MikroTik

RouterOS V5

中文教程

前 言

很多人认为 RouterOS 是美国人开发，其实这套软件来至东欧的小国家拉脱维亚，公司名称是 MikroTik（Mikrotikls SIA），官方网站是（www.mikrotik.com/www.RouterOS.com）。最初 MikroTik 开发 RouterOS 目的是解决无线局域网传输问题（WLAN），后来通过不断扩展功能实现了各种功能的应用。而在国内最大的使用人群是网吧和小区宽带、这个和国外的情况有点差别，在国外 RouterOS 不仅用于解决路由管理，大多应用在 WLAN 的覆盖和传输，RouterOS 在基于 802.11abgn 协议上的高带宽传输有自己的明显优势，特别是他独有的 Nstrem 和 Nv2 协议。

RouterOS 从功能和性能方面已经超过了许多中端路由器，随着 RouterOS 在国内越来越多的人接受，不过从最开始的网吧多线路与流控和小区宽带，后来的 VPN 方案解和企业管理，还是 RouterOS 的 WLAN 无线应用，都在不断冲击整个网络行业！甚至在 2005 年后出现了几款类似的软件路由系统，虽然从个别方面比起 RouterOS 优越，但整体上仍然难以超越。

我从 2003 年底接触 RouterOS 近 8 年多了，这几年,积累了不少 RouterOS 的经验，整体来说 RouterOS 比其他路由器有他的长处，如对一些功能提供较开放的设置，特别脚本编写，灵活的处理各种功能，让管理员能对路由器进行一些弹性的控制！当然开放的系统，也对管理员有一定的技术要求。

这套教程是我通过整理各种资料，并结合自己的经验编辑而成，可能也有不足的地方希望大家提出各自的见解，毕竟 RouterOS 的应用是比较灵活，在不同硬件平台上反映也有差别。教程从最基础的安装、如何登陆配置到深入多线路策略和流量控制，以及各种服务器、WLAN 配置和脚本编写都进行了讲解，也分析 RouterOS 工作流程、多线路路由和负载均衡、Simple Queue、PCQ 和 HTB 等原理。如果你是个初学者可以从头开始学习 RouterOS，如果你想提高自己的 RouterOS 水平，同样能学习到更多的功能。

我希望通过这份教程能对准备和正在使用 RouterOS 朋友起到帮助，我从 2006 年第一次在青岛做培训时就逐步地接触了较早几个玩 RouterOS 的朋友，他们中间有很多很好的想法，对 RouterOS 大相径庭，但也有各自的认识，当然这个从技术和商业上都有，我一共编写和整理了 3 份教程 RouterOS 中文教程，WLAN 无线操作手册和 Script 脚本教程，以及各种培训资料。

RouterOS 也被一些客户用在较大的网络上，在 2007 年某地长宽使用 16 台 RouterOS 架设了 PPPoE 服务器，每 2 台组成一组，当时注册用户达到 23000 人，并发在线 12650 人，带宽 2.4G 这个是我已知最大的一次应用，当时对那边工程师这样大规模应用很佩服！那时是 2.9 的版本，当然现在的 v5.0 在发挥多核心处理有很高好的提升，同样 CPU 的不断发展也促使 RouterOS 的性能在不断提升，当然我希望还有很大规模的应用。

MikroTik 发展历程：

- 1995 年开始基于 WISP 方案解决；
- 1996 年将无线 ISP 计划推向全球；
- 1997 年开发 RouterOS 的软件，基于 Intel（PC）解决路由技术问题；
- 2002 年拥有 RouterBOARD 的硬件。

该教程对 WLAN 的基础知识和常见问题都做了普通的讲解，对现在主流的 WLAN 网络的常见连接方式如：点对点、点对多点、中继、WDS 和 Mesh 网络做了讲解和配置说明，能让阅读者对 MikroTik 产品的 WLAN 应用和配置等得到充分了解。

构建基于 MikroTik 产品的 WLAN 网络，需要基于对 MikroTik RouterOS 一定的掌握，如对 RouterOS 的基本操作和 IP 网络协议的了解。

内容如有更新，恕不通知。

版本: V5.0
适用于: RouterOS v3.x，v4.x，v5.x
编: 余松
E-mail: yus_sds@yahoo.com.cn

目 录

应用说明

主要特征

RouterOS 基于 Linux2.6 内核，以下是 RouterOS 功能与简介：

等级 / 功能	Level 0	Level3	Level 4	Level 5	Level 6
升级	24 小时	4.x	4.x	5.x	6.x
无线 AP	24 小时	不支持	支持	支持	支持
无线桥接和客户端	24 小时	支持	支持	支持	支持
RIP，OSPF，BGP 协议	24 小时	支持	支持	支持	支持
EoIP 隧道在线用户	24 小时	1 条	无限制	无限制	无限制
PPTP 隧道在线用户	24 小时	1 条	200	无限制	无限制
PPPoE 隧道在线用户	24 小时	1 条	200	500	无限制
L2TP 隧道在线用户	24 小时	1 条	200	无限制	无限制
OVPN 隧道在线用户	24 小时	1 条	200	无限制	无限制
SSTP 隧道在线用户	24 小时	1 条	200	无限制	无限制
Hotspot 认证在线用户	24 小时	1 条	200	500	无限制
VLAN	24 小时	1 条	无限制	无限制	无限制
P2P 防火墙规则	24 小时	1 条	无限制	无限制	无限制
NAT 规则	24 小时	无限制	无限制	无限制	无限制
Radius 客户端	24 小时	支持	支持	支持	支持
Queue 流量控制规则	24 小时	无限制	无限制	无限制	无限制
Web 代理	24 小时	支持	支持	支持	支持
User Manager 在线用户	24 小时	10	20	50	无限制

x86 硬件支持

- AMD、Intel、VIA 和其他兼容的 x86 平台
- SMP – RouterOS 3.0 后兼容的多核心处理器和多处理器（RouterOS v5.x 对多核心处理做更好的优化）
- 内存：最小 32MB，最大：RouterOS v2.9 仅支持 1G 内存，RouterOS v3.0 后支持 2G 内存
- 存储：IDE、SATA,、CF 存储卡、USB、DOM 闪存盘和 SCSI（5.x 版本），最小需要 64MB 空间，建议系统硬盘不大于 80G
- Linux v2.6 内核支持的扩展槽 PCI、PCI-e、PCI-X

MIPS 硬件

- **支持系统** – 4kc RouterBOARD 500 (532, 512 和 511)与 RouterBOARD 100 (133、133c、150、192)
- **支持系统** – 24kc RouterBOARD 400(411/411A/411AH、433/433AH/433UAH、450/450G、493/493AH)
- **支持系统** – 24kc RouterBOARD 700(711、711A、750/750G、750UP、751、751G)
- **RAM** – 最小 16MiB
- **ROM** – 板载 NAND 驱动，最小 64Mb

PPC 硬件

- RouterBOARD1000、RouterBOARD1100、RouterBOARD800、RouterBOARD600、RouterBOARD333
- RouterBOARD1100AH, RouterBOARD1100AHX2, RouterBOARD1200

安装

- Netinstall: 网络安装，基于 PXE 或 EhterBoot 启用的网卡的网络安装
- Netinstall: 在 windows 中安装到从属硬盘，比如 U 盘安装
- CD 镜像文件安装

配置

- MAC 地址访问与配置
- WinBox - 基于窗口化配置的 GUI 图形工具
- Web 接口配置工具（webfig 和 webbox）
- 强大的命令行配置接口，集成脚本编辑功能，可通过本地终端、console 管理、telnet 和 ssh 访问配置
- API - 能创建你自己的配置和监测应用程序

备份与恢复

- 支持二进制配置备份文件 Binary configuration backup saving and loading
- 通过 Exprot 和 import 可生成读写的文本格式

Firewall

- 状态过滤功能 Statefull filtering
- 源和目标 NAT
- NAT 助手(h323, pptp, quake3, sip, ftp, irc, tftp)
- 内部链接状态，路由和数据包标记
- 防火墙过滤，对 IP 地址和地址范围，端口和端口范围，IP 协议，DSCP 等其他功能
- 支持地址列表创建

- 专业的 Layer7 匹配器
- IPv6 支持
- PCC – 每次连接分类器，用于负载均衡配置
- Nth – 第 N 次连接数据标记，用于连接排序和负载均衡

路由

- 静态路由
- 虚拟路由转发（Virtual Routing and Forwarding - VRF）
- 路由策略
- 接口路由
- ECMP 路由策略
- IPv4 动态路由协议：RIP v1/v2, OSPFv2, BGP v4
- IPv6 动态路由协议：RIPng, OSPFv3, BGP
- 双向转发检测 (BFD)

MPLS

- 支持 IPv4 静态标记绑定
- IPv4 的标记分布协议
- RSVP 传输工程隧道
- VPLS MP-BGP 自动探测和信号发送
- MP-BGP 基于 MPLS IP VPN

VPN

- Ipsec – 隧道和传输模式，证书或者 PSK，AH 和 ESP 安全协议，RB1000 支持硬件加密
- 点对点隧道(OpenVPN、PPTP、PPPoE、L2TP、SSTP)
- 高级 PPP 功能(MLPPP、BCP)
- 简单隧道(IPIP、EoIP)
- 支持 6to4 隧道(IPv6 基于 IPv4 网络)
- VLAN – IEEE802.1q 虚拟局域网，支持 Q-in-Q 模式
- 基于 MPLS 的 VPN

Wireless

- IEEE802.11a/b/g 无线 AP 和客户端
- 支持 IEEE802.11n
- Nstreme 和 Nstreme2 私有协议
- 无线分布式系统(WDS)
- 虚拟 AP
- 安全支持 WEP, WPA, WPA2
- 访问控制列表
- 无线客户漫游
- 支持 WMM
- HWMP+ 无线 Mesh 协议
- MME 无线路由协议

DHCP

- 支持每个接口的 DHCP 服务
- DHCP 客户端和中继
- 静态和动态 DHCP 租约
- 支持 RADIUS
- 自定义 DHCP 选项

Hotspot

- 即插即用网络访问功能
- 通过 web 对本地网络客户验证
- 用户账户记录
- 支持 RADIUS 验证与计费

QoS

- 令牌桶(HTB)QoS 体系，定义优先级
- 简单快速的 QoS 工具(Simple queues)
- 动态客户端速率均衡(PCQ)

Proxy

- HTTP 缓存代理服务器
- 透明 HTTP 代理
- 支持 SOCKS 协议支持
- DNS 静态条目
- 支持缓存到一个指定的驱动器
- 支持父级代理
- 访问控制列表
- 缓存列表

工具

- Ping, traceroute
- 带宽测试 Bandwidth test, ping flood
- 数据包 sniffer 工具, torch 工具
- Telnet, ssh
- E-mail 和 SMS 短信发送工具
- 自动脚本执行工具
- 文件 Fetch 工具

其他功能

- 桥接 Bridging - 生成树协议(STP, RSTP), 桥接防火墙与 MAC nat 功能
- DDNS 工具
- NTP 客户端/服务器和 GPS 系统
- VRRP 虚拟冗余路由协议
- SNMP
- M3P - MikroTik 数据包封装协议
- MNDP - MikroTik 邻居探测协议，支持 CDP（思科探测协议）
- 支持 RADIUS 验证与计费

- TFTP 服务器
- 支持 Synchronous 接口(仅支持 Farsync 卡)
- Asynchronous - 串口 PPP 拨号 dial-in/dial-out
- 支持 ISDN

操作配置

RouterOS 提供了强大的命令配置接口。你同样可以通过简易的 Windows 远程图形软件 WinBox 管理路由器。也提供了网页配置的 Webfig，webfig 提供了类似 winbox 的配置功能上，主要特征：

- 完全一至的用户接口
- 运行时配置和监控
- 支持多个连接访问
- 用户策略配置
- 活动历史记录，undo/redo 操作
- 安全模式操作
- Scripts 能事先安排执行时间和执行内容，脚本支持所有的命令操作。

路由器可用通过下面的接口进行管理：

- **本地 teminal console** - PS/2 或 USB 键盘和 VGA 显示卡进行控制
- **Serial console** – 任何 (默认使用 COM1) RS232 异步串口，串口默认设置为 9600bit/s, 8 data bits, 1 stop bit, no parity, hardware (RTS/CTS) flow control。
- **Telnet** – telnet 服务默认运行在 TCP 端口 23
- **SSH** - SSH (安全 shell) 服务默认运行在 TCP 端口 22
- **MAC Telnet** - MikroTik MAC Telnet 协议被默认启用在所以类以太网卡接口上。
- **Winbox** – Winbox 是 RouterOS 的一个 Windows 远程图形管理软件，使用 TCP 端口 8291（3.0rc13 版本后支持修改 winbox 的端口），同样也可用通过 MAC 地址连接。

第一章：RouterOS 基本操作

1.1 RouterOS 安装介绍

1、使用带 ISO 的镜像文件通过光盘引导安装（用于 x86 系统）； 支持 AMD、Intel、VIA 以及其他 X86 系统，硬盘支持 IDE、SATA 硬盘接口

2、使用 U 盘安装基于 X86（限 3.0 版本后）

3、使用 netinstall 网络安装程序（主要用于 RouterBOARD）；RB100、RB300、RB500、RB400、RB600、RB700、RB800、RB1000 系列

CD 光盘安装

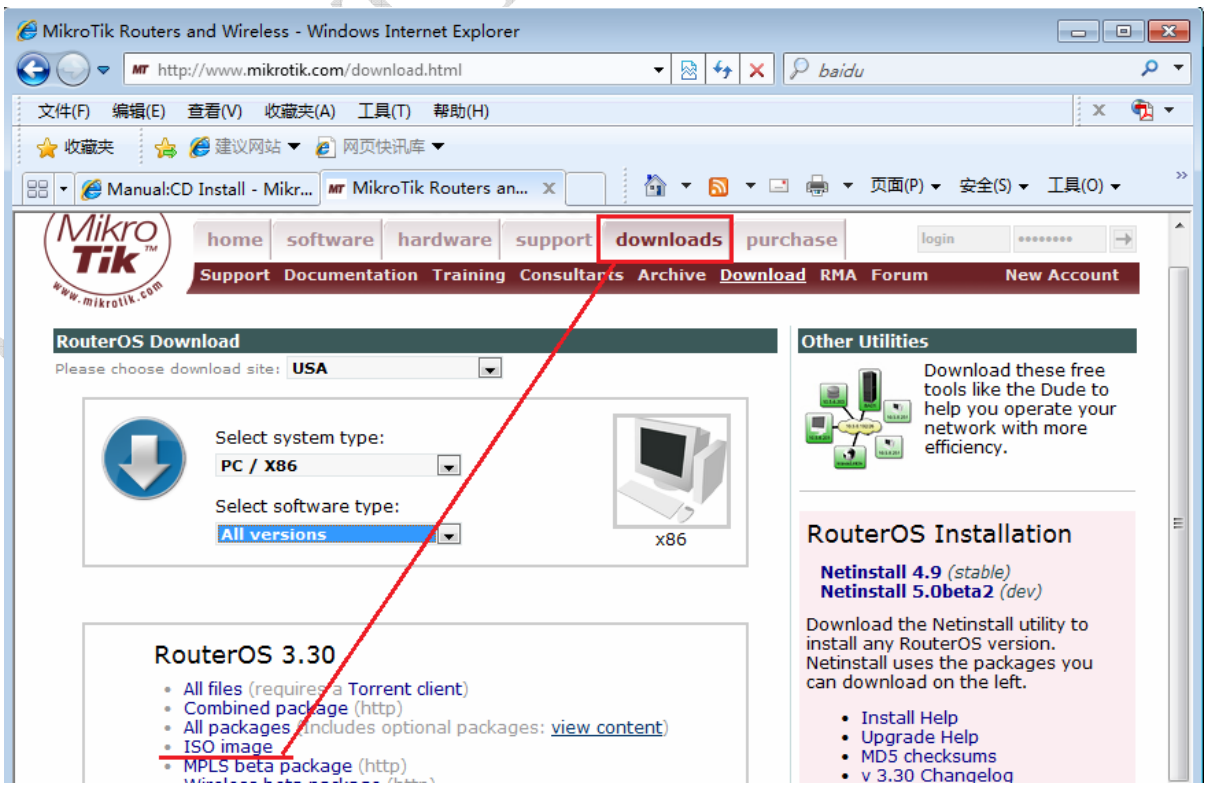
通过 CD 安装可以将 MikroTik RouterOS 安装到基于 PC 的 x86 的硬件上，有些 PC 硬件不支持网络安装（Netinstall）（所有 RouterBOARD 重装都必须通过 Netinstall）。

CD 安装要求：

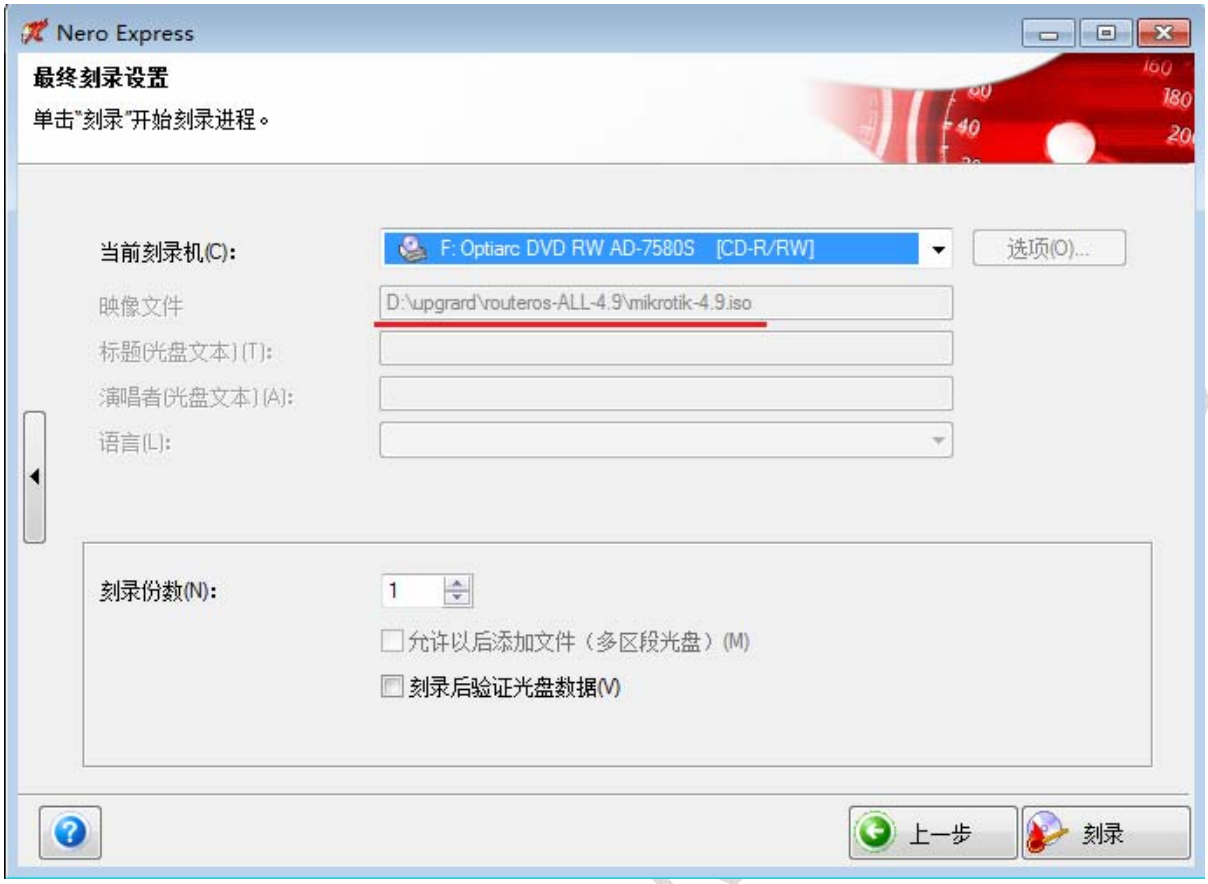
- 基于 PC 的 x86 硬件
- CD-ROM
- MikroTik RouterOS ISO 镜像文件
- 安装 CD 通过刻录软件刻录

准备 MikroTik RouterOS 安装

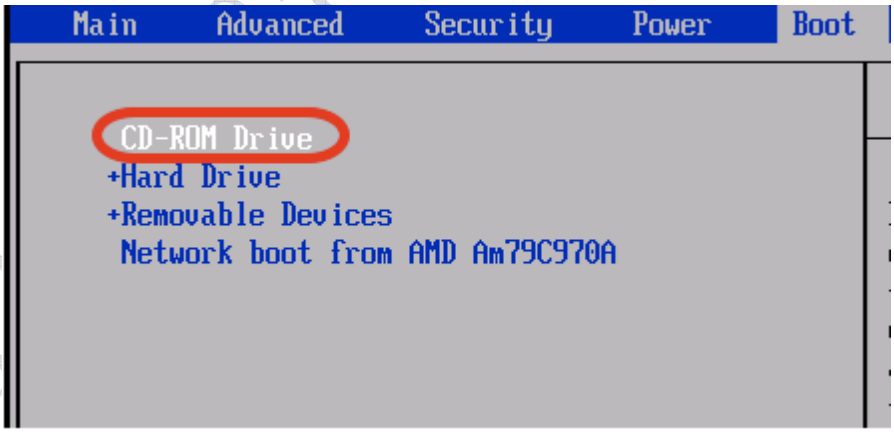
1. 下载 MikroTik 的镜像文件下载页面,



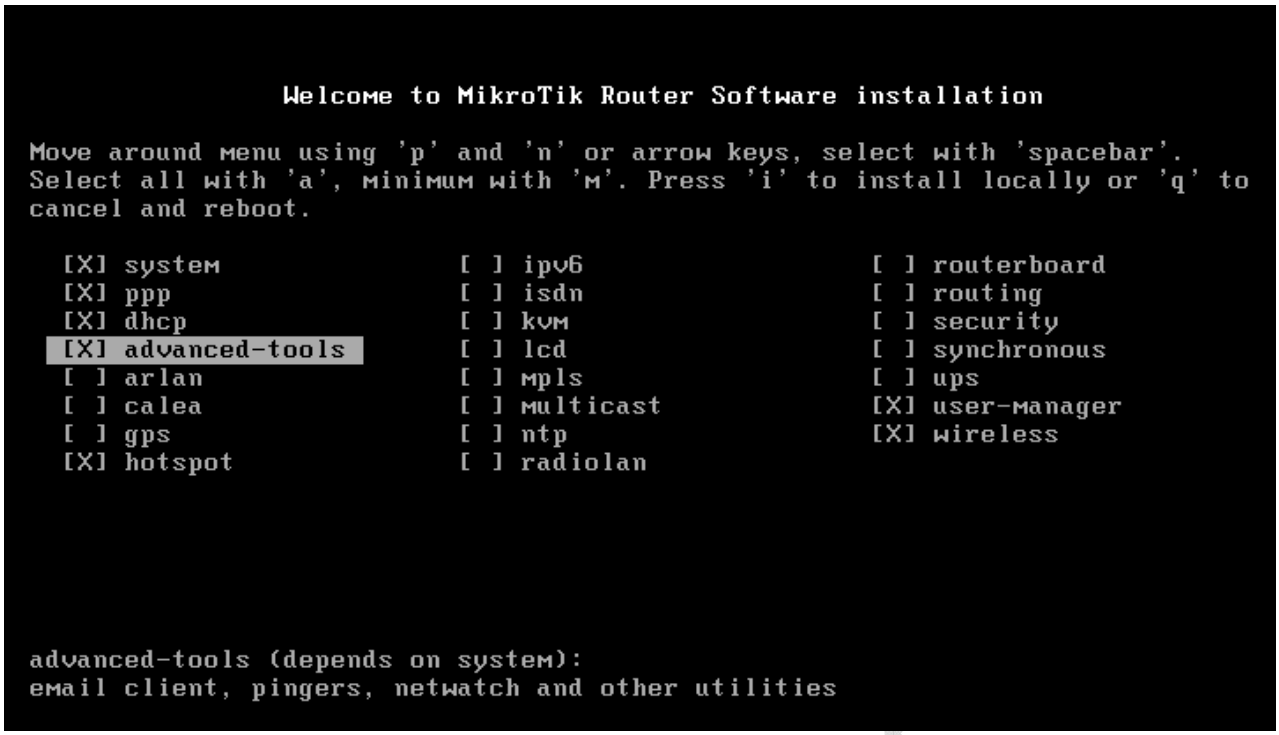
2.把 ISO 镜像下载到硬盘后，你需要通过 PC 的 CD-ROM 和刻录程序，刻录 CD，通过



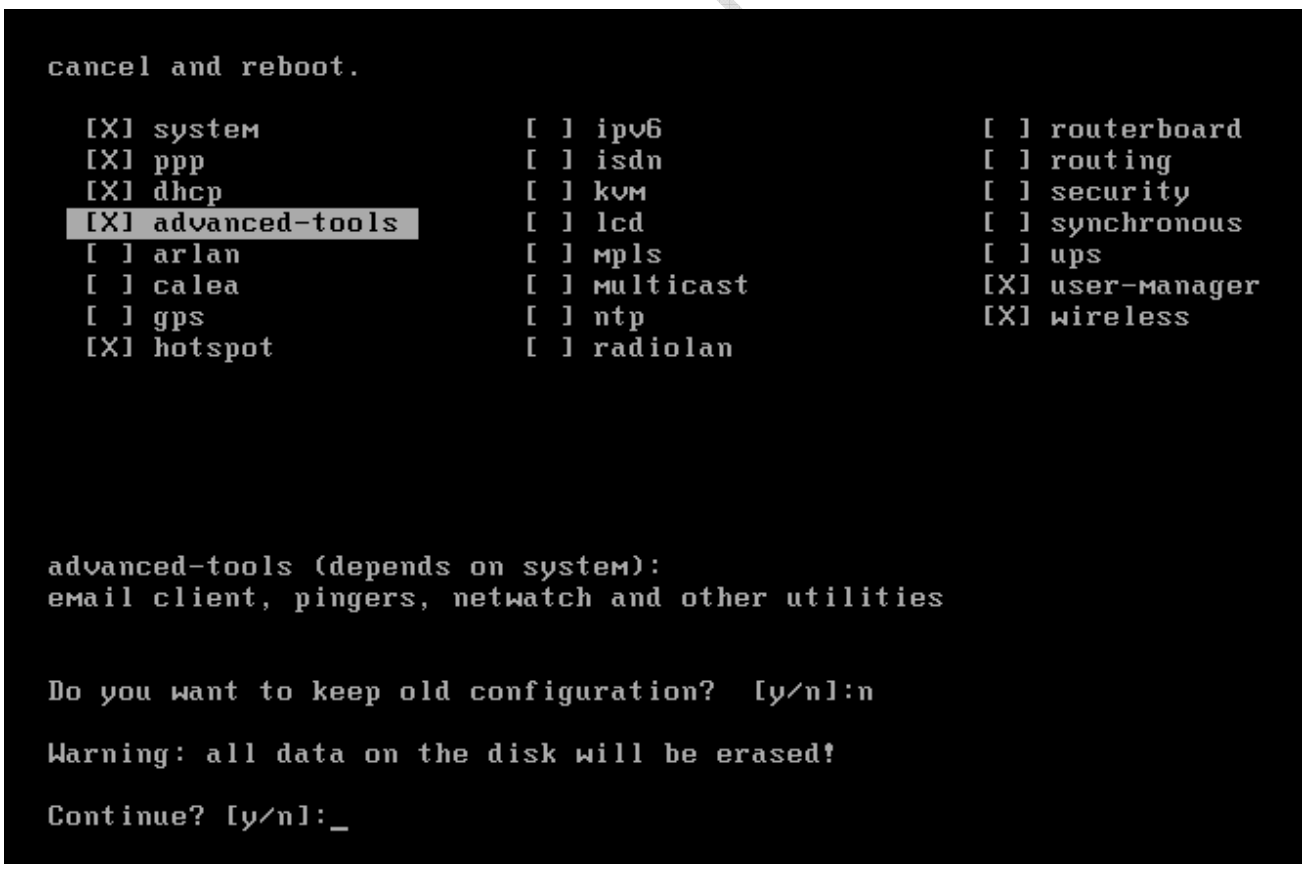
3. 刻录好 CD 后，将准备好的用于 RouterOS 运行的 PC 装好光驱，并设置 BIOS 为 CD-ROM 引导，让后放入 CD 进行安装：



4. PC 将从 RouterOS CD 光盘引导启动后，安装并选择相应的功能包



5. 选择你需要安装的功能包，使用“空格”选择功能包，通过“a”可以选择所有功能包，或者“m”选择最小安装，按“i”开始安装 RouterOS，如果你之前在同一台 PC 上安装过 RouterOS，你可以复位和保存设置，提示如下“**Do you want to keep old configuration?**”如果选择“**n**”复位设置，选择“**y**”保存之前配置



6. 安装完成后，路由器将提示你重启路由器

```
advanced-tools (depends on system):
email client, pingers, netwatch and other utilities


Do you want to keep old configuration?  [y/n]:n

Warning: all data on the disk will be erased!

Continue? [y/n]:y

Creating partition........
Formatting disk...

installed system-5.0
installed wireless-5.0
installed user-manager-5.0
installed hotspot-5.0
installed advanced-tools-5.0
installed dhcp-5.0
installed ppp-5.0
Checking disk integrity...

Software installed.
Press ENTER to reboot
_
```

7. MikroTik RouterOS 安装成功后，不要忘记将将 CD-ROM 引导修改为硬盘引导

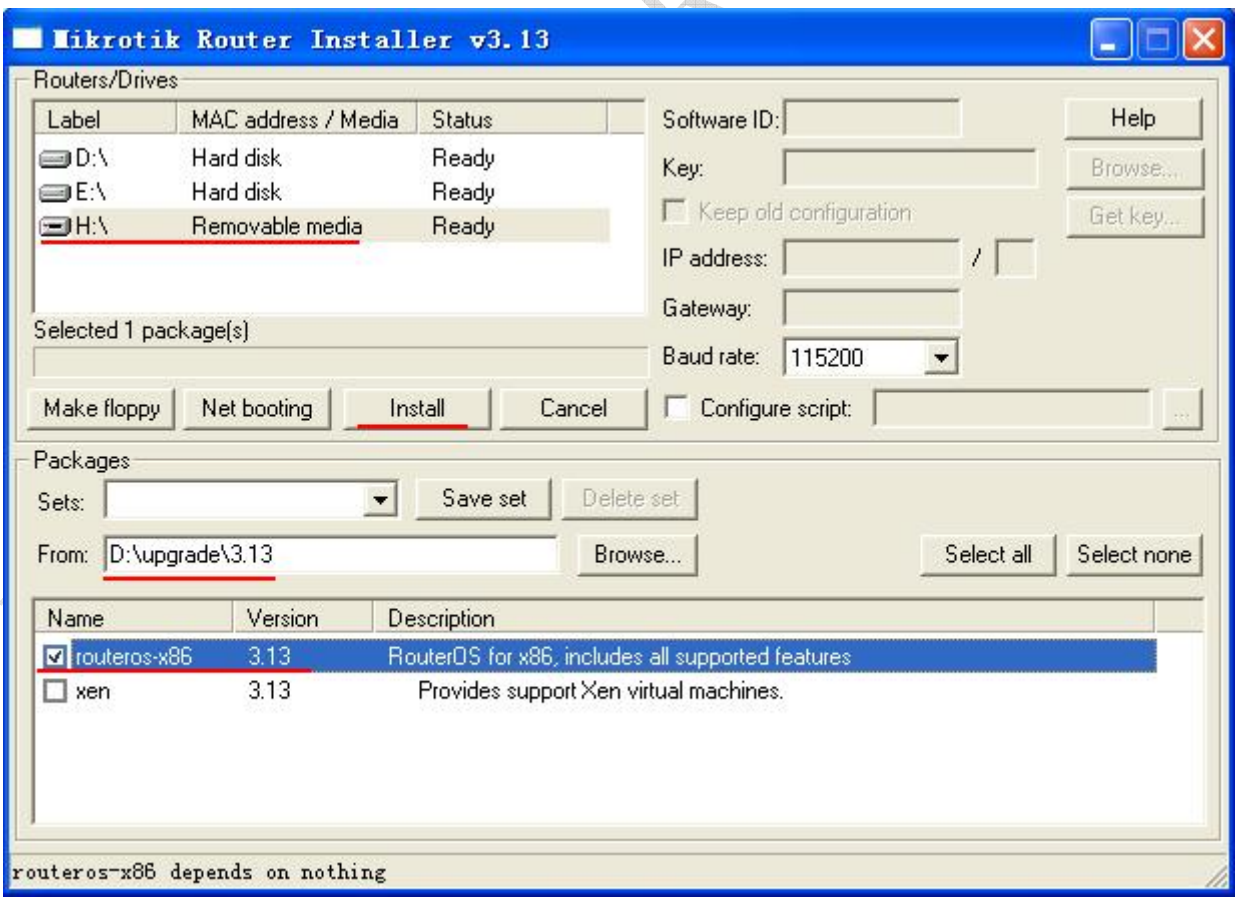
8. 启动后 RouterOS 提示登录信息，登录默认账号是 admin，没有密码

10. 最后，新装的 RouterOS 需要注册许可，否则只能使用 24 小时，在下面我们可以看到 **software-id** ，你可以与当地分销商联系购买许可,

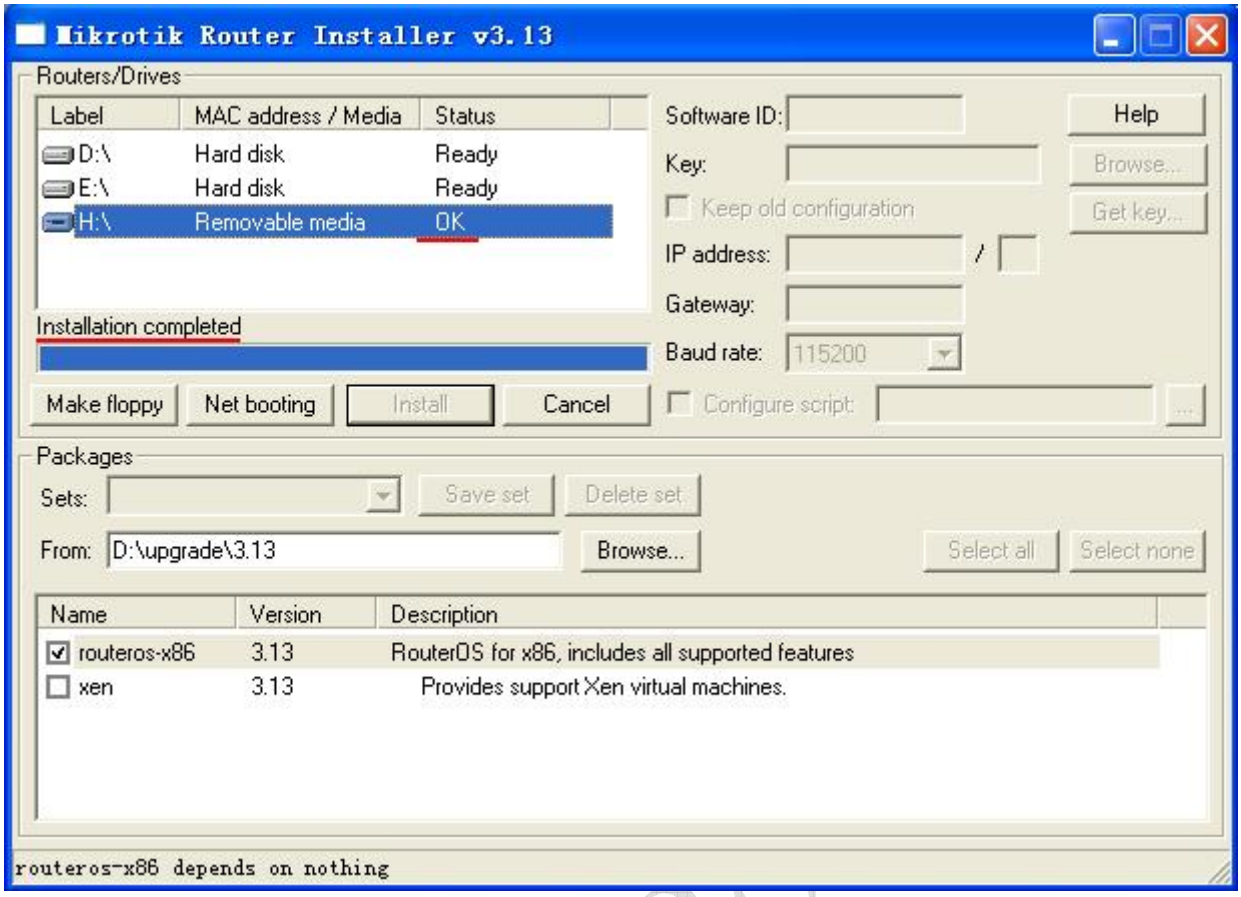
```
  MMM       MMM        KKK                          TTTTTTTTTTT      KKK
  MMMM     MMMM        KKK                          TTTTTTTTTTT      KKK
  MMM MMMM MMM   III   KKK  KKK   RRRRRR     OOOOOO      TTT      III   KKK  KKK
  MMM  MM  MMM   III   KKKKK     RRR  RRR  OOO  OOO     TTT      III   KKKKK
  MMM       MMM   III   KKK KKK   RRRRRR    OOO  OOO     TTT      III   KKK KKK
  MMM       MMM   III   KKK  KKK  RRR  RRR   OOOOOO      TTT      III   KKK  KKK

  MikroTik RouterOS 5.0 (c) 1999-2011       http://www.mikrotik.com/


UPGRADE NOW FOR FULL SUPPORT
----------------------------
FULL SUPPORT benefits:
- receive technical support
- one year feature support
- one year online upgrades
    (avoid re-installation and re-configuring your router)
To upgrade, register your license "software ID"
 on our account server www.mikrotik.com

Current installation "software ID": ZYBI-N3R2

Please press "Enter" to continue!
```

USB 安装

U盘安装需要使用3.0以上版本netinstall软件，将U盘插入一台Windows电脑的USB接口后，启动Netinstall软件，选择 RouterOS-X86 安装包：



通过 Netinstall 安装 RouterOS 到 U 盘上：



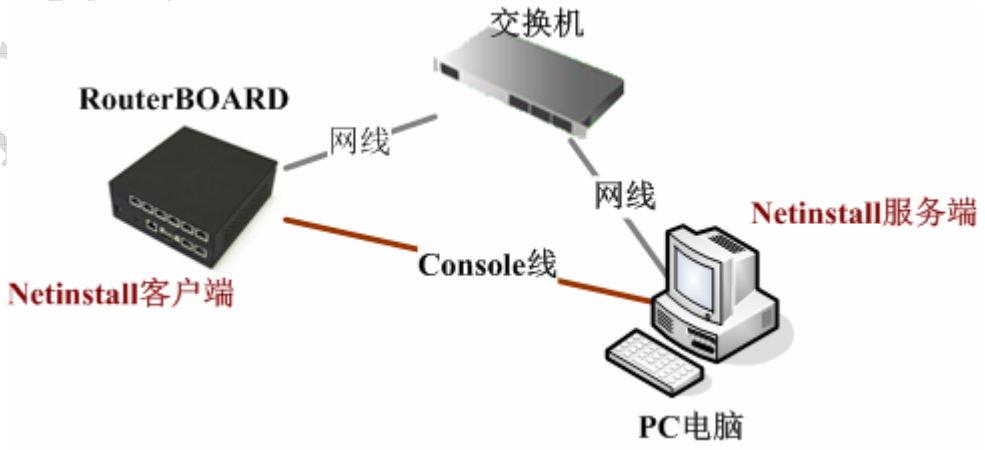
最后取出 U 盘，将 U 盘插入到 PC 上，并设置电脑的 BIOS 通过 USB 引导启动，启动后可以看到系统正在安装。

NetInstall 安装和复位 RouterRoard

安装和复位 RouterBOARD

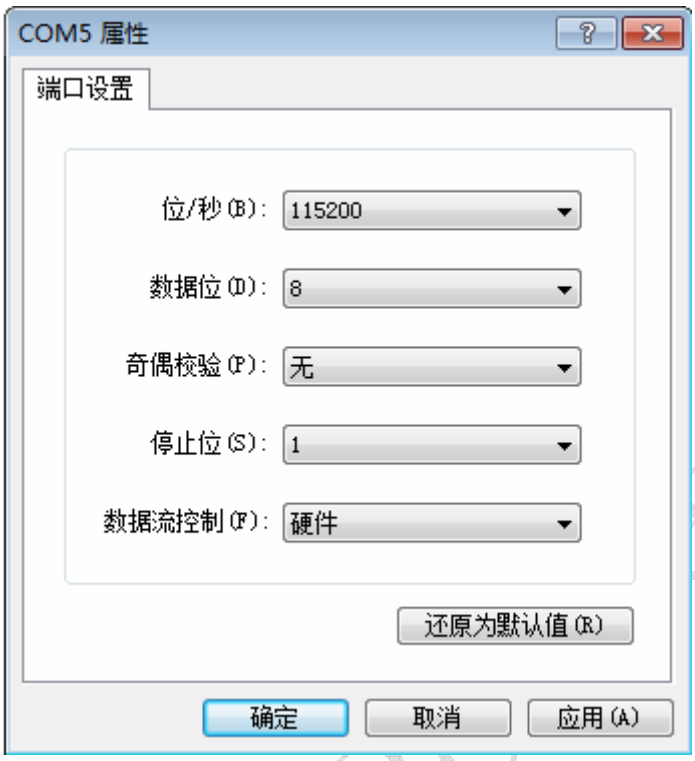
这个事例将介绍如何一步一步在一个 RouterBOARD 上重新安装 RouterOS 软件，同样在你丢失了 RouterBOARD 登录密码后，也可以通过该方法复位 RouterOS。

1. 使用 ether1 网卡通过交换机或者直接通过网线连接到 RouteBoard 上，然后在使用串口线和 RouterBoard 相联接。



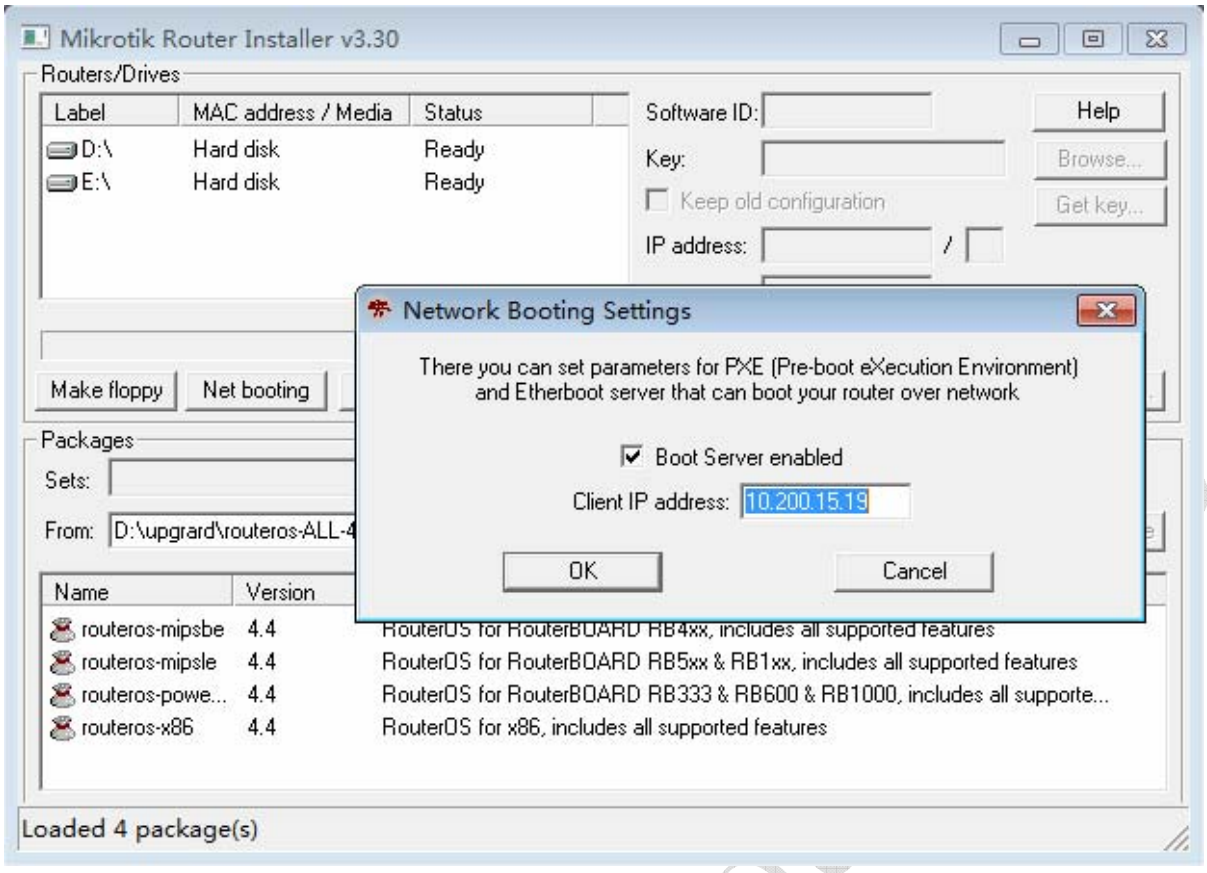
在你的电脑上运行 NetInstall 程序 ，确定 RouterOS 软件包(*.npk 文件)在你本地磁盘上。

2. 设置好 Windows 工作站的超级终端连接，每秒位数为 115200，（如果是 PC 通过串口连接每秒位数为 9600）其他参数为系统默认值（如果你的 vista 或者 WIN 7 操作系统，可以直接从 windows xp 里将超级终端程序拷贝到 vista 和 win 7 系统上，文件分别是 hypertrm.dll 和 hypertrm.exe）：



3. 启动 Netinstall 程序，打开 Net Booting 按钮，在启用 Boot Server 功能，由 Netinstall 做网络安装引导服务端。安装 Netinstall 本机的 IP 地址是 **10.200.15.18/24**，那需要给 Boot Server 客户端的分配一个 IP 地址段，用于临时分配给 RouterBoard 的 IP 地址，该事例的地址为 **10.200.15.19**）。

注意：网线连接的是 RouterBoard 的 ether1 网卡接口，不然无法获取引导信息。



4. 设置 RouterBoard 从以太网卡引导，首先进入 RouterBoard BIOS (重起 RouterBOARD 后，在超级终端下出现提示时 press any key... 后按任意键进入 BIOS 设置):

```
RouterBoard 450G

CPU frequency: 680 MHz
  Memory size: 256 MB

Press any key within 2 seconds to enter setup

RouterBOOT-2.20
What do you want to configure?
   d - boot delay
   k - boot key
   s - serial console
   o - boot device
   u - cpu mode
   f - cpu frequency
   r - reset booter configuration
   e - format nand
   g - upgrade firmware
   i - board info
   p - boot protocol
   t - do memory testing
   x - exit setup
your choice:
```

进入 BIOS 后你可以看到可用命令的列表，设置引导设备，选择“boot device”，按“o”键可以进入

```
your choice: o - boot device

Select boot device:
   e - boot over Ethernet
 * n - boot from NAND, if fail then Ethernet
   1 - boot Ethernet once, then NAND
   o - boot from NAND only
   b - boot chosen device
your choice:
```

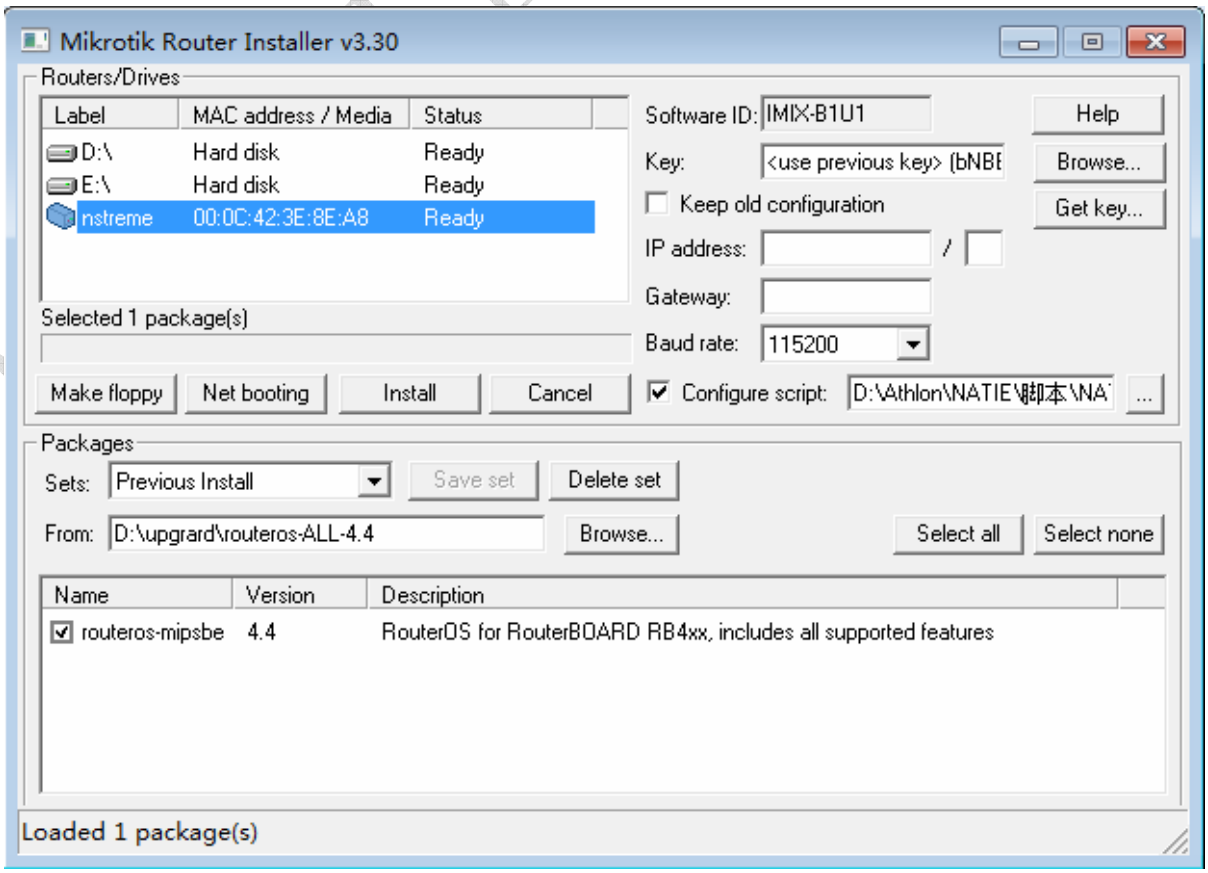
按“e”键，是选择从以太网卡引导 RouterBoard：

```
Select boot device:
   e - boot over Ethernet
 * n - boot from NAND, if fail then Ethernet
   1 - boot Ethernet once, then NAND
   o - boot from NAND only
   b - boot chosen device
your choice: e - boot over Ethernet
```

当选择完成后，返回 RouterBoard BIOS 首页，选择“x“，退出 BIOS。路由器将会重启。

5. 在启动时，RouterBoard 将试着从以太网卡上去寻找引导信息。如果成功，运行 Netinstall 的 Windows 工作站，将会分配给 RouterBoard 一个 IP 地址。在上面过程完成后，RouterBoard 将等待安装信息。

   在 Windows 上，将会出现一个新的设备列表，显示当前连接的 RouterBoard 设备。

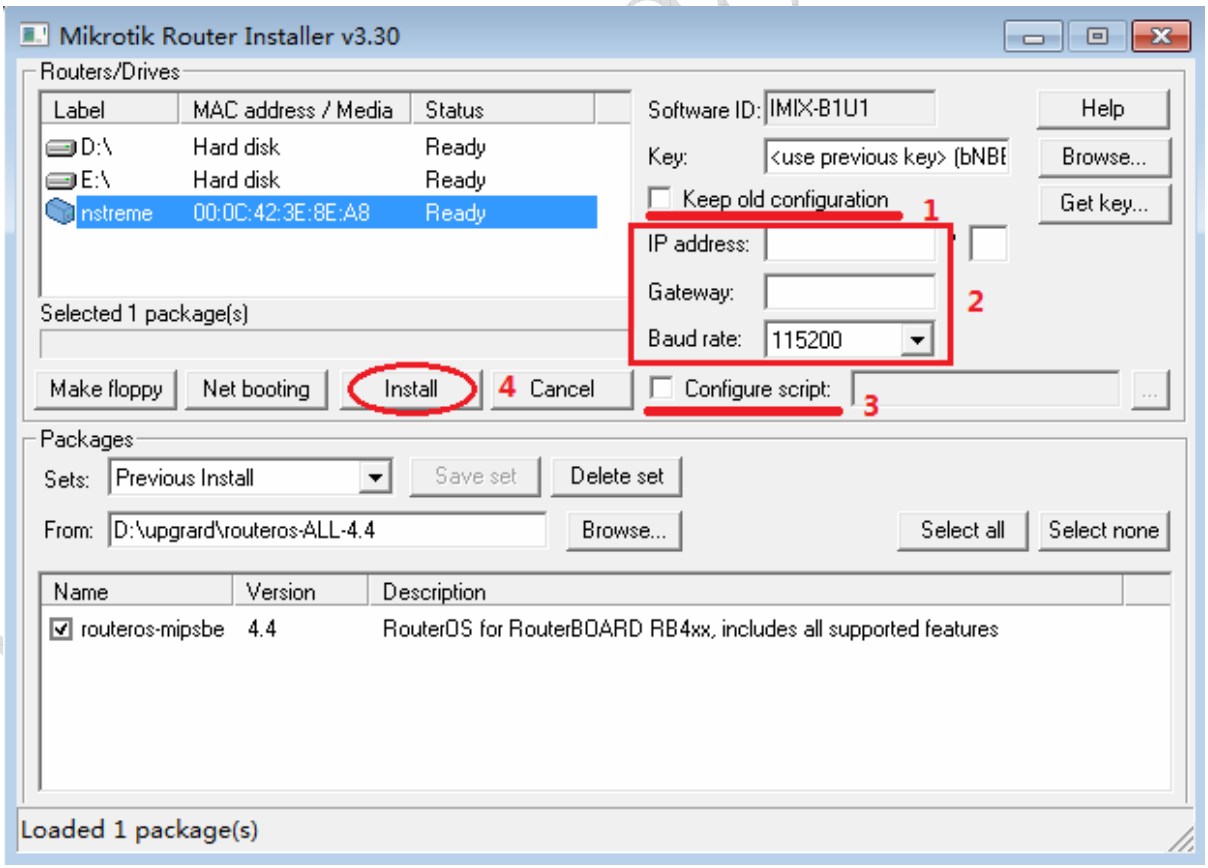


这时 Netinstall 会自动识别 RouterBOARD 的型号，并在指定的目录下查找对应的功能包，Netinstall 自动为 RB450G 查找到了合适的 RB4xx 的安装包，你也可以手动指定安装功能包路径。

在超级终端显示如下等待安装信息

```
Welcome to MikroTik Router Software remote installation
Press Ctrl-Alt-Delete to abort

mac-address: 00:0C:42:3E:8E:A8
mac-address: 00:0C:42:3E:8E:A9
mac-address: 00:0C:42:3E:8E:AA
mac-address: 00:0C:42:3E:8E:AB
mac-address: 00:0C:42:3E:8E:AC

software-id: IMIX-B1U1 key:
bNBBSe/onQwGhhk/RW1XBfWTVeOnnja/UsnbuTgcDVckt7fl5zf0Iobz03GWXjCr6vUQ34XSfB9pdGmX
czOmEA==

Waiting for installation server...
```

根据自己的需要配置预设置参数：

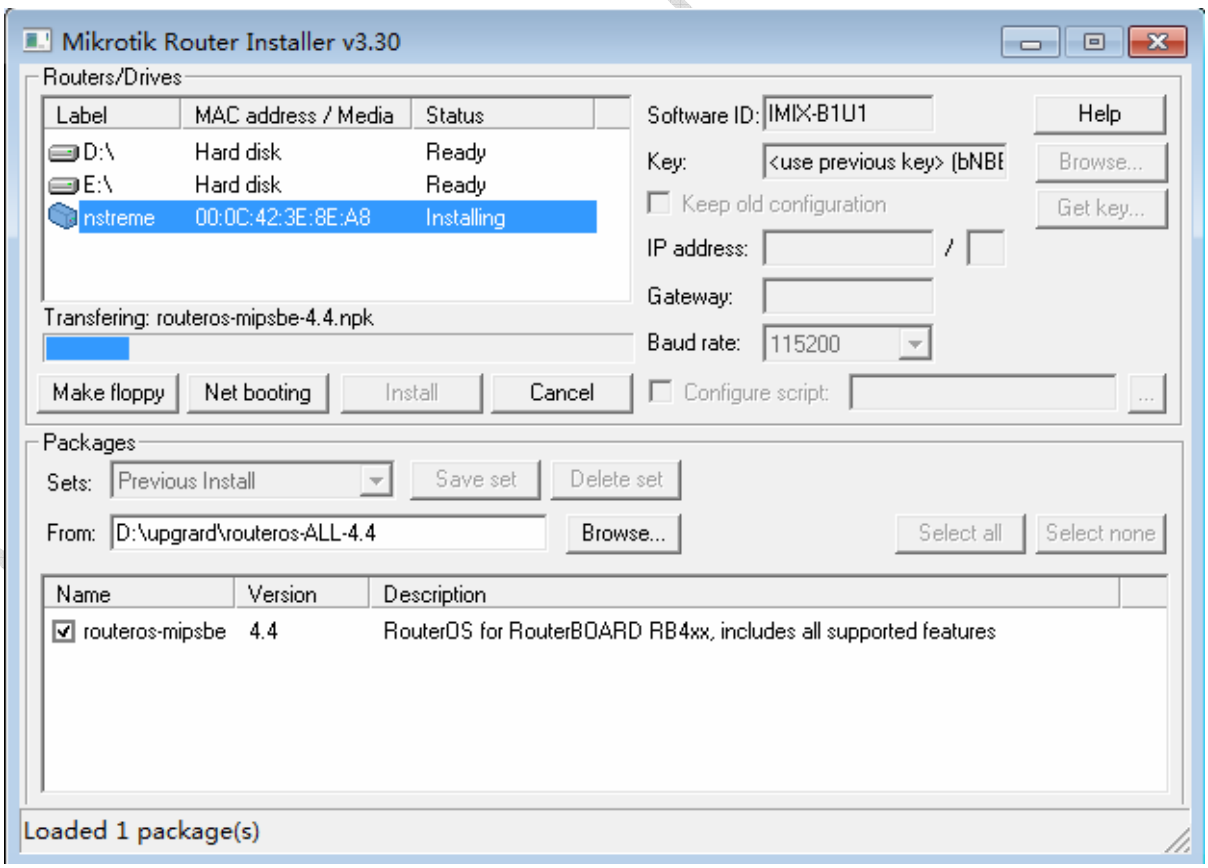


1、Keep old configuration　保留原来的配置不变
2、配置 ip 地址和网关，并设置传输速率采用 115200
3、配置 RouterBOARD 的脚本
4、开始安装

安装过程在超级终端显示的安装进度

```
Welcome to MikroTik Router Software remote installation
Press Ctrl-Alt-Delete to abort

mac-address: 00:0C:42:3E:8E:A8
mac-address: 00:0C:42:3E:8E:A9
mac-address: 00:0C:42:3E:8E:AA
mac-address: 00:0C:42:3E:8E:AB
mac-address: 00:0C:42:3E:8E:AC

software-id: IMIX-B1U1 key:
bNBBSe/onQwGhhk/RW1XBfWTVeOnnja/UsnbuTgcDVckt7fl5zf0Iobz03GWXjCr6vUQ34XSfB9pdGmX
czOmEA==

Waiting for installation server...
Found server at 00:1E:EC:B0:B2:17

Formatting disk......

installing routeros-mipsbe-4.4 [#############                          ]
```

Netinstall 显示的安装情况



6. 当安装工作完成，在安装程序中按“Reboot“键或在超级终端里敲击“回车”，路由器将重启。

这里需要注意：记住设置完后回到 RouterBoard BIOS 中设置为 boot from NAND only（仅从 RouterBoard 的闪存引导）。 这样完成后，就能正常启动 RouterOS。

```
Select boot device:
 * e - boot over Ethernet
   n - boot from NAND, if fail then Ethernet
   1 - boot Ethernet once, then NAND
   o - boot from NAND only
   b - boot chosen device
your choice: n - boot from NAND, if fail then Ethernet
```

在路由器启动完成后，会发出连续两声短触“嘀嘀”的明鸣音，之后在显示屏上，出现登录的提示，如果在终端显示中，没有提示任何信息，标示安装正常。

1.2 第一次登陆 RouterOS

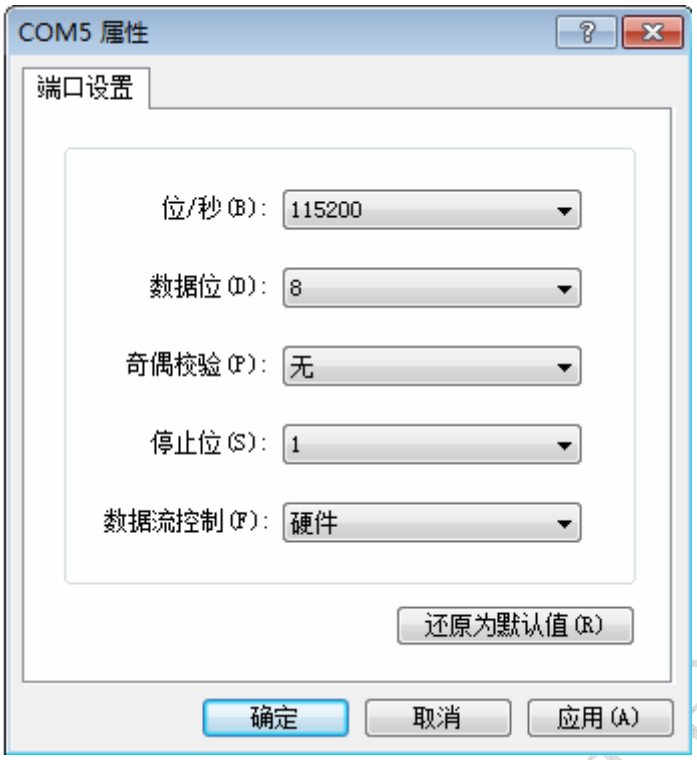
在你安装完成 RouterOS 后，准备第一次登陆 RouterOS，这里将告诉你如何连接到 RouterOS

方式 1，Console 连接

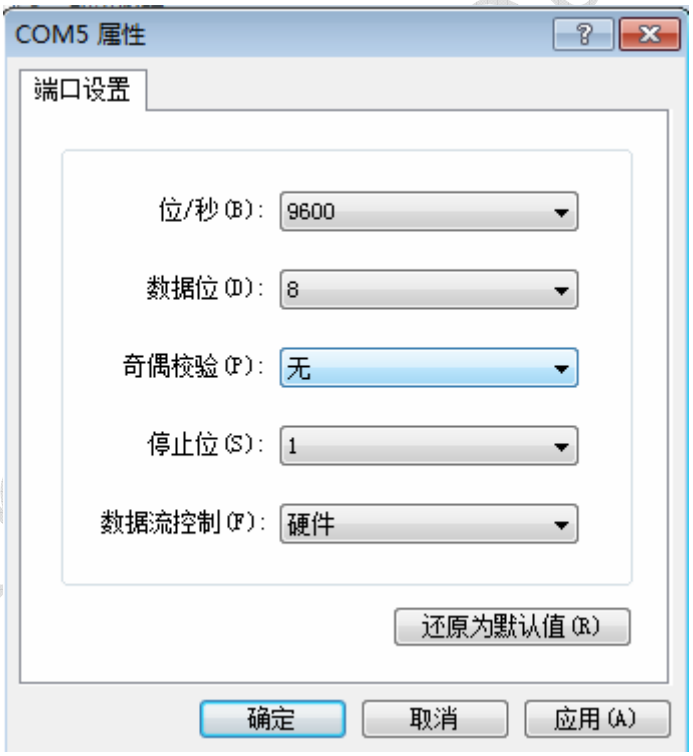
如果你的设备是 RouterBOARD，没有显示器接口连接功能，你必须找到一条 Console 线（普通的 Console 线），或者采用方式 2。

任何 PC 通过标准的 DB9 模式串口线连接到路由器，PC 串口连接的默认设置为每秒位数：9600 bits/s (**RouterBOARD 系列串口是 115200 bits/s**),使用终端仿真程序（如在 windows 中的超级终端或 SecureCRT，UNIX/Linux 的 minicom）连接到路由器。超级终端的具体参数设置如下：

将 Console 线的一端插到路由设备的 Console 口上,另一端插到 PC 上(运行 windows 或者 linux 操作系统)，如果你 PC 没有 Console 口，你可以使用一个 USB-Serial 适配器（USB 转串口适配器），然后运行一个终端程序 HyperTerminal 或者 Putty(windowsXP 以前系统都自带有超级终端,Vista 和 win7 可以将 windowsXP 的超级终端文件拷贝直接使用，文件 hypertrm.dll 和 hypertrm.exe）。RouterBOARD 连接参数如下：



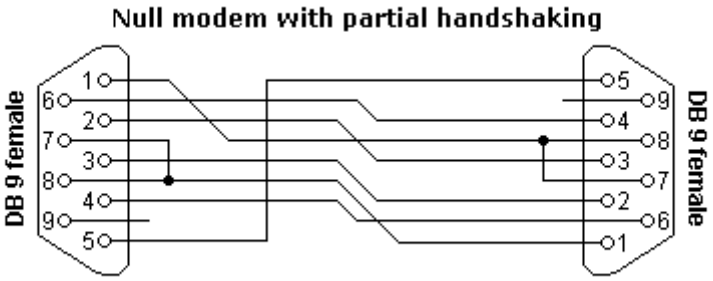
基于 PC 的 RouterOS 连接参数：



通过以上配置你可以连接到 RouterOS，通过配置参数实现你的要去，具体参考相应的配置手册

串口连接线（Null-modem 连接线）

下面是串口连接线的线序：



Connector 1	Connector 2	Function
1	7 + 8	RTS_2 → CTS_2 + CD_1
2	3	Rx ← Tx
3	2	Tx → Rx
4	6	DTR → DSR
5	5	Signal ground
6	4	DSR ← DTR
7 + 8	1	RTS_1 → CTS_1 + CD_2

在路由器启动完成后，会发出连续两声短触“嘀嘀”的明鸣音，之后在显示屏上，出现登录的提示，如果在终端显示中，没有提示任何信息，需要检查一下网线或是串口线是否连接好。

串口控制（管理端）功能允许通过一个 MikroTik Router 串行接口访问路由器的串口终端控台一个特殊的串行接口线通过工作站或者便携式电脑的串口（COM）连接到路由器的串口。在 windows 电脑上常用的串口连接程序是超级终端（HyperTerminal）。

串口控制线配置

基于 PC 的 RouterOS 的 DB9 串口线序排列如下：

Router Side (DB9f)	Signal	Direction	Side (DB9f)
1, 6	CD, DSR	IN	4
2	RxD	IN	3
3	TxD	OUT	2
4	DTR	OUT	1, 6
5	GND	-	5
7	RTS	OUT	8
8	CTS	IN	7

基于 RouterBOARD 系列的串口线序如下：

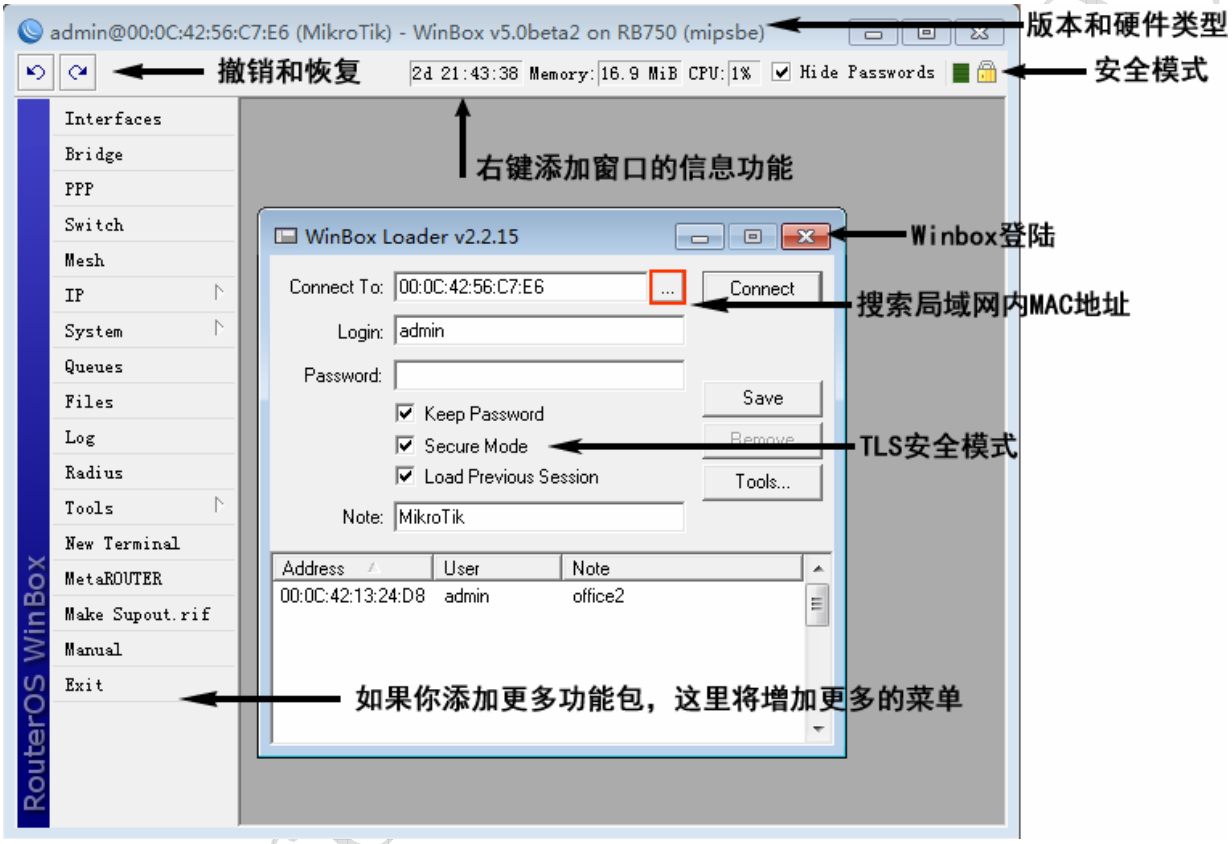
DB9f	功能	DB9f	DB25f
1+4+6	CD+DTR+DSR	1+4+6	6+8+20
2	RxD	3	2
3	xD	2	3
5	GND	5	7
7+8	RTS+CTS	7+8	4+5

注： MikroTik RouterOS 需要定义以上的串口线序，才可以正常通信。

方式 2，Winbox 和 MAC telnet

你可以下载 winbox 应用程序，通过网上搜索可以找到，或者链接 WinBox。下载完成后你要确定你的电脑与路由器已通过以太网线连接，或者他们两个连接在同一局域网内的交换机上。

运行 winbox，点击 … 按钮，winbox 会寻找你的路由器 MAC 地址，如果找到将会显示在 winbox 列表框内，选择连接并登陆，你可以设置一下初始化参数，但最好配置一个 IP 地址到对应接口上，通过 IP 连接到 RouterOS，因为通过 MAC 连接设备不是 100%的可靠



这个方法适用于任何 RouterOS 设备，注意你的 PC 网卡的 MTU 值必须是 1500

方式 3.显示+键盘

如果将 RouterOS 安装到 PC 上，简单的方式就是通过显示+键盘进行配置(注意：RouterBOARD 产品不支持，仅采用方法 1 和方法 2) ，启动后可以在显示屏上看到如下登陆提示：

```
MikroTik v5.0
Login:
```

输入登陆名 **admin** 回车后，密码为空，你可以看到如下信息 as the login name, and hit enter twice (because there is no password yet), you will see this screen:

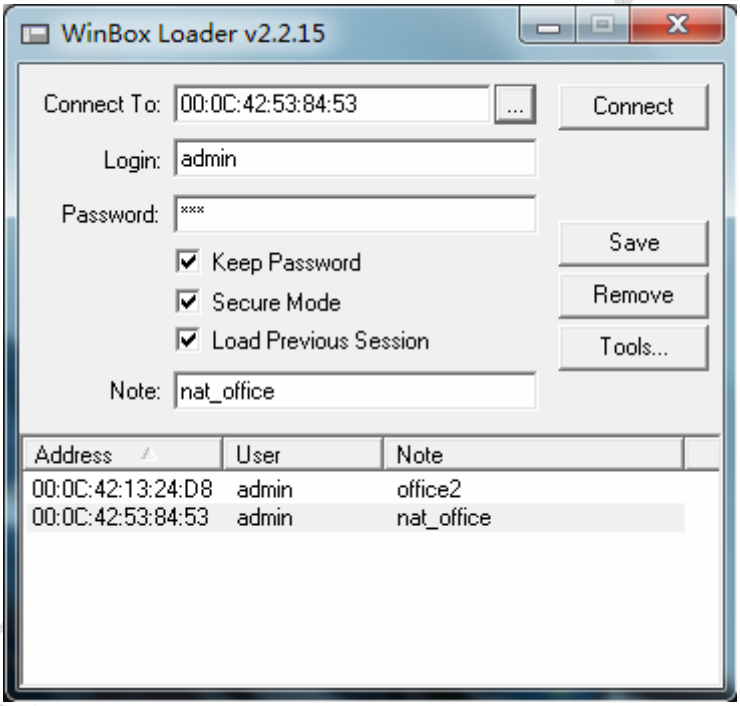
```
  MMM       MMM       KKK                          TTTTTTTTTTT      KKK
  MMMM     MMMM       KKK                          TTTTTTTTTTT      KKK
```

```
  MMM MMMM MMM   III   KKK   KKK   RRRRRR      000000       TTT      III   KKK   KKK
  MMM  MM  MMM   III   KKKKK       RRR  RRR   000  000      TTT      III   KKKKK
  MMM      MMM   III   KKK KKK     RRRRRR     000  000      TTT      III   KKK KKK
  MMM      MMM   III   KKK  KKK   RRR  RRR    000000       TTT      III   KKK  KKK

  MikroTik RouterOS 5.0 (c) 1999-2011       http://www.mikrotik.com/


Terminal ansi detected, using single line input mode
[admin@MikroTik] >
```

现在你可以配置路由器了，你可以通过 setup 命令进行向导配置

Winbox 与 web 登录

MikroTik RouterOS 内能通过远程配置各种参数，包括 **Telnet**, **SSH**, **WinBox** 和 **Webbox**。在这里我们将着重介绍怎样使用 **WinBox**：



MAC-telnet 是在路由器没有 IP 地址的情况下或者配置防火墙参数后无法连接，通过路由器网卡 MAC 地址登录的方式远程连接到路由器。MAC-telnet 仅能使用在来自同一个广播域中（因此在网络中不能有路由的存在），且路由器的网卡应该被启用。注:在 Winbox 中嵌入了通过 MAC 地址连接路由器的功能，并内置了探测工具。这样在管理员忘记或复位了路由器后，同样可以通过 MAC 登陆到 RouterOS 上，进行图形界面操作。

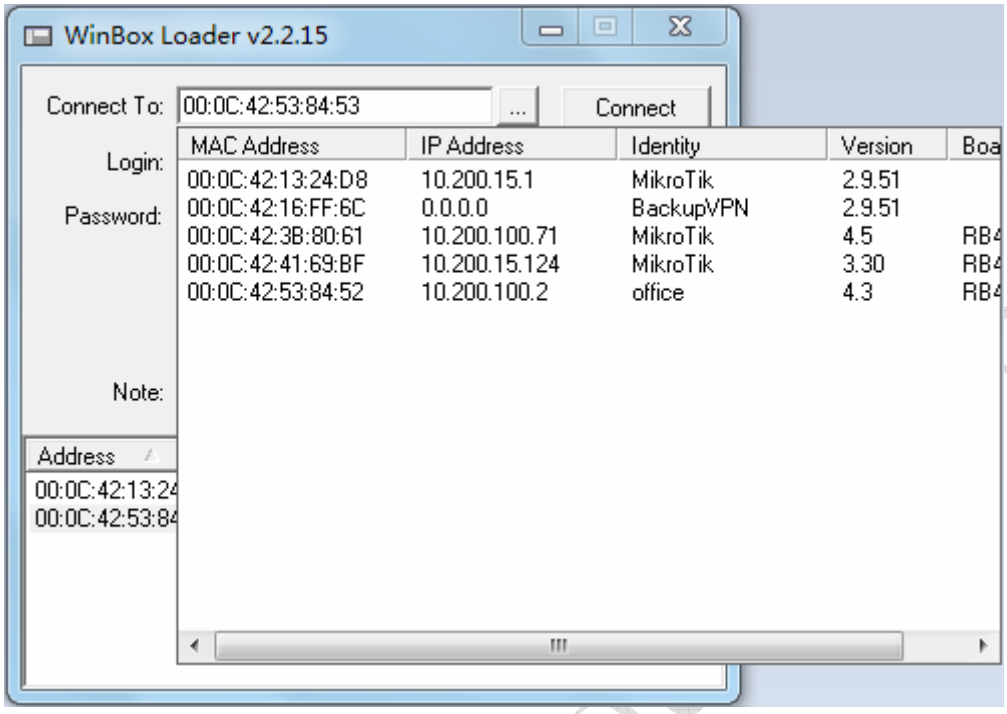
注：在 winbox2.2.12 后增加了可选择的 MAC 登陆或者 IP 登陆的功能

Winbox 控制台是用于 MikroTik RouterOS 的管理和配置，使用图形管理接口（GUI）。通过连接到 MikroTik 路由器的 HTTP（TCP 80 端口）欢迎界面下载 Winbox.exe 可执行文件 ， 下载并保存在你的 Windows 中，之后直接在你 Windows 电脑上运行 Winbox.exe 文件

下面是对相应的功能键做介绍:

- 

搜索和显示 MNDP (MikroTik Neighbor Discovery Protocol) 或 CDP (Cisco Discovery Protocol) 设备。可以通过该功能键搜索同一子网内 MikroTik 和 Cisco 设备。并能通过 MAC 地址登陆到 MikroTik RouterOS 进行操作。



注：在 winbox2.2.12 后的版本增加了 MAC 地址和 IP 地址选择功能，可根据搜索内容选择使用 MAC 地址连接或是 IP 地址连接。

- Connect

通过指定的 IP 地址（默认端口为 80，不许特别指定，如果你修改了端口需要对具体访问端口做自定）或 MAC 地址（如果路由器在同一子网内）登陆路由器。

- Save

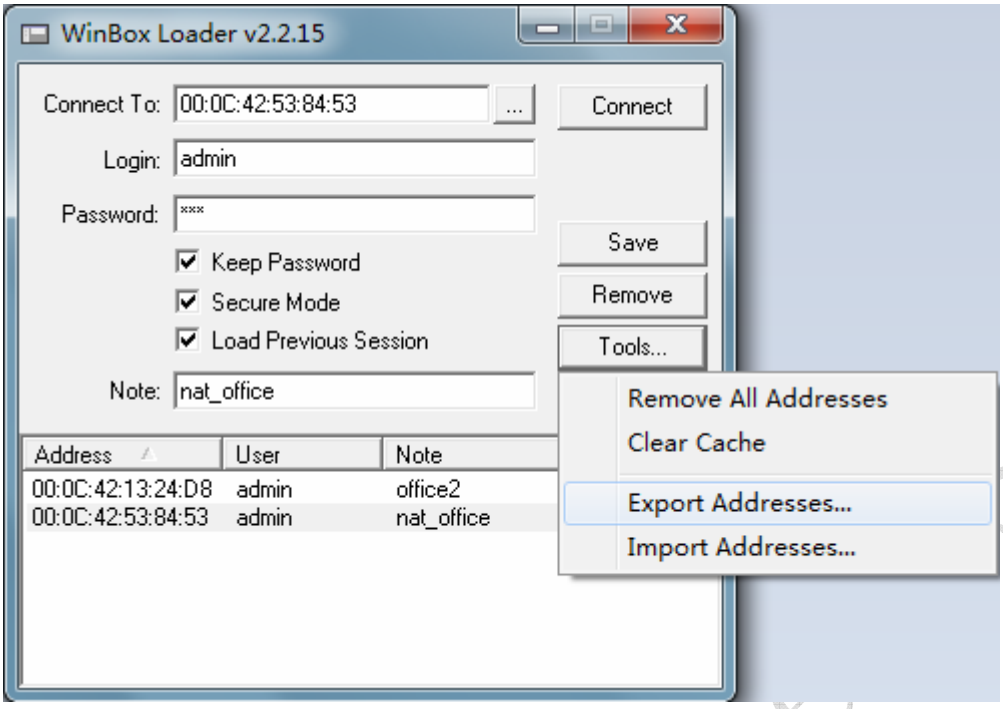
保存当前连接列表（当需要运行它们时，只需双击）

- Remove

删除从列表中选择的项目

- Tools...

