

面向网络安全审计的轻量级多用途安全网关设计与开发

Design and Development of Lightweight
Multipurpose Security Gateway for Network
Security Audit

摘要

本研究基于开源的嵌入式系统，设计了一款轻量级的多用途安全网关设备，能够及时、准确的发现并记录可疑恶意行为，实时报警，以提供安全可靠的网络环境。

首先，利用树莓派3代B型板载的有线网卡和无线网卡的硬件支持，刷以LEDE开源固件，使树莓派实现了市面上大多数无线路由器的功能，经过二次开发、处理，使其兼具网络代理、广告过滤、离线下载、文件共享等特色功能。

其次，使用libpcap工具包编写了流量统计分析代码，当网络不通畅时能够及时发现局域网中恶意占用网络带宽用户并实施处理。

最后，使用软镜像的方式将流量封装成TZSP数据包，发送给流量分析机器，能够对敏感关键词及主流应用进行针对性识别，并通过邮件及时推送给网络管理人员进行告警。

关键词：树莓派 LEDE 安全网关 流量分析

Abstract

Based on the open source embedded system, this study designed a lightweight multi-purpose security gateway device that can detect and record suspicious malicious behaviors in a timely and provide real-time alarms to provide a secure and reliable network environment.

First of all, using the hardware support that Raspberry Pi 3rd generation B has wired network adapter and wireless network adapter on its board, brush to LEDE open source firmware, so that the Raspberry Pi achieves the function of most of the wireless routers on the market, after secondary development, processing, It combines features such as web agency, ad filtering, off-line downloading, file sharing and more.

Secondly, using the libpcap toolkit to compile traffic statistics analysis code, when the network is not smooth, it can discover the malicious users who occupy network bandwidth in the LAN and perform processing.

Finally, using the soft mirroring way to encapsulate traffic into TZSP packets and send it to the traffic analysis machine, which can identify targeted keywords and mainstream applications and send them to the network administrators for alerts in a timely manner.

Keyword: Raspberry Pi LEDE Security gateway Traffic Analysis

目录

第一章 绪论.....	1
1.1 项目开发背景及意义.....	1
1.2 路由器发展现状研究.....	1
1.2.1 路由器市场发展现状.....	1
1.2.2 OpenWrt 和 LEDE 简介.....	3
1.3 设计报告的组织结构.....	3
第二章 树莓派、LEDE 搭建基础路由器平台.....	5
2.1 所需软硬件资源简介.....	5
2.1.1 硬件.....	5
2.1.2 软件.....	7
2.2 LEDE 固件安装.....	7
2.3 LEDE 基本配置.....	8
2.3.1 网络设置.....	8
2.3.2 防火墙设置.....	10
2.3.3 界面设置.....	10
第三章 LEDE 扩展功能模块设计.....	12
3.1 科学上网模块.....	12
3.1.1 搭建交叉编译环境.....	12
3.1.2 编译软件包.....	13
3.1.3 客户端模块安装.....	15
3.1.4 服务端模块安装.....	16
3.2 Aria2 离线下载.....	18
3.2.1 Aria2 简介.....	18
3.2.2 LEDE 安装 Aria2 和 WebUI-Aria2.....	18
3.2.3 Aria2 使用.....	19
3.3 文件共享.....	20
3.3.1 Samba 文件共享.....	20
3.3.2 FTP 服务器.....	22
3.4 Ngrok 内网穿透.....	24
3.4.1 ngrok 简介.....	24
3.4.2 LEDE 中搭建 ngrok.....	24
3.4.3 外网连接.....	25
第四章 LEDE 搭载 Debian 实现网络安全审计.....	27
4.1 SD 卡扩容及分区挂载.....	27
4.1.1 安装磁盘操作必要的软件包.....	27
4.1.2 磁盘分区.....	27
4.1.3 分区格式化及分区挂载.....	29
4.2 LEDE 中使用 debootstrap 安装 Debian.....	29
4.2.1 Debootstrap、Debian 简介.....	29
4.2.2 Debian 系统安装.....	30
4.3 配置 Debian 文件系统.....	31
4.4 基于 libpcap 编写流量统计代码.....	33

4.4.1 libpcap 功能函数简介.....	33
4.4.2 通信流量采集及元信息留存模块实现.....	33
4.4.3 通信流量日志信息统计分析模块实现.....	36
4.4.4 编写 makefile 和 shell 脚本实现模块整合.....	37
4.5 代码运行及效果展示.....	38
第五章 局域网内主机行为监管.....	39
5.1 系统架构设计.....	39
5.2 软件安装配置.....	40
5.2.1 路由器端软件安装及配置.....	40
5.2.2 监控主机端软件安装及配置.....	41
5.3 网络连接实时封堵.....	42
5.4 网页关键词及违规软件告警.....	43
5.4.1 配置 SMTP 服务器.....	43
5.4.2 设置网络关键词告警.....	44
5.4.3 设置违规软件告警.....	45
总 结.....	47
参考文献.....	48

第一章 绪 论

1.1 项目开发背景及意义

互联网技术经过二十多年的飞速发展，网络带宽不断提速，网络流量数据急剧增加。与此同时，越来越多的新式网络协议兴起，各种新型的网络服务层出不穷，其中网络安全面临的风险不断提高，针对计算机系统、局域网络及智能互联设备等的攻击事件频繁发生，使用的攻击手段也越来越复杂化、智能化和多样化，给整个网络系统带来了各式各样的安全问题。安全审计系统通过对用户访问系统和网络的行为进行记录和分析，挖掘出其中存在的违规行为并及时采取解决措施，以此来保障整个局域网络的安全^[1]。而在家庭、中小型企业等的网络场景下，使用网络防火墙、入侵检测系统等重量级安全系统受到诸多不便因素的影响，与此同时，各种开源的嵌入式系统、固件、硬件，例如树莓派、OpenWrt 等不断涌现，为定制化的多功能、多目标硬件设备提供了基础，因此研究设计与开发具备网络安全审计功能的轻量级多用途安全网关设备具有重要的现实意义及使用价值。

本研究面向多种中小型网络环境，研究基于网络行为分析和网络安全审计技术，并融合多种网络功能的硬件设备，能够及时、准确的发现并记录可疑恶意行为，实时报警，以提供安全可靠的网络环境。

1.2 路由器发展现状研究

1.2.1 路由器市场发展现状

近几年随着智能无线设备的快速发展，如智能手机、轻薄笔记本、平板电脑、智能联网家居设备等，对无线 WiFi 信号的需求量也越来越大，滋生了巨大的无线路由器设备市场，众多厂商纷纷加入进来以从中谋取利润点^[2]。通过解读及分析运营商世界网在 2018 年初发布的《2017 年路由器领域分析报告》，能够清楚地看到各个无线路由器品牌的市场占有率以及大众对各个无线路由器品牌的关注度^[3]。

在市场占有率方面，大多数的传统路由器厂商优势较为明显，位居榜首的是 TP-Link 品牌路由器，市场占有率为 20.7%，小米、斐讯、腾达、华硕分别位居

第二名至第五名^[3]。很多小企业生产的路由器也逐渐受到用户的青睐，其总市场占有率达到 16.2%，可见其发展势头迅猛。

2017 年各个路由器品牌市场占有率排行榜如图 1-1 所示：

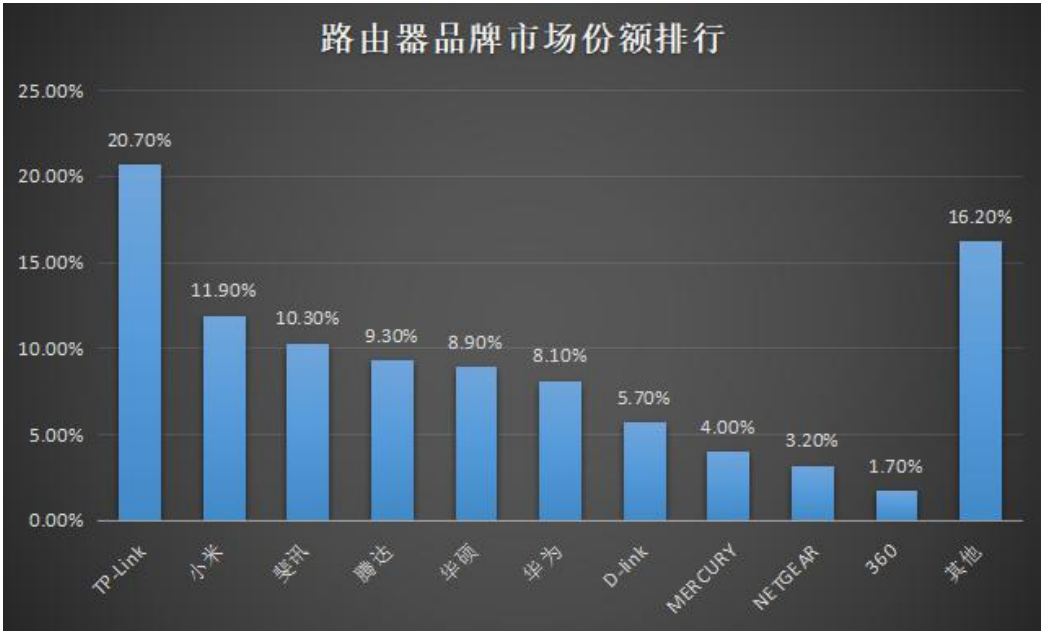


图 1-1 路由器品牌市场份额排行

从家用路由器品牌关注度方面看，TP-Link 的品牌关注度排名仍然是第一，但近几年主要发展电竞市场中游戏路由器及高端路由器的华硕品牌排名跃居第二^[3]。华为、小米、360 等一些手机生产厂商推出的路由器，依靠其产品外表美观、智能小巧、更新速度快等优势，大众对其关注度普遍较高。

从中小企业路由器品牌关注度方面看，中小企业对中高端路由器品牌的关注度较高，华硕品牌关注度排名第一，其次是 TP-Link、NETGEAR、斐讯、华为^[3]。小型创业公司、创微企业、购物商场、餐厅、咖啡厅、图书馆、酒店等场合对无线路由器的需求巨大^[2]，而他们最看重的点是网络安全，相信在接下来的市场中偏重网络安全的无线路由器将会热卖。

家用路由器和中小企业路由器品牌关注度排名如表 1-1 所示：

表 1-1 家用、中小企业路由器品牌关注度排名

排名	家用路由器品牌关注度排名	中小企业路由器品牌关注度排名
1	TP-Link	华硕
2	华硕	TP-Link
3	斐讯	NETGEAR

4	腾达	斐讯
5	小米	华为
6	华为	腾达
7	荣耀	荣耀
8	360	极路由
9	MERCURY	360
10	NETGEAR	D-Link

通过上述分析可以得出未来路由器的发展趋势，即朝着更加智能化、美观化方向发展^[3]，同时将会更加注重网络安全。在这些方面，各个新起的路由器厂商更具优势，传统路由器品牌正受到越来越多的挑战。

1.2.2 OpenWrt 和 LEDE 简介

OpenWrt 是一个针对嵌入式设备的 Linux 操作系统^[4]，于 2004 年 1 月推出第一个版本，其前身是 Linksys WRT54G。OpenWrt 是一个可写的操作系统，官方提供了 3000 多个软件包，用户可以根据自己的需求在上面添加各种软件包以实现定制化的目的。OpenWrt 支持各种架构的处理器，如 ARM、MIPS、X86、PowerPC，开发者可以轻松地将自己开发的项目移植到 OpenWrt 下^[5]，正是居于此优势，OpenWrt 项目得到了迅猛发展。

由于历史原因，OpenWrt 项目/社区中存在的一些问题迟迟得不到解决，例如项目中缺少沟通、透明和协调，尽管核心开发者处于历史低点但却没有引进新人的流程等等^[6]。在 2016 年 5 月初，用户社区发布了另一个版本的 OpenWrt，起名 LEDE (Linux Embedded Development Environment)，更加注重协作和分权、测试和定期编译等。

出于长远发展的考虑，在 2018 年初，官方宣布合并 OpenWrt 和 LEDE 两个项目以提供更加稳定的服务，同时更加注重 bug 修改的速度与安全问题。合并后的项目仍然使用 OpenWRT 命名，但整个项目将按照 LEDE 制定的规范进行管理^[7]。

1.3 设计报告的组织结构

本文共分为五章：

第一章，介绍项目开发背景及意义，以及国内市场主流消费级路由器的发展现状和未来发展趋势，对开源的路由固件 OpenWrt 和 LEDE 做一个简短的介绍。

第二章，介绍如何在树莓派 3B 上使用 LEDE 固件搭建基本路由器环境，使其具有市面上大多数路由器的功能。

第三章，介绍 LEDE 上主要功能模块的设计，包括科学上网模块、Aria2 离线下载、Samba 文件共享、FTP 服务、KoolProxy 广告过滤等，使其实现智能化。

第四章，介绍使用 libpcap 工具包编码实现原始流量捕获、通信元信息留存、通信数量和关系统计，异常发现等。同时介绍如何在 LEDE 环境中使用 debootstrap 安装 debian 文件系统，使其代码能够成功运行且更具移植性。

第五章，介绍如何通过 port-mirroring 软镜像的方式将经过路由器的数据打包成 TZSP 数据包发送到电脑端，在电脑端使用 WFilter 软件实现对敏感关键词、主流应用等的实时监控与告警。

第二章 树莓派、LEDE 搭建基础路由器平台

2.1 所需软硬件资源简介

2.1.1 硬件

(1) 树莓派 3B 一个

树莓派(Raspberry Pi)，是一款针对电脑业余爱好者、教师、学生以及小型企业等用户的基于 Linux 的单片机电脑，体积仅信用卡大小，搭载 ARM 架构处理器，运算性能和智能手机相仿^[8]。

