是一个有力的指导。时而告诉你，这里坑不浅你必须造梯子过去，时而告诉你这坑不深，涉水过去也未尝不可。

另外感谢班上每一位同学，或许我们在一起得时间不算太长。但是在他/她们身上，我收获了很多精彩与感动得瞬间，这些经历足以令我深刻而难忘。

# 参考文献

[1] 中国互联网络信息中心.中国互联网络发展状况统计报告[R/OL](2019-02-28)[2019-04-15].www.cac.gov.cn/wxb\_pdf/0228043.pdf.

[2] 于志军.互联网对中国人生活方式的改变及影响[J].计算机与网络,2018,44(22):9-10.

[3] 刘维嘉.“互联网+”环境下高职《计算机网络技术》课程教学改革与研究[J].福建电脑,2017,33(02):57-58.

[4] 柯辉.计算机网络技术专业卓越人才需求调查分析[J].电脑知识与技术,2015,11(33):83-84.

[5] 刘春梅.学好网络技术的三大技巧[J].计算机与网络,2018,44(18):42-43.

[6] Kevin J. Connolly . Law of Internet Security and Privacy[M]. Aspen Publishers. 2003:131.

[7] 于鹏飞,孙春静,薄红岩,彭斌.基于windows平台的网络嗅探器系统的设计与实现[J].黑龙江科技信息,2017(06):179.

[8]A. Dabir, A. Matrawy.Bottleneck Analysis of Traffic Monitoring Using Wireshark[R].4th International Conference on Innovations in Information Technology, 2007, IEEE Innovations.2007:158-162.

[9] 谢希仁.计算机网络[M].(7).电子工业出版社.2017:29-30.

[10] W.Richard Stevens. TCP/IP详解(卷1):协议(英文版) [M].(2).人民邮电出版社. 2016:2-28.

[11]刘大成.Python数据可视化之matplotlib实践[M].(1).电子工业出版社.2018:3-16.

[12] 王硕, 孙洋洋. PyQt5快速开发与实战[M].(1). 电子工业出版社.2017:317-373.