删除所有列表中的项目，清除在本地的缓存，从 wbx 文件导入地址或导出为 wbx 文件



- Secure Mode（安全模式）：提供保密并在 winbox 和 RouterOS 之间使用 TLS（Transport Layer Security）协议
- Keep Password（保存密码）：保存密码到本地磁盘的文本文件中

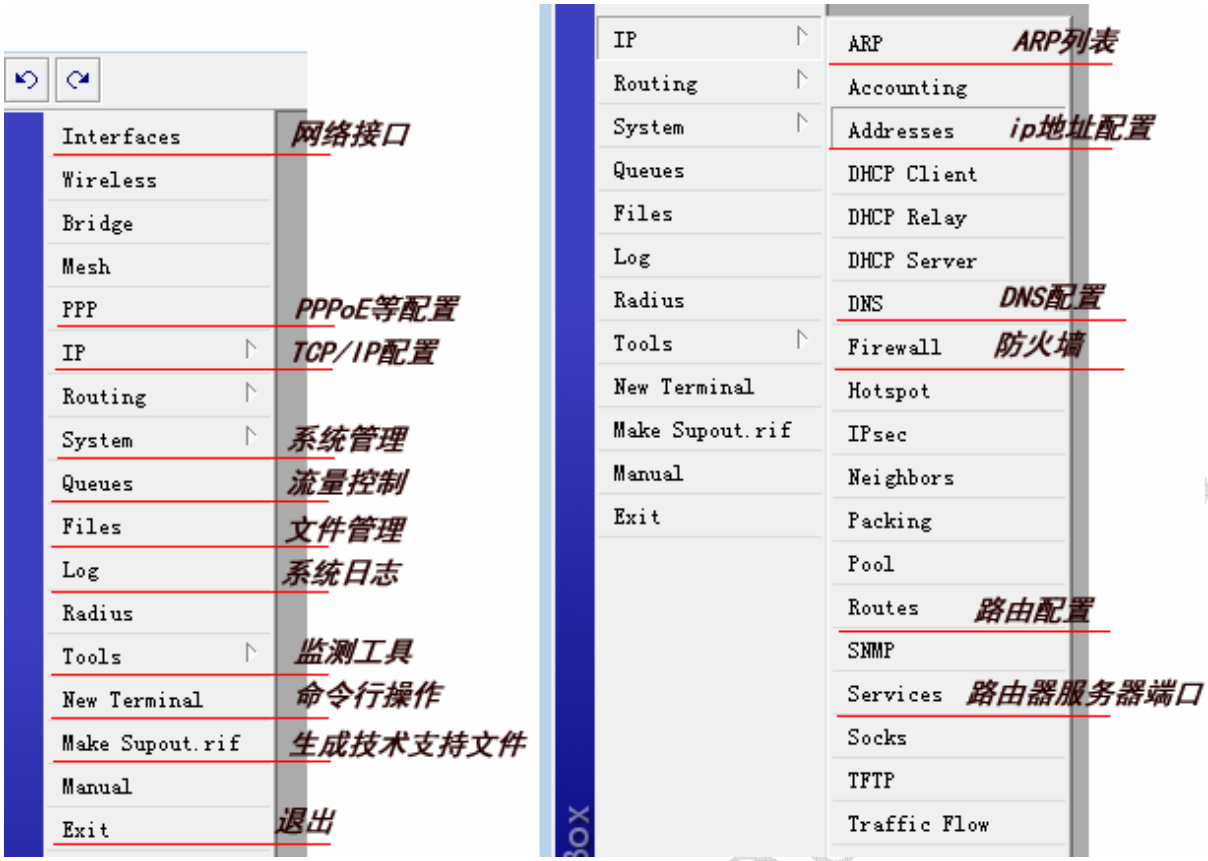
Winbox 控制台使用 TCP 8291 端口，在登陆到路由器后可以通过 Winbox 控制台操作 MikroTik 路由器的配置并执行与本地控制台同样的任务。

命令功能概述

下面是对 Winbox 控制台的操作按钮：

图标	功能		图标	功能
	添加一条项目			定义或编辑一个注释
	删除一条存在项目			查询关键字
	启用一个项目			撤销操作
	禁用一条项目			恢复操作

Winbox 常用管理



故障分析

- **我能在 Linux 上运行 Winbox？**

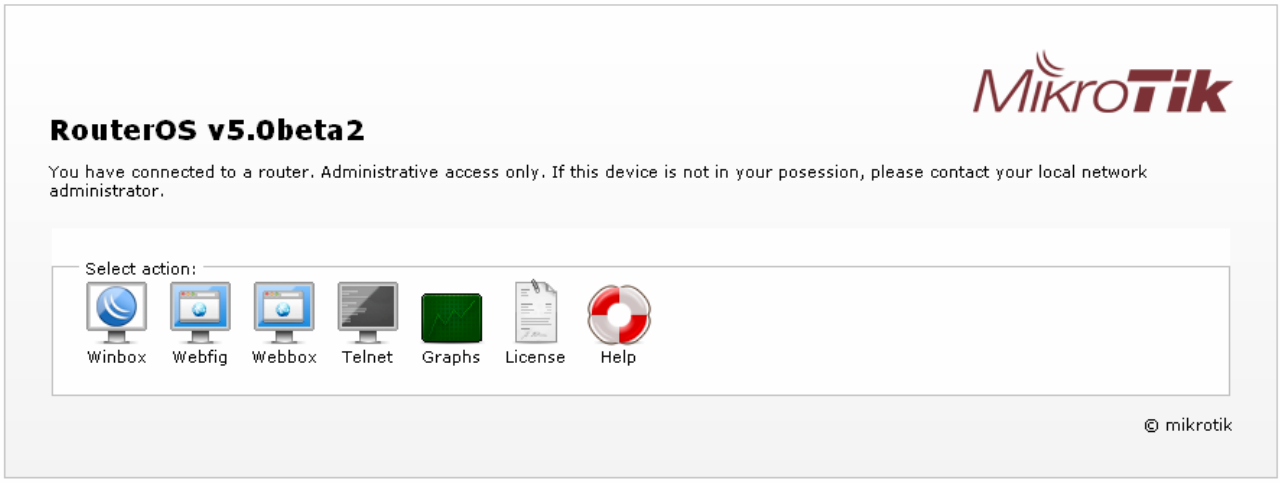
  能，使用 Wine 图形接口，可以运行 Winbox 并连接到 RouterOS。

- **我不能打开 Winbox 控制台**

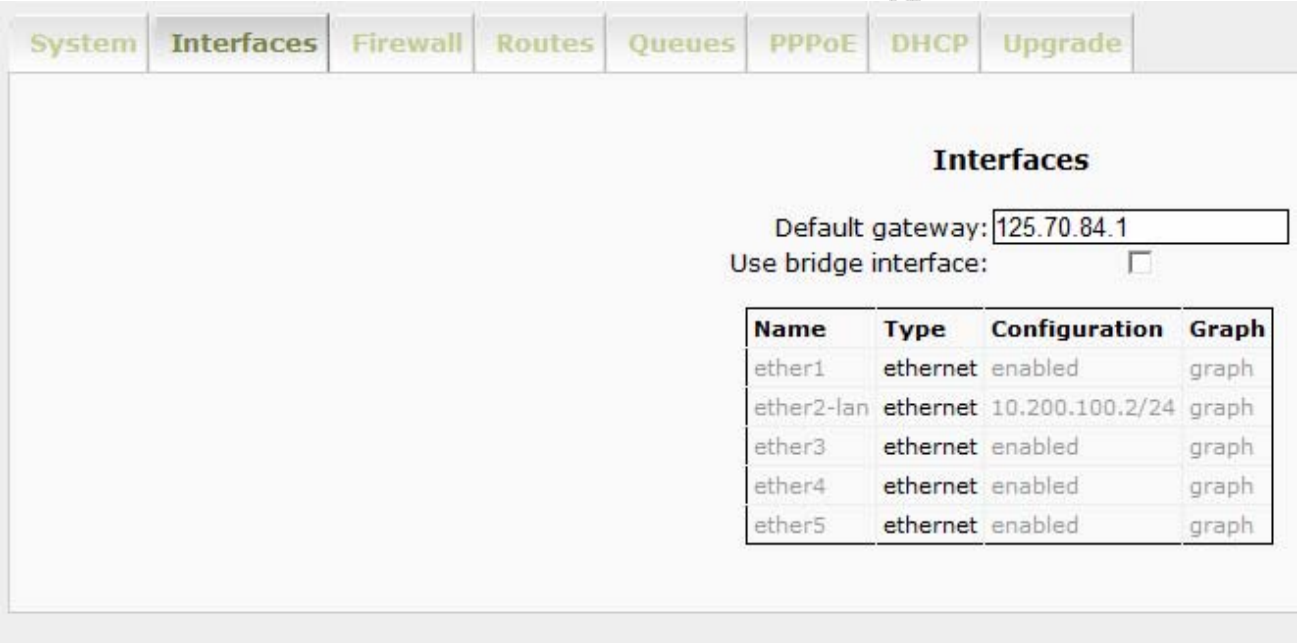
  检查路由器上**/ip service print** 的 **www** 服务端口和地址是否正确，确定地址是你能连接到的指定网络，确定端口为你指定的端口。如果你的服务端口和访问地址被修改，你可以通过下面的命令设置回默认值 **/ip service set www port=80 address=0.0.0.0/0** 。Winbox 控制台使用 TCP8291 端口在防火墙中是否做了访问限制。

Webbox 界面

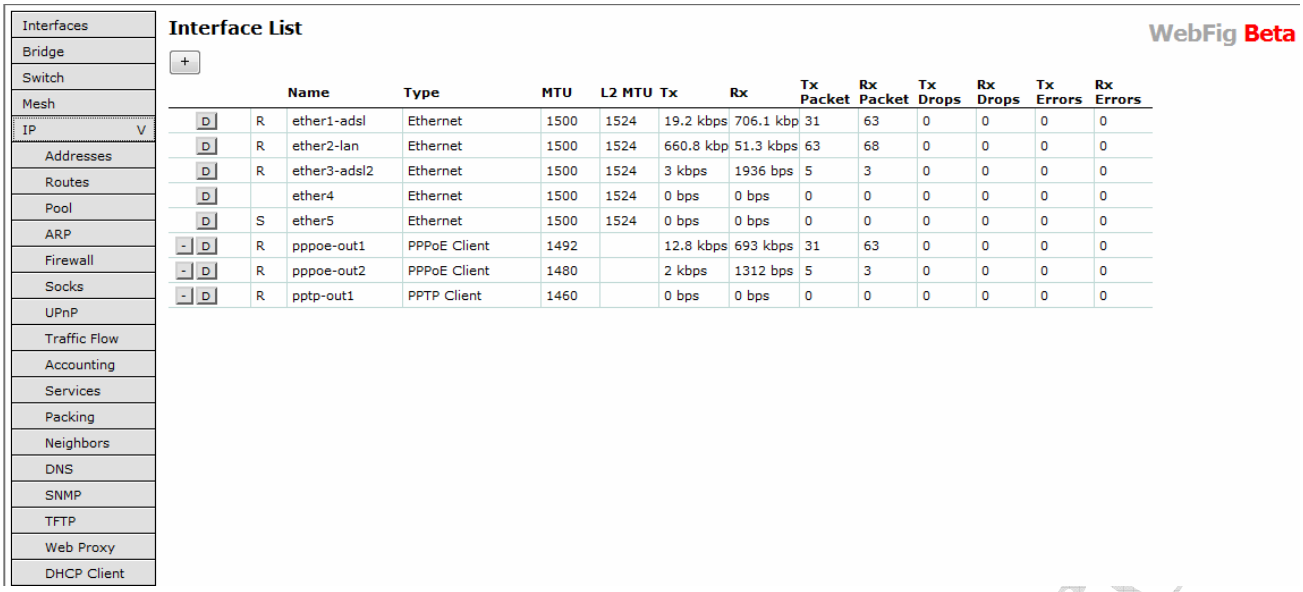
当你配置好 RouterOS 的 IP 地址后，通过在浏览器中输入 http://RouterIP 可以访问到 RouterOS 的 web 页面，我们在右上角输入 RouterOS 的登陆帐号和密码进入 webbox。



下面是 webbox 的页面，在 webbox 中只能对 RouterOS 的一些基本参数配置，不能进行详细的网络配置，主要包括 IP 地址、路由、防火墙（建议不要除 NAT 可以使用外，其他不要启用以免无法登陆），基于 simple 的流量控制、PPPoE 和 DHCP。



Webfig 配置页面，完全是 winbox 的 web 版本，使管理员有更多方式对 RouterOS 进行管理



		Name	Type	MTU	L2 MTU	Tx	Rx	Tx Packet	Rx Packet	Tx Drops	Rx Drops	Tx Errors	Rx Errors
D	R	ether1-adsl	Ethernet	1500	1524	19.2 kbps	706.1 kbp	31	63	0	0	0	0
D	R	ether2-lan	Ethernet	1500	1524	660.8 kbp	51.3 kbps	63	68	0	0	0	0
D	R	ether3-adsl2	Ethernet	1500	1524	3 kbps	1936 bps	5	3	0	0	0	0
D		ether4	Ethernet	1500	1524	0 bps	0 bps	0	0	0	0	0	0
D	S	ether5	Ethernet	1500	1524	0 bps	0 bps	0	0	0	0	0	0
- D	R	pppoe-out1	PPPoE Client	1492		12.8 kbps	693 kbps	31	63	0	0	0	0
- D	R	pppoe-out2	PPPoE Client	1480		2 kbps	1312 bps	5	3	0	0	0	0
- D	R	pptp-out1	PPTP Client	1460		0 bps	0 bps	0	0	0	0	0	0

MAC 层访问 (Telnet 与 Winbox)

通过 MAC 地址进行链接是用来访问没有设置 IP 地址的 RouterOS 路由设备. 这种连接类似于 IP 地址连接. 通过 MAC 地址仅在限于 2 台 MikroTik RouterOS 路由器之间进行.

操作路径: ***/tool mac-server***

属性描述

interface (name | all; 默认: **all**) –连接 MAC 服务器客户端的接口名
all – 所有接口

注: 这是一个在菜单选项的接口列表.如果你添加一些接口进列表, 你就能允许通过 mac 地址连接这些接口. Disabled (**disabled=yes**) 状态的意思是不允许在接口列表中添加的接口通过 mac 地址进行访问. **all** interfaces 默认设置为允许任何接口进行 mac 地址远程访问.

使只有 **ether1** interface 能通过 mac 远程访问服务器:

```
[admin@MikroTik] tool mac-server> print
Flags: X - disabled
 #   INTERFACE
 0   all
[admin@MikroTik] tool mac-server> remove 0
[admin@MikroTik] tool mac-server> add interface=ether1 disabled=no
[admin@MikroTik] tool mac-server> print
Flags: X - disabled
 #   INTERFACE
 0   ether1
[admin@MikroTik] tool mac-server>
```

MAC WinBox Server

操作路径: ***/tool mac-server mac-winbox***

属性描述

interface (name | all; 默认: **all**) – 允许使用 mac 地址的协议连接的接口名
all – 所有接口

*注：*这是一个在菜单选项的接口列表. 如果你添加接口在列表中, 即允许通过 mac 地址访问到这个接口. Disabled (**disabled=yes**) 意思是在这些接口中是不允许使用 mac 地址连接的接口.

仅启用 **ether1** 接口的 MAC 服务器

```
 [admin@MikroTik] tool mac-server mac-winbox> print
Flags: X - disabled
 #   INTERFACE
 0   all
[admin@MikroTik] tool mac-server mac-winbox> remove 0
[admin@MikroTik] tool mac-server mac-winbox> add interface=ether1 disabled=no
[admin@MikroTik] tool mac-server mac-winbox> print
Flags: X - disabled
 #   INTERFACE
 0   ether1
[admin@MikroTik] tool mac-server mac-winbox>
```

动态监控列表

操作路径: ***/tool mac-server sessions***

属性描述

interface (*只读: name*) –连接客户端的接口
src-address (*只读: MAC address*) – 客户 mac 地址（源地址）
uptime (*只读: 时间*) –客户端连接到服务器上的时间

查看 mac 地址连接访问:

```
[admin@MikroTik] tool mac-server sessions> print
 # INTERFACE SRC-ADDRESS      UPTIME
 0 wlan1    00:0B:6B:31:08:22 00:03:01
[admin@MikroTik] tool mac-server sessions>
```

MAC telnet 访问客户端

操作路径: ***/tool mac-telnet***

(MAC address) – 兼容设备的 mac 地址

通过 MAC 地址登陆远程的 RouterOS:

```
[admin@MikroTik] > /tool mac-telnet 00:02:6F:06:59:42
Login: admin
Password:
Trying 00:02:6F:06:59:42...
Connected to 00:02:6F:06:59:42
```

```
  MMM      MMM       KKK                          TTTTTTTTTTT      KKK
  MMMM    MMMM       KKK                          TTTTTTTTTTT      KKK
  MMM MMMM MMM  III  KKK  KKK  RRRRRR     OOOOOO      TTT     III  KKK  KKK
  MMM  MM  MMM   III  KKKKK    RRR  RRR  OOO  OOO     TTT     III  KKKKK
  MMM      MMM   III  KKK KKK  RRRRRR    OOO  OOO     TTT     III  KKK KKK
  MMM      MMM   III  KKK  KKK  RRR  RRR   OOOOOO      TTT     III  KKK  KKK

  MikroTik RouterOS 3.0beta10 (c) 1999-2007      http://www.mikrotik.com/

Terminal linux detected, using multiline input mode
[admin@MikroTik] >
```

1.3 CLI（command Line interface）命令行操作

命令提示显示路由器的身份名称和当前的操作路径，如下：

```
[admin@MikroTik] >
[admin@MikroTik] interface>/ip address
[admin@MikroTik] ip address>
```

命令

在任何操作目录使用‘？’都可用获取在当前目录中的命令信息。

```
[admin@MikroTik] >

log/ -- 系统日志
quit – 退出控制台
radius/ -- Radius 客户端设置
certificate/ -- 授权管理
special-login/ -- 特殊登录用户
redo – 返回以前执行的操作
driver/ -- 驱动管理
ping – ping 命令
setup – 做基本的系统设置
interface/ -- 接口配置
password – 修改密码
undo – 撤销以前的操作
port/ -- 串口控制
import – 运行导入的配置脚本
snmp/ -- SNMP 设置
user/ -- 用户管理
file/ -- 路由器本地文件存储
system/ -- 系统信息和应用程序
queue/ -- 带宽管理
ip/ -- IP 选项
tool/ -- 诊断工具
ppp/ -- 点对点协议
```

```
routing/ -- 各种路由协议设置
export  -- 导出脚本

[admin@MikroTik] >
[admin@MikroTik] ip>

 .. – 回到根目录
service/ -- IP 服务
socks/ -- SOCKS 4 代理
arp/ -- ARP 项目管理
upnp/ -- UPNP 管理
dns/ -- DNS 设置
address/ -- 地址管理
accounting/ -- 传输记录
the-proxy/ --
vrrp/ -- 虚拟路由冗余协议
pool/ -- IP 地址池
packing/ -- 数据包封装设置
neighbor/ -- 邻居
route/ -- 路由管理
firewall/ -- 防火墙管理
dhcp-client/ -- DHCP 客户端设置
dhcp-relay/ -- DHCP 中继设置
dhcp-server/ -- DHCP 服务设置
hotspot/ -- HotSpot 管理
ipsec/ -- IP 安全设置
web-proxy/ -- HTTP 代理
export --

[admin@MikroTik] ip>
```

上面是对可用命令和目录的简短描述，在下面的例子中，你可用通过输入目录名称移动到不同的目录中去。

```
[admin@MikroTik] >                | 根目录
[admin@MikroTik] > driver         | 输入'driver'进入到驱动管理目录中
[admin@MikroTik] driver> /        | 输入'/'从任何目录中回到根目录
[admin@MikroTik] > interface      | 输入'interface'进入接口管理目录中
[admin@MikroTik] interface> /ip   | 输入'/ip'从任何目录进入 IP 管理目录
[admin@MikroTik] ip>              |
```

一个指令或一个变量参数不需要完整的输入，如果是含糊不清的指令或变量参数需要完整的输入。如输入 interface 时，你只要输入 **in** 或 **int**，需要显示完整的指令可以使用**[Tab]**键

通过指令的组合，可以在当前的目录执行在不同目录操作，如：

```
[admin@MikroTik] ip route> print                打印路由表
[admin@MikroTik] ip route> .. address print     打印 IP 地址列表
[admin@MikroTik] ip route> /ip address print    打印 IP 地址列表
```

指令执行概述

Command	指令
command [Enter]	执行指令
[?]	显示该目录中的所有指令列表
command [?]	显示指令的帮助和变量列表
command argument [?]	显示指令的变量帮助
[Tab]	使指令/字段完整，如果输入内容含糊不清，第二次键入 [Tab]会给出存在的选项
/	移动到根目录
/command	执行根目录中的指令
..	移动到上一级目录
""	指定一个空字符串

在配置 IP 地址中，配置'address'和'netmask'参数时，在许多事例中你可以将 IP 地址和子网掩码一起定义，也可以将子网掩码单独定义，这两种方式是相同的，例如下面的两个输入是等价的：

```
/ip address add address 10.0.0.1/24 interface ether1
/ip address add address 10.0.0.1 netmask 255.255.255.0 interface ether1
```

基本操作命令

接口管理（Interface Management）

在配置 IP 地址和路由前，如果你有即插即用卡安装到路由器中，请检查/interface 中的接口列表，多数情况下设备驱动会自动安装，并且相关的接口信息会显示在**/interface print** 列表中，例如：

```
[admin@MikroTik] interface> print
Flags: X - disabled, D - dynamic, R - running
 #    NAME                TYPE           RX-RATE    TX-RATE    MTU
 0  R ether1               ether           0          0         1500
 1  R ether2               ether           0          0         1500
 2 X  wavelan1             wavelan         0          0         1500
 3 X  prism1               wlan            0          0         1500
[admin@MikroTik] interface>
```

如果你想使用这些设备，一般都需要启用，使用**/interface enable name** 指令给出接口名称或标号启用，例如：

```
[admin@MikroTik] interface> print
Flags: X - disabled, D - dynamic, R - running
 #    NAME                TYPE           RX-RATE    TX-RATE    MTU
 0 X  ether1              ether           0          0         1500
 1 X  ether2              ether           0          0         1500
[admin@MikroTik] interface> enable 0
[admin@MikroTik] interface> enable ether2
```

```
[admin@MikroTik] interface> print
Flags: X - disabled, D - dynamic, R - running
 #    NAME                    TYPE           RX-RATE    TX-RATE    MTU
 0  R ether1                   ether           0         0         1500
 1  R ether2                   ether           0         0         1500
[admin@MikroTik] interface>
```

接口的名称能通过**/interface set** 指令来改变其描述：

```
[admin@MikroTik] interface> set ether1 name=Local; set ether2 name=Public
[admin@MikroTik] interface> print
Flags: X - disabled, D - dynamic, R - running
 #    NAME                    TYPE           RX-RATE    TX-RATE    MTU
 0  R Local                    ether           0         0         1500
 1  R Public                   ether           0         0         1500
[admin@MikroTik] interface>
```

通过 add 命令添加规则，如添加 IP 地址操作：

```
[admin@Office] /ip address> prin
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS             NETWORK          BROADCAST      INTERFACE
 0   10.200.15.1/24     10.200.15.0       10.200.15.255   lan
 1 D 222.212.60.227/32  222.212.48.1      0.0.0.0         ADSL
[admin@Office] /ip address> add address=192.168.10.1/24 interface=lan
[admin@Office] /ip address> prin
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS          NETWORK             BROADCAST      INTERFACE
 0   10.200.15.1/24    10.200.15.0       10.200.15.255    lan
 1 D 222.212.60.227/32  222.212.48.1     0.0.0.0          ADSL
 2   192.168.10.1/24   192.168.10.0      192.168.10.255   lan
[admin@Office] /ip address>
```

通过 remove 命令删除不需要的规则

```
[admin@Office] /ip firewall filter> prin
Flags: X - disabled, I - invalid, D - dynamic

 0 X chain=forward action=drop layer7-protocol=qq

 1 X chain=forward action=drop dst-address-list=qq

 2 X chain=forward action=log log-prefix=""
[admin@Office] /ip firewall filter> remove 2
[admin@Office] /ip firewall filter> prin
Flags: X - disabled, I - invalid, D - dynamic
 0 X chain=forward action=drop layer7-protocol=qq

 1 X chain=forward action=drop dst-address-list=qq
```

```
[admin@Office] /ip firewall filter>
```

Setup 指令

当初始化路由器时，通过使用**/setup** 指令设置下列配置内容：

- 重新设置路由器配置
- 载入接口驱动
- 配置 IP 地址和网关
- 设置 DHCP 客户端
- 设置 DHCP 服务端
- 设置 pppoe 客户端
- 设置 pptp 客户端

使用 Setup 指令，在路由器上配置 IP 地址，执行**/setup** 指令行：

```
[admin@MikroTik] > setup
  Setup uses Safe Mode. It means that all changes that are made during setup
are reverted in case of error, or if Ctrl-C is used to abort setup. To keep
changes exit setup using the 'x' key.
[Safe Mode taken]
  Choose options by pressing one of the letters in the left column, before
dash. Pressing 'x' will exit current menu, pressing Enter key will select the
entry that is marked by an '*'. You can abort setup at any time by pressing
Ctrl-C.
Entries marked by '+' are already configured.
Entries marked by '-' cannot be used yet.
Entries marked by 'X' cannot be used without installing additional packages.
   r - reset all router configuration
 + l - load interface driver
 * a - configure ip address and gateway
   d - setup dhcp client
   s - setup dhcp server
   p - setup pppoe client
   t - setup pptp client
   x - exit menu
your choice [press Enter to configure ip address and gateway]: a
```

配置 IP 地址和网关，输入 **a** 或[Enter]

```
 * a - add ip address
 - g - setup default gateway
   x - exit menu
your choice [press Enter to add ip address]: a
```

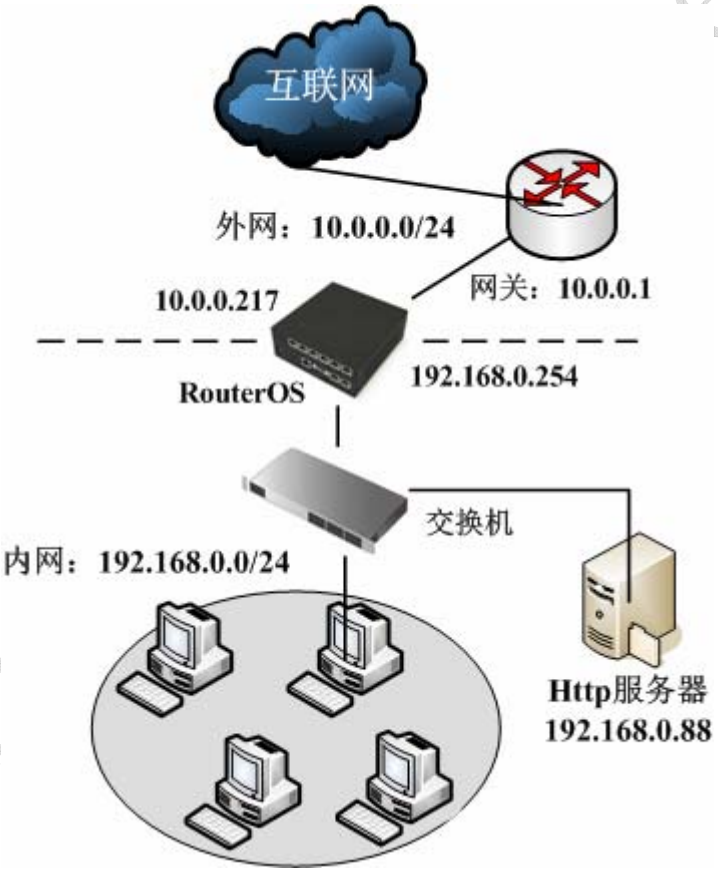
选择 **a** 添加一个 IP 地址，首先，设置程序将要询问你选择那一个接口添加 IP 地址，如果设置程序没有指定出，合适的接口，可以通过键入[Tab]两次，查看可选的接口。 在接口选择后，分配 IP 地址和子网淹码：

```
your choice: a
enable interface:
```

```
ether1  ether2  wlan1
enable interface: ether1
ip address/netmask: 10.1.0.66/24
#Enabling interface
/interface enable ether1
#Adding IP address
/ip address add address=10.1.0.66/24 interface=ether1 comment="added by setup"
 + a - add ip address
 * g - setup default gateway
   x - exit menu
your choice: x
```

1.4 RouterOS 简单网络配置事例

例如： 假如你需要通过 MikroTik router 配置下面的网络：



在当前的事例中我们使用到两个网络（外网和内网）：

- 内网使用地址为：192.168.0.0 子网淹码 24-bit（255.255.255.0）。路由器的地址在这个网络中为 192.168.0.254
- ISP 的网络为 10.0.0.0 子网淹码 24-bit（255.255.255.0）。路由器的地址是在网络中为 10.0.0.217
- 外网 DNS 为 61.139.2.69，202.98.68.96

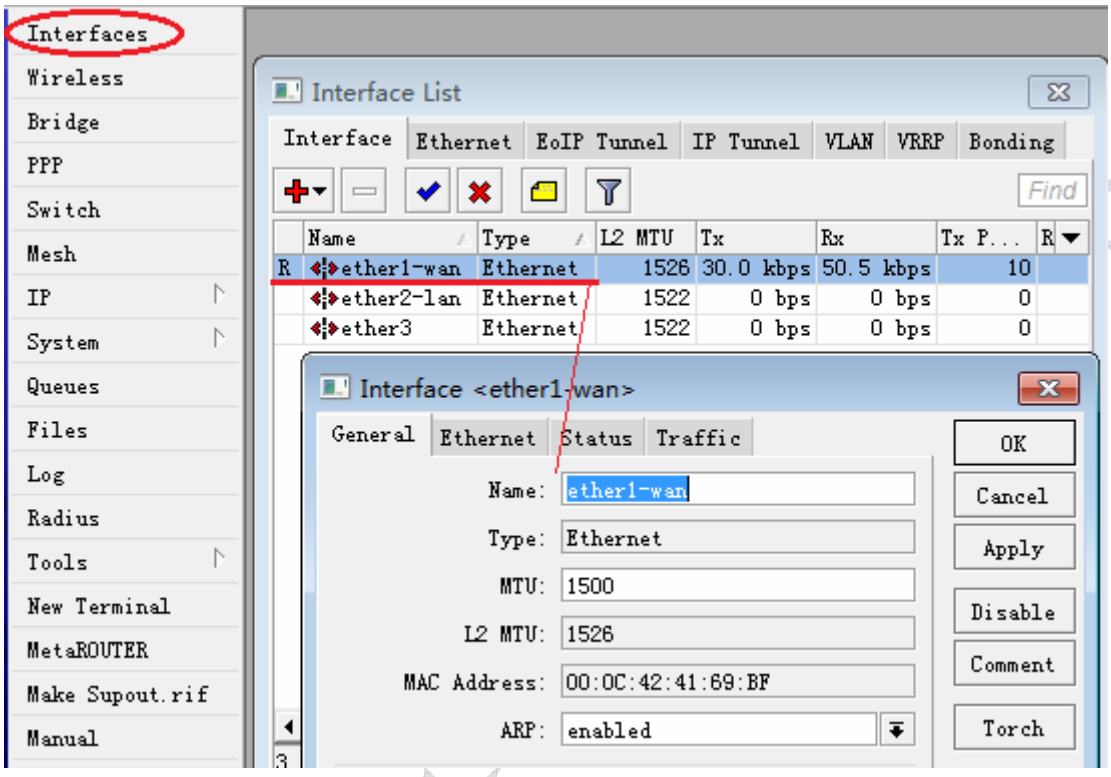
我们的步骤一共分为五步

- 首先：启动设备后，检查 interface 接口网口连接是否正常，并定义网口名称
- 第二：将对应网口的 IP 地址配置好
- 第三：配置网关路由

- 第四：配置 nat 地址转换规则
- 第五：配置 DNS 服务器

第一步：网络接口配置

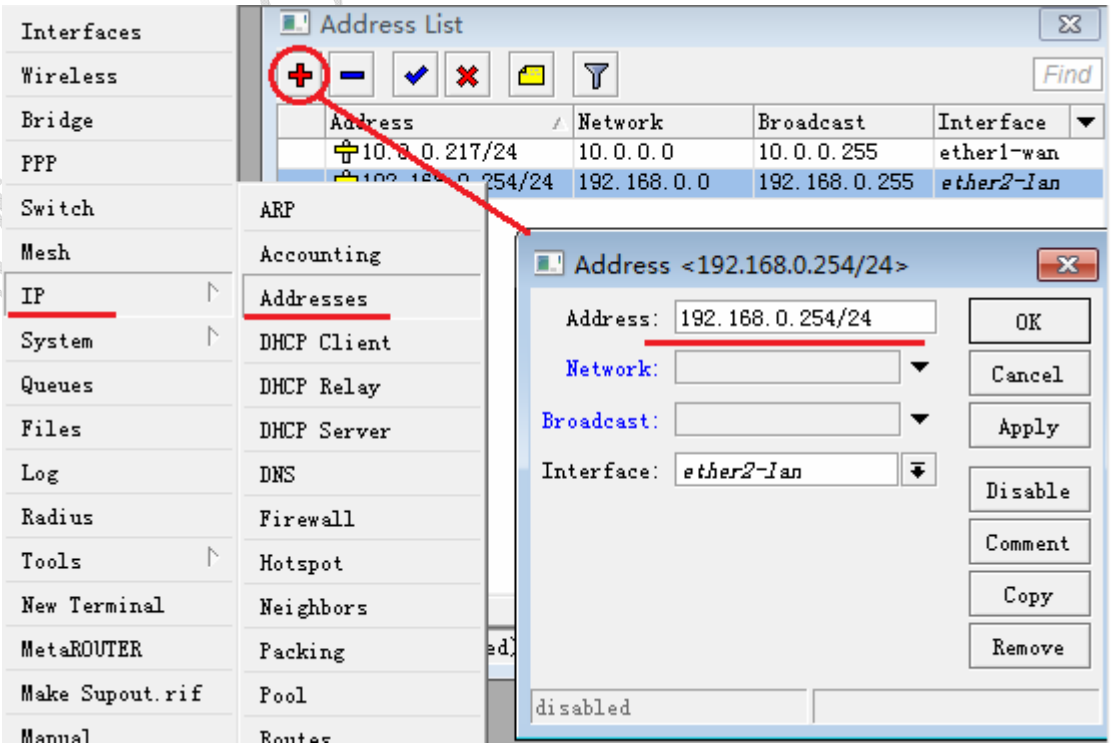
在/interfaces 列表中修改 ether1 为 ether1-wan，定义为外网接口；修改 ether2 为 ether2-lan 定义为内网接口，如图：



同样将 ether2 修改为 ether2-lan，指定内网接口：

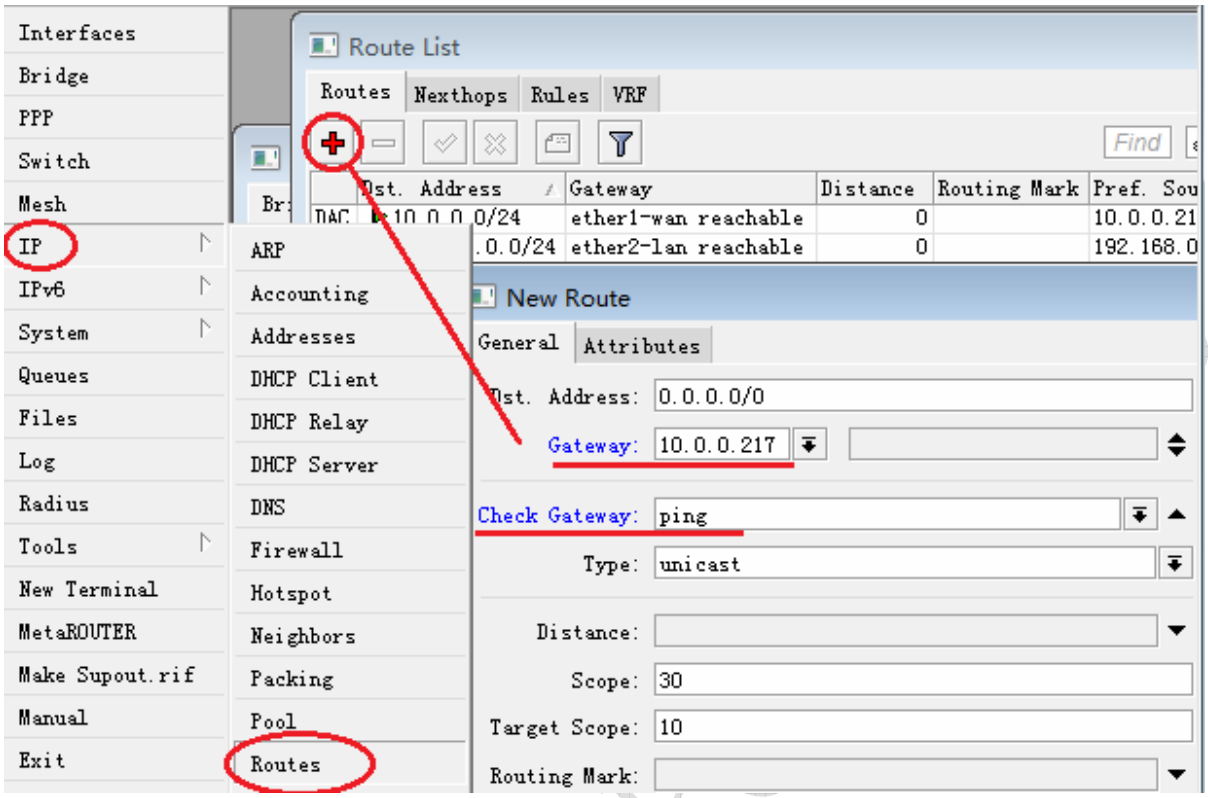
第二步：添加 IP 地址

在/ip address 中添加 IP 地址和选择网卡接口，添加内网和外围的 IP 地址如图：



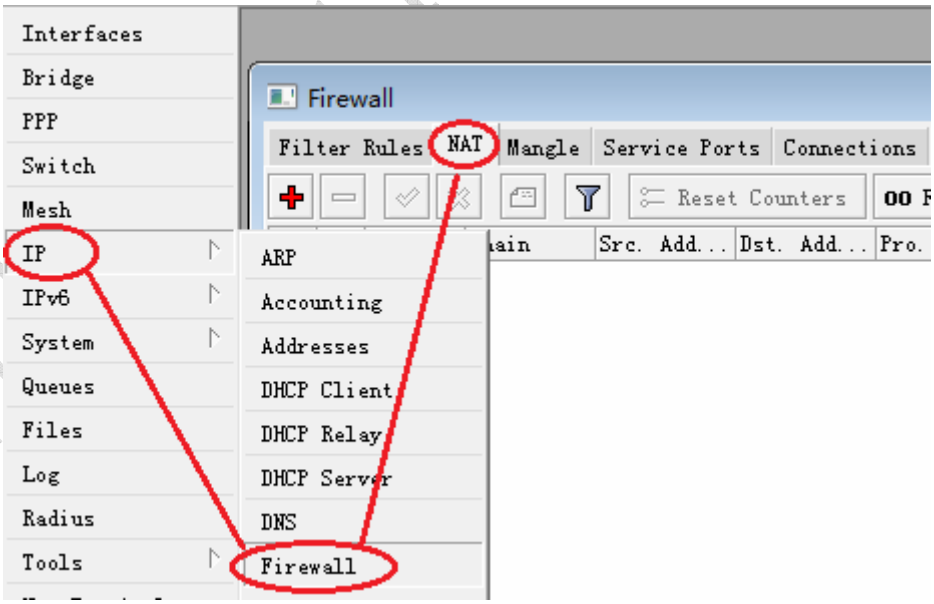
第三步：添加默认网关

在/ip routes 里添加默认网关 10.0.0.1，开启 check-gateway=ping（网关 ping 监测）如图：

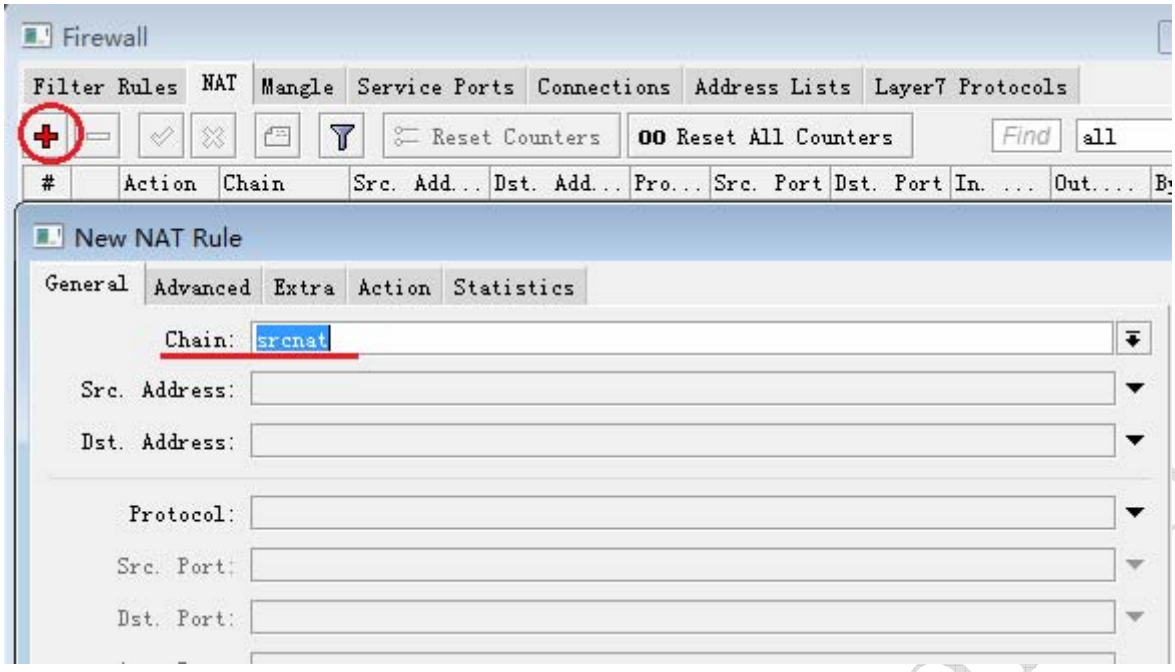


第四步，NAT 地址转换

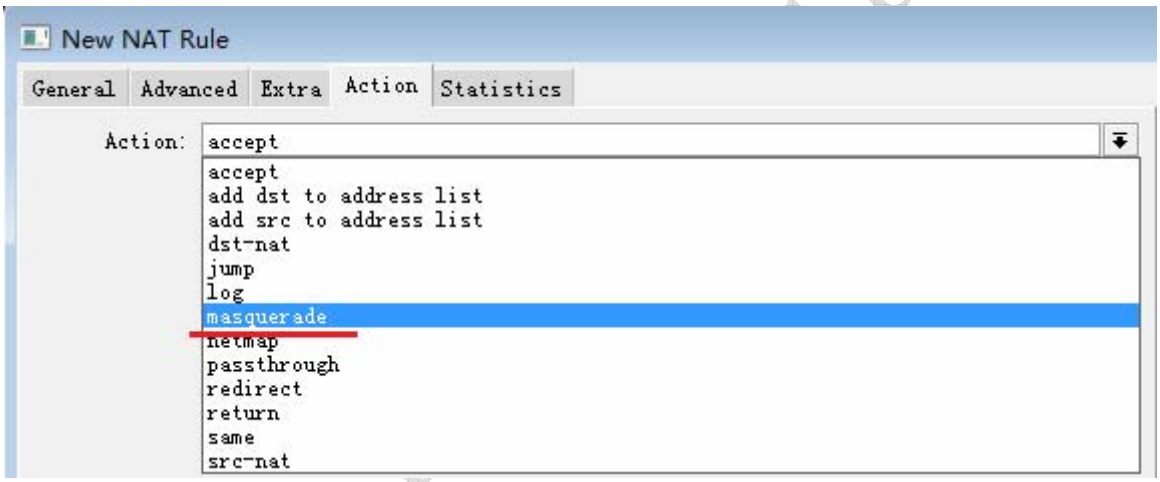
在/ip firewall nat 里点击“+”添加伪装规则：

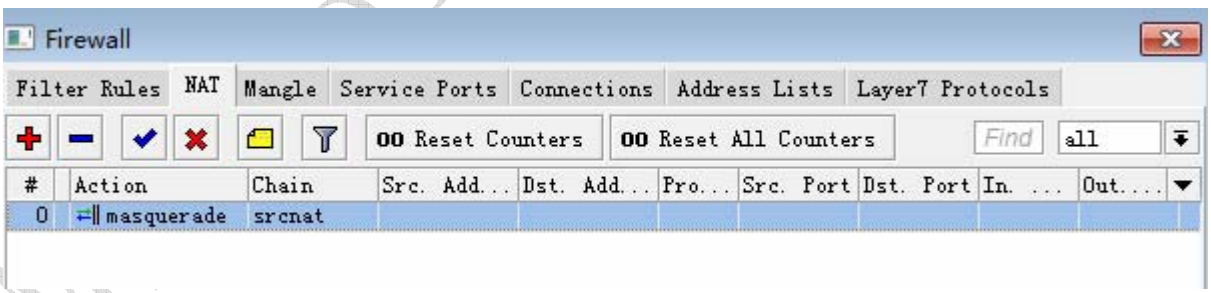


在 NAT 里添加新的规则，在 chain 里选择 srcnat 链表：



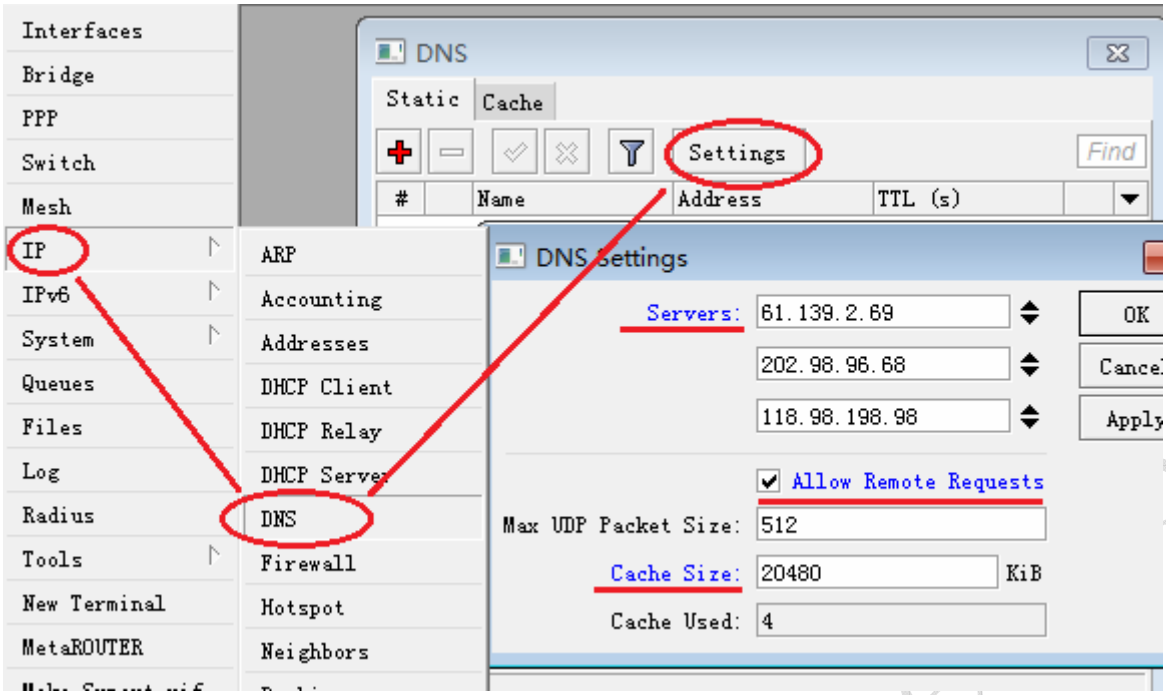
在选择 action 里的 action=masquerade 规则：





第五步，DNS 配置

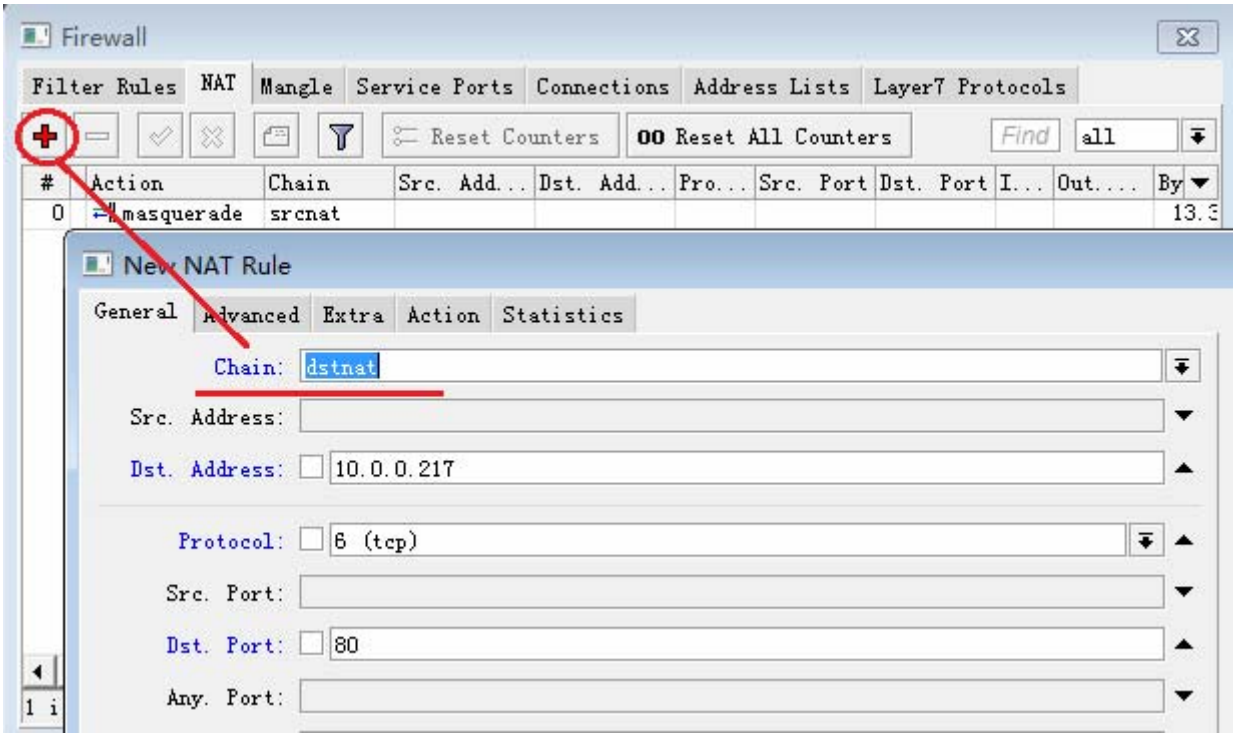
在/ip dns 的 settings 中添加多个 DNS 服务器地址，根据需要启用 DNS 缓存（allow remote requests）：



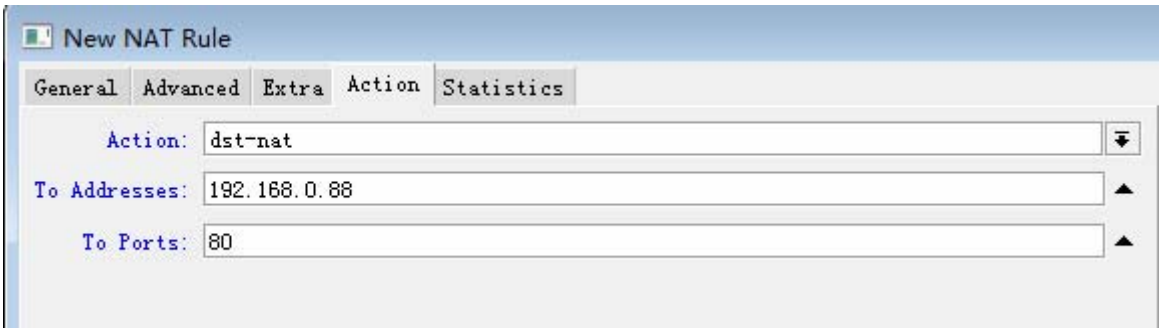
到此，上述的单线上网事例就已经配置完成！

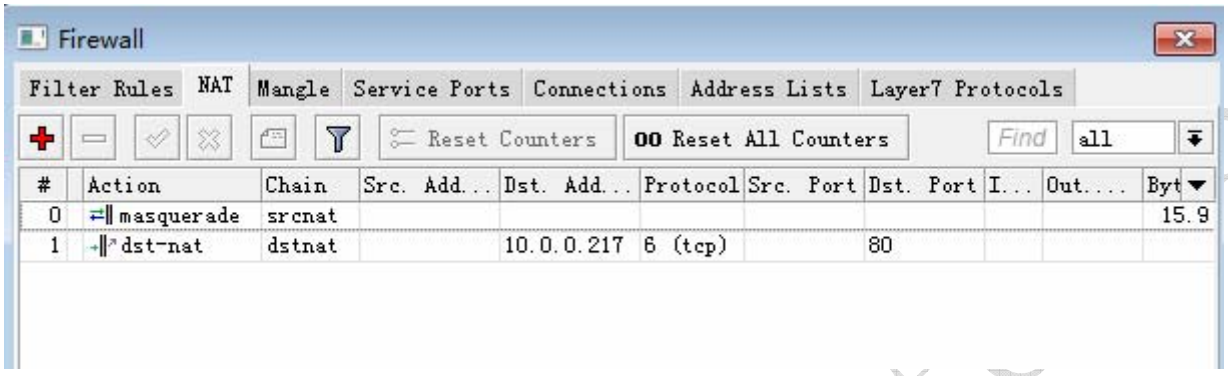
端口映射

这里我们需要将内网的 http 服务器发布到外网，内网的 http 服务器 IP 地址 192.168.0.88，这里需要做端口映射规则，进入 ip firewall nat 里，选择 chain=dstnat，我们的外网 IP 地址是 10.0.0.217 配置到 dst-address，dst-port 为 tcp 协议 80 端口，如下图



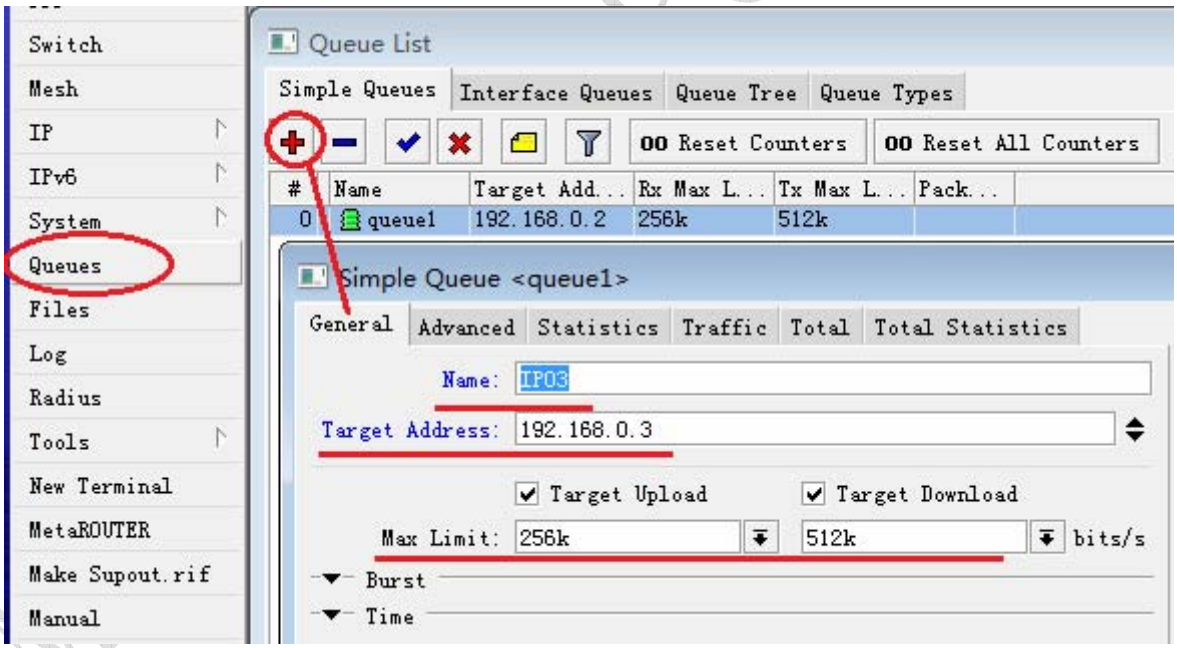
在 action 选择 dst-nat 操作，to-address 设置内网 http 服务器 IP 地址，和端口 80





单机带宽控制

进入 Queue 添加带宽控制规则，选择 simple queue，添加主机 IP 是 192.168.0.3，并取名为 IP03，设置带宽为上行(upload)256kbps，下行(download)512kbps



第二章：系统管理

2.1 RouterOS 备份与复位管理

- 系统备份
- 系统通过备份文件还原
- 导出配置

- 导入配置
- 系统复位

RouterOS 可以通过 backup 将系统备份为二进制文件，采用 FTP 访问或通过在 winbox 中的 file 中下载备份文件，并可以通过备份文件还原路由器设置。

RouterOS 通过导出配置文件，可生成文本文件（可编辑脚本），同样使用 FTP 或通过在 winbox 中的 file 中下载文件，导入配置则将脚本文本文件导入路由器。

系统复位是将所有的配置信息从 RouterOS 中全部删除掉，在做此操作前，最好先将路由器的配置备份一次。

注： 为了保证备份不会失败，请在将备份的文件恢复到同样的软件版本和同样的硬件配置上去

系统备份

操纵路径：***/system backup***

Save 指令是保存当前配置到一个备份文件中，显示文件在**/file** 目录中。如需要回复指定的备份文件，可通过**/system backup** 中的 **load** 指令载入配置，还原系统。

指令描述

load name=[filename] – 载入备份文件的配置
save name=[filename] – 保存当前的配置到文件中

例如：

将当前的配置保存到文件 **test**：

```
[admin@MikroTik] system backup> save name=test
Saving system configuration
Configuration backup saved
[admin@MikroTik] system backup>
```

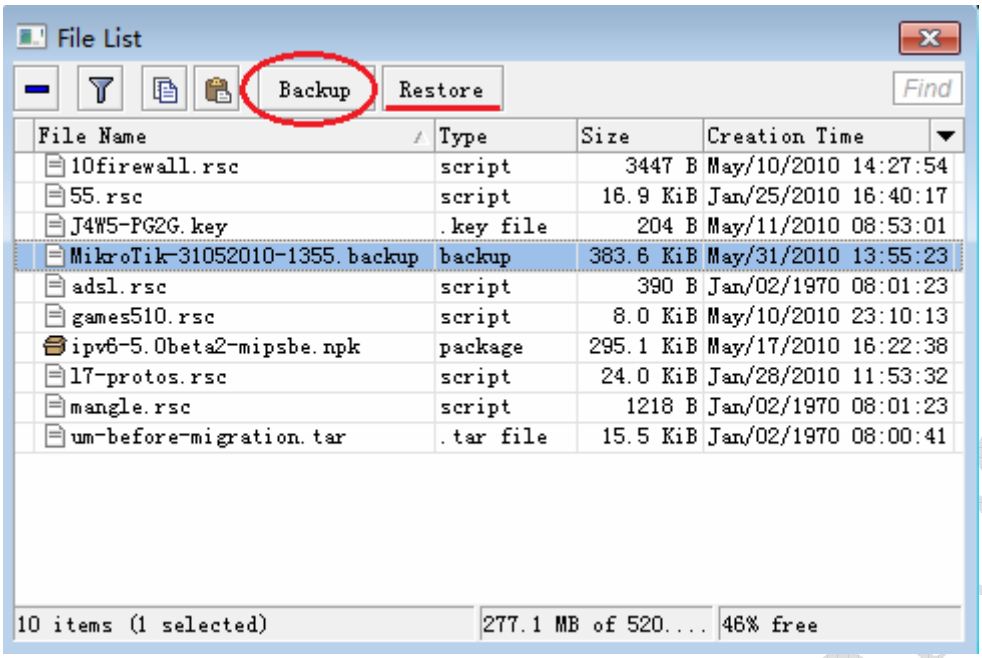
在路由器中查看保存的文件：

```
[admin@MikroTik] > file print
  # NAME                       TYPE        SIZE      CREATION-TIME
  0 test.backup                 backup       12567     aug/12/2002 21:07:50
[admin@MikroTik] >
```

导入备份文件 **test**:

```
[admin@MikroTik] system backup> load name=test
Restore and reboot? [y/N]: y
...
```

Winbox 下配置直接在 files 菜单下，通过 backup 和 restore 操作



导出指令（Export）

指令名称：***export***

Export 指令用于导出脚本配置信息，这个命令可以在任何目录下。**export** 同样也可以通过 **file** 生成脚本配置文件，可用 FTP 或者进入 winbox 下载，并进行编辑。

指令描述

from=[number] – 指定需要导出的项目编号
file=[filename] – 保存的文件名称。

例如：

```
[admin@MikroTik] > ip address print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         BROADCAST       INTERFACE
 0   10.1.0.172/24      10.1.0.0        10.1.0.255      bridge1
 1   10.5.1.1/24        10.5.1.0        10.5.1.255      ether1
[admin@MikroTik] >
```

制作一个导出文件：

```
[admin@MikroTik] ip address> export file=address
[admin@MikroTik] ip address>
```

在路由器中查看导出的文件：

```
[admin@MikroTik] > file print
 # NAME                          TYPE        SIZE       CREATION-TIME
0  address.rsc                    script       315       dec/23/2003 13:21:48
 [admin@MikroTik] >
```

在不创建导出文件名，使用同样的指令导出显示出配置内容：

```
[admin@MikroTik] ip address> export from=0,1
# dec/23/2003 13:25:30 by RouterOS 2.8beta12
# software id = MGJ4-MAN
#
/ ip address
add address=10.1.0.172/24 network=10.1.0.0 broadcast=10.1.0.255 \
   interface=bridge1 comment="" disabled=no
add address=10.5.1.1/24 network=10.5.1.0 broadcast=10.5.1.255 \
   interface=ether1 comment="" disabled=no
[admin@MikroTik] ip address>
```

导入指令

操作路径：***/import***

import 命令只能在根目录下使用**/import file_name** 指令还原指定配置。这种方式适用于部分配置或者功能，例如 ip firewall filter、queue simple 等

指令描述

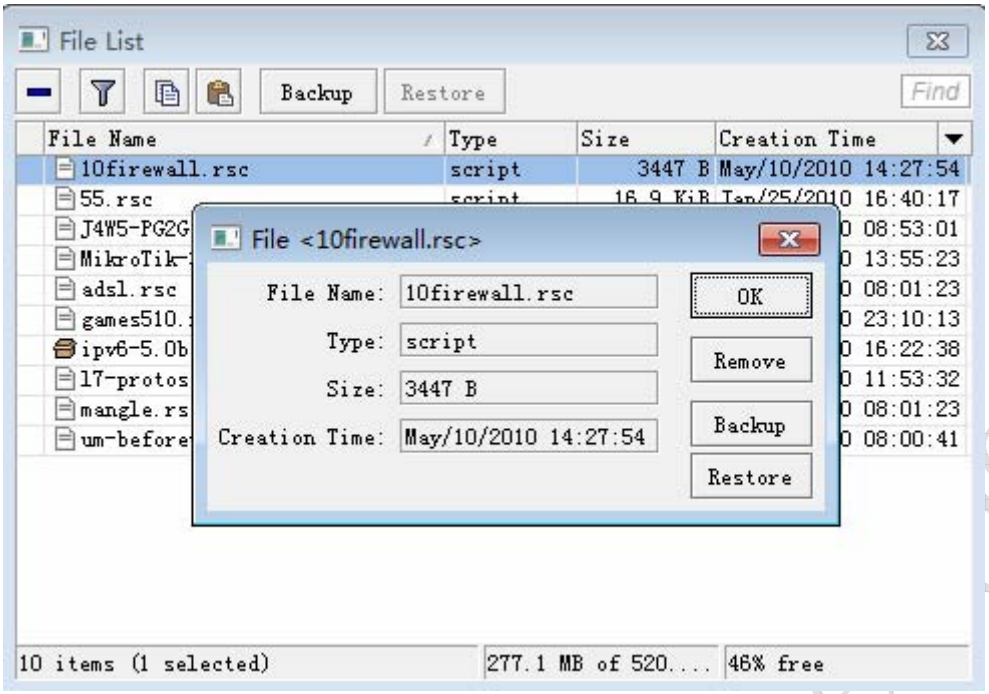
file=[filename] – 导入的路由器配置文件

例如：

使用下面的指令操作导入配置文件：

```
[admin@MikroTik] > import address.rsc
Opening script file address.rsc
Script file loaded successfully
[admin@MikroTik] >
```

Winbox 里可以查看生产的.rsc 的文件：



系统复位

操作路径：***/system> reset-configuration***

这个指令将会清除掉路由器的所有配置，包括登陆的账号和密码（恢复为“admin“和空密码）IP 地址和其他配置将会被抹去，在 **reset** 指令执行后路由器将会重启。RouterOS v3.x 后版本，在复位后默认的 ether1 接口 IP 地址将设置为 192.168.88.1/24

例如：

```
[admin@Office] /system> reset-configuration
Dangerous! Reset anyway? [y/N]: y
```

2.2 系统重启与关机

操作路径：***/system reboot***

注意：当重启时系统会自动搜索是否有新版本的安装包或者功能包，如果发现系统将升级和安装功能包。

重启命令将发送信息给运行中的处理器，并停止和卸载系统文件，重启路由器。

```
[admin@MikroTik] > system reboot
Reboot, yes? [y/N]: y
system will reboot shortly
[admin@MikroTik] >
```

操作路径：***/system shutdown***

在路由器电源关闭前，应停止路由系统的运行，重启命令将发送信息给运行中的处理器，并停止和卸载系统文件，停止路由器。

在一些系统需要大概 10 秒（如果没有升级操作，通常最少需要 5 秒）才能安全关闭电源。

```
[admin@MikroTik] > system shutdown
Shutdown, yes? [y/N]: y
system will shutdown promptly
[admin@MikroTik] >
```

2.3 RouterOS 身份名

操作路径：***/system identity***

通过命令可以查看路由系统身份名，这个身份名也会被初始化到 DHCP 客户端的“主机名（host name）”或者 Wlan 的 SSID 名等，下面是查看路由器身份名：

```
[admin@MikroTik] > system identity print
   name: "MikroTik"
[admin@MikroTik] >
```

设置路由器身份名：

```
[admin@MikroTik] > system identity set name=Gateway
[admin@Gateway] >
```

2.4 系统资源管理

操作路径：***/system resource***

查看系统资源可以了解 RouterOS 的运行情况

*注：*通过 **monitor** 命令显示出 CPU 占用率、内存和硬盘等使用情况。

查看基本的系统资源情况：

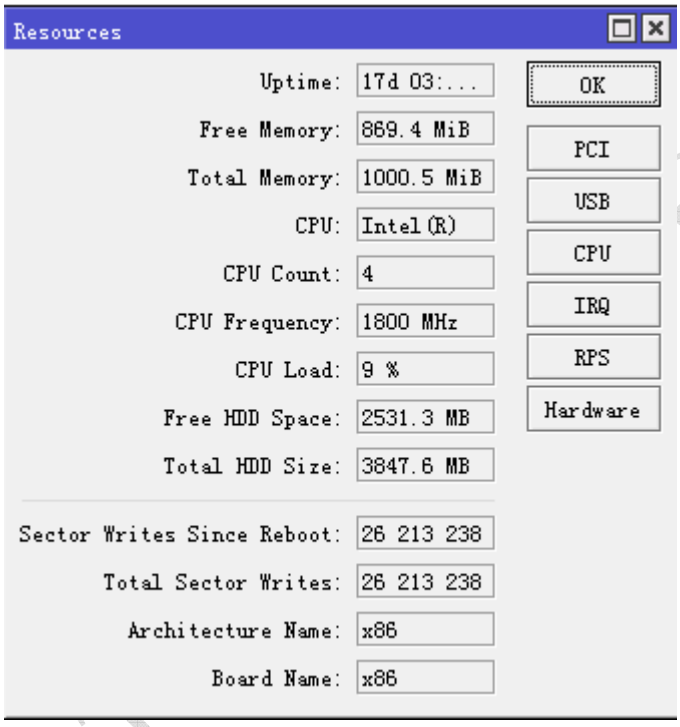
```
[admin@MikroTik] system resource> print
                uptime: 5h26m12s
               version: "3.0"
            free-memory: 17000kB
           total-memory: 30200kB
                 model: "RouterBOARD 500"
                   cpu: "MIPS 4Kc V0.10"
             cpu-count: 1
         cpu-frequency: 333MHz
             cpu-load: 3
        free-hdd-space: 14208kB
       total-hdd-space: 61440kB
 write-sect-since-reboot: 1047
       write-sect-total: 379983
            bad-blocks: 0
```

```
[admin@MikroTik] system resource>
```

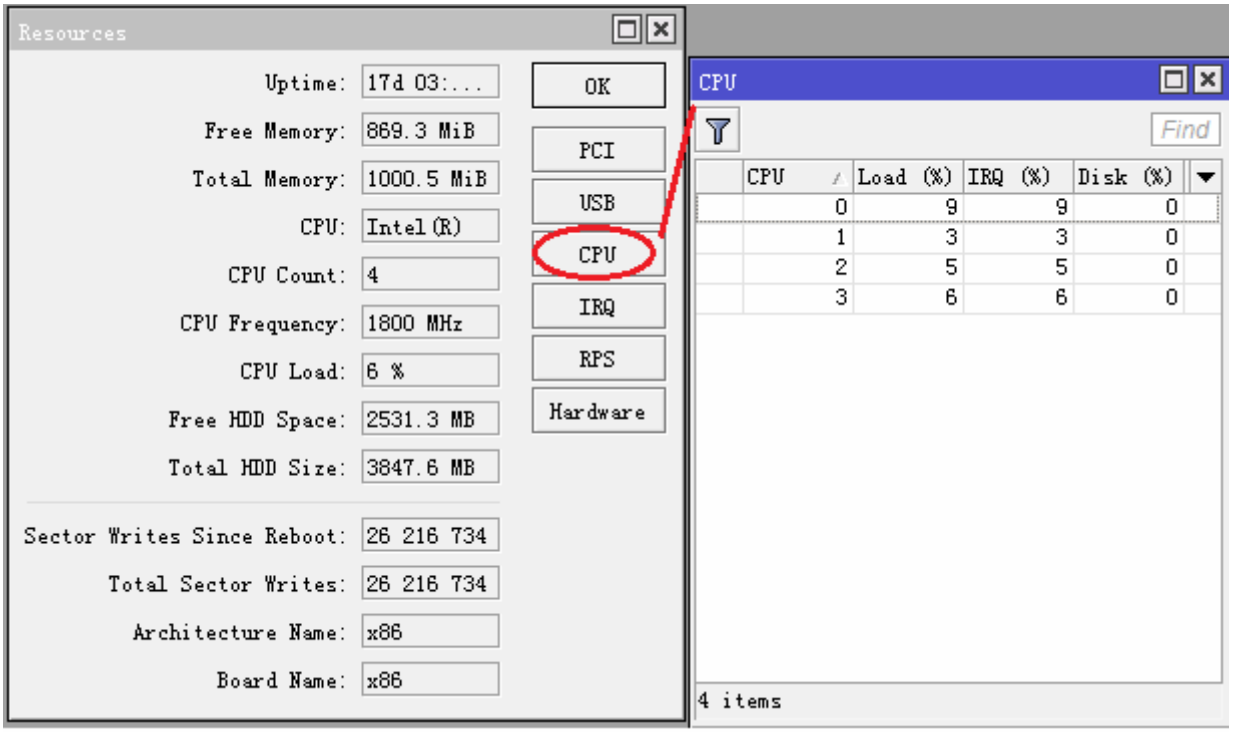
连续查看系统 CPU 和空闲内存使用情况：

```
[admin@MikroTik] > system resource monitor
      cpu-used: 0
   free-memory: 115676

[admin@MikroTik] >
```

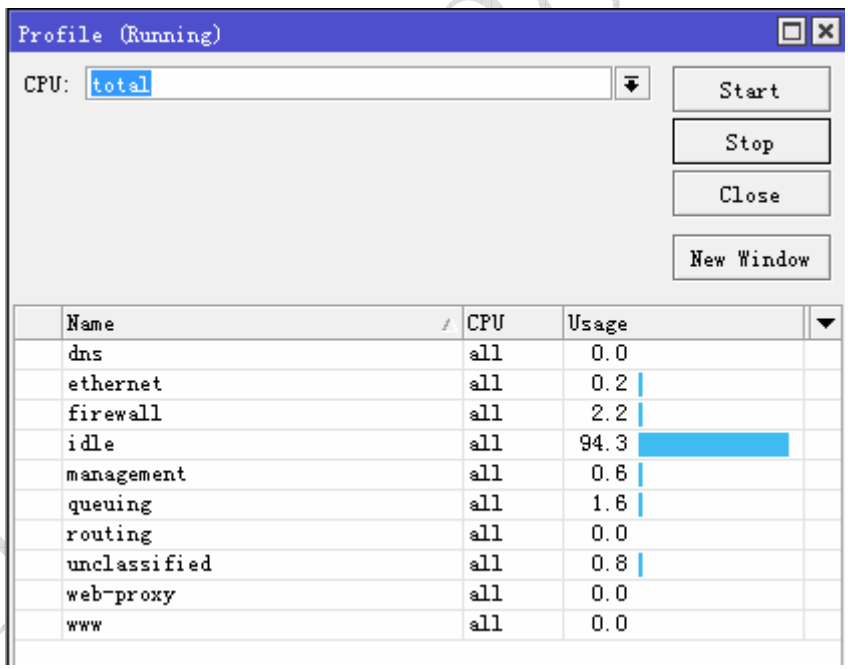
使用 winbox 查看：



RouterOS 5.0 后对多 CPU 进行了优化，可以查看每个 CPU 的占用情况



在 tool 里还增加了每个功能的 CPU 占用情况，进入 tool profile 下可以查看 RouterOS 各个功能 CPU 使用情况，和 windows 资源管理器类似



IRQ 使用监测

命令路径：***/system resource irq print***

显示当前 IRQ 使用情况

```
[admin@MikroTik] > system resource irq print
Flags: U - unused
  IRQ OWNER
  1   keyboard
  2   APIC
```

```
U 3
  4   serial port
  5   [Ricoh Co Ltd RL5c476 II (#2)]
U 6
U 7
U 8
U 9
U 10
  11  ether1
  12  [Ricoh Co Ltd RL5c476 II]
U 13
  14  IDE 1
[admin@MikroTik] >
```

IO 端口监视

操作路径: ***/system resource io print***

显示当前硬件的 IO (Input/Output) 端口使用情况

```
[admin@MikroTik] > system resource io print
PORT-RANGE       OWNER
0x20-0x3F        APIC
0x40-0x5F        timer
0x60-0x6F        keyboard
0x80-0x8F        DMA
0xA0-0xBF        APIC
0xC0-0xDF        DMA
0xF0-0xFF        FPU
0x1F0-0x1F7      IDE 1
0x2F8-0x2FF      serial port
0x3C0-0x3DF      VGA
0x3F6-0x3F6      IDE 1
0x3F8-0x3FF      serial port
0xCF8-0xCFF      [PCI conf1]
0x4000-0x40FF    [PCI CardBus #03]
0x4400-0x44FF    [PCI CardBus #03]
0x4800-0x48FF    [PCI CardBus #04]
0x4C00-0x4CFF    [PCI CardBus #04]
0x5000-0x500F    [Intel Corp. 82801BA/BAM SMBus]
0xC000-0xC0FF    [Realtek Semiconductor Co., Ltd. RTL-8139/8139C/8139C+]
0xC000-0xC0FF    [8139too]
0xC400-0xC407    [Cologne Chip Designs GmbH ISDN network controller [HFC-PCI]
0xC800-0xC87F    [Cyclades Corporation PC300/TE (1 port)]
0xF000-0xF00F    [Intel Corp. 82801BA IDE U100]

[admin@MikroTik] >
```

USB 端口信息

操作路径: ***/system resource usb print***

显示所有路由器可用的 USB 端口。

device (*只读: 文本*) – 设备编号
name (*只读: 文本*) – USB 端口名称
speed (*只读: 整型*) – 该端口工作的带宽速度
vendor (*只读: 文本*) – USB 设备销售商名称

显示所有可用 USB 端口:

```
[admin@MikroTik] system resource usb> print
 # DEVICE VENDOR               NAME                SPEED
 0 1:1                       USB OHCI Root Hub     12 Mbps
[admin@MikroTik] system resource usb>
```

PCI 信息

操作路径: ***/system resource pci print***
category (*只读: 文本*) – 设备类型
device (*只读: 文本*) – 设备编号
device-id (*只读: 整型*) – 十六进制设备 ID
irq (*只读: 整型*) – 该设备使用的 IRQ 编号
memory (*只读: 整型*) – 该设备使用的内存长度
name (*只读: 文本*) – 设备名称
vendor (*只读: 文本*) – 设备销售商名称
vendor-id (*只读: 整型*) – 设备十六进制销售商