树莓派 3B 实物如图 2-1 所示：

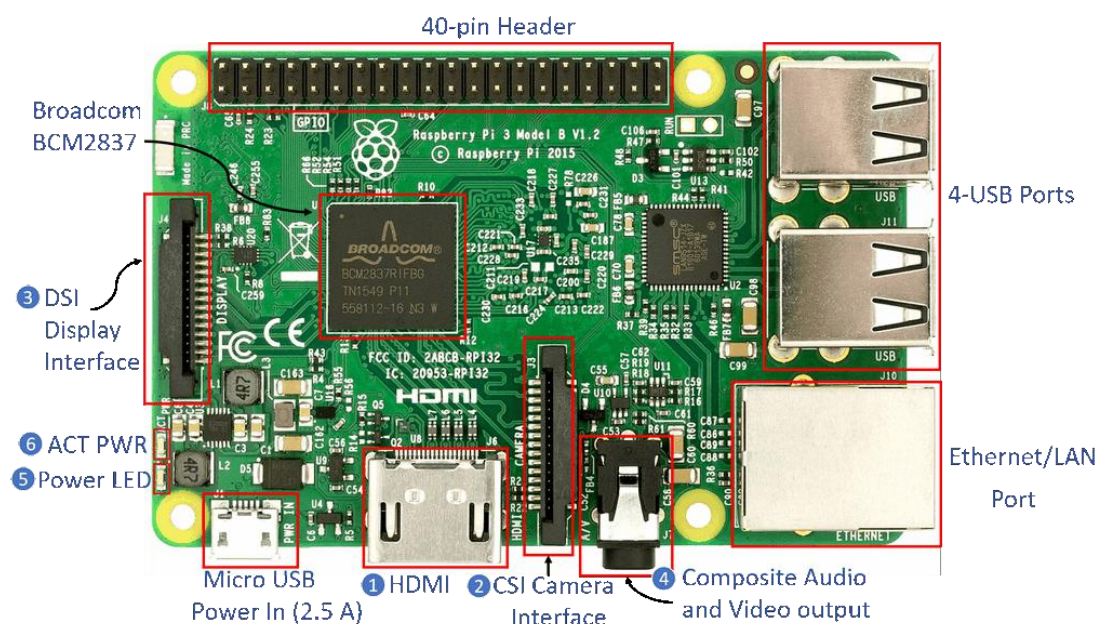


图 2-1 树莓派 3B 实物图

树莓派目前存在两种型号，A 型和 B 型，两种型号差别不大，只是性能上的些许区别，B 型要比 A 型更强悍一些。自从 2011 年最初版发布之后，在经历了 5 年的发展和设计迭代后，官方在 2016 年 2 月推出了一款新品树莓派 3^[9]。树莓派 3 在性能有很大的提升，主要体现在更高的处理速度（采用基于 Cortex-A53 的博通 BCM2837，64 位处理器）和更高的互联性（集成蓝牙 4.0 和 WiFi）^[9]。

树莓派各版本差异如表 2-1 所示：

表 2-1 树莓派各版本差异对照表

产品	Raspnberry Pi B	Raspnberry Pi B+	Raspnberry Pi A+	Raspnberry Pi 2 Model B	Raspnberry Pi 3 Model B
发布时间	2011-12	2014-07	2014-11	2015-02	2016-02
SoC	BCM2835	BCM2835	BCM2835	BCM2836	BCM2837
CPU	ARM1176JZF-S 核心 700MHz 单核	ARM1176JZF-S 核心 700MHz 单核	ARM1176JZF-S 核心 700MHz 单核	ARM Cortex-A7 900MHz 四核	ARM Cortex-A53 1.2GHz 四核
GPU	Brocom VideoCore IV, OpenGL ES 2.0, 1080p 30 h.264/MPEG-4 AVC 高清解码器				
RAM	512MB	512MB	256MB	1GB	1GB
USB 接口	两个 USB2.0	四个 USB2.0	一个 USB2.0	四个 USB2.0	四个 USB2.0
视频接口	RCA 视频接口输出, 支持 PAL 和 NTSC 制式, 支持 HDMI(1.3 和 1.4), 分辨率为 640*350 至 1920*1200 支持 PAL 和 NTSC 制式 ^[10]				
音频接口	3.5mm 插孔, HDMI(高清晰度多音频/视频接口)				
SD 卡接口	标准 SD 卡接口	Micro SD 卡接口			
网络接口	10/100 以太网接口 (RJ45 接口)	10/100 以太网接口 (RJ45 接口)		10/100 以太网接口 (RJ45 接口)	10/100 以太网接口 (RJ45 接口), 内置 WIFI 和蓝牙
GPIO 接口	26PIN	40PIN	40PIN	40PIN	40PIN
额定功率	700 毫安(约 3.5W)	600 毫安(约 3.0W)	\	1000 毫安(约 5.0W)	\
电源接口	MicroUSB 5V				
尺寸	85.60*53.98 mm	85*56 mm	65*56 mm	85*56 mm	85*56 mm

(2) 网线一根

(3) 内存卡一张

因为树莓派中的系统是刷入在内存卡中的, 所以内存卡的存取速度对性能也有一定的影响, 在 SD 卡的选取上尽量选取一些读写速度快的。SD 卡上的 U, C 标识, 是两种不同总线模式的最低写入速度。UHS 速度有 U1、U3 两种, CLASS 标识有 C2、C4、C6、C10 四种^[11]。具体的读写速度数据如表 2-2 所示:

表 2-2 不同标识 SD 卡读写速度

	标志	串列数据最低写入速度	SD 总线模式
UHS 速度等级	U3	30MB/S	UHS-II
	U1	10MB/S	UHS-I

Speed Class	CLASS10	10MB/S	高速 (HS)
	CLASS6	6MB/S	普通速度 (NS)
	CLASS4	4MB/S	
	CLASS2	2MB/S	

(4) USB 读卡器一个

2.1.2 软件

(1) SD Card Formatter

下载地址: https://www.sdcard.org/chs/downloads/formatter_4/index.html

大多数的 SD 卡格式化可以通过电脑系统内置的格式化功能来实现,但是有些厂家生产的 SD 对格式化的要求比较高,需要使用专门的格式化软件才能够对其进行完整的格式化操作,可以使用 Formatter 来安全可靠地对 SD 卡进行格式化^[12]。

(2) Win32DiskImager

下载地址: <https://sourceforge.net/projects/win32diskimager/>

使用该软件可以往内存卡中刷入 LEDE 固件,同时,当我们在原始固件上进行二次开发之后,也可以使用该软件从 SD 卡中导出属于自己的固件镜像,方便共享。

(3) LEDE

下载地址: https://lede-project.org/toh/views/toh_fwdownload

可以根据自己的硬件在此页中找到适合自己的固件系统。

2.2 LEDE 固件安装

2.2.1 SD 卡格式化

选择好要格式化的 SD 卡盘符,有快速格式化和重写格式化两种模式,一般选择速度较快的快速格式化即可。

2.2.2 LEDE 固件烧录

烧录过程如图 2-2 所示

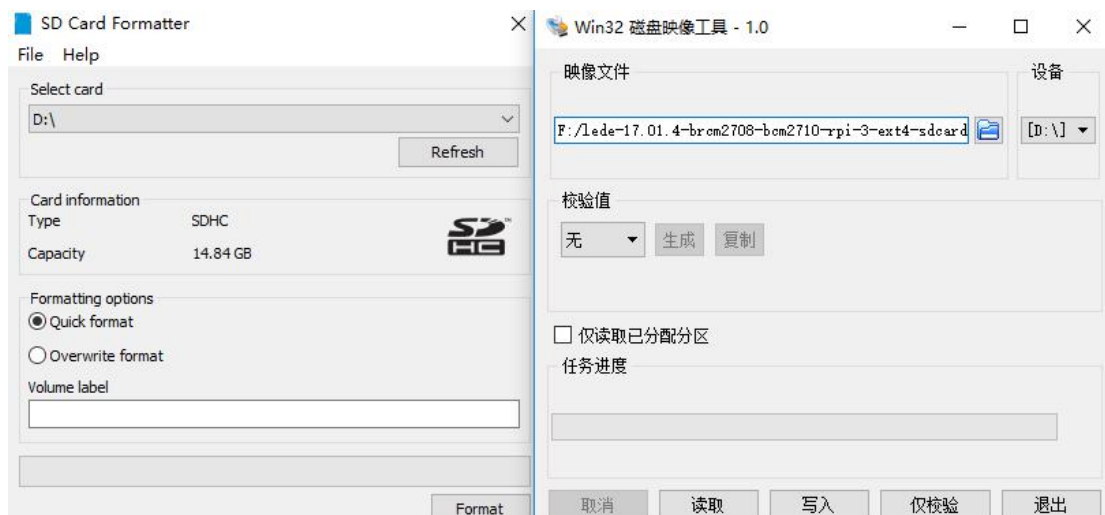


图 2-2 SD 卡格式化和镜像写入

2.3 LEDE 基本配置

2.3.1 网络设置

将之前准备的网线一端连接树莓派，另一端连接电脑，在电脑端使用 ssh 连接 LEDE，初始网址为 192.168.1.1，密码为空。为了安全起见，在首次登陆时最好设置管理员密码，方便以后在浏览器页面安全操作。具体操作如图 2-3 所示。

```
$ ssh root@192.168.1.1
The authenticity of host '192.168.1.1 (192.168.1.1)' can't be established.
RSA key fingerprint is SHA256:syBSedFDv1D5602k8Z9w0QP4x68K3+svRBjiPFQL+pg.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.1' (RSA) to the list of known hosts.

BusyBox v1.25.1 () built-in shell (ash)

  LE  LE
  DE  DE
  LE  LE
  DE  DE

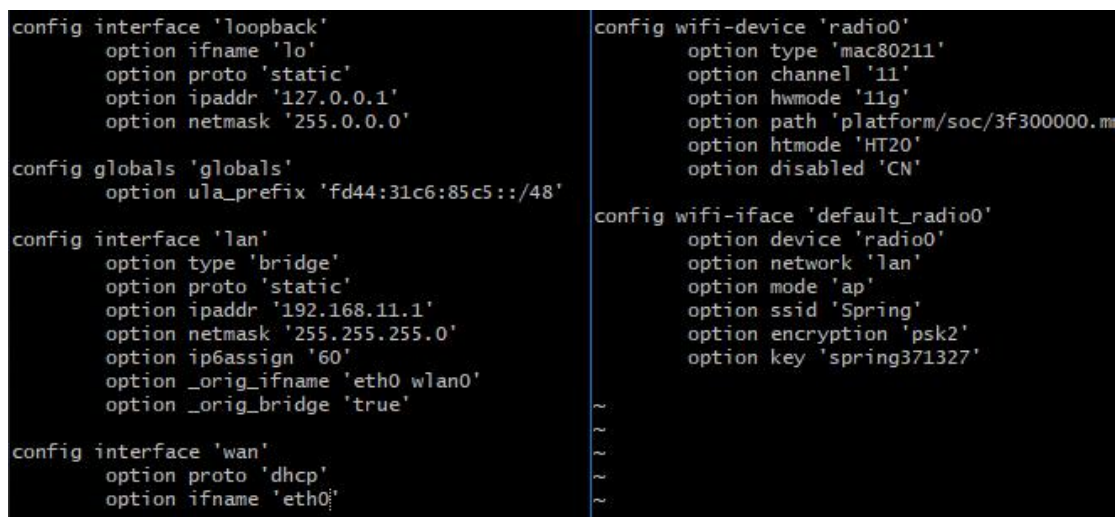
Reboot (17.01.4, r3560-79f57e422d)

===== WARNING! =====
There is no root password defined on this device!
Use the "passwd" command to set up a new password
in order to prevent unauthorized SSH logins.
-----
root@LEDE:~# passwd
Changing password for root
New password:
Bad password: too weak
Retype password:
passwd: password for root changed by root
root@LEDE:~#
```

图 2-3 登陆 LEDE 并设置密码

修改/etc/config 目录下的 wireless 配置文件和 network 配置文件,修改这两个文件的目的是开启无线功能、设置网络端口桥接、设置默认网关、配置 wan 口等等。

修改后的 network 文件和 wireless 文件如图 2-4 所示。



```
config interface 'loopback'
    option ifname 'lo'
    option proto 'static'
    option ipaddr '127.0.0.1'
    option netmask '255.0.0.0'

config globals 'globals'
    option ula_prefix 'fd44:31c6:85c5::/48'

config interface 'lan'
    option type 'bridge'
    option proto 'static'
    option ipaddr '192.168.11.1'
    option netmask '255.255.255.0'
    option ip6assign '60'
    option _orig_ifname 'eth0 wlan0'
    option _orig_bridge 'true'

config interface 'wan'
    option proto 'dhcp'
    option ifname 'eth0'
```

```
config wifi-device 'radio0'
    option type 'mac80211'
    option channel '11'
    option hwmode '11g'
    option path 'platform/soc/3f300000.m'
    option htmode 'HT20'
    option disabled 'CN'

config wifi-iface 'default_radio0'
    option device 'radio0'
    option network 'lan'
    option mode 'ap'
    option ssid 'Spring'
    option encryption 'psk2'
    option key 'spring371327'
```

图 2-4 network 和 wireless 配置文件

其中设置了 wifi 的名称、加密方式 (psk2) 和密码, 图片的左侧, 由于我的 wan 口是通过 dhcp 方式获取的网络地址, 如果想设置成固定的 IP、DNS 等信息, 可以按照图 2-5 的模式进行修改保存。



```
config interface 'wan'
    option ifname 'eth0'
    option proto 'static'
    option ipaddr '****.****.****.****'
    option netmask '****.****.****.****'
    option gateway '****.****.****.****'
    option dns '****.****.****.****'
```

图 2-5 wan 口设置静态 ip

将以上修改保存并退出, 同时将连在电脑端的网线接入以太网, 重启树莓派, 此时已经将树莓派打造成了一个简单的路由器, 使用手机或电脑可以搜索到一个名为 Spring 的 WiFi 信号。输入密码进行连接, 在浏览器中输入刚刚设置的默认网关 192.168.11.1, 即可打开如图 2-6 所示的后台管理界面, 之后的路由器设置既可以选择命令行设置也可以选择在此界面设置, 建议两者相结合。