查看 PCI 槽相信情况:

```
[admin@MikroTik] system resource pci> print
 # DEVICE   VENDOR                   NAME                     IRQ
 0 00:13.0  Compaq                   ZFMicro Chipset USB (rev... 12
 1 00:12.5  National Semi             SC1100 XBus (rev: 0)
 2 00:12.4  National Semi             SC1100 Video (rev: 1)
 3 00:12.3  National Semi             SCx200 Audio (rev: 0)
 4 00:12.2  National Semi             SCx200 IDE (rev: 1)
 5 00:12.1  National Semi             SC1100 SMI (rev: 0)
 6 00:12.0  National Semi             SC1100 Bridge (rev: 0)
 7 00:0e.0  Atheros Communications     AR5212 (rev: 1)           10
 8 00:0d.1  Texas Instruments          PCI1250 PC card Cardbus ... 11
 9 00:0d.0  Texas Instruments          PCI1250 PC card Cardbus ... 11
10 00:0c.0  National Semi             DP83815 (MacPhyter) Ethe... 10
11 00:0b.0  National Semi             DP83815 (MacPhyter) Ethe... 9
12 00:00.0  Cyrix Corporation          PCI Master (rev: 0)
[admin@MikroTik] system resource pci>
```

2.5 系统监测 Watchdog

Watchdog 监测系统运行情况，当系统软件万一出现错误，将会自动重启。

功能包需求: ***system***
等级需求: *Level1*
操作路径: ***/system watchdog***

当一个 IP 地址没有响应或者系统被锁死，将发出重启指令。软件计时器是用来提供上一次的记录，但是在特殊的情况下(由硬件故障引起的) 它能锁定自己. 对于 RouterBOARD 的硬件监测设备来说它能在任何异常情况下重启。

属性描述

auto-send-supout (yes | no; 默认: **no**) –帮助文件是自动产生它能通过邮件发送
automatic-supout (yes | no; 默认: **yes**) –当软件错误发生时，有一个文件"autosupout.rif" 是自动产生. 这个"autosupout.rif" 文件是对"autosupout.old.rif"的重命名
no-ping-delay (时间; 默认: **5m**) – 在重启以后多久去测试和 ping **watch-address**. 默认设置是如果 **watch-address** 被设置为不可达，这时路由器将在每 6 分钟的时候重启.
send-email-from (文本; 默认: **""**) –邮件地址是发送来自于帮助文件. 如果没有设置，可以通过操作路径 **/tool e-mail** 开启功能
send-email-to (*文本*; 默认: **""**) – 邮件地址是向帮助文件发送
send-smtp-server (*文本*; 默认: **""**) – SMTP 服务地址是向通过帮助文件发送. 如果没有设置可以通过操作路径**/tool e-mail** 开启功能
watch-address (*IP 地址*; 默认: **none**) – 如果设置这功能了的话,万一 6 次按照顺序 ping 指定 ip 地址(没 10 秒发送一次）出现错误，系统会重启
none – 不可用的选项
watchdog-timer (yes | no; 默认: **no**) – 是否重启取决于在一段时间馁系统无法响应

下面是一个系统崩溃的邮件发送配置，万一系统崩溃，自动产生的 supout.rif 帮助文件会自动通过 192.0.2.1 发送到 **support@example.com**:

```
[admin@MikroTik] system watchdog> set auto-send-supout=yes \
\... send-to-email=support@example.com send-smtp-server=192.0.2.1
[admin@MikroTik] system watchdog> print
     watch-address: none
    watchdog-timer: yes
     no-ping-delay: 5m
   automatic-supout: yes
   auto-send-supout: yes
   send-smtp-server: 192.0.2.1
      send-email-to: support@example.com
[admin@MikroTik] system watchdog>
```

RouterOS 多 CPU 设置

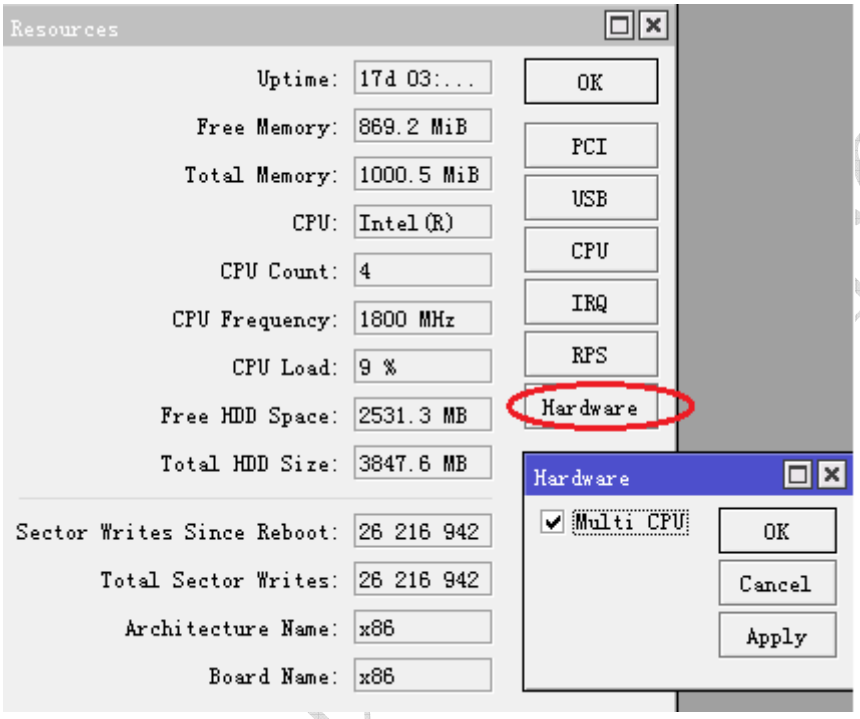
操作路径：***/system hardware***

如果你使用的是多 CPU，可以通过 hardware 功能开关多 CPU 的支持，这个功能仅在 x86 的系统下配置，也可以根据自己的运行情况改变多 CPU 的支持，通过下面的命令启用多 CPU 功能：

```
[admin@MikroTik] > system hardware
```

```
[admin@MikroTik] /system hardware>
..  /  :  edit  export  get  print  set
[admin@MikroTik] /system hardware> set multi-cpu=yes ;
[admin@MikroTik] /system hardware> prin
  multi-cpu: yes
[admin@MikroTik] /system hardware>
```

设置完成后，重启路由即可。



2.6 RouterOS 功能包（Packages）

RouterOS 提供了各种功能包的安装和管理，功能包可以从 MikroTik download 页面下载，提供多种方式下载

RouterOS 安装和管理时每个功能包的组成：

功能包	包含功能
advanced-tools (mipsle, mipsbe, ppc, x86)	包含各种工具 ping、netwatch、ip-scan、sms tool 和 wake-on-LAN
calea (mipsle, mipsbe, ppc, x86)	数据收集功能，指定适用于在美国标准的"Communications Assistance for Law Enforcement Act"
dhcp (mipsle, mipsbe, ppc, x86)	动态主机控制协议客户端和服务器
gps (mipsle, mipsbe, ppc, x86)	支持全球定位系统设备
hotspot (mipsle, mipsbe, ppc, x86)	HotSpot 热点认证系统
ipv6 (mipsle, mipsbe, ppc,	支持 IPv6

x86)	
mpls (mipsle, mipsbe, ppc, x86)	多协议标签交换（Multi Protocol Labels Switching）
multicast (mipsle, mipsbe, ppc, x86)	协议无关组播–稀疏模式; IGMP（Internet Group Managing Protocol）- 代理 Proxy
ntp (mipsle, mipsbe, ppc, x86)	网络对时协议客户端和服务端
ppp (mipsle, mipsbe, ppc, x86)	MlPPP 客户端、PPP、PPTP、L2TP、PPPoE, ISDN PPP 客户端和服务器
routerboard (mipsle, mipsbe, ppc, x86)	访问和管理 RouterBOOT，仅支持 RouterBOARD 硬件
routing (mipsle, mipsbe, ppc, x86)	动态路由协议如 RIP, BGP, OSPF 和路由管理如 BFD 和路由过滤
security (mipsle, mipsbe, ppc, x86)	IPSEC、SSH 和安全的 winbox 连接
system (mipsle, mipsbe, ppc, x86)	路由器基本功能，如静态路由、ip 地址、sNTP、telnet、API、queue、firewall、web-proxy、DNS 缓存、TFTP、IP 地址池、SNMP、sniffer、e-mail 工具、graphing、Bandwidth 测试、torch、EoIP、IPIP、桥接、VLAN、VRRP，在 RouterBOARD 平台也包含 MetaROUTER 虚拟器
ups (mipsle, mipsbe, ppc, x86)	支持 APC ups
user-manager (mipsle, mipsbe, ppc, x86)	MikroTik User Manager 类 Radius 系统
wireless (mipsle, mipsbe, ppc, x86)	Wireless 接口支持，802.11abgn
arlan (x86)	支持传统的 Aironet Arlan
isdn (x86)	支持 ISDN
lcd (x86)	支持 LCD 显示面板
radiolan (x86)	支持 RadioLan 网卡
synchronous (x86)	支持 FarSync
xen (discontinued x86)	XEN 虚拟机（在 4.0 后已经取消）
kvm (x86)	KVM 虚拟机
routeros-mipsle (mipsle)	mipsle 组合包(RB100 系列和 RB500 系列) 包含 system、hotspot, wireless、ppp、security、mpls、advanced-tools、dhcp、routerboard、ipv6 和 routing)
routeros-mipsbe (mipsbe)	mipsbe 组合包(RB400 系列和 700 系列)包含 system、hotspot, wireless、ppp、security、mpls、advanced-tools、dhcp、routerboard、ipv6 和 routing)
routeros-powerpc (ppc)	PowerPC 组合包(RB333、RB600/A、RB800 和 RB1000 系列) 包含 system、hotspot, wireless、ppp、security、mpls、

	advanced-tools、dhcp、 routerboard、ipv6 和 routing)
routeros-x86 (x86)	x86 组合包(Intel/AMD PC, RB230) 包含 system、hotspot, wireless、ppp、security、mpls、advanced-tools、dhcp、routerboard、ipv6 和 routing)

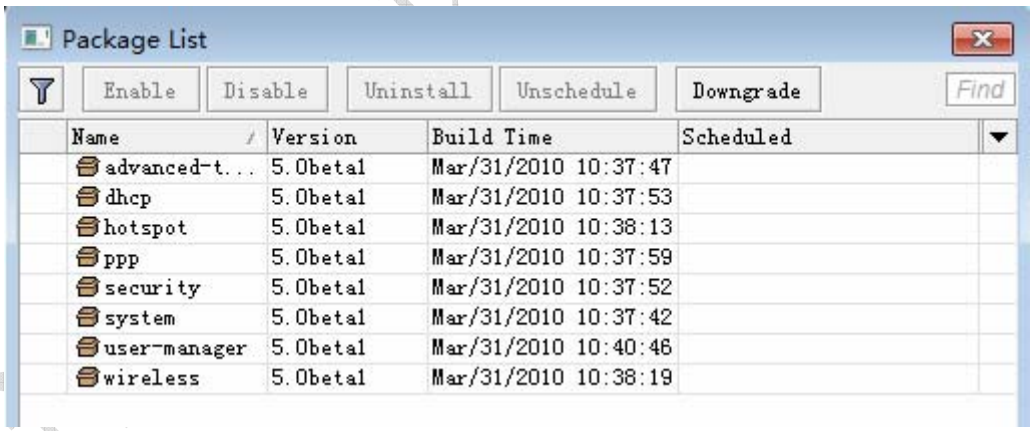
功能包操作

操作路径: **/system package**

在目录下执行命令生效仅在路由器重启后，即选择了对某一功能包进行操作后，必须正常重启路由器，执行的命令才能生效。

命令	属性
disable	计划下一次重启后禁用功能包，该功能提供的所有功能将无法获得
downgrade	将提示重启，在重启过程中将检查是否有低版本的 RouterOS 功能包上传到路由器，并试着降级 RouterOS
print	输出功能包信息，如版本、功能包状态和计划状态
enable	计划下一次重启后启用功能包
uninstall	计划下一次重启后从路由器删除功能包，
unschedule	为功能包取消计划任务

在 winbox 下查看功能包信息，进入 system packetage：



事例

显示出可获得的功能包

```
[admin@MikroTik] > /system package print
Flags: X - disabled
 #   NAME                 VERSION                SCHEDULED
 0 X ipv6                  3.13
 1   system                3.13
 2 X mpls                  3.13
 3 X hotspot               3.13
 4   routing               3.13
```

```
5   wireless              3.13
6 X dhcp                  3.13
7   routerboard           3.13
8   routeros-mipsle       3.13
9   security              3.13
10 X ppp                  3.13
11  advanced-tools        3.13
```

删除功能包，并重启

```
[admin@MikroTik] > /system package uninstall ppp;
[admin@MikroTik] >/system reboot;
Reboot, yes? [y/N]:
```

禁用功能包，并重启

```
[admin@MikroTik] > /system package disable hotspot;
[admin@MikroTik] >/system reboot;
Reboot, yes? [y/N]:
```

降级 RouterOS，确定以将低版本的安装包上传到路由器。

```
[admin@MikroTik] > /system package downgrade;
[admin@MikroTik] >/system reboot;
Reboot, yes? [y/N]:
```

取消删除和禁用的命令

```
[admin@MikroTik] > /system package unschedule ipv6
```

2.7 升级和降级 RouterOS

当你通过 BT 下载完 RouterOS 软件后如“routeros-ALL-3.30”，里面有多个文件，每个文件对应不同的硬件做升级和降级设置，具体区分如下：

BT 包里有四个文件名：

all_packages_mipsbe 对应所有 Atheros 芯片的 RB400 系列产品和 700 系列产品

all_packages_mipsle 对应 RB100 系列和 RB500 系列（RB133、RB133c、RB150、RB192、RB532）MIPS 4Kc 芯片

all_packages_ppc 对应 RB300、RB600、RB800 和 RB1000 系列（RB333、RB600、RB800、RB1000 和 RB1100）PowerPC 芯片

all_packages_x86 对应所有 x86 构架的 PC 设备（AMD、Intel、VIA 和其他 x86 PC）

其他文件：

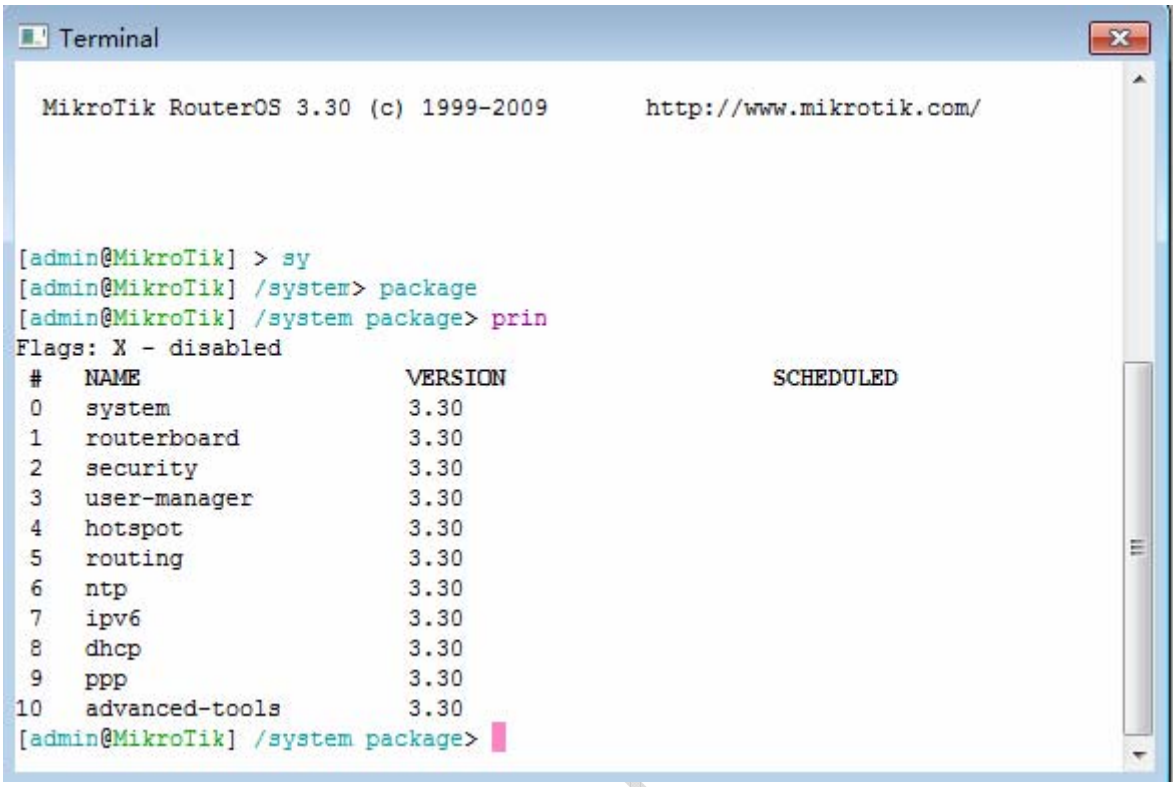
mikrotik-x.x.iso 光盘镜像文件,用于 x86 安装

如果是 2.9 版本的 BT 文件区分如下：

all_packages_ns 对应 RB100 系列和 RB500 系列（RB133、RB133c、RB150、RB192、RB532）MIPS 4Kc 芯片

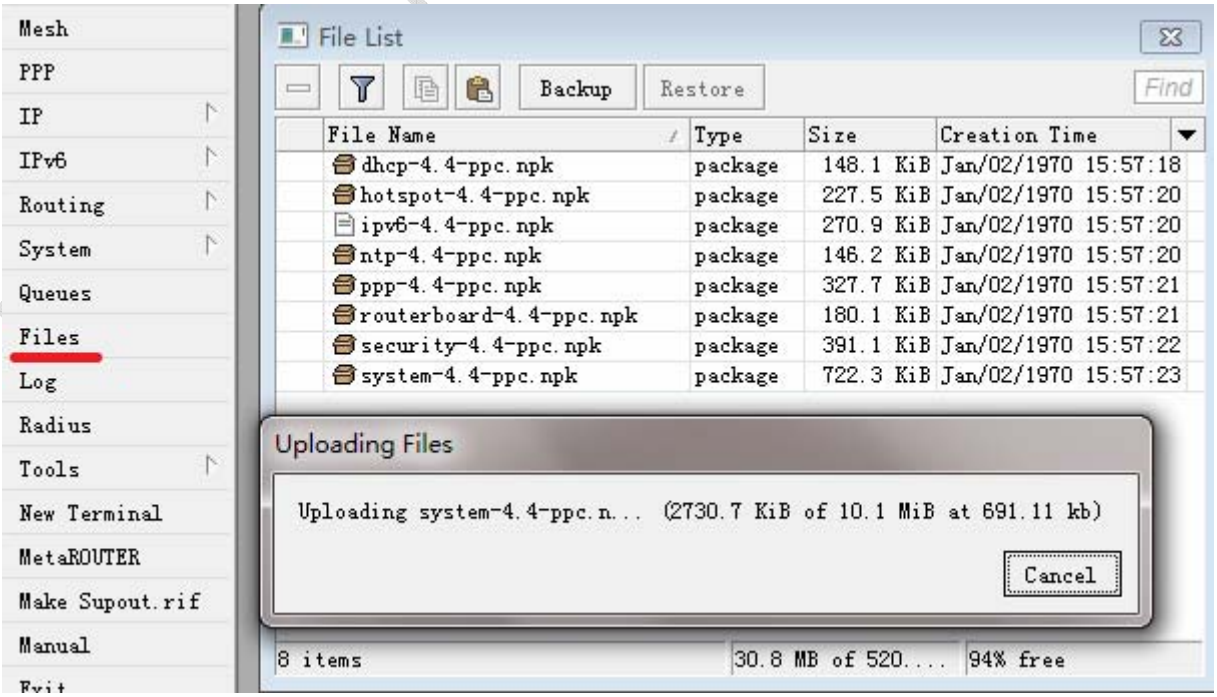
all_packages_x86 对应所有 x86 构架的 PC 设备（AMD、Intel、VIA 和其他 x86 PC）

2、 根据你使用 RouterOS 的情况不同，选择上传的升级包（***注：*system-x.x.x.npk** 的升级包是必须要，否则无法升级）。如何来确定你使用需要的功能，可以通过在 **system package>**的目录中查询对照。建议根据自己的需求安装或升级功能包(无线选择 wireless、PPPoE 认证选择 PPP 等等)，过多的安装功能会下降路由器的性能如下图：

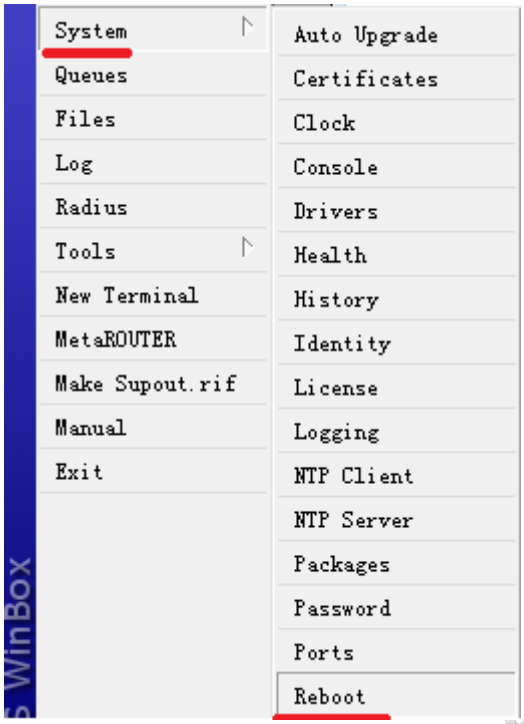


根据你在 system package 中的功能包选择 一一对应的功能包进行升级,systemg 功能包是必须安装的。

3、 选择好对应的 RouterOS 功能包后，通过“FTP：//路由器 IP 地址”上传功能包，或者直接通过打开 Winbox 的 Files 目录，通过拖放的方式将升级报上传到 RouterOS 根目录下：



4、 功能包上传完成后，通过 System Reboot 命令正常重启路由器：



RouterOS 在重启时，同时也在执行功能包的安装，重启后根据路由器性能不同会花费十几秒到 1 分钟的升级时间，如果是 PC 或者通过串口连接的 RB 设备可以在显示屏上看到安装进度条。重启完路由器后回看到路由器已经升级为新的版本。

降级选项

在 system package 中可以看的右上角有一个 Downgrade 的命令，这个将高版本降级到低版本的选项（需要同样将低版本的功能包上传到 RouterOS 的 FTP 的 files 中）。

2.8 升级 RouterBOARD 固件

RouterBOARD 产品启动 BOIS 程序都有更新，对 RouterBOARD 系列的启动引导和硬件兼容性进行修正和更新。RouterBOARD 固件后缀名为.fwf，每个系列的 RouterBOARD 所使用的估计都不相同，固件下载地址可以到 www.routerboard.com，如下面的列表：

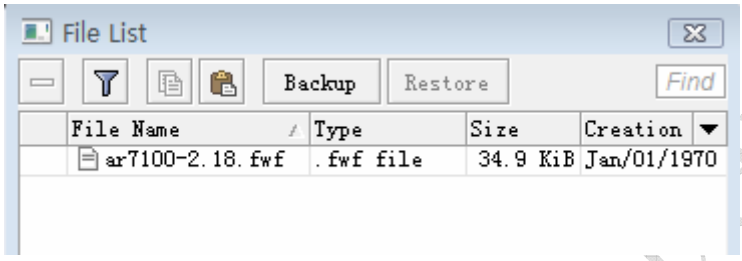
RB 型号	固件前缀
RB1000，RB1100	mpc8548
RB800	mpc8343
RB600	mpc8343
RB333	mpc8323
RB400 系列(411/A/AH、433/AH、433AH、450/G、493/AH)：	ar7100
RB700 系列（750、750G）	ar7100
RB532	rc32434
RB100 系列（112、133/C、150、192）	adm5120

每隔一段时间，RouterBOARD 的固件都会更新一次，所以通过在 RouterOS 中操作更新最新的 RouterBOARD 固件，升级固件只能在命令行操作，首先我们需要查看 RouterBOARD 当前的固件情况如下图：

```
[admin@Office] /system> routerboard
```

```
[admin@Office] /system routerboard> prin
          routerboard: yes
                model: "450"
        serial-number: "188901ED9E57"
    current-firmware: "2.16"
    upgrade-firmware: "2.18"
[admin@Office] /system routerboard>
```

如上面看到的，current-firmware 是当前的固件信息：2.16，而我们最新的估计是 2.18 所以我们要通过上传固件到 RouterOS 的 file 目录中（通过拖放的方式放到 winbox 中的 file list），当前的 RouterBOARD 是 RB450，所以这里我们上传的 ar7100 的固件文件如下图：



上传完后，然后我们通过 upgrade 命令升级：

```
[admin@Office] /system routerboard> upgrade
Do you really want to upgrade firmware? [y/n]
y
firmware upgraded successfully, please reboot for changes to take effect!
[admin@Office] /system routerboard>
```

按照提示升级固件，升级后要求重启设备，才可以更新。

注意：你也可以通过网络更新固件，要求 RouterBOARD 能连接到网络，会自动检测最新的固件，如果有新固件发布，也可以直接通过 upgrade 升级。

2.9 RouterOS 常用协议与端口

本文档列举了各种 MikroTik RouterOS 服务用到的协议及端口。它将帮助你决定为什么你的 MikroTik 路由器监听某些端口，以及如果你想要对某些服务禁止或者授权访问你都应该禁用/启用什么。请参见其他相关手册以获得更多解释。

操作路径: ***/ip service***

属性描述

name - 服务名称
port (*整型*: 1..65535) - 监听的端口
address (*IP 地址 掩码*; 默认: **0.0.0.0/0**) - 可使用服务的 IP 地址
certificate (*名称*; 默认: **none**) - （对于不需要认证的服务缺省）特定服务所使用的认证名称

实例

设置 WWW 服务能够从 10.10.10.0/24 网络 8081 端口可访问：

```
[admin@MikroTik] > ip service
[admin@MikroTik] /ip service> prin
Flags: X - disabled, I - invalid
 #   NAME                            PORT  ADDRESS          CERTIFICATE
 0   telnet                           23    0.0.0.0/0
 1   ftp                             21    0.0.0.0/0
 2   www                             80    0.0.0.0/0
 3 X www-ssl                          443   0.0.0.0/0        none
 4 X api                              8728  0.0.0.0/0
 5   winbox                           8291  0.0.0.0/0
[admin@MikroTik] /ip service>
[admin@MikroTik] ip service> set www port=8081 address=10.10.10.0/24
[admin@MikroTik] ip service> print
Flags: X - disabled, I - invalid
  #   NAME                             PORT  ADDRESS          CERTIFICATE
 0   telnet                           23    0.0.0.0/0
 1   ftp                             21    0.0.0.0/0
 2   www                             8081  10.10.10.0/24
 3 X www-ssl                          443   0.0.0.0/0        none
 4 X api                              8728  0.0.0.0/0
 5   winbox                           8291  0.0.0.0/0
[admin@MikroTik] ip service>
```

服务列表

以下便是 MikoTik RouterOS 服务所用的协议和端口的列表。一些服务需要安装附加功能包，并且需要管理员启用，例如：带宽服务器。

端口/协议	描述
20/tcp	文件传输协议 FTP [数据连接]
21/tcp	文件传输协议 FTP [控制连接]
22/tcp	安全命令行解释 SSH 远程登录协议 (仅与安全封装一起)
23/tcp	远程通信网络协议
53/tcp	域名服务器 DNS
53/udp	域名服务器 DNS
67/udp	自举协议 或 DHCP 服务器 (仅与 dhcp 功能包一起)
68/udp	自举协议 或 DHCP 客户 (仅与 dhcp 功能包一起)
80/tcp	万维网 (WWW) HTTP
123/udp	网络时间协议 NTP (仅与 ntp 功能包一起)
161/udp	简单网络管理协议 SNMP (仅与 snmp 功能包一起)

443/tcp	安全接口层 SSL 加密 HTTP(仅与 hotspot 功能包一起)
500/udp	Internet Key Exchange IKE protocol (仅与 ipsec 功能包一起)
520/udp	选路信息协议 RIP (仅与路由功能包一起)
521/udp	选路信息协议 RIP (仅与 routing 功能包一起)
179/tcp	边界网关协议 BGP (仅与 routing 功能包一起)
1080/tcp	SOCKS 代理协议
1701/udp	Layer 2 Tunnel Protocol L2TP (仅与 ppp 功能包一起)
1718/udp	H.323 Gatekeeper Discovery (仅与 telephony 功能包一起)
1719/tcp	H.323 Gatekeeper RAS (仅与 telephony 功能包一起)
1720/tcp	H.323 呼叫安装 (仅与 telephony 功能包一起 e)
1723/tcp	点对点隧道协议 PPTP (仅与 ppp 功能包一起)
1731/tcp	H.323 音频呼叫控制 (仅与 telephony 功能包一起)
1900/udp	通用即插即用 uPnP
2828/tcp	通用即插即用 uPnP
2000/tcp	带宽测试服务器
3986/tcp	Winbox 代理
3987/tcp	安全 winbox SSL 代理 (仅与安全功能包一起)
5678/udp	MikroTik Neighbor Discovery Protocol
8080/tcp	HTTP 网络协议 (仅与 WEB 代理功能包一起)
8291/tcp	Winbox
20561/udp	MAC winbox
5000+/udp	H.323 RTP 音频流 (仅与 telephony 功能包一起)
/1	ICMP - 网际控制报文协议
/4	IP - IP in IP (encapsulation)
/47	GRE - 普通路由封装 (仅限 PPTP 与 EoIP)
/50	ESP - IPv4 压缩的安全有效载荷(仅与安全功能包一起)
/51	AH - IPv4 认证标题 (仅与安全功能包一起)
/89	OSPFIGP - OSPF 内部网关协议
/112	VRRP - 虚拟路由器冗余协议

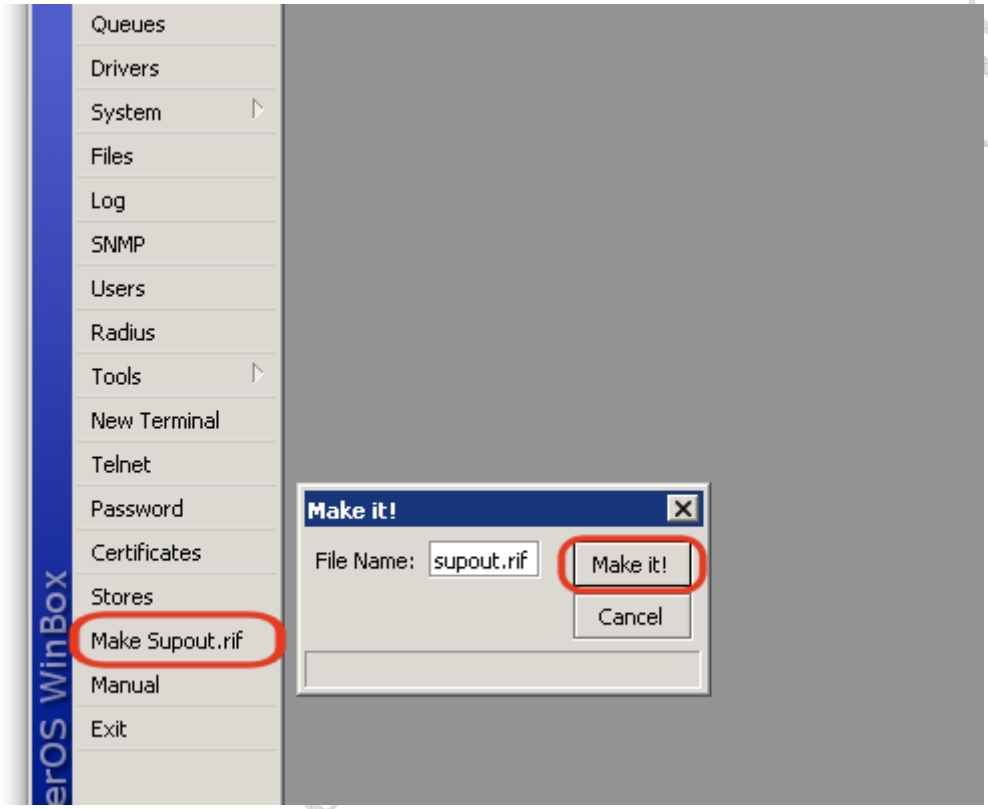
2.10 Supout.rif 技术支持文件

技术支持文件被用于调试 RouterOS 和快速解决技术支持问题，通过 Make supout.rif 功能，所有的 MikroTik RouterOS 的信息都被存储在这个技术支持文件（默认 supout.rif），生产后都会存储在路由器，可以通过 FTP 或者 winbox 将其下载。这个文件不会包含路由器密码，所以最好不要想他人透露文件信息，直接发送给 MikroTik 技术支持（support@mikrotik.com）

生成 Support 文件

Winbox 操作

生成技术支持文件（support Output file）通过点击 **Make Supout.rif**,



不要阻止文件生成过程，请等到 supout 窗口消失，可以在 files 目录下找到文件

Console 操作

生成 suppout.rif 在 console 下运行命令

```
[admin@MikroTik] >
[admin@MikroTik] >
[admin@MikroTik] >
[admin@MikroTik] >
[admin@MikroTik] >
[admin@MikroTik] >
[admin@MikroTik] >
[admin@MikroTik] > system sup-output
creating supout.rif file, might take a while
done
[admin@MikroTik] >
```

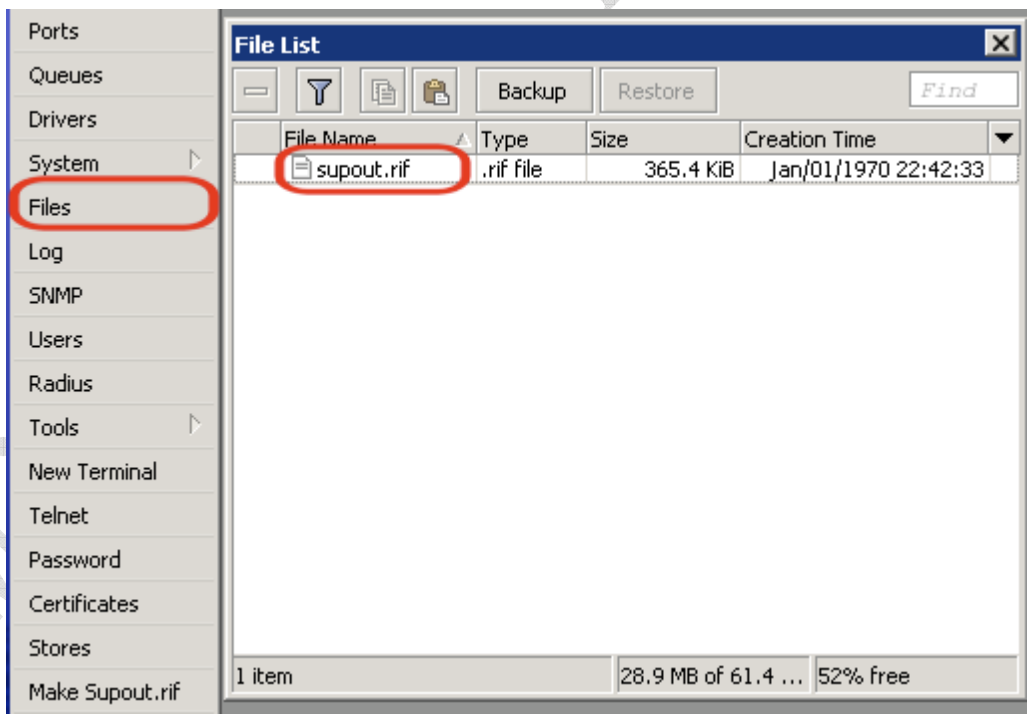
命令执行后，请等到 console 提示 **done**.

下载技术支持文件

技术支持文件能通过 FTP 下载，注意检查防火墙没有阻止 FTP 连接，且 RouterOS 本地 FTP 服务没有禁用

请在一些目录设置 FTP 服务：

```
/ ip service set ftp disabled=no
```



找到文件后，通过 e-mail 的附件，并说明问题发送给 MikroTik 技术支持(support@mikrotik.com)

第三章 MikroTik RouterBOARD 介绍

RouterBOARD 是 MikroTik 基于嵌入式平台为 RouterOS 开发路由主板，预装 RouterOS 系统，即硬件的 RouterOS，cisco 的路由器使用的是 IOS，MikroTik 的路由器则使用的是 RouterOS，都是由硬件和操作系统软件组成，他们之间仅仅是硬件和软件的不同，所以 RouterBOARD 是硬件路由器，而这样的路由器针对的是低功耗、高稳定性的无线网络和中小型有线网络。

如果我们换一种思路，我们把 PC 也当作一种路由器硬件平台，把 RouterOS 安装到 PC 上那是硬件路由器，只是 PC 对我们的日常生活来说太普片了，一时我们不能理解和接受，就叫 RouterOS 为软路由，其实市面上很多路由器使用的是嵌入式平台，如 ARM、MIPS 或者 Intel 的 IXP 等，他们只是换了一个硬件平台，运行的软件其实也是 Linux 嵌入系统或者 FreeBSD 等等这些软件和 RouterOS 本质上没有什么区别，要是他们的软件也开放出来，基于 PC 平台，大家都可以叫他软路由，只是开放了后系统就会贬值。

RouterBOARD 为低功耗的路由器，一般无扩展卡（无线网卡、USB 扩展和其他设备连接）功耗大多在 4-5w，PowerPC 处理器略高 5-12w 左右，具备 MiniPCI 或者 MiniPCI-e 扩展槽的设备，在扩张无线网卡后，根据扩张的数量和功率大小不同，功耗也有所变化。

我们可以把 RouterBOARD 分了 3 类：

1、 无线型 RouterBOARD：侧重于无线网络的 RouterBOARD 设备，如 RB411、RB711 等
2、 有线型 RouterBOARD：侧重于有线网络的 RouterBOARD 设备，如 RB450,、RB750、RB1100 等
3、 混合型 RouterBOARD：以上两者兼有，如 RB433、RB493 等

注：在之后的内容里，我将 RouterBOARD 缩写为 RB

3.1 RouterBOARD 的发展

早期的 RB 是 RB230，这款是 x86 的平台在 2002 年推出，虽然是低功耗平台，但非嵌入式，真正的嵌入式平台是从 2006 年的 RB112 开始，嵌入式的 RB 已经走过 5 年的时间，包括 RB100、RB300、RB400、RB500、RB600、RB700、RB800、RB1000、RB1100、RB1200，但许多型号已经被停产和淘汰，

2006 年

RB112、RB150、RB153、RB532、RB502 推出，接着推出了 RB133、RB133c、RB532rc5 和 RB192 等这些产品是 RB 第一代产品、RB100 和 RB500 系列为后期的 RB 产品奠定了基础，这些产品都是基于 MIPS 4kc 的处理器

2007 年

分别推出了 RB333 和 RB600 两款基于 PowerPC 平台，性能有大幅提升，等成本相对较高，只能算过度产品

2008-2009 年

推出了 RB400 系列、包括 RB411、RB433、RB450、RB493 等系列产品，到现在仍然是 RB 产品线的主流产品，RB1000 也在 08 年推出

2010 年

RouterOS4.0 支持 11n 协议后，RB 进入 11n 时代，并推出了 RB700 系列，RB711 针对 11n 的 5G 传输，针对低端市场的有线产品推出了 RB750 系列

2011 年至今

RB 将有更多新产品， 711 演变的 RBSXT 的 5G11n 室外设备、400 系列高性能版本 RB435G、2.4G 大功率 11n 的 RB711-2Hn、具备 USB 和 POE 的 RB750UP、集成 2.4G 11n 的 RB751 和扩展 USB 的 RB751U 和 RB751G。已经 RB1100、RB1100AH 和双核的 RB1100AH×2，还增加了一款万兆网卡的 RB1200，以及支持 SFP 光模块的 RB 和各种整合的无线设备

型号	基本信息	以太网口	MiniPCI	集成 WLAN
RB100 系列				
RB112	MIPS 4kc 175Mhz，16MB RAM	1×100M	2	无
RB133c	MIPS 4kc 175Mhz，16MB RAM	1×100M	1	无
RB133	MIPS 4kc 175Mhz，32MB RAM	3×100M	3	无
RB150	MIPS 4kc 175Mhz，32MB RAM	5×100M	无	无
RB153	MIPS 4kc 175Mhz，32MB RAM	5×100M	3	无
RB192	MIPS 4kc 175Mhz，32MB RAM	9×100M	2	无
RB500 系列				
RB502	MIPS 4kc 266Mhz，32MB RAM	1×100M	1	无
RB532	MIPS 4kc 266Mhz，32MB RAM	3×100M	2	无
RB532rc5	MIPS 4kc 399Mhz，64MB RAM	3×100M	2	无
RB300 系列				
RB333	PowerPC 333MHz, 64MB DDR RAM	3×100M	3	无
RBCRD 验证型				
RB/CRD	MIPS 4kc 184Mhz，32MB RAM	3×100M	无	802.11bg
RB400 系列				
RB411	Atheros 300Mhz , 32MB RAM (CPE)	1×100M	1	无
RB411R	Atheros 300Mhz , 32MB RAM (CPE)	1×100M	无	802.11bg
RB411A	Atheros 300Mhz , 64MB RAM	1×100M	1	无
RB411AR	Atheros 300Mhz , 64MB RAM	1×100M	1	802.11bg
RB411U	Atheros 300Mhz , 64MB RAM	1×100M	1+1pci-e	无
RB411AH	Atheros 680MHz （超频 800MHz）	1×100M	1	无
RB411UAHR	Atheros 680MHz （超频 800MHz）, 64MB RAM,1 USB	1×100M	1+1pci-e	802.11bg
RB433	Atheros 300Mhz , 64MB RAM	3×100M	3	无
RB433AH	Atheros 680MHz （超频 800MHz）, 128MB RAM	3×100M	3	无
RB433UAH	Atheros 680MHz , 128MB RAM,2 USB	3×100M	3	无
RB435G	Atheros 680MHz , 128MB RAM,2 USB	3×1G	5	无
RB493AH	Atheros 680Mhz , 64MB RAM	9×100M	3	无
RB493G	Atheros 680Mhz , 256MB RAM.1USB	9×1G	3	无
RB450	Atheros 300Mhz , 32MB RAM	5×100M	无	无
RB450G	Atheros 680Mhz（超频 800MHz）, 256MB RAM	5×1G	无	无
RB600 系列				
RB600	PowerPC 400MHz（超频 533MHz）, 64MB DDR RAM	3×1G	4	无
RB600A	PowerPC 400MHz（超频 533MHz）, 128MB DDR RAM	3×1G	4	无
RB700 系列				
RB711	Atheros 400MHz , 32MB RAM(CPE)	1×100M	无	802.11an

RB711A	Atheros 400MHz , 64MB RAM	1×100M	无	802.11an
RB711-2Hn	Atheros 400MHz , 32MB RAM(CPE), 1 USB	1×100M	无	802.11bgn
RB750	Atheros 300Mhz CPU, 32MB RAM	5×100M	无	无
RB750G	Atheros 680Mhz CPU, 32MB RAM	5×1G	无	无
RB750UP	Atheros 300Mhz CPU, 32MB RAM, 1 USB ,	5×100M	无	无
RB751	Atheros 300Mhz CPU, 32MB RAM,	5×100M	无	802.11bgn
RB751U	Atheros 300Mhz CPU, 32MB RAM, 1 USB	5×100M	无	802.11bgn
RB751G	Atheros 680Mhz CPU, 32MB RAM, 1 USB	5×1G	无	802.11bgn
RBSXT	Atheros 400MHz , 32MB RAM(CPE), 1 USB	1×100M	无	802.11an
RB800 系列				
RB800	PowerPC 800MHz ， 256M DDR RAM,1 CF	3×1G	4+1pci-e	无
RB1000 系列				
RB1000	PowerPC 1.3GHz ， 512M DDR RAM	4×1G	无	无
RB1100	PowerPC 800MHz ， 512M DDR RAM	13×1G	无	无
RB1100AH	PowerPC 1066MHz ， 2G DDR RAM	13×1G	无	无
RB1100AH×2	PowerPC 双核 ， 2G DDR RAM	13×1G	无	无
RB1200	PowerPC 1066MHz ， 2G DDR RAM	10×10G	无	无

RB 淘汰和停产的情况

- RB100 系列-淘汰
- RB200 系列-淘汰
- RB/CRD-淘汰
- RB300 系列-淘汰
- RB400 系列：RB411-停产，RB411A-停产，RB411UAHR-停产、RB411R-停产
- RB500 系列-淘汰
- RB600 系列-淘汰
- RB1000 系列：RB1000 –淘汰

RouterBOARD 基本编号适合大多型号，除 RB600，RB800 和 RB1000 系列型号外，他们的区别如下：

- RB1XX，即 RB100 系列
- RB133，即是 100 系列，有 3 个以太网口，3 个 MiniPCI 无线扩展接口
- RB493，即 400 系列，有 9 个以太网口，3 个 MiniPCI 扩展

后缀代表含义：

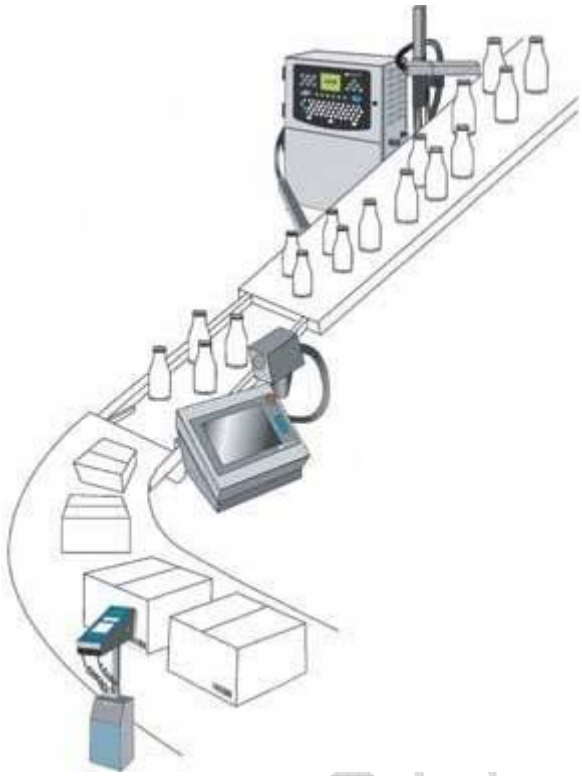
- AH，A 代表高内存，H 代表高性能（高 CPU）
- G，代表高性能和千兆网口
- U，代表 USB 扩展
- R，代表集成无线模块
- P，POE

更多具体的 RouterBOARD 信息请访问 www.routerboard.com 的网站

3.2 RouterBOARD Throughput（吞吐量）

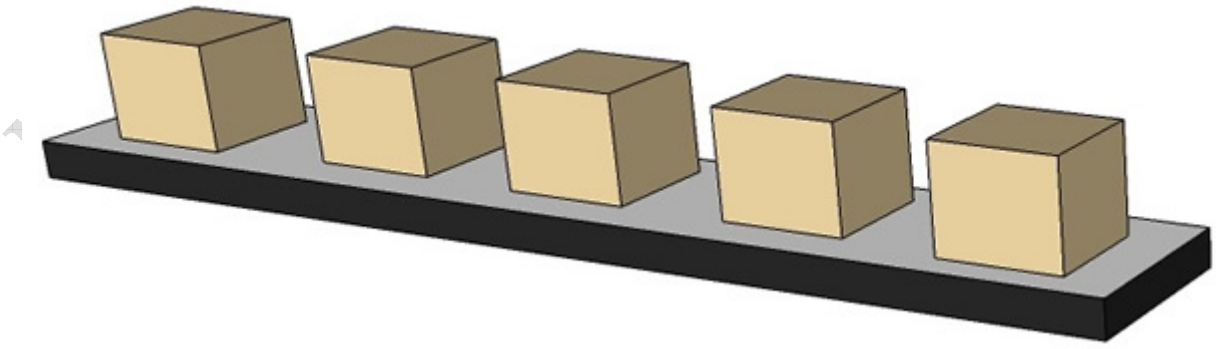
路由器 Throughput，一般是以太网到另一以太网的 Throughput，数据流出或流入需要路由器处理，这代表了路由器性能。RouterBOARD 仅给出了桥接和路由下开启和关闭防火墙的结果，并没有给出真正的 nat 测试结果，这些仅作参考。

首先我们要了解什么是 throughput，我们知道一般数据包处理都要经过 CPU，而 CPU 的处理能力和速度直接决定转发数据包的能力，处理数据包就像流水线一样，对每个包进行拆包、分析、重新封装和转发

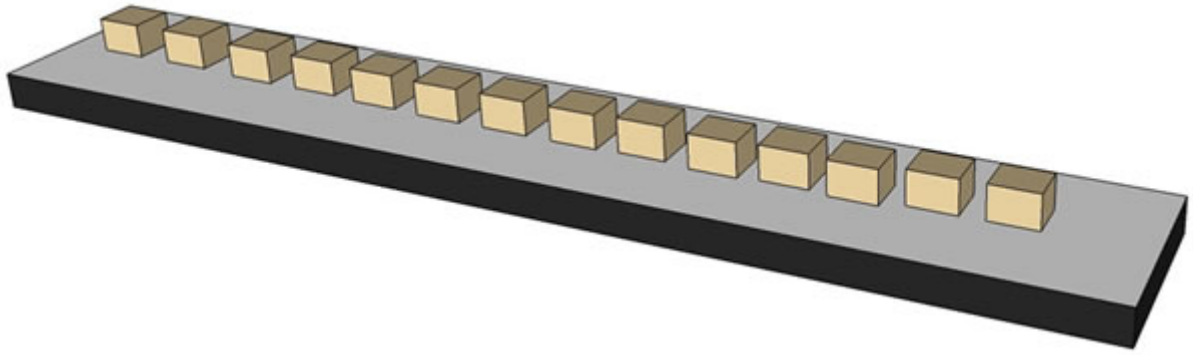


数据包的大小也决定了处理速度和吞吐量、如 128Byte 包每秒能处理 10000 个，并不能做到 64 Byte 包每秒处理 20000 个，而是只比 10000 个略多一点点，比如 10100 个。如他的路由器在处理最大的 1518Byte 包时每秒 8000 个，根据理论计算处理 1518Byte 包 100M 线速的极限值是 8127 个，所以折算出来的 Throughput 就是 100M*8000/8127=98.44M，有些厂家就很自豪地宣布，我的路由器 Throughput 高达 98.44M，殊不知，原来这个路由器在处理最小的 64Byte 包时每秒是 11000 个，根据理论计算处理 64Byte 包 100M 线速的极限值是 148810 个，所以折算出来的 Throughput 只有 100M*11000/148810=7.39M，两者相差 13 倍多。

为什么数据包大小能影响路由器的吞吐量能力，我们举一个例子，假设一个员工在流水线上检验每个产品的包装是否合格，第一天流水线上是 4 个产品装成 1 个大箱子的包装，他一分钟能检查 20 个大箱包装，即一分钟 80 个产品的包装就检验完成，但事实上他只检查了 20 个大箱子的包装，并非 80 个产品的每个包装，



结果第二天工厂要求每个产品的包装都要检查，但他每分钟仍然只能检查 20 个产品的包装，反而比昨天检验 80 个产品产品少了 4 倍，和昨天的 4 个产品一个大包装完全不一样，导致今天只有 20 个产品，是因为包装方式完全不一样。如果要让这个员工检验每个产品的包装提高到 80 个，只能是提升员工的检验水平，要让员工提高到每分钟 80 个产品的检验，肯定这个员工肯定要疯掉。



这个道理和我们的路由器处理大包和小包的道理是一样的，CPU 处理大数据包很轻松，而处理小数据包就很吃力，大包装的数据多，一次就可以转发很多数据，而小包数据少，数量又多，很考验 CPU 的性能。我们衡量路由器的吞吐量是以 64byte 的小包，在每秒钟转发了多少个包，单位是 pps（per packet seconds）

参照思科官方 Cisco 3745 两个百兆以太网端口在 64 字节时达到 225018pps 的转发速率，即 225kpps。RB1100AH 在 1333MHz 的频率下路由模式 262kpps，桥接模式下 400kpps

RouterBOARD 测试结果又以下几种：

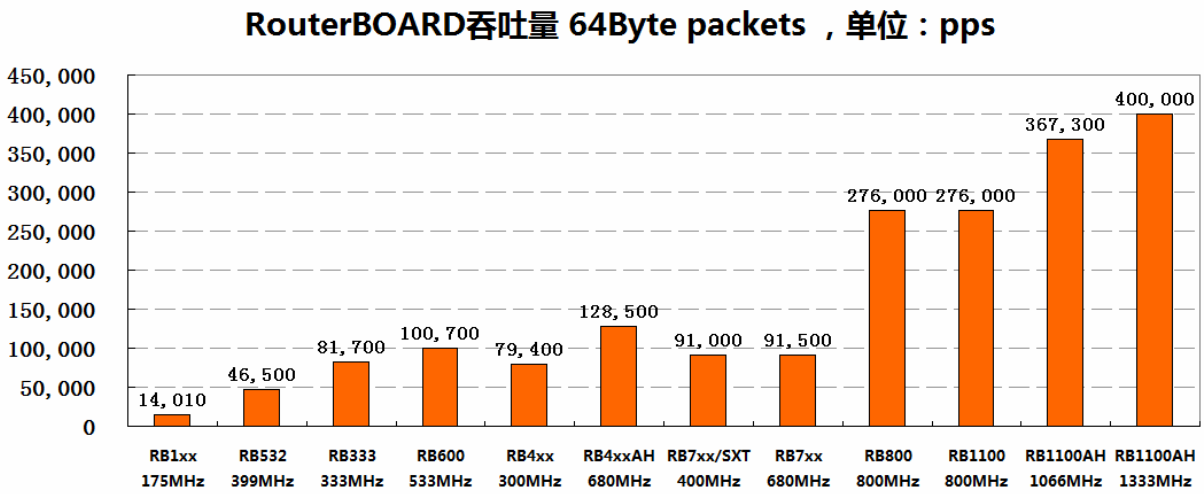
- 64byte 包吞吐量反应了 CPU 的性能
- 1500byte 包吞吐量反应了内存性能
- 512byte 包反应了 CPU 和内存整体性能

RouterBOARD 最大以太网吞吐量，对照表可以在下面的链接找到：
http://www.routerboard.com/pdf/routerboard_performance_tests.pdf

所有测试是通过以下方式完成：

- 通过路由器转发测试 through the router
- 精简 RouterBOARD 设置，仅安装 system 功能包
- 通过 Agilent N2X 设备测试

以下是 RouterBOARD 在 64byte 数据包，使用桥接模式下的对比图



3.3 RouterBOARD 的型号与分析

RB411 系列与 RB433 系列

- RB411 与 RB433 主要区别是无线 miniPCi 的接口数量不同，根据安装网卡数量的不同对应不同的应用场合。
- RB411 与 RB433 都可以作为城市的无线覆盖产品，但根据节点不同，应用方式也不同。
- 433 系列主要用于中继与多点覆盖和多点传输
- 411 系列主要用于普通无线覆盖，点对点，与点对多点的低成本解决方案

RB411AR 与 RB711

- 都是集成无线网卡，RB411AR 更侧重于 WiFi 覆盖，RB711 和 RB711A 侧重于 5G 骨干无线传输
- RB711/A 是集成 5G-a/n 的无线网卡， 23dBm 仅支持 5G 传输，不支持 2.4G， 支持协议是 802.11a 和 802.11n，不支持 MiniPCI 扩展
- RB411R/AR 是集成 2.4G 无线网卡，仅支持 2.4G 传输，不支持 5G，支持协议是 802.11bg
- RB411R 没有 MiniPCI 扩展功能，RB411AR 支持 1 个 MiniPCI 槽扩展
- 新的 RB711-2Hn 更多考虑到 11n 的 WiFi 覆盖

RB450 系列与 RB750 系列

- RB450 和 RB450G 是较早的 5 口路由器，没有无线功能，CPU 分别是 300MHz 和 680MHz，主板与外壳分开销售，
- RB750 和 RB750G 其实是 RB450 和 RB450G 的简化版本，销售用一整套塑料外壳和电源 RB411R/AR 是集成 2.4G 无线网卡，仅支持 2.4G 传输，不支持 5G，支持协议是 802.11bg
- RB450 的 CPU 是 AR7130 300MHz 其实不如 RB750 的 AR7240 的 400MHz，但 MikroTik 为了不让 RB450 在性能和价格上尴尬境地，刻意将将后期的 RB750 的 CPU 降频到 300MHz。
- RB450G 整体性能上要强于 RB750G，不管从内存还是交换芯片，都好于 RB750G，他们的 CPU 都相同，所以性能上 RB450G 强于 RB750G，但性价比上 RB750G 更有优势
- RB750 和 RB450 完全能满足 50 台电脑的网络环境，RB450G 表现就比较好，如果不建议 CPU 较高，可以支持 180 台电脑的网络，RB750G 就相对低点 150 台
- 从发展来看 RB750 系列被 MikroTik 重点发展，后期会增加 RB751 系列，增加 USB 和集成 11n 的无线网卡

注：RB400 支持 switch 功能，即通过 IC 控制二层数据转发，不需要经过 CPU 处理（RB100 系列有这样的功能，但不被完全支持），也具备数据镜像的设置

RB1100 系列

RB1100 是 13 个千兆以太网口的，非常满足多线路接入的网络，一次可以接入 12 条外线，这样节省了交换机的费用，同样如果你只有一条外线可以把 12 个以太网口设置成类似交换机的功能，直接替代了主交换机的功能，RB1100AH 和 RB1100AH×2 也是同样的 13 个口，只是性能更强。

RB1200 的 CPU 和 RB1100AH 相同，只是采用了 10 个万兆网卡的解决方案，看似网卡吞吐量大了，但设备性能没有任何提升

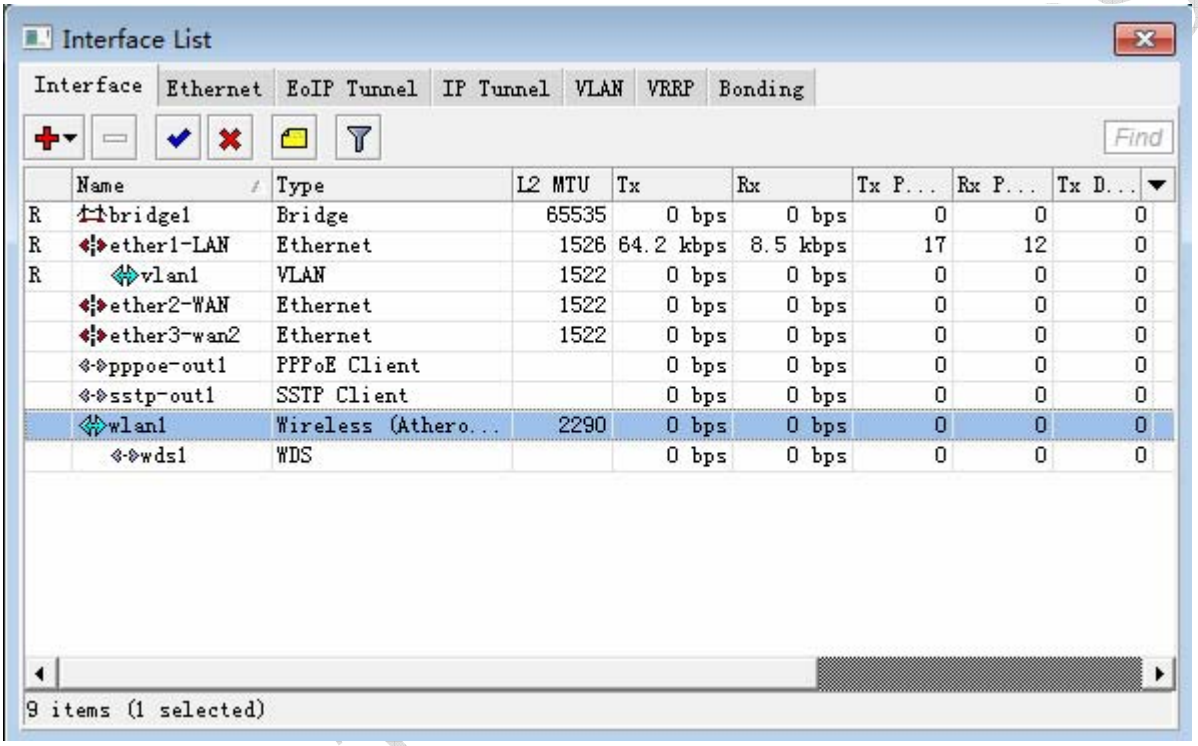
> 注：RB1000 产品是 MikroTik 第一款 1.3G 处理器的高性能路由器，但也是最失败的，因为返修率太高，曾经出现 10 台里有 4-5 台无法启动，但 MikroTik 一直否认，这就是为什么后来被 800MHz 处理器的 RB1100 替代，RB1100 系列（RB1100、RB1100AH 和 RB1100AH×2）作为 13 口的多线路由器，针对有线网络设计，特别是网吧和小区宽带，这款产品是 MikroTik 比较成功的产品，特点是性能够、针对性强！

第四章 接口配置（Interface）

4.1 Interface 基本操作

在 interface 中包括物理接口网卡配置与虚拟接口的网卡配置，物理接口：Ethernet、wireless、ISDN 等，虚拟接口：PPP、PPPoE、PPTP、L2TP、SSTP、EoIP、IPIP 和 Bonding 等等。

MikroTik RouterOS 支持各种网络接口卡，同样也支持一些虚拟接口像 VLAN、Bridge 等。 这些接口属性在接口列表中可以按你的需要进行配置。



操作路径：***/interface***

属性描述

name (文本) – 接口名称
status – 显示接口状态
type (只读: arlan | bridge | cyclades | eoip | ethernet | farsync | ipip | isdn-client | isdn-server | l2tp-client | l2tp-server | moxa-c101 | moxa-c502 | mtsync | pc | ppp-client | ppp-server | pppoe-client | pppoe-server | pptp-client | pptp-server | pvc | radiolan | sbe | vlan | wavelan | wireless | xpeed) – 接口类型
mtu (*整型*) – 接口最大传输单位(bytes)
rx-rate (*整型*; 默认: **0**) – 最大数据接收率
0 - no limits
tx-rate (*整型*; 默认: **0**) – 最大数据发送率
0 - no limits

例如：

查看下面的接口列表：

```
[admin@MikroTik] interface> print
Flags: X - disabled, D - dynamic, R - running
 #    NAME                      TYPE           RX-RATE    TX-RATE    MTU
 0  R ether1                     ether          0          0          1500
 1  R bridge1                    bridge         0          0          1500
 2  R ether2                     ether          0          0          1500
 3  R wlan1                      wlan           0          0          1500
[admin@MikroTik] interface>
```

进入/interface bridge 桥接配置，添加一个桥：

```
[admin@MikroTik] /interface bridge> add
[admin@MikroTik] /interface bridge> prin
Flags: X - disabled, R - running
 0  R name="bridge1" mtu=1500 arp=enabled mac-address=00:00:00:00:00:00
     protocol-mode=none priority=0x8000 auto-mac=yes
     admin-mac=00:00:00:00:00:00 max-message-age=20s forward-delay=15s
     transmit-hold-count=6 ageing-time=5m
[admin@MikroTik] /interface bridge>
```

4.2 流量监视

指令名称：***/interface monitor-traffic***

注：可以监控通过接口的任何数据流量，并且能同时监视多个网卡的流量情况，在执行监控命令是，参数只能是网卡名称，不能使用网卡的编号

例如：在命令行下的多网卡流量监视：

```
[admin@MikroTik] interface> monitor-traffic ether1,wlan1
   received-packets-per-second: 1       0
      received-bits-per-second: 475bps   0bps
       sent-packets-per-second: 1       1
          sent-bits-per-second: 2.43kbps 198bps
-- [Q quit|D dump|C-z pause]
```

4.3 以太网接口（Ethernet）

MikroTik RouterOS 支持各种以太网卡，完全支持的以太网卡型号可以通过登陆 mikrotik.com.cn 的技术支持中查找到

功能包需要：***system***
等级需要：*Level1*
操作路径：***/interface ethernet***
标准与技术协议：IEEE 802.3

以太网接口配置

操作路径：***/interface ethernet***

属性描述

arp (*disabled* | *enabled* | *proxy-arp* | *reply-only*; 默认: **enabled**) **-** 地址解析协议的模式

auto-negotiation (*yes* | *no*; 默认: **yes**) – 当被启用，接口将发挥最大的性能获得最好的连接

注: Auto-negotiation 在两个终端必须禁用，否则以太网将不能正常工作

注 2: Gigabit（千兆）传输不能工作在 auto-negotiation

bandwidth(*整型/整型*，默认: **unlimited/unlimited**) – 设置最大的接口带宽 rx/tx 带宽（该功能仅限于特定的 RouterBOARD）

cable-setting (*default* | *short* | *standard*; 默认: **default**) – 修改网线连接长度 (仅适用于 NS DP83815/6 网卡)

disable-running-check (*yes* | *no*; 默认: **yes**) – 路由器网卡自动探测网络设备功能，如果这个参数设置为“no”，路由器网卡将禁用监测功能

full-duplex (*yes* | *no*; 默认: **yes**) – 定义数据传输是否同时在两个方向上，即全双工。

l2mtu (*整型*; 默认:) – 二层最大传输单位

mac-address (*MAC*; 默认:) – 一个网卡的介质访问控制地址

master-port (*name* | *none*; 默认: **none**) – 设置交换组的主接口

mdix-enable (*yes* | *no*; 默认:) – 是否在该接口上开启 MDI/X 自动线序纠正功能

mtu (*integer*; 默认: **1500**) – 三层最大传输单位

name (*string*; 默认:) – 一个网口的名称

speed (*10Mbps* | *100Mbps* | *1Gbps*; 默认: **最大带宽**) – 设置网卡的数据传输速度，默认情况下根据网卡支持最大传输单位

例如: interface 操作

```
[admin@MikroTik] /interface ethernet> print detail
Flags: X - disabled, R - running, S - slave
 0 R  name="ether1" mtu=1500 l2mtu=1526 mac-address=00:0C:42:37:58:66
      arp=enabled auto-negotiation=yes full-duplex=yes speed=100Mbps

 1    name="ether2" mtu=1500 l2mtu=1522 mac-address=00:0C:42:37:58:67
      arp=enabled auto-negotiation=yes full-duplex=yes speed=100Mbps
      master-port=none bandwidth=unlimited/unlimited switch=switch1

 2    name="ether3" mtu=1500 l2mtu=1522 mac-address=00:0C:42:37:58:68
      arp=enabled auto-negotiation=yes full-duplex=yes speed=100Mbps
      master-port=none bandwidth=unlimited/unlimited switch=switch1

 3    name="ether4" mtu=1500 l2mtu=1522 mac-address=00:0C:42:37:58:69
      arp=enabled auto-negotiation=yes full-duplex=yes speed=100Mbps
      master-port=none bandwidth=unlimited/unlimited switch=switch1

 4    name="ether5" mtu=1500 l2mtu=1522 mac-address=00:0C:42:37:58:6A
      arp=enabled auto-negotiation=yes full-duplex=yes speed=100Mbps
      master-port=none bandwidth=unlimited/unlimited switch=switch1
[admin@MikroTik] /interface ethernet>
```

接口状态监测命令

指令名称：***/interface ethernet monitor***

属性描述

status (link-ok | no-link | unknown) – 接口的状态，情况为：
link-ok – 网卡以连接到网络
no-link – 网卡没有连接到网络
unknown – 连接未确认
rate (10 Mbps | 100 Mbps | 1000 Mbps) – 实际的连接速率
auto-negotiation (done | incomplete) – 相邻连接的状态判断。
done – 判断完成
incomplete – 判断失败
full-duplex (yes | no) – 是否为全双工数据传输

例如：

通过 Monitor 命令可以查看现在以太网卡的连接状态，link-ok 为以太网络连接正常：

```
[admin@MikroTik] interface ethernet> monitor ether1,ether2
           status: link-ok link-ok
  auto-negotiation: done    done
             rate: 100Mbps 100Mbps
      full-duplex: yes     yes
```

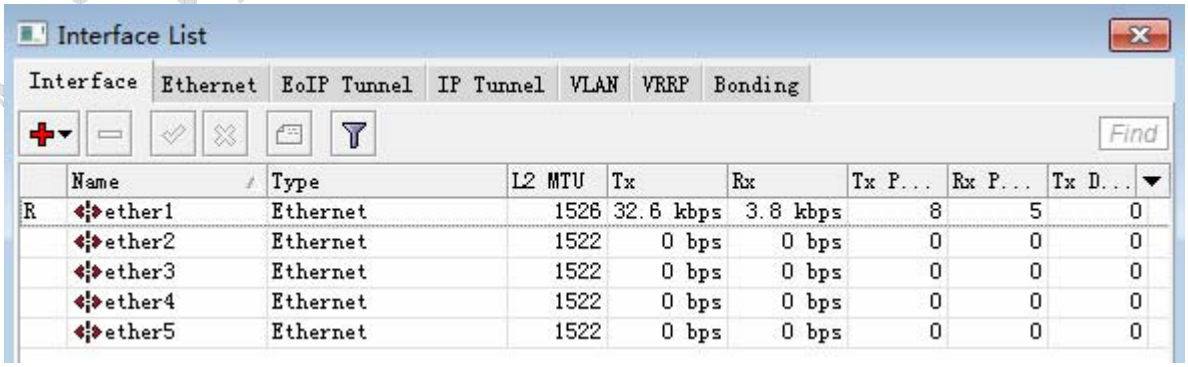
修改以太网卡 mac 地址：

```
[admin@MikroTik] interface ethernet>set 0 mac-address=00:0C:42:03:11:0A
```

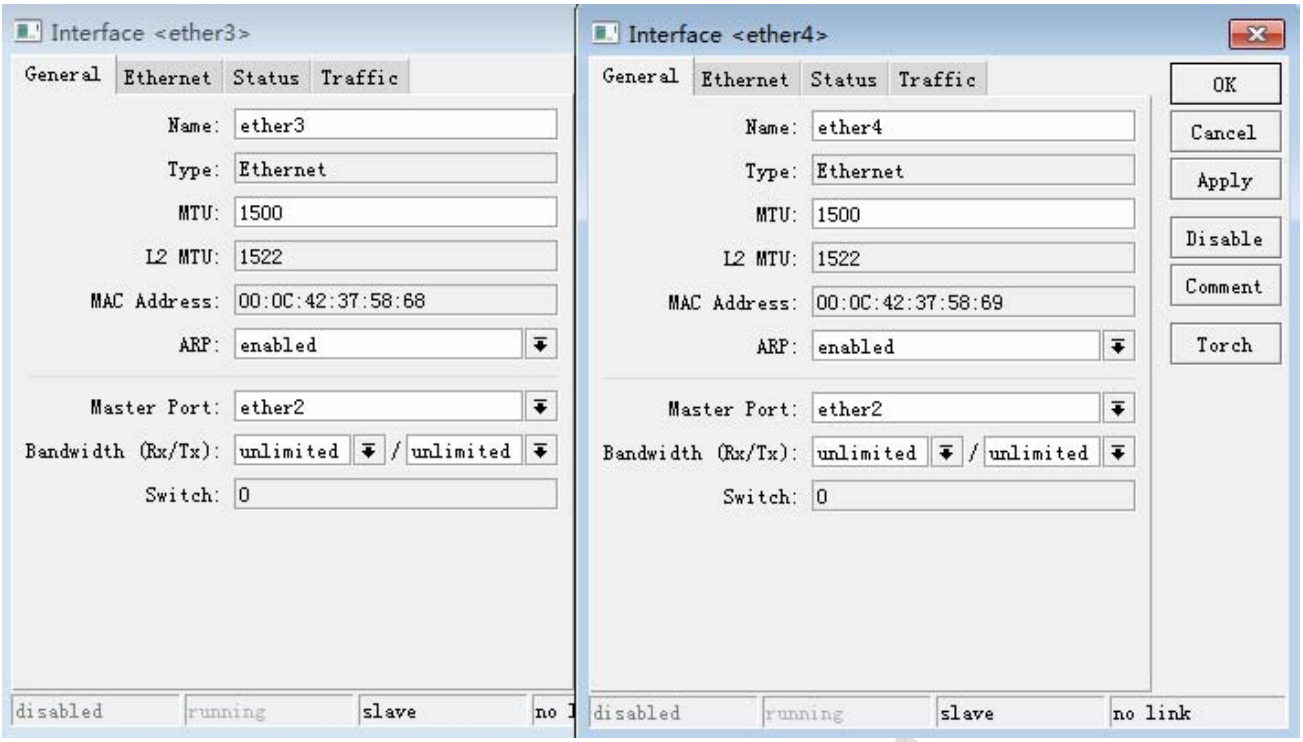
4.4 RouterBOARD 交换配置

在 RB100 系列、RB400 系列、RB700 系列支持网卡的交换功能，即设置 Master 端口就可以将多个网卡进行以太网交换，而非采用 bridge 方式的软件二层交换，这样的功能不需要通过 CPU 处理。

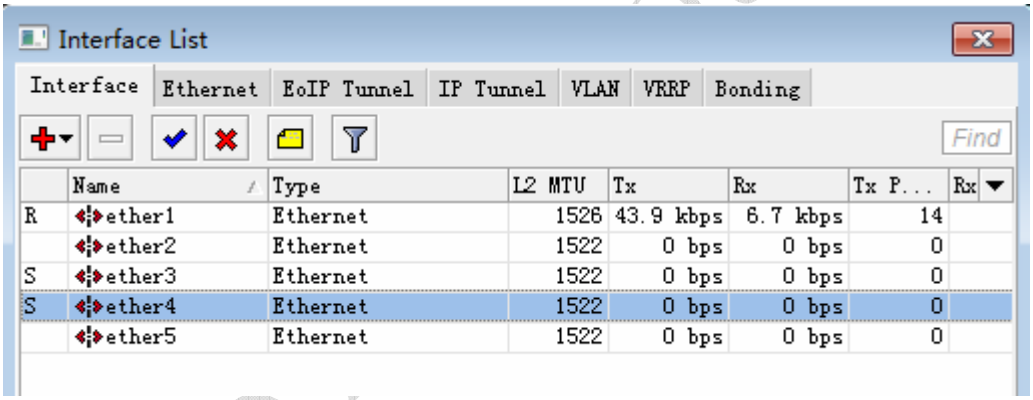
如下图，我们可以看到一个 RB450 的 5 个以太网接口，我们需需要将 ether3、ether4 与 ether2 进行交换功能配置：



打开 ether3 和 ether4 接口配置，并将 Master Port 为 ether2



设置完成后，我们可以在 interface 列表中看到在



第五章 IP 配置与 ARP

下面讨论 IP 地址管理和地址解析协议设置， 通过基于 TCP/IP 协议我们可以使用 IP 地址连接其它网络设备，并借助于地址解析协议（ARP）与同一子网的设备通信。

功能规格

需要功能包：***system***
需要等级：*Level1*
操作路径：***/ip address***, ***/ip arp***
标准与技术： IP, ARP
硬件应用：无要求

5.1 IP 地址

操作路径：***/ip address***

Internet 上的每台主机(Host)都有一个唯一的 IP 地址。IP 协议就是使用这个地址在主机之间传递信息，这是 Internet 能够运行的基础。IP 地址的长度为 32 位，分为 4 段，每段 8 位，用十进制数字表示，每段数字范围为 0～255，段与段之间用句点隔开。一个完整的 IP 地址选址需要子网掩码配置，子网掩码区分了不同网段的 IP 地址。

RouterOS 能在一个接口上添加多个 IP 地址， 当桥模式在两个接口间被使用，在物理接口上添加 IP 地址并不是必须的（此从 RouterOS 的 2.8 版本起），在桥模式的事例中，IP 地址能分配给属于桥模式的任何接口，但实际上地址将属于桥接口。你能使用/ip address print detail 查看地址归属的接口。

MikroTik RouterOS 有下面的地址类型：

- **Static** – 用户手动分配给接口
- **Dynamic** – 确定 ppp, ppptp, 或 pppoe 以连接，自动分配的接口

属性描述

address (*IP 地址*) – 主机的 IP 地址，组成 X.X.X./子网掩码
broadcast (*IP 地址*; 默认: **255.255.255.255**) – 广播 IP 地址，通过默认 IP 地址和子网掩码自动计算出的
disabled (yes | no; 默认: **no**) – 指定那一个地址禁用或启用
interface (*名称*) – 接口名称
actual-interface (只读: *名称*) – 仅适用于逻辑接口，像桥（bridges）或隧道（tunnels）
netmask (*IP 地址*; 默认: **0.0.0.0**) – 指明网络地址，属于一个 IP 地址的一部份。
network (*IP 地址*; 默认: **0.0.0.0**) – IP 地址网段。点对点连接时，网段到远端地址结束。

注:你不能有两个不同的 IP 地址来至相同的网段，例如:**10.0.0.1/24** 地址分配到 **ether1** 接口上，并且 **10.0.0.132/24** 地址分配到 **ether2** 接口上，这样是非法的。因为这两个地址属于同一个网段 **10.0.0.0/24。**

例如: 添加 IP 地址 10.10.10.1/24 到 ether2 接口上

```
[admin@MikroTik] ip address> add address=10.10.10.1/24 interface=ether2
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS           NETWORK        BROADCAST      INTERFACE
  0   2.2.2.1/24         2.2.2.0        2.2.2.255      ether2
  1   10.5.7.244/24      10.5.7.0       10.5.7.255     ether1
  2   10.10.10.1/24      10.10.10.0     10.10.10.255    ether2

[admin@MikroTik] ip address>
```

5.2 地址解析协议 ARP

操作路径：***/ip arp***

每个主机通过 IP 地址通信，但同一子网的设备，通过 MAC 地址进行数据交换，三层通信建立在二层的基础上，地址解析协议用于 OSI 第三层解析第二层的 IP 和 MAC。 一个路由器会有一个当前 ARP 列表，同一子网通信通常是建立为动态 ARP 列表，但为增强网络稳定性和安全性，可建立静态 ARP 列表，如防御 ARP 病毒。

属性描述

address (*IP 地址*) – 对应的 IP 地址
interface (名称) – 被分配 IP 地址的接口名称
mac-address (*MAC 地址*; 默认: **00:00:00:00:00:00**) – 相应的 MAC 地址

注： 最大的 ARP 的登记数为 8192.

如果 ARP 功能在接口上被关闭，例如：使用 **arp=disabled**，来至客户端的 ARP 请求将不被路由器回应，因此必须添加静态的 ARP 才行。 例如，通过 **arp** 命令将路由器的 IP 和 MAC 地址必须添加到 windows 工作站中：

```
C:\> arp -s 10.5.8.254  00-aa-00-62-c6-09
```

如果在接口上的 **arp** 属性设置为 **reply-only**，这时路由器只应答来至静态 ARP 的请求。邻近的 MAC 地址将通过**/ip arp** 设置唯一的静态 ARP 列表。

例如：

```
[admin@MikroTik] ip arp> add address=10.10.10.10 interface=ether2
mac-address=06:21:00:56:00:12
[admin@MikroTik] ip arp> print
Flags: X - disabled, I - invalid, H - DHCP, D - dynamic
  #   ADDRESS         MAC-ADDRESS       INTERFACE
  0 D 2.2.2.2         00:30:4F:1B:B3:D9 ether2
  1 D 10.5.7.242      00:A0:24:9D:52:A4 ether1
  2   10.10.10.10     06:21:00:56:00:12 ether2
[admin@MikroTik] ip arp>
```

如果在一个接口上使用静态 ARP 记录会使网络更安全，你必须将该接口上的 arp 设置为 'reply-only' ，相关操作在下面的 **/interface** 目录中：

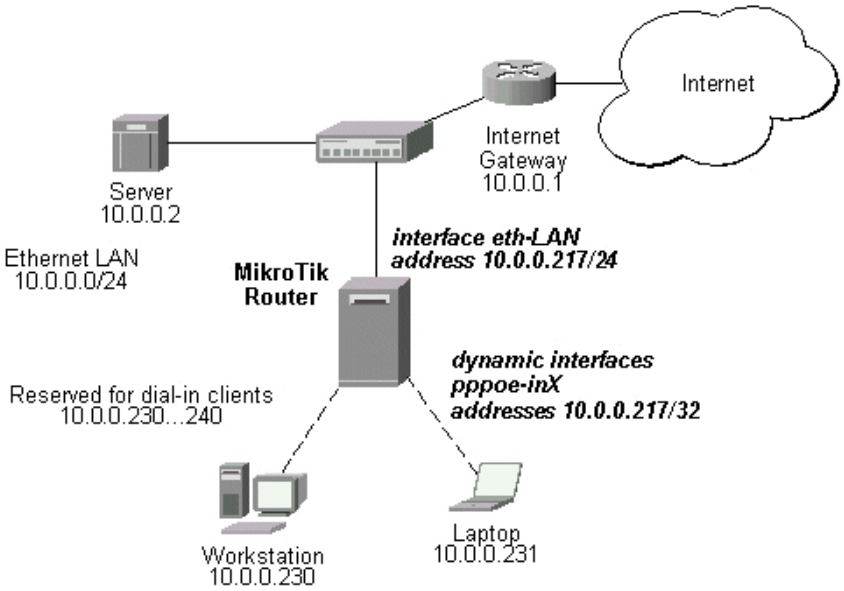
```
[admin@MikroTik] ip arp> /interface ethernet set ether2 arp=reply-only
[admin@MikroTik] ip arp> print
Flags: X - disabled, I - invalid, H - DHCP, D - dynamic
  #   ADDRESS         MAC-ADDRESS       INTERFACE
  0 D 10.5.7.242      00:A0:24:9D:52:A4 ether1
  1   10.10.10.10     06:21:00:56:00:12 ether2

[admin@MikroTik] ip arp>
```

5.3 ARP 代理

所有的物理接口，像以太网、Atheros 和 Prism (wireless), Aironet (PC), WaveLAN 等，都可设置地址解析协议或不设置。其他则可设置使用 ARP 代理。如果 ARP 请求是从一个网络的主机发往另一个网络上的主机，那么连接这两个网络的路由器就可以回答该请求，这个过程称作委托 ARP 或 ARP 代理(ProxyARP)。这样可以欺骗发起 ARP 请求的发送端，使它误以

为路由器就是目的主机，而事实上目的主机是在路由器的“另一边”。路由器的功能相当于目的主机的代理，把分组从其他主机转发给它

例如： 看下列的网络配置：



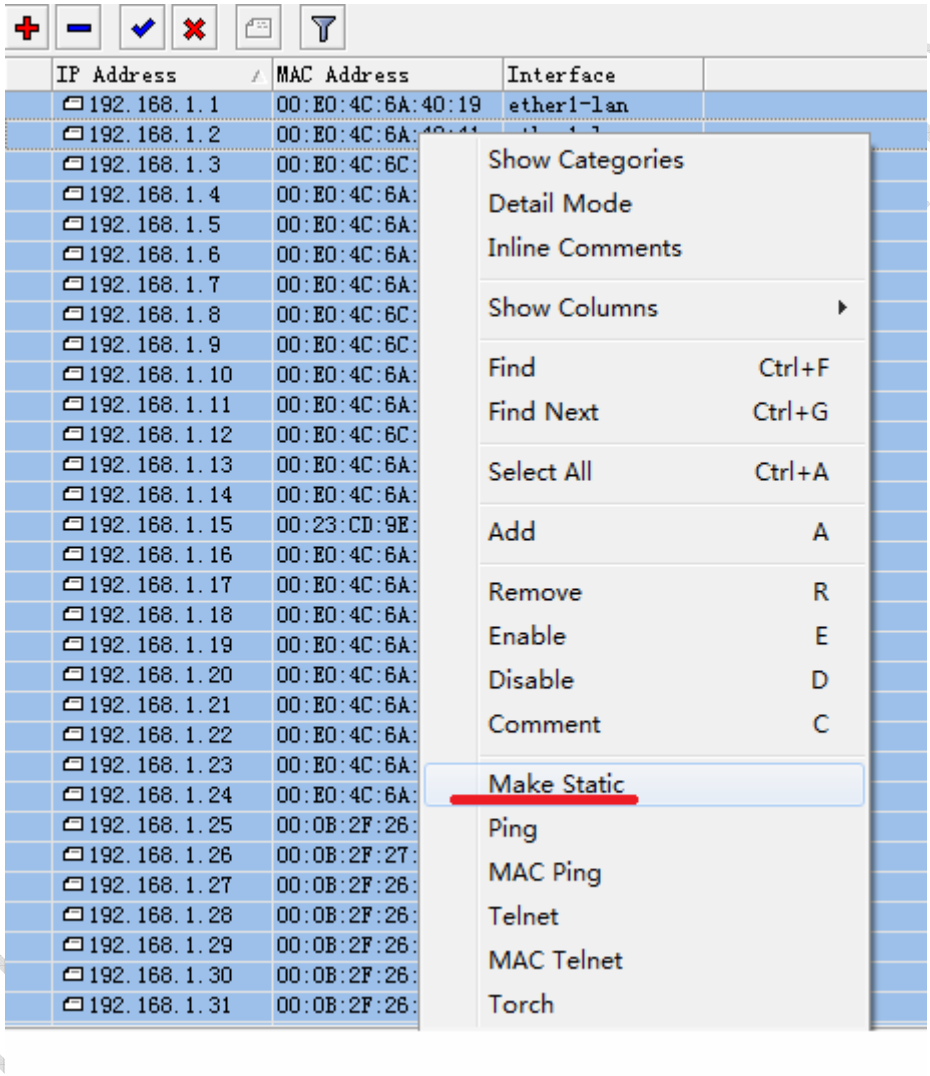
下面是 Router 设置：

```
admin@MikroTik] ip arp> /interface ethernet print
Flags: X - disabled, R - running, S – slave
#    NAME        MTU     MAC-ADDRESS      ARP      MA.. SWITCH
0 R  ether1      1500  00:0C:42:11:54:F5 enabled    none 0
 [admin@MikroTik] ip arp> /interface print
Flags: X - disabled, R - running, D - dynamic, S - slave
  #    NAME                                      TYPE           MTU
  0 R  ether1                                    ether          1500
  1    prism1            prism          1500
  2 D  pppoe-in25         pppoe-in
  3 D  pppoe-in26         pppoe-in
[admin@MikroTik] ip arp> /ip address print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS          NETWORK        BROADCAST      INTERFACE
  0   10.0.0.217/24     10.0.0.0       10.0.0.255     eth-LAN
  1 D 10.0.0.217/32     10.0.0.230     0.0.0.0        pppoe-in25
  2 D 10.0.0.217/32     10.0.0.231     0.0.0.0        pppoe-in26
[admin@MikroTik] ip arp> /ip route print
Flags: X - disabled, I - invalid, D - dynamic, J - rejected,
C - connect, S - static, R - rip, O - ospf, B - bgp
   #    DST-ADDRESS       G GATEWAY        DISTANCE INTERFACE
   0  S 0.0.0.0/0         r 10.0.0.1       1       eth-LAN
   1 DC 10.0.0.0/24       r 0.0.0.0        0       eth-LAN
   2 DC 10.0.0.230/32     r 0.0.0.0        0       pppoe-in25
   3 DC 10.0.0.231/32     r 0.0.0.0        0       pppoe-in26
[admin@MikroTik] ip arp>
```

5.4 ARP 绑定操作

虽然主机在 IP 网络中是通过 IP 地址通话，但实际上硬件地址（MAC 地址）被用于主机到其他主机的数据传输。地址解析协议 Address resolution protocol (ARP) 是提供硬件地址与 IP 地址之间的解析。每个路由器都一个 ARP 列表，记录 ARP 信息，由 IP 地址和相符合的 MAC 地址构成，一般 ARP 提供动态的 IP 与 MAC 地址对于关系，自动在 ARP 列表中产生。 路由器通过 ARP 列表的记录来回应各个主机的数据。我们也可通过静态的 ARP 记录，要求路由器只对静态的 ARP 做回应。这样就可以避免出现如有用户擅自修改 IP 地址或者通过 ARP 病毒影响路由路由器工作。如通过下面的设置：

1. 在 WinBox 中将动态 ARP 设置为静态的 ARP 记录。



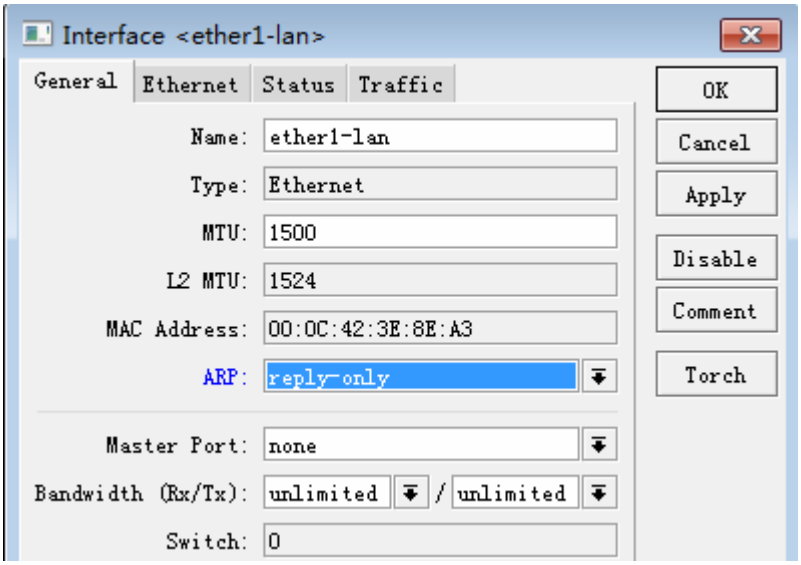
或者通过命令操作：

```
[admin@MikroTik] ip arp> add address=192.168.1.248 interface=ether1-lan
mac-address=00:21:00:56:00:12
```

同样的我们可以使用：

将所有的的 ARP 记录修改为静态的。

2. 设置 ether1-lan interface 仅回应静态 ARP 的请求，arp=reply-only：



命令操作如下：

```
[admin@RB230] > interface ethernet set ether2 arp=reply-only
```

ARP 双向绑定事例

首先将所有/ip arp 列表中的所有 LAN 口的 ARP 信息变为静态的，我们可以通过脚本做批处理的修改。注意，可能通过脚本命令不一定能将所有的内网的 ARP 参数修改完，可能需要手动添加。

```
:foreach i in [/ip arp find dynamic=yes interface=LAN] do={
  /ip arp add copy-from=$i}
```

然后设置 LAN 的网卡为：disabled

如果 ARP 功能在接口上被关闭，例如：使用 arp=disabled，来至客户端的 ARP 请求将不被路由器回应，因此必须添加静态的 ARP 才行。 例如，通过 arp 命令将路由器的 IP 和 MAC 地址必须添加到 windows 工作站中：

```
[admin@MikroTik] ip arp> /interface ethernet set LAN arp=disabled
```

现在路由器已经绑定了内网主机的所有 IP 地址后，现在需要对 Windows 电脑做对路由器绑定的设置

```
C:\> arp -s 10.5.8.254  00-aa-00-62-c6-09
```

也可以编辑 windows 自己的批处理文件（.dat）操作

第六章 路由设置（Route）

下面的内容介绍了 RouterOS 路由管理，针对目标、源地址和策略路由等，在各种网络环境具体使用哪一种路由方式，需要根据用户来选择。

需要功能包: ***system***
软件等级: *Level1*
操作路径: ***/ip route***, ***/ip route rules***

6.1 RouterOS 路由介绍

RouterOS 支持各种路由规则和策略，在实际环境中可以选择调用适合自己网络的规则，如电信网通多线接入、端口数据的路由指定和多线路绑定的负载均衡。

- **支持源 IP 地址的策略路由**
- **支持目标 IP 地址策略路由**
- **支持网页等 TCP 或 UDP 端口策略路由**
- **支持 Nth 和 PCC 负载均衡**
- **支持 IP 地址列表的策略路由**
- **各种策略都可以组合使用**

静态路由

RouterOS 的基本路由有下列类型的路由：

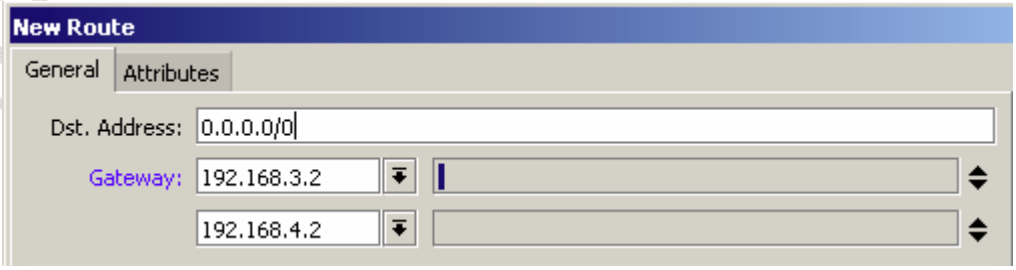
- **自动添加路由** 是当在一个网卡上添加了 IP，会自动创建一个动态的路由（如 PPPoE-Client、PPTP-Client 和 DHCP-Client 等自动添加网关）。
- **静态路由** 是用户自定义将 IP 数据包指定到默认网关的路由，这需要手动指定默认的网关。

当然在使用 PPPoE-Client、PPTP-Client 和 DHCP-Client 自动添加路由外，只能手动添加默认的路由规则，即默认路由，除非使用了动态路由协议（RIP 或 OSPF）

负载均衡路由

当使用同一类型网络，多线路接入时，可以采用负载均衡。早期的负载均衡功能称为“Equal-Cost Multi-Path Routing”，这种负载均衡缺点是每 10 分钟内核会重新均衡线路，这样一些链接会被指定到其他出口，出现频繁掉线情况。所以“Equal-Cost Multi-Path Routing”已被淘汰。

- Equal-Cost Multi-Path Routing 的操作，通过在 ip route 添加多网关的静态路由（格式如：**gateway=x.x.x.x,y.y.y.y**）路由协议会建立动态的多路路由。



之后接替的是采用第 N 次链接的负载均衡，这样基本实现了不掉线的真正负载均衡能，但存在一个弊端就是要求对 IP 严重的网站会反复要求验证，这样需要通过策略指定一些 IP 或者端口走固定的线路。

最后是 PCC（Per connection classified）每次连接分类的负载均衡，这样的负载均衡对每次的链接进行分类大多保持连续性的负载均衡，弥补了 Nth 的不足。

基于策略的路由

在策略路由在 RouterOS 中配置多条线路，如我们可以指定源地址策略路由，目标地址策略路由、端口策略路由、地址列表的策略路由，也可以通过多种方式组合使用：

- 标记期望的数据（源地址、目标地址和端口），设置一个 **routing-mark**
- 在 **ip route** 或 **ip route rules** 中配置目标和各种路由协议
- 使用 **address-list** 定义地址列表，并进行 routing-mark 标记

注意：这里你能重复一些网关路由的不同类型的参数，多次设置到一个网关上。

静态路由的简单配置

操作路径: ***/ip route***

在一个路由器两个 IP 段中，添加内网静态路由到网络 10.1.12.0/24 和默认网关出口路由 0.0.0.0/0 ：

```
[admin@MikroTik] ip route> add dst-address=10.1.12.0/24 gateway=192.168.0.253
[admin@MikroTik] ip route> add gateway=10.5.8.1
[admin@MikroTik] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf
 #    DST-ADDRESS       G GATEWAY        DISTANCE INTERFACE
 0 A S 10.1.12.0/24      r 192.168.0.253          Local
 1 ADC 10.5.8.0/24                               Public
 2 ADC 192.168.0.0/24                            Local
 3 A S 0.0.0.0/0         r 10.5.8.1               Public
[admin@MikroTik] ip route>
```

6.2 电信与网通目标地址路由

电信网通是现在比较常见的双线方式，通过路由指定分别让内网主机访问走电信和网通线路。该双线中，我们需要选择一条线路为主线，即默认路由出口，比如电信为主线，缺省网关设置为电信网关地址；网通线路需要通过导入路由表（电信和网通的路由表脚本在 www.mikrotik.com.cn 的网站上可以下载到 ）

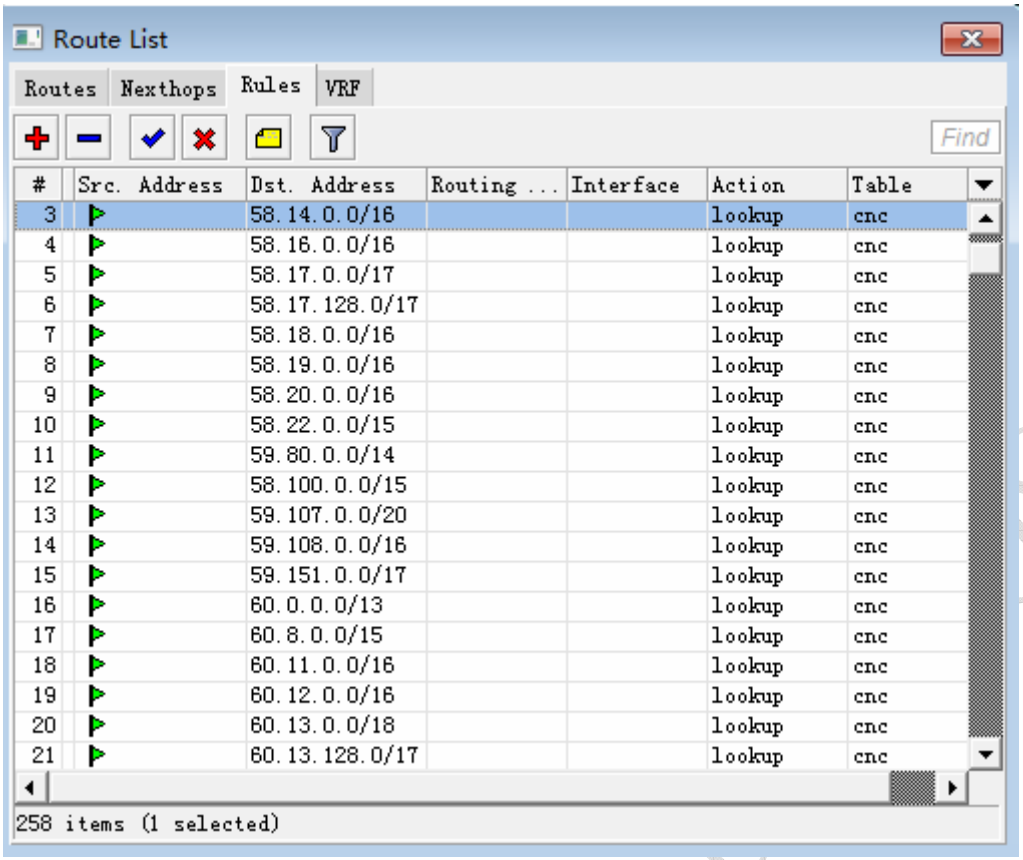
上传电信或网通路由脚本后，在根目录下使用 import 命令导入：

```
[admin@MikroTik] > import cnc1.rsc
```

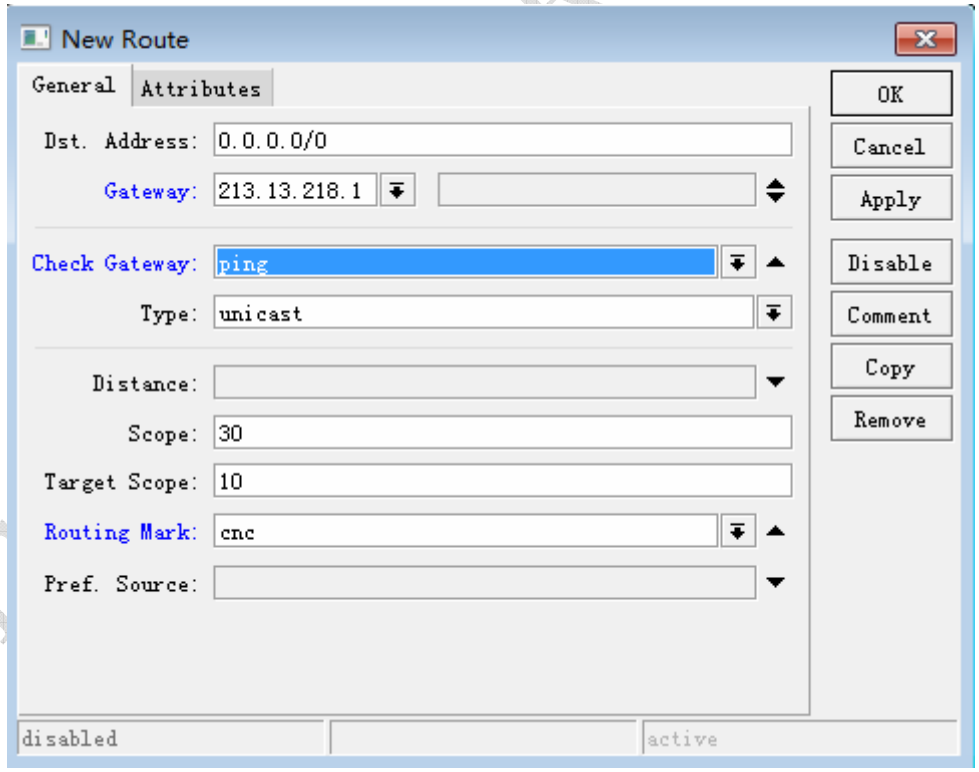
设置路由规则时命令如下：

```
/ip route add gatewall="对应网关地址" check-gateway=ping routing-mark=telecom 或者 cnc
```

如导入的网通线路标记为 cnc，我们可以在 ip route rules 里找到：



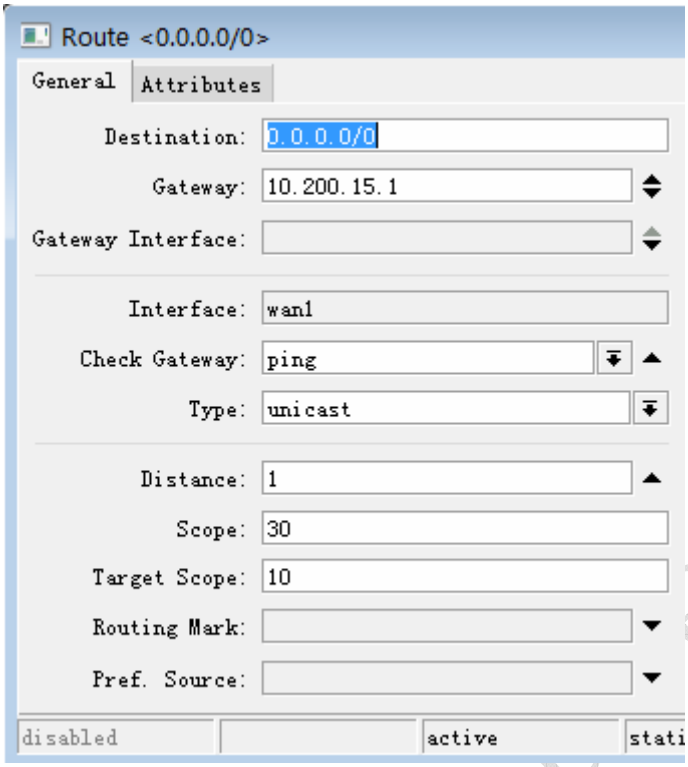
之后我们在 ip route 中 routing-mark 选择对应的 cnc 路由表



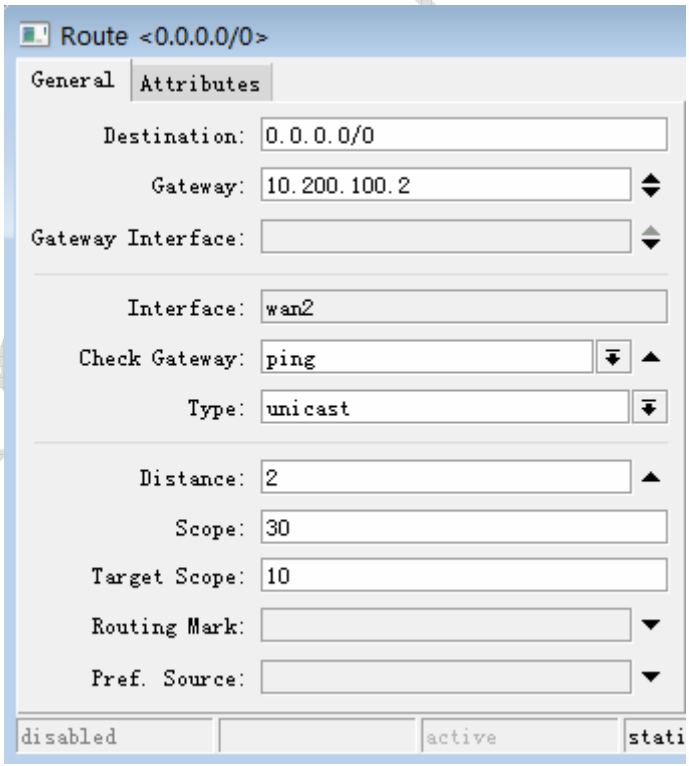
6.3 网关断线处理

在多线路情况下，我们可以通过配置备份线路，避免默认网关异常断开后，其他线路进行备份，即配置默认网关和备份网关，我们通过定义 **distance（路由距离）**对多个网关进行备份，根据 distance 来判断 1 为最优先，2 其次，依次类推。

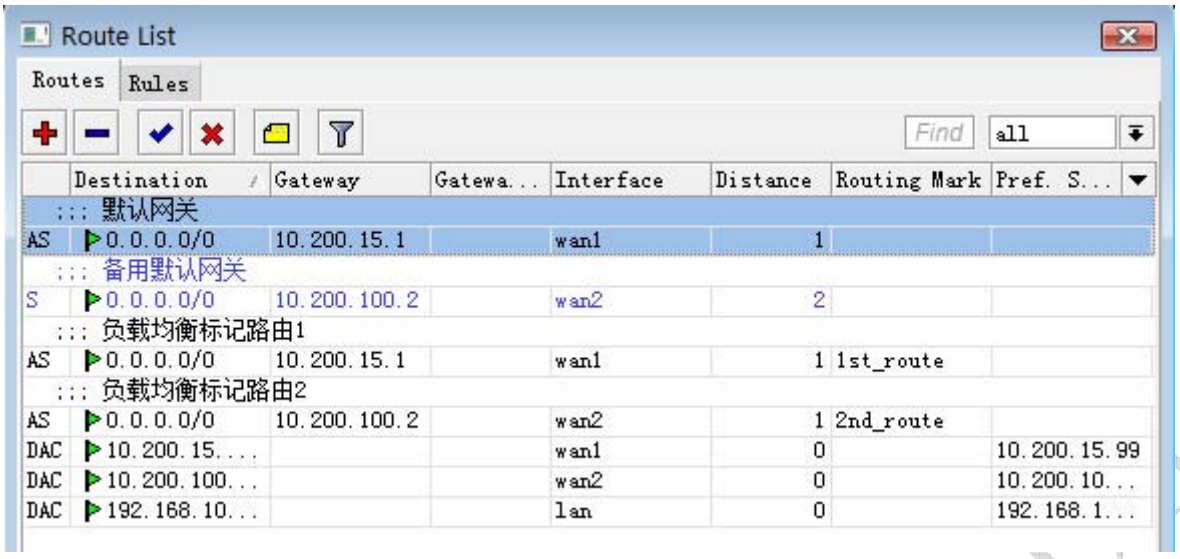
如下图：默认网关的 **distance** 设置为 1，并设置 check-gateway=ping，通过 ping 监测网关状态：



备份网关的 **distance** 设置为 2，并设置 check-gateway=ping，通过 ping 监测网关状态：



配置完成后的路由标如下图：



6.4 源地址策略路由奇偶标记

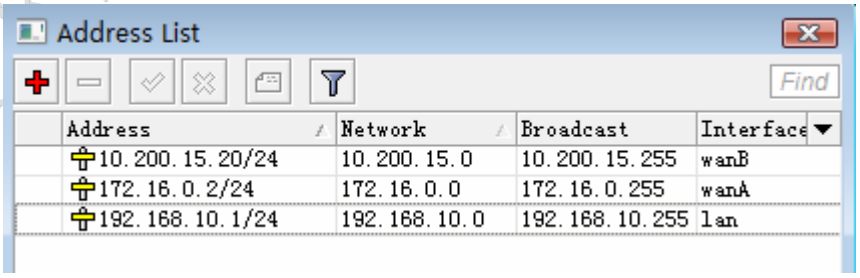
在双线策略路由器情况下，早期的策略路由是通过标记一段地址走一条线路，如我们标记 192.168.10.2-192.168.10.127 走线路 A，剩下的 IP 地址则走线路 B，这样的策略路由在一定的情况下出现效率不高的问题，如当用户 IP 地址是顺序增加，但没有到 127 的时候。线路 B 就不会起到流量分配的作用。

为了解决这样的问题，我们通过 RouterOS 的 address-list 建立一个地址列表，分别将奇数和偶数的 IP 地址分开，即奇数的 IP 地址走线路 A，而偶数的 IP 地址走线路 B，这样的策略路由便提高了双线的使用效率。

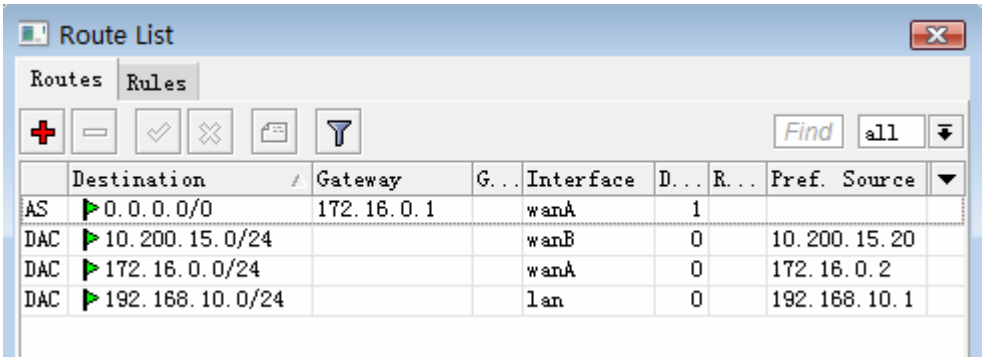
操作步骤如下：

1、 配置好网络的 IP 地址和路由；
2、 在 ip firewall address-list 列表中建立奇数或者偶数地址列表；
3、 进入 ip firewall mangle 通过 src-address-list 标记数据包；
4、 在 ip route 中调用标记好的地址策略，配置路由。

步骤 1：我们首先进入路由器配置 IP 地址，假设我们有两条线路，分别是 A 和 B，A 的 IP 地址是 172.16.0.2，网关是 172.16.0.1；B 的 IP 地址是：10.200.15.20，网关是 10.200.15.1。

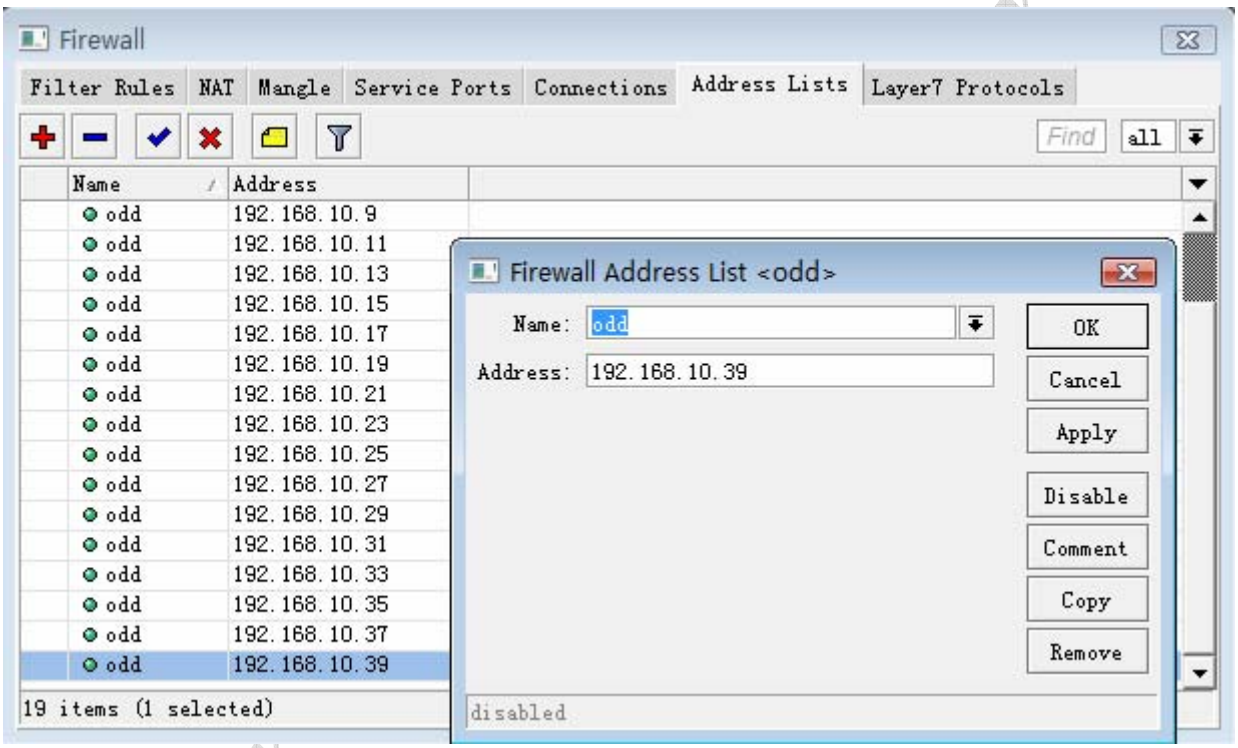


在 ip route 中配置以线路 A 的网关 172.16.0.1 为默认路由：



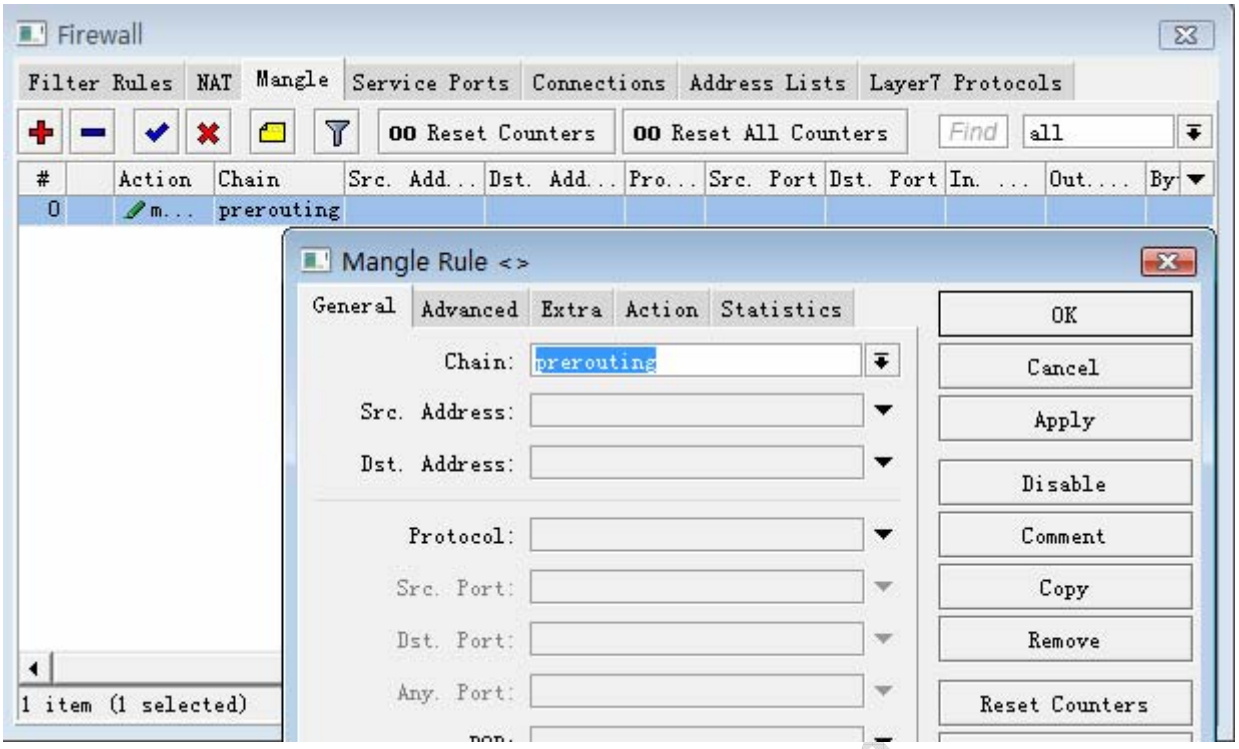
步骤 2：配置好 IP 地址和路由后，接下在 ip firewall address-list 中添加奇数的地址列表，因为是双线路由，我们只需要配置一条奇数的列表，而偶数的列表可以不用配置，因为奇数被标记后，剩下的就为偶数地址。

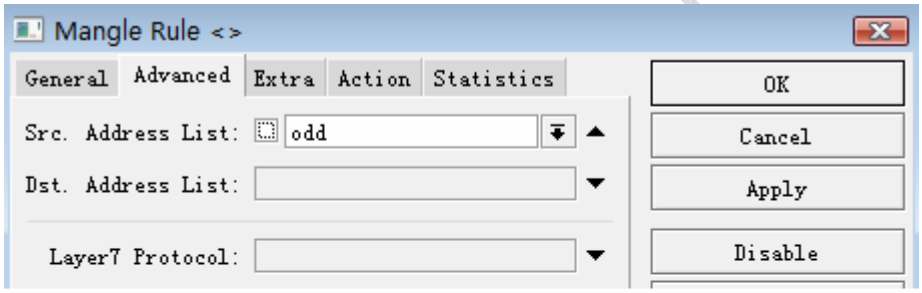
我们将奇数列表地址取名为 odd，并向地址列表里面添加你网络内所有的奇数的 IP 地址：

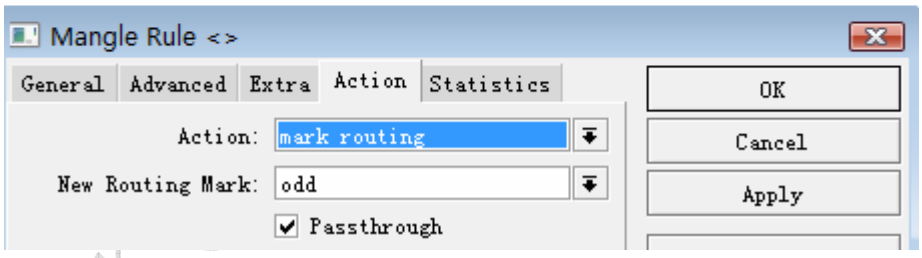


步骤 3：当奇数 IP 地址添加完成后，我们进入 ip firewall mangle 标记路由规则，我们选择 chain=prerouting：

```
[admin@CDNAT] /ip firewall mangle> add chain=prerouting action=mark-routing new
-routing-mark=odd src-address-list=odd
[admin@CDNAT] /ip firewall mangle> print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=prerouting action=mark-routing new-routing-mark=odd passthrough=yes
     src-address-list=odd
```

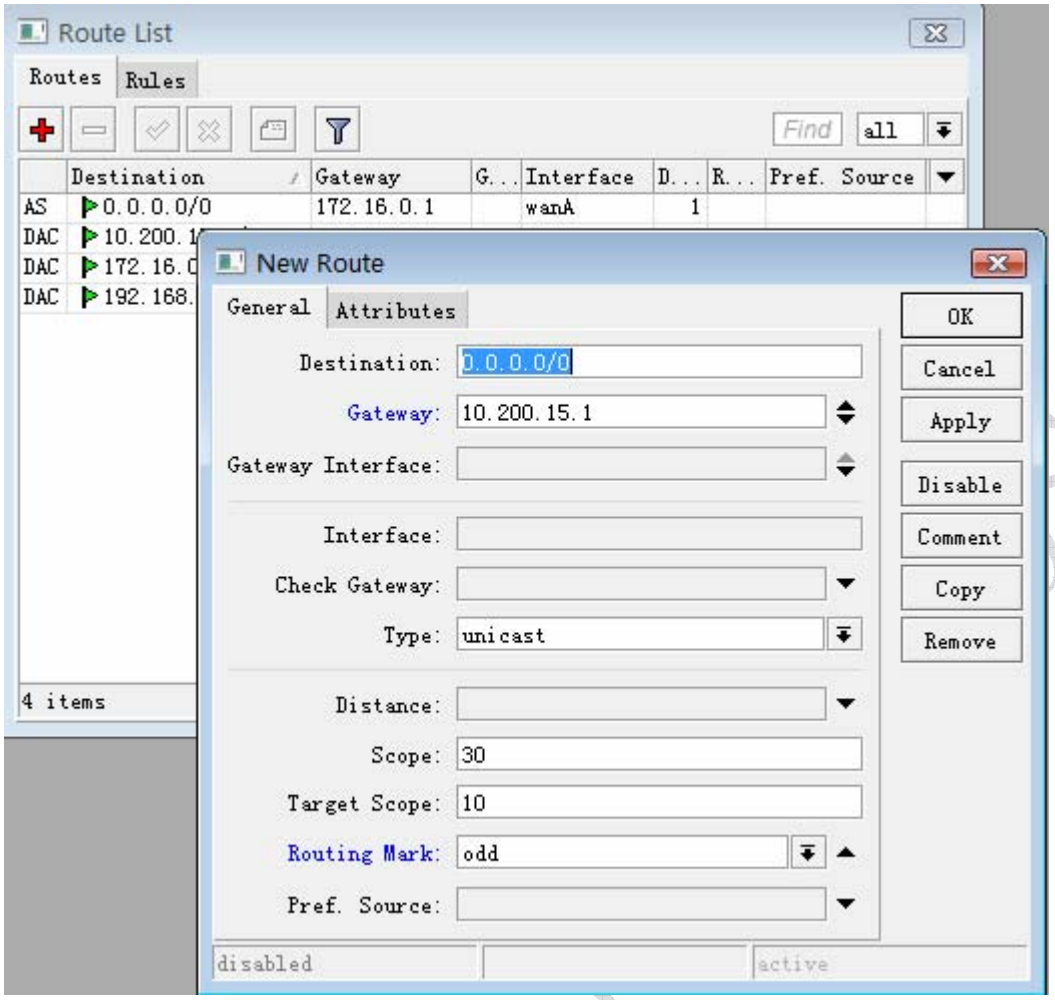




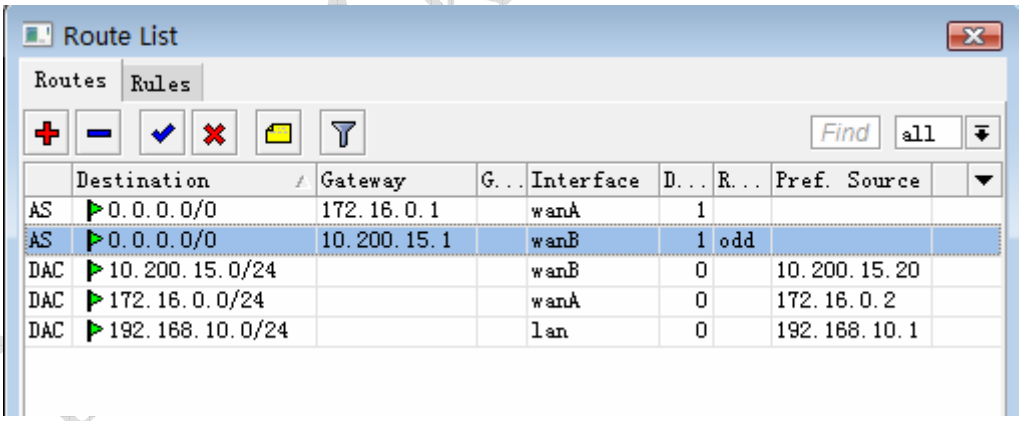


因为只标记了奇数的 ip 地址，剩下的便是偶数的，所以不用再做配置。

步骤 4：配置完标记后，我们进入 ip route 配置路由，我们只需要将奇数的 IP 地址标记到线路 B 的网关即可，操作如下配置地址如下：



配置只需要添加 gateway=10.200.15.1 和 routing-mark=odd，点确认：



奇数的 IP 地址便从 B 线路的 10.200.15.1 的网关出去，剩下的偶数 IP 地址从默认的线路 A 的 172.16.0.1 的网关出去。

6.5 光纤和 ADSL 静态路由

基本情况：假设用户有两条 Internet 线路，一条是使用固定地址的网通光纤 2M，另一条是使用电信拨号的 ADSL 通用为 2M。使用 NAT 伪装让局域网共享上网。在路由器上共有 3 块网卡，WAN1 用于网通光纤，WAN2 用于 ADSL 拨号，LAN 用于连接内网终端。

首先我们设置 WAN1 与 WAN2 的 IP 地址：ADSL 拨号大致如下：具体参考 PPPoE 设置说明

```
配置 ADSL 线路
/interface pppoe-client 配置 ADSL 拨号信息。
/interface pppoe-client add name pppoe-line1 service CHN-Telecom/ user 以 c999@166 password
123 interface WAN2 use-peer-dns yes mtu 1942 mru 1942
```

注: 设置 pppoe-client 时当得到 ADSL 默认网关后，将 pppoe-client 中的 add-default-route=yes，修改为 **add-default-route=no** 避免自动添加默认的电信路由。

```
[admin@MikroTik] ip address> add address 61.193.77.77/24 interface WAN1
 [admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK        BROADCAST      INTERFACE
  0   61.193.77.77/24   61.193.77.0    61.193.77.255    WAN1
D 1   218.88.32.10/24   218.88.32.1    0.0.0.0          pppoe-out1
[admin@MikroTik] ip address>
```

下面配置内网地址为 192.168.0.1/24：

```
[admin@MikroTik] ip address> add address 192.168.0.1/24 interface LAN
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK        BROADCAST      INTERFACE
  0   61.193.77.77/24   61.193.77.0    61.193.77.255    WAN1
D 1   218.88.32.10/24   218.88.32.1    0.0.0.0          pppoe-out1
  2   192.168.0.1/24    192.168.0.0    192.168.0.255    LAN
[admin@MikroTik] ip address>
```

下面我们需要配置一个默认网关，在这里我们以网通的 61.193.77.1 网关为默认网关，电信的作为静态路由：

```
[admin@MikroTik] ip route> add gateway=61.193.77.1
[admin@MikroTik] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf
 #     DST-ADDRESS       PREFSRC        G GATEWAY        DISTANCE INTERFACE
 0 ADC 61.193.77.0/24    61.193.77.77                              WAN1
1 ADC 218.88.32.1/32    218.88.32.10                             pppoe-out1
 2 ADC 192.168.0.0/24    192.168.0.1                               LAN
 3 A S 0.0.0.0/0                      r 61.193.77.1                WAN1
 [admin@MikroTik] ip route>
```

现在我们导入电信的静态路由表，电信和网通的路由表脚本在 www.mikrotik.com.cn 的网站上可以下载到，操作根据说明要求设置，网通电信双线路由脚本操作方式：

将你的正确的电信或网通的网关，使用用编辑-替换掉脚本里的“网关”，然后打开 winbox，点击 Terminal（控制终端）然后复制脚本，并在 Terminal（控制终端）中点右键选择“paste”粘贴脚本，粘贴完后敲回车，也可以生产.rsc 的文件上传到路由器的 files 根目录下，通过 import 命了完成操作。

这里我们将电信的网关 218.88.32.1 在“电信 IP 脚本”文本文件中使用替换操作将所有含“网关”的关键字替换为 218.88.32.1，然后复制并在 Terminal 控制台中粘贴脚本。这样电信脚本即可导入。

```
[hcf@NAT] ip route> prin
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf
 #     DST-ADDRESS       PREFSRC        G GATEWAY        DIS    INTERFACE
0 ADC 61.193.77.0/24    61.193.77.77                             WAN1
1 ADC 218.88.32.1/32    218.88.32.10                          pppoe-out1
2 ADC 192.168.0.0/24    192.168.0.1                              LAN
3 A S 0.0.0.0/0                      r 61.193.77.1              WAN1
4 A S 218.4.0.0/15                    r 218.88.32.1          pppoe-out1
5 A S 218.6.0.0/16                    r 218.88.32.1          pppoe-out1
6 A S 218.13.0.0/16                   r 218.88.32.1          pppoe-out1
7 A S 218.14.0.0/15                   r 218.88.32.1          pppoe-out1
8 A S 218.16.0.0/14                   r 218.88.32.1          pppoe-out1
9 A S 218.20.0.0/16                   r 218.88.32.1          pppoe-out1
10 A S 218.21.0.0/17                  r 218.88.32.1          pppoe-out1
11 A S 218.22.0.0/15                  r 218.88.32.1          pppoe-out1
12 A S 218.30.0.0/15                  r 218.88.32.1          pppoe-out1
13 A S 218.62.128.0/17                r 218.88.32.1          pppoe-out1
14 A S 218.63.0.0/16                  r 218.88.32.1          pppoe-out1
15 A S 218.64.0.0/15                  r 218.88.32.1          pppoe-out1
16 A S 218.66.0.0/16                  r 218.88.32.1          pppoe-out1
.....
```

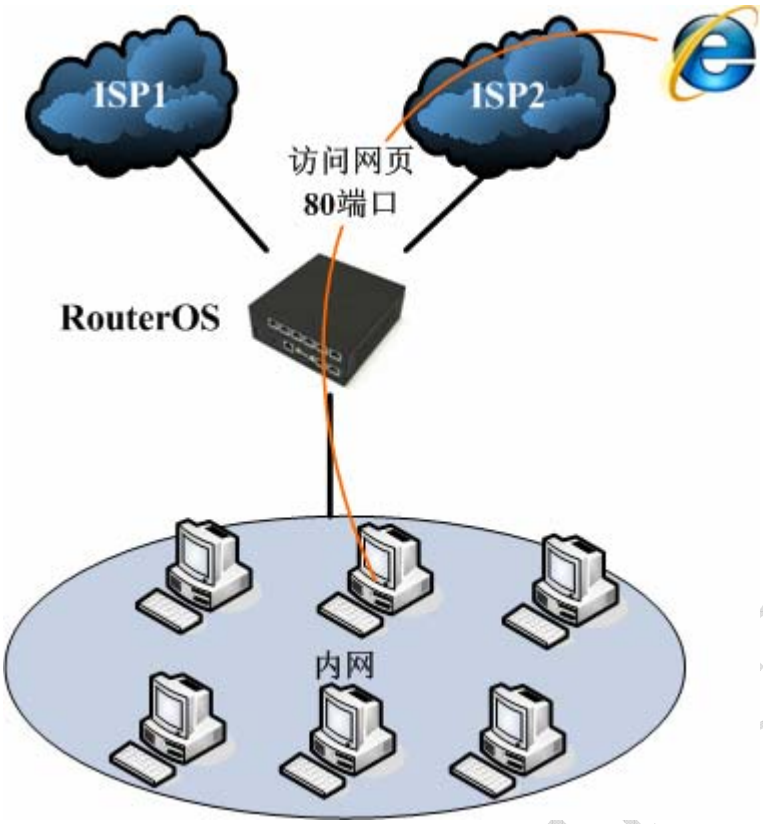
为保证一条线路断线时，到其他目标地址能正常连接，在/tool netwatch 中设置主机网关监控（具体设置参考 Network 监控），并配置脚本编译。

如果当你使用静态路由指定网通或电信线路的时候，其中一条线路出现故障，需要切换到另外一条线路时我们需要设置以下脚本,如电信的线路出现故障，需要禁用掉电信网关的静态路由策略，让所有的数据走默认的网通线路，电信网关为：222.212.48.1。脚本设置如下

```
当电信线路出现故障的时候，禁用掉所有到电信网关的策略
:foreach i in=[/ip route find gateway=218.88.32.1] do={/ip rout disable  $i}
当电信线路正常后，启用所有电信策略
:foreach i in=[/ip route find gateway=218.88.32.1] do={/ip rout enable $i}
```

6.6 http 端口的策略路由

MikroTik RouterOS 可以支持多种策略路由，如我们常见的源地址、目标地址，同样支持端口的策略路由，多种规则可以根据用户情况配合使用，如下图：

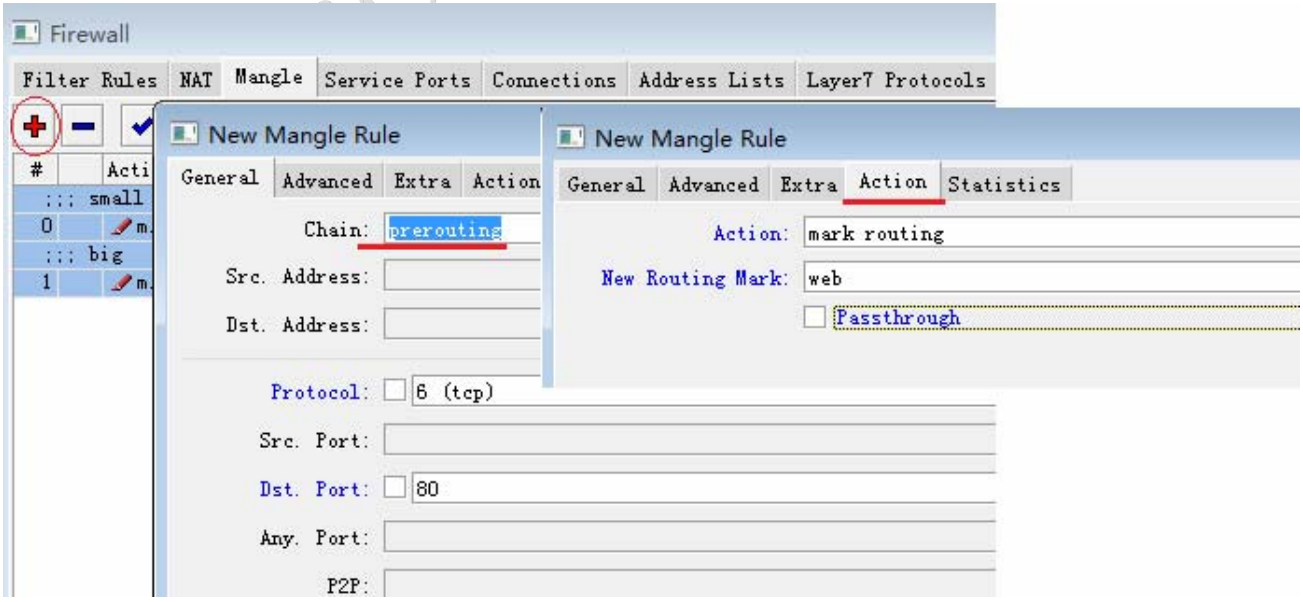


网络情况：

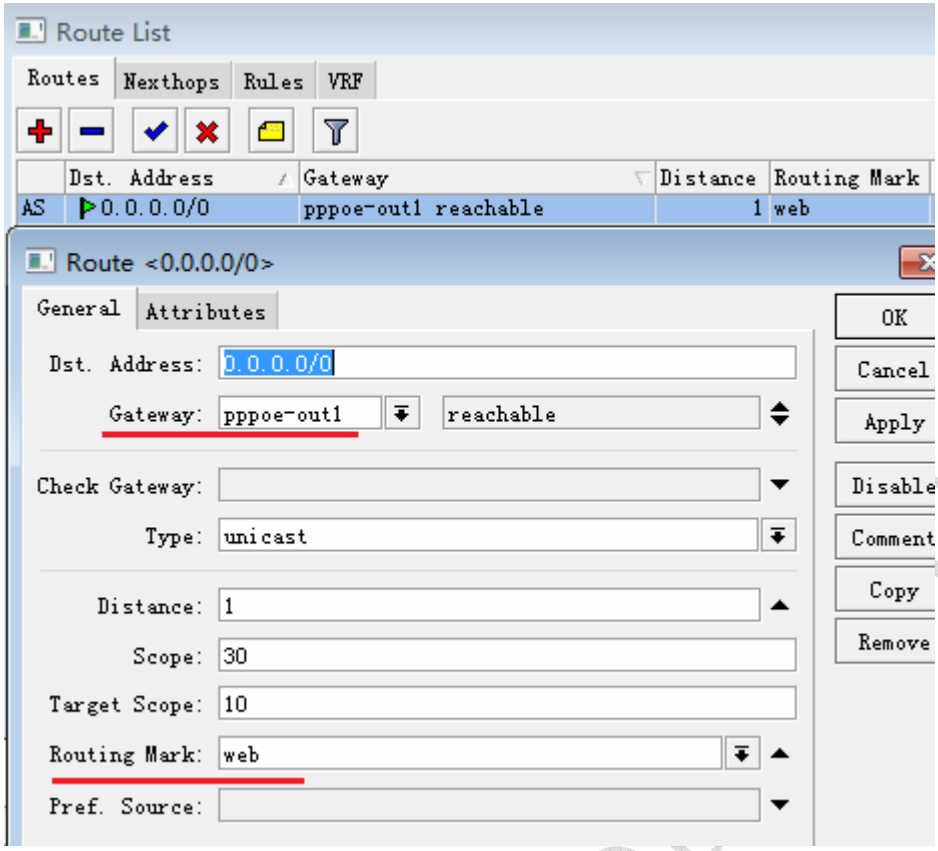
我们有两个 ISP 接入的线路，一个是 ISP1 通过光纤接入，另外一个 ISP2 的 PPPoE 拨号链接，我们需要通过将访问网页的数据都转移到 PPPoE 拨号上，其他的的数据默认走 ISP1 的光纤。

实际配置

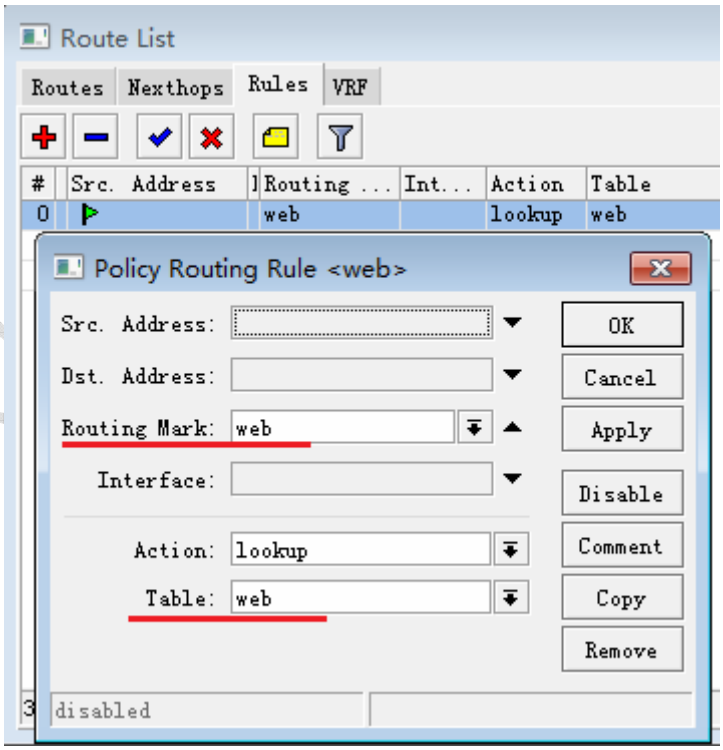
现在我们定义访问网页的端口，访问网页的端口是 TCP 80 端口，我们进入/ip firewall mangle 中做数据标记，从标记中提取路由标记，命名为“web”，因为我们在前面的连接标记中做过了 passthrough 的设置，在这里就不用在重复设置。



然后我们进入/ip route，配置路由我们让标记好的 80 端口通过 pppoe-out1 出去：



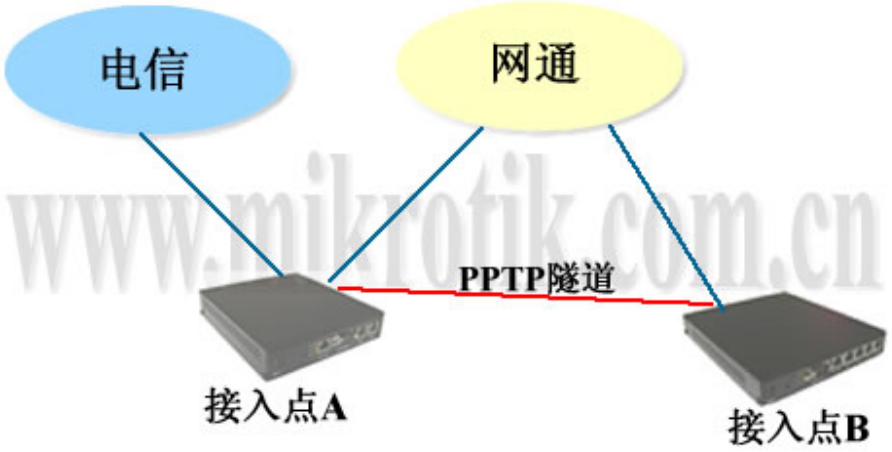
在这里，如果在 ip route rule 里有其他的策略规则出现，我们最好是在/ip route rule 里再次定义 80 端口的规则：



在 ip route rules 定义的 web 标记在 web 路由表中去查找路由。

6.7 PPTP 借线路由操作

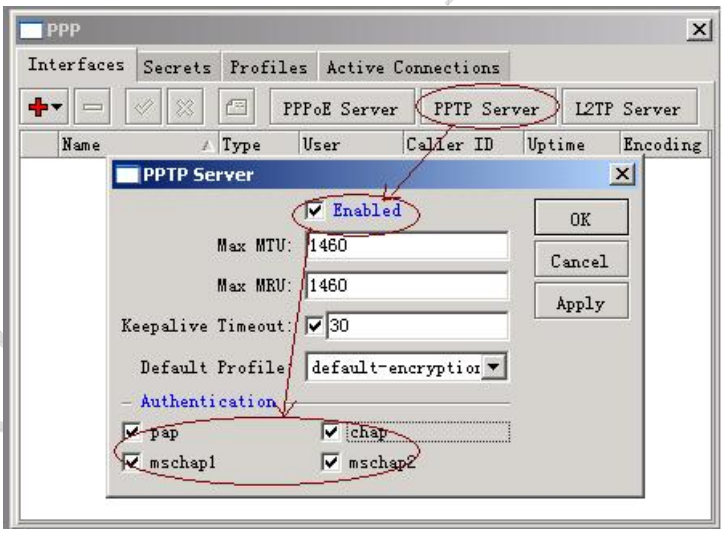
假设一个接入点 A 有电信和网通两条线路，并做了以网通为主，电信为静态路由策略设置。而另一个接入点 B 接入了网通的线路，并且想通过 PPTP 隧道的方式借用接入点 A 的电信线路，现在看下面的图例



根据上面的案例，接入点 A 和 B 他们都是共同使用了网通的线路，这里网通两个点之间的延迟小于 10ms，网络延迟小才能保证足够的网速给 B 做电信的访问。首先建立从接入点 B 到 A 的 PPTP 隧道，我们在接入点 A 设置 PPTP 服务器，在接入点 B 设置客户端。这里接入点 A 的网通 IP 地址为 202.112.12.10，B 网通地址为 202.112.12.12。

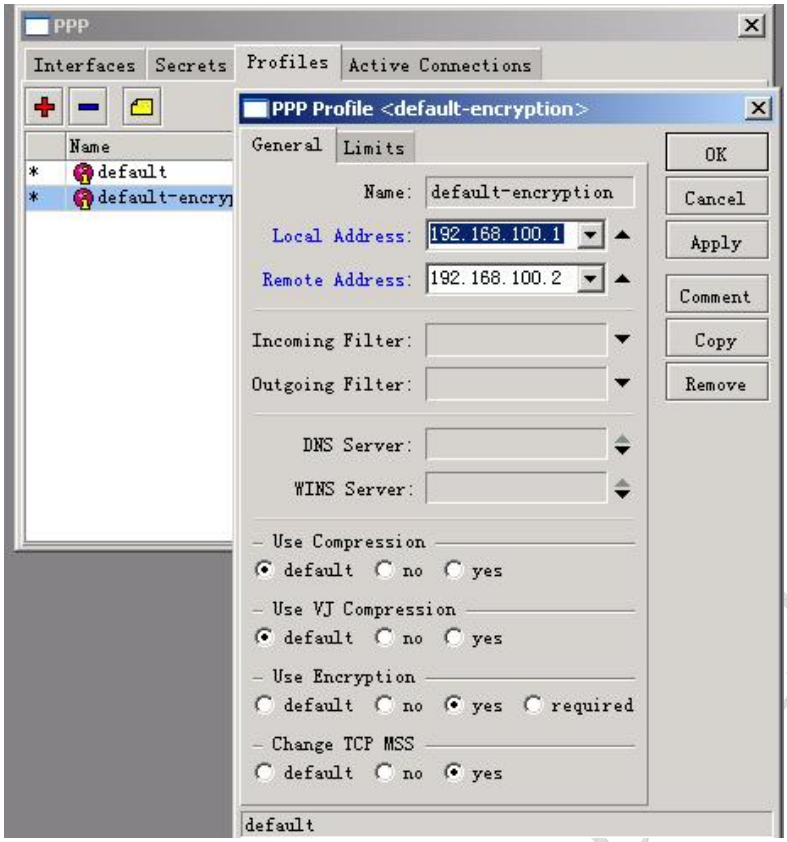
配置 PPPTP-Server

在接入点 A 启用 PPTP-Server，并设置密码传输的加密类型：



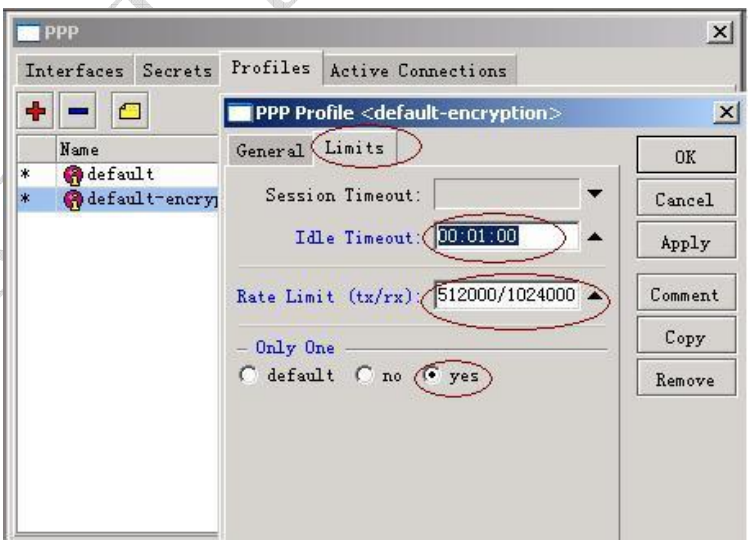
在这里 Default-Profile 我们采用 default-encryption，同样你也可以在 PPTP-Server 的 profiles 中创建自己的规则。Keepalive-Timeout 是 PPTP-Server 主动使用 ICMP 协议探测客户端是否在线，如果客户端使用了防火墙或禁止 ICMP 探测，那无法探测到客户端，Server 就会主动断开该客户端的连接，这个设置需要用户自己根据网络情况判断。

设置 Profile 定义客户和主机的访问地址：



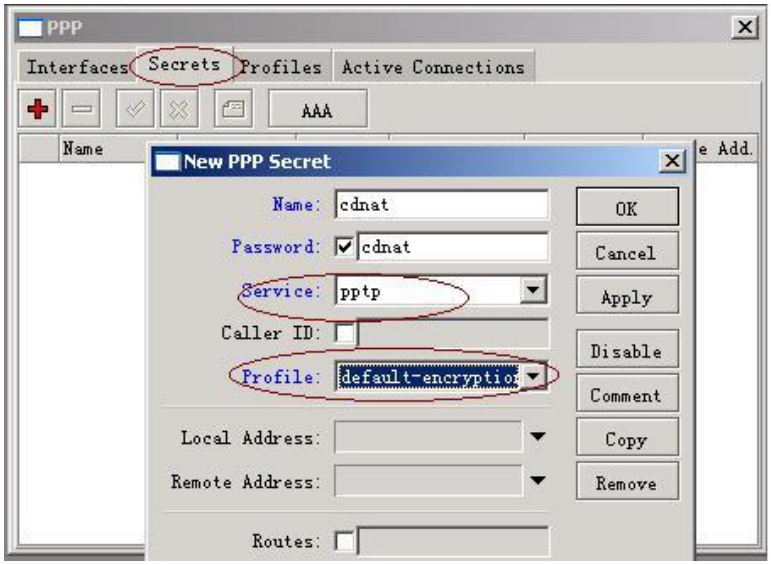
在这里我们给 PPTP-Server 分配的 IP 地址为 192.168.100.1(local-address) ，给客户端分配的地址为 192.168.100.2(remote-address) 。分配 IP 地址也可以通过账号设置 Secrets 进行，在这里我们只有一个客户端所有可以直接通过 profile 中的规则设置，如果有多个客户端也可以通过/ip pool 中的地址池做 DHCP 的分配。

配置 limit 参数：



在 limit 参数中，我们可以看到 idle-timeout，这个是客户端在没有流量超过 1 分钟后，就断开客户端。Rate-limit 是对该类用户的流量控制这里设置的上行为 512K，下行 1M 的带宽。最后是 only-one 该账户是否为唯一，这里设置为 yes。

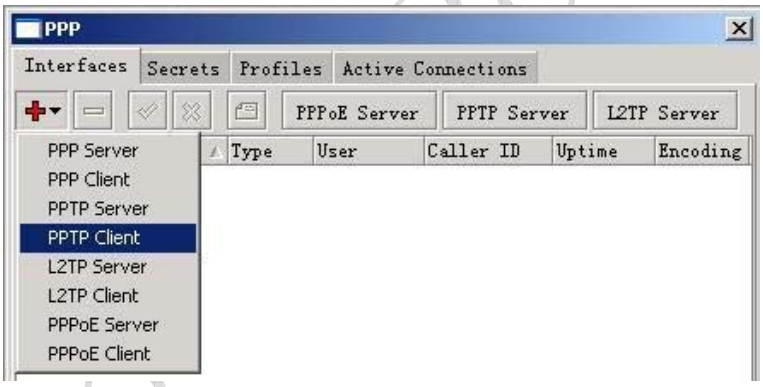
设置客户端的账号密码：



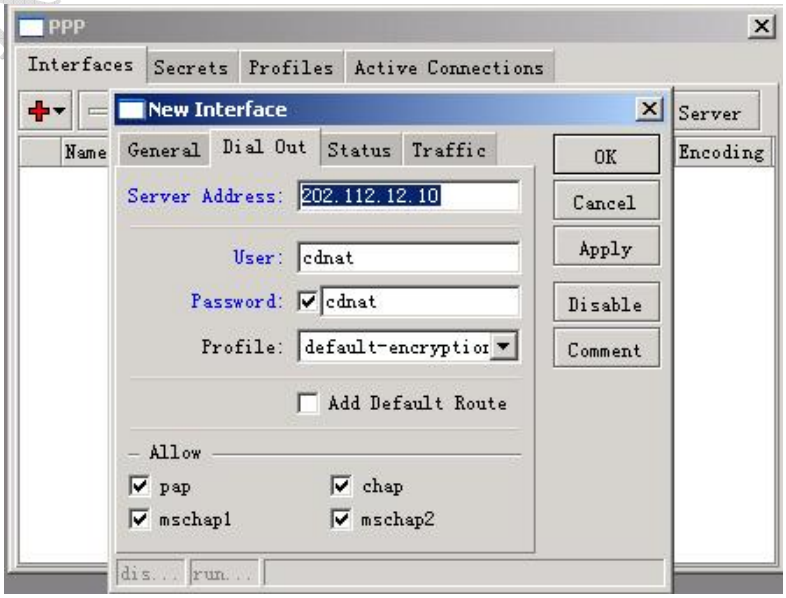
进入 secret 设置账号和密码以及相关信息，设置好 name 和 password 后，选择 service 服务类型为 pptp，profile 规则为 default-encryption。这样 PPTP-Server 就已经设置完成。

配置 PPTP-Client

完成 PPTP 服务设置后，现在开始设置接入点 B 的 PPTP-Client，进入 PPP 选项添加 PPTP-Client：



进入 dial-out 设置 PPTP 拨号信息，在 server-address 的地址为 202.112.12.10 级接入点 A 的网通地址：

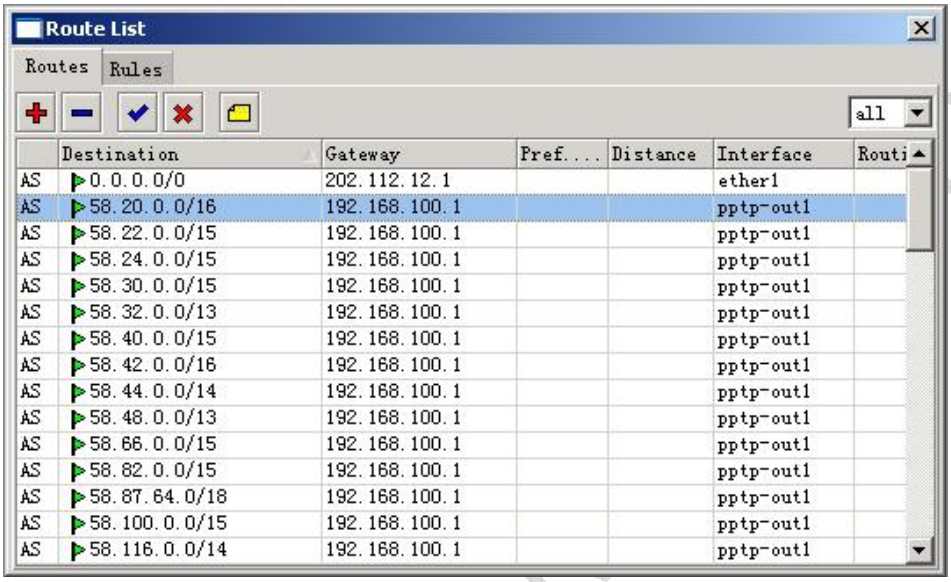


设置账号和密码分别为 cdnat，设置完成后，便可以与接入点 A 的 PPTP-Server 连接。

路由配置

在这里接点 A 和 B 都做了 IP 地址的 NAT 转换，且接点 A 已经做了电信的静态路由规则，即 A 点可以实现访问网通和电信的分流，在 A 点不需要在做任何设置。B 点就需要指定通过 AB 两点间的 PPTP 隧道到电信的线路，他指定的网关为 A 点的 PPTP 的 IP 地址（192.168.100.1）

设置电信访问的网关：



	Destination	Gateway	Pref....	Distance	Interface	Routi
AS	0.0.0.0/0	202.112.12.1			ether1	
AS	58.20.0.0/16	192.168.100.1			pptp-out1	
AS	58.22.0.0/15	192.168.100.1			pptp-out1	
AS	58.24.0.0/15	192.168.100.1			pptp-out1	
AS	58.30.0.0/15	192.168.100.1			pptp-out1	
AS	58.32.0.0/13	192.168.100.1			pptp-out1	
AS	58.40.0.0/15	192.168.100.1			pptp-out1	
AS	58.42.0.0/16	192.168.100.1			pptp-out1	
AS	58.44.0.0/14	192.168.100.1			pptp-out1	
AS	58.48.0.0/13	192.168.100.1			pptp-out1	
AS	58.66.0.0/15	192.168.100.1			pptp-out1	
AS	58.82.0.0/15	192.168.100.1			pptp-out1	
AS	58.87.64.0/18	192.168.100.1			pptp-out1	
AS	58.100.0.0/15	192.168.100.1			pptp-out1	
AS	58.116.0.0/14	192.168.100.1			pptp-out1	

通过编辑电信的路由脚本，并导入路由表中，则实现了通过 PPTP 隧道使用 A 接入点的电信线路，完成了借线功能。

6.8 RouterOS 路由表操作原则

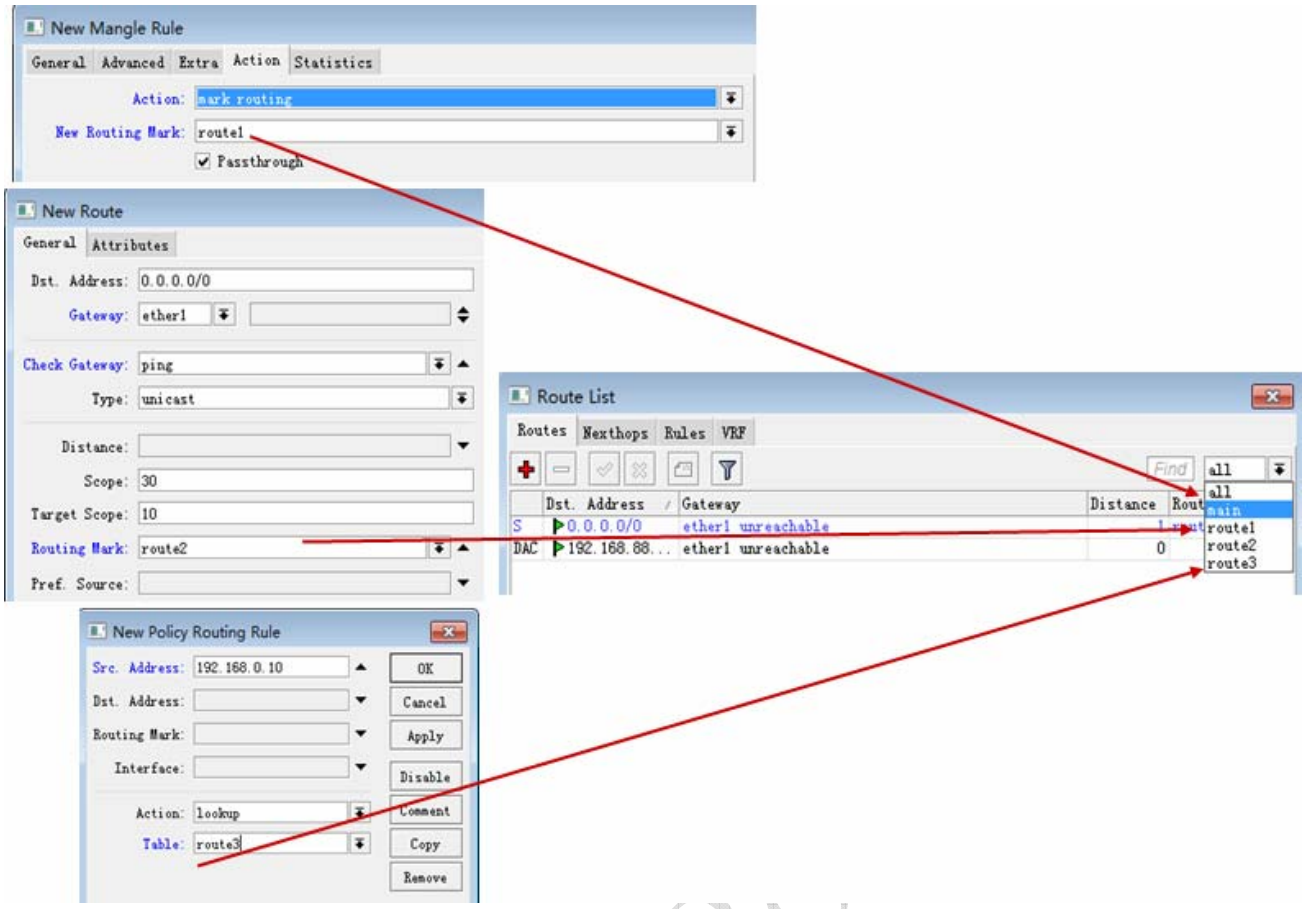
RouterOS 能维护多个独立的路由表，能灵活的分配策略路由规则，通过下面的操作命令可以标记路由与定义路由策略表