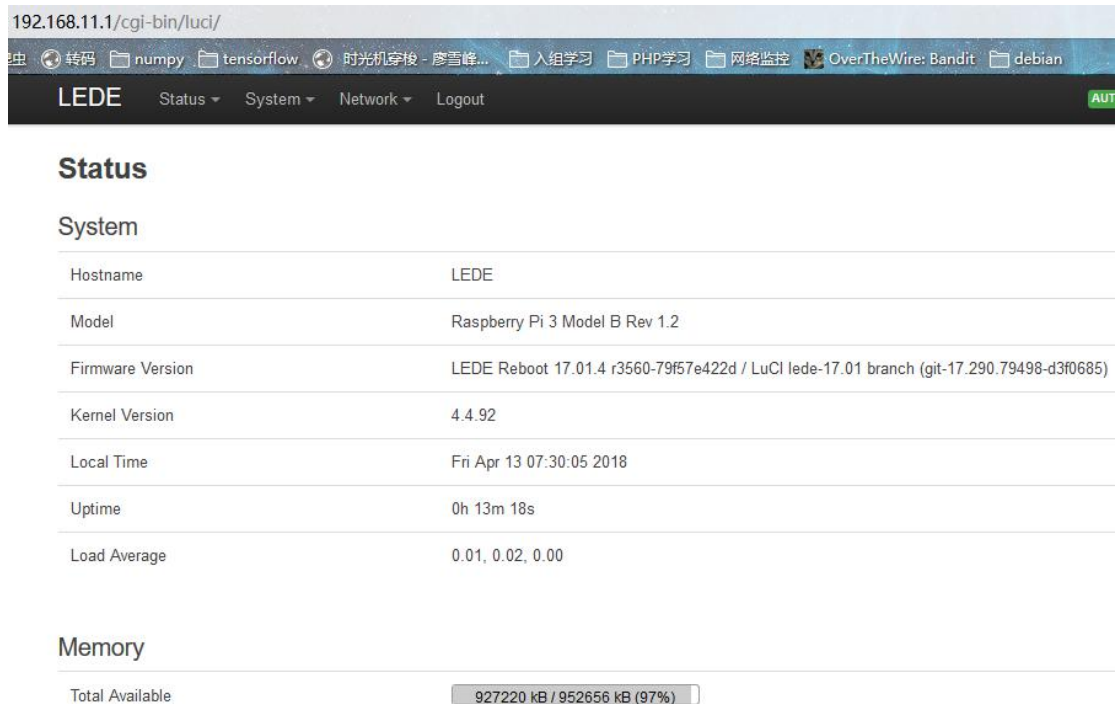


图 2-6 LEDE 后台管理界面

2.3.2 防火墙设置

一般来说，我们希望能够通过外网对 LEDE 后台界面进行访问，并能够通过 SSH 协议来设置路由器，但是，LEDE 的安全策略使得防火墙默认并未开放这两个端口的数据流通，所以需要进一步的设置。

导航：Network >> Firewall >> Traffic Rules

参照图 2-7 开放 80 端口和 22 端口的 TCP 通信。

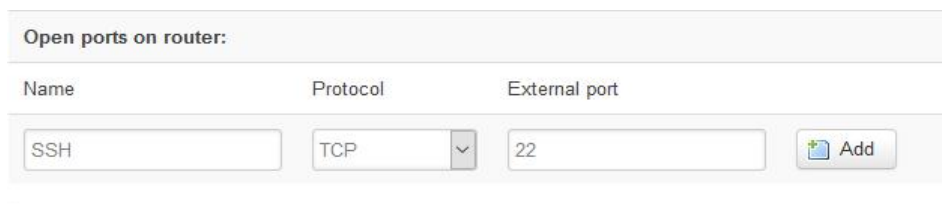


图 2-7 开放 wan 口 22 号端口通信

保存并应用后，以后就可以使用 wan 口对 LEDE 进行相关设置。

2.3.3 界面设置

- (1) ssh 连接上 LEDE，执行 `opkg update` 命令，更新一下软件包，其中的 `opkg` 是 LEDE 的包管理工具。
- (2) 执行 `opkg install luci-il8n-base-zh-cn` 命令，安装界面汉化包。
- (3) 在界面中 System >> System Properties >> Language and Style 中设置

界面为中文选项，刷新路由器界面，可以看到已经汉化成功。

(4) 在“主题 >> 软件包”中搜索 theme 关键词，可以看到官网提供的几种主题，如图 2-8 所示，当然很多论坛也提供第三方主题，选择自己喜欢的主题一键安装即可，应用方法和第（3）步中一样。

安装	luci-theme-bootstrap	git-18.098.72829-575e327-1	13826	Bootstrap Theme (default)
安装	luci-theme-bootstrap	git-17.290.79498-d3f0685-1	0	
安装	luci-theme-freifunk-generic	git-18.098.72829-575e327-1	13545	Freifunk Generic Theme
安装	luci-theme-material	git-18.098.72829-575e327-1	51395	Material Theme
安装	luci-theme-openwrt	git-18.098.72829-575e327-1	7748	LuCI OpenWrt.org theme

图 2-8 系统主题列表

(5) 在“状态 >> 总览”中默认是不显示 CPU 温度的，实时温度显示还是很必要的，可以通过修改 index.html 实现温度的显示。

index.html 路径： /usr/lib/lua/luci/view/admin_status/index.htm

在第 692 行添加如下一行代码：

```
<tr><td width="33%"><%=CPU Temp%></td><td><%=luci.sys.exec("cat -c1-2 /sys/class/thermal/thermal_zone0/temp")%>&#8451;</td></tr>
```

其中 ℃ 代表英文的摄氏度符号，添加后的效果如图 2-9 所示。

运行时间	1h 10m 58s
CPU Temp	40 °C
平均负载	0.33, 0.20, 0.08

图 2-9 CPU 温度显示

至此，路由器的基本配置过程已经结束，接下来，我们可以在上面扩展我们想要的功能。

第三章 LEDE 扩展功能模块设计

3.1 科学上网模块

3.1.1 搭建交叉编译环境

3.1.1.1 什么是交叉编译

在某一种计算机环境中运行编译程序，使其能编译出另一种计算机环境中能运行的代码就是交叉编译。也就是说在一个平台上生成另一个平台上的可执行代码^[13]。

如果要对某一源代码进行交叉编译，需要在电脑上安装与目标平台对应的交叉编译工具链（crosscompilation tool chain），然后用这个交叉编译工具链进行编译，最终生成可在目标平台上运行的代码^[14]。

3.1.1.2 为什么要使用交叉编译

一是由于树莓派所提供的硬件资源有限，如 CPU、RAM 等性能都不是特别高，而编译源代码对相关资源的要求很高，所以很难在树莓派上进行本地编译。二是由于大量的优秀开源软件并没有被 LEDE 官方收录进 feeds 源内，因此我们需要自己编译这些软件包来扩充路由器的功能，使其更加智能化。

3.1.1.3 本人搭建的是虚拟机环境，使用的是 VMware Workstation 14.1，系统安装的是 Ubuntu16.04 LTS，相关的软件都可以在相应的官网下载到。

Step1: 先在 ubuntu 上安装一些必要的软件包，它们可以为编译过程提供必要的支持。

```
sudo apt-get install build-essential subversion libncurses5-dev zlib1g-dev gawk  
gcc-multilib flex git-core gettext libssl-dev
```

Step2: 下载树莓派 3B 环境下的 LEDE 的 SDK 包，下载地址为：

<https://downloads.lede-project.org/releases/17.01.1/targets/brcm2708/bcm2710/>

Step3: 解压下载下来的压缩包

```
tar -xvJf lede-sdk-17.01.4-brcm2708-bcm2710_gcc-5.4.0_musl-1.1.16_eabi.Linux-x86_64.tar.xz  
cd lede-sdk-17.01.4-brcm2708-bcm2710_gcc-5.4.0_musl-1.1.16_eabi.Linux-x86_64
```

经过上述 3 步操作，编译环境已经搭建好。

3.1.2 编译软件包

(1) GitHub 上提供了 openwrt-shadowsocksr 源码，具体的交叉编译操作过程在主页上有提供^[9]，如图 3-1 所示，源码下载地址如下：

<https://github.com/AlexZhuo/openwrt-shadowsocksr>

- 从 OpenWrt 的 SDK 编译

```
# 以 ar71xx 平台为例
tar xjf OpenWrt-SDK-ar71xx-for-linux-x86_64-gcc-4.8-linaro_uClibc-0.9.33.2.tar.bz2
cd OpenWrt-SDK-ar71xx-*
# 安装 feeds
# 方案一(推荐):使用自定义feeds
git clone https://github.com/AlexZhuo/openwrt-feeds.git package/feeds
# 方案二:使用官方feeds
./scripts/feeds update base packages
./scripts/feeds install zlib libopenssl libpolarssl libmbedtls libpcrc
rm -rf package/feeds/base/mbedtls/patches
# 获取 shadowsocksr-libev Makefile
git clone https://github.com/AlexZhuo/openwrt-shadowsocksr.git package/shadowsocksr-libev
# 选择要编译的包 Network -> shadowsocksr-libev
make menuconfig
# 开始编译
make package/shadowsocksr-libev/compile V=99
```

图 3-1 交叉编译过程图

依次执行图 3-1 中的指令，当执行 make menuconfig 后，进入编译配置菜单，在 network 栏中进行如图 3-2 的选择。

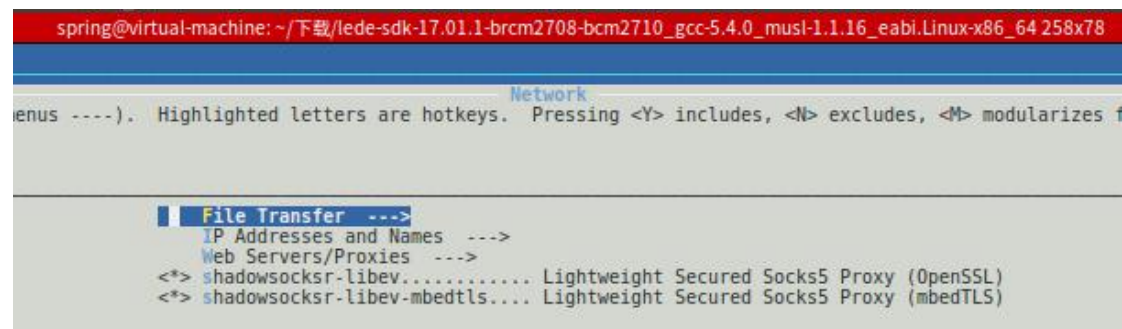


图 3-2 编译配置菜单

编译过程如图 3-3 所示，该过程消耗时间比较长，耐心等待。

```

[ 84%] Building C object library/CMakeFiles/mbedcrypto.dir/version.c.o
[ 85%] Building C object library/CMakeFiles/mbedcrypto.dir/version_features.c.o
[ 85%] Building C object library/CMakeFiles/mbedcrypto.dir/xtea.c.o
[ 86%] Linking C shared library libmbedcrypto.so
make[5]: Leaving directory '/home/spring/下载/lede-sdk-17.01.1-brcm2708-bcm2710_gcc-5.4.0_musl-1.1.16_eabi.Linux-x86_64'
[ 86%] Built target mbedcrypto
make[5]: Entering directory '/home/spring/下载/lede-sdk-17.01.1-brcm2708-bcm2710_gcc-5.4.0_musl-1.1.16_eabi.Linux-x86_64'
Scanning dependencies of target mbedx509
make[5]: Leaving directory '/home/spring/下载/lede-sdk-17.01.1-brcm2708-bcm2710_gcc-5.4.0_musl-1.1.16_eabi.Linux-x86_64'
make[5]: Entering directory '/home/spring/下载/lede-sdk-17.01.1-brcm2708-bcm2710_gcc-5.4.0_musl-1.1.16_eabi.Linux-x86_64'
[ 87%] Building C object library/CMakeFiles/mbedx509.dir/certs.c.o
[ 87%] Building C object library/CMakeFiles/mbedx509.dir/pkcs11.c.o
[ 88%] Building C object library/CMakeFiles/mbedx509.dir/x509.c.o
[ 89%] Building C object library/CMakeFiles/mbedx509.dir/x509_create.c.o
[ 89%] Building C object library/CMakeFiles/mbedx509.dir/x509_crl.c.o
[ 90%] Building C object library/CMakeFiles/mbedx509.dir/x509_crt.c.o
[ 91%] Building C object library/CMakeFiles/mbedx509.dir/x509_csr.c.o
[ 91%] Building C object library/CMakeFiles/mbedx509.dir/x509write_crt.c.o
[ 92%] Building C object library/CMakeFiles/mbedx509.dir/x509write_csr.c.o
[ 93%] Linking C shared library libmbedx509.so
make[5]: Leaving directory '/home/spring/下载/lede-sdk-17.01.1-brcm2708-bcm2710_gcc-5.4.0_musl-1.1.16_eabi.Linux-x86_64'
[ 93%] Built target mbedx509
make[5]: Entering directory '/home/spring/下载/lede-sdk-17.01.1-brcm2708-bcm2710_gcc-5.4.0_musl-1.1.16_eabi.Linux-x86_64'
Scanning dependencies of target mbedtls
make[5]: Leaving directory '/home/spring/下载/lede-sdk-17.01.1-brcm2708-bcm2710_gcc-5.4.0_musl-1.1.16_eabi.Linux-x86_64'
make[5]: Entering directory '/home/spring/下载/lede-sdk-17.01.1-brcm2708-bcm2710_gcc-5.4.0_musl-1.1.16_eabi.Linux-x86_64'

```

图 3-3 编译过程图

编译成功后，在自己对应的目录：

***/lede-sdk-17.01.1-brcm2708-bcm2710_gcc-5.4.0_musl-1.1.16_eabi.Linux-x86_64

/bin/packages/arm_cortex-a53_neon-vfpv4/base

下可以看到编译好的 .ipk 软件包，共 7 个，如图 3-4 所示。

名称	修改日期
 libmbedtls_2.4.0-3_arm_cortex-a53_neon-vfpv4.ipk	2018/4/13 20:28
 libopenssl_1.0.2k-1_arm_cortex-a53_neon-vfpv4.ipk	2018/4/13 20:36
 openssl-util_1.0.2k-1_arm_cortex-a53_neon-vfpv4.ipk	2018/4/13 20:36
 shadowsocksr-libev_2.5.6-2_arm_cortex-a53_neon-vfpv4.ipk	2018/4/13 20:44
 shadowsocksr-libev-mbedtls_2.5.6-2_arm_cortex-a53_neon-vfpv4.ipk	2018/4/13 20:47
 zlib_1.2.11-1_arm_cortex-a53_neon-vfpv4.ipk	2018/4/13 20:36
 zlib-dev_1.2.11-1_arm_cortex-a53_neon-vfpv4.ipk	2018/4/13 20:36

图 3-4 编译好的 ipk 软件包

(2) GitHub 同时提供了 luci-app-shadowsocksr 源码，该源码编译安装后支持在浏览器界面上修改 ssr 代理的配置，对非专业人员来说非常方便。下载地址如下：

<https://github.com/AlexZhuo/luci-app-shadowsocksR>

编译过程参照 shadowsocksr 的编译过程即可，编译完成后，会生成如下一个 ipk 软件包：

luci-app-shadowsocksR_1.6-1_all.ipk

至此，LEDE 客户端安装 shadowsocksr 所需的所有软件包已经编译完成，等待安装。

3.1.3 客户端模块安装

将 3.1.2 中生成的所有软件包通过 scp 命令上传到 LEDE 的 /tmp/ssr 目录下，执行如下指令：

```
scp *.ipk root@192.168.11.1:/root/ssr
```

接下来执行一系列的 ipk 安装操作。

(1) `opkg install shadowsocksr-libev_2.5.6-2_arm_cortex-a53_neon-vfpv4.ipk`

它会自动安装 `libpcrc` `zlib` `libopenssl` 等依赖包。

(2) `opkg install luci-app-shadowsocksR_1.6-1_all.ipk`

提示错误，如图 3-5 所示。因为要依赖 `dnsforwarder`，接下来我们在 GitHub 上找到 `dnsforwarder`，进行交叉编译。

```
root@LEDE:~/ssr# opkg install luci-app-shadowsocksR_1.6-1_all.ipk
Installing luci-app-shadowsocksR (1.6-1) to root...
Collected errors:
* satisfy_dependencies_for: Cannot satisfy the following dependencies for luci-app-shadowsocksR:
*   dnsforwarder
* opkg_install_cmd: Cannot install package luci-app-shadowsocksR.
root@LEDE:~/ssr#
```

图 3-5 ssr 错误提示信息

GitHub 地址：

<https://github.com/AlexZhuo/openwrt-dnsforwarder.git>

编译方式和 SSR 一样，按照 GitHub 上该项目主页上的教程操作即可。

同样，将编译好的 ipk 上传到 /root/ssr 文件夹下。

(3) `opkg install dnsforwarder_6-1_arm_cortex-a53_neon-vfpv4.ipk`

因为 `dnsforwarder` 需要依赖 `dnsmasq-full`，而 LEDE 预装的是 `dnsmasq` 所以会出现如图 3-6 所示的错误提示。

```
Configuring ipset.
Collected errors:
* check_data_file_clashes: Package dnsmasq-full wants to install file /etc/hotplug.d/ntp/25-dnsmasqsec
  But that file is already provided by package * dnsmasq
* check_data_file_clashes: Package dnsmasq-full wants to install file /etc/init.d/dnsmasq
  But that file is already provided by package * dnsmasq
* check_data_file_clashes: Package dnsmasq-full wants to install file /usr/sbin/dnsmasq
  But that file is already provided by package * dnsmasq
* opkg_install_cmd: Cannot install package dnsforwarder.
```

图 3-6 dnsforwarder 错误信息

(4) `opkg update`

`opkg install dnsmasq-full --force-overwrite`

其中的参数 `--force-overwrite` 会进行复写式的强制安装。

(5) `dnsmasq-full` 安装完成后，再依次安装上面的两个 ipk，分别是

dnsforwarder 和 luci-app-shadowsocksR_1.6-1_all.ipk。

(6) 安装成功后，在浏览器中接入 LEDE 后台界面，刷新，可以在“服务 >> SSR 影梭”选项，可以在该页面对 SSR 进行相关参数配置，如图 3-7 所示。

代理设置

黑白名单

影梭透明代理

一个帮你翻墙的快速代理工具
使用教程[请点击这里](#)

状态 - **未运行**

配置

启用

☐

服务器地址

1.2.3.4

服务器端口

443

Proxy Tool

ShadowsocksR

ShadowsocksR

Shadowsocks

Redsocks2

密码

加密方式

chacha20-ietf

混淆协议

origin

图 3-7 SSR 设置界面

进一步，在高级选项中，可以使用 GFWList 进行路由选择，国内网址使用直连模式，国外网址使用代理模式。

3.1.4 服务端模块安装

购买一台国外 VPS(Virtual Private Server) 服务器，在上面搭建 shadowsocksr 服务端。本人购买的是香港的云服务器，上面预装 ubuntu 16.04，初次开机，执行如下指令：

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install git
git clone https://github.com/spring371327/shadowsocksr.git
```



```
cd shadowsocksr
```

可以看到目录下有很多文件，如图 3-8 所示：

```
root@chen:/home# cd shadowsocksr/
root@chen:/home/shadowsocksr# ls
CHANGES      LICENSE      README.rst   config.json  debian       initcfg.sh   mudb.json
CONTRIBUTING.md MANIFEST.in  apiconfig.py configloader.py importloader.py initmudbjson.sh mujson_mgr.py
Dockerfile    README.md    asyncmgr.py  db_transfer.py initcfg.bat  logrun.sh    mysql.json
root@chen:/home/shadowsocksr#
```

图 3-8 服务器端 SSR 配置文件

接下来进行相关参数的设置，分别执行：

```
./initmudb.sh
```

```
vim mudb.json
```

如图 3-9 所示：

```
root@chen: /home/shadowsocksr
{
  "d": 0,
  "enable": 1,
  "method": "aes-128-ctr",
  "obfs": "tls1.2_ticket_auth_compatible",
  "passwd": "Yry2dixt",
  "port": 60557,
  "protocol": "auth_aes128_md5",
  "transfer_enable": 9007199254740992,
  "u": 0,
  "user": "60557"
}
```

图 3-9 SSR 配置文件

在该配置文件中编辑选择的加密方式，混淆方式，服务端口，密码等。

最后执行：

```
./run.sh
```

开启服务。

至此，客户端和服务端的 shadowsocksr 服务都已经全部搭建完毕，使用手机或电脑连上该 wifi，可以实现透明代理，方便使用谷歌搜索、谷歌学术等应用，如图 3-10 所示，方便许多；尤其是对一些做网络流量分析的人员来说，可以获得纯净的、未经代理软件处理过的流量，方便许多。



图 3-10 使用谷歌搜索

3.2 Aria2 离线下载

3.2.1 Aria2 简介

Aria2 是一个命令行下的下载工具，具有轻量级、多协议、多来源的特点，支持多线程、断点及分段下载，支持 HTTP/HTTPS、TFP、BitTorrent、Metalink 协议，内建 XML-RPC 和 JSON-RPC 用户界面^[15]。虽然它是一个命令行工具，但我们可以为它安装网页控制程序，从而实现通过图形面板来控制下载任务。Aria2 因其用户量十分庞大，已经被 LEDE 收进其预置软件源中。

3.2.2 LEDE 安装 Aria2 和 WebUI-Aria2

在图形化的包管理界面中搜索 aria2，结果如图 3-11 所示。

软件包名称	版本	大小 (.ipk)	描述
aria2	1.30.0-1	491552	aria2 is a lightweight multi-protocol & multi-source com
luci-app-aria2	1.0.1-2	5064	LuCI Support for Aria2
luci-i18n-aria2-pt-br	1.0.1-2	3194	Translation for luci-app-aria2 - Português do Brasil (Bra
luci-i18n-aria2-ru	1.0.1-2	3474	Translation for luci-app-aria2 - Русский (Russian)
luci-i18n-aria2-zh-cn	1.0.1-2	2907	Translation for luci-app-aria2 - 中文 (Chinese)
webui-aria2	2016-10-11-1	3079645	The aim for this project is to create the worlds best and aria2. Very simple to use

其中 aria2 是核心执行部分，luci-app-aria2 是 luci 界面程序，luci-i18n-aria2-zh-cn 是 luci 界面程序的汉化翻译包，webui-aria2 是 aria2 的网页控制面板。通过界面或者 opkg 安装上面 4 个软件包，安装后的界面显示如图 3-12 所示。

Aria2 状态

Aria2 正在运行 打开WebUI-Aria2

Aria2 配置

基本设置 文件和目录 任务设置 BT设置

启用 ☒

以此用户权限运行

RPC端口

RPC认证方式

图 3-12 aria2 界面程序

得益于 luci 界面，我们不需要在命令行下修改配置文件，只要按照提示在文本框中填写相应内容即可。

3.2.3 Aria2 使用

Aria2 有两种常用的下载模式，第一种是直接下载，如：

```
aria2c "http://host/file.zip"
```

使用这种模式，当它下载完成后就会自动退出，如同 wget 一样。

第二种下载模式是 rpc server 模式，与直接下载不同的是它启动之后什么都不干，一直等待从 rpc 接口添加任务，下载完成之后不会自动退出，如同迅雷下载一样^[16]。

在 aria2 基本设置界面中设置 RPC 端口和认证方式，此处选择令牌认证方式，复制随机生成的 RPC 令牌，在 aria2 webui 界面的连接管理中输入该令牌，即可成功连接，如图 3-13 所示。

连接设置 X

Aria2 RPC 主机和端口

主机:
 输入 Aria2 RPC 所在服务器的 IP 或域名 (默认: localhost)

端口:
 输入 Aria2 RPC 端口号 (默认: 6800)

RPC 路径:
 输入 Aria2 RPC 路径 (默认: /jsonrpc)

SSL/TLS 加密: ☐ 启用 SSL/TLS 加密

密码令牌 (可选):

正在尝试使用新的连接设置来连接到 Aria2X

通过 RPC 连接到 Aria2 成功...X

图 3-13 RPC 连接设置

现在就可以下载我们想要网络资源了，拿一张百度图片测试一下，效果如图 3-14 所示。



图 3-14 aria2 下载测试

使用迅雷 U 享版和 Aria2 做下载速度的对比测试，当下载同一 BT 资源时，Aria2 的下载速度是七级迅雷会员下载速度的 1.5 倍左右。之后我们会讲到如何将移动硬盘等存储介质挂载到 LEDE 中并通过文件共享或者 FTP 服务共享下载的资源。

3.3 文件共享

3.3.1 Samba 文件共享

3.3.1.1 Samba 简介

Samba 是一款运行在 Linux 环境下的基于 SMB (Server Messages Block) 协议实现的共享文件的免费软件。其中 SMB (信息服务块) 是局域网中的一种通信协议，可共享局域网中的文件、打印机等资源。它是一种 C/S 模式的协议，客户端通过该协议访问服务器上的资源。当前，该协议并不仅仅局限于局域网，还可以通过“NetBIOS over TCP/IP”与其他网络中的主机实现资源共享^[17]。

3.3.1.2 目标需求

文件目录结构：

```
/
|
|---download
```

```
|      |  
|      |---share
```

实现目标是：匿名用户 guest(nobody) 只能访问文件目录 (/download/share)，对其内的文件及目录具有添加、修改、删除权限，方便多人共享；管理员账户 root 通过密码方式登录，管理整个分区 (/download)，对其内的所有文件都具有添加、修改、删除权限，方便管理。

3.3.1.3 需求实现

1、安装

```
opkg update  
opkg install luci-i18n-samba-zh-cn
```

直接安装中文语言包，会自动安装 samba-server、luci-app-samba 等依赖包。

2、全局设置

导航：服务 >> 网络共享 (samba)

打开设置面板，做如下修改：

```
unix charset = UTF-8  
#invalid users = root  
map to guest = Bad user  
null passwords = yes
```

作用分别是：开启 UTF-8 国际字符集，允许 root 用户登录，当用户名输入错误时则以匿名用户登录，允许密码为空。

3、访问控制

执行如下指令为 root 用户设置 smaba 登录密码；修改 /download/share 目录权限为 777，否则匿名用户无法对其下的内容进行添加、修改、删除操作。

```
smbpasswd -a root  
chmod 777 /download/share
```

导航：服务 >> 网络共享 >> 共享目录，按照如图 3-15 修改，点击“保存&应用”。

共享目录

共享名	目录	允许用户	只读	允许匿名用户	创建权限	目录权限	
					新文件权限	新目录权限	
share	/root/download/share		<input type="checkbox"/>	<input checked="" type="checkbox"/>	0755	0755	
root	/root/download	root	<input type="checkbox"/>	<input type="checkbox"/>	0777	0777	

图 3-15 共享目录设置

4、实现效果

路由器设置的网关是 192.168.11.1，所以用 file://192.168.11.1 或者在网络共享中心进行访问，效果如图 3-16 所示。

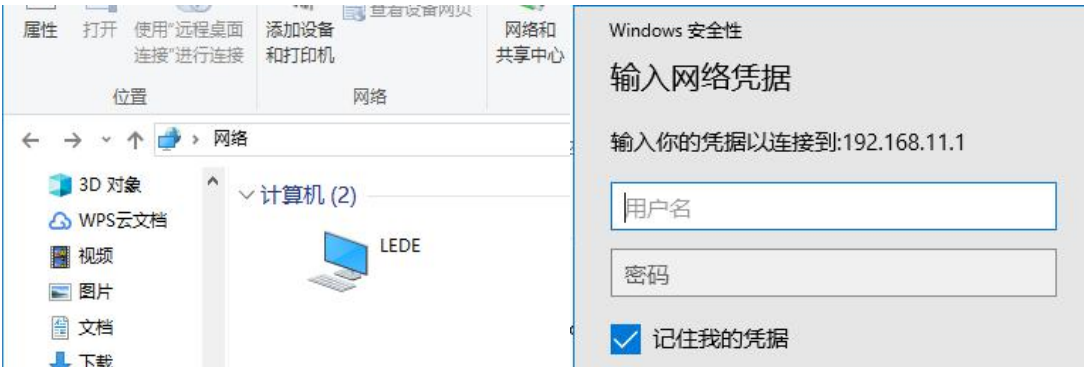


图 3-16 共享文件访问效果图

3.3.2 FTP 服务器

3.3.2.1 FTP 简介

FTP（文件传输协议）是一种基于 C/S 模式的文件传输协议，使用该协议，可以实现不同主机之间的资源共享，如下载、上传、修改文件等。

FTP 协议通信过程如图 3-17 所示。

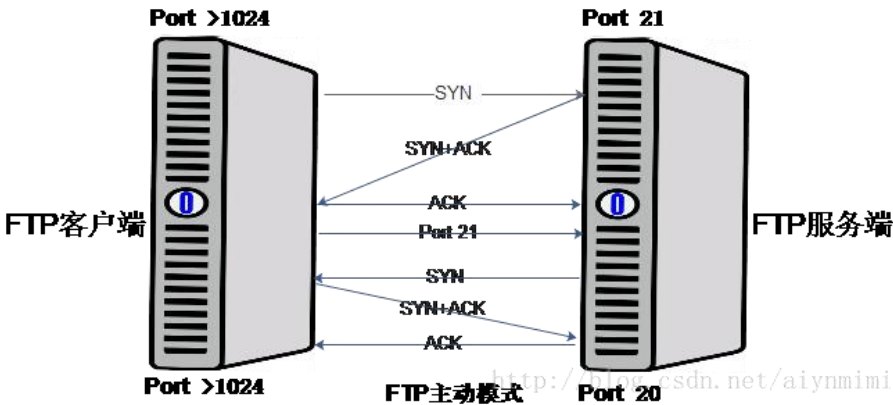


图 3-17 FTP 通信过程

3.3.2.2 安装并配置FTP 服务