- /ip firewall mangle mark-routing （支持源目标和端口路由）
- /ip route routing-mark （支持源目标路由）
- /ip route rule table （支持源目标路由）

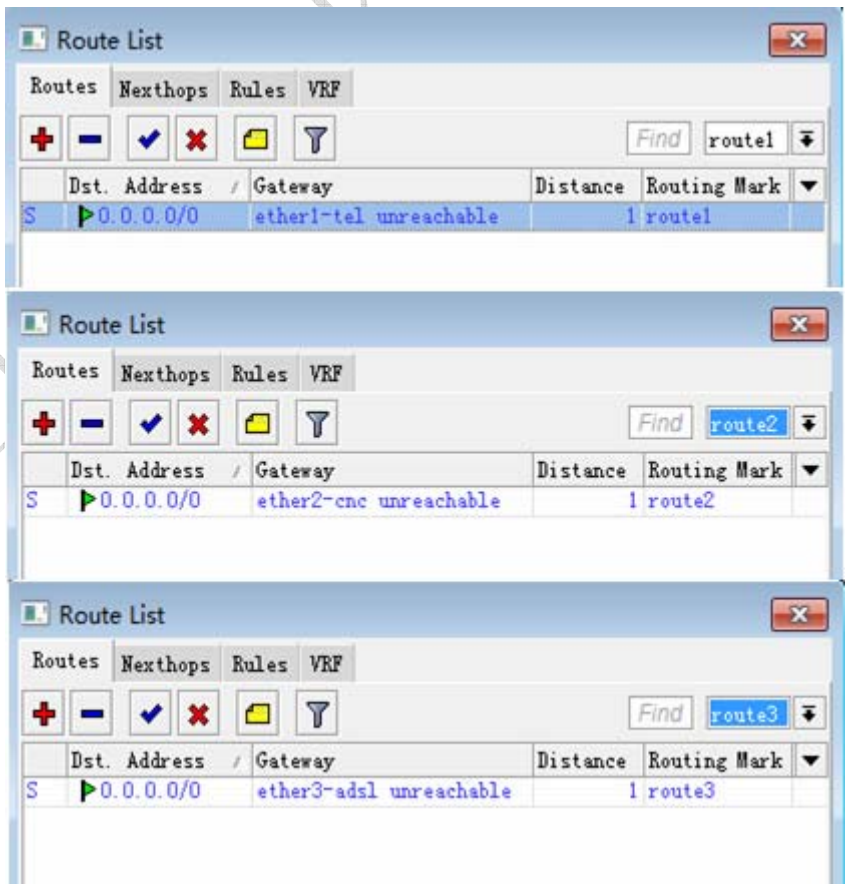
他们之间关系是平等的：

mark-routing = routing-mark = table

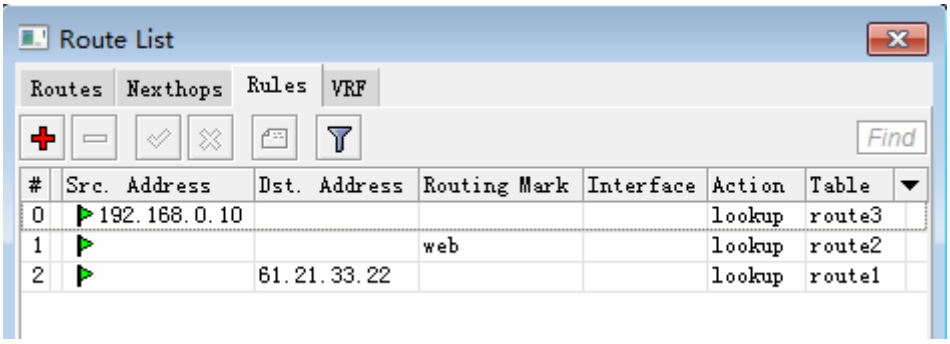
从下面的的操作图中可以看到，在 ip firewall mangle、routing-mark 和 table 中的建立的路由表，可以在 ip route 的右侧列表找到对应的路由表



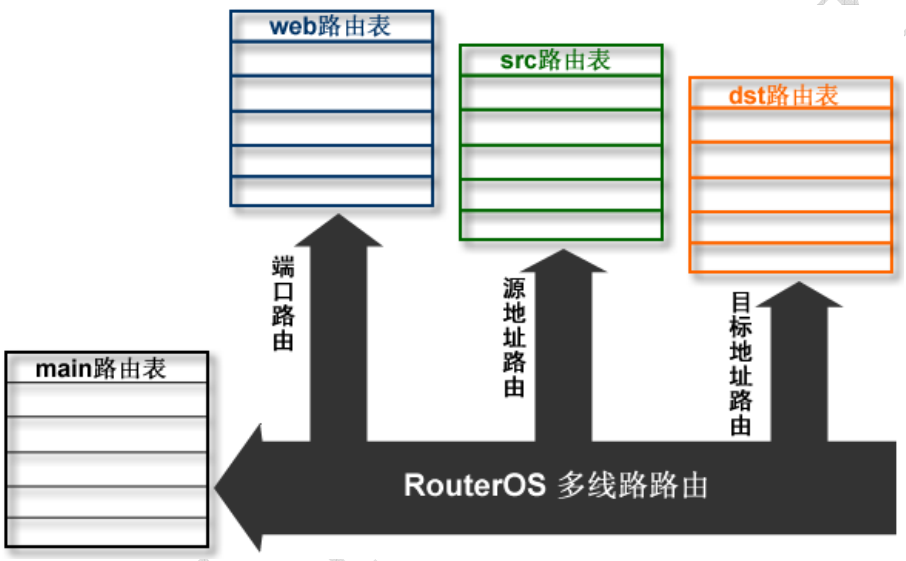
我们选择这些定好的路由表 route1、route2 和 route3 等，给他们定义各自的路由和网关，但他们并不是缺省的默认网关，而是建立在路由标记下的网关



我们可以通过 ip route rules 来分配源地址、目标地址和端口的策略路由，并通过 table 选择各自的路由表，如下图：



当他们被定义后都会在 ip route 中新建路由表，如图：



如果建立了多个路由表，RouterOS 会首先处理新建的路由表，最后剩下的数据到 Main 表，注意：在 ip route rule 中的规则是从上往下的执行，最上规则优先执行。

6.9 PCC 负载均衡

PCC 匹配器允许分离传输流做到平衡流量的功能(能指定这个属性选择 src-address, src-port, dst-address,dst-port)

PCC 原理

PCC 从一定范围内分析选择 IP 数据包头，通过哈西散列算法的帮助下，将选定的区域转换为 32bit 值。这个值除以指定 Denominator（分母），余数将比较一个指定的余数（Remainder），如果相等这时数据包将会被捕获，你可以选择 src-address, dst-address, src-port, dst-port 等使用此操作。

```
per-connection-classifier=
PerConnectionClassifier ::= [!]ValuesToHash:Denominator/Remainder
  Remainder ::= 0..4294967295    (integer number)
  Denominator ::= 1..4294967295    (integer number)
```

```
ValuesToHash ::= src-address|dst-address|src-port|dst-port[,ValuesToHash*]
```

per-connection-classifier 分类器，通过判断源地址、目标地址、源端口和目标端口，对数据进行分类，如

事例：这个配置将所有连接基于源地址和端口分类的 3 个组：

```
/ip firewall mangle add chain=prerouting action=mark-connection
new-connection-mark=1st_conn per-connection-classifier=both-addresses:3/0
/ip firewall mangle add chain=prerouting action=mark-connection
new-connection-mark=2nd_conn per-connection-classifier=both-addresses:3/1
/ip firewall mangle add chain=prerouting action=mark-connection
new-connection-mark=3rd_conn per-connection-classifier=both-addresses:3/2
```

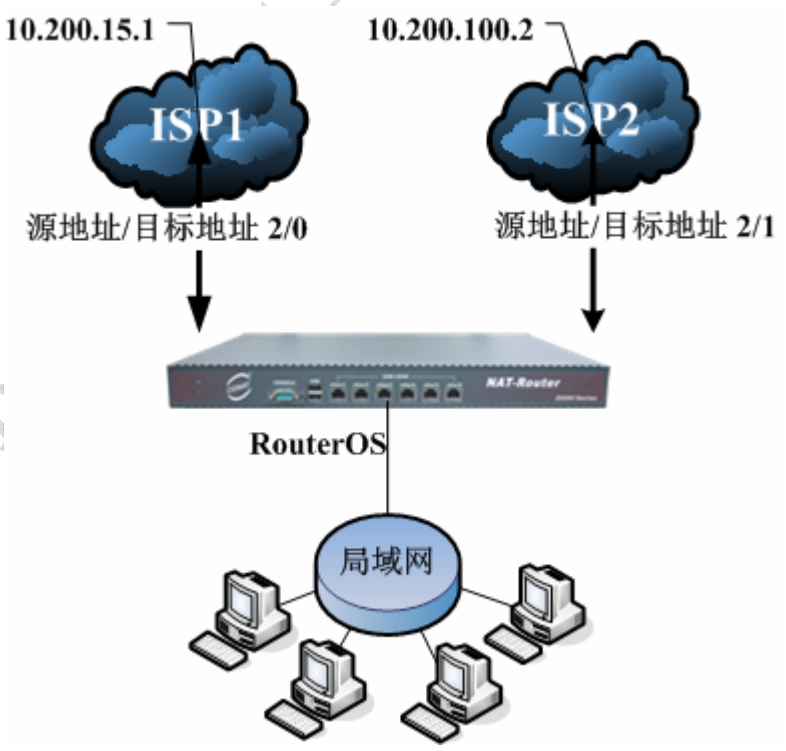
per-connection-classifier=both-addresses:3/0，这条规则的含义为我们对原地址的端口进行分类，3/0 为一共有 3 条出口，定义第一条，3/1 则是第二条，以此类推。

注意：PCC 从 RouterOS v3.24 开始支持，这个功能解决了多网关的负载均衡问题。

PCC 的负载均衡事例

一、 双向地址负载均衡

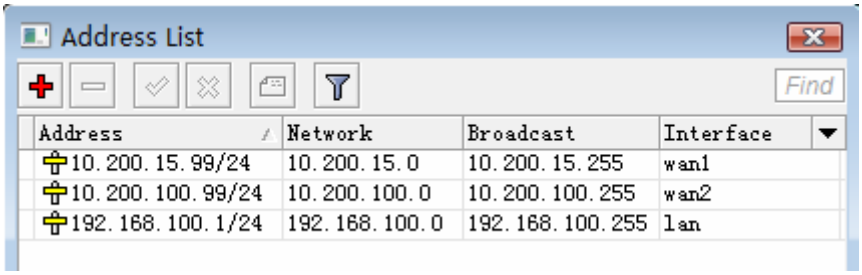
通分组源地址和源端口实现负载平衡，这里我们建立 2 个 WAN 出口分别是 wan1 和 wan2，网络环境如下：



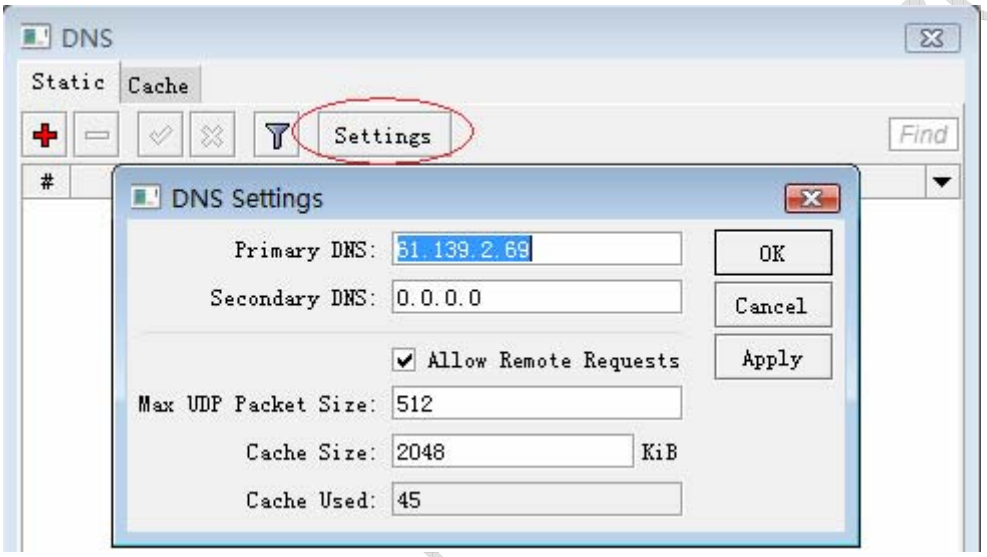
- ISP1 地址 10.200.15.99/24，网关：10.200.15.1；
- ISP2 地址 10.200.100.99/24，网关：10.200.100.2；
- 内网 IP 地址 192.168.100.1/24；
- 启用 DNS 缓存功能，用 192.168.100.1 作内网 DNS 解析；

基本配置

首先进入 ip address 配置 IP 地址：



在 ip dns setting 中配置好 DNS 缓存，DNS 为：61.139.2.69

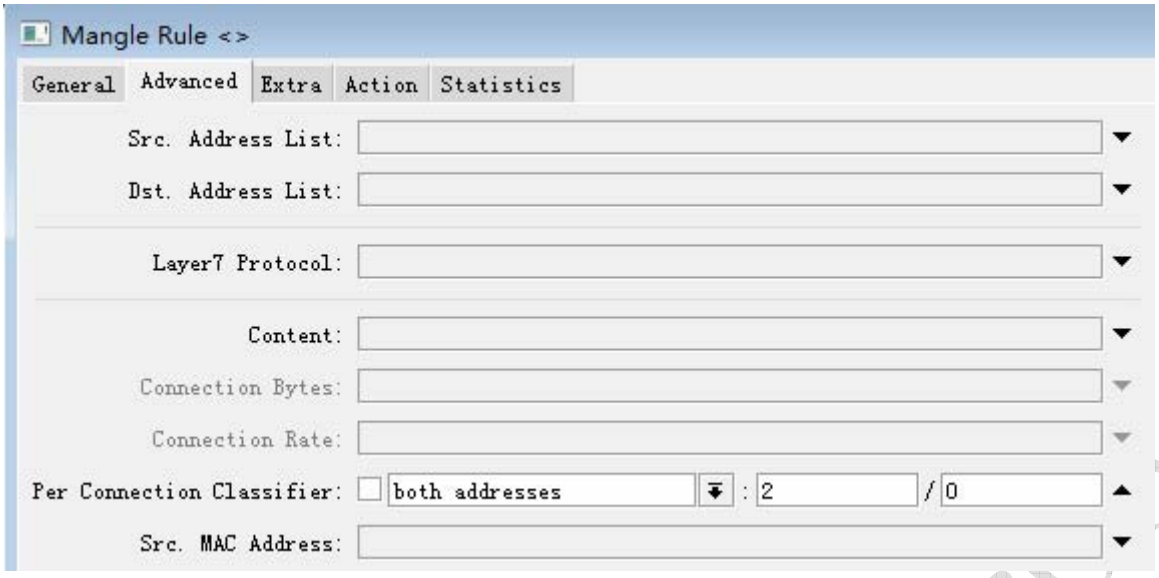


Mangle 标记配置

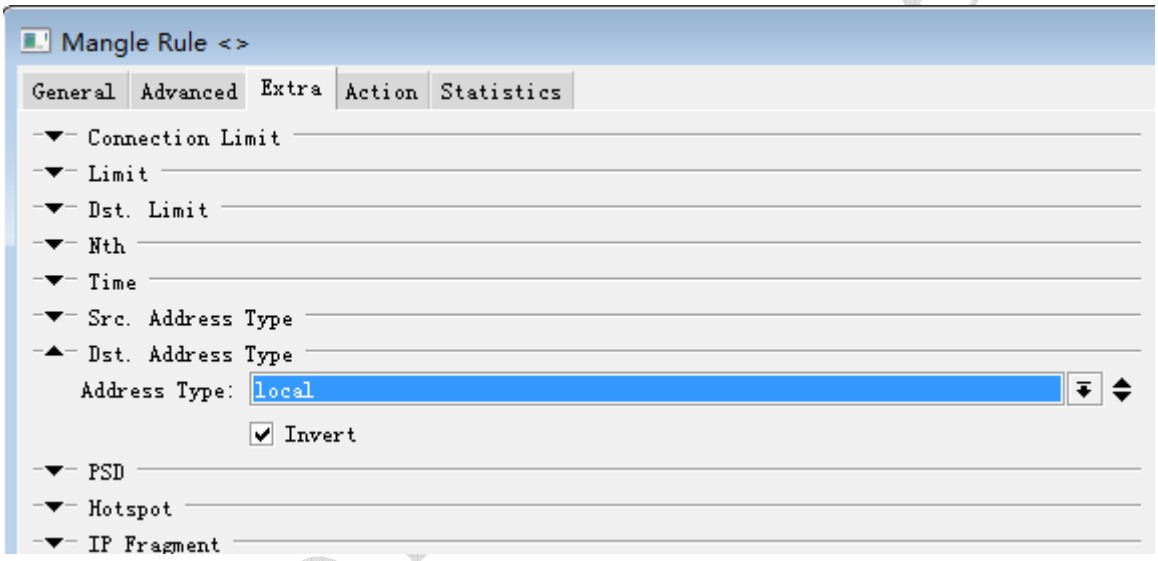
接下来我们进入 ip firewall mangle 标记连接和路由，我们使用 per-connection-classifier 双向地址进行分类做连接分类标记。

首先我们需要将进入路由的的链接进行标记

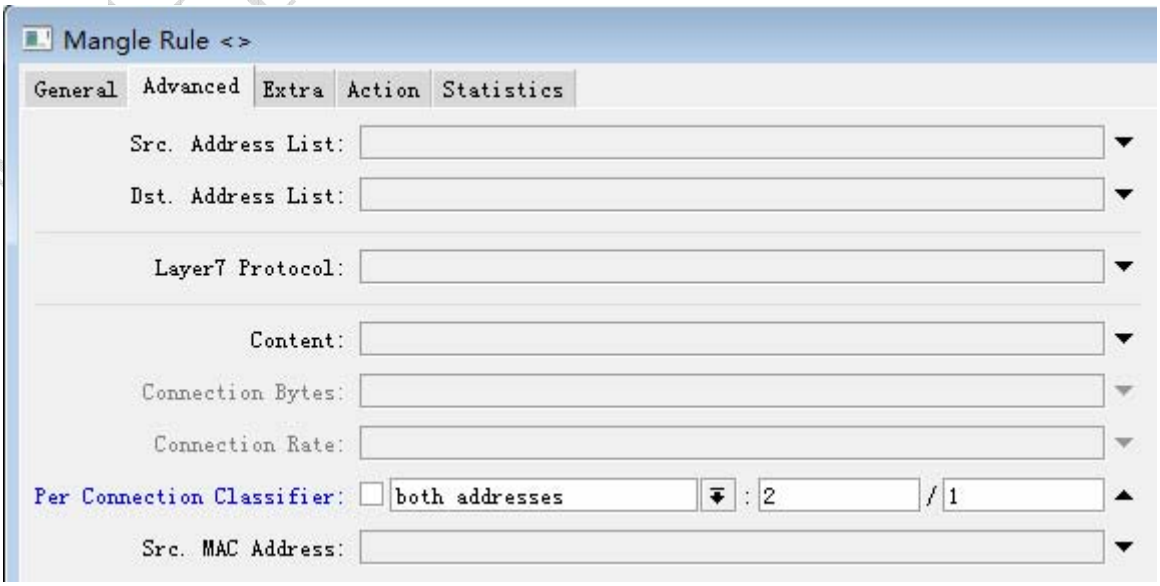
如下图，我们进入一条 mangle 规则，中的 advanced 标签内容可以看到 per-connection-classifier 分类器，选择 both-addresses 的分类：



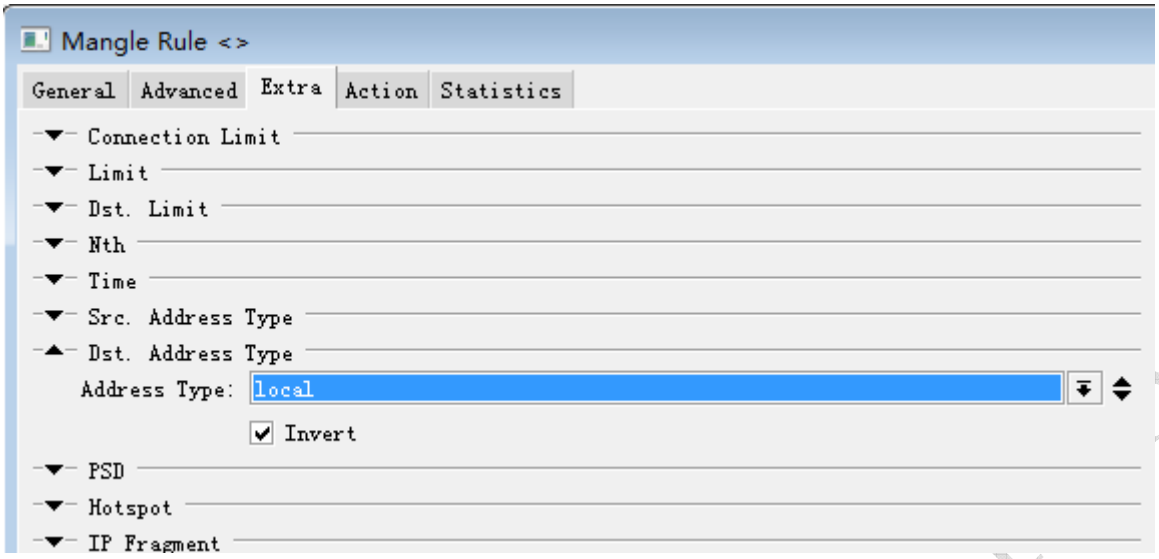
然后选择 dst-address-type=!local，即除了目标地址是本地以前的地址：



注：2 条线的分类代码定义是第一条线为 2/0，第二条为 2/1



同样选择一下地址类型：



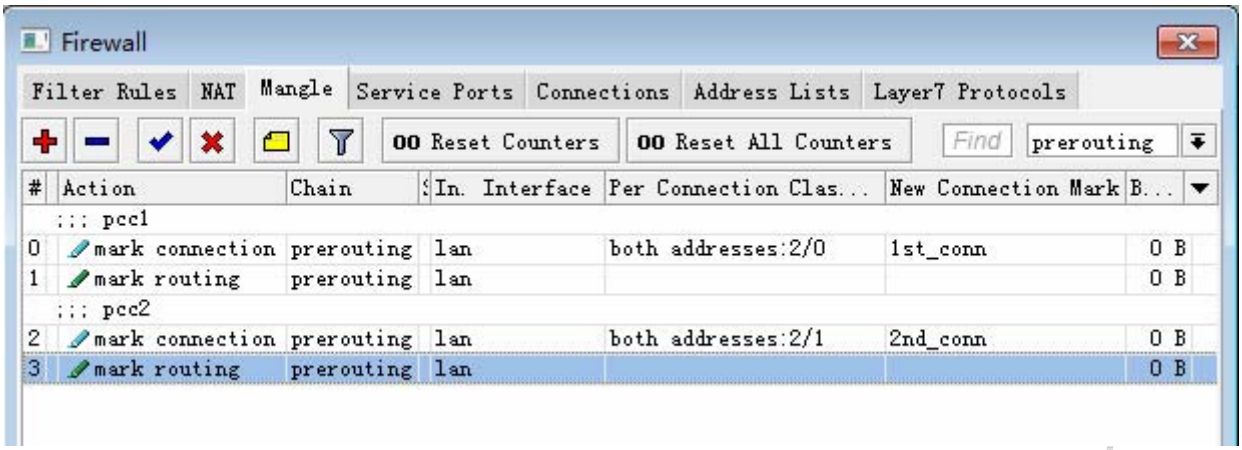
下面命令是提取走第一条线路的连接标记取名位 **1st_conn**，并从连接里提取路由标记名位 **1st_route**，设置：per-connection-classifier=both-addresses:2/0，设置 in-interface=lan

```
/ip firewall mangle
add action=mark-connection chain=prerouting comment="" disabled=no \
    in-interface=lan new-connection-mark=1st_conn passthrough=yes \
    per-connection-classifier=both-addresses:2/0
add action=mark-routing chain=prerouting comment="" connection-mark=1st_conn \
    disabled=no in-interface=lan new-routing-mark=1st_route passthrough=yes
```

提取走第二条线路的连接标记取名位 **2nd_conn**，并从连接里提取路由标记名位 **2nd_route**，设置：per-connection-classifier=both-addresses:2/1，设置 in-interface=lan:

```
/ip firewall mangle
add action=mark-connection chain=prerouting comment="" disabled=no \
    in-interface=lan new-connection-mark=2nd_conn passthrough=yes \
    per-connection-classifier=both-addresses:2/1
add action=mark-routing chain=prerouting comment="" connection-mark=2nd_conn \
    disabled=no in-interface=lan new-routing-mark=2nd_route passthrough=yes
```

在 winbox 在 mangle 中设置完成后如下：

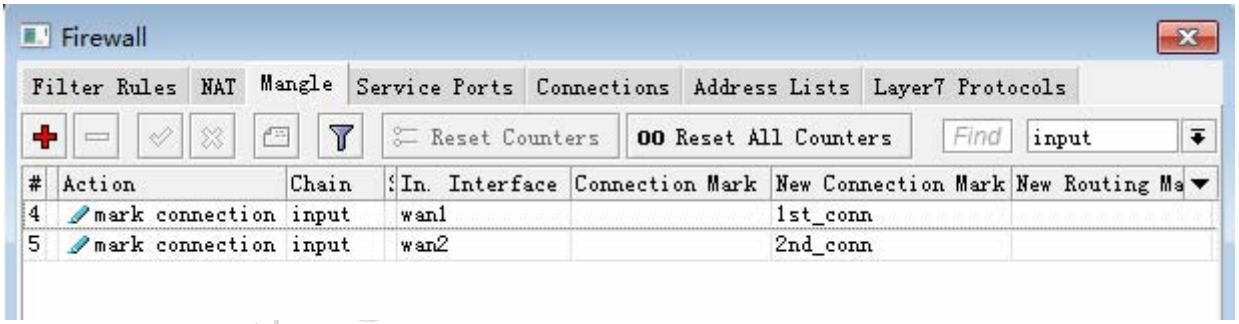


回程路由设置

我们需要将从那个口进入就从相应的口回去，即保证每个外网口的数据能得到正确的路由

```
/ ip firewall mangle
add chain=input in-interface=wan1 action=mark-connection
new-connection-mark=1st_conn
add chain=input in-interface=wan2 action=mark-connection
new-connection-mark=2nd_conn
```

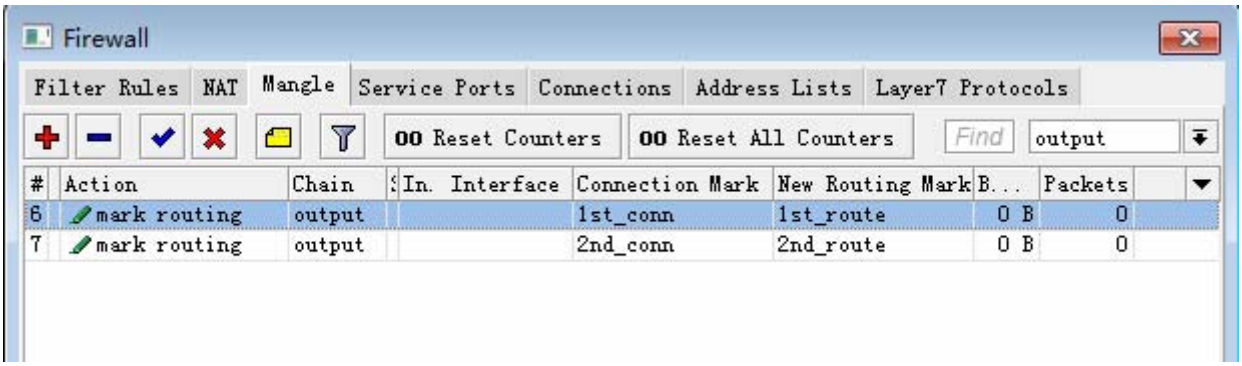
winbox 设置



标记完进入接口的链接后，将这些链接指定到相应的路由标记上：

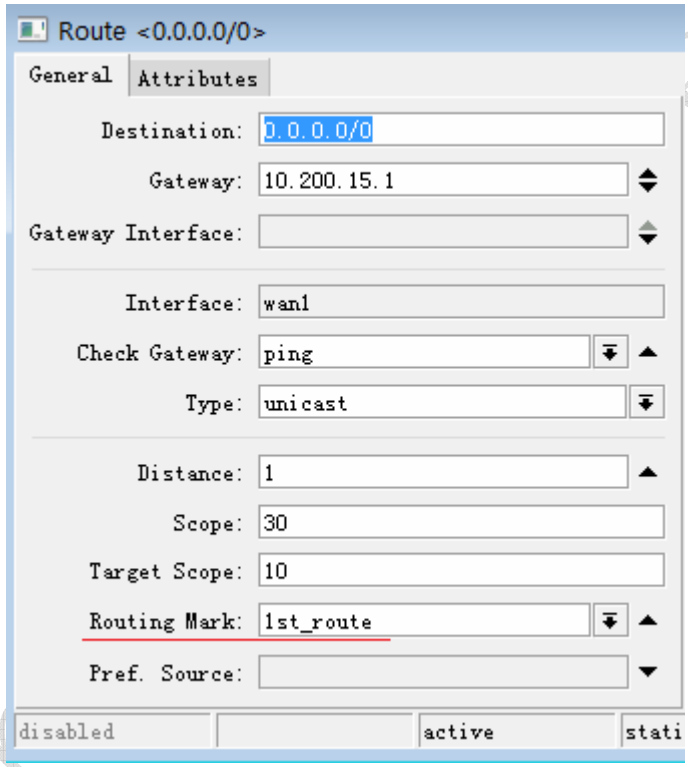
```
add chain=output connection-mark=1st_conn action=mark-routing
new-routing-mark=1st_route
add chain=output connection-mark=2nd_conn action=mark-routing
new-routing-mark=2nd_route
```

winbox 设置

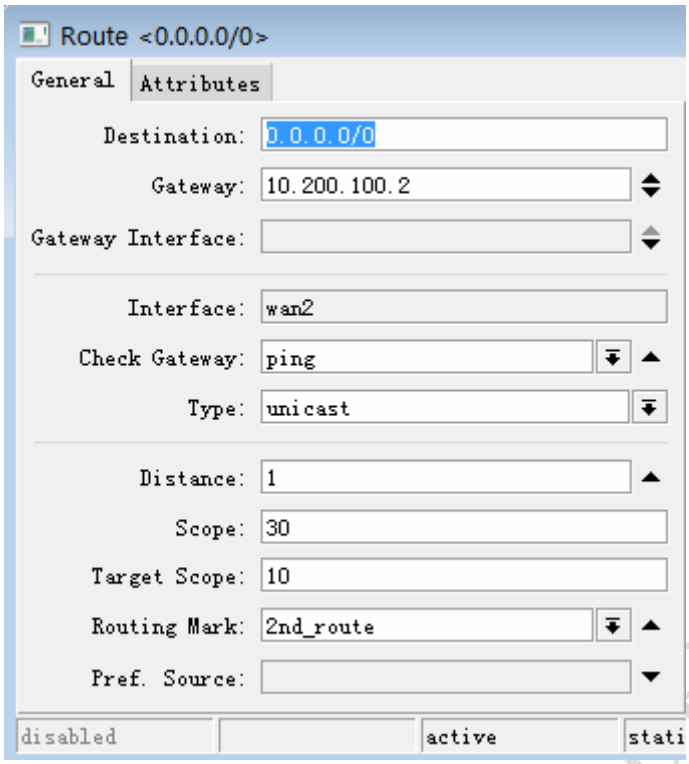


路由配置

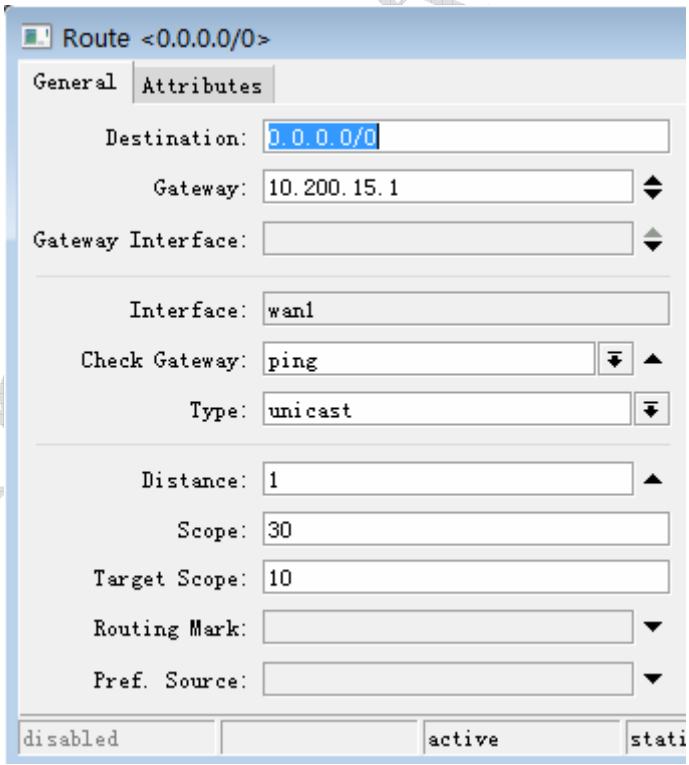
配置完标记后路由后，我们进入 ip route 配置路由，首先设置负载均衡的标记路由，首先设置第一条线路的路由标记，设置 routing-mark=1st_route：



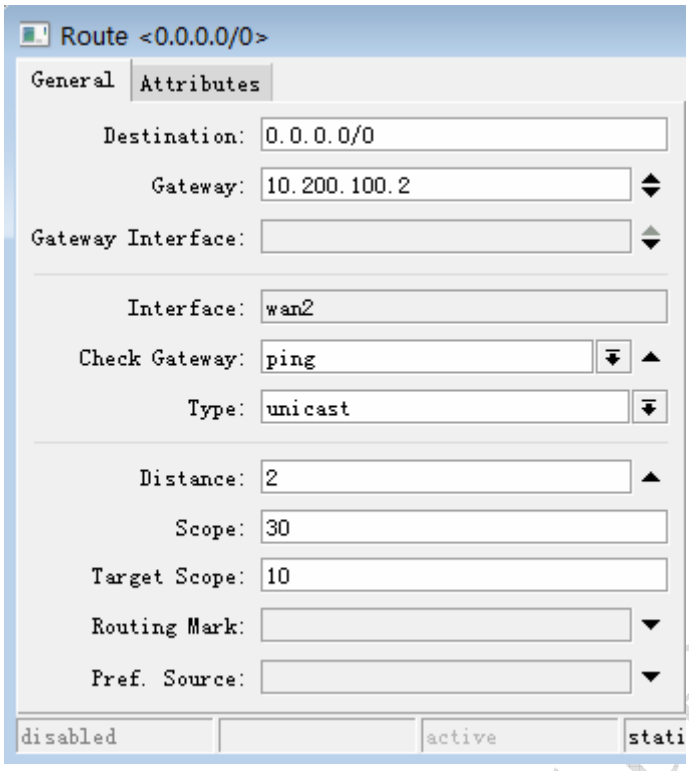
设置第二条线路的路由标记，设置 routing-mark=2nd_route：



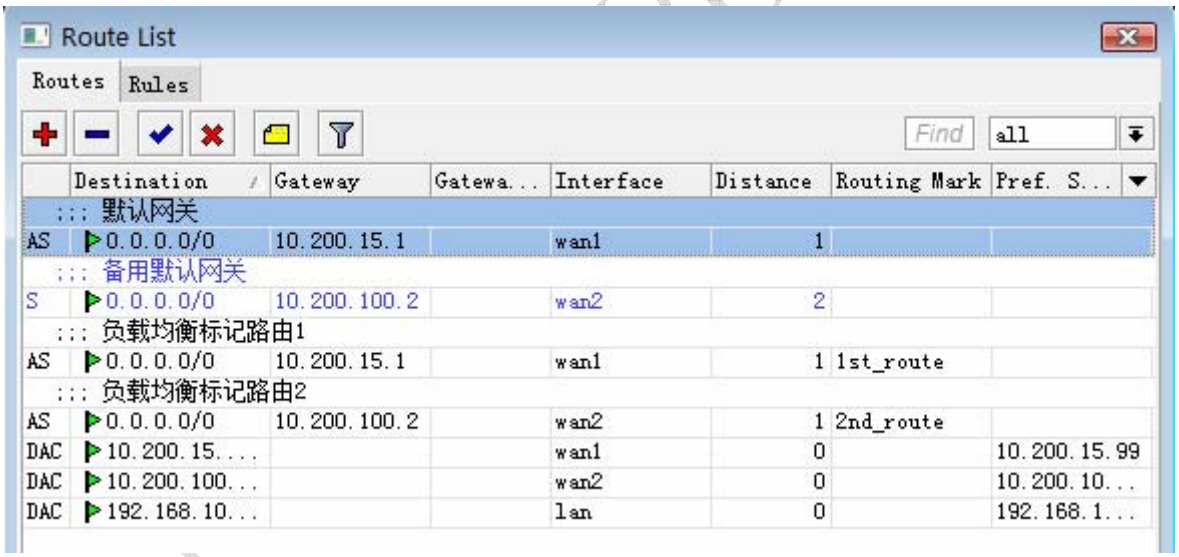
配置默认网关和备份网关，默认网关的 **distance** 设置为 1，并设置 check-gateway=ping，通过 ping 监测网关状态:



备份网关的 **distance** 设置为 2，并设置 check-gateway=ping，通过 ping 监测网关状态:



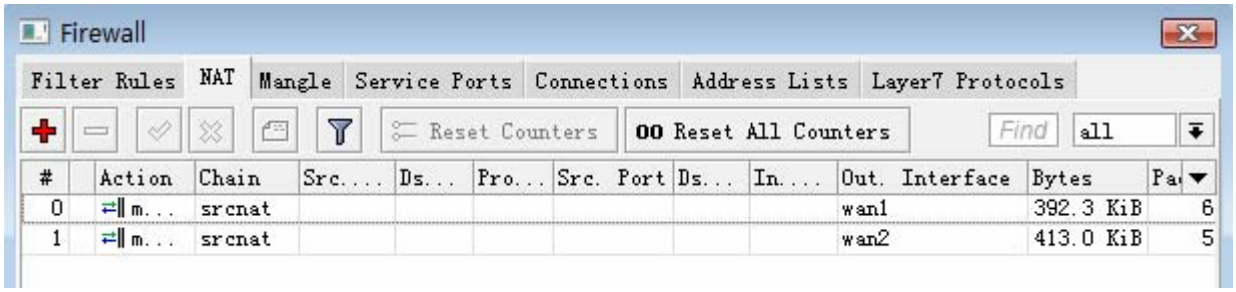
配置完成后的路由标如下图：



配置 nat

最后配置 nat 转换规则，进入 ip firewall nat 中配置 action=masquerade，分别对 2 条线路做伪装：

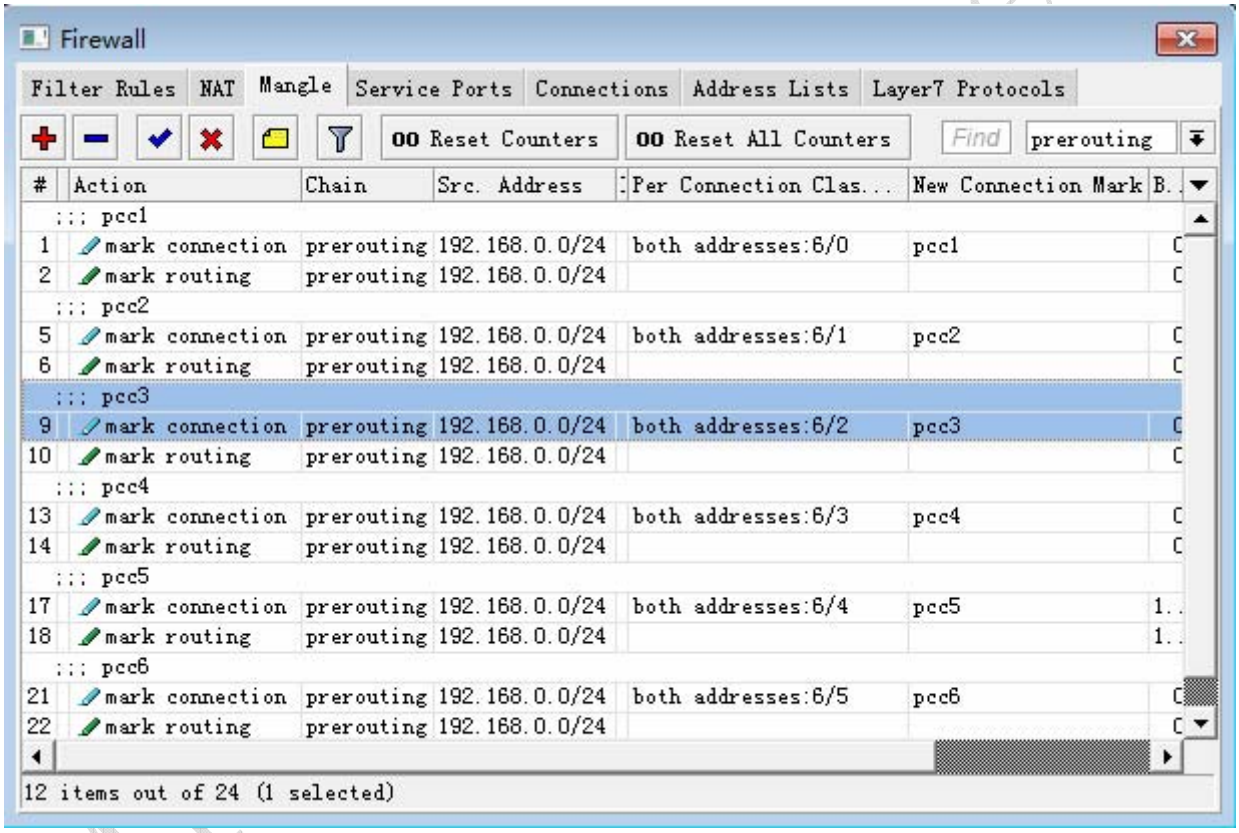
```
/ip firewall nat
add action=masquerade chain=srcnat out-interface=wan1
add action=masquerade chain=srcnat out-interface=wan2
```



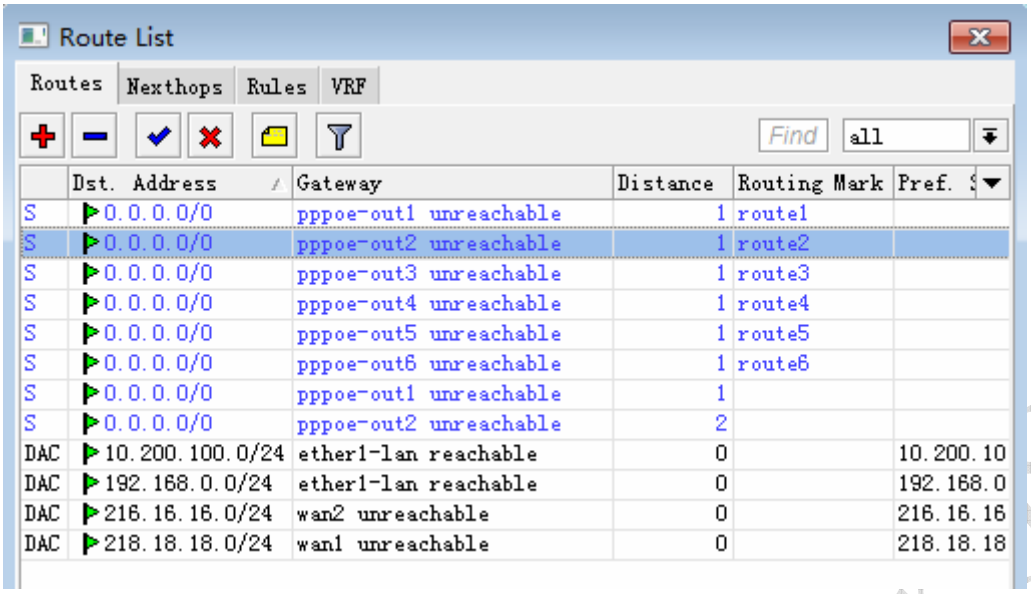
六线的 PCC 负载均衡

在许多网络中会出现相同运营商，相同带宽多线路的接入，比如下面是一个 6 条线路的接入，我们通过 PCC 解决带宽的均衡，通过分类源地址和目标地址的负载均衡，即双向地址（both addresses），设置于之前的操作基本相同，只是变成了 6 组规则

下面是 6 条 ADSL 的情况，mangle 标记的 prerouting 规则：



进入 ip route 设置路由，这个是 PPPoE 拨号上网的策略路由配置：



第七章 DHCP 操作配置

DHCP(动态主机分配协议)，负责分配和接收网络中的 IP 地址信息，能让网络内的主机动态获取地址，连接指定的网络。RouterOS 支持服务端（Server）和客户端 （Client）,同时支持 DHCP-relay 接力传输功能。

7.1 DHCP-Client 设置

操作路径: ***/ip dhcp-client***

MikroTik RouterOS DHCP-client 在任意一个以太网卡或者 WLAN 上启用，client 将会自动接受一个地址、子网掩码、默认网关和 DNS 服务器地址。收到的 IP 和子网掩码将添加到选择的网卡上，默认网关将添加到路由表中显示为动态添加，如果 DHCP-client 被禁用或没有获取信息，动态添加的路由将自动删除。

属性描述

add-default-route (yes | no; 默认: **yes**) – 是否添加指定的 DHCP 服务器的默认路由
client-id (*文本*) – 与 administraor 或 ISP 相符合的参数
enabled (yes | no; 默认: **no**) – 是否启用 DHCP 客户端
host-name (*文本*) – 客户端的主机名
interface (*名称*; 默认: **(unknown)**) – 任何以太网 interface (这包括 wireless 和 EoIP 隧道)
use-peer-dns (yes | no; 默认: **yes**) – 是否使用对端 DHCP 服务器的 DNS 的设置(将会添加到/ip dns 中)
default-route-distance (integer:0..255; 默认:) – 自动添加默认路由的路由距离，这个功能要求 add-default-route（添加默认路由）设置为 yes 才会生效
status (*bound* | *error* | *rebinding...* | *requesting...* | *searching...* | *stopped*) – 显示 DHCP-Client 的状态

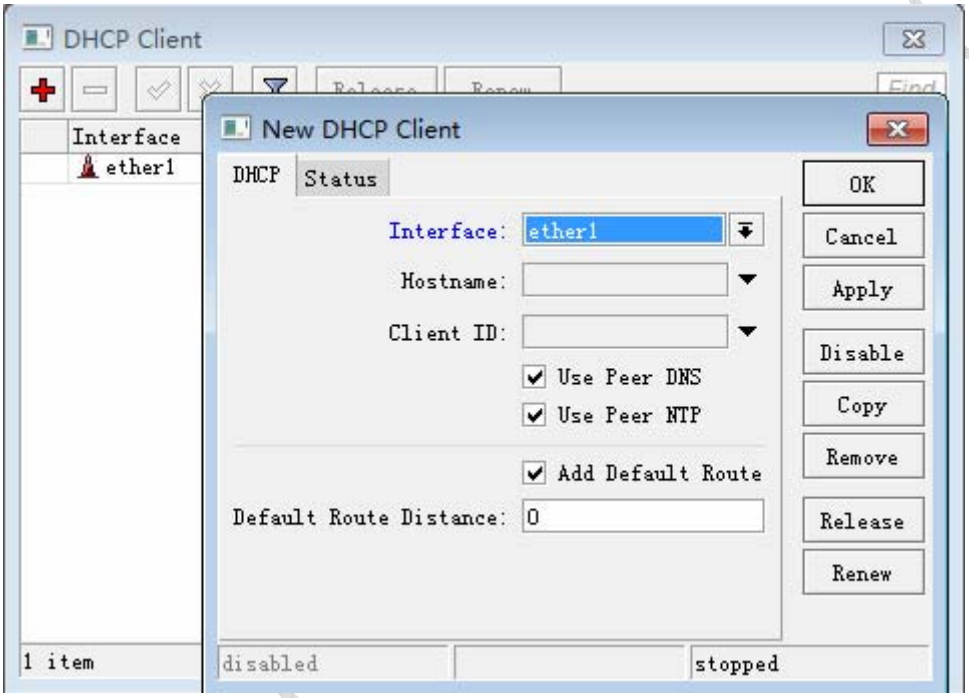
命令描述

renew（id） – 更新当前的租约，如果更新操作没有成功，客户端将试着初始化租约。
release (id) – 释放当前 DHCP 绑定，并重启 DHCP 客户端

事例：在 **ether1** interface 启用 DHCP-client：

```
/ip dhcp-client add interface=ether1 disabled=no
[admin@MikroTik] ip dhcp-client> print detail
Flags: X - disabled, I - invalid
 0   interface=ether1 add-default-route=yes use-peer-dns=yes use-peer-ntp=yes
     status=bound address=192.168.0.65/24 gateway=192.168.0.1
     dhcp-server=192.168.0.1 primary-dns=192.168.0.1 primary-ntp=192.168.0.1
     expires-after=9m44s
[admin@MikroTik] ip dhcp-client>
```

Winbox 操作：



7.2 DHCP-Server 设置

操作路径：***/ip dhcp-server***
关联操作：***/ip pool***

属性描述

dhcp server interface (*名称*) – 运行 DHCP 服务器的 interface
dhcp address space (*IP 地址/掩码*; 默认: **192.168.0.0/24**) – DHCP 服务器将出租给客户端的网络地址段
gateway (*IP 地址*; 默认: **0.0.0.0**) – 分配给客户端的网关地址
dhcp relay (*IP 地址*; 默认: **0.0.0.0**) – 在 DHCP 服务器与 DHCP 客户端的 DHCP 接力的 IP 地址
addresses to give out (*文本*) – DHCP 服务器分配给客户端的 IP 地址池
dns servers (*IP 地址*) – 分配给 DHCP 客户端的 DNS 服务器地址
lease time (时间; 默认: **3d**) – 使用的租期时间

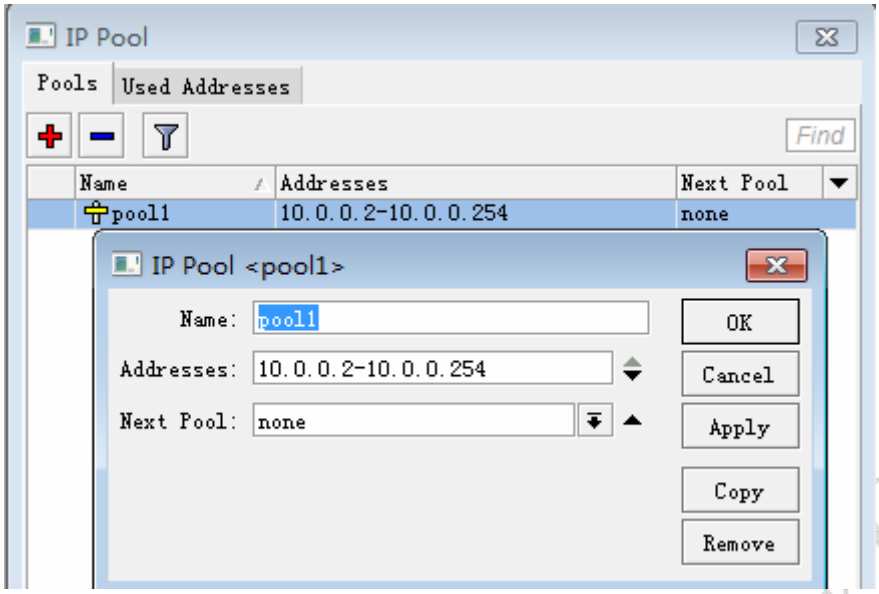
事例:配置 DHCP 服务器在 **ether1** interface 上，并分配给 10.0.0.2 到 10.0.0.254 的网络地址段，设置网关为 **10.0.0.1**，DNS 服务器为 **159.148.60.2**，租约时间为 3 天：

```
[admin@MikroTik] ip dhcp-server> setup
选择 DHCP 服务器运行的 interface

dhcp server interface: ether1
选择 DHCP 网络地址段

dhcp address space: 10.0.0.0/24
设置网关地址

gateway for dhcp network: 10.0.0.1
选择 IP 地址池给 DHCP 服务器

addresses to give out: 10.0.0.2-10.0.0.254
设置 DNS 服务器

dns servers: 159.148.60.2
设置租约时间

lease time: 3d
[admin@MikroTik] ip dhcp-server>
```

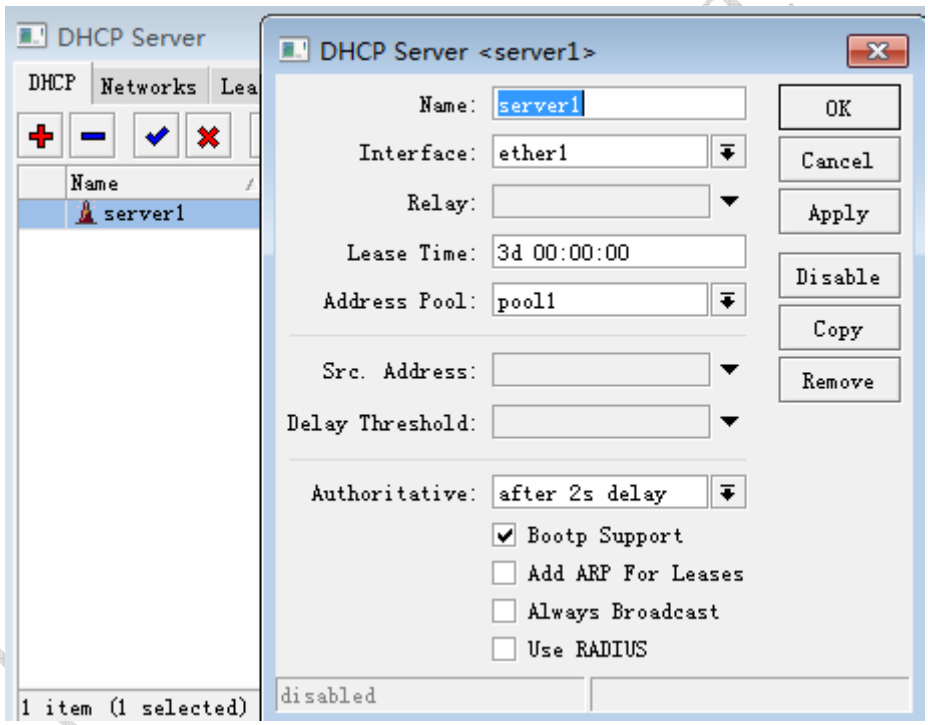
在上面向导中设置的内容，通过命令查看如下：

```
[admin@MikroTik] ip dhcp-server> print
Flags: X - disabled, I - invalid
 #   NAME          INTERFACE RELAY         ADDRESS-POOL LEASE-TIME ADD-ARP
 0   dhcp1         ether1    0.0.0.0       dhcp_pool1   3d         no
[admin@MikroTik] ip dhcp-server> network print
 # ADDRESS          GATEWAY        DNS-SERVER     WINS-SERVER    DOMAIN
 0 10.0.0.0/24       10.0.0.1       159.148.60.2
[admin@MikroTik] ip dhcp-server> /ip pool print
 # NAME                                RANGES
 0 dhcp_pool1                           10.0.0.2-10.0.0.254
[admin@MikroTik] ip dhcp-server>
```

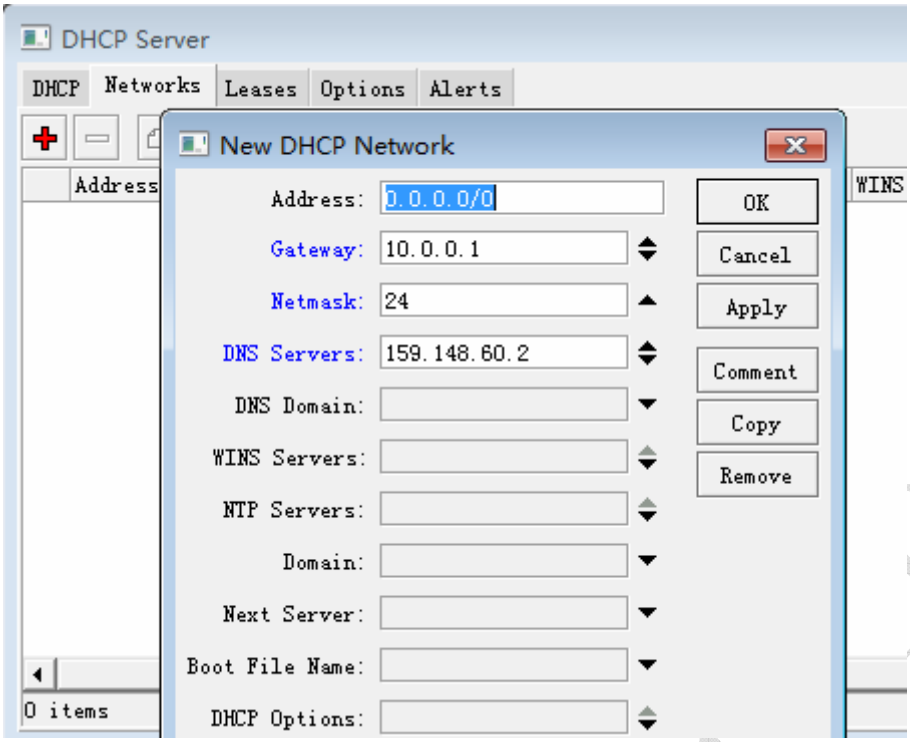
Winbox 操作：添加 DHCP 服务，首先进入/ip pool 中分配地址池范围，注意添加时要排除网关地址：



进入/ip dhcp-server 添加 DHCP 服务器接口道 ether1 和对应的地址池



在/ip dhcp-server network 中添加分配的网关和 DNS 参数：



第八章 DNS 配置

DNS 缓存是使用最小的 DNS 请求时间连接到外部的 DNS 服务器，这相当于一个简单的本地 DNS 服务。

需要功能包: ***system***
需要等级: *Level1*
操作路径: ***/ip dns***

8.1 DNS 配置

属性描述

allow-remote-requests (yes | no) – 是否允许指定远程网络的请求
primary-dns (*IP 地址*; 默认: **0.0.0.0**) – 首选 DNS 服务器
secondary-dns (*IP 地址*; 默认: **0.0.0.0**) – 备用 DNS 服务器
cache-size (整型: 512..10240; 默认: **2048 kB**) – 指定 DNS 缓存的长度单位为 KB
cache-max-ttl (时间; 默认: **7d**) – 指定缓存记录的最大存活周期
cache-used (只读:*整型*) – 显示当前使用的缓存大小 KB

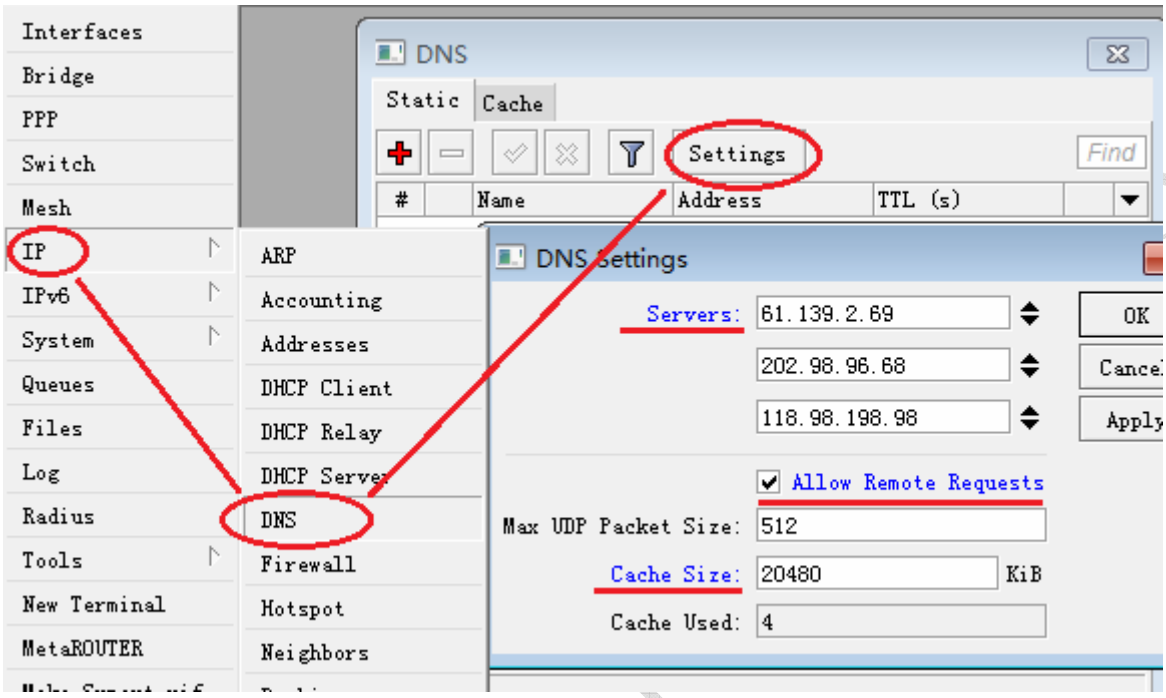
注: 如果**/ip dhcp-client** 属性下的 **use-peer-dns** 设置为 **yes**，这时**/ip dns** 下的 **primary-dns** 将会改变，并修改 DHCP 服务的 DNS 设置。

事例: 设置首选 DNS 服务器为 61.139.2.69:

```
[admin@MikroTik] ip dns> set primary-dns=61.139.2.69
[admin@MikroTik] ip dns> print
```

```
    resolve-mode: remote-dns
     primary-dns: 61.139.2.69
   secondary-dns: 0.0.0.0
[admin@MikroTik] ip dns>
```

在 4.6 版本后增加了多个 DNS 服务器的支持，操作如下：



注意：这里的 allow remote requests 为启用 DNS 缓存功能，cashe size 设定缓存存储大小

缓存状态

操作路径：***/ip dns cache***

name (只读：*名称*) – 主机的 DNS 名称

address (只读：*IP 地址*) – 主机 IP 地址

ttl (时间) – 剩余的存活周期

8.2 内部 DNS 域名解析

操作路径：***/ip dns static***

MikroTik RouterOS 在 DNS 缓存中嵌入了 DNS 服务器的一些特征，如通过使用路由器的 DNS 作域名解析 IP 地址。

属性描述

name (文本) – 分配给 IP 地址的 DNS 名称。

address (*IP 地址*) – 分配给域名的 IP 地址

事例：为 **www.example.com** 域名添加静态 DNS，IP 地址是 **10.0.0.1**：

```
[admin@MikroTik] ip dns static> add name www.example.com address=10.0.0.1
[admin@MikroTik] ip dns static> print
```

```
# NAME                                              ADDRESS         TTL
0 aaa.aaa.a                                          123.123.123.123 1d
1 www.example.com                                     10.0.0.1       1d
[admin@MikroTik] ip dns static>
```

刷新 DNS 缓存

操作指令: ***/ip dns cache flush***

flush – 清除内部 DNS 的缓存 clears internal DNS cache

```
[admin@MikroTik] ip dns> cache flush
[admin@MikroTik] ip dns> print
            primary-dns: 159.148.60.2
          secondary-dns: 0.0.0.0
   allow-remote-requests: no
             cache-size: 2048 kB
          cache-max-ttl: 7d
             cache-used: 10 kB
[admin@MikroTik] ip dns>
```

第九章 防火墙过滤（Firewall Filte）

在 RouterOS 通过 ip firewall 能对 IP 数据包过滤、P2P 协议过滤、源和目标 IP、端口、IP 协议、协议（ICMP、TCP、MSS 等）、网络接口、对内部的数据包和连接作标记、ToS 字节、内容过滤、顺序优先与数据频繁和时间控制、包长度控制…

从数据传输上分类：分为 input、foreward 和 output 三种链表（chain）过滤，不管是二层或者三层过滤上都包含这三个链表。RouterOS 的防火墙包括了对 address-list 和 L7-protocol 等调用

快速设置向导

- 添加一条 firewall 规则，将所有通过路由器到目标协议为 TCP，端口为 135 的数据包丢弃掉：

```
/ip firewall filter add chain=forward dst-port=135 protocol=tcp action=drop
```

- 拒绝通过 Telnet 访问路由器(协议 TCP，端口 23)：

```
/ip firewall filter add chain=input protocol=tcp dst-port=23 action=drop
```

9.1 Firewall 过滤

操作路径: ***/ip firewall filter***

网络防火墙始终保持对那些有威胁敏感的数据进入内部网络中，无论怎样网络都是连接在一起的，总是会有某些从外闯入你的 LAN，窃取资料和破坏内部网络。适当的配置防火墙可以有效的保护网络。

MikroTik RouterOS 是功能非常强大的防火墙，包括以下特征：

- 包过滤功能
- P2P 协议过滤
- 7 层协议过滤
- IPv6 防火墙过滤
- 数据传输分类：
  - 源 MAC 地址
  - IP 地址（网段或列表）和地址类型（广播、本地、组播）
  - 端口或端口长度
  - IP 协议
  - 协议选择选项(ICMP 类型和代码字段、TCP 标记、IP 选项和 MSS)
  - Interface 的数据包从那里到达或通过那里里去
  - 内部数据流与连接标记
  - ToS (DSCP)
  - 数据包内容
  - Connection-rate 连接速率
  - PCC 分离器
  - 数据包大小
  - 包到达时间

基本过滤规则

防火墙操作是借助于防火墙的策略，一个策略规则是告诉路由器如何处理一个 IP 数据包，每一条策略都由两部分组成，一部份是传输状态配置和定义如何操作数据包。数据链（Chains）是为更好的管理和组织策略。

过滤功能有三个默认的数据链（chains）：**input**, **forward** 和 **output** 他们分别负责从哪里进入路由器的、通过路由器转发的与从路由器发出的数据。用户也可用自定义添加链，当然这些链没有默认的传输配置，需要在三条默认的链中对 **action=jump** 策略中相关的 **jump-target** 进行配置。

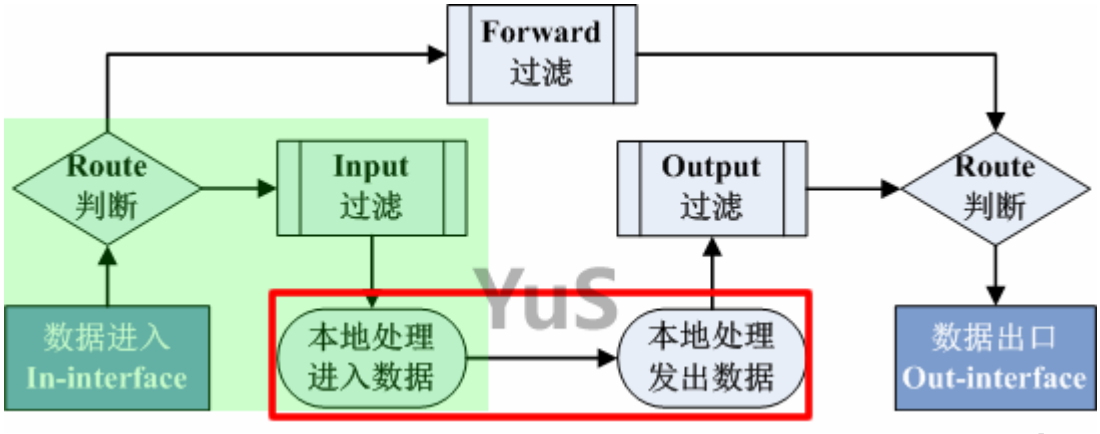
过滤链

下面是三条预先设置好了的 chains，他们是不被能删除的：

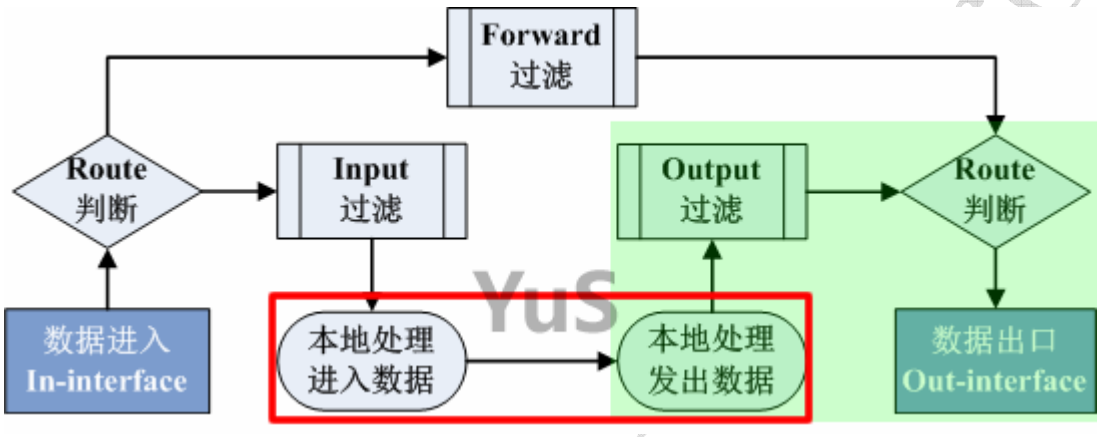
- **input** – 用于处理进入路由器的数据包，即数据包目标 IP 地址是到达路由器一个接口的 IP 地址，经过路由器的数据包不会在 input-chains 处理。
- **forward** – 用于处理通过路由器的数据包
- **output** – 用于处理源于路由器并从其中一个接口出去的数据包。

他们具体的区别如下：

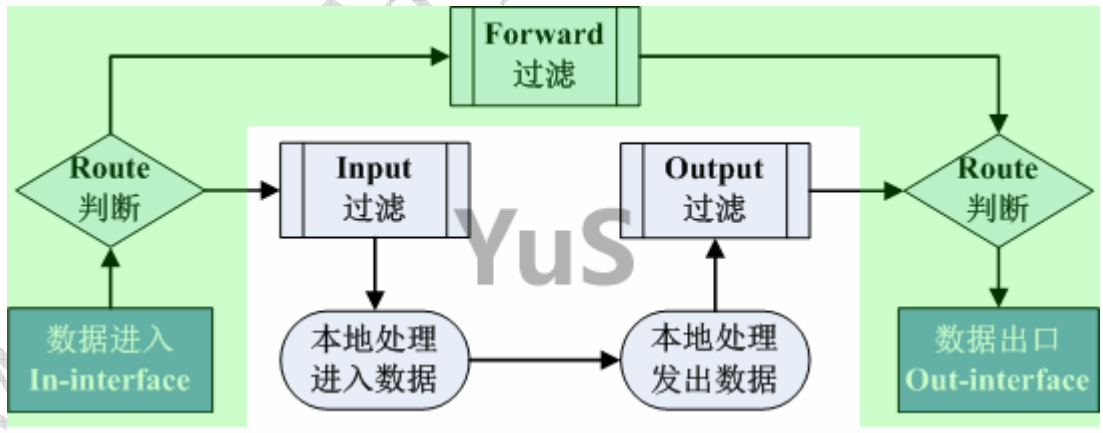
IP 数据包进入 input 链表的数据工作流程：



IP 数据包进入 output 链表的流程：

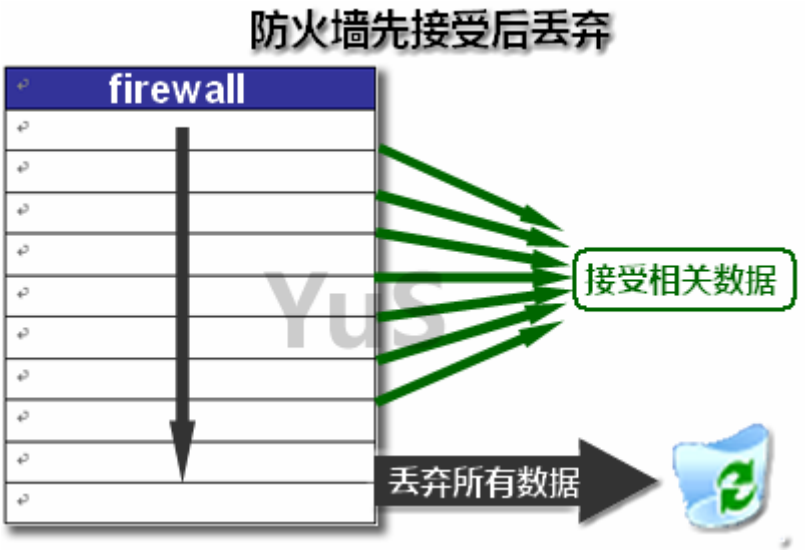


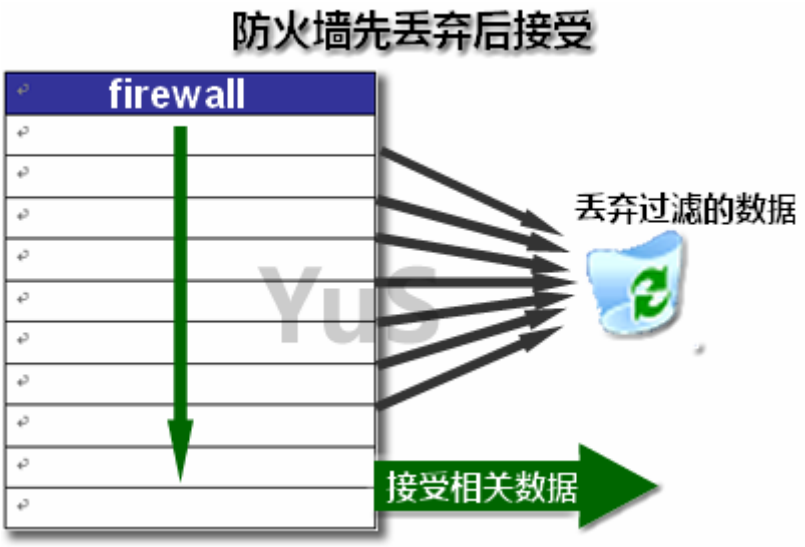
IP 数据进入 forward 链表的流程



当处理一个 chain（数据链），策略是从 chain 列表的顶部从上而下执行的。如果一个数据包满足策略的条件，这时会执行该操作。

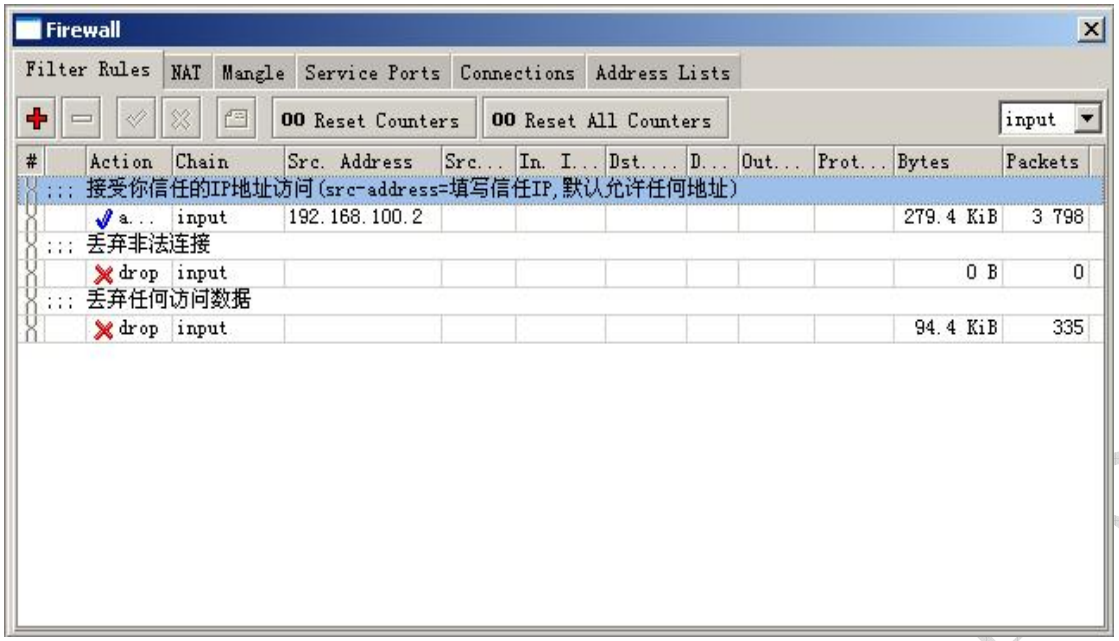
我们来看看防火墙过滤原则：





9.2 防火墙规则事例

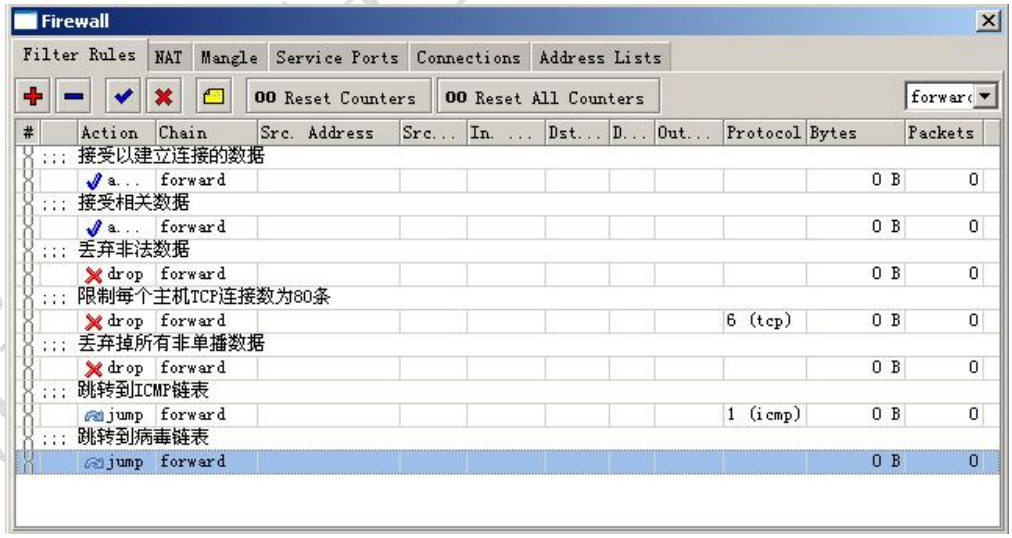
我先从 input 链表开始，这里是对所有访问路由的数据进行过滤和处理：



从 input 链表的第一条开始执行，这里一共有三条规则：

```
0   ;;;  接受你信任的 IP 地址访问(src-address=填写信任 IP,默认允许任何地址)
     chain=input src-address=192.168.100.2 action=accept
1   ;;;  丢弃非法连接
     chain=input connection-state=invalid action=drop
2   ;;;  丢弃任何访问数据
     chain=input action=drop
```