```
opkg update
```

```
opkg install vsftpd
```

```
opkg install openssh-sftp-server
```

配置 /etc/vsftpd.conf

不允许匿名用户登录，添加如下内容：

```
secure_chroot_dir=/root
```

```
ftp_username=root
```

3.3.2.3 访问效果展示

使用 xftp，实现上位机和下位机的文件访问传输，效果如图 3-18 所示。

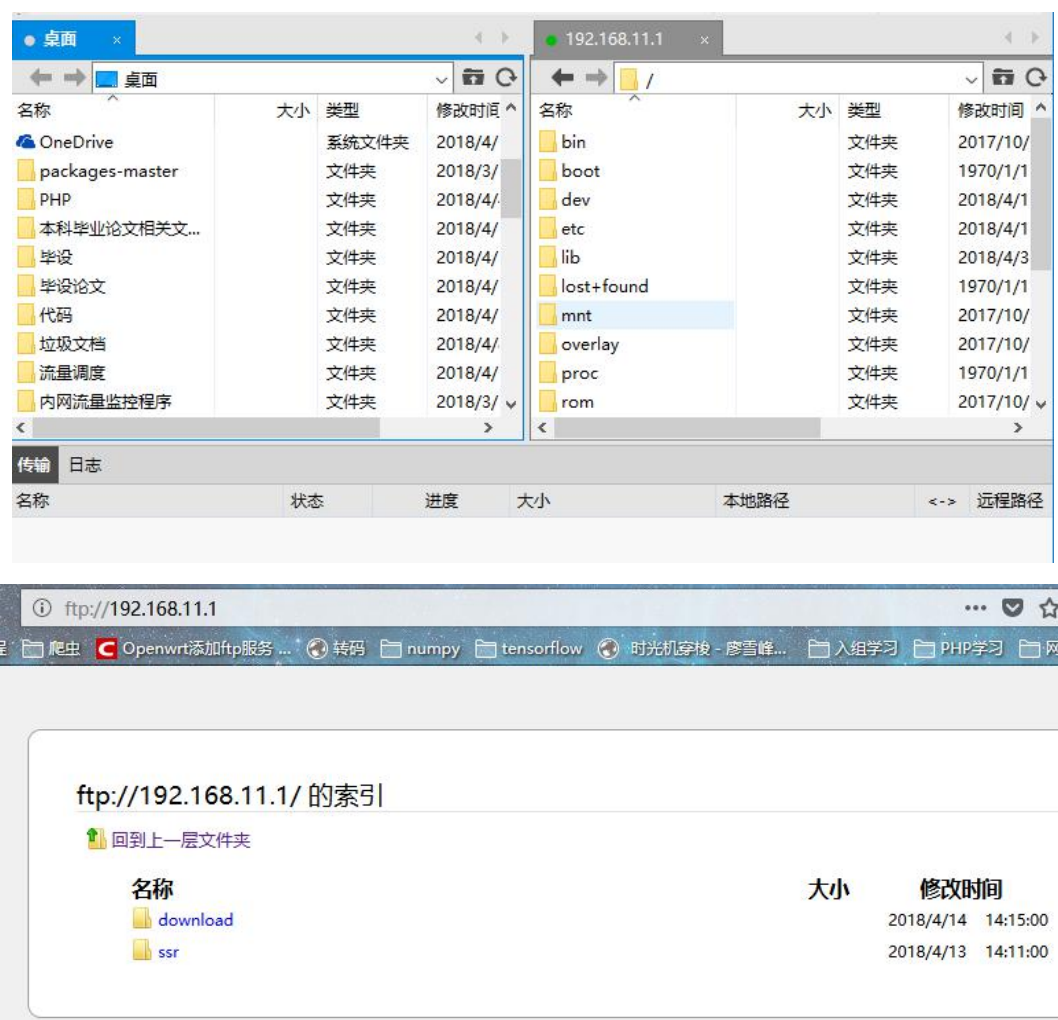


图 3-18 ftp 访问图

3.4 Ngrok 内网穿透

3.4.1 ngrok 简介

无论身在何处都能及时方便地管理局域网路由器，这对网络管理员来说异常重要。通过在公共节点和运行在本地的路由器之间搭建一条安全的通道，即可实现上述需求。Ngrok 就是这样的一个反向代理软件，并且可以记录和分析通道上的流量，方便网络管理员发现问题并及时处理。当然，还可以使用 Ngrok 在 TCP 层提供的端口映射功能，使其不仅仅局限于某一特定的服务^[18]。其原理如图 3-19 所示。

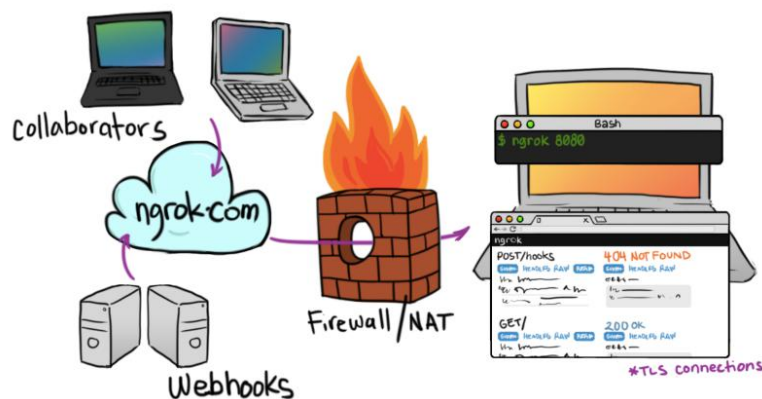


图 3-19 ngrok 原理图

3.4.2 LEDE 中搭建 ngrok

3.4.2.1 设置账号

登录 ngrok 的官方网站 (<https://ngrok.com>)，创建一个账号。完成账号注册后，将会得到一个认证令牌，需要使用这个令牌来连接 LEDE 和 ngrok 账号。

3.4.2.2 下载 ngrok

在 LEDE 中使用如下命令下载 ngrok：

```
wget https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-arm.zip
```

接下来，使用下列命令提取文件：

```
opkg update
```

```
opkg install unzip
```

```
unzip ngrok-stable-linux-arm.zip
```

解压完成后，在当前文件夹下会出现一个名叫“ngrok”的文件。

3.4.2.3 创建自己的账号

使用如下命令来设置自己的认证令牌(Auth)：

```
./ngrok authtoken ***** (自己的令牌)
```

3.4.2.4 运行 ngrok

使用如下命令来运行 ngrok：

```
./ngrok http 80
```

想要远程 ssh 连接，可以使用 `./ngrok tcp 22`，更具体的可以使用 `./ngrok help` 查看。

成功运行后，界面如图 3-20 所示。

```
Session Status      online
Account             王纪宝 (Plan: Free)
Version             2.2.8
Region              United States (us)
Web Interface        http://127.0.0.1:4040
Forwarding           http://51cd1d52.ngrok.io -> localhost:80
                    https://51cd1d52.ngrok.io -> localhost:80

Connections          ttl    opn    rt1    rt5    p50    p90
                    1      3      0.01   0.00   11.30   11.30

HTTP Requests
-----
GET /cgi-bin/luci/admin/system/clock_status 200 OK
GET /luci-static/resources/cbi/add.gif      200 OK
GET /luci-static/resources/cbi/remove.gif   200 OK
GET /cgi-bin/luci/admin/system/clock_status 200 OK
GET /luci-static/resources/cbi/help.gif     200 OK
GET /cgi-bin/luci/admin/system/system       200 OK
GET /luci-static/resources/icons/signal-0.png 200 OK
GET /luci-static/resources/icons/wifi.png    200 OK
GET /luci-static/resources/icons/signal-50-75.png 200 OK
GET /luci-static/resources/icons/ethernet.png 200 OK
```

图 3-20 ngrok 运行图

3.4.3 外网连接

根据图 3-20 中的 url 信息，手机连接 4G 网络，打开浏览器在地址栏中输入 `http://51cd1d52.ngrok.io`，即可以打开路由器的后台管理界面，实现路由器的远程访问与控制。效果如图 3-21 所示。



由于使用的是免费版,使用时会随机生成ngrok 域名下的一个二级域名,每次重启都会发生变化。当然也可以使用付费版,使用自己专属的 CNAME,或者购买 VPS 自己搭建 ngrok 服务器,在此不再赘述该方法。

如果想让该项服务放到后台永久运行,可以使用如下命令:

```
opkg install coreutils_nohup
nohup ./ngrok http 80 > /dev/null
2>&1 &
```

nohup 代表不挂断地运行命令,最后的&代表后台运行。命令中间部分表示将标准输出(linux 用 1 表示)和标准错误输出(linux 用 2 表示)都重定向到/dev/null,相当于丢弃掉。

图 3-21 LEDE 实现远控

至此,一些比较常用的扩展模块已经全部搭建完毕,但是并不满足于此,接下来我们在 LEDE 中搭建一个完整的 linux 文件系统(debian),在其中编写 C/C++, python 等代码,实现嵌入式开发工作。

第四章 LEDE 搭载 Debian 实现网络安全审计

4.1 SD 卡扩容及分区挂载

我们从 LEDE 官网下载下来的固件只拥有两个分区，第一个是一个大小为 20MB 的 Windows 下可见的分区，采用的是 FAT32 分区格式的文件系统，用于存放驱动引导文件，如图 4-1 所示。



图 4-1 LEDE 分区

第二个是一个 256MB 的 Ext4 分区，被挂载到根目录下。然而我们的 SD 卡容量通常为 8GB、16GB 甚至更大，剩下的空间默认是未被划分的。因此我们需要执行一些命令来为这块空闲的空间分区并挂载，这个分区可用于安装 Debian 系统或者充当 Samba 文件共享目录等。

4.1.1 安装磁盘操作必要的软件包

```
opkg install fdisk kmod-usb-storage kmod-fs-ext4 kmod-fs-vfat block-mount  
kmod-usb-uhci mkfs2fs f2fsck e2fsprogs
```

4.1.2 磁盘分区

安装完毕后，输入命令 `fdisk /dev/mmcblk0`，准备对 SD 卡进行操作，如图 4-2。

```
root@LEDE:/# fdisk /dev/mmcblk0  
Welcome to fdisk (util-linux 2.29.2).  
Changes will remain in memory only, until you decide to write them.  
Be careful before using the write command.  
  
Command (m for help):
```

图 4-2 fdisk 分区命令

输入 `m` 可以看到相关参数的介绍。我们这里输入 `p` 并回车，可以看到列出了 SD 卡的分区目录，从图 4-3 中可以看到，默认的第二个分区在 581631 Sector 位置结束。


```

Command (m for help): p
Disk /dev/mmcblk0: 14.9 GiB, 15931539456 bytes, 31116288 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x5452574f

Device            Boot Start      End Sectors  Size Id Type
/dev/mmcblk0p1 *    8192   49151   40960    20M  c W95 FAT32 (LBA)
/dev/mmcblk0p2      57344  581631  524288   256M  83 Linux

Command (m for help):

```

图 4-3 SD 卡分区目录

然后键入命令 n(new) 并回车，新建分区，选择 Primary 主分区，序列号为 3（Linux 只支持一块硬盘内存在 4 个主分区），起始位置填入第二个分区结束位置+1，这里也就是 581632，结束位置保持默认，回车。

```

Command (m for help): n
Partition type
  p   primary (2 primary, 0 extended, 2 free)
  e   extended (container for logical partitions)
Select (default p):

Using default response p.
Partition number (3,4, default 3): 3
First sector (2048-31116287, default 2048): 581632
Last sector, +sectors or +size{K,M,G,T,P} (581632-31116287, default 31116287):

Created a new partition 3 of type 'Linux' and of size 14.6 GiB.
Partition #3 contains a ext4 signature.

Do you want to remove the signature? [Y]es/[N]o: Y

The signature will be removed by a write command.

Command (m for help): |

```

图 4-4 SD 分区操作

使用命令 i 来查看分区的信息，如图 4-5 所示，可以看到，分区操作是没有问题，使用 w 命令，应用修改并退出 fdisk 程序。重启树莓派，即可应用新的分区系统。

```

Command (m for help): i
Partition number (1-3, default 3): 3

Device: /dev/mmcblk0p3
Start: 581632
End: 31116287
Sectors: 30534656
Cylinders: 477105
Size: 14.6G
Id: 83
Type: Linux
Start-C/H/S: 896/0/1
End-C/H/S: 815/3/16

Command (m for help):

```

图 4-5 检查 SD 卡分区

4.1.3 分区格式化及分区挂载

分区操作已经完成，但是新建的分区还需要格式化来赋予文件系统并进行其他初始化操作，不然分区挂载的时候会不识别，使用下列命令对其格式化为 Ext4 文件系统。

```
mkfs.ext4 /dev/mmcblk0p3
```

在 windows 中，只要磁盘插入主机并且磁盘存在分区，系统通常会默认为其分配盘符。但在 linux 系统中，分区通常不会被自动挂载，我们需要使用命令将其挂载到某个空目录下，然后通过对这个目录的读写操作来访问磁盘分区。

进入 Luci 界面，在“系统 >> 挂载点”页面可以看到 LEDE 默认挂载了 2 个分区，点击“添加”按钮，将新建的 mmcblk0p3 分区添加进去，高级选项保持默认即可，操作完成后如图 4-6 所示。

已挂载的文件系统

文件系统	挂载点	可用	已用	卸载分区
/dev/root	/	210.57 MB / 251.97 MB	15% (36.28 MB)	
tmpfs	/tmp	464.45 MB / 465.16 MB	0% (728.00 KB)	
/dev/mmcblk0p1	/boot	11.12 MB / 19.94 MB	44% (8.82 MB)	
tmpfs	/dev	512.00 KB / 512.00 KB	0% (0.00 B)	
/dev/mmcblk0p3	/debian	13.49 GB / 14.27 GB	0% (40.02 MB)	

图 4-6 分区挂载

至此，LEDE 中对 SD 卡的扩容及挂载操作已完成。

4.2 LEDE 中使用 debootstrap 安装 Debian

4.2.1 Debootstrap、Debian 简介

Debootstrap 工具简介：

Debootstrap 是 debian 系统的一个引导工具，使用该工具可以快速获得并安装一个基本的 Debian 系统。和 debian installer 不同的是它安装系统时直接连接 debian 的软件仓库，所以并不需要 iso 文件，安装成功后会在安装目录中生成符合 linux 文件系统标准的根文件系统，但它和其他发行版本的 linux 系统相比体积要小很多^[19]。此后，可以在原操作系统上执行 chroot 命令切换到 debian

文件系统，方便快捷。

Debian 系统简介：

Debian 是一个免费、自由的操作系统，其操作系统里的大部分基本工具都来自 GUN 计划。Debian 的发行版本主要有三个分支：stable（稳定版本）、testing（测试版本）、unstable（不稳定版本）。其所有的版本代号均出自 Pixar 的电影《玩具总动员》，如 stretch(debian 9)、jessie(debian 8)、wheezy(debian 7)、squeeze(debian 6)。

4.2.2 Debian 系统安装

执行下列指令在 LEDE 中安装一个简易的 debian 文件系统。

```
opkg update
```

```
opkg install debootstrap
```

```
debootstrap --arch=armel --variant=minbase stretch /debian
```

<http://ftp2.cn.debian.org/debian/>

安装完成后日志最后一行会有提示：Base system installed successfully，如图 4-7 所示。

```
I: Configuring libapt-pkg5.0:armel...
I: Configuring apt...
I: Configuring libc-bin...
I: Base system installed successfully.
root@LEDE:/# ls
```

图 4-7 Debian 安装成功

Debootstrap 的各参数含义如下：

--arch 指定 CPU 架构，armel(ARM)、amd64(AMD64 & Intel 64)、armhf(带 FPU 的 ARM)、i386(Intel x86-based)、mips(MIPS)。

--variant 指定要使用的引导脚本的名称，目前支持四种，分别是 minbase、buldd、fakechroot、scratchbox，其中 minbase 只包含一些基本包和 apt。

stretch 代表要安装的 debian 的版本代号。

/debian 代表文件系统的安装目录。

最后的 url 代表 debian 镜像的来源。

到现在，debian 就已经初步安装完毕，但为了更好的运行，需要把 LEDE 的系统

信息连接到 debian 子系统中。具体的操作方案是在 `/etc/rc.local` 中的 `exit 0` 之前加入如下几行代码，然后 `reboot` 进行重启。

```
mount -o bind /proc /debian/proc
mount -o bind /sys /debian/sys
mount -o bind /dev /debian/dev
```

执行 `chroot /debian /bin/bash` 命令进入 debian 子系统。

4.3 配置 Debian 文件系统

因为此时的系统没有经过任何的配置，所以进入 debian 系统后变化并不明显，如图 4-8 所示。接下来，对该文件系统进一步的设置。

```
root@LEDE:/# chroot /debian /bin/bash
root@LEDE:/# ls
bin  dev  home  media  opt  root  sbin  sys  usr
boot  etc  lib   mnt    proc  run   srv   tmp  var
root@LEDE:/# |
```

图 4-8 进入 debian 子系统

1、更新软件源信息

```
apt-get update
```

2、安装一些必要的开发工具包

```
apt-get install c c++ vim build-essential lrzsz
```

```
apt-get install python3 python3-pip
```

```
pip install IPy
```

```
apt-get install net-tools flex m4 bison libpcap-dev tcpdump
```

3、更改终端显示

将 `/root` 目录下的 `.bashrc` 文件的内容改为如图 4-9 所示。

```
# ~/.bashrc: executed by bash(1) for non-login shells.

# Note: PS1 and umask are already set in /etc/profile. You should not
# need this unless you want different defaults for root.
PS1='${debian_chroot:+($debian_chroot)}\[\033[01;31m\]debian\[\033[00m\]@spring:\[\033[01;33m\]\w\[\033[00m\]\$ '
umask 022

# You may uncomment the following lines if you want 'ls' to be colorized:
export LS_OPTIONS='--color=auto'
eval "$(dircolors)"
alias ls='ls $LS_OPTIONS'
alias ll='ls $LS_OPTIONS -al'
# alias l='ls $LS_OPTIONS -lA'
#
# Some more alias to avoid making mistakes:
# alias rm='rm -i'
# alias cp='cp -i'
# alias mv='mv -i'
```

图 4-9 `.bashrc` 文件内容

设置了命令 `ls` 的显示样式，头部信息的字体及颜色等，效果如图 4-10 所示。

```
bindresvport.blacklist default group-
ca-certificates deluser.conf gshadow
debian@spring:/etc# cd apt
debian@spring:/etc/apt# ls
apt.conf.d preferences.d sources.list sources.list.d
debian@spring:/etc/apt# cd apt.conf.d/
debian@spring:/etc/apt/apt.conf.d# ls
01autoremove 01autoremove-kernels 70debconf
debian@spring:/etc/apt/apt.conf.d#
debian@spring:/etc/apt/apt.conf.d#
debian@spring:/etc/apt/apt.conf.d#
```

图 4-10 命令行界面

4、解决系统中文乱码

Debian9 安装以后中文显示如同麻将块一样的乱码并且不支持中文输入，解决办法如下：

```
apt-get install aptitude
aptitude install locales
apt-get install tty-wqy-zenhei #安装中文字体
dpkg-reconfigure locales #选择 zh_CN.utf-8
apt-get install fcitx
```

在 `/root/.bashrc` 中添加如下一行代码：

```
export LANG=zh_CN.utf-8
```

5、设置 vim

编辑 `/root/.vimrc` 文件，设置 vim 中语法高亮，自动缩进，显示行号，设置解码格式防止中文显示乱码等，代码如图 4-11 所示。

```
1 |syntax on
2 set fileencodings=utf-8,gb2312,gb18030,gbk,ucs-bom,cp936,latin1
3 set fileencoding=utf-8
4 set nu!
5 set ts=4
6 set expandtab
7 set autoindent
8 set cindent
9 set tabstop=4
10 set shiftwidth=4
```

图 4-11 vim 配置文件

6、设置系统时间

由于初始系统使用的是 UTC 标准时间，对于以后开发时查看文件修改时间等操作带来不便，需要改成东八区时间，具体操作如下：

先使用 `tzselect` 命令选择时区为 Asia/Shanghai

再 `cp /usr/share/zoneinfo/Asia/Shanghai /etc/localtime`

完成设置后，使用 `date -R` 命令查看当前系统时间，如图 4-12 所示。

```
debian@spring:~# date -R          debian@spring:/etc# date -R
Mon, 16 Apr 2018 14:00:05 +0000    Mon, 16 Apr 2018 22:06:39 +0800
```

图 4-12 系统时间显示

4.4 基于 libpcap 编写流量统计代码

4.4.1 libpcap 功能函数简介

libpcap 是 Linux 平台下的网络数据包捕获函数库，提供了一些独立于系统用户包捕获的 API 接口函数，使得开发者可以轻松实现网络编程^[20]。

使用 libpcap 函数库进行网络编程的基本步骤如图 4-13 所示：

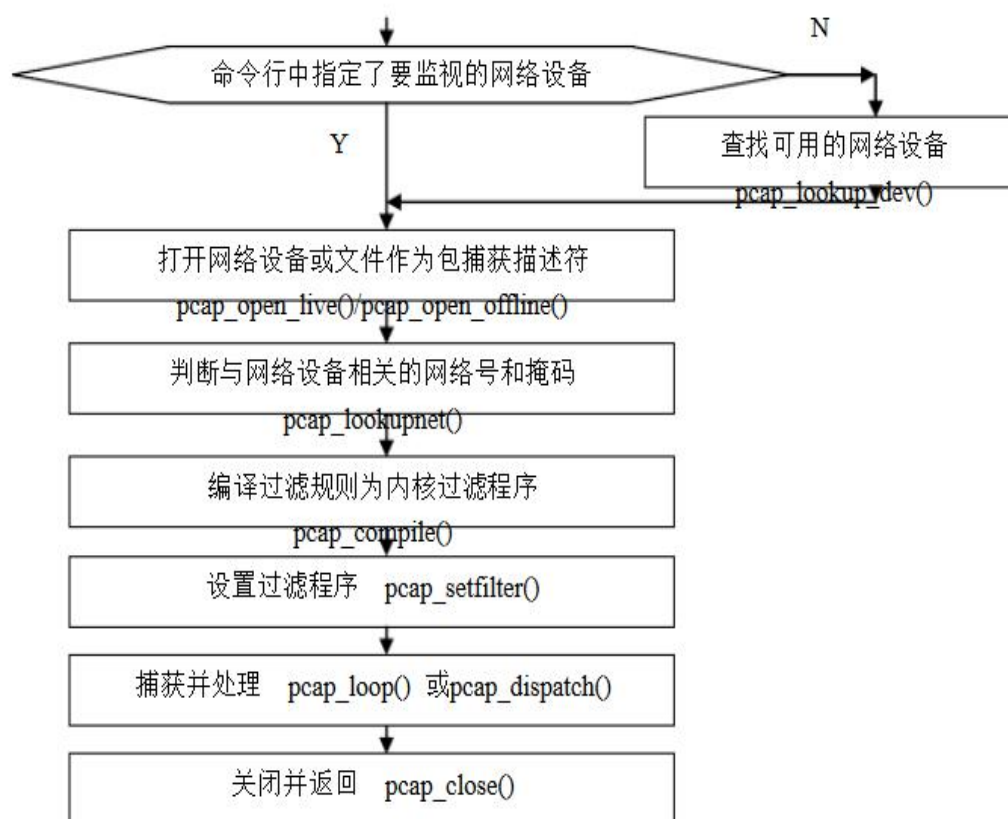


图 4-13 libpcap 网络编程流程图

4.4.2 通信流量采集及元信息留存模块实现

通过编码实现实时抓取流经指定网卡的数据包，并且可以自定义过滤条件以

抓取所需要的数据包。分析各个网络协议的格式、各协议字段的实际意义并将其头部信息显示在程序界面中。

解析数据包头部信息的关键代码如图 4-14 所示。

```
/* 解析数据包 */
void parse(const u_char *packet_data, char *catch_time, int packet_size){
    /* 获得IP数据包头部 */
    IP_HEADER *ip;
    unsigned int protocol;
    unsigned int ip_head_len;
    unsigned int ip_total_len;

    //以太网头部占14字节
    ip = (IP_HEADER *) (packet_data + 14);
    //数据包协议
    protocol = *((unsigned int *)(&ip->protocol)) & 0xff;
    //IP头部长度
    ip_head_len = (ip->version_headLength & 0xf)*4;
    //IP总长度
    ip_total_len = ntohs(ip->total_length);
    //数据包长度
    unsigned int packet_len = ip_total_len - ip_head_len;
    //源IP
    char src_ip[50];
    sprintf(src_ip, "%d.%d.%d.%d",
            ip->src_ip_address.byte1,
            ip->src_ip_address.byte2,
            ip->src_ip_address.byte3,
            ip->src_ip_address.byte4);
    //目的IP
    char dest_ip[50];
    sprintf(dest_ip, "%d.%d.%d.%d",
            ip->dest_ip_address.byte1,
            ip->dest_ip_address.byte2,
            ip->dest_ip_address.byte3,
            ip->dest_ip_address.byte4);
}
```

图 4-14 数据包头部信息解析

TCP 协议数据包解析代码如图 4-15 所示。

```
if(protocol == IP_TCP){ //TCP协议报头
    /* 获得TCP首部 */
    TCP_HEADER *tcp;
    tcp = (TCP_HEADER *) ((unsigned char *)ip + ip_head_len);

    //从网络字节顺序(大端)转换为主机字节顺序(小端)
    //大端模式：最高位放低地址
    //小端模式：最低位放低地址
    src_port = ntohs(tcp->src_port);
    dest_port = ntohs(tcp->dest_port);
    ack = ntohl(tcp->ack);
    sequence = ntohl(tcp->sequence);
    //TCP首部长度
    unsigned short hrf = ntohs(tcp->headlen_retain_flag);
    unsigned int tcp_head_len = ((hrf & 0xf000) >> 12)*4;
    //TCP数据长度
    data_len = packet_len - tcp_head_len;
    //标记
    flag = hrf & 0x3f;
}
```

```

strcpy(flag_str, "");
if((flag & TH_ACK) == TH_ACK){
    strcat(flag_str, "ACK=1 ");
}else{
    strcat(flag_str, "ACK=0 ");
}

if((flag & TH_SYN) == TH_SYN){
    strcat(flag_str, "SYN=1 ");
}else{
    strcat(flag_str, "SYN=0 ");
}

if((flag & TH_FIN) == TH_FIN){
    strcat(flag_str, "FIN=1");
}else{
    strcat(flag_str, "FIN=0");
}

strcpy(pro_name, "TCP");

```

图 4-15 TCP 协议解析

如果发现数据包是 HTTP 包，可以看到明文的头部信息，实现留存的代码如下图 4-16 所示。