下面是 forward 链表

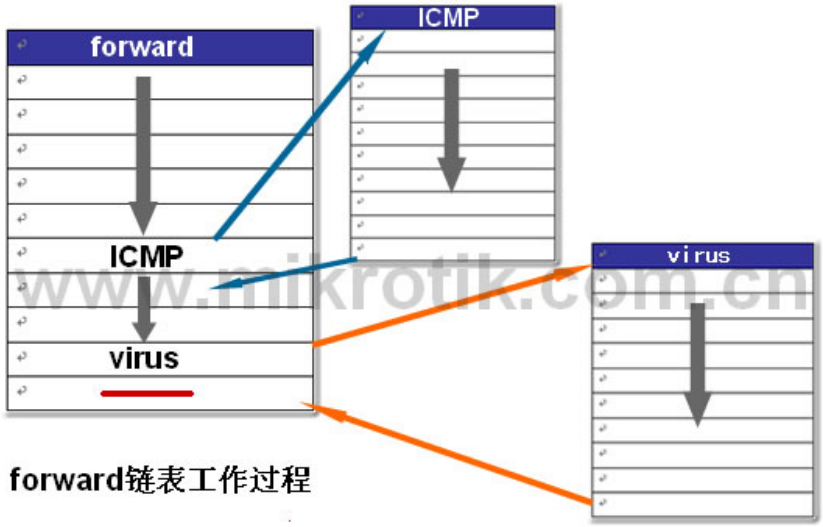


forward 链表，一共有 7 条规则，包括两个跳转到自定义链表 ICMP 和 virus 链表：

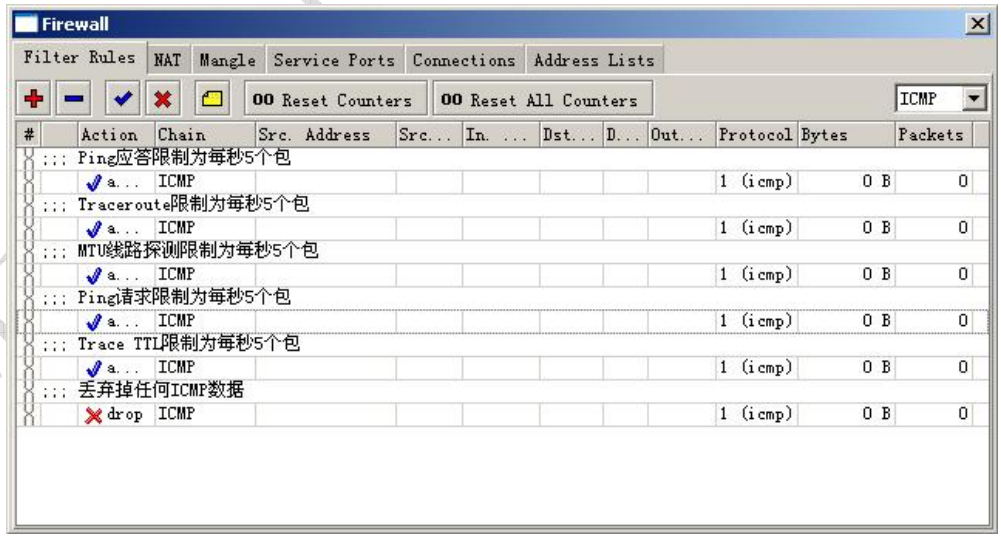
```
0  ;;;  接受已建立连接的数据
     chain=forward connection-state=established action=accept
1  ;;;  接受相关数据
     chain=forward connection-state=related action=accept
2  ;;;  丢弃非法数据包
```

```
    chain=forward connection-state=invalid action=drop
3   ;;;  限制每个主机 TCP 连接数为 80 条
    chain=forward protocol=tcp connection-limit=80,32 action=drop
4   ;;;  丢弃掉所有非单播数据
    chain=forward src-address-type=!unicast action=drop
5   ;;;  跳转到 ICMP 链表
    chain=forward protocol=icmp action=jump jump-target=ICMP
6   ;;;  跳转到病毒链表
    chain=forward action=jump jump-target=virus
```

forward 工作过程如下：



在自定义链表 ICMP 中，是定义所有 ICMP（Internet 控制报文协议），ICMP 经常被认为是 IP 层的一个组成部分。它传递差错报文以及其他需要注意的信息。ICMP 报文通常被 IP 层或更高层协议（TCP 或 UDP）使用。例如：ping、traceroute、trace TTL 等。我们通过 ICMP 链表来过滤所有的 ICMP 协议：



ICMP 链表操作过程：

```
0   ;;;  Ping 应答限制为每秒 5 个包
    chain=ICMP protocol=icmp icmp-options=0:0-255 limit=5,5 action=accept
1   ;;;  Traceroute 限制为每秒 5 个包
```

```
   chain=ICMP protocol=icmp icmp-options=3:3 limit=5,5 action=accept
2  ;;;  MTU 线路探测限制为每秒 5 个包
   chain=ICMP protocol=icmp icmp-options=3:4 limit=5,5 action=accept
3  ;;;  Ping 请求限制为每秒 5 个包
   chain=ICMP protocol=icmp icmp-options=8:0-255 limit=5,5 action=accept
4  ;;; Trace TTL 限制为每秒 5 个包
   chain=ICMP protocol=icmp icmp-options=11:0-255 limit=5,5 action=accept
5  ;;;  丢弃掉任何 ICMP 数据
   chain=ICMP protocol=icmp action=drop
```

ICMP 类型：代码值

通过指令保护你的路由器和相连接私有网络，你需要通过配置防火墙丢弃或拒绝 ICMP 协议的传输。然而一些 ICMP 数据包则需要用来维护网络和故障判断用。

下面是 ICMP 类型列表：通常下面的 ICMP 传输建议被允许通过

Ping

- **8:0** – 回应请求
- **0:0** – 回应答复

Trace

- **11:0** – TTL 超出
- **3:3** – 端口不可到达

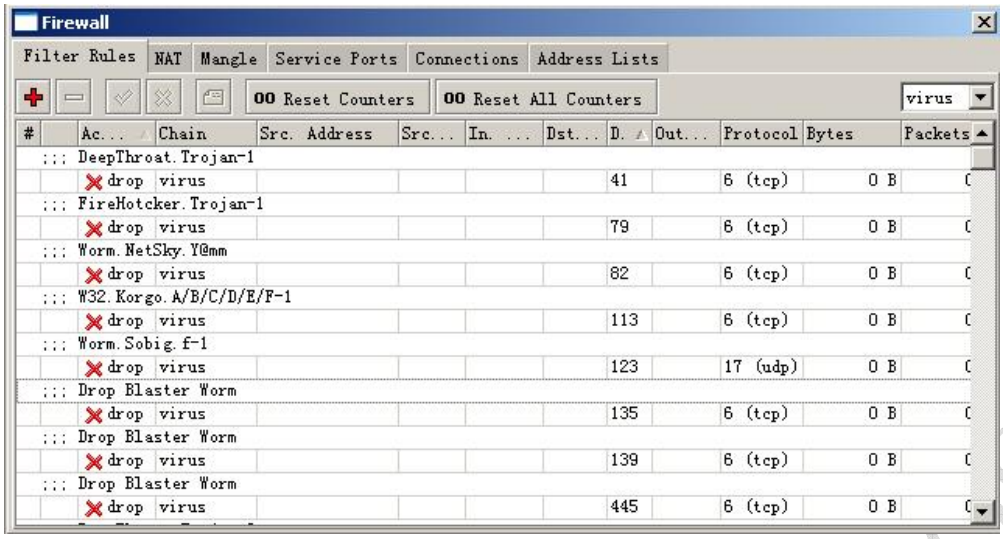
路径 MTU 探测

- **3:4** – 分段存储 Fragmentation-DF-Set

一般 ICMP 过滤建议：

- 允许 ping—ICMP 回应请求向外发送和回应答复进入
- 允许 traceroute—TTL 超出和端口不可到达信息进入
- 允许路径 MTU—ICMP Fragmentation-DF-Set 信息进入
- 阻止其他任何数据

在 virus 链表中过滤常见的病毒，我可以根据需要在该链表中添加新的病毒对他们做过滤：



阻止不必要的 IP 广播：

```
add chain=forward src-address=0.0.0.0/8 action=drop
add chain=forward dst-address=0.0.0.0/8 action=drop
add chain=forward src-address=127.0.0.0/8 action=drop
add chain=forward dst-address=127.0.0.0/8 action=drop
add chain=forward src-address=224.0.0.0/3 action=drop
add chain=forward dst-address=224.0.0.0/3 action=drop
```

建立新的跳转数据链（chains）：

```
add chain=forward protocol=tcp action=jump jump-target=tcp
add chain=forward protocol=udp action=jump jump-target=udp
add chain=forward protocol=icmp action=jump jump-target=icmp
```

建立 tcp-chain 并拒绝一些 tcp 端口：

```
add chain=tcp protocol=tcp dst-port=69 action=drop comment="deny TFTP"
add chain=tcp protocol=tcp dst-port=111 action=drop comment="deny RPC portmapper"
add chain=tcp protocol=tcp dst-port=135 action=drop comment="deny RPC portmapper"
add chain=tcp protocol=tcp dst-port=137-139 action=drop comment="deny NBT"
add chain=tcp protocol=tcp dst-port=445 action=drop comment="deny cifs"
add chain=tcp protocol=tcp dst-port=2049 action=drop comment="deny NFS"
add chain=tcp protocol=tcp dst-port=12345-12346 action=drop comment="deny NetBus"
add chain=tcp protocol=tcp dst-port=20034 action=drop comment="deny NetBus"
add chain=tcp protocol=tcp dst-port=3133 action=drop comment="deny BackOriffice"
add chain=tcp protocol=tcp dst-port=67-68 action=drop comment="deny DHCP"
```

在 udp-chain 中拒绝非法的 udp 端口 Deny udp ports in udp chain:

```
add chain=udp protocol=udp dst-port=69 action=drop comment="deny TFTP"
add chain=udp protocol=udp dst-port=111 action=drop comment="deny PRC portmapper"
add chain=udp protocol=udp dst-port=135 action=drop comment="deny PRC portmapper"
add chain=udp protocol=udp dst-port=137-139 action=drop comment="deny NBT"
add chain=udp protocol=udp dst-port=2049 action=drop comment="deny NFS"
```

```
add chain=udp protocol=udp dst-port=3133 action=drop comment="deny BackOriffice"
```

在 icmp-chain 允许相应需要的 icmp 连接：

```
add chain=icmp protocol=icmp icmp-options=0:0 action=accept
        comment="drop invalid connections"
add chain=icmp protocol=icmp icmp-options=3:0 action=accept \
        comment="allow established connections"
add chain=icmp protocol=icmp icmp-options=3:1 action=accept \
        comment="allow already established connections"
add chain=icmp protocol=icmp icmp-options=4:0 action=accept \
        comment="allow source quench"
add chain=icmp protocol=icmp icmp-options=8:0 action=accept \
        comment="allow echo request"
add chain=icmp protocol=icmp icmp-options=11:0 action=accept \
        comment="allow time exceed"
add chain=icmp protocol=icmp icmp-options=12:0 action=accept \
        comment="allow parameter bad"
add chain=icmp action=drop comment="deny all other types"
```

8.3 Peer-to-Peer 协议过滤

Peer-to-peer 协议即我们所说的用于主机间点对点传输 *p2p*。这个技术有许多优秀的应用如 Skype，但同时也带了需要的为许可的软件和媒体在网络中泛滥。甚至影响到 http 和 e-mail 的正常使用。RouterOS 能识别大多 P2P 协议的连接，并能通过 QOS 进行过滤，丢弃所有的 P2P 协议：

```
[admin@MikroTik] /ip firewall filter> add chain=forward p2p=all-p2p action=drop
[admin@MikroTik] /ip firewall filter> print chain=forward
Flags: X - disabled, I - invalid, D - dynamic
0   chain=forward action=drop p2p=all-p2p
```

能探测到该协议的列表：

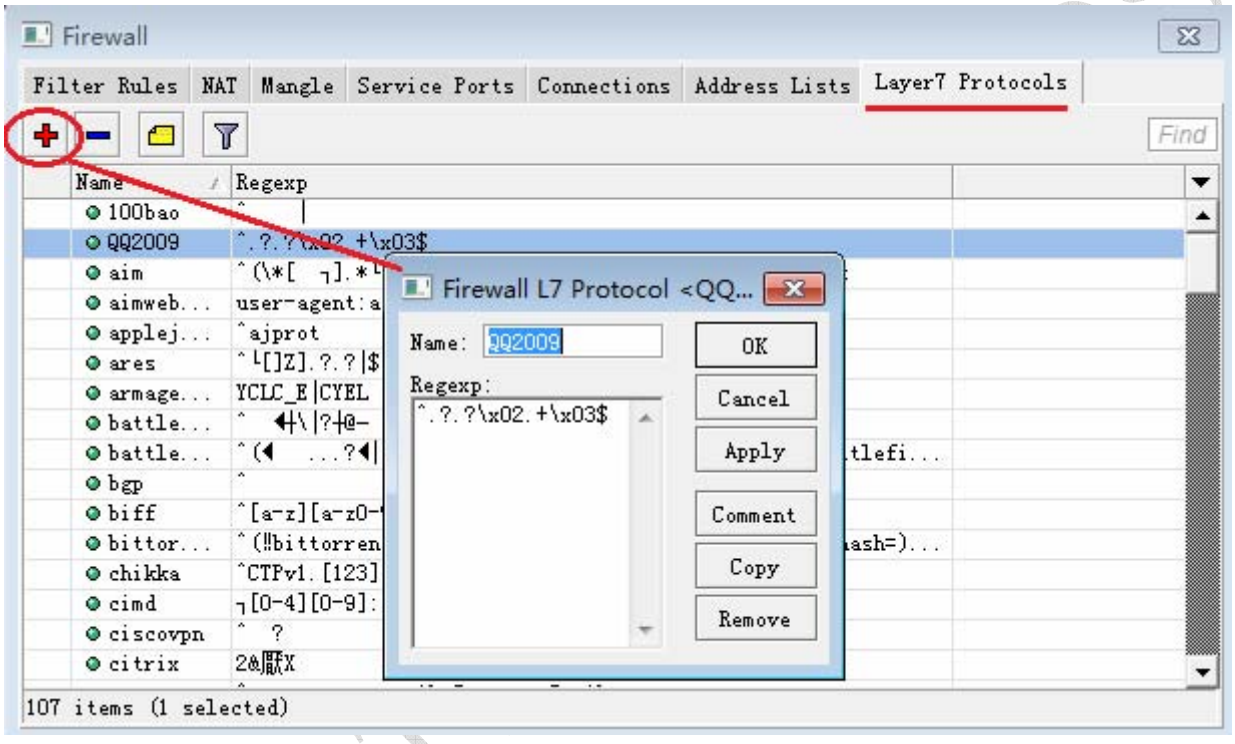
- **Fasttrack** (Kazaa, KazaaLite, Diet Kazaa, Grokster, iMesh, giFT, Poisoned, mlMac)
- **Gnutella** (Shareaza, XoLoX, , Gnucleus, BearShare, LimeWire (java), Morpheus, Phex, Swapper, Gtk-Gnutella (linux), Mutella (linux), Qtella (linux), MLDonkey, Acquisition (Mac OS), Poisoned, Swapper, Shareaza, XoloX, mlMac)
- **Gnutella2** (Shareaza, MLDonkey, Gnucleus, Morpheus, Adagio, mlMac)
- **DirectConnect** (DirectConnect (AKA DC++), MLDonkey, NeoModus Direct Connect, BCDC++, CZDC++)
- **eDonkey** (eDonkey2000, eMule, xMule (linux), Shareaza, MLDonkey, mlMac, Overnet)
- **Soulseek** (Soulseek, MLDonkey)
- **BitTorrent** (BitTorrent, BitTorrent++, uTorrent, Shareaza, MLDonkey, ABC, Azureus, BitAnarch, SimpleBT, BitTorrent.Net, mlMac)
- **Blubster** (Blubster, Piolet)
- **WPNP** (WinMX)
- **Warez** (Warez, Ares; starting from 2.8.18) – 该协议能被丢弃掉（drop），但不能被限制速度

9.4 RouterOS 7 层协议

RouterOS V3.0 在防火墙中增加了一个新得功能——7 层协议过滤。针对一些应用程序如 skype、QQ、MSN、魔兽世界……网络程序做限制和过滤。

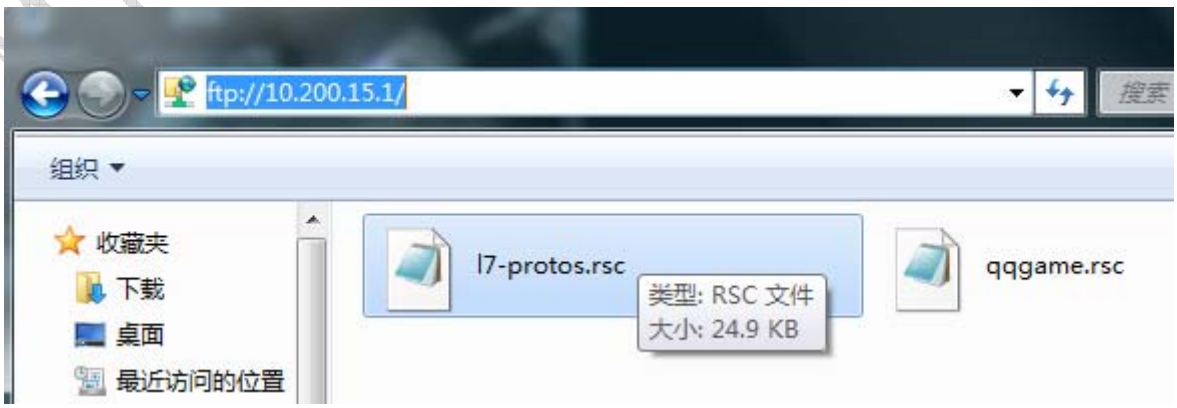
在防火墙 Layer7-protocol 目录下，你可以添加正则表达式字符串协议，定义他们的名称，并在防火墙 filter 目录下丢弃他们的数据。这个功能将不会查看单独的数据包，会查看整个数据的相关连接，收集数据直到第 10 个数据包或者 2kb 数据，来执行第一次操作

下面介绍一下具体方法的使用：7 层协议过滤增加在 ip firewall 中 Layer7 Protocols，我们可以在下面的图中看到：



7 层协议通过 Regexp 脚本编写相应应用程序的过滤代码，Regexp 可以通过网上搜索相关资料了解。在这里我们已经提供了一些常用程序的 7 层协议脚本：

通过在 http://www.mikrotik.com.cn/download/m3dex.htm 下载 MikroTik RouterOS 3.0 7 层协议过滤脚本。然后我们可以通过 FTP 上传或者直接拖放到 Files 对话框中。

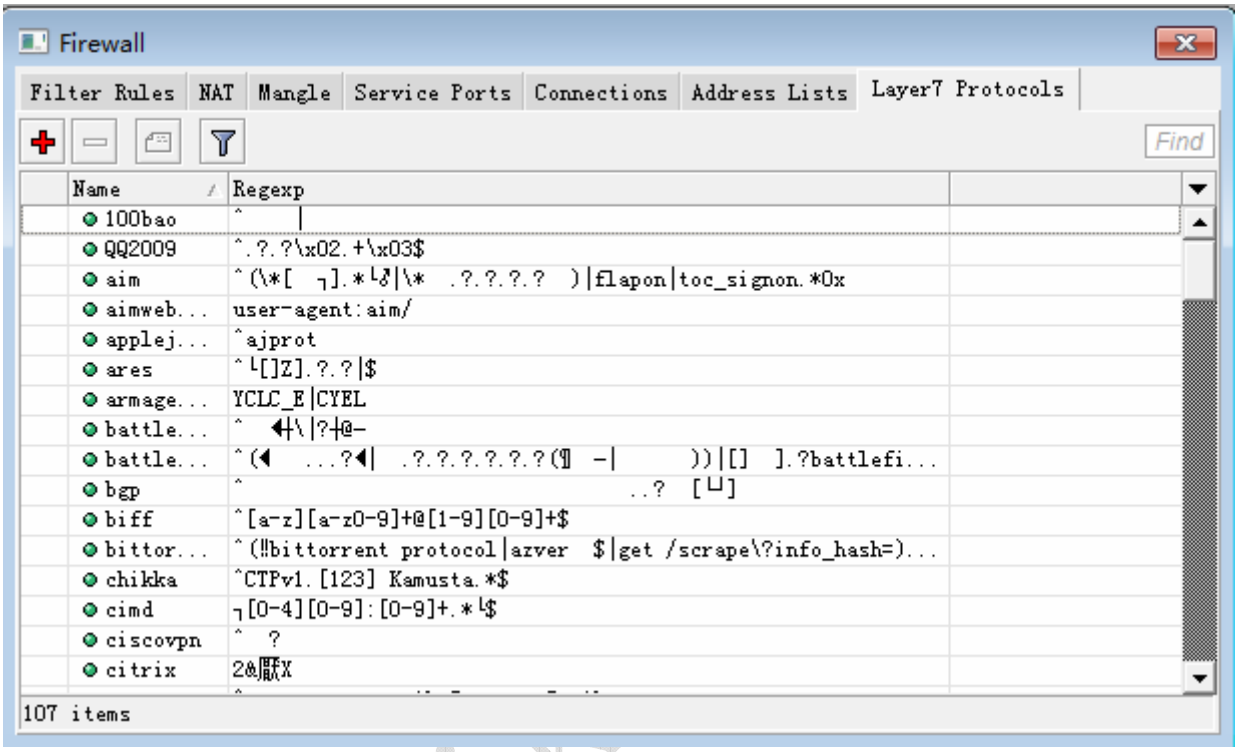


之后我们在命令行(Terminal)中导入 7 层协议脚本，用 import 17-protos.rsc 命令来导入脚本

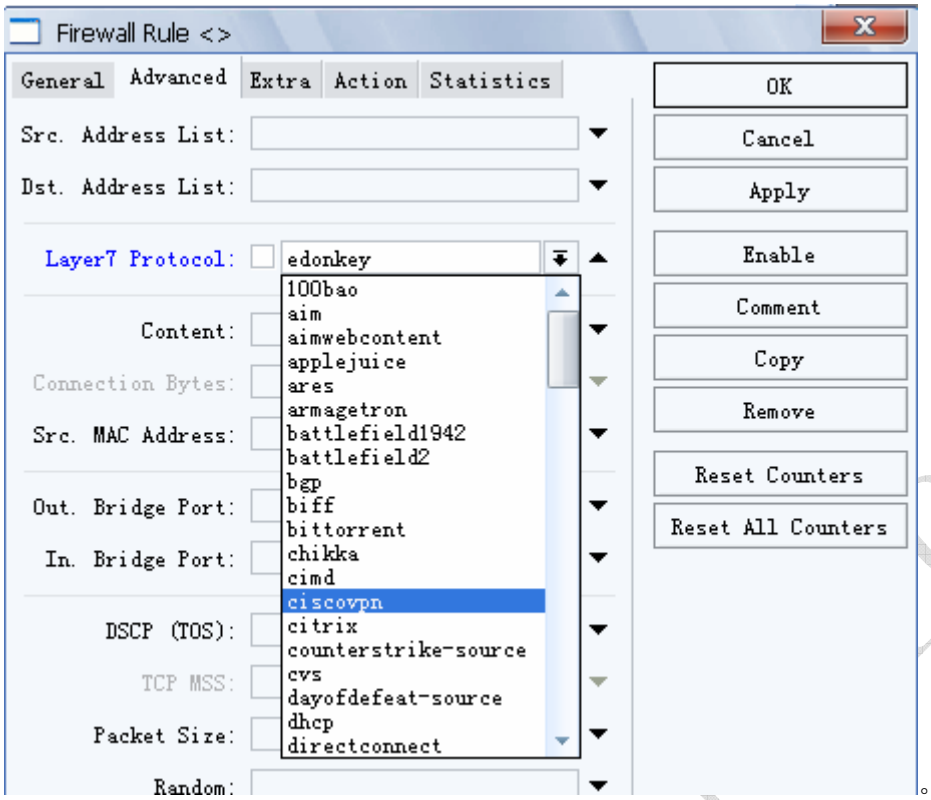
```
[admin@MikroTik] > import l7-protos.rsc
Opening script file l7-protos.rsc
Script file loaded and executed successfully
[admin@MikroTik] >
```

当系统提示 Script file loaded and executed successfully，说明脚本成功导入。

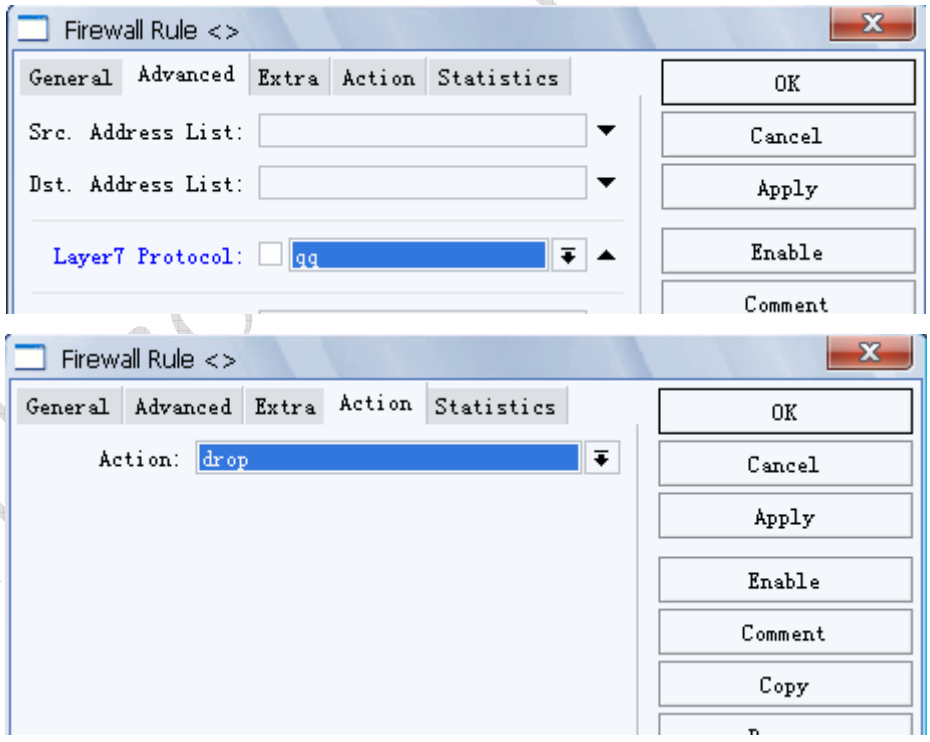
导入脚本后，我们可以在 Layer7 Protocols 中看到



导入后，我们就可以在 ip firewall 中通过 Layer7 Protocols 参数调用，并做相应的规则处理，下面是一个在防火墙得 Filter Rules 里面调用 L7 脚本



在这里我们通过禁止登陆 QQ 为例，在这里我们禁止所有用户无法登陆 QQ。添加一条规则后，进入 Advanced 中的 Layer7 Protocols 选项选择 qq，然后在 Action 中设置为 drop 丢弃。注意：L7 禁止 QQ 的规则设置好后，需要重启才能生效。



其他的操作也同以上设置类似，如果需要对 IP 地址或者 IP 段控制可以通过 src-address 或者 dst-address 进行设置。

9.5 DMZ 配置事例

DMZ 是英文“demilitarized zone”的缩写，中文名称为“隔离区”，也称“非军事化区”。它是为了解决安装防火墙后外部网络不能访问内部网络服务器的问题，而设立的一个非安全系统与安全系统之间的缓冲区，这个缓冲区位于企业内部网络和外部网络之间的小网络区域内，在这个小网络区域内可以放置一些必须公开的服务器设施，如企业 Web 服务器、FTP 服务器和论坛等。另一方面，通过这样一个 DMZ 区域，更加有效地保护了内部网络，因为这种网络部署，比起一般的防火墙方案，对攻击者来说又多了一道关卡。

路由器一般需要 3 张网卡（Public 公网，Local 本地网络，DMZ-Zone 非军事区）：

```
[admin@gateway] interface> print
Flags: X - disabled, D - dynamic, R - running
 #    NAME                      TYPE           RX-RATE    TX-RATE    MTU
 0  R Public                    ether           0          0         1500
 1  R Local                     ether           0          0         1500
 2  R DMZ-zone                   ether           0          0         1500
[admin@gateway] interface>
```

- 给相应的 Interface 添加对应的 IP 地址，如下：

```
[admin@gateway] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
#   ADDRESS          NETWORK        BROADCAST      INTERFACE
0   192.168.0.2/24    192.168.0.0    192.168.0.255   Public
 1   10.0.0.254/24     10.0.0.0       10.0.0.255        Local
 2   10.1.0.1/32       10.1.0.2       10.1.0.2          DMZ-zone
 3   192.168.0.3/24    192.168.0.0    192.168.0.255    Public
[admin@gateway] ip address>
```

- 添加静态默认路由到本地路由器上

```
[admin@MikroTik] ip route> print
Flags: X - disabled, I - invalid, D - dynamic, J - rejected,
C - connect, S - static, r - rip, o - ospf, b - bgp
 #    DST-ADDRESS       G GATEWAY        DISTANCE INTERFACE
 0  S 0.0.0.0/0         r 10.0.0.254      1       ether1
 1 DC 10.0.0.0/24       r 0.0.0.0         0       ether1
[admin@MikroTik] ip route>
```

- **配置 DMZ 服务器的 IP 地址为 IP 地址 10.1.0.2，网络地址段 10.1.0.1/24，以及网关 10.1.0.1**
- **配置能从因特网访问 DMZ 服务的 dst-nat 规则，将地址 192.168.0.3 配置给 DMZ 服务器：**

```
[admin@gateway] ip firewall nat> add chain=dst-nat action=dst-nat \
\... dst-address=192.168.0.3 to-dst-address=10.1.0.2
[admin@gateway] ip firewall dst-nat> print
Flags: X - disabled, I - invalid, D – dynamic
0  Chain=dst-nat dst-address=192.168.0.3 action=dst-nat to-dst-address=10.1.0.2
[admin@gateway] ip firewall nat>
```

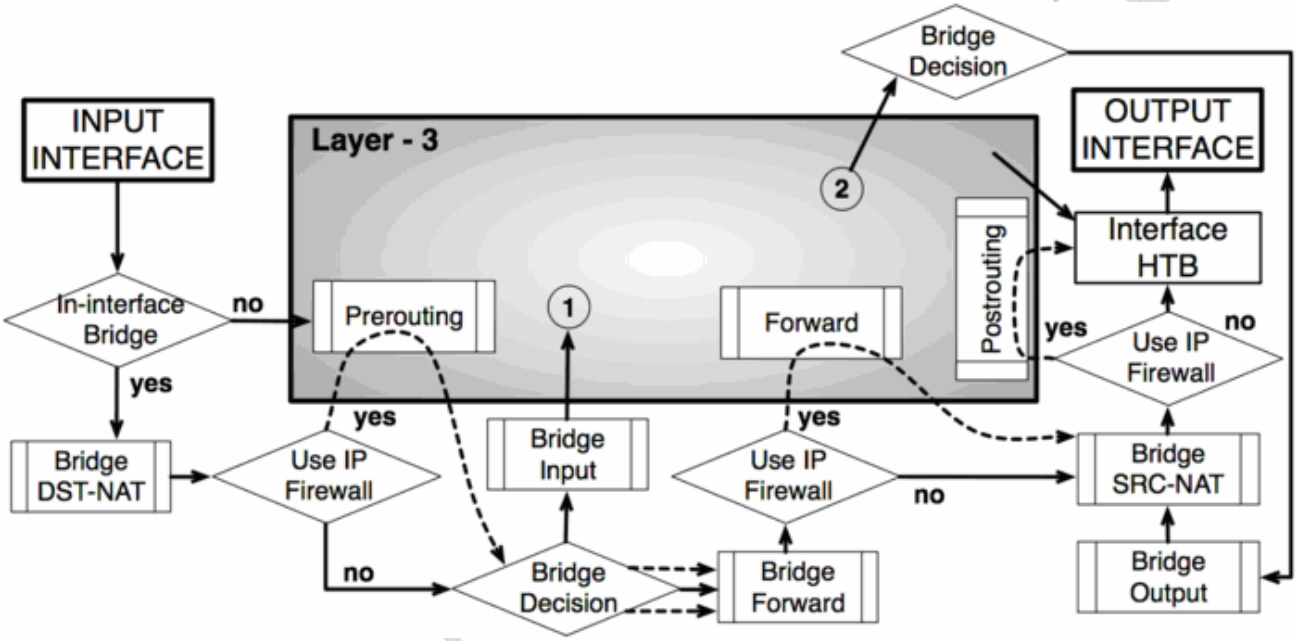
第十章 RouterOS 数据流(Packet Flow)

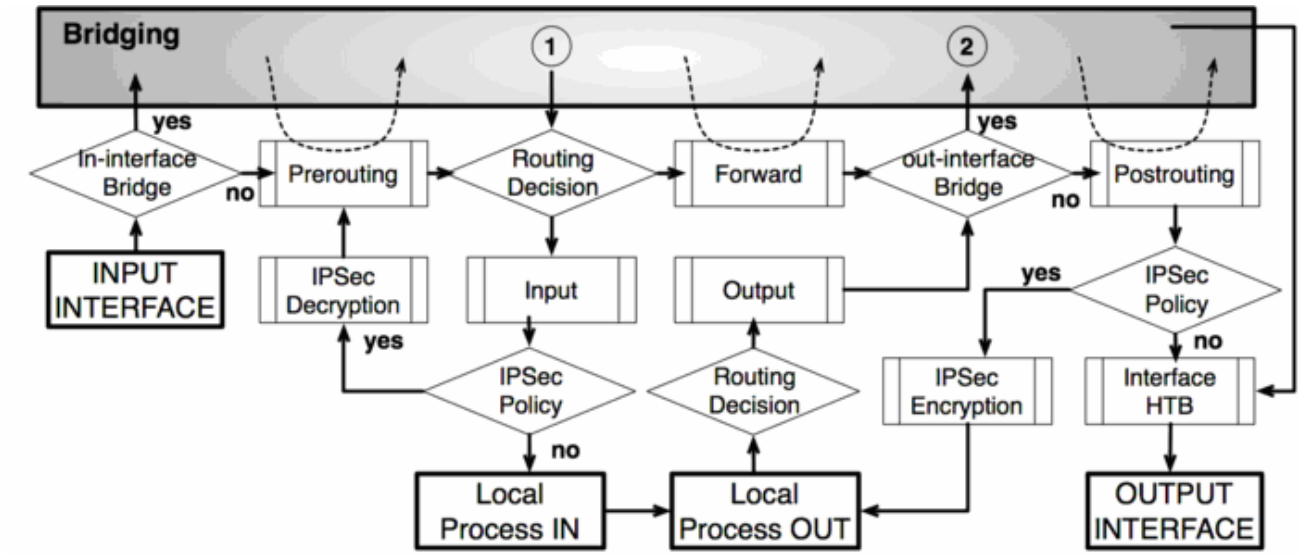
了解 RouterOS 的核心就需要对 IP 数据流进入 RouterOS 是如何被处理的，这里将通过各种流程图来介绍 RouterOS 在处理 IP 数据流的过程

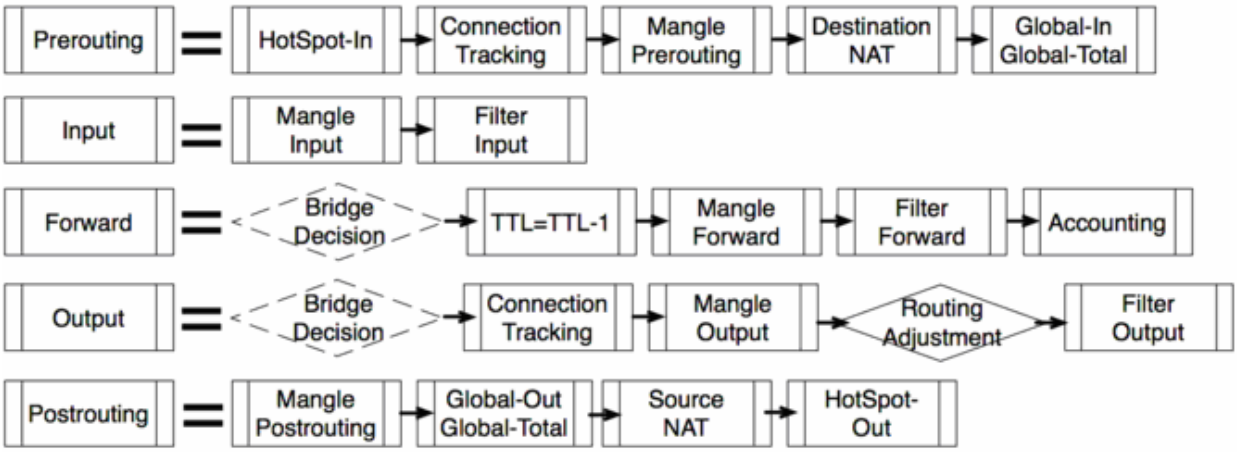
9.1 IP 数据流程图

下面是 RouterOS 3.0 以上版本数据流原理图，这里不可能将所有的原理放到一个图中，所以我们将原理图分解成 2 部分：

- **Bridging（桥接）或二层(MAC)** 路由的部分简化为一个“Layer-3”的框
- **Routing or Layer-3 (IP)** 桥接的部分简化为一个“Bridging”的框







INPUT INTERFACE - 数据包进入路由器的起点，不论什么样的接口（物理接口或虚拟接口）数据包都会从这里开始。

OUTPUT INTERFACE - 数据包离开路由器的终点，在之前运行的数据包确定被发送出路由器。

Local Process IN - 最终到达路由器自身的数据包。

Local Process OUT - 由路由器**自身发出**的数据包起点。

9.2 功能模块与结构

每一个功能模块在 RouterOS 中指定的目录下对应不同的功能，用户可以进入相应的目录配置属性。

Connection Tracking - **/ip firewall connection tracking**

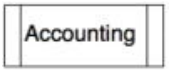
Filter Input | Filter Forward | Filter Output - **/ip firewall filter**

- **/ip firewall nat**

- **/ip firewall mangle**

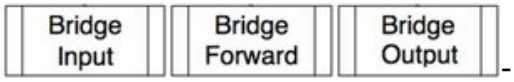
- **/queue simple 和 /queue tree**

- **/ip ipsec policy**



- **/ip accounting**

- **/interface bridge settings** 当其用桥接后，该功能才能生效，如果 Use IP Firewall 设置为 Yes，则数据将进入 Layer-3 层处理。



- **/interface bridge filter**

- **/interface bridge nat**

自动处理

- 检查输入接口是否为桥接数据（输入接口是一个端口到桥或者输入接口是桥）。

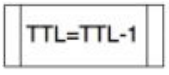
- 运行抓取传输 Hotspot 特征数据，否则从连接跟踪丢弃掉。

- 桥通过 MAC 地址列表查找数据包所匹配的目标 MAC 地址，当目标被找到，数据包将会被发送到相应的桥接口，如果没有匹配则会将数据包复制多份放送到所有的桥接端口。

- 这是一个变通方案，在实际桥属性判断前，允许使用“out-bridge-port”

- 路由器通过路由命令查找一个匹配数据包的目标 IP 地址。当查找到后，数据包将发送到对应的端口或者路由器本身。如果在该事件中没有找到匹配路径，数据包将会被丢弃。

- 这是一个变通的方案，允许在 mangle 中的 output 链表设置路由策略规则。



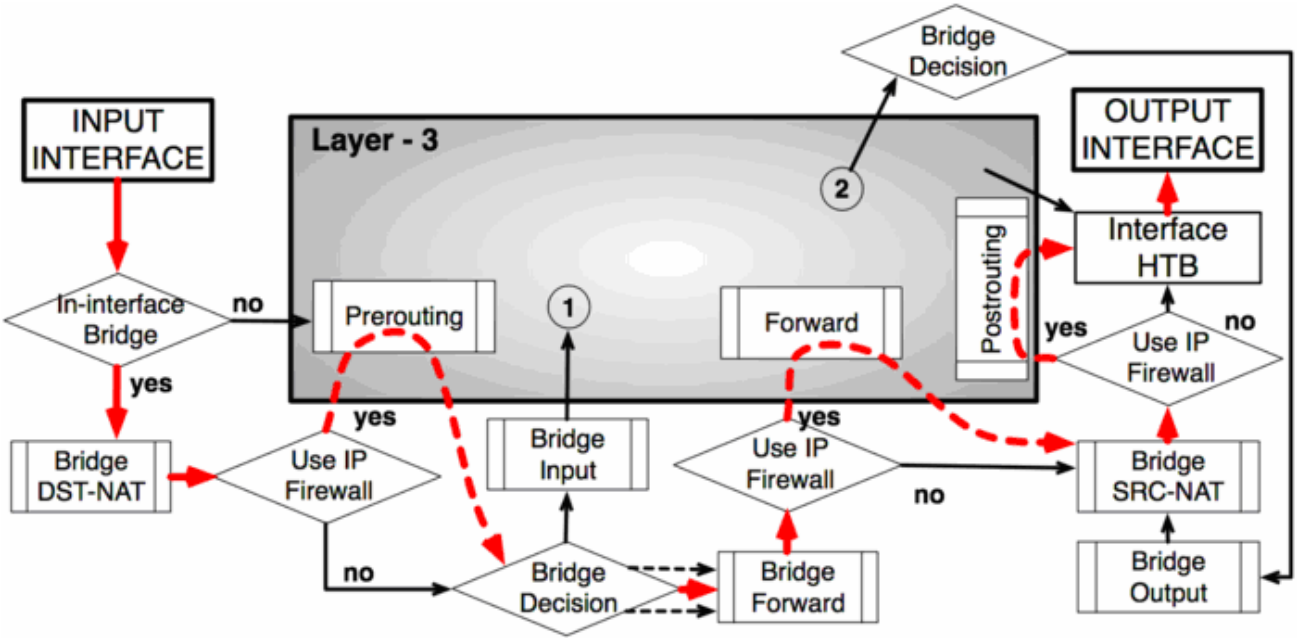
- 指明准确的地点，在什么时候路由包的 TTL 值被减少 1。如果 TTL 值变为 0 将会被丢弃掉。

- 自我处理 IPsec 判断和加密。

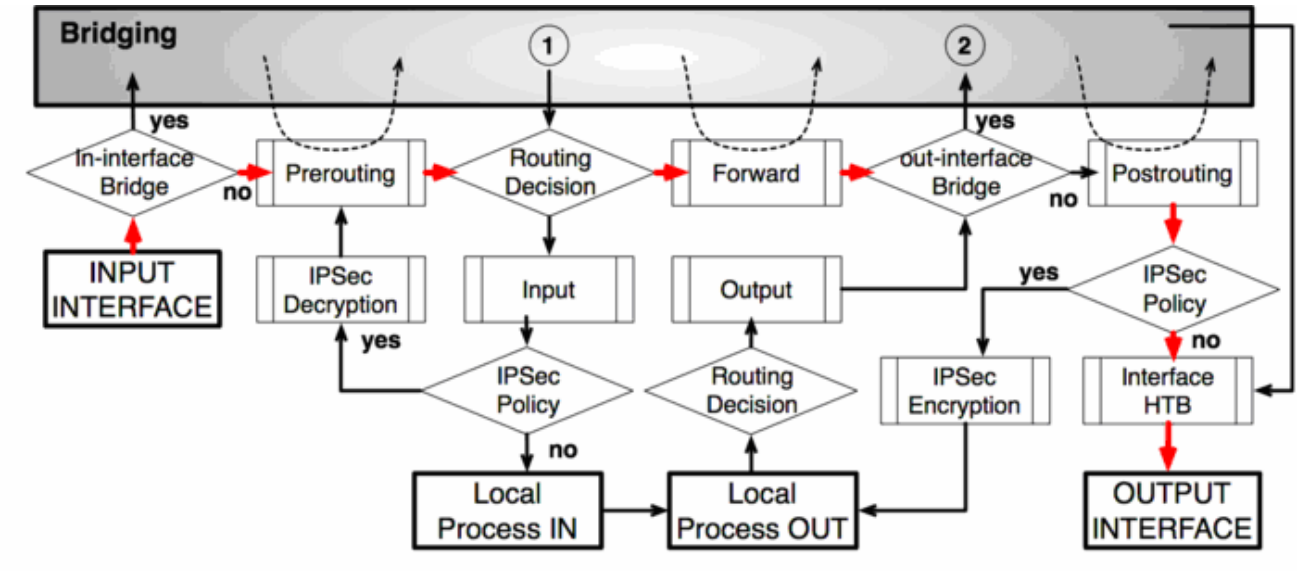
- 判断实际输出接口是否为桥接端口或判断输出接口是否为桥。

HotSpot-Out - 撤销所有通过 Hotspot-in 的数据包操作，并发送回客户端。

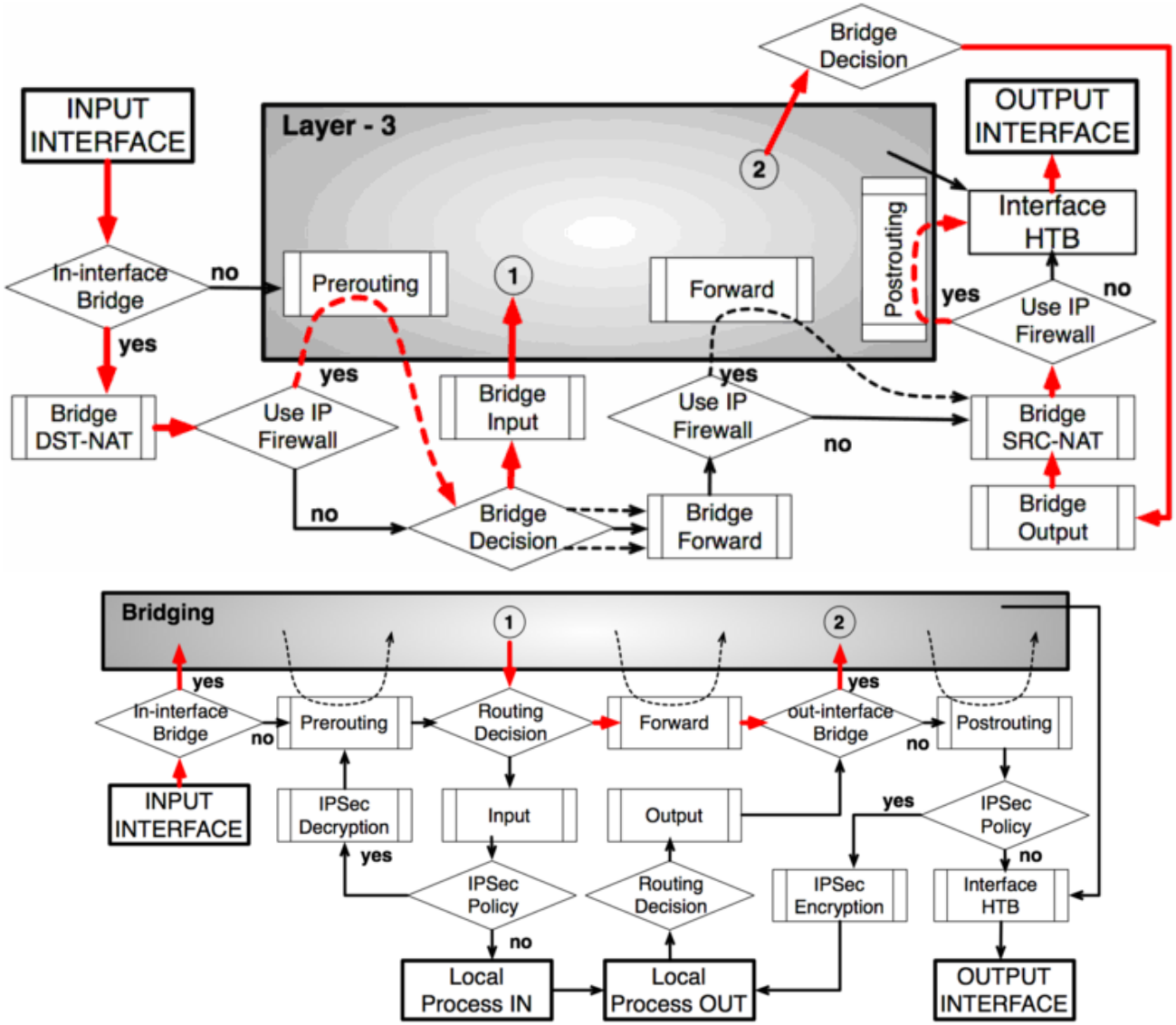
桥接设置 use-ip-firewall=yes



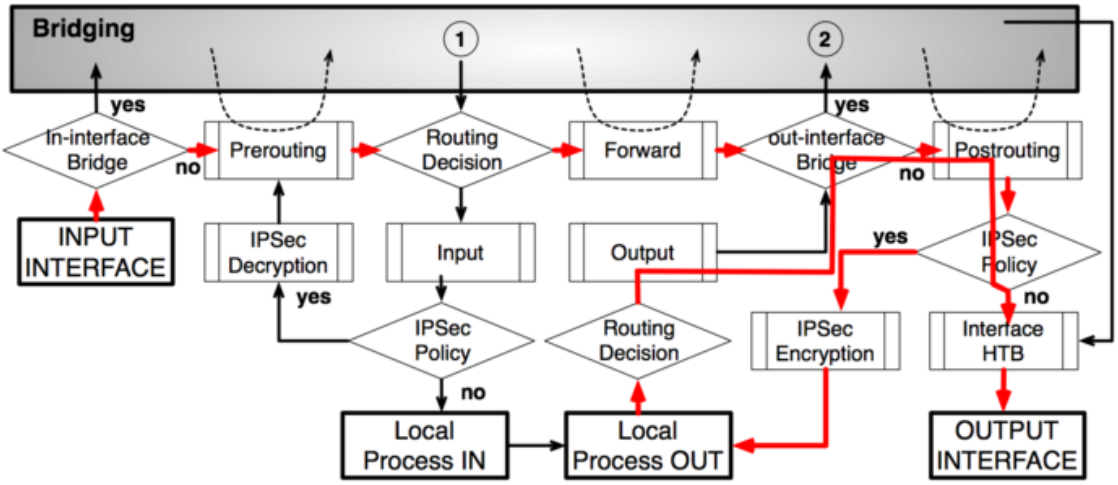
路由 – 从 Ethernet 到 Ethernet 接口



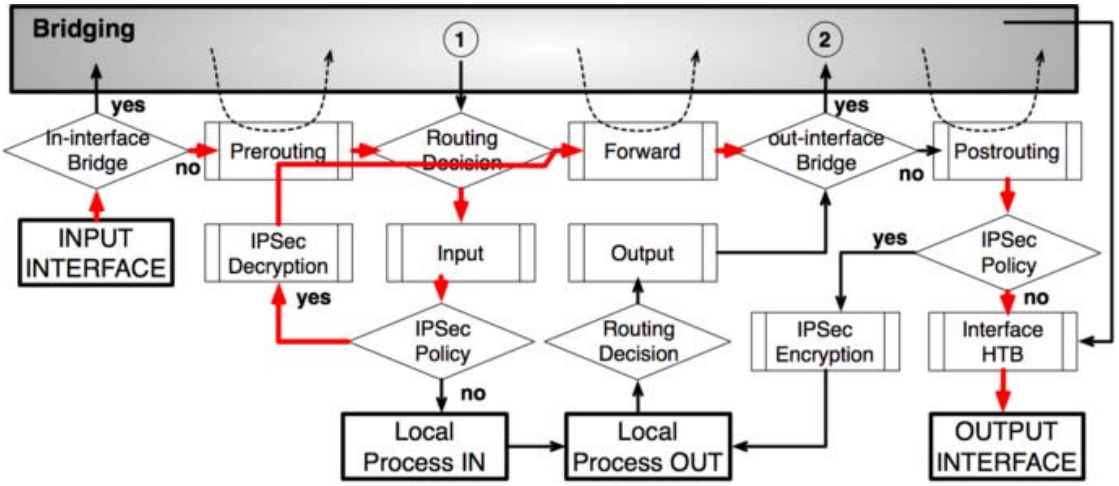
路由从一个桥接口道不同的桥接口



IPsec 编码



IPsec 解码



第十一章 带宽控制（Queue）

带宽控制是一套控制数据率分配，延迟易变性，及时转发(delivery)，可靠转发的机制。 MikroTik RouterOS 支持队列规则:

- **PFIFO** - 包先进先出
- **BFIFO** – 字节先进先出
- **SFQ** - 随机公平队列
- **RED** – 随机早先探测
- **PCQ** – 每次连接队列
- **HTB** – 等级令牌桶

规格

功能包要求: ***system***
等级要求: *Level1 (限 1 条规则)* ， *Level3*
操作路径: ***/queue***

11.1 Queue 机制

服务质量(QoS)即路由器应该优先考虑保证数据流的质量，并形成新的网络数据流。 QoS 并非是只是对流量的控制，它更多的是与提供优良品质的服务相关。以下是一些 RouterOS 带宽控制机制的特征:

- 对特定 IP 地址，子网，协议，端口以及其他参数限制数据率
- 限制 P2P 流量
- 优先考虑一些数据包流
- 为更快的 WEB 浏览使用队列脉冲串
- 对固定的时间间隔执行队列
- 在用户间平等的或者根据通道负担共享可用流量
- 队列应用在通过路由器真实接口的数据包上(比如:队列应用在向外的接口，像业务流)，或者三个添加的虚拟接口中的任何一个或几个(global-in， global-out， global-total)。

QoS 是通过掉包的方法工作的。被丢掉的包会被再次发送以防止丢弃了 TCP 协议，所以没必要担心会丢失 TCP 信息。用于描述网络应用的 QoS 等级的术语有:

- **Queuing discipline (qdisc)** - 一个保存并维护队列包的算法。它指定了向外的数据包(也就是说队列规则可以对包再排序)以及在没有空间的情况下哪些包需要丢弃。
- **CIR (Committed Information Rate)** - 约定好了的数据率。 即通信量速率，在不超过这个值的时候应该总是被转发
- **MIR (Maximal Information Rate)** - 路由器可以提供的最大数据率
- **Priority** - 流量将处理的重要性顺序。你可以设置优先级以便一些数据流可以在其他数据留之前被处理
- **Contention Ratio** - 定义的数据率在用户中共享的比率(当数据率分配给许多用户时)。正是用户的数量拥有应用于它的简单速度限制。例如:连接比率是 1:4，即分配的数据率将会在最多 4 个用户中共享。

数据包在从接口发送之前会用队列规则进行处理。默认地，队列规则在物理接口的**/queue interface** 设置(对于虚拟接口没有默认的规则)。一旦我们对物理接口添加了一个队列(在**/queue tree**)，在**/queue interface** 定义的默认队列，对于特定接口将被忽视。就是说，当一个包没有匹配任何过滤器时，它将被发送到带有最高优先权的接口。

调度机和成型机 qdiscs

我们按照对业务流的影响分类队列规则如下:

- **调度机(schedulers)** - 队列规则只根据它们的算法对数据包进行重新调度并丢弃在队列中不匹配的数据包。调度机队列规则包括: PFIFO， BFIFO， SFQ， PCQ， RED
- **成型机(shapers)** - 队列规则也履行限制规则，成型机有 PCQ 及 HTB。

虚拟接口

RouterOS 对实际接口增加了三个虚拟接口:

- **global-in** - 代表了所有普通的输入接口(INGRESS 队列)。请注意在数据包过滤前与 **global-in** 相关的队列应用到路由器接的数据流。 **global-in** 排序就是在 mangle 和 dst-nat 之后执行。
- **global-out** - 代表了所有普通的输出接口。附属于它的队列会在附属于特定接口的队列之前应用。
- **global-total** - 表了一个流经路由器的数据都能通过的虚拟接口。当把一个 qdisc 附属到 global-total 时，限制需要在两个方向起作用。例如，如果我们设置一个为 **total-max-limit** 256000 限制，我们将得到 upload+download=256kbps(最大值)

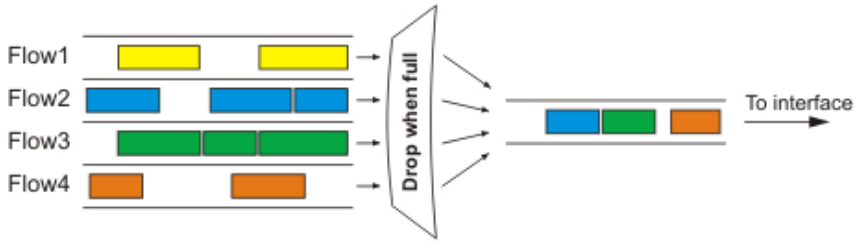
11.2 队列类型（Queue Type）

操作路径: ***/queue type***

在这个子目录你可以创建自己的客户队列类型。之后，将可以在**/queue tree**， **/queue simple** 或 **/queue interface** 使用了

PFIFO 及 BFIFO

这些队列规则是基于先进先出算法的(FIFO：First-In First-Out)。PFIFO 和 BFIFO 的区别在于一个是以数据包为单位衡量的，而另一个是以字节为单位。其中只有一个叫做 **pfifo-limit** (**bfifo-limit**)的参数，它是用来定义一个 FIFO 队列可以容纳多少数据的。每一个不能排队（如果队列满了）的包都要被丢弃，队列长度过大会增加执行时间。

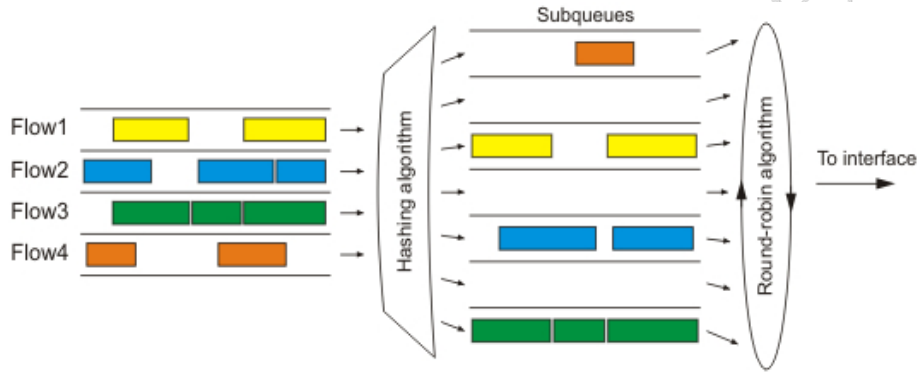


如果你的连接不拥塞的话，建议使用 FIFO 队列规则。

SFQ

随机公平排序（SFQ）不会一开始就对流量限制。它的主旨是当你的连接完全满的时候均衡业务流（TCP 会话或者 UDP 流）。

SFQ 的公平性是由散列法和 round-robin 算法保证的。散列算法把会话流分成一个有限数量的子队列。在 **sfq-perturb** 时间之后散列算法改变并划分会话流为其他子队列。Round-robin 算法把从每个子队列的 **pcq-allot** 字节按照顺序出队列。

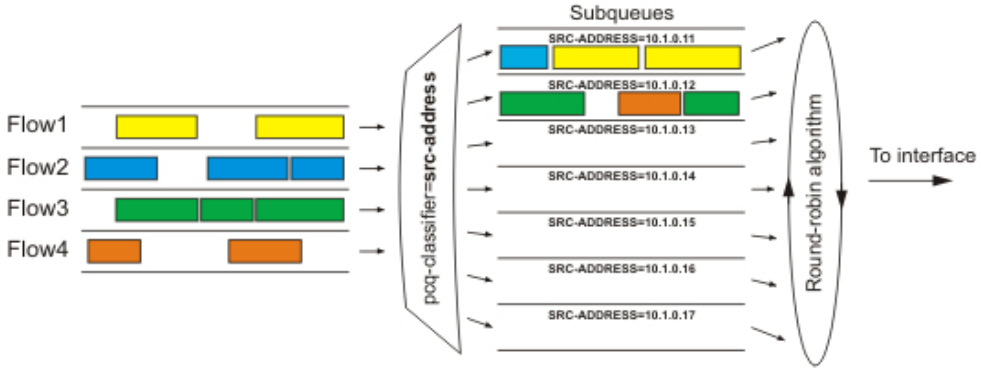


整个 SFQ 队列可以容纳 128 个数据包并且对这些包有 1024 个子队列可用。对拥挤的连接使用 SFQ 可以保证一些连接不至于空等待（starve）。

PCQ

为了解决 SFQ 的不完美，每次连接排序 Per Connection Queuing (PCQ)便产生了。它是唯一一种能限流的无等级排序类型。它是一种去掉了随机特性的进化版 SFQ。PCQ 也会根据 **pcq-classifier** 参数产生子队列。每个子队列都有一个 **pcq-rate** 的数据率限制和 **pcq-limit** 大小的数据包。PCQ 队列的总大小不能大于 **pcq-total-limit** 包。

以下实例说明了 PCQ 对数据包的用法，以它们的源地址分类。

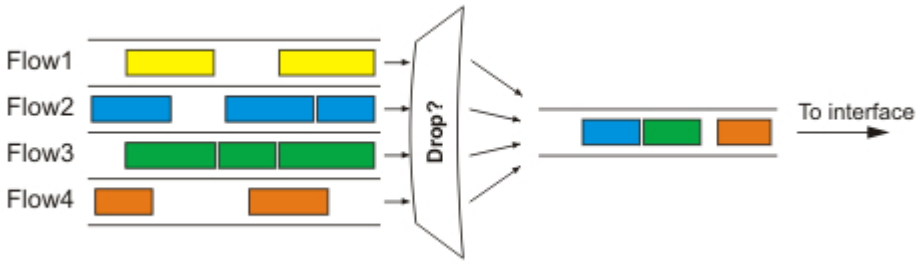


如果你以 **src-address** 对包分类那么所有带有不同源 IP 地址的包将被集合在不同的子队列中。现在你可以使用 **pcq-rate** 参数对每一个子队列进行限制或均衡。或许最重要的部分是决定我们到底应该把这个队列附属到哪个接口上。如果我们把它依附在本地接口上，那么所有来自公网接口的数据流都将以 src-address（很可能这不是我们想要的）地址分组；相反地如果我们把它依附到公共接口，所有来自我们客户的数据都会以 src-address 分组——于是我们可以很容易的限制或者均衡客户的上载。

用 **pcq-classifier** 分类后为了在子队列中均衡速率，设置 **pcq-rate** 为 **0** 几乎不用管理，PCQ 也可以用来对多用户动态均衡或者形成流量，

RED

随机早先探测（RED）是一种通过控制平均队列长度避免网络拥塞的排序机制。当平均队列长度达到 **red-min-threshold** 时，RED 随机选择该丢弃哪个包。当平均队列长度变长时，堆砌多少包数的可能性会增加。如果平均队列长度达到 **red-max-threshold**，则丢弃该包。尽管如此，也存在真实队列长度（非平均的）远大于 **red-max-threshold** 时，丢弃所有超过 **red-limit** 的数据包的情况。



注意：RED 应用在高数据率的拥挤的连接上，它在 TCP 协议上工作的很好，但在 UDP 上就没那么理想了。

属性描述

bfifo-limit (整数; 默认: **15000**) - BFIFO 队列可以容纳的最大字节数
kind (bfifo | pcq | pfifo | red | sfq) – 选择队列控制类型
- ***bfifo*** - *字节先进先出*
- ***pcq*** – *每次连接队列*
- ***pfifo*** – *数据包先进先出*
- ***red*** – *随机早先探测*
- ***sfq*** - *随机公平队列*

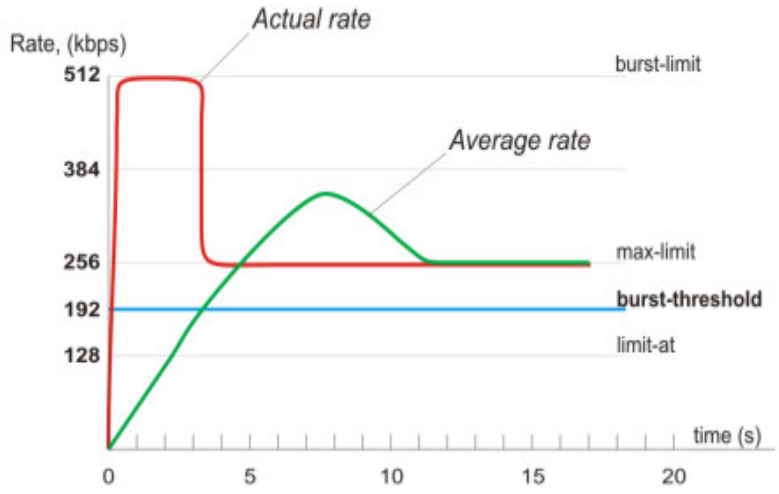
name (名称) - 队列类型相关名称
pcq-classifier (dst-address | dst-port | src-address | src-port; 默认: **""**) - PCQ 对其子队列进行分组的分类器。可以同时被数个分类器使用。例如：src-address，src-port 可使用不同源地址和源端口把所有包分为独立的子队列
pcq-limit (*整数*; 默认: **50**) - 可以容纳一个单个 PCQ 子队列的包的数目
pcq-rate (*整数*; 默认: **0**) - 对每个子队列允许的最大数据率。 **0** 值指的是没有任何限制
pcq-total-limit (*整数*; 默认: **2000**) - 可以容纳整个 PCQ 队列的包的数目
pfifo-limit (*整数*) - PFIFP 队列可以容纳包的最大数目
red-avg-packet (整数; 默认: **1000**) - 被 RED 用来对平均队列长度计算
red-burst (*整数*) - 用来决定平均队列长度被真实队列长度影响的快慢的字节值。较长的值将减慢 RED 的计算速度——较长的脉冲串也是允许的
red-limit (*整数*) - 以字节计算。如果真实队列长度（非平均值）超过了这个值那么所有大于这个值的包都将被丢弃。
red-max-threshold (整数) - 以字节计算。数据包标记概率最高的平均队列长度
red-min-threshold (*整数*) - 当平均 RED 队列长度达到这个值时，数据包标记才有可能

sfq-allot (*整数*; 默认: **1514**) - 在一个 round-robin 循环中从子队列发出的字节数
sfq-perturb (*整数*; 默认: **5**) - 以秒计时。指定改变 SFQ 的散列算法的频率

Bursts 脉冲串

脉冲串用来在一段很短的时间允许更高数据率。每 1/16 **burst-time** 时间，路由器都会计算每个类在上一个 **burst-time** 时间的平均数据率。如果这个平均数据率小于 **burst-threshold**，脉冲串就会被启用且实际数据率达到 **burst-limit** bps，否则实际数据率将跌至 **max-limit** 或 **limit-at**。

让我们考虑如果我们有个 **max-limit**=256000， **burst-time**=8， **burst-threshold**=192000 以及 **burst-limit**=512000 的设置情况。当一个用户通过 HTTP 下载一个文件，我们可以观察到这样的现象：



在最开始的 8 秒中平均数据率是 0bps 因为在应用队列规则前没有流量通过。由于这个平均数据率小与 **burst-threshold** (192kbps)，所以脉冲串会被使用。在第一秒之后，平均数据率为(0+0+0+0+0+0+0+512)/8=64kbps，低于 **burst-threshold**。在第二秒后，平均数据率为(0+0+0+0+0+0+512+512)/8=128kbps。在第三秒之后达到临界点此时平均数据率变得大于 **burst-threshold**。这个时候脉冲串将被禁用且当前数据率降至 **max-limit** (256kbps)。

11.3 Simple Queue 简单队列

限制数据率的 IP 地址和子网的最简单方法就是使用简单队列。你也可以使用简单队列建立高级 QoS 应用：

- P2P 流量队列
- 在选定时间间隔执行队列规则
- FIFO 优先级
- 从 */ip firewall mangle* 使用多重包标记
- 形成双向流量（对上传和下载的带宽限制）

属性描述

burst-limit (*整数/整数*) - 当脉冲串以 in/out（目标上传/下载）形式激活时可以达到的最大数据率
burst-threshold (*整数/整数*) - 用于计算是否允许脉冲串。如果上一次脉冲时间的平均数据率低于 *burst-threshold* 则实际数据率可能达到 *burst-limit*。以 in/out（目标上传/下载）的形式。
burst-time (*整数/整数*) - 用于计算平均数据率。以 in/out（目标上传/下载）的形式。
direction (非同时上传和下载) - 流量控制方向
none – *队列停止有效的工作*
both – *队列同时限制目标上行和目标下行*

upload – *队列仅限制目标上行，下行的数据不会被限制*

download - *队列仅限制目标下行，上行的数据不会被限制*

dst-address (*IP 地址/子网掩码*) - 要匹配的目标地址

dst-netmask (*子网掩码*) - *dst-address* 的掩码

interface (*文本*) - 队列应用的对象端口。

limit-at (*整数/整数*) – 该队列以 in/out（目标上传/下载）的形式约定的数据率

max-limit (*整数/整数*) - 在有足够带宽情况下可以达到的数据率，以 in/out（目标上传/下载）的形式。

name (*文本*) - 队列的描述性名称

p2p (any | all-p2p | bit-torrent | blubster | direct-connect | edonkey | fasttrack | gnutella | soulseek | winmx) – 控制匹配的 P2P 流量类型

all-p2p – *匹配的所有 P2P 传输*

any – *匹配任何数据包（即不会检查该属性）*

packet-marks (*名称*; 默认: **""**) - **/ip firewall** 中的数据包标记

mangle 更多数据包标记使用逗号(",")搁开。

parent (*名称*) - 父队列在等级制度中的名称。只能是其他简单队列

priority (*整数*: 1..8) -队列的优先级。 1 是最高级的，8 是最低的

queue (*名称/名称*; 默认: **default/default**) - 以 in/out（目标上传/下载）的形式来自**/queue type** 的队列名称

target-addresses (*IP 地址/子网掩码*) - 限制目标 IP 地址（源地址）。使用多地址用逗号分搁开

time (*时间*, sat | fri | thu | wed | tue | mon | sun{+}; 默认: **""**) - 限制队列在一个特定时间段的影响

total-burst-limit (*整数*) - *global-total* 队列的脉冲串限制

total-burst-threshold (*整数*) - *global-total* 队列的脉冲串门限

total-burst-time (*时间*) - *global-total* 队列脉冲串时间

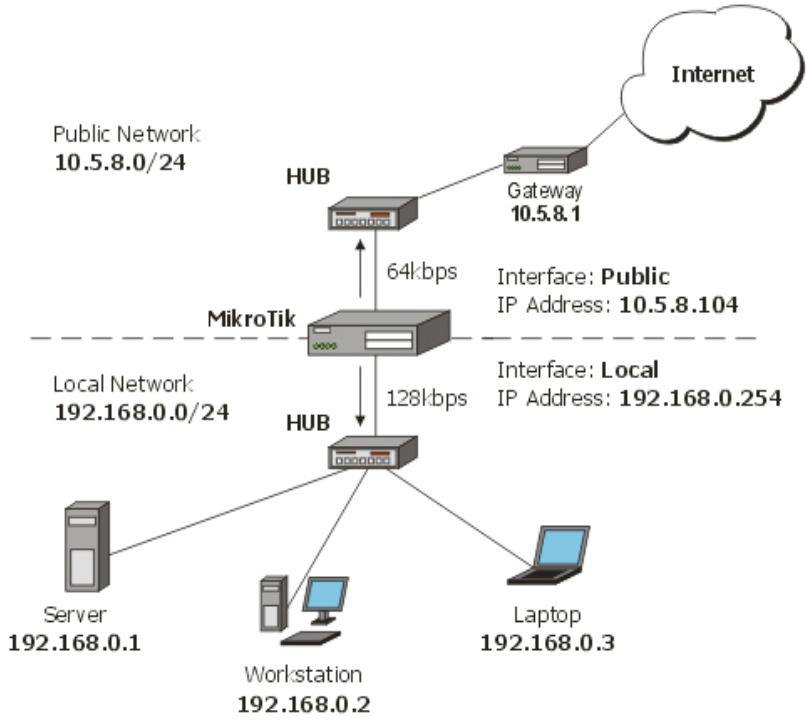
total-limit-at (*整数*) - 限制累计的上传和下载为 *total-limit-at* bps

total-max-limit (*整数*) - *global-total* 队列的限制上限 (限制累计的上传和下载为 *total-max-limit* bps)

total-queue (*名称*) - **global-total** 队列的队列规则

应用举例

下面假设我们想要对网络 192.168.0.0/24 流量限制为：下行 1Mb 上行 512kb，这里我们需要让服务器 192.168.0.1 不受流量控制。网络的基本设置如图：



这里我们使用（simple queue）简单队列，首先我们配置 RouterOS 的 IP 地址、网关和 NAT 等基本网络参数：

```
 [admin@MikroTik] ip address> print
Flags: X – disabled, I – invalid, D - dynamic
 #   ADDRESS          NETWORK        BROADCAST      INTERFACE
 0   192.168.0.254/24    192.168.0.0    192.168.0.255   Local
 1   10.5.8.104/24     10.5.8.0       10.5.8.255     Public
[admin@MikroTik] ip address>
```

路由配置：

```
[admin@MikroTik] ip route> print
Flags: X – disabled, A – active, D – dynamic,
C – connect, S – static, r – rip, b – bgp, o - ospf
 #    DST-ADDRESS       G GATEWAY        DISTANCE INTERFACE
 0 ADC 10.5.8.0/24                              Public
 1 ADC 192.168.0.0/24                            Local
 2 A S 0.0.0.0/0         r 10.5.8.1              Public
[admin@MikroTik] ip route>
```

最后不要忘记在 ip firewall nat 中配置 src-nat 的伪装或 nat，做地址转换操作。

为网络 192.168.0.0/24 的所有客户端添加一个限制下载流量为 2Mb 上传流量 1Mb 的简单队列规则。

```
[admin@MikroTik] queue simple> add name=Limit-Local target-address=192.168.0.0/24
max-limit=1000000/2000000
[admin@MikroTik] queue simple> print
Flags: X - disabled, I - invalid, D - dynamic
 0   name="Limit-Local" target-addresses=192.168.0.0/24 dst-address=0.0.0.0/0
     parent=none priority=8 queue=default/default  limit-at=0/0 max-limit=1000000/2000000
total-queue=default
[admin@MikroTik] queue simple>
```

max-limit 限制了最大可用带宽，从客户的角度看，参数 **target-addresses** 定义限制带宽的目标网络或者主机（也可以用逗号分隔开网络段或主机地址）。

这里不想让服务器受到我们添加上面规则的任何流量限制，我们可以通过添加一个没有任何限制的规则（**max-limit=0/0** 代表没有任何限制）并把它移到列表的顶部：

```
[admin@MikroTik] queue simple> add name=Server target-addresses=192.168.0.1/32
[admin@MikroTik] queue simple> print
Flags: X - disabled, I - invalid, D - dynamic
 0   name="Limit-Local" target-addresses=192.168.0.0/24 dst-address=0.0.0.0/0
     parent=none priority=8 queue=default/default limit-at=0/0 max-limit=65536/131072
total-queue=default

 1   name="Server" target-addresses=192.168.0.1/32 dst-address=0.0.0.0/0
     parent=none priority=8 queue=default/default limit-at=0/0 max-limit=0/0
total-queue=default
[admin@MikroTik] queue simple> move 1 0
```

```
[admin@MikroTik] queue simple> print
Flags: X - disabled, I - invalid, D - dynamic
 0   name="Server" target-addresses=192.168.0.1/32 dst-address=0.0.0.0/0
     parent=none priority=8 queue=default/default
     limit-at=0/0 max-limit=0/0 total-queue=default

 1   name="Limit-Local" target-addresses=192.168.0.0/24 dst-address=0.0.0.0/0
     parent=none priority=8 queue=default/default
     limit-at=0/0 max-limit=65536/131072 total-queue=default
[admin@MikroTik] queue simple>
```

11.4 HTB 令牌桶介绍

HTB 等级令牌桶允许创建一个等级队列结构，并确定队列之间的关系，就像“父亲与儿子”或“兄弟之间”。

一旦队列添加了一个 **Child**（子队列）将会变为 **inner**(内部队列)，所有向下没有 **Children**（子队列）称为 **Leaf** 队列（叶队列），内部队列仅负责传输的分配,所有 Leaf 队列对符合的数据进行处理。 在 RouterOS 必须指定 **Parent**(父级)选项并指定一个队列为子队列。

双重限制

每个队列在 HTB 有 2 个速率限制：

- **CIR**（约定信息速率 Committed Information Rate） - （在 RouterOS 中的参数为 **limit-at**）最坏的情况下，无论如何都会将得到给定的的 CIR 传输量(假设我们能发送那么多的数据量)
- **MIR**（最大信息速率 Maximal Information Rate） - （在 RouterOS 中的参数为 **max-limit**）最好的情况下，如果父级有剩余带宽，将获得该速率值

换句话说，首先 Limit-at（CIR）都会被满足，仅当子队列尝试借调必要的数据传输从他们的父级，以达到最大的带宽 **max-limit** (**MIR**).