```

/* 解析HTTP */
int parseHTTP(char *tcp_packet, int tcp_data_len, int tcp_head_len, char *src_ip, char *dest_ip){
    if(tcp_data_len > 0){
        char *data = (tcp_packet + tcp_head_len);
        //@@@printf("%s\n", data);
        char http_text[3000];
        strcpy(http_text, "");
        //检查请求报文
        int find = 0;
        if((strncmp(data, "GET", 3) == 0) || (strncmp(data, "POST", 4) == 0)){
            find = 1;
        }

        //检查响应报文
        if(find == 0){
            if(strncmp(data, "HTTP/1.1", 8) == 0){
                find = 1;
            }
        }

        if(find == 1){
            FILE *f;
            f = fopen(HTTP_CONTENT_FILE_NAME, "a+");
            if(f == NULL){
                printf("\n该数据包写入文件失败\n");
            }else{
                for(int i=0; i<strlen(data); i++){
                    if( data[i]=='\r' && data[i+1]=='\n' && data[i+2]=='\r' && data[i+3]=='\n' ){
                        data[i] = '\0';
                        break;
                    }
                }
                fprintf(f, "src_ip:%s----->dst_ip:%s\n", src_ip, dest_ip);
                fprintf(f, "%s", data);
                fprintf(f, "%s", "\n\n");
            }
            fclose(f);

            return 1;
        }
    }

    return 0;
}

```

图 4-16 HTTP 明文头部信息留存

实现抓取固定时长的数据包，使用 C 语言的多线程机制，开启一个子线程实现计时功能，主要代码如图 4-17 所示。

```
/* 子线程运行函数 */
void* thread_clock(void *argv){
    //使自身变成非阻塞线程
    pthread_detach(pthread_self());

    pcap_t *handle = ((argument*)argv)->handle;
    int timelen = ((argument*)argv)->timelen;
    //单位是毫秒
    sleep(timelen);
    //停止抓包
    pcap_breakloop(handle);
    return NULL;
}
```

图 4-17 子线程计时

4.4.3 通信流量日志信息统计分析模块实现

使用 python 代码进行某个自定义网段的网络流量日志信息的统计分析，读取日志文件，使用 IP 当键，自定义类做值，将每条信息依次插入字典中进行统计，最后将字典中的数据按序输出。主要功能代码如图 4-18 所示。

```
import os
from IPy import *

class Item(object):
    def __init__(self):
        self.UpCount = 0
        self.UpSize = 0
        self.DownloadCount = 0
        self.DownloadSize = 0

CIDR = "192.168.11.0/24"
top_dic = {"1": Item()}
top_dic.clear()
tcp_sd_dic = {"1": Item()}
tcp_sd_dic.clear()
udp_sd_dic = {"1": Item()}
udp_sd_dic.clear()

def tcp_analysis():
    file_object = open("file/TCP.txt", "r")
    line = file_object.readline()
    while line:
        words = line.split()
        if (words[1] not in IP(CIDR) and words[2] not in IP(CIDR)):
            continue
```

```

if (words[1] in IP(CIDR)):
    if (words[1] not in tcp_sd_dic.keys()):
        tcp_sd_dic[words[1]] = Item()
        tcp_sd_dic[words[1]].UpCount += 1
        tcp_sd_dic[words[1]].UpSize += int(words[5])
    else:
        tcp_sd_dic[words[1]].UpCount += 1
        tcp_sd_dic[words[1]].UpSize += int(words[5])

if (words[2] in IP(CIDR)):
    if (words[2] not in tcp_sd_dic.keys()):
        tcp_sd_dic[words[2]] = Item()
        tcp_sd_dic[words[2]].DownloadCount += 1
        tcp_sd_dic[words[2]].DownloadSize += int(words[5])
    else:
        tcp_sd_dic[words[2]].DownloadCount += 1
        tcp_sd_dic[words[2]].DownloadSize += int(words[5])
line = file_object.readline()
file_object.close()

```

图 4-18 通信元信息统计分析

4.4.4 编写 makefile 和 shell 脚本实现模块整合

Make 是软件开发中一种常用的软件构建工具，通过读取 makefile 文件来实现软件的自动化构建工作。这里，为了方便开发调试工作，使用它来自动编译连接编写的源代码，生成可执行文件。

Shell 脚本是一串符合 shell 语法的指令集合，可以将 shell 脚本看成是一个批处理文件，既然是批处理，因此可以在一定程度上简化人们的操作。此处所用到的 shell 脚本主要功能是将流量信息抓取模块与流量日志信息统计模块结合在一起。

Makefile 和 shell 代码如图 4-19 所示。

<pre> cc = gcc prom = demo deps = \$(shell find ./ -name "*.h") src = \$(shell find ./ -name "*.c") obj = \$(src:%.c=%.o) \$(prom) : \$(obj) \$(cc) \$(obj) -o \$(prom) -lpthread -lpcap %.o : %.c \$(deps) \$(cc) -c \$< -o \$@ clean: rm -rf \$(obj) \$(prom) </pre>	<pre> 1 #!/bin/bash 2 3 if [! -d "./file/"];then #判断file目录是否存在 4 mkdir ./file 5 fi 6 make #执行make命令，将C语言代码编译成可执行文件 7 ./demo 8 make clean 9 if [! -d "./analysis/"];then 10 mkdir ./analysis 11 fi 12 echo "正在进行日志统计分析..." 13 python3 net.py 14 echo "流量日志分析结果统计完成！" 15 </pre>
--	---

图 4-19 makefile 和 shell

4.5 代码运行及效果展示

在 code 目录下执行 `./shell` 命令，将会依次运行流量信息采集模块和日志信息统计模块，在结果的各个文件中可以看到 HTTP 通信信息，每个 IP 的 UDP 上、下包数，上、下包量，TCP 上、下包数，上、下包量，以及总包数、总包量等信息。当网络发生故障时能够根据日志信息进行快速准确地溯源，或者当局域网中有人恶意占用带宽等情况发生时，能够快速发现并及时采取措施。信息如图 4-20 所示。

```
src_ip:192.168.11.195----->dst_ip:124.193.0.2
GET /bootstrap/3.3.7/fonts/glyphicons-halflings-regular.woff2 HTTP/1.1
Host: cdn.bootcss.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:59.0) Gecko/20100101 Firefox/59.0
Accept: application/font-woff2;q=1.0,application/font-woff;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-Tw;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: identity
Referer: http://cdn.bootcss.com/bootstrap/3.3.7/css/bootstrap.min.css
Origin: http://firmware.koolshare.cn
Connection: keep-alive
```

```
debian@spring:~/code/analysis# cat udp_tcp_analysis.txt
```

IP	UDP上包数	UDP上包量	UDP下包数	UDP下包量	TCP上包数	TCP上包量	TCP下包数	TCP下包量	总包数	总包量
192.168.11.164	0	0.0KB	1	0.01KB	1104	4.81KB	6283	935.74KB	7388	940.56KB
192.168.11.195	179	1.47KB	101	1.93KB	1875	56.34KB	2882	207.33KB	5037	267.07KB
192.168.11.1	45	0.89KB	52	0.27KB	109	3.5KB	95	1.03KB	301	5.68KB
192.168.11.255	0	0.0KB	38	0.29KB	0	0.0KB	0	0.0KB	38	0.29KB

```
debian@spring:~/code/analysis# cat top_analysis.txt
```

IP	上包数	上包量	下包数	下包量	总包数	总包量
124.193.0.50	6067	910.01KB	900	3.6KB	6967	913.61KB
14.215.133.114	530	68.85KB	235	2.82KB	765	71.67KB
119.75.216.20	487	31.12KB	288	15.34KB	775	46.47KB
61.160.200.201	198	25.71KB	97	1.36KB	295	27.08KB
60.206.195.14	158	24.09KB	147	0.66KB	305	24.75KB
124.192.164.48	215	16.1KB	140	3.54KB	355	19.64KB
124.192.164.33	199	11.89KB	139	4.69KB	338	16.58KB

图 4-20 运行结果展示

第五章 局域网内主机行为监管

5.1 系统架构设计

局域网主机行为安全监管系统就是对网络中活动主机的各种行为进行监控和管理。在局域网监控方面，国内的许多网络安全公司也推出了自己的产品，如大家都熟知的 Net-eye 网眼监控、WorkWin、威眼 WeaView、LaneCat 网猫等^[21]。

目前，市面上的大多数局域网主机行为监管系统都是基于客户端/服务器架构的，这种架构设计的系统就需要在局域网中的每台主机上都安装配套的客户端软件，服务器端主机安装服务端软件，通过管理客户端软件来实现主机行为的监管与控制。这种客户端/服务器架构的监管系统在部署上存在环境复杂多变且工作量大等多方面的困难。

本章搭建的局域网监管系统采用的是 WFilter 软件。WFilter 是一个嗅探器，通过监听网络通讯数据包然后加以分析还原以实现监控功能；是一个基于内容的管理工具，可以详细记录网内所有的 Internet 访问记录并且基于内容实现访问控制、告警等功能。在部署上，只需要在局域网中的任何一台主机上安装 WFilter，即可对整个网络中的电脑主机使用互联网的状况进行有效的监管，但我们要监控的是一个无线路由器搭建的局域网环境，因此需要使用 LEDE 中开源的 port-mirroring 软件将流经路由器的所有网络流量封装成 TZSP 协议数据包，然后发送给监控主机，从而实现局域网主机行为监管。其系统架构如图 5-1 所示。

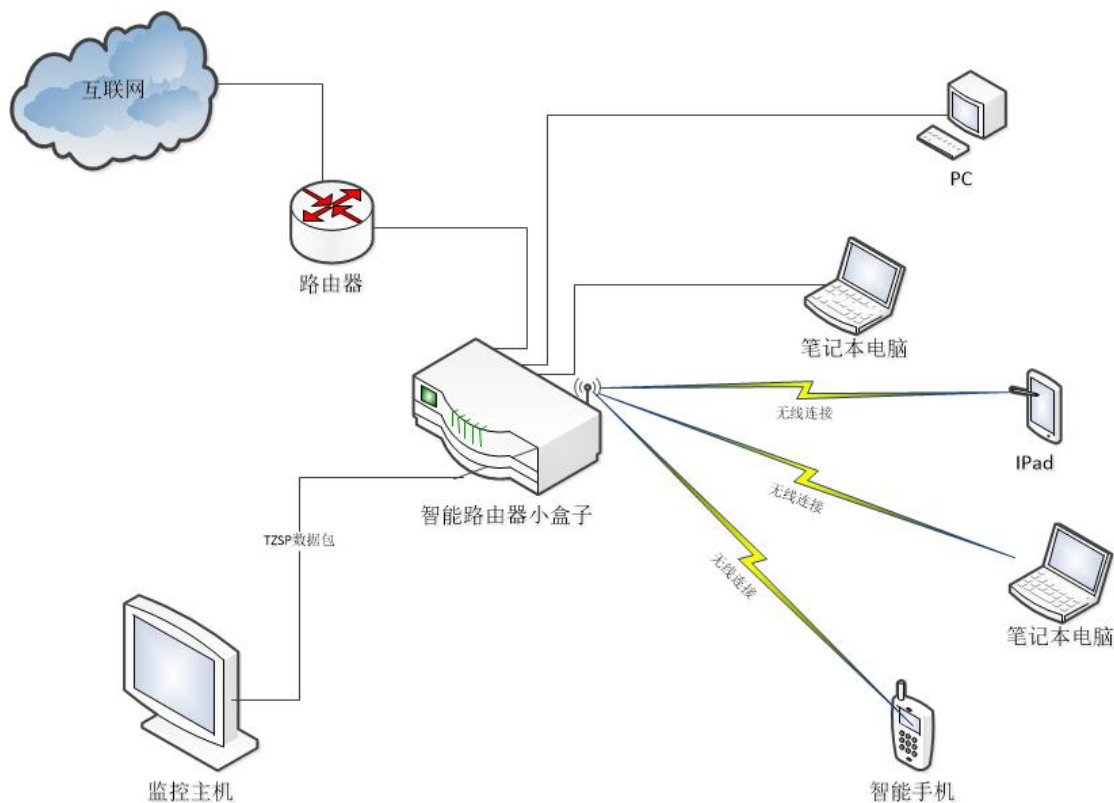


图 5-1 系统架构图

5.2 软件安装配置

5.2.1 路由器端软件安装及配置

先在 LEDE 中执行如下命令安装必要的软件包：

```
opkg update
```

```
opkg install port-mirroring
```

在 `/etc/config` 文件夹下编辑 `port-mirroring` 配置文件，其内容修改为如图 5-2 所示。

```
config 'port-mirroring'
  option 'target' '192.168.11.195'
  option 'source_ports' 'wlan0'
  option 'filter' ''
  option 'protocol' 'TZSP'
```

图 5-2 port-mirroring 配置图

其中 `target` 设置成安装有 WFilter 主机的 IP 地址，即数据包发送的目的地。将 `source_ports` 设置成 `wlan0` 代表镜像所有的无线流量，`filter` 设置为空也就是无过滤条件，`protocol` 设置为 `TZSP` 代表将所有网络流量封装成 TZSP 协

议数据包发送到目的主机^[16]。

最后执行 `/etc/init.d/port-mirroring restart`

5.2.2 监控主机端软件安装及配置

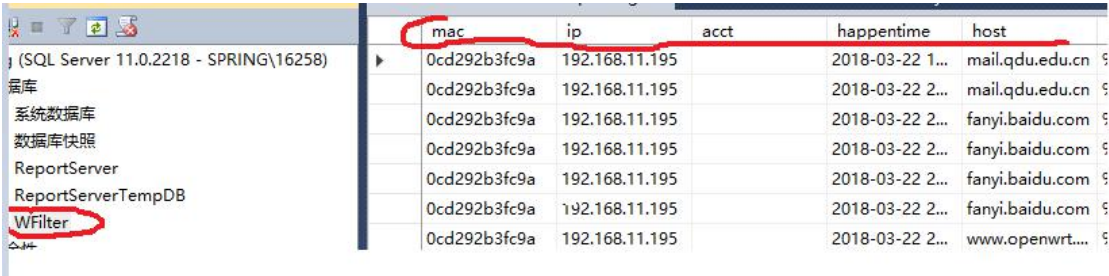
从官网下载 WFilter 软件并安装在监控主机上,其 ICF 版本支持文件数据库、Mysql 和 SQL Server 共三种数据库,此处使用 SQL Server2012,在安装 WFilter 之前确保主机已经安装其对应的数据库。

由于 SQL Server 安装的时候默认只设置了 Windows 身份登录,没有设置 sa 登录,因此需要添加 sa 用户登录。首先打开 Management Studio 菜单,在“安全性 >> 登录名”下找到 sa 用户双击,在状态栏选项下启用 sa 用户,并在常规选项栏中设置登录密码,如图 5-3 所示。



图 5-3 SQL 设置 sa 登录密码

WFilter 安装完成后,打开 SQL Server 可以看到一个名为 Wfilter 的数据库,里面的各个表存储当前局域网中用户的上网记录信息,如图 5-4 所示。



mac	ip	acct	happentime	host
0cd292b3fc9a	192.168.11.195		2018-03-22 1...	mail.qdu.edu.cn
0cd292b3fc9a	192.168.11.195		2018-03-22 2...	mail.qdu.edu.cn
0cd292b3fc9a	192.168.11.195		2018-03-22 2...	fanyi.baidu.com
0cd292b3fc9a	192.168.11.195		2018-03-22 2...	fanyi.baidu.com
0cd292b3fc9a	192.168.11.195		2018-03-22 2...	fanyi.baidu.com
0cd292b3fc9a	192.168.11.195		2018-03-22 2...	fanyi.baidu.com
0cd292b3fc9a	192.168.11.195		2018-03-22 2...	www.openwrt...

图 5-4 WFilter 数据库表

接下来,在浏览器中打开 WFilter 的管理界面,按照设置向导设置控制模式(旁路监听)、选择监听网卡和通讯网卡、设置监控模式(根据 IP 地址监控)、填写监控的 IP 段(192.168.11.0/24)等信息。设置完成后,运行配置检测选项,

查看当前配置的正确与否，如图 5-5 所示。

配置检测			
序号	检测项目		检测结果
1	监听网卡	192.168.11.195--Microsoft	✓检测正常。
2	通讯网卡	192.168.11.195--Microsoft	✓检测正常。
3	监控模式		✓检测正常。
4	IP段		✓检测正常。
5	本地服务器配置		✓检测正常。

图 5-5 监听网卡配置检测

至此，局域网主机行为监测控制系统已经搭建完毕。接下来可以根据具体的业务需求进行深层次的设计与配置，如实施网络连接封堵、违规软件实时告警等。

5.3 网络连接实时封堵

局域网在网络通信技术的推动下以势不可挡的态势向前发展，为用户提供了越来越多便利的同时也产生了一些安全和监管方面的问题。如在中小企业中，网络中的一些用户在上班期间聊天、观看下载视频、浏览不健康网页等，带来安全问题的同时严重消耗有限的带宽资源。

WFilter 使用的是旁路监控技术来封堵因特网连接，这样可以最大限度的减少对原始网络的影响，当发现有未经允许的连接时，通过发送一些类似于 reset 包等的数据包可以很容易地断开 TCP 连接。

在“监控配置 >> 封堵策略级别配置”中，设置一个节省带宽的封堵策略，用来禁止通过网页观看视频，设置界面如图 5-6 所示。

封堵策略定义

级别名称

节省带宽(示例)

谁在使用此策略 ?

级别描述

出于节省带宽的目的，禁止占带宽多的网站和应用

Web

黑白名单

协议

网页分类

带宽管理

不封堵

时间段配置

常用项

☐ 禁止网页浏览

☒ 禁止网页视频

☒ 禁止网页音频

☐ 禁止Web邮件 ?

图 5-6 节省带宽封堵策略示例

设置好封堵策略后，需要将该策略应用到局域网中的用户。在“监控设备列表”中可以看到当前局域网中的所用用户，在最后一列设置某用户应用该策略，封堵效果如图 5-7 所示。



图 5-7 音、视频网站封堵效果图

当然，可以进行更加精细化的配置，然后实施网段策略下发或批量策略下发，实现更好地利用局域网中的带宽资源。

5.4 网页关键词及违规软件告警

5.4.1 配置 SMTP 服务器

SMTP 即简单邮件传输服务，当一封邮件由源地址传送到目的地址时要使用该协议定义的规则，用它来控制信件的中转方式。SMTP 协议是 TCP/IP 协议簇的一部分，计算机在发送或中转信件时由它来找到下一个目的地。

在 WFilter 中设置 SMTP 服务器用来发送告警邮件。在“监控配置 >> 告警配置”栏中进行相关参数的配置，示例如图 5-8 所示。

SMTP配置	
管理员邮箱	<input type="text" value="17854230851@163.com"/>
SMTP服务器	<input type="text" value="smtp.163.com"/> 发送邮件服务器的IP或者名称
SMTP端口	<input type="text" value="25"/>
<input type="checkbox"/> 此服务器要求安全连接 (SSL)	
<input checked="" type="checkbox"/> 发送邮件服务器需要身份验证	
用户名	<input type="text" value="17854230851"/>
密码	<input type="password" value="....."/>
确认密码	<input type="password" value="....."/> <input type="button" value="测试"/>

图 5-8 SMTP 服务器设置

设置完成后保存配置，点击右下角的测试按钮，全部设置正确将会在自己的邮箱中收到一封测试邮件。

5.4.2 设置网络关键词告警

网页浏览管理在上网行为管理中具有举足轻重的地位，本章使用 WFilter 搭建的监控环境不仅可以记录网址，禁止一些不相关的网页浏览，还可以对网站进行分类管理，实施不同的禁用策略，做统计报表，分析当前局域网网页浏览动向与趋势等。更进一步，还可以解析用户浏览网页中的一些特定关键词，一旦网页标题或者域名里出现这些违规关键词，就会发送告警邮件给网络管理员，邮件中包含触发该关键词告警的用户的 MAC 地址等信息，方便管理员采取进一步的处置措施。

在图 5-9 所示的页面中设置想要监控的关键词，保存并启用该配置。

告警配置

关键词告警 | 违规软件告警 | 流量告警 | 其它告警 | 告警方式配置 | 告警范围配置

关键词告警: ... ☒ 启用 ☐ 不启用

扫描项目: ☒ 聊天 ☒ 邮件 ☒ 发帖 ☒ 网页

关键词列表

色情
暴力
枪支弹药

☒ 配置关键词告警邮件格式

图 5-9 关键词告警配置

在“告警方式配置”选项中指定告警邮件的发送地址，当局域网中的用户浏览的网页标题中含有配置文件中的关键词时就会触发该告警，给之前配置的管理员邮箱发送告警邮件，邮件格式如图 5-10 所示，当然，也可以自定义告警邮件的格式。



图 5-10 关键词告警邮件

5.4.3 设置违规软件告警

WFilter 使用协议识别技术来判断当前局域网中用户使用的应用软件和协议。该项技术并不是简单的通过软件的通讯端口来判定使用的协议，通过对网络通讯数据进行分析、数字信号匹配等技术手段来维护一个实时更新的协议数据库。每当局域网络中用户建立一个新的连接时，WFilter 会将该连接在协议数据库中进行搜索匹配，对一些常用的协议可以完全解析。

在“监控配置 >> 告警配置 >> 违规软件告警”页面中编辑想要实施告警的违规软件，如图 5-11 所示。

违规软件告警						
分类:	网页浏览	影响工作:	所有	协议来源:	所有	
	聊天软件	占用带宽:	所有	状态:	所有	
	邮件	风险:	所有	搜索条件:		
	文件传输					
序号	协议名称	分类	影响工作	占用带宽	风险	是否告警
117	通过加密连接接收邮件 (SMTP)	邮件	-	-	-	<input type="checkbox"/>
118	通过加密连接接收邮件 (POP3)	邮件	-	-	-	<input type="checkbox"/>
119	通过加密连接接收邮件 (IMAP)	邮件	-	-	-	<input type="checkbox"/>
120	FTP	文件传输	-	高	中	<input checked="" type="checkbox"/>
121	QQ文件传输	文件传输	中	中	中	<input checked="" type="checkbox"/>
122	Yahoo!文件传输	文件传输	中	中	中	<input type="checkbox"/>
123	AIM, ICQ文件传输	文件传输	中	中	中	<input type="checkbox"/>

图 5-11 违规软件告警配置

当局域网中的用户使用告警列表里的软件时，即可实时报警，告警统计页面如图 5-12 所示。

名称	告警类型	摘要
192.168.11.164	违规软件告警	饿了么
192.168.11.164	违规软件告警	知乎
192.168.11.195	关键词告警	网页浏览(HTTP): 暴力_词语_成语_百度
192.168.11.195	关键词告警	网页浏览(HTTP): 暴力的英文_暴力翻译_暴
192.168.11.195	关键词告警	网页浏览(HTTP): 暴力_词语_成语_百度
192.168.11.195	关键词告警	网页浏览(HTTP): 暴力的英文_暴力翻译_暴
192.168.11.164	违规软件告警	知乎

图 5-12 违规软件告警日志

总 结

设计搭建完成一个轻量级的多功能智能路由器，使得自己对树莓派、LEDE等软硬件有了更深层次的理解与体会。该系统的适用对象是小范围的局域网环境，如家庭、咖啡厅、餐厅、中小企业办公室等，能够为当前局域网中的用户提供便利的智能化操作与服务，同时能够保障整个局域网的安全性。

本环境的大部分模块与服务都已经实现完毕，但还有一些模块因开发技术有限和时间原因还没有完全实现，如路由器中的 KMS 激活服务、KCPTUN 加速服务、流量捕获模块获得的网络数据元信息入数据库等，这些模块的设计与搭建将成为未来工作的推进点。

回顾毕业设计的开发与搭建历程，虽有艰辛但在此过程中也学到了许多知识，体验到了自身的不足点，在接下来的日子里，将会继续努力学习网络安全知识并完善此智能路由器，同时将会以更加饱满的热情投入到科研工作中去。

参考文献

- [1] 姚志东. 一种网络安全审计系统的设计与实现[D]. 北京邮电大学, 2009.
- [2] 手握玻璃碴. 2017 年上半年中国无线路由器市场研究报告[EB/OL].
http://www.360doc.com/content/17/0726/13/18154423_674258795.shtml,
2017-07-26
- [3] 运营商世界网. 2017 年路由器行业报告: TP-Link 市场份额超过小米华为之
和 [EB/OL]. http://www.sohu.com/a/223814905_452858, 2018-02-24
- [4] tmomas. OpenWrt[EB/OL]. <https://openwrt.org/zh/start>, 2018-04-13
- [5] OpenWrt. OpenWrt 中文应用技术网[EB/OL]. <http://www.openwrt.org.cn/>,
2018-04-22
- [6] 玩宇宙. 更加优秀的 Openwrt 路由固件编译系统 LEDE+界面预览[EB/OL].
[http://baijiahao.baidu.com/s?id=1565446296103532&wfr=spider&for=p](http://baijiahao.baidu.com/s?id=1565446296103532&wfr=spider&for=pc)
c, 2017-04-23
- [7] Linux. OpenWrt 和 LEDE 宣布正式合并[EB/OL].
<https://www.linuxidc.com/Linux/2018-01/150237.htm>, 2018-01-11
- [8] 一起晒太阳. Raspberry Pi Plotter:由 CD 和计算机零件组装[EB/OL].
[http://tech.feng.com/2015-05-18/Raspberry-Pi-Plotter-by-CDS-and-c](http://tech.feng.com/2015-05-18/Raspberry-Pi-Plotter-by-CDS-and-computer-parts-for-assembly_614513.shtml)
omputer-parts-for-assembly_614513.shtml, 2015-05-01
- [9] 戴晓天. 树莓派 3 代介绍及历代树莓派比较[EB/OL].
<http://www.yfworld.com/?p=3436>, 2018-04-22
- [10] 小五义. 树莓派学习笔记[EB/OL].
<http://www.cnblogs.com/xiaowuyi/p/3968959.html>, 2014-09-12
- [11] hy1397471. 内存卡上那个 u3 和 c10 都是啥意思啊? [EB/OL].
<https://zhidao.baidu.com/question/199716689890945165.html>,
2017-11-24
- [12] Apprcn. SD Formatter - 正确格式化 SD 卡[EB/OL].
<http://www.apprcn.com/sd-formatter.html>, 2014-04-11

- [13] whatday. 什么是交叉编译 [EB/OL].
<https://blog.csdn.net/whatday/article/details/73930604>, 2017-06-30
- [14] mars1743. 什么是交叉编译, 为什么要使用交叉编译?[EB/OL].
<https://blog.csdn.net/mars1743/article/details/20492285>,
2014-03-04
- [15] m浩瀚孤鸿. Aria2 一个命令行下运行、多协议、多来源下载工具[EB/OL].
<https://blog.csdn.net/myweishanli/article/details/25119987>,
2014-05-06
- [16] m 浩瀚孤鸿. aria2 配置示例[EB/OL].
<https://blog.csdn.net/myweishanli/article/details/42220925?locationnum=14>, 2014-12-28
- [17] 飛星小星星. 解决 Win10.4 无法访问 samba 协议小米路由盘修复方法
[EB/OL]. <http://mini.eastday.com/a/171031011513205.html>,
2018-04-25
- [18] xp5xp6. Ngrok 让你的本地 Web 应用暴露在公网上[EB/OL].
<https://blog.csdn.net/xp5xp6/article/details/53393488>, 2016-11-29
- [19] NKCZG. ubuntu 系统 debootstrap 的使用[EB/OL].
<http://www.cnblogs.com/NKCZG/p/4192557.html>, 2014-12-29
- [20] 徐美华. 局域网主机上网行为监测系统的设计与实现[D]. 华北电力大学(河北), 2006.
- [21] 伍丽芳. 局域网安全行为监控系统的设计与实现[D]. 电子科技大学, 2012.