注: 无论如何 **CIR** 都将会被分配到符合队列的带宽（即使父级的 max-limit 满载），那就是为什么，确保最佳的使用双重限制功能，我们建议坚持这些规则：

- **CIR** 约定速率之和，即所有子级速率必须小于或等于可获得父级传输量。

CIR(parent)* ≥ CIR(child1) +...+ CIR(childN)
***如果父级与主父级可以设置为 CIR(parent)=MIR(parent)**

- 任何子级的最大速率必须小于或者等于父级的最大速率

MIR (parent) ≥ MIR(child1) & MIR (parent) ≥ MIR(child2) & ... & MIR (parent) ≥ MIR(childN)

在 winbox 中队列的颜色变化：

- 0% - 50% 使用情况 - 绿色
- 51% - 75% 使用情况 - 黄色
- 76% - 100% 使用情况 - 红色

优先级

这里已经知道，所有队列的 **limit-at (CIR)**都有可能将会被耗尽，优先级则主要负责分配父级队列剩余的带宽给 **Child**（子队列）达到 **max-limit**。队列高的优先级最优先达到 **max-limit**，优先级低的则不会。8 是最低优先级，1 则最高。

注意，优先级工作环境：

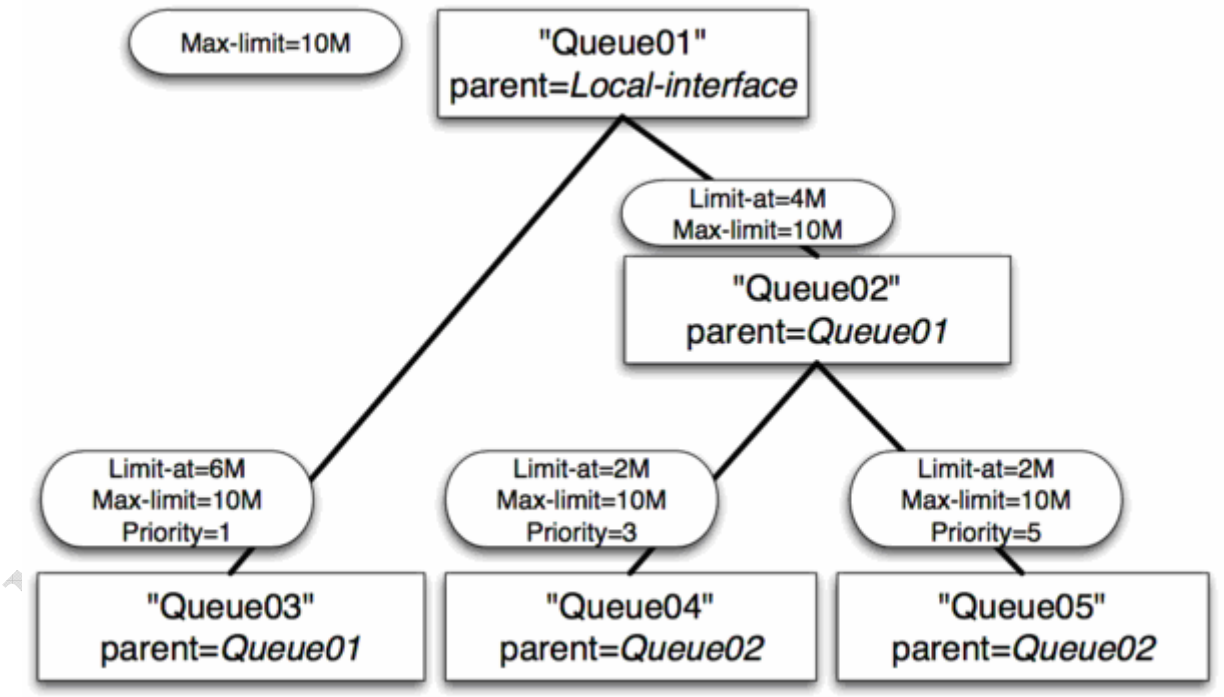
- 对于 **leaf** 叶队列 - 优先级对于 **inner**（内部队列）没有任何意义, 即 **inner** 内部队列与 **leaf**（叶队列）的优先级不可比较
- 如果 **max-limit** 被设定（非 0）

下面这部分我们将分析 **HTB** 的操作，将演示一个 HTB 结构并将涵盖可能出现的所有情况和功能，我们的 HTB 结构由下面 5 个队列构成：

- **Queue01** 内部队列有 2 个子级 - **Queue02** 和 **Queue03**
- **Queue02** 内部队列有 2 个子级 - **Queue04** 和 **Queue05**
- **Queue03** 叶队列
- **Queue04** 叶队列
- **Queue05** 叶队列

Queue03，Queue04 和 Queue05 的需要 10Mbps，我们接口处理能力在 10Mbps 的流量

事例 1: 普通事例

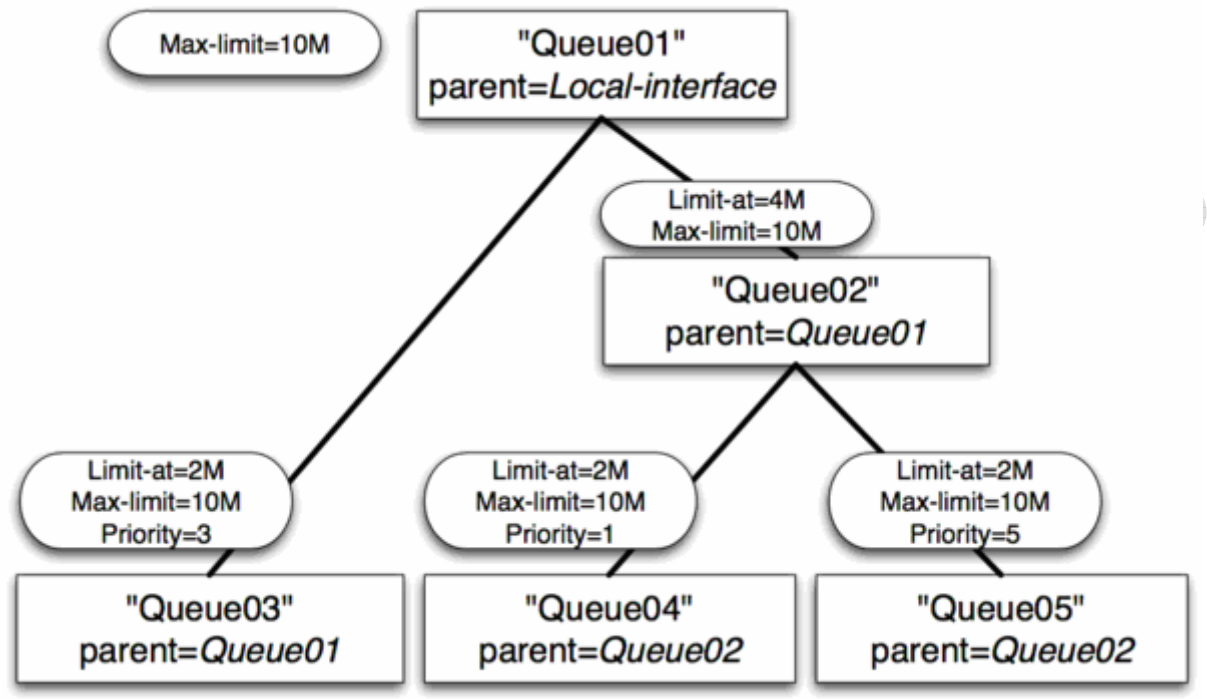


- **Queue01 limit-at=0Mbps max-limit=10Mbps**
- **Queue02 limit-at=4Mbps max-limit=10Mbps**
- **Queue03 limit-at=6Mbps max-limit=10Mbps priority=1**
- **Queue04 limit-at=2Mbps max-limit=10Mbps priority=3**
- **Queue05 limit-at=2Mbps max-limit=10Mbps priority=5**

事例 1 结果：

- **Queue03** 得到 **6Mbps**
- **Queue04** 得到 **2Mbps**
- **Queue05** 得到 **2Mbps**
- **结论：HTB** 建立在一种方式上，通过满足所有的 **limit-at**，主队列已没有带宽进行分发。

事例 2：max-limit 事例

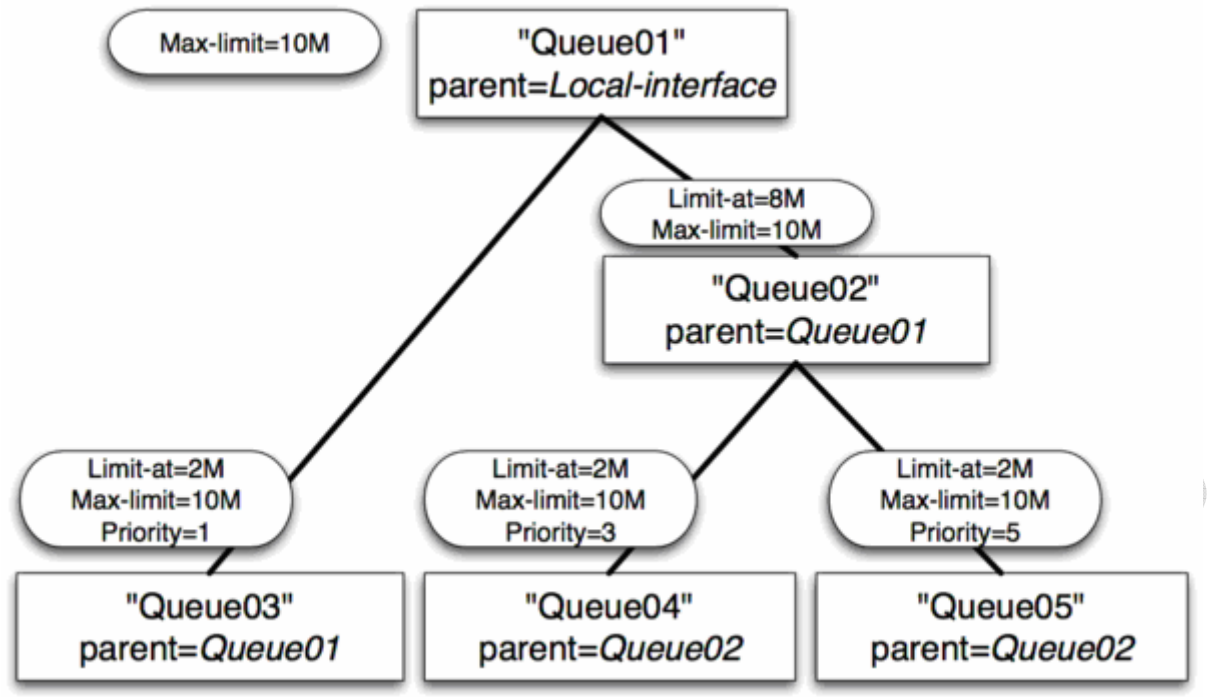


- **Queue01 limit-at=0Mbps max-limit=10Mbps**
- **Queue02 limit-at=4Mbps max-limit=10Mbps**
- **Queue03 limit-at=2Mbps max-limit=10Mbps priority=3**
- **Queue04 limit-at=2Mbps max-limit=10Mbps priority=1**
- **Queue05 limit-at=2Mbps max-limit=10Mbps priority=5**

事例 2 结果

- **Queue03** 得到 **2Mbps**
- **Queue04** 得到 **6Mbps**
- **Queue05** 得到 **2Mbps**
- **结论：** 在满足所有的 **limit-at** 后，**HTB** 将把剩余的带宽分配给优先级高的队列。

事例 3: inner 队列 limit-at



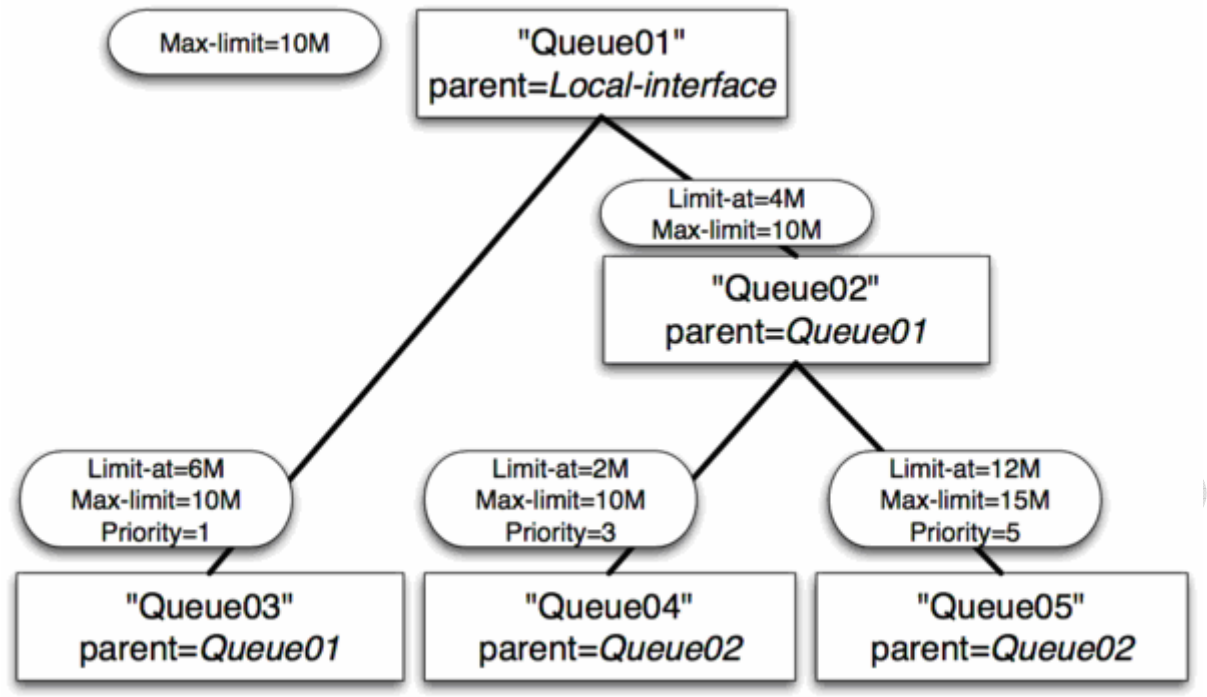
- **Queue01 limit-at=0Mbps max-limit=10Mbps**
- **Queue02 limit-at=8Mbps max-limit=10Mbps**
- **Queue03 limit-at=2Mbps max-limit=10Mbps priority=1**
- **Queue04 limit-at=2Mbps max-limit=10Mbps priority=3**
- **Queue05 limit-at=2Mbps max-limit=10Mbps priority=5**

事例 3 结果

- **Queue03** 得到 **2Mbps**
- **Queue04** 得到 **6Mbps**
- **Queue05** 得到 **2Mbps**

- **结论：**在满足所有的 **limit-at** 后，**HTB** 将分配剩余带宽给优先级高的，但在这个事例中，内部对列 **Queue02** 指定了 **Limit-at**， 这样他会保留 **8Mbps** 的流量给 **Queue04** 和 **Queue05**，**Queue04** 有更高的优先级，那就是为什么会得到更高的带宽。

事例 4：leaf 队列的 Limit-at



- **Queue01 limit-at=0Mbps max-limit=10Mbps**
- **Queue02 limit-at=4Mbps max-limit=10Mbps**
- **Queue03 limit-at=6Mbps max-limit=10Mbps priority=1**
- **Queue04 limit-at=2Mbps max-limit=10Mbps priority=3**
- **Queue05 limit-at=12Mbps max-limit=15Mbps priority=5**

事例 4 结果

- **Queue03** 得到 **3Mbps**
- **Queue04** 得到 **1Mbps**
- **Queue05** 得到 **6Mbps**

- 结论:为了满足所有的 Limit-at,HTB 被强迫分配 20Mbps,Queue03 为 6Mbps ,Queue04 为 2Mbps,Queue05 为 12Mbps，但我们的接口只能处理 10Mbps，因此接口队列通常 FIFO 带宽发分配将保持比例 6:2:12，即 3:1:6。

RouterOS 中的 HTB

在 RouterOS 中有 4 个 HTB 树:

- global-in
- global-total
- global-out
- interface queue

当添加一个简单队列时，将产生 3 个 HTB 类(in global-in， global-total and global-out)，但在接口队列中不添加任何类。

当数据包通过路由器时，它将穿过所有 4 个 HTB 树——global-in， global-total， global-out 和 interface queue。如果是指向路由器的它将穿过 global-in 及 global-total HTB 树，如果数据包是从路由器发出的，它们将穿过 global-total，global-out 及 interface 队列。

11.5 Queue tree 队列树

操作路径: **/queue tree**

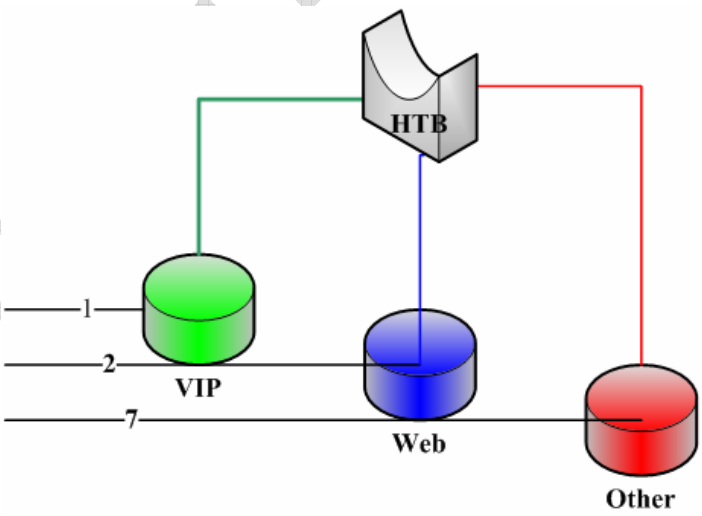
当你想使用基于协议，端口，IP 地址等的复杂数据分配流量时，你需要使用队列树。首先通过在**/ip firewall mangle** 下标记数据包流然后使用这个标记作为在这个队列树的数据包流标识。

属性描述

burst-limit (*整数*) - 当脉冲串激活时可以达到的最大数据率
burst-threshold (*整数*) - 用于计算是否允许脉冲。如果上一次脉冲时间的平均数据率低于 *burst-threshold* 则实际数据率可能达到 *burst-limit*。
burst-time (*整数*) - 用于计算平均数据率。
flow (*文本*) - 在**/ip firewall mangle** 下标记的数据包流。 当前队列参数仅应用于用这个数据流标记标识了的数据包。
limit-at (*整数*) - 这个队列的约定流量
max-limit (*整数*) - 在有足够带宽可用的情况下可达到的流量
name (*文本*) - 队列的描述性名称
parent (*文本*) - 父队列的名称。顶级的父队列是可用的接口（实际上是主 HTB）。低级点的父队列可能是其他的队列。
priority (*整数*: 1..8) - 队列的优先级。 1 是最高级等级， 8 为最低。
queue (*文本*) - 队列类型名称。类型是在**/queue type** 下定义的。这个参数仅应用于树等级制中的子队列。

Queue tree HTB 实例

这个事例中，设定 3 类数据 VIP、Web 和 Other，这三类数据中 VIP 为网络内的重要用户优先级最高为 1，访问网页的数据 web 其次为 2，而剩下的数据 Other 级别最低为 7，假设我们的网络是 1M 的 ADSL，我们通过配置 HTB 策略来保证网络内的优先数据。



通过用 **new-connection-mark** 标记向外的连接，并采取 **mark-connection** 动作。当这个完成时你可以使用 **new-packet-mark** 标记属于这个连接的所有数据包并采用 **mark-packet**。

首先 VIP 数据标记，我们通过 ip firewall address-list 定义 VIP 用户的地址列表，定义完成后通过 src-address-list 调用：

```
[admin@Office] /ip firewall mangle> print
Flags: X - disabled, I - invalid, D - dynamic
 0   ;;; vip
```

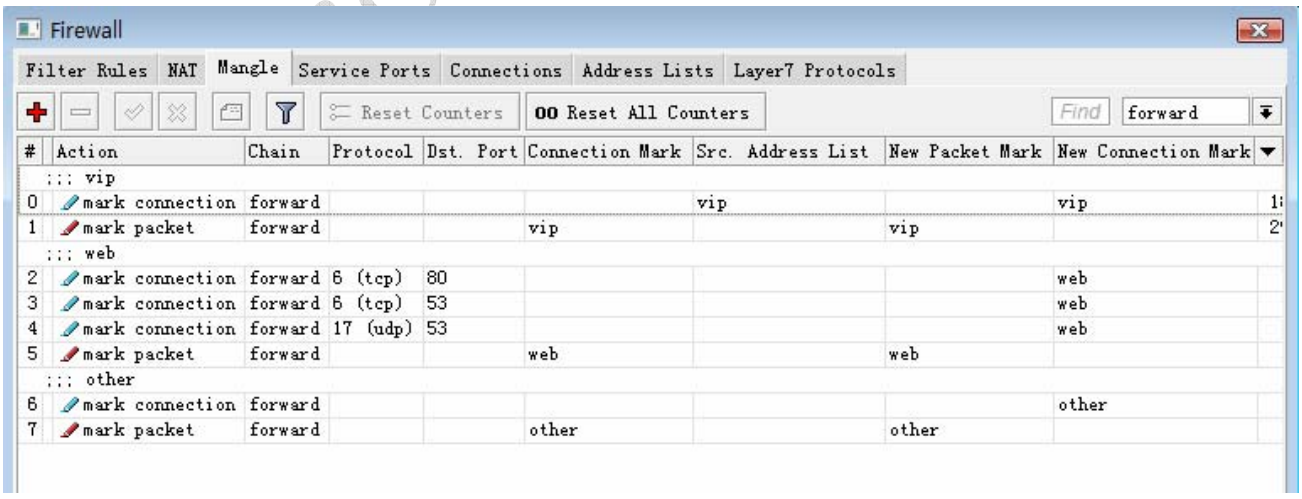
```
    chain=forward action=mark-connection new-connection-mark=vip
    passthrough=yes src-address-list=vip

1   chain=forward action=mark-packet new-packet-mark=vip passthrough=no
    connection-mark=vip
```

跟着定义 web 数据，这里我们需要针对访问网页的 tcp/80 端口和域名解析的 DNS 端口 tcp/53 和 udp/53 端口标记：

```
2   ;;; web
    chain=forward action=mark-connection new-connection-mark=web
    passthrough=yes protocol=tcp dst-port=80

3   chain=forward action=mark-connection new-connection-mark=web
    passthrough=yes protocol=tcp dst-port=53

4   chain=forward action=mark-connection new-connection-mark=web
    passthrough=yes protocol=udp dst-port=53

5   chain=forward action=mark-packet new-packet-mark=web passthrough=no
    connection-mark=web
```

最后对剩下的 Other 数据进行标记，因为前面已经标记了 VIP 和 Web 的数据包，所有剩下数据就是其他的 Other 数据：

```
6   ;;; other
    chain=forward action=mark-connection new-connection-mark=other
    passthrough=yes

7   chain=forward action=mark-packet new-packet-mark=other passthrough=no
    connection-mark=other
```



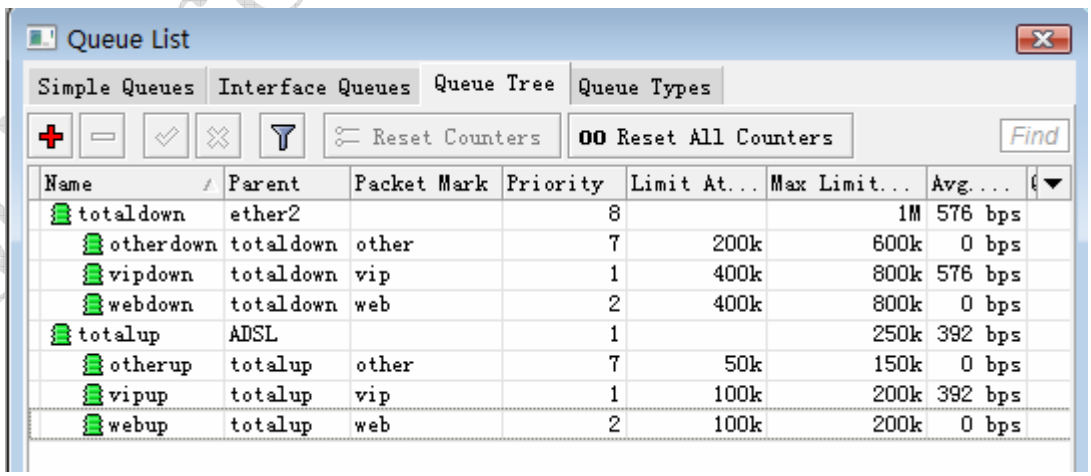
标记数据完成后，我们进入 queue tree 中，对数据进行优先级的配置，ADSL 总带宽为 1Mbps 下行，250kps 的上行，给三类数据带宽分配如下

- **VIP**：下行 Max-limit=800k limit-at=400k，上行 Max-limit=2200k limit-at=200k，优先级 1

- **Web**：Max-limit=800k limit-at=400k，上行 Max-limit=200k limit-at=200k，优先级 2
- **Other**：Max-limit=600k limit-at=200k，上行 Max-limit=150k limit-at=50k，优先级 7

根据以上参数，我们在 queue tree 中配置队列优先级：

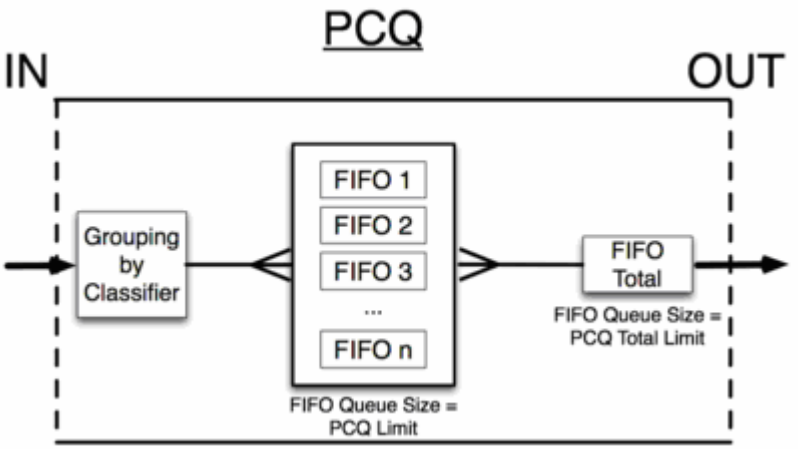
```
[admin@Office] /queue tree> print
Flags: X - disabled, I - invalid
 0   name="totalup" parent=ADSL packet-mark="" limit-at=0 queue=default
     priority=1 max-limit=250000 burst-limit=0 burst-threshold=0 burst-time=0s

 1   name="totaldown" parent=ether2 packet-mark="" limit-at=0 queue=default
     priority=8 max-limit=1000000 burst-limit=0 burst-threshold=0
     burst-time=0s

 2   name="vipdown" parent=totaldown packet-mark=vip limit-at=0 queue=default
     priority=2 max-limit=700000 burst-limit=0 burst-threshold=0 burst-time=0s

 3   name="vipup" parent=totalup packet-mark=vip limit-at=0 queue=default
     priority=2 max-limit=150000 burst-limit=0 burst-threshold=0 burst-time=0s

 4   name="otherdown" parent=totaldown packet-mark=other limit-at=0 queue=down
     priority=8 max-limit=500000 burst-limit=0 burst-threshold=0 burst-time=0s

 5   name="otherup" parent=totalup packet-mark=other limit-at=0 queue=up
     priority=8 max-limit=150000 burst-limit=0 burst-threshold=0 burst-time=0s

 6   name="webup" parent=totalup packet-mark=web limit-at=0 queue=default
     priority=1 max-limit=150000 burst-limit=0 burst-threshold=0 burst-time=0s

 7   name="webdown" parent=totaldown packet-mark=web limit-at=0 queue=default
     priority=1 max-limit=700000 burst-limit=0 burst-threshold=0 burst-time=0s
```



11.6 PCQ 配置

PCQ 算法比较简单，首先利用分类器从相应数据流中区分一个子数据流，然后在每一个子数据流上建立独立的 FIFO 队列长度和限制，再归类所有的子数据流在一起，并应用全局 FIFO 队列长度和限制。 PCQ 参数:

- **pcq-classifier** (dst-address | dst-port | src-address | src-port; 默认: "") : 选择子数据流分类类型。
- **pcq-rate (数字) :** 每个子数据流可获得的最大数据带宽。
- **pcq-limit (数字) :** 在数据包中一个子数据流的队列长度
- **pcq-total-limit (数字) :** 全局 FIFO 队列的队列长度

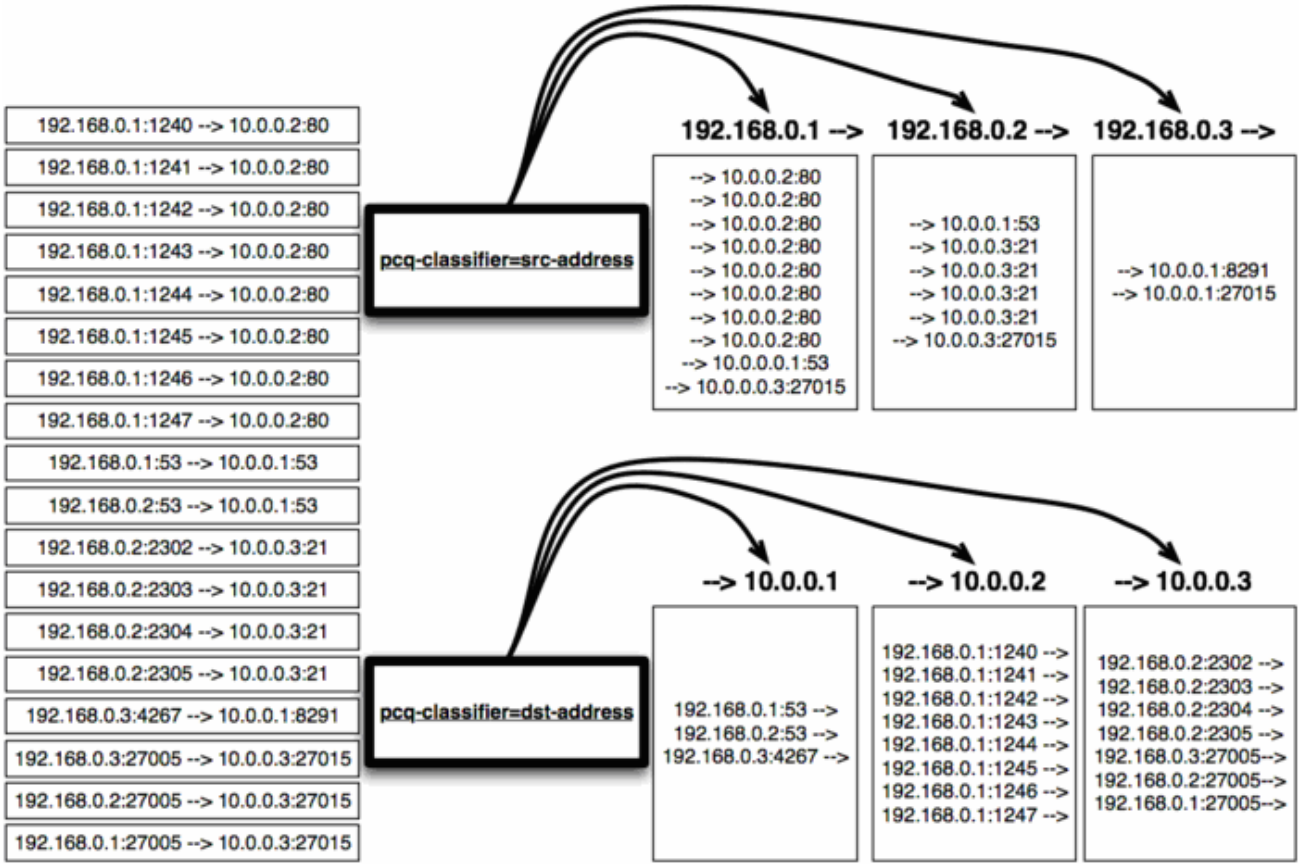


因此，当有 100 个队列需要限制 1000kbps 下载时，我们可以使用 1 个 PCQ 队列和该 PCQ 队列包含的 100 子数据流队列。

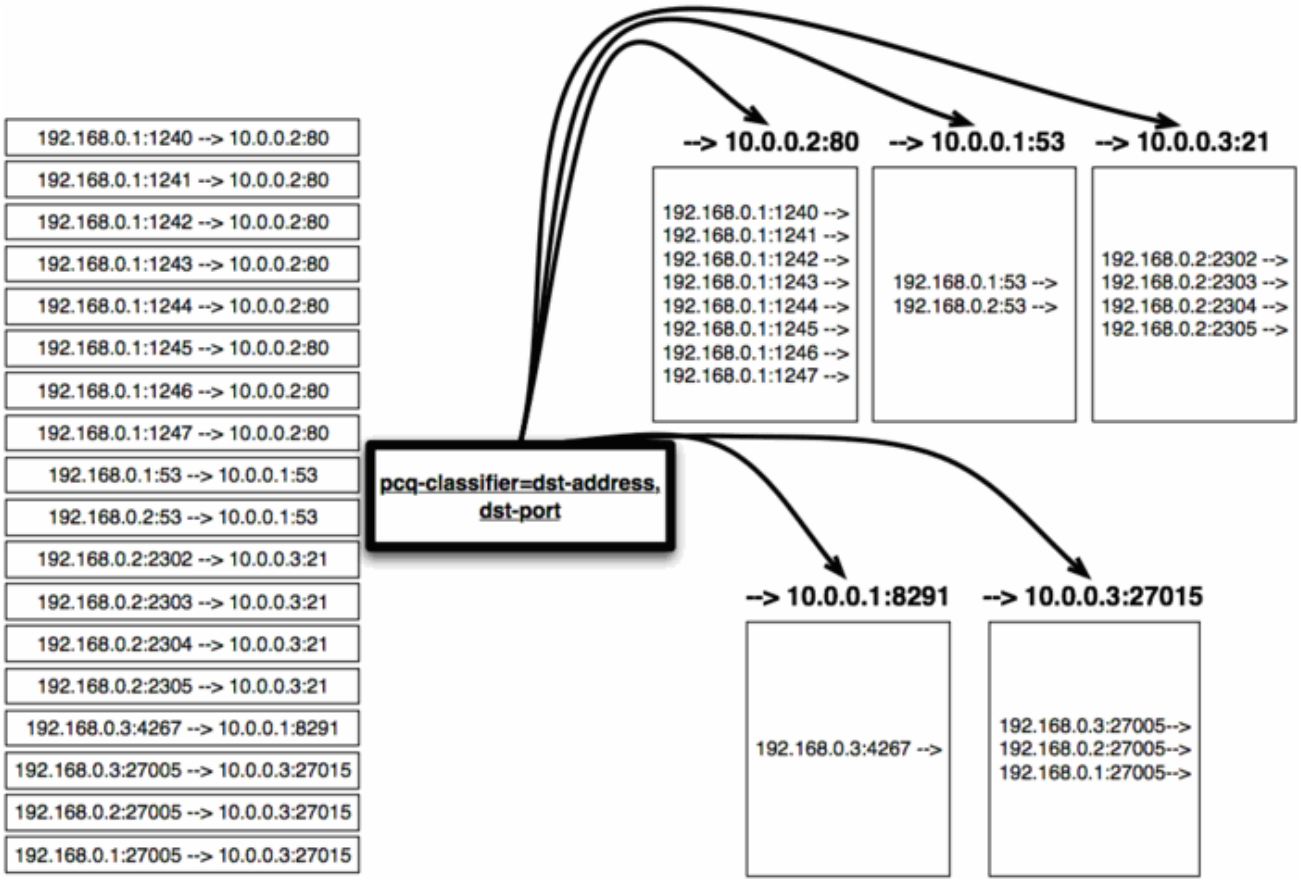
分类器

为更好的理解分类器，我用一组 IP 地址和端口到对应的地址和端口数据流的实例，这时我们将选择一种分类器，并通过 PCQ 将 18 个数据流从中分离到 PCQ 的子数据流中进行分类。

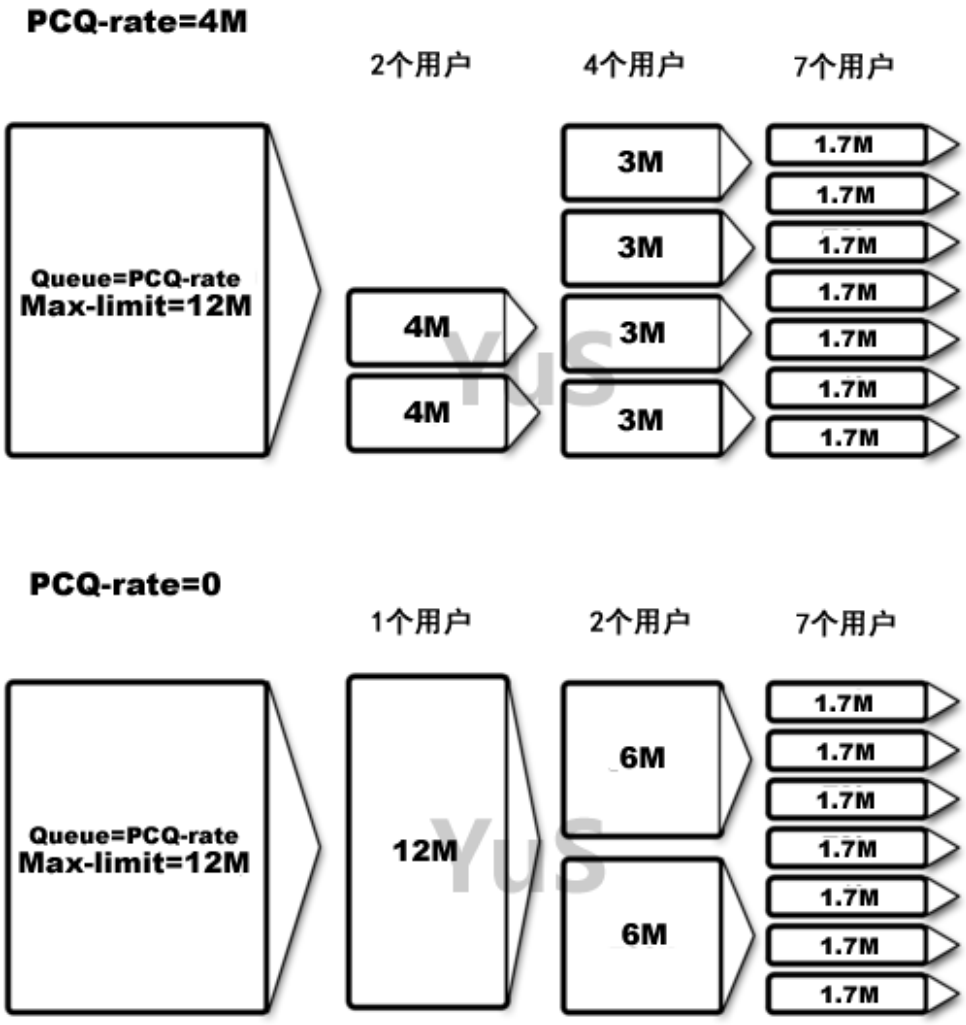
PCQ 的目标和源地址分类原理图:



端口分类

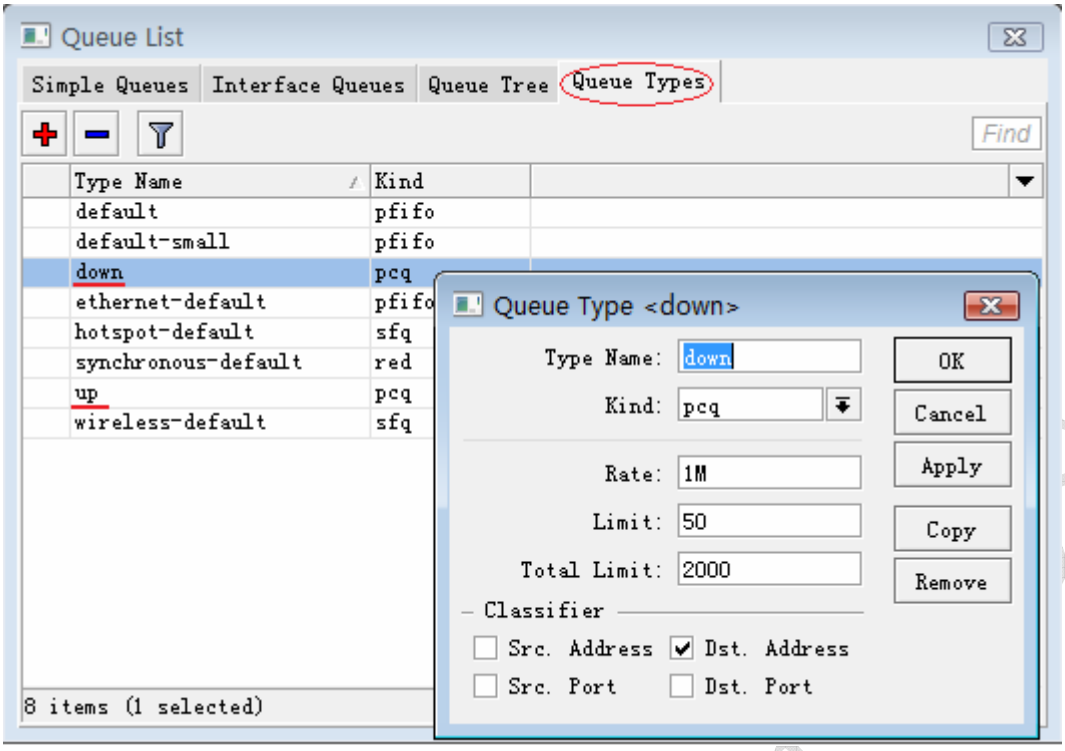


在局域网中因为网络带宽的问题，需要对网络流做控制，但又因为做固定的流量控制的时候，会造成在上网空闲时候带宽的浪费，这里我们可以同 RouterOS 的 PCQ 算法完成对内部局域网流量的动态分配,如下图所示：

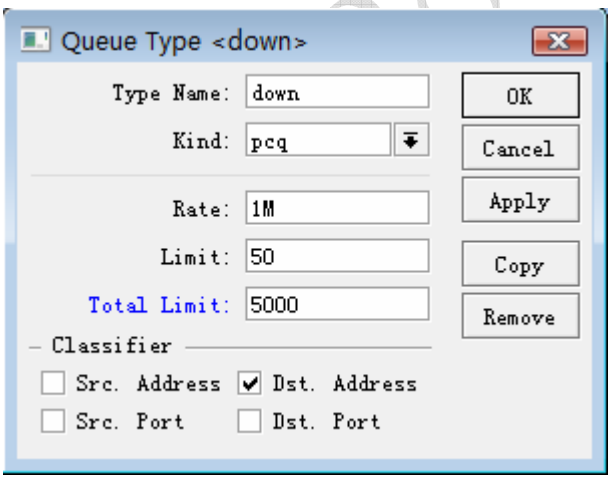


通过上图，我们可以看到当 PCQ 的速率设定为 128k 的时候，平均每个用户将会得到同样的带宽 128k，当上网高峰期的时候 PCQ 才会做二次流量分配，如果 PCQ 的速率在开始就设定为 0k，这样在一个用户的时候就可以得到全部带宽，之后是 2 个用户平均分配，依次类推，但最后带宽会控制在 73k 的范围内，控制最小使用带宽，保证用户正常使用。

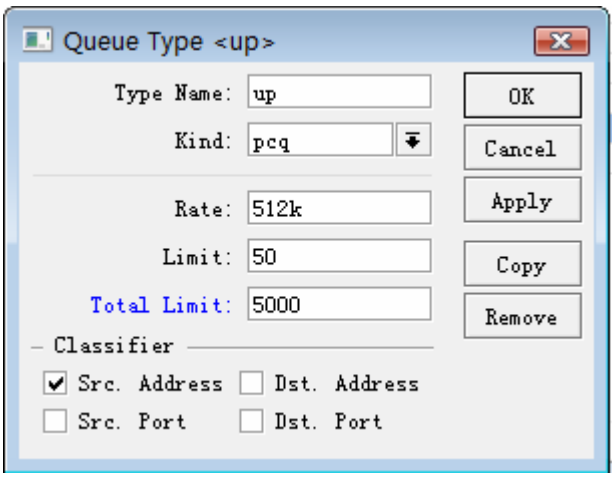
配置这里我们配置 192.168.10.0/24 这个段的 PCQ 流量控制，估计有 100 个用户在线，首先进入 Queue Type 中配置 PCQ 的上行和下行分别为 512k 和 1m：



首先我们配置下行，每个用户获取 1m 的下行流量。由于是 100 个用户在线，所以在 limit 不变的情况下，total-limit 应该设置为 50*100=5000，下行指向的是目标地址，所以我们选择 dst-address：



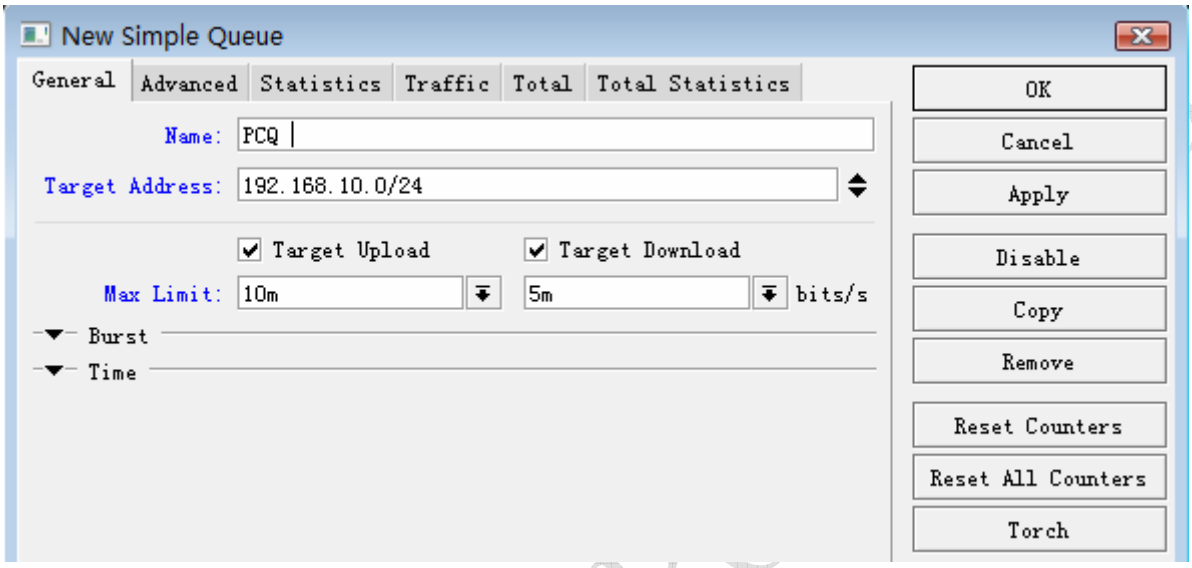
上行选择 src-address，并配置 512k 的上行流量配置如下：



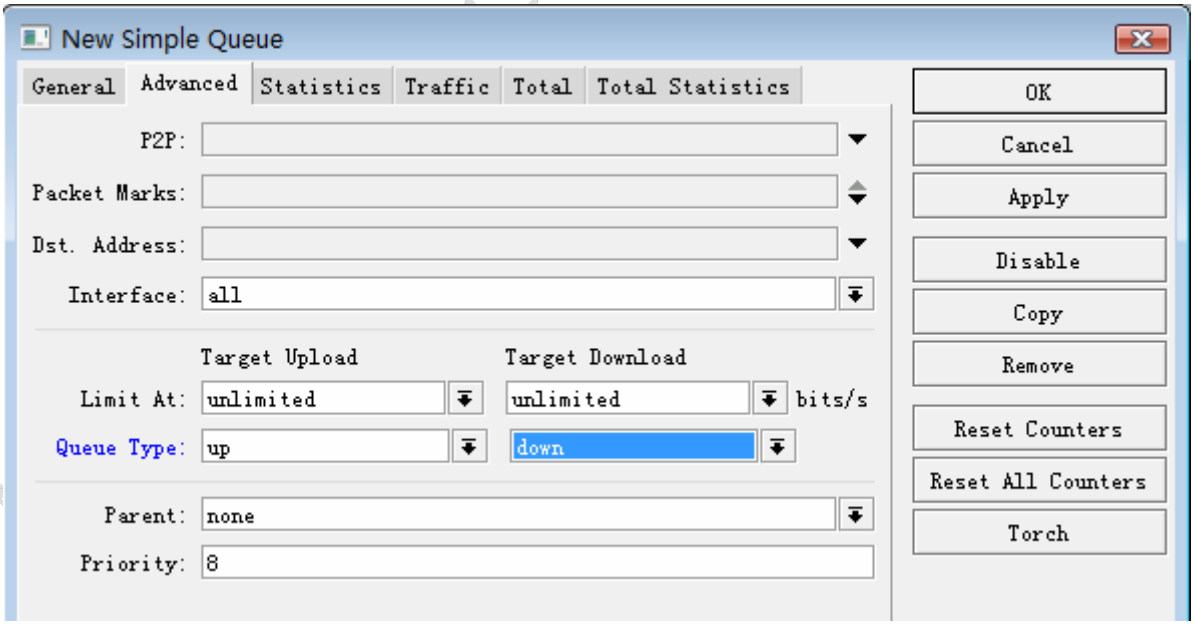
注意，Limit 和 Total-Limit 的关系：

- 默认情况下 total-limit 是 2000，该规则仅能容纳 40 个用户（total-limit/limit=2000/50=40）
- 解决方法必须增加 total-limit 或者减少 limit
- 但必须保证每个用户队列(limit)获取 10-20 个数据包

在配置好 Queue Type 后我们进入 Simple Queue 中配置流量控制规则，这里我们在 General 中配置总出口带宽假设为 10M，上行带宽为 5M，内网地址段为 192.168.10.0/24：



接下来配置 Queue-type 类型，进入 advanced 目录，选择上行和下行为刚才定一的 PCQ 类型 Up 和 Down：

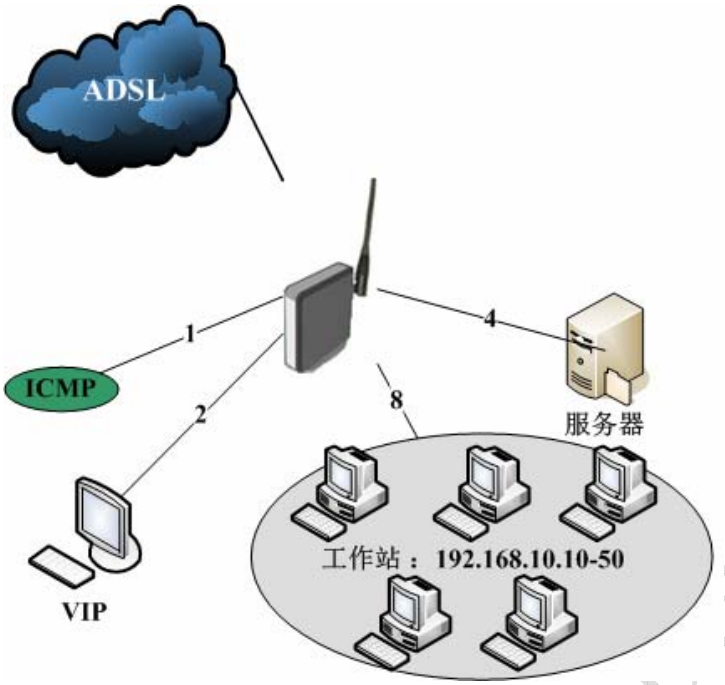


这样 PCQ 配置就完成，只需要在 simple queue 中配置一条规则，就可以控制所有用户的流量。

11.7 HTB 与 PCQ 流量控制

我使用 MikroTik RouterOS 通过 PPPoE 连接到互联网（基于 ADSL 拨号），为了了内网的用户得到优质的网络环境，通过对流量上下行流量进行优先控制，保证特定的网络用户得到优质的网络带宽。

局域网采用以太网有线和 WiFi 无线接入方式，通过一个 bridge 将有线口与无线连接起来，网络拓扑图如下：



注意：

- ADSL 是一个 PPPoE 客户端接口.运行在 ether1 的网卡上，ether1 连接到一个 ADSL-modem
- 内网 IP 地址 192.168.10.1/24，内网通过以太网和 WiFi 无线接入方式
- 通过 NAT/Masquerad 隐藏内网用户。
- ADSL 连接速度：3Mbps

客户主机

主机	IP	优先级	备注
服务器	192.168.10.6	优先级 4	公司服务器
VIP	192.168.10.7	优先级 2	重要上网人员，优先级最高
工作站	192.168.10.0/24	等级低 8，但除 icmp 协议	员工工作电脑

服务端口

协议	端口协议	优先级	目标
ICMP	icmp	最高 1	所有主机

如何获得优先的互联网带宽和流量控制，通过下面的步骤来实施：

- 由于 ADSL 有较小的缓冲空间，并当带宽满载下载速度会变慢。所以 RouterOS 配置上传或者下载不能超过 90%；
- 当 VIP 想与外面通信，将得到最优先带宽；
- ICMP 协议优先通过，得到较小的延迟；
- 需要考虑 CIR（约定带宽）即 Limit-at，MIR（最大带宽）即 Max-limit，每个流量控制需要考虑他们的 CIR 与 MIR 值

基本配置

下面我可以看到通过 ether2-wan 拨号的 ADSL 外网拨号接口，桥接 ether1-lan 与 wlan1 的 bridge1 的接口

```
[admin@MikroTik] /interface> print
Flags: D - dynamic, X - disabled, R - running, S - slave
 #    NAME                                TYPE            MTU   L2MTU
 0  R  ADSL                                pppoe-out       1480
 1  R  bridge1                             bridge          1500   65535
 2  R  ether1-lan                          ether           1500  1526
 3  R  ether2-wan                          ether           1500  1524
 4     ether3                              ether           1500  1524
 5  R  wlan1                               ether           1500  1524
 [admin@MikroTik] /interface>
```

定义 HTB 流量控制

HTB 里，我们需要考虑到父级、子级等关系，首先定义父级（parents），即上下行的总带宽，即定义整个 HTB 的总带宽

经过带宽测试后，计算出 ADSL 带宽为 2850/420kbps,我们需要在 queue tree 添加带宽限制，使用 90%的实际带宽分配给下载和上传，这里我们定义总的上下行带宽，parent 定义接口，bridge1 对应内网的下行数据（这里我们将），ADSL 则对应发出的上行数据

```
/queue tree add name=Download parent=bridge1 max-limit=2600k
/queue tree add name=Upload parent=ADSL max-limit=360k
```

ICMP 协议

对 ICMP 协议进行标记和流量控制，ICMP 协议我们需要首先满足，让所有用户得到较低 ICMP 延迟。进入 mangle 标记连接和数据包

```
/ip firewall mangle add protocol=icmp action=mark-connection new-connection-mark=icmp-con
chain=forward
/ip firewall mangle add connection-mark=icmp-con action=mark-packet new-packet-mark=icmp
chain=forward
```

我们进入 Queue tree,我们考虑到 ICMP 协议主要是网络监测，对带宽需求不大，CIR 定义为 100kbps，最大 MIR 带宽为 500kbps，保证正常的 ICMP 通信就可以了

```
/queue tree add name=icmp-down parent=Download packet-mark=icmp limit-at=100k max-limit=500k
priority=1
/queue tree add name=icmp-up parent=Upload packet-mark=icmp limit-at=100k max-limit=500k
priority=1
```

VIP 优先级高于其他主机

192.168.10.7 为 VIP 需要得到更多的带宽，但需要考虑到 CIR 保证使用到最低带宽，这里我们为 VIP 分配最低下行 800kbps，上行 200kbps 带宽，当然 MIR 最大可以获取到 2600kbps

标记 VIP 的连接传输与数据：

```
/ip firewall mangle add src-address=192.168.10.7/32 action=mark-connection
new-connection-mark=vip-con chain=forward
/ip firewall mangle add connection-mark=vip-con action=mark-packet new-packet-mark=vip
chain=forward
```

接下来进入 Queue tree 对 VIP 配置带宽规则：

```
/queue tree add name=vip-down parent=Download limit-at=1024 packet-mark=vip max-limit=5000k
priority=2
/queue tree add name=vip-up parent=Upload limit-at=512 packet-mark=vip max-limit=100k
priority=2
```

服务器规则

我们将 192.168.10.6 的服务器标记，并定义他们的 HTB 带宽规则

```
/ip firewall mangle add src-address=192.168.10.6/32 action=mark-connection
new-connection-mark=server-con chain=forward
/ip firewall mangle add connection-mark=server-con action=mark-packet new-packet-mark=server
chain=forward
```

进入 Queue tree 对服务器带宽规则：

```
/queue tree add name=server-down parent=Download limit-at=1024 packet-mark=server
max-limit=2600k priority=4
/queue tree add name=server-up parent=Upload limit-at=512 packet-mark=server max-limit=300k
priority=4
```

工作主机最低级别

剩下工作主机需要标记所有的传输，所有传输来至 192.168.10.0/24，因此我们使用 src-address 获取，通过标记连接（users-con），然后从连接中提取数据包（users）。

```
/ip firewall mangle add chain=forward src-address=192.168.10.0/24 action=mark-connection
new-connection-mark=users-con
/ip firewall mangle add connection-mark=users-con action=mark-packet new-packet-mark=users
chain=forward passthrough=no
```

这时我们需要添加 2 条新的 PCQ 规则，第一条为 ADSL-down，定义组的 dst-address 分类，即 ADSL 的下载，将 pcq-rate 设置为 0，这样将建立每个主机的动态带宽。第二条为 ADSL-up，即 ADSL 的上行，定义组为 src-address 分类，pcq-rate=100kbps，限制每台主机的上行带宽（因为 ADSL 上行相对较小）。

```
/queue type add name=ADSL-down kind=pcq pcq-classifier=dst-address
/queue type add name=ADSL-up kind=pcq pcq-rate=100k pcq-classifier=src-address
```

在 queue tree 定义

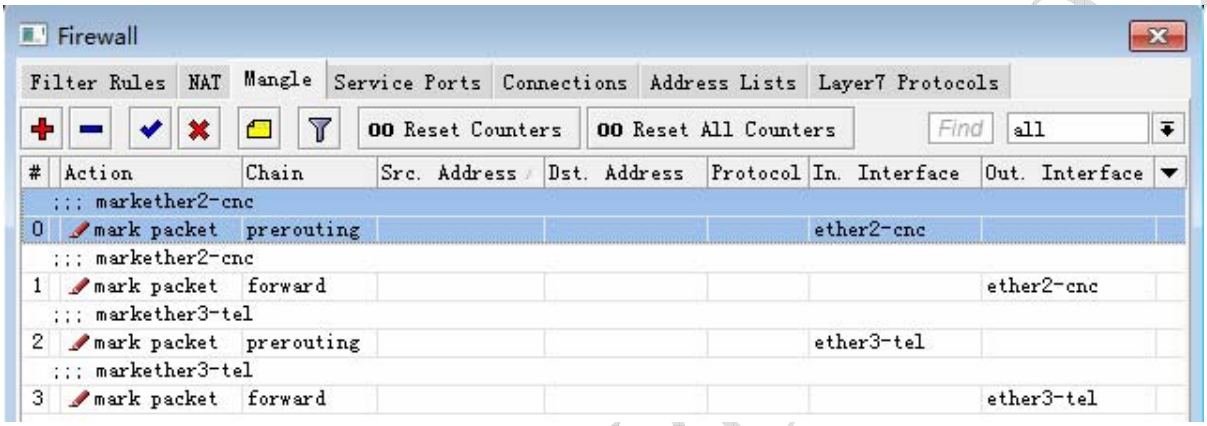
```
/queue tree add parent=Download queue=users-down packet-mark=users
/queue tree add parent=Upload queue=users-up packet-mark=users
```

11.8、网吧的 PCQ 与 HTB

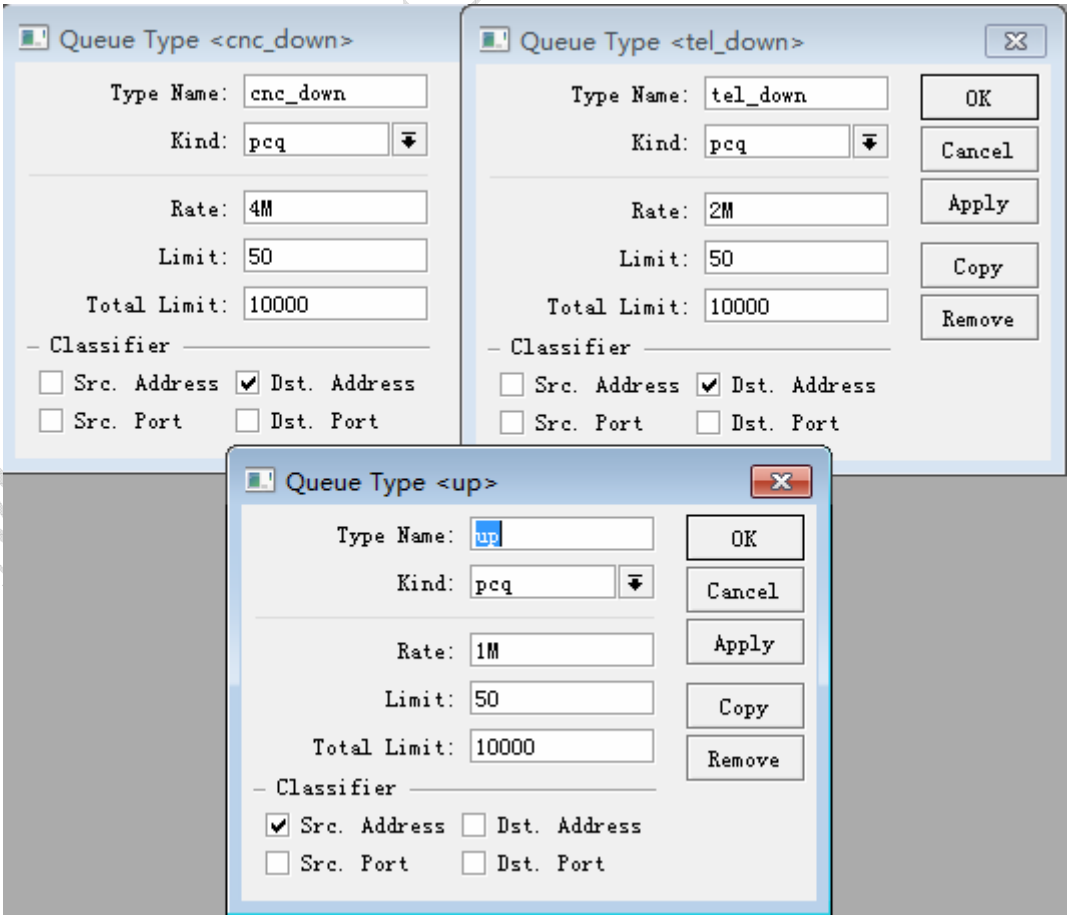
这里我们有一个实际环境，我们需要实现对带宽的动态分配；电信带宽为 6M，网通带宽为 12M；

配置步骤：
1、在 ip firewall mangle 标记上下行数据流
2、进入 queue type 定义单机带宽
3、在 queue tree 定义总带宽和流量控制规则

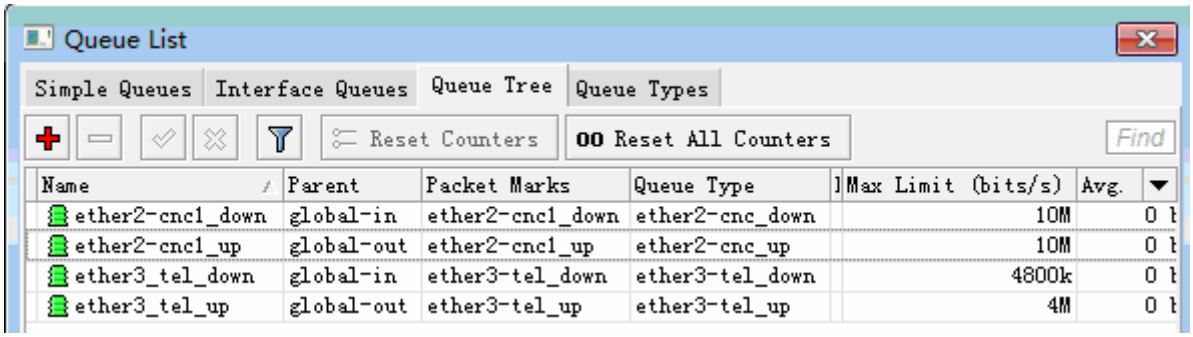
步骤 1：在 Mangle 标记上下行的标记：



步骤 2：在 Queue Type 里按照 200 台主机的数量，定义 PCQ 规则：



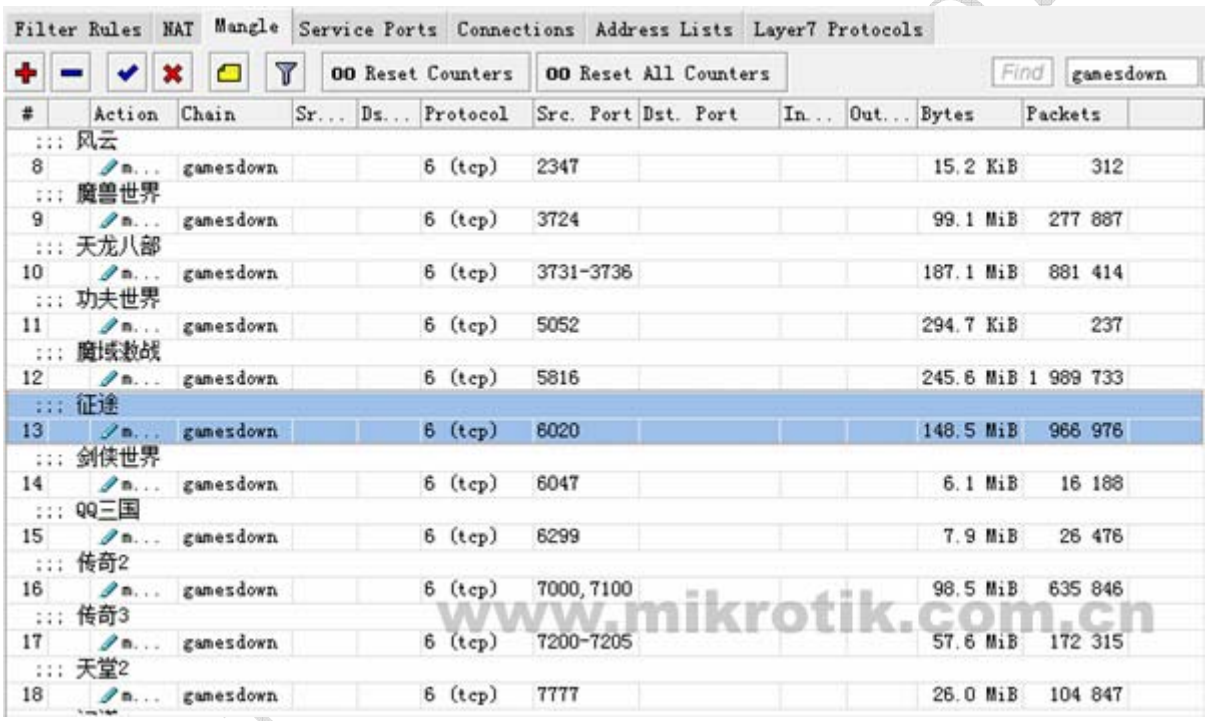
步骤 3：建立 Queue Tree 规则，记住保留一定带宽为缓冲，网通我们保留 2M，电信我们保留 1.2M 带宽



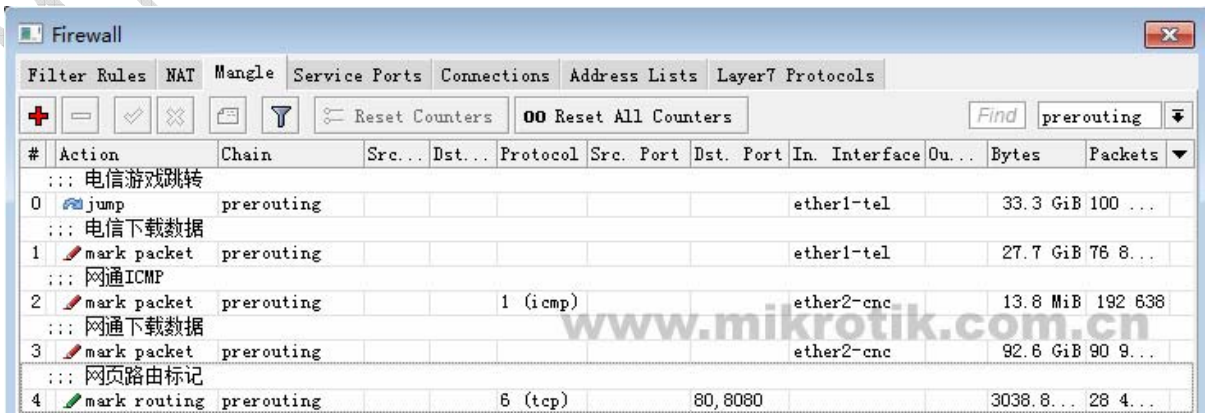
HTB 游戏优先

通过 HTB 为游戏预留带宽，保证在下载和视频情况下，游戏照样流畅，HTB+PCQ 组合实现：

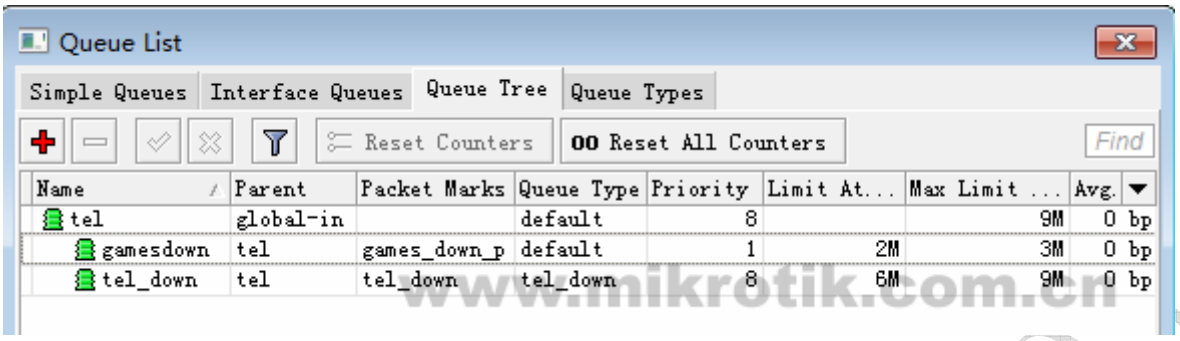
步骤 1：在原有的动态的 PCQ 流控规则上进行改进，首先导入游戏端口，建立新的 gamesdown 链表，将游戏与其他数据区分出来



通过将指定的数据转移到游戏链表进行过滤和数据包处理：



假设电信带宽是 11M，预留 2M 为缓冲带宽，最大带宽为 9M，电信线路下行的 HTB 设置，游戏优先级为 1 最高，其他下行数据为 8 最低；这里游戏只分配了 3M 最大带宽，最低保证 2M，对于游戏带宽较小不需要那么大；其他下行数据最低保证 6M。



如果需要也可以为游戏流量配置 PCQ 规则，定义一个游戏的 PCQ 队列类型 Queue-type 对每个用户进行带宽控制。

11.9 Connection Rate 流量控制

Connection Rate 是一个防火墙标记器，允许捕获在当前传输的连接速度

Connection Rate 原理

每个连接项目在 connection tracking 表中是双向通讯。每次得到相关的数据包到特定的项目，数据包的长度值（包括 IP 数据包头）被添加到“Connection-bytes”值，换句话说，Connection-bytes 包括两部分上行和下行。

Connection Rate 计算连接的速度基于"connection-bytes"的变化。Connection Rate 每秒会被重新计算，且没有任何平均值。

两个选项 "connection-bytes" 和"connection-rate" 工作只能在 TCP 和 UDP 传输。(你需要指定协议激活这些选项) 在"connection-rate"您可以指定速度，和你想捕获范围。

例如：这个规则是捕获当连接速度低于 100kbps 通过路由器的 TCP/UDP 传输

```
/ip firewall filter
add action=accept chain=forward connection-rate=0-100k protocol=tcp
add action=accept chain=forward connection-rate=0-100k protocol=udp
```

注意：Connection Rate 从 3.30 才能获得，这个选项是用于捕获传输密集的连接

传输优先级

Connection-rate 能被使用在各种方式，通常的方式是使用队列树进行 HTB 的优先级控制，检测并设置低优先级给“heavy connections”（连接在一段时间内保持较快速率，例如：P2P、HTTP、FTP 下载）通过这样做，你可以区分所有其它传输的优先次序，通常包括 VOIP、HTTP 浏览和在线游戏

connection-rate 选项没有任何平均值，我们需要确定识别"heavy connections"的差额。如果我们假设正常的 HTTP 浏览连接小于 500kB(4Mb,即 connection-bytes 值)长度,VOIP 需要不超过 200kbps 的流量,那么每次连接当超过 500kB 后，仍然有 200kbps 的流量将被认为是"heavy connections"

(对于 HTTP 浏览和 VOIP 可能有不同的"connection-bytes"在你的网络环境中，所以请你在实际操作时，这个实例仅做参考。)

下面实例让我们假设，我们有 6Mbps 上传和下载

```
per-connection-classifier=
PerConnectionClassifier ::= [!]ValuesToHash:Denominator/Remainder
  Remainder ::= 0..4294967295   (integer number)
  Denominator ::= 1..4294967295   (integer number)
  ValuesToHash ::= src-address|dst-address|src-port|dst-port[,ValuesToHash*]
```

实例脚本

```
/ip firewall mangle
add chain=forward action=mark-connection connection-mark=!heavy_traffic_conn
new-connection-mark=all_conn
add chain=forward action=mark-connection connection-bytes=500000-0 \
    connection-mark=all_conn connection-rate=200k-100M \
    new-connection-mark=heavy_traffic_conn protocol=tcp
add chain=forward action=mark-connection connection-bytes=500000-0 \
    connection-mark=all_conn connection-rate=200k-100M \
    new-connection-mark=heavy_traffic_conn protocol=udp
add chain=forward action=mark-packet connection-mark=heavy_traffic_conn \
    new-packet-mark=heavy_traffic passthrough=no
add chain=forward action=mark-packet connection-mark=all_conn \
    new-packet-mark=other_traffic passthrough=no

/queue tree
add name=upload parent=public max-limit=6M
add name=other_upload parent=upload limit-at=4M max-limit=6M \
    packet-mark=other_traffic priority=1
add name=heavy_upload parent=upload limit-at=2M max-limit=6M \
    packet-mark=heavy_traffic priority=8
add name=download parent=local max-limit=6M
add name=other_download parent=download limit-at=4M max-limit=6M \
    packet-mark=other_traffic priority=1
add name=heavy_download parent=download limit-at=2M max-limit=6M \
    packet-mark=heavy_traffic priority=8
```

脚本说明

在 mangle 中，我们需要分离所有连接到 2 个组中，这样数据包标记从 2 个组取得。我们讨论的客户的传输理论上大多标记在 forward 链表中。

请记住，“heavy”连接将有低的优先级，队列将打压 max-limit—heavy 连接将被限制速度。这样引起改变高优先级连接获取更多的带宽，当在一次产生 connection-rate 将上升，并导致其改变为较低优先级。为了避免这一点，我们必须确保，一旦发现“heavy”连接，剩下的标记在所有时间仍然是"heavy"连接

Mangel 规则配置

```
/ip firewall mangle
add chain=forward action=mark-connection connection-mark=!heavy_traffic_conn
new-connection-mark=all_conn
```

这个规则将确定"heavy"连接，连接将只剩下“heavy”

```
add chain=forward action=mark-connection connection-bytes=500000-0 \
   connection-mark=all_conn connection-rate=200k-100M \
   new-connection-mark=heavy_traffic_conn protocol=tcp
add chain=forward action=mark-connection connection-bytes=500000-0 \
   connection-mark=all_conn connection-rate=200k-100M \
   new-connection-mark=heavy_traffic_conn protocol=udp
```

这两个规则将根据我们的标准标记所有 heavy 连接，每次连接在第一次超过 500KB 流量后，仍然保持 200kbps 以上速度被认为“heavy”。

```
add chain=forward action=mark-packet connection-mark=heavy_traffic_conn \
   new-packet-mark=heavy_traffic passthrough=no
add chain=forward action=mark-packet connection-mark=all_conn \
   new-packet-mark=other_traffic passthrough=no
```

最后 2 条规则在 mangle 中将标记数据包传输从相应的连接中。

队列配置

这个是一个简单的队列树被放到接口的 HTBThis is a simple queue tree that is placed on the Interface HTB – 你的 ISP 连接的“wan”接口，“lan“连接你的内网客户。如果你有多个 wan 接口或者多个 lan，你将需要标记上行和下行分离标记

```
/queue tree
add name=upload parent=public max-limit=6M
add name=other_upload parent=upload limit-at=4M max-limit=6M \
   packet-mark=other_traffic priority=1
add name=heavy_upload parent=upload limit-at=2M max-limit=6M \
   packet-mark=heavy_traffic priority=8
add name=download parent=local max-limit=6M
add name=other_download parent=download limit-at=4M max-limit=6M \
   packet-mark=other_traffic priority=1
add name=heavy_download parent=download limit-at=2M max-limit=6M \
   packet-mark=heavy_traffic priority=8
```

Connection-rate HTB 事例

通过 Connection-rate 配合 HTB 分离正常的网页浏览和网页视频，方法是标记网页的 tcp/80 端口，并区分他们连接速率，将高速率的流量认为是网页视频、剩下的则是正常的网页浏览，通 HTB 将正常的网页浏览设置为优先。

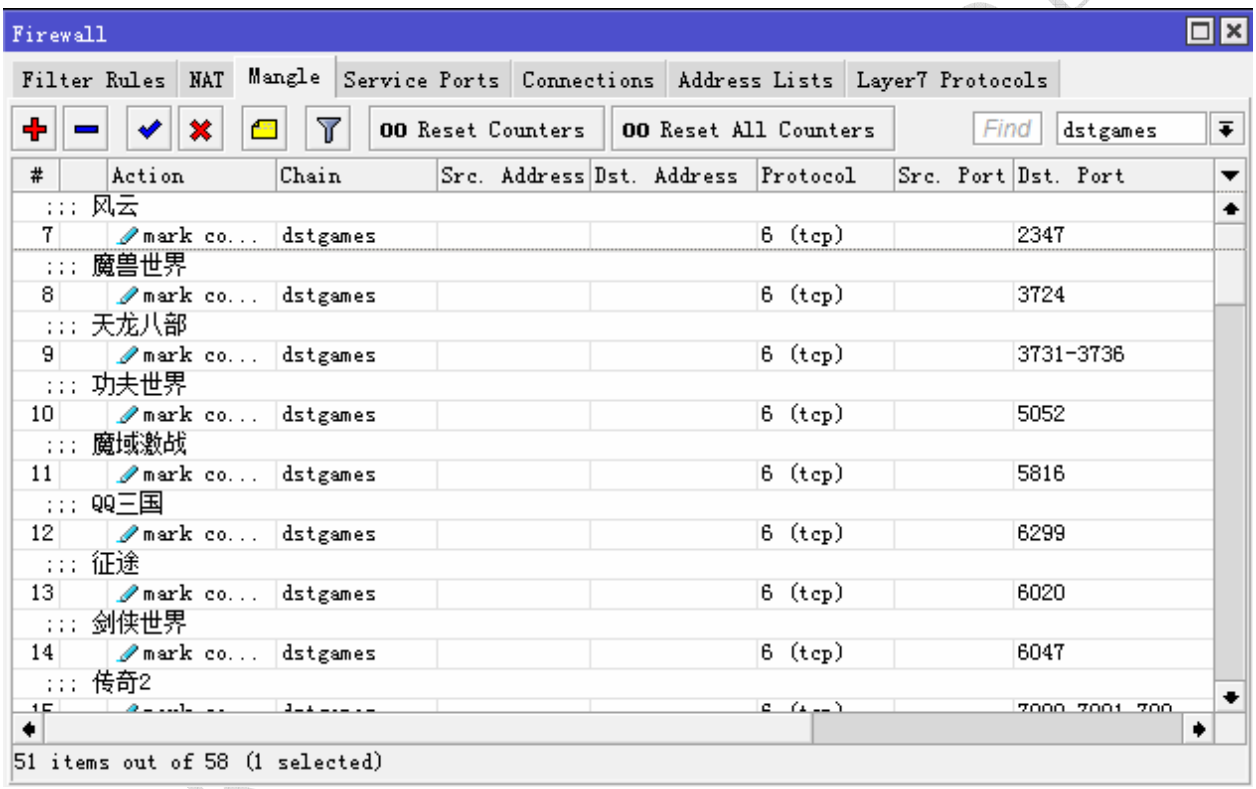
通过 Mangle 标记

我们需要考虑一个网络的优先处理结构，首先是游戏带宽最优先、其次是正常浏览网页带宽、然后是网页视频的带宽，最后在才是其他的下载数据，根据这一的结构我们在 mangle 标记也是从游戏开始到最后的下载数据。

首先我们需要导入游戏优先的标记，将游戏优先文件 games_v2.rsc 放入 RouterOS 的 files 根目录下，然后在命令行下使用 import 命令，导入脚本文件，如下图输入以下命令，并回车执行提示：脚本文件载入，并执行成功：

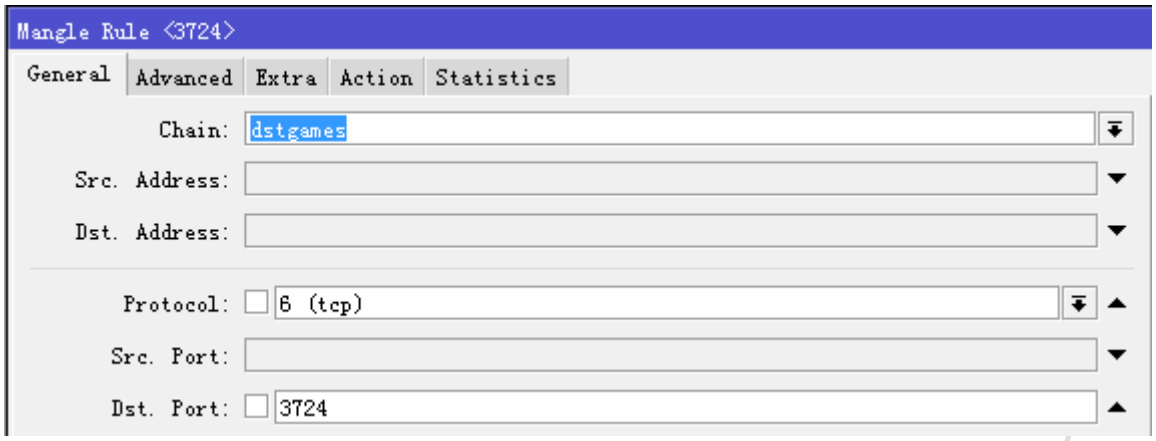
```
[admin@MikroTik] > import games_v2.rsc
Opening script file games_v2.rsc
Script file loaded and executed successfully
[admin@MikroTik] >
```

我们进入 ip firewall mangle 中可以找到导入游戏标记规则，我们选择右上角 dstgames 的自定义链表，可以看到各种游戏的标记，你也可以在此链表添加自己的游戏标记

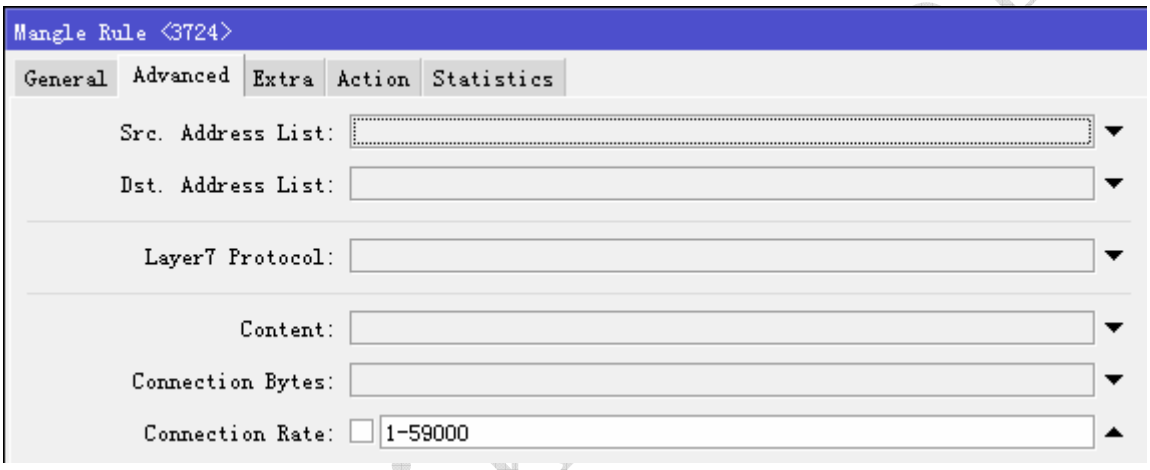


每条规则都标记的是目标端口，因为这里我将使用 forward 链表跳转数据到这个游戏标记，如果使用 prerouting 链表注意接口方向。

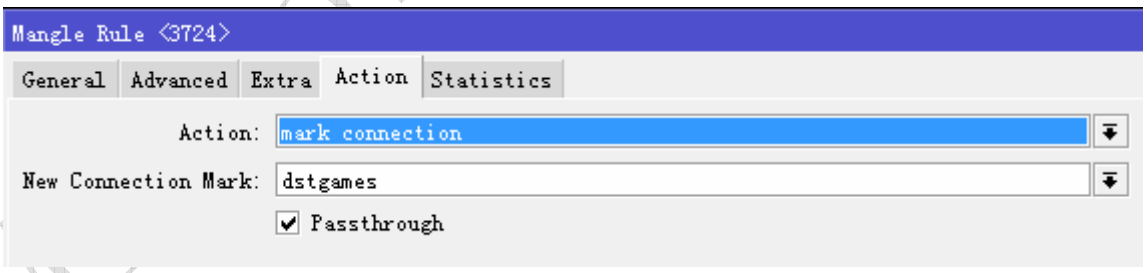
下面是魔兽世界的端口 tcp/3724，标记服务器的端口



在每个游戏标记规则里，都使用了 connection-rate 的参数，参数值为 1-59k，即带宽使用在 1-59kps 的数据认为是该端口的游戏流量，否则认为是其他数据，这样的目的是避免该端口被下载所占用，导致游戏带宽控制失效，当然这个值可以根据你自己的需要调整和修改



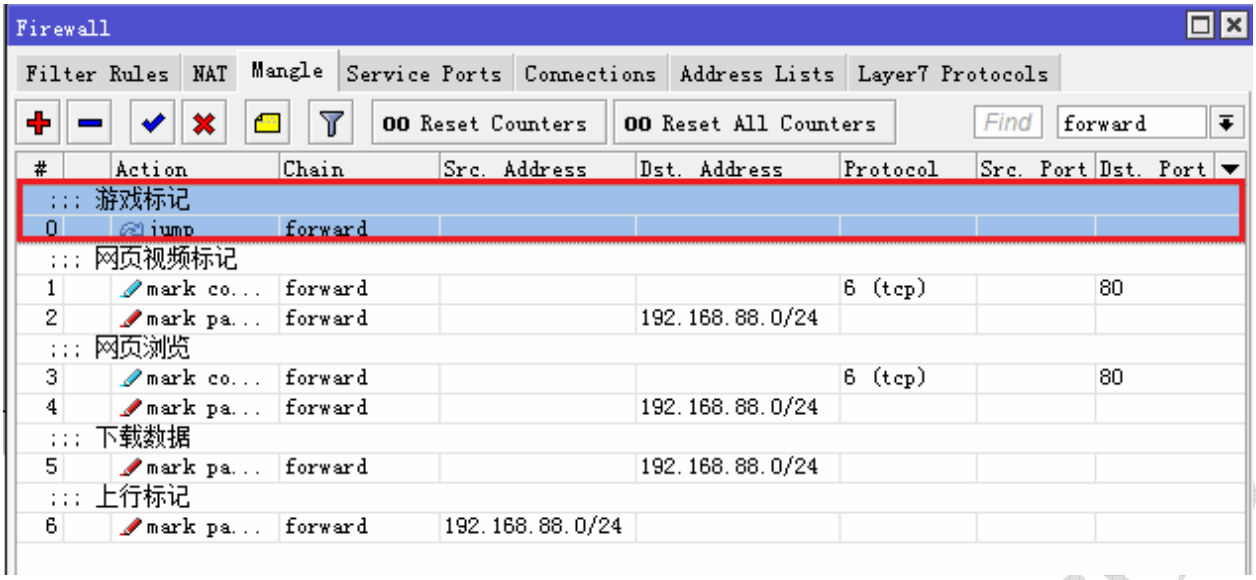
最后是在 action 里标记为 mark-connection，并填写标记名称 dstgames，passthrough=yes 要传递给后面的数据包标记规则



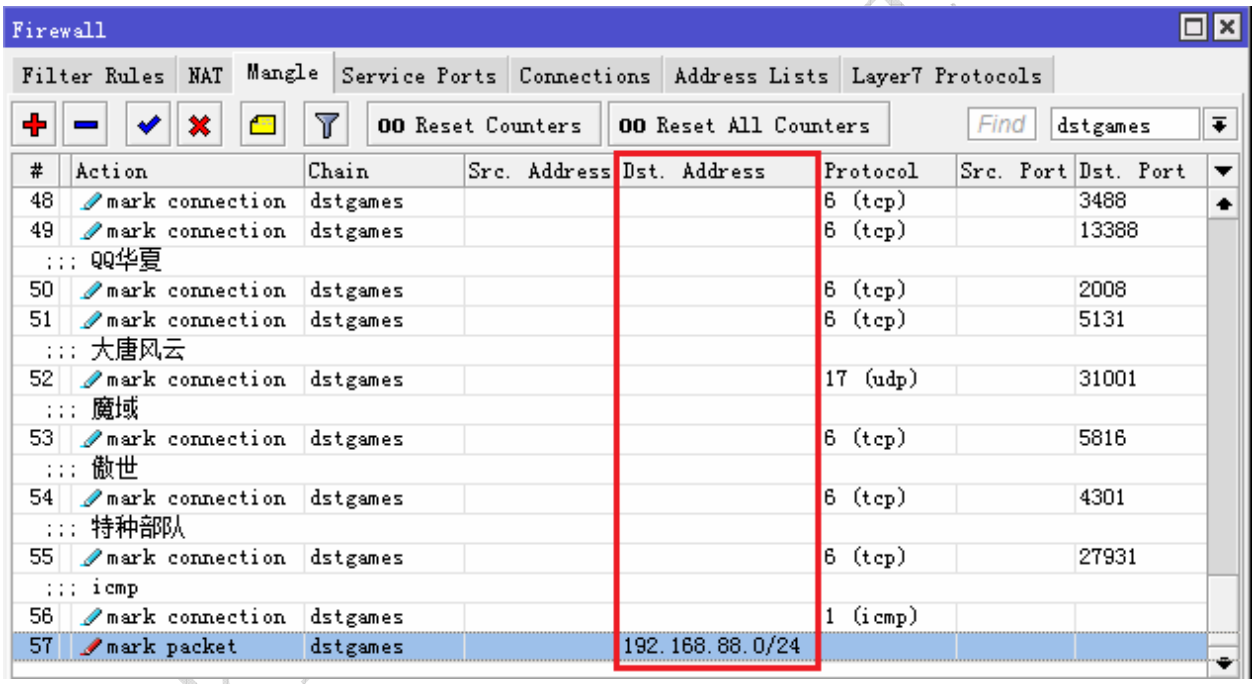
Forward 标记

我们进入 forward 链表，添加一条跳转的规则，将 forward 中所有数据首先进入 dstgames 的链表过滤一次，将游戏分离出来

```
/ip firewall mangle add chain=forward action=jump jump-target=dstgames
```



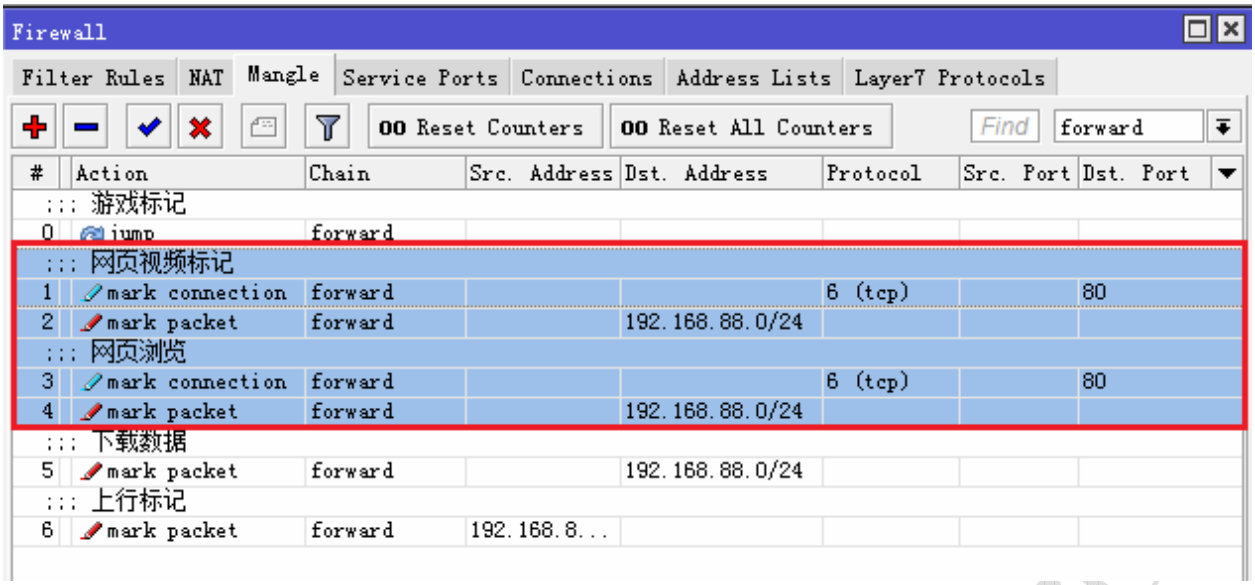
这条规则在 forward 链表里排在最前面，因为是首先处理的数据。这里我们的内网地址段是 192.168.88.0/24，我们需要通过地址来区分下载和上传，所以我们需要返回 dstgames 链表里，将数据包标记规则修改下



在 dstgames 里最后一条规则说 mark-packet，即汇总和标记之前所有游戏连接的数据，并生成可以被 queue 流控规则使用的 packet，之前的游戏连接是没有区分上行和下行的，所以我们这里我们通过 dst-address 将到内网（即下行）的数据标记出来，这样可以得到游戏的下行数据，该规则的 passthrough=no。注意：你的内网地址根据你的设置修改。

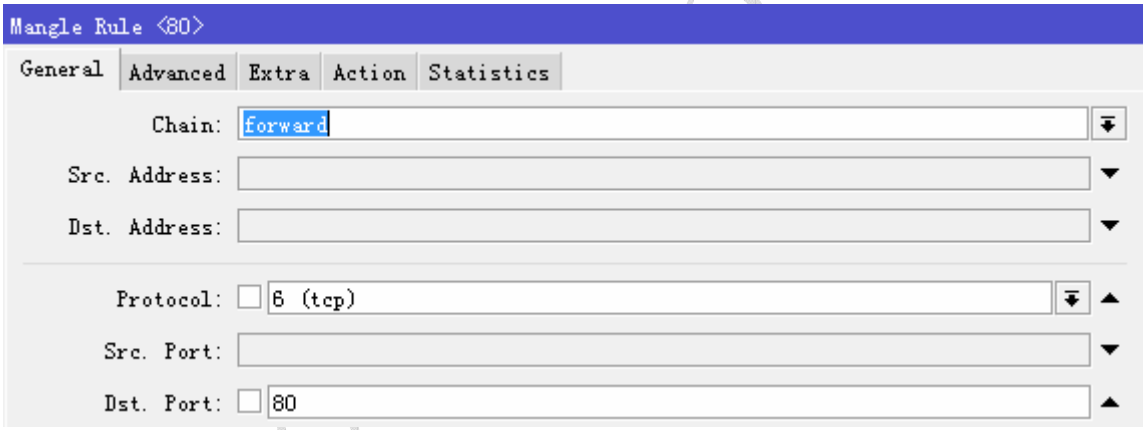
网页视频和浏览区分

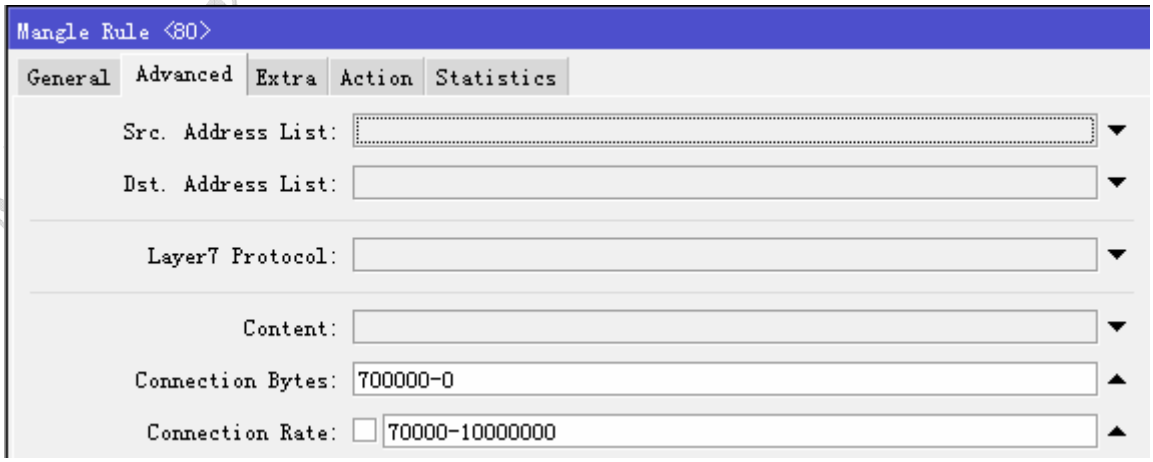
网页视频和网页浏览的区分，仍然采用 connection-rate 参数，区分网页的规则一共 4 条，如下图：

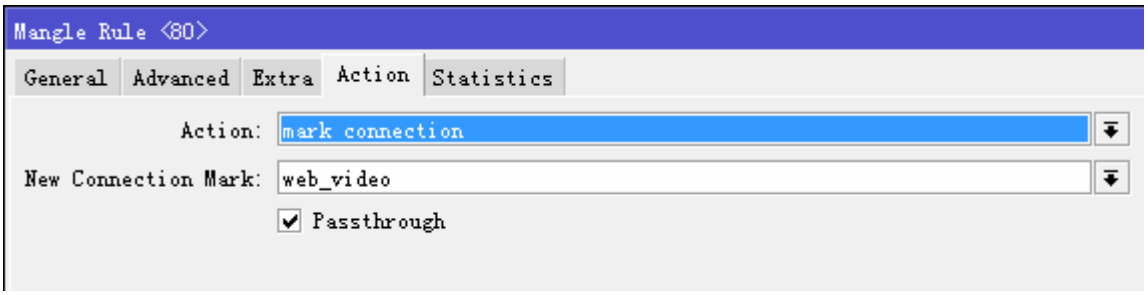


4 条规则里关键是在第一条连接标记，用于区分网页视频的速率，

```
chain=forward action=mark-connection new-connection-mark=web_video passthrough=yes protocol=tcp dst-port=80 connection-bytes=700000-0 connection-rate=70k-10M
```

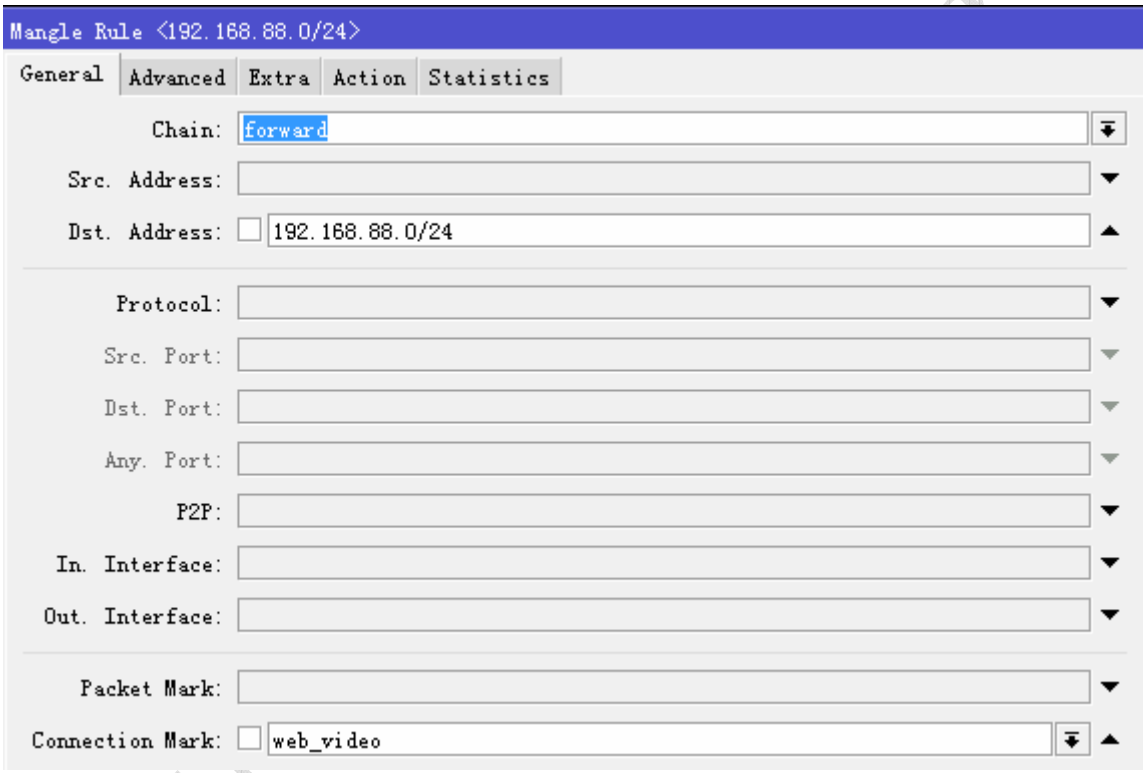


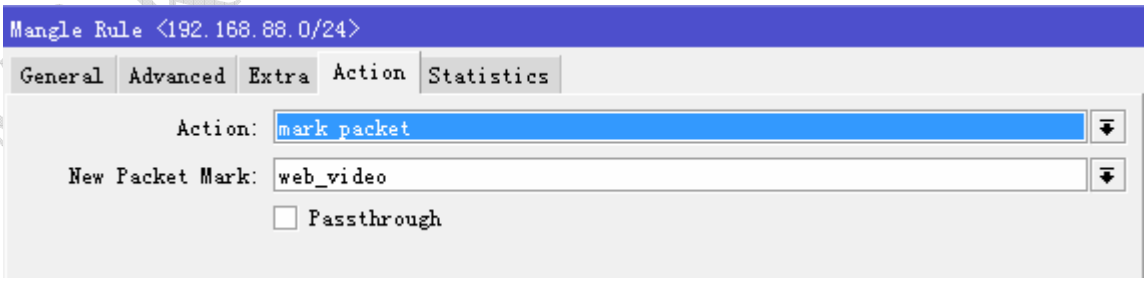




标记所有 tcp/80 端口，并设置 connection-bytes 为 700k 范围内，即初次使用了 700K 的字节，之后仍然保持 70kbps~10Mbps 的速率，就认为是网页视频，并标记名称为 web_video

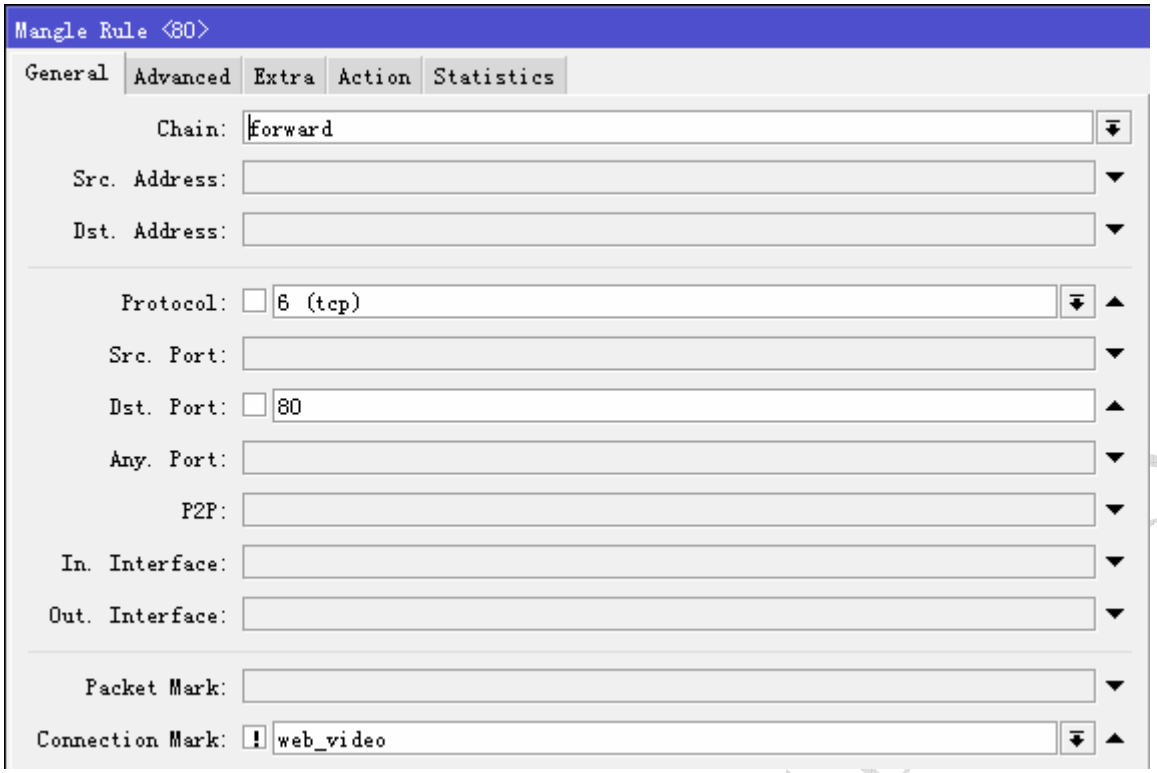
接下来的一条规则就只需要从这个连接里提取数据包，并指明目标地址是 192.168.88.0/24 的下行数据，action=mark-packet，取名标记为 web_video，注意这个 web_video 和连接的 web_video 并不相同，一个是连接一个是数据，这里的 passthrough=no，因为已经被数据标记处理，不需要交给后面的规则

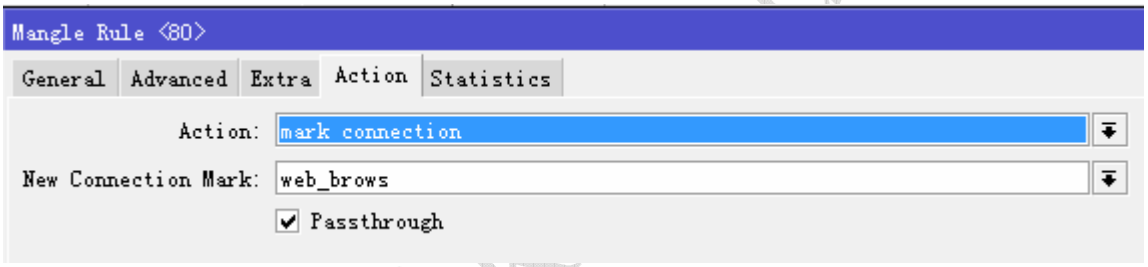




接下来我们标记网络浏览连接，仍然是是 tcp/80，但这里的 connection-mark 设置为非 web_video 的连接

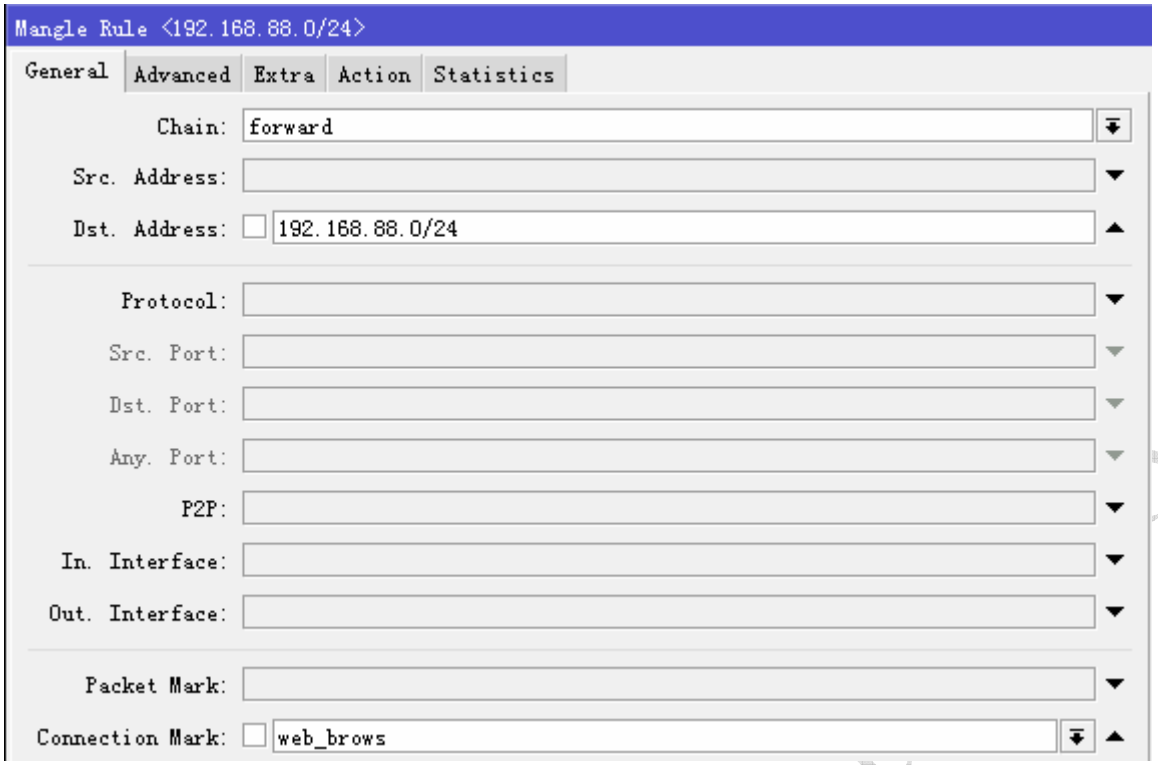
```
chain=forward action=mark-connection new-connection-mark=low_web passthrough=yes protocol=tcp
dst-port=80 connection-mark=!webhighspeed
```

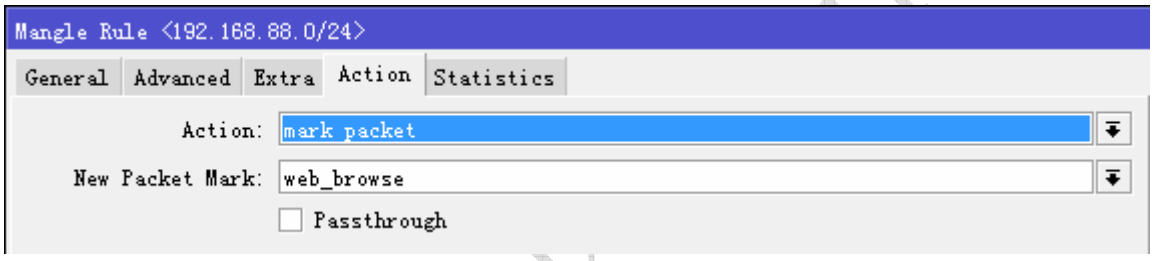




标记连接名称为 web_brows，并传递给后面的数据包标记规则

同样的方式，提取 web_brows 的数据包标记，选择目标地址并同样取名为 web_brows，passthrough=no

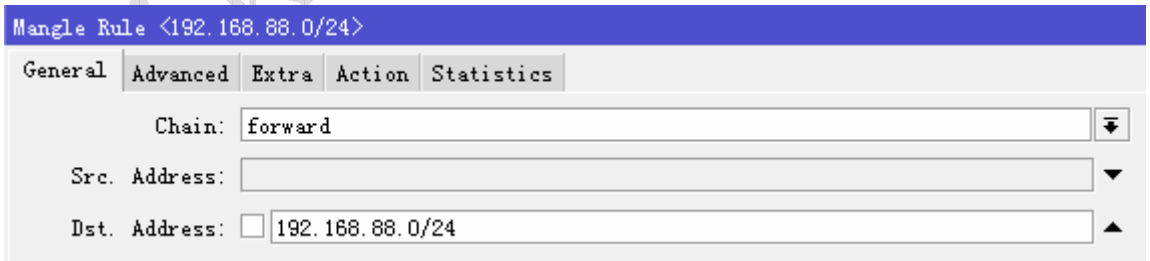


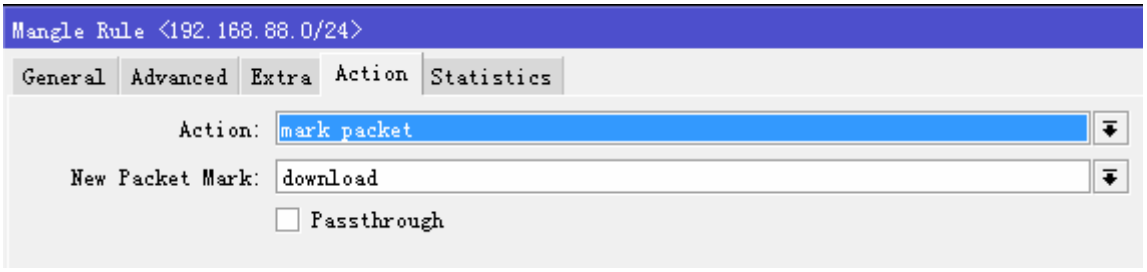


其他下载数据标记

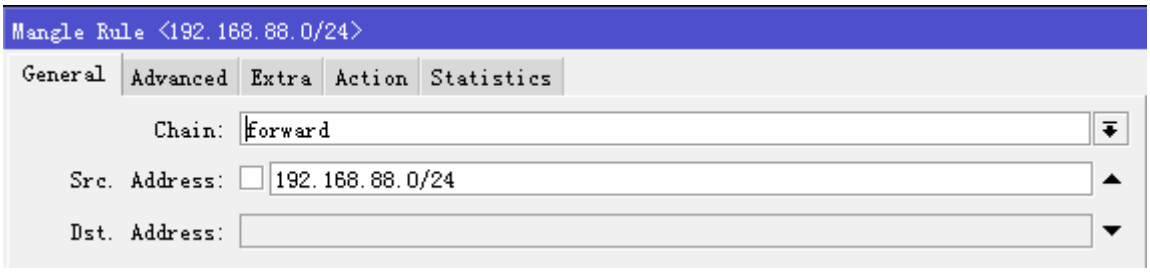
这样我们已经将部分游戏、网页视频和网页浏览区分出来，剩下的被认为是其他下载数据，直接做数据包标记

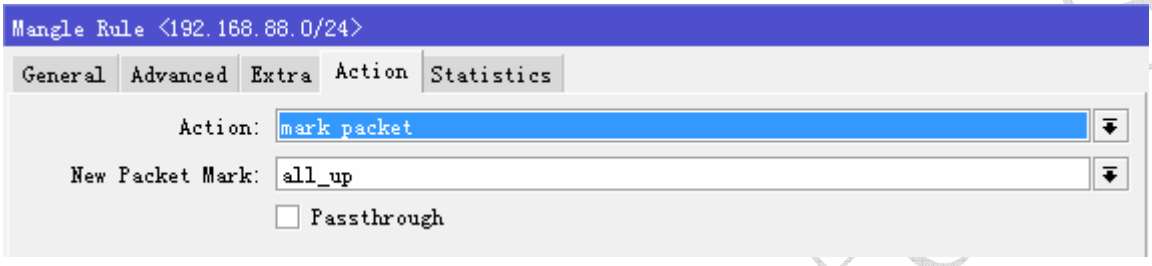
```
chain=forward action=mark-packet new-packet-mark=download passthrough=no
dst-address=192.168.88.0/24
```





最后一个规则则是标记上行数据，我们主要处理数据优先是下行，上行需要的带宽相对较小，我们只需要做一次性的标记，取名标记为 all_up，通过后面的 HTB 的 PCQ 给一个适合的带宽即可。

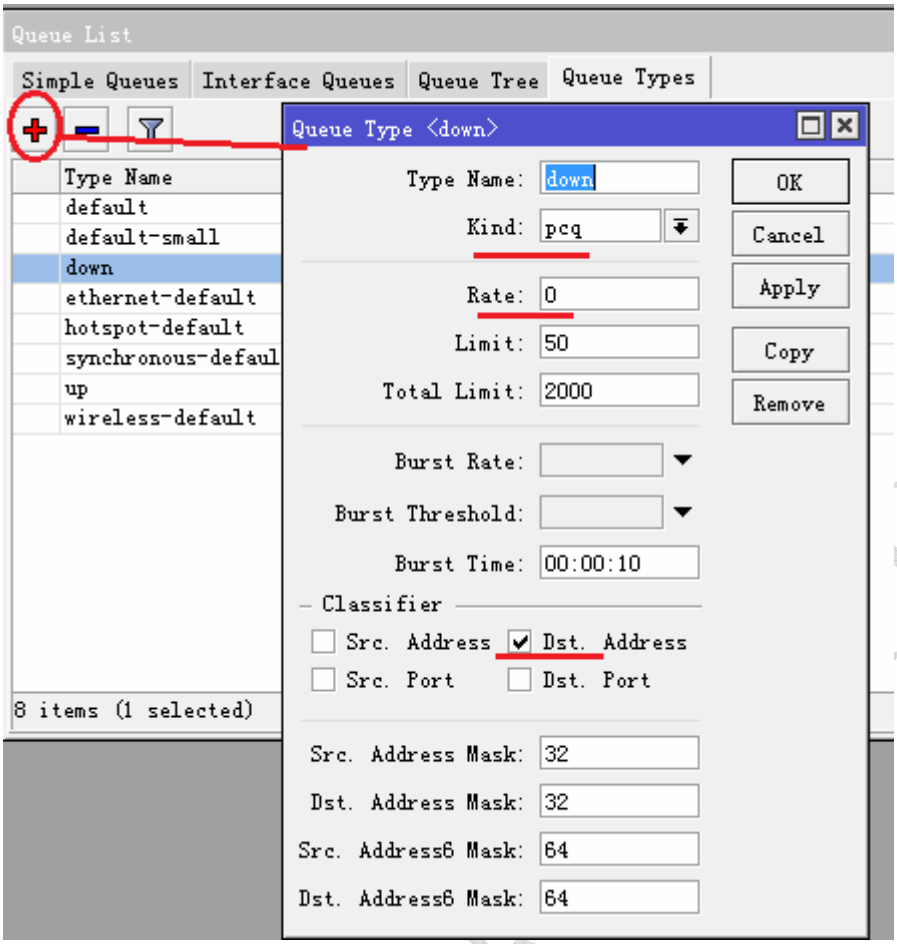




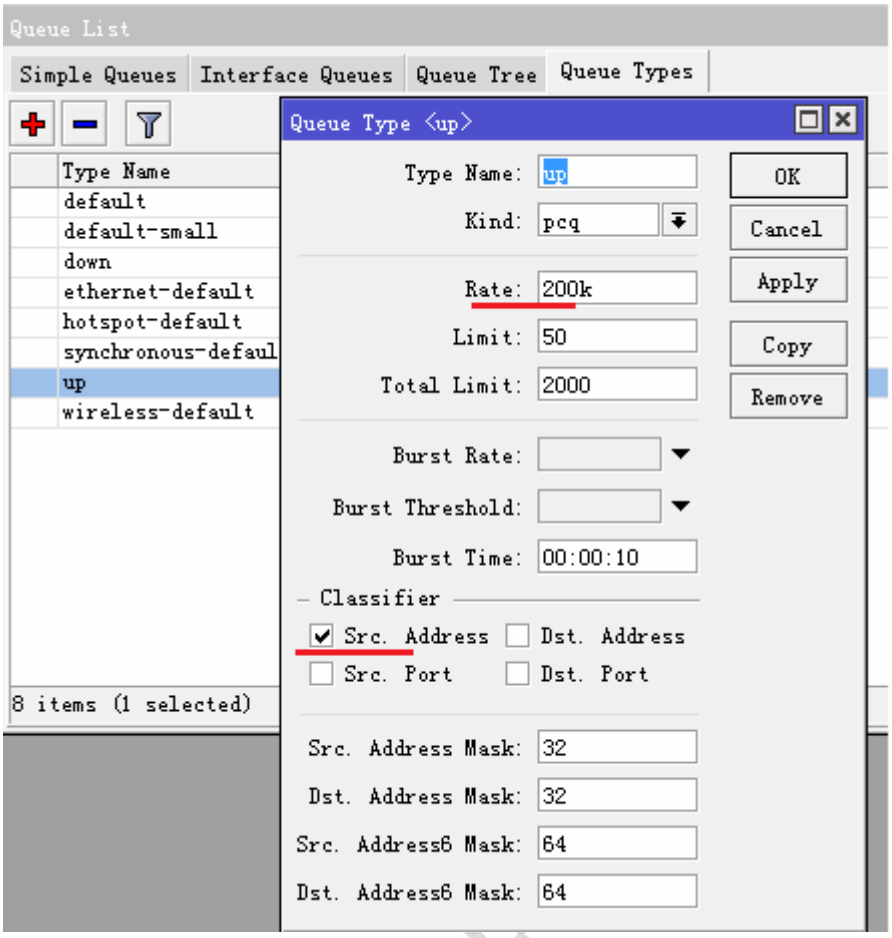
HTB 和 PCQ 设置

HTB 令牌桶方式控制各个数据的优先级，并通过 PCQ 动态分配带宽，我们进入 queue 的 tree 设置 HTB，但我们首先还是要先设置 PCQ 参数

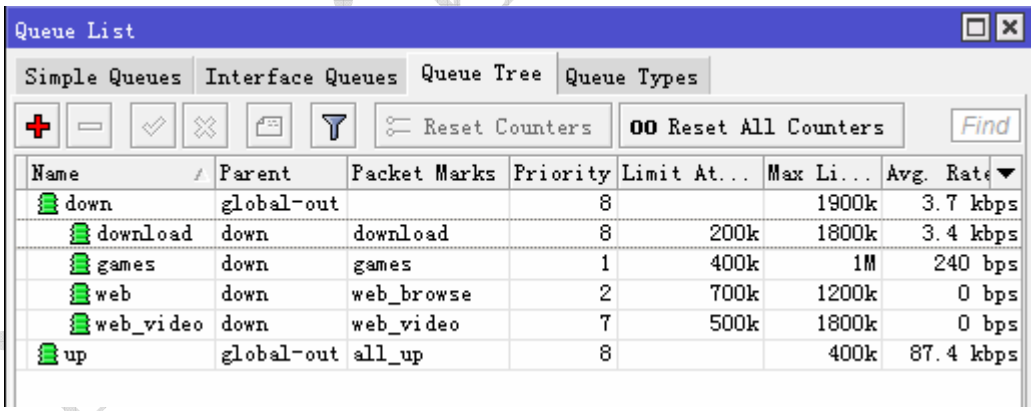
进入 queue type 添加 PCQ 规则，我们添加一个 down 的 PCQ，kind=pcq，rate 设置每台主机的带宽，这里我们选择 0，即动态分配带宽，你也可以设置一个固定值，以 k 或者 m 为单位，选择 dst-address 为下行的规则



添加上行带宽控制规则，这里我们将上行的 PCQ 带宽参数，rate=200k，即每台主机的上行为 200k（由于本人使用的是 2M ADSL，上行只有 512kbps，所以上行设置较小），上行带宽类型为 src-address



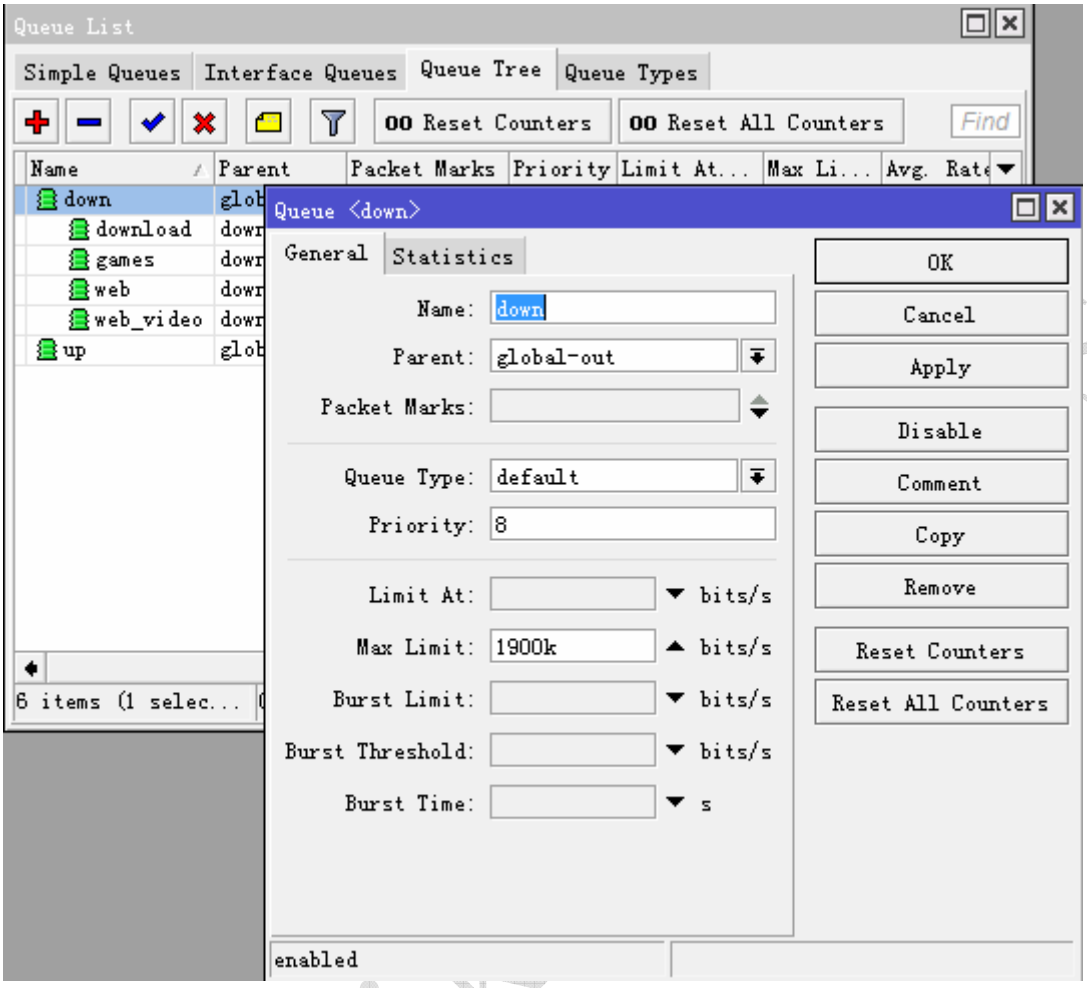
PCQ 设置完成后，我们接下来就需要配置 HTB，HTB 是由父级队列和子队列组成，我们只对下行做 HTB，上行数据则采用普通的 PCQ 限速（没有子队列的规则不能称为 HTB），如下图



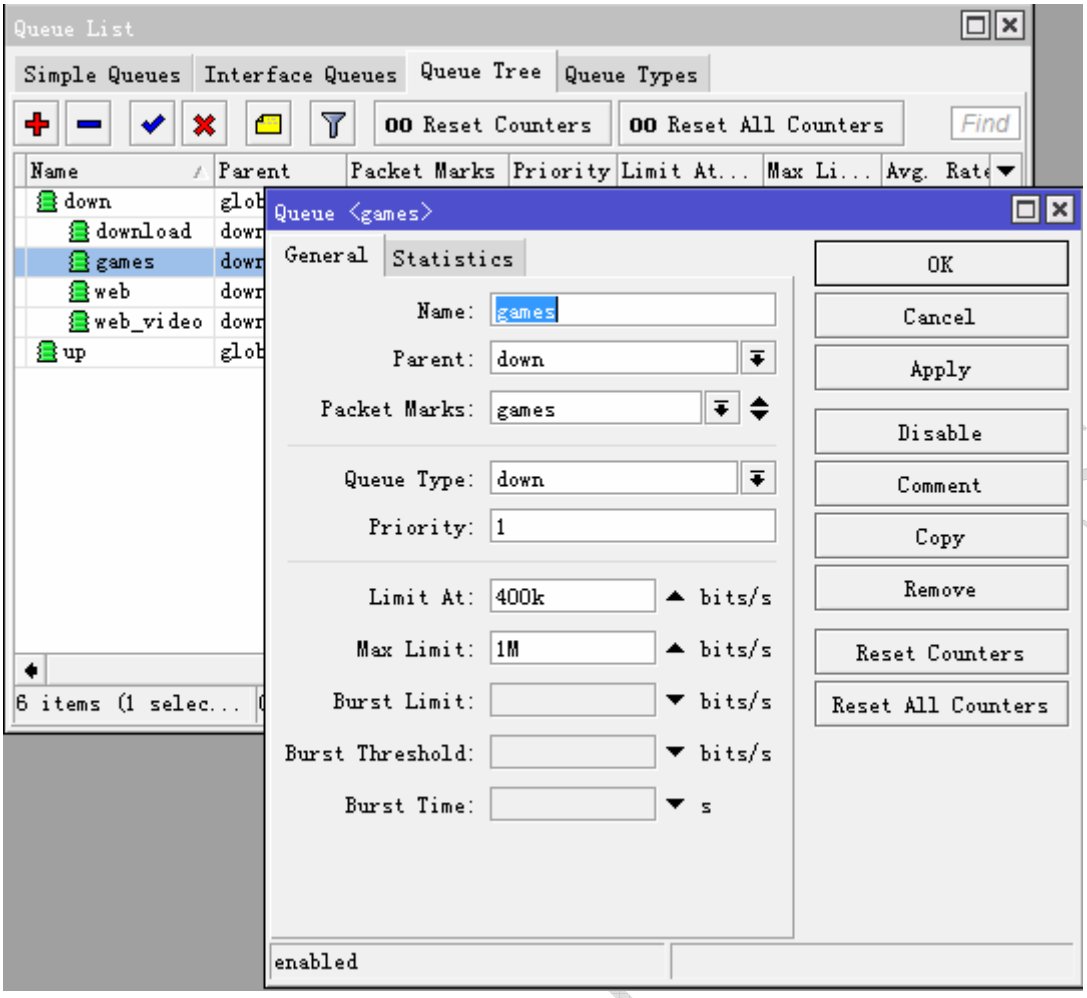
配置原则说明：

1、down 是下行父级队列，负责总带宽分配，up 是上行队列，仅控制上行。
2、down 和 up 都是通过 forward 链表标记的，属于 global-out，所以我们使用的 parent 为 global-out
3、down 父级队列下有 4 个子队列，分配是从属于 down，从父级获取带宽，子队列自己有各自的优先级 priority，1 为最高，8 最低，游戏优先级最高 1，web 优先级其次 2，web 视频 7，下载数据最低 8.
4、子队列的优先级与父级队列是不能比较，max-limit 是最大能获取的带宽，limit-at 为保障带宽，所有子队列 limi-at 之和小于等于父级 max-limit 带宽
5、这里是 2M 的 ADSL，我们分配父级带宽不能将 2Mbps 分配完，并须预留一部分作为缓冲，这里下行是 1900kbps，上行 400kbps（如果你是 10Mbps 的带宽建议，预留 2M 作为缓冲）

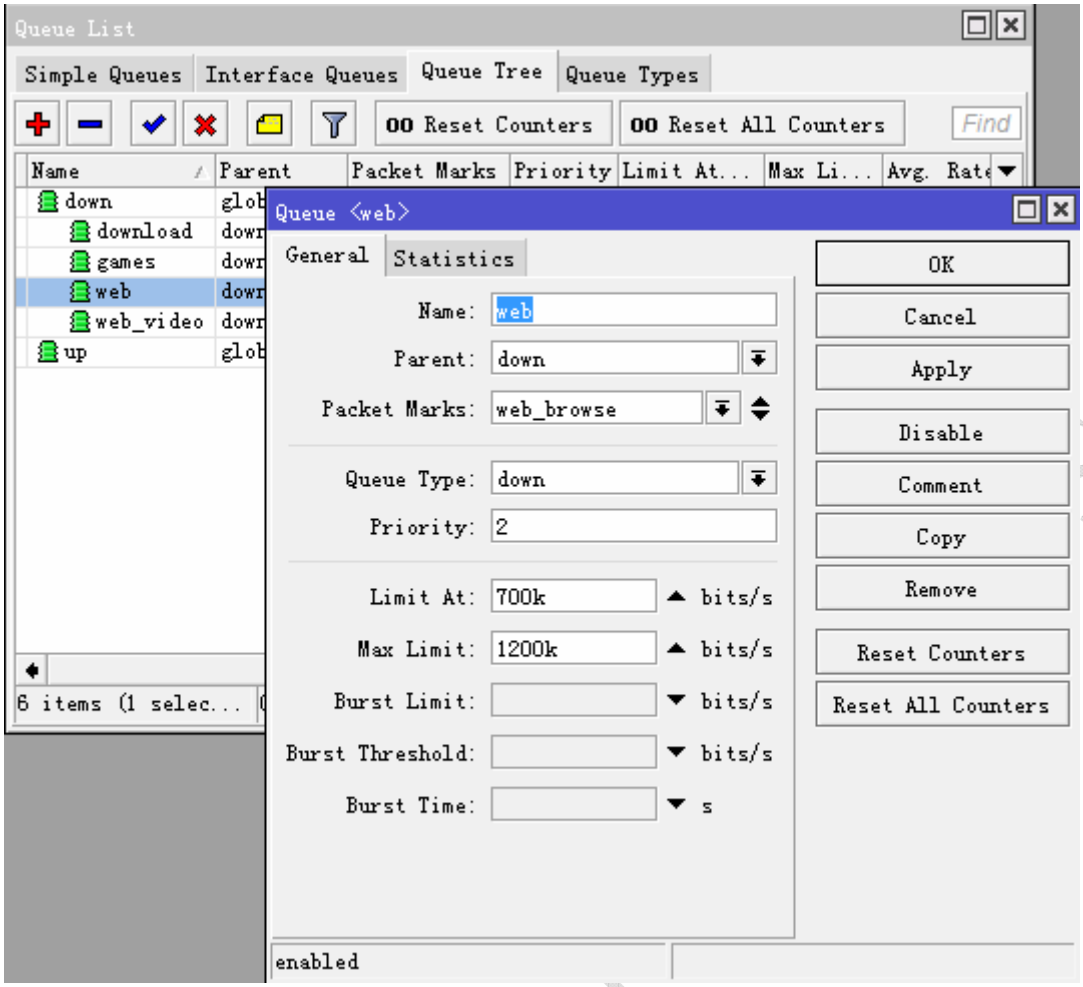
根据以上原则，添加 HTB 的规则的思路就明确了，首先添加父级队列，这里我们不设置任何 packet marks 标记，仅作为父级带宽分配给下面的子队列



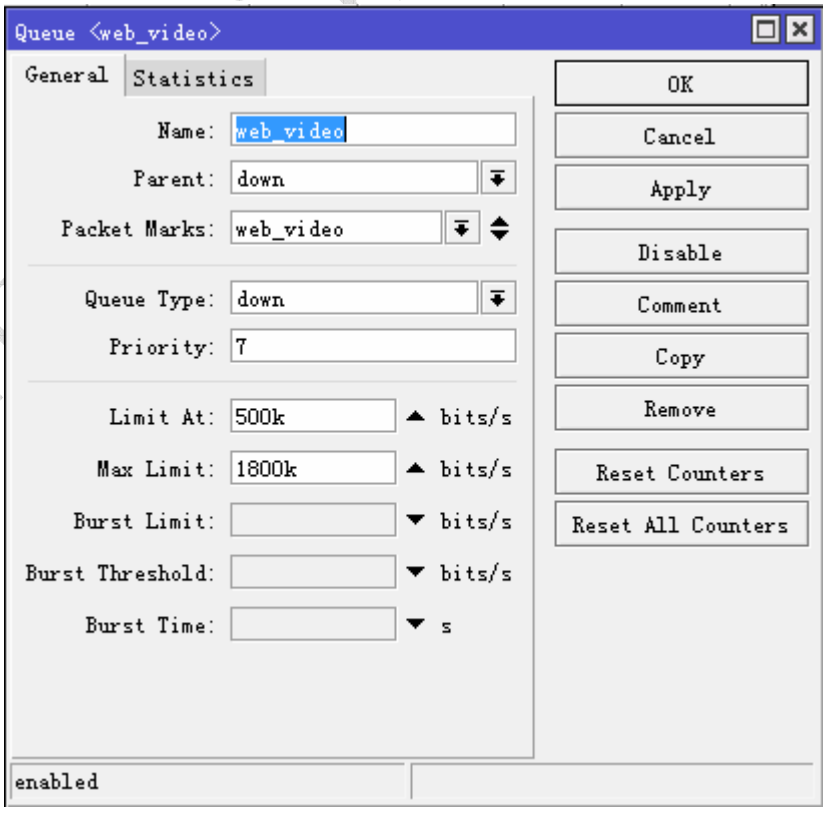
我们添加子队列，首先添加游戏子队列，我们需要设置父级 parent=down，需要从 down 获取带宽，packet-mark 为之前标记的 games，选择 queue type 为刚才的 PCQ 规则 down（给每个主机动态分配带宽），priority 优先级为 1 最高，我们给游戏分配 1M 带宽，最低保证 400kbps 带宽，如果有剩余带宽可以从父级获取得到最大 1Mbps



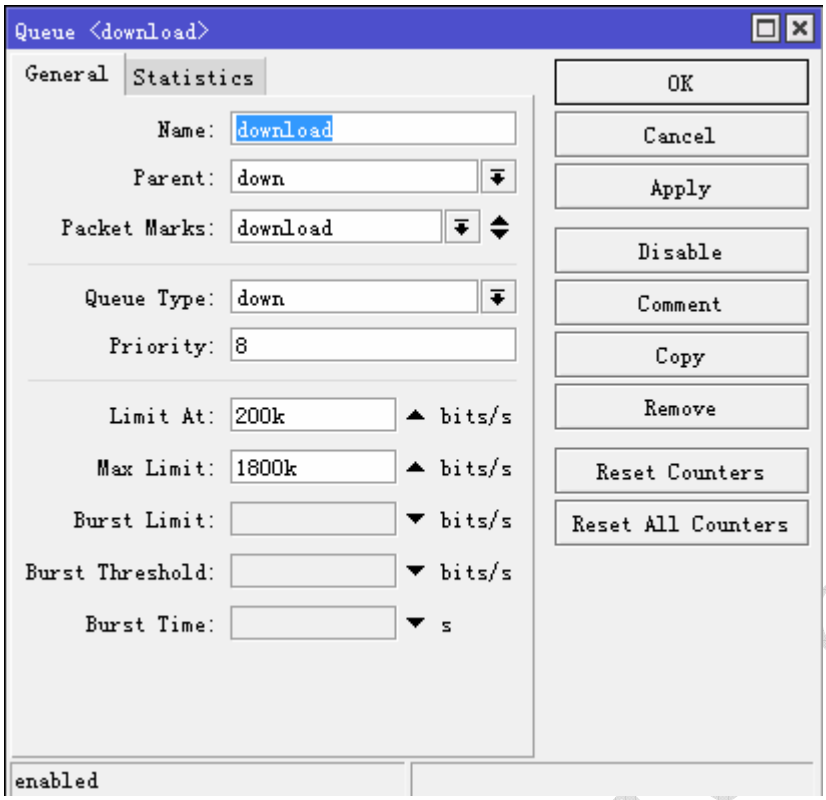
其次我们添加网页浏览的流控，使用同样的方法，priority 为 2，max-limit 为 1200k，保证最低获得 700k



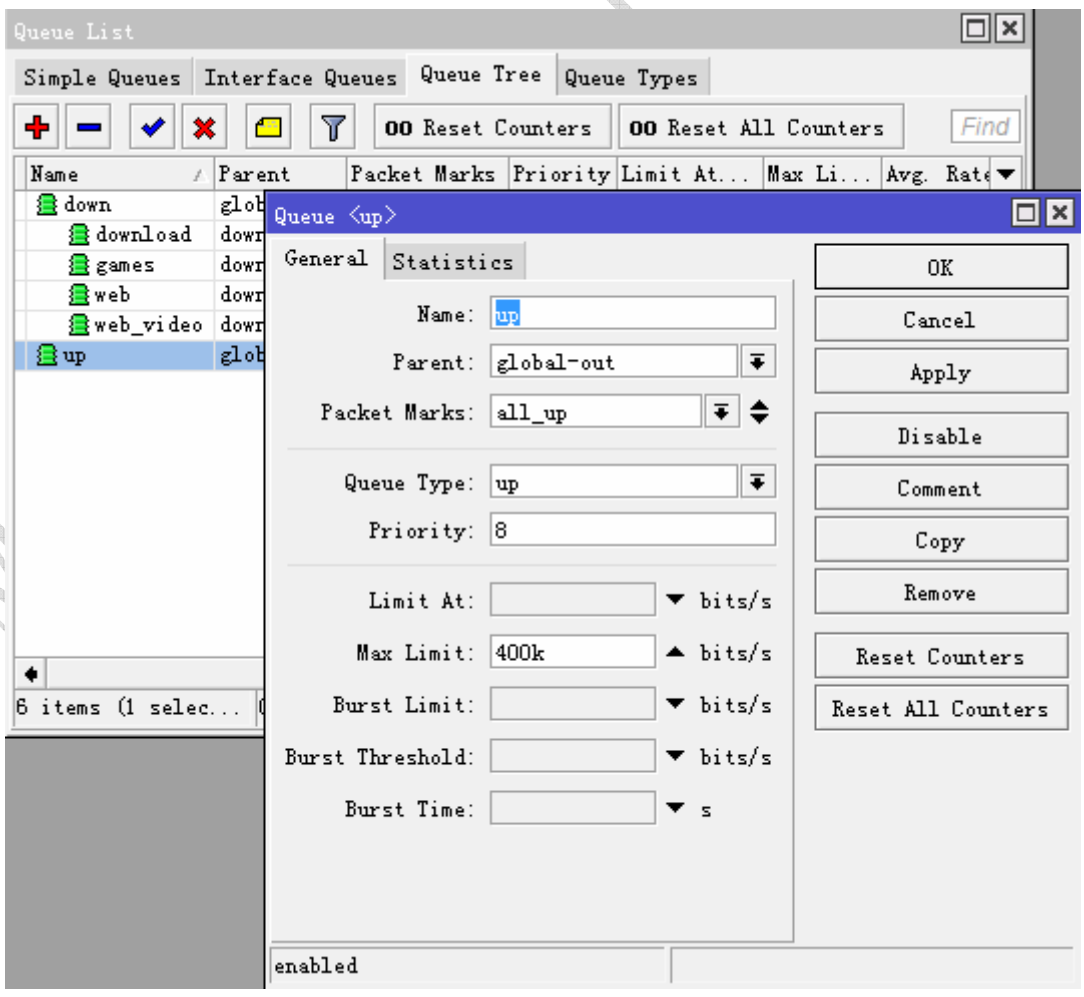
网页视频的优先级为 7，最大可以获得 1800k，最低保证 500k



其他下载数据优先级最低 8，最大带宽为 1800k，最低保证 200k



上行带宽的 PCQ 规则，取名为 up，因为上行是独立的 PCQ 规则，所以父级是 global-out，选择 packet-mark 为 all_up，queue-type 为刚才的 PCQ 规则 up，每台主机 200kbps 带宽，max-limit 为 400k（因为 2M 的 ADSL 上行带宽为 512k，保留一部分带宽）



第十二章 网络地址翻译 nat

网络地址翻译(NAT)是一种当 IP 包通过路由器时取代其源和(或)目标地址的路由协议。它通常被用来启用专用网络的多个主机使用一个公用 IP 地址访问因特网。

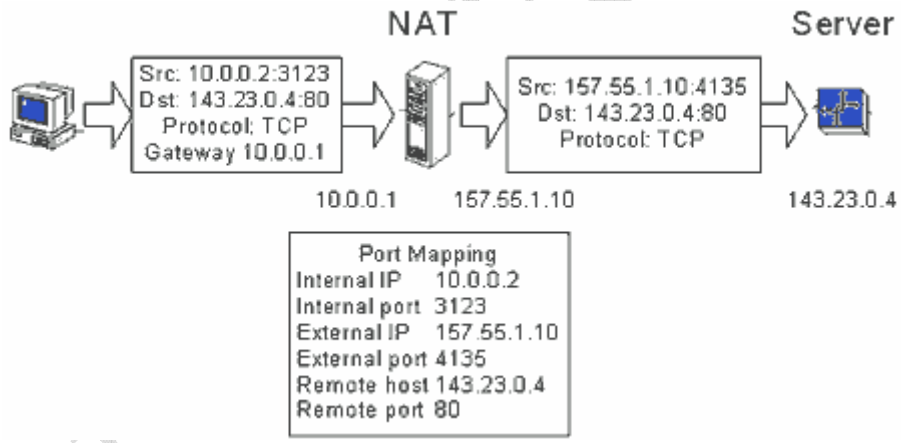
规格说明

功能包要求: ***system***
等级要求: *Level1，Level3*
操作路径: ***/ip firewall nat***
标准及技术: IP， RFC1631， RFC2663
硬件使用: 提升 CPU 和内存有助于 NAT 规则的处理

12.1 nat 介绍

网络地址翻译是一种允许本地网络主机使用一段 IP 地址进行本地通信，使用另一段 IP 地址进行外部通信的因特网标准。一个使用网络地址翻译的局域网就被称为 natted(已翻译)网络。为了使网络地址翻译进行工作必须在每个 natted 网络都有一个 nat 网关。nat 网关的作用就是在数据包进/出局域网时重写 IP 地址的作用。

输出数据包转换的示例:



网络地址翻译包括两种类型:

- 源网络地址翻译或者 srcnat。这种类型的网络地址翻译工作在从一个 natted 网络产生的数据包上。nat 路由器在 IP 包通过它的时候用一个新的公网 IP 地址代替了其私有源地址。相反的操作适用于响应包从相反方向通过路由器时。
- 目标网络地址翻译或者 dstnat。 这种类型的网络地址翻译工作在到达一个 natted 网络的数据包上。它通常用于使一个私有网络上的主机能够被因特网访问。dstnat 路由器在 IP 包通过该路由器到达私有网络时替换了 IP 包的目标 IP 地址。

nat 缺点

在一个使用了网络地址翻译的路由器背后的主机并不拥有真实的端对端的连接。因此一些因特网协议就不在有网络地址翻译的情况下工作。一些来私有用网络外部或者无连接协议如 UDP 协议且需要 TCP 连接初始化的服务将被打断。此外，一些协议内在与 NAT 不兼容，一个鲜明的事例就是 IPsec 中的 AH 协议。

重定向与伪装

重定向和伪装分别是目的 nat 和源 nat 的特殊形式。重定向类似与普通的目的网络地址翻译就好比伪装类似与源网络地址翻译——伪装是一种不需要指定 **to-addresses** 的源网络地址翻译的特殊形式——对外接口地址将被自动使用。重定向同理——进入接口地址将被使用。注意，**to-ports** 对于重定向规则来说很有意义——这就是在路由器起上处理这些请求的的服务端口。（比如：web 代理）

当数据包进行了目的网络地址翻译（dst-nat）时（不论 action=nat 或者 action=redirect），目的地址都将改变。有关地址翻译的任何信息（包括初始的目的地址）将被保存在路由器的内部维护表。当 web 请求被重定性到路由器的代理端口时，工作在路由器上的透明 web 代理将访问从内部表这个信息并从其中取得 web 服务器的地址。如果你正在对几个不同的代理服务器进行目的网络地址翻译，那你将不会从 IP 包头找到 web 服务器的地址，因为 IP 包的目的地址之前是 web 服务器的地址但现在已经变成了代理服务器的地址。从 HTTP/1.1 开始在 HTTP 请求中出现了特殊的可以告知 web 服务器地址的包头，于是代理服务器使用它取代了 IP 包的目的地址。如果没有这样的包头（如：老版本的 HTTP），代理服务器将不能确定 web 服务器地址也将无法工作。

这也就是说，对 HTTP 流从一个路由器到其他一些透明代理服务器进行正确的透明的重定向是有可能的。只有在路由器本身添加透明代理并配置才是正确的方法，因此你的“真实的”代理就是上级代理。这种情况下你的“真实的”代理再也不用是透明的，因为在路由器上的代理将成为透明的并将向“真正的”代理转交代理方式请求（根据标准，这些请求包括了所有必须的 web 服务器信息）。

属性描述

action (accept | add-dst-to-address-list | add-src-to-address-list | dst-nat | jump | log | masquerade | netmap | passthrough | redirect | return | same | src-nat; 默认: **accept**) -如果数据包与规则匹配 action 将启用

- **accept** - 接收数据包。不进行任何动作。例如：数据包通过而且没有其他任何适用于它的规则
- **add-dst-to-address-list** - 向 **address-list** 参数指定的地址表中添加 IP 包的目的地址
- **add-src-to-address-list** - 向 **address-list** 参数指定的地址表中添加 IP 包的源地址
- **dst-nat** - 用 **to-addresses** 及 **to-ports** 参数指定的变量取代 IP 包的目的地址
- **jump** - 跳转到由 **jump-target** 参数指定的链
- **log** - action 的每个匹配都将对系统日志添加一条消息
- **masquerade** - 以一个路由策略自动分配的 IP 地址取代 IP 包的源地址
- **netmap** - 创造一个 IP 地址从一端到另一端的静态 1：1 映像。通常用于分配公用 IP 地址到专用内网的主机上
- **passthrough** - 忽略次条规则并转到下一个规则
- **redirect** - 把 IP 包的目的地址替换成一个路由器的本地地址
- **return** -返回到跳转发生的链
- **same** - 从允许范围内分配给特定客户每个连接相同的源/目的 IP 地址。这种情况通常用于来自期望相同客户的相同客户地址对多重连接的服务。
- **src-nat** - 把 IP 包的源地址替换成由 **to-addresses** 和 **to-ports** 参数指定的值

address-list (*名称*) -指定地址列表的名称以收集使用了 **action=add-dst-to-address-list** 或 **action=add-src-to-address-list** 动作规则的 IP 地址。

address-list-timeout (*时间*; 默认: **00:00:00**) - 在 address-list 参数指定的地址列表删除地址之后的时间间隔。与 **add-dst-to-address-list** 或 **add-src-to-address-list** 动作一起起使用

00:00:00 - 从地址列表中永久删除

chain (dstnat | srcnat | *name*) – 选择或者定义一个规则的链。由于不同的数据流通过不同的链，所以为新规则选择正确的链必须很小心。如果输入一个与默认链名（srcnat 和 dstnat）不匹配，那么会生产一个新的链。

dstnat - 在这个链中的规则会在路由前被应用。代替 IP 包目的地址的规则应放在这里。

srcnat - 在这个链中的规则会在路由后被应用。代替 IP 包源地址的规则应放在这里。

comment (*文本*) - 对规则的描述性注解。一条注解能被用于从脚本中识别规则。

connection-bytes (*整型-整型*) - 当且仅当一定给定量字节从特定连接传输时与数据包进行匹配。

0 - 代表无穷大。例如：**connection-bytes=2000000-0** 如果大于 2MB 数据从相关连接传输就与规则匹配。

connection-limit (*整型*, 子网掩码) - 限制每个地址或地址群的连接限度。

connection-mark (*名称*) - 与通过 mangle 机制标记的特定连接数据包进行匹配

connection-type (ftp | gre | h323 | irc | mms | pptp | quake3 | tftp) - 与基于连接跟踪助手信息的相关连接的包进行匹配。相关连接助手必须在**/ip firewall service-port** 下启用

content (*文本*) - 文本数据包必须按顺序排列以与匹配规则

dst-address (*IP 地址*/掩码 | *IP address-IP address*) - 指定 IP 包的目的地址范围

address/netmask – 对合法网络地址的换算，例如：1.1.1.1/24 被转换为 1.1.1.0/24

dst-address-list (*名称*) - 在用户自定义的地址列表中匹配数据包的目的地址

dst-address-type (unicast | local | broadcast | multicast) - 在 IP 包的目的地址类型中匹配其中之一

unicast - 用于点对点传输的 IP 地址。这种情况仅限于一个发送者和一个接受者

local -与分配到路由器接口的地址匹配

broadcast - 这个 IP 包从 IP 子网的一个点到其他所有点发送信号

multicast - 这种类型的 IP 地址负责从一个或多个点到其他一系列点的传输

dst-limit (*整型*/*时间*{0,1},*整型*, dst-address | dst-port | src-address{+},*time*{0,1}) - 在每个目的 IP 或者每个目的端口库上限制每秒数据包绿（pps）。与 **limit** 匹配相反，每个目的 IP 地址/目的端口都有自己的限度。其选项如下（按出现次序）：

Count - 最大平均包率。以 pps 衡量，除非跟随在 **Time** 选项之后。

Time - 指定包率衡量的时间间隔

Burst - 以成组方式匹配的包数量

Mode - 包率限制分类方式

Expire - 指定已记录的 IP 地址/端口将被删除的过期时间，时间间隔。

dst-port (*整型*: 0..65535-*整型*: 0..65535{*}) - 目的端口数或范围

hotspot (*多选项:* from-client | auth | local-dst) - 从各种不同的 Hot-Spot 中匹配从客户获得的包。所有值都可以被取消。

from-client - 如果一个包来自于 HotSpot 客户则为真

auth - 如果一个包来自验证用户则为真

local-dst - 如果一个包拥有本地目的 IP 地址则为真

icmp-options (*整型*:*整型*) - 与 ICMP 的 Type:Code 域匹配

in-interface (*name*) - interface the packet has entered the router through

ipv4-options (any | loose-source-routing | no-record-route | no-router-alert | no-source-routing | no-timestamp | none | record-route | router-alert | strict-source-routing | timestamp) - 与 ipv4 标题选项匹配

any - 与 ipv4 选项中至少一个匹配

loose-source-routing - 与发射源路由选项的包进行匹配。次选项一般用于路由基于源提供信息的因特网数据报

no-record-route - 以无记录路由选项匹配包。次选项一般用于路由基于源提供信息的因特网数据报

no-router-alert - 以无路由警报选项匹配包

no-source-routing - 以无源路由选项匹配包

no-timestamp - 以无时间印章选项匹配包

record-route - 以记录路由选项匹配包

router-alert - 以路由警报选项匹配包

strict-source-routing - 以严密的源路由选项匹配包

timestamp - 以时间印章选项匹配包

jump-target (dstnat | srcnat*name*) - 将要跳转的目标链名称，如果使用了动作 **action=jump**
limit (*整型*/*时间*{0，1}，*整型*) - 按给定限度限制包匹配率。对于减少日志信息数量有用
Count - 最大平均包率。以 pps 衡量，除非跟随在 **Time** 选项之后。
Time - 指定包率衡量的时间间隔
Burst - 以成组方式匹配的包数量
log-prefix (*文本*) - 所有写入日志的信息都包含次中指定的前缀。与 **action=log** 一起使用。
nth (*整型*, *整型*: 0..15，*整型*{0,1}) - 与特定的由规则获取的第 N 个包匹配。16 个可用计数器之一可被用来计算包数
Every - 匹配每第 **Every+1** 个包。例如：如果 **Every=1** 那么规则匹配每第二个包
Counter - 指定要使用的计数器。
Packet - 以给定包的数量进行匹配。显然地，这个值必须在 **0** 和 **Every** 之间。如果这个选项用于一个给定的计数器，那么在这个选项里必须至少有 **Every+1** 个规则，以包含所有在 **0** 和 **Every** 之间的值
out-interface (*name*) - 离开路由器的包的接口
packet-size (*整型*: 0..65535-*整型*: 0..65535{0,1}) - 按字节匹配指定大小或大小范围的包
Min - 指定大小范围或独立的值的下限
Max - 指定大小范围的上限
phys-in-interface (*name*) -与添加到一个桥设备的桥端口物理输入设备匹配。仅在数据包从桥到达并通过路由器时有用
phys-out-interface (*name*) - 与添加到一个桥设备的桥端口物理输出设备匹配。仅在数据包从桥离开路由器时有用
protocol (ddp | egp | encap | ggp | gre | hmp | icmp | idrp-cmtp | igmp | ipencap | ipip | ipsec-ah | ipsec-esp | iso-tp4 | ospf | pup | rdp | rspf | st | tcp | udp | vmtp | xns-idp | xtp | *整型*) - 与由协议名称或编号指定的特定 IP 协议匹配。如果你想指定端口就应该进行这个配置。
psd (*整型*，*时间*，*整型*，*整型*) - 试图探测 TCP 及 UDP 扫描。建议对高号码端口分配低权重以减少被误判的频率，例如来自被动模式的 FTP 迁移
WeightThreshold - 来自不同主机且被作为端口扫描序列的带有不同目的端口的最新的 TCP/UDP 包的总权重值
DelayThreshold - 来自同意主机且被当作可能端口扫描子序列带有不同目的端口的包延迟
LowPortWeight - 特权目的端口（<=1024）的数据包权重值
HighPortWeight –非特权目的端口（<=1024）的数据包权重值
random (*整型*) - 以给定概率随机匹配包
routing-mark (*name*) - 对 mangle 标记的特定路由的包进行匹配
same-not-by-dst (yes | no) - 当选择要与 **action=same** 规则匹配的包的新源 IP 地址时指定是否对目的 IP 地址进行计数
src-address (*IP 地址*/子网掩码 | *IP 地址* - *IP 地址*) - 指定源 IP 包产生的地址范围。
src-address-list (*name*) - 与用户定义的地址列表中的数据包源地址匹配
src-address-type (unicast | local | broadcast | multicast) - 与 IP 包的源地址类型中的一个匹配
unicast - 用于点对点传输的 IP 地址。这种情况仅限于一个发送者和一个接受者
local -与分配到路由器接口的地址匹配
broadcast - 这个 IP 包从 IP 子网的一个点到其他所有点发送信号
multicast - 这种类型的 IP 地址负责从一个或多个点到其他一系列点的传输
src-mac-address (*MAC address*) - 源 MAC 地址
src-port (*整型*: 0..65535-*整型*: 0..65535{*}) - 源端口数或范围
tcp-mss (*整型*: 0..65535) - 与 IP 包的 TCP MSS 值匹配
time (*时间*-*时间*，sat | fri | thu | wed | tue | mon | sun{+}) - 允许产生基于数据包到达时间和日期的过滤器，或者对于本地产生的数据包的离开时间和日期
to-addresses (*IP address*-*IP address*{0,1}; default: **0.0.0.0**) -取代初始 IP 包地址的地址或地址范围
to-ports (*整型*: 0..65535-*整型*: 0..65535{0,1}) - 取代初始 IP 包端口的端口或端口范围

tos (max-reliability | max-throughput | min-cost | min-delay | normal) - 对 IP 头服务类型（ToS）域的值指定一个匹配

max-reliability – 最大的可靠性 (ToS=4)
max-throughput – 最大的吞吐量 (ToS=8)
min-cost – 最低的成本代价(ToS=2)
min-delay – 最小的延迟 (ToS=16)
normal – 普通服务 (ToS=0)

12.2 源地址 nat

如果你想在 ISP 给你的 10.5.8.109 地址后“隐藏”你的 192.168.0.0/24 的专用局域网，你应该使用 MikroTik 路由器的源网络地址翻译特性。当数据包通过路由器时，伪装将把从 192.168.0.0/24 产生的源 IP 地址和包端口改变成路由器的 10.5.8.109 地址。为了使用伪装，必须向 nat 配置中添加一个带有“隐藏”动作的的源网络地址翻译规则：

```
/ip firewall nat add chain=srcnat action=masquerade out-interface=Public
```

所有从 192.168.0.0/24 出去的向外连接都将使用路由器的 10.5.8.109 作为源地址，1024 作为源端口。因特网将不可能访问本地地址。如果你允许对本地网络服务器访问，你应该使用目的网络地址翻译（nat）。

RouterOS 支持两种隐藏私有网络方式，‘masquerade’与’src-nat’都是改变源 IP 地址或一个数据包的端口，Masquerade 和 source nat 典型的应用都是将私有网络隐藏在一个或多个外网后，设置一个新的源地址 nat

- ‘masquerade’使用的是路由器默认的 IP 地址
- ‘src-nat’需要指定‘to-address’

Masquerade 操作

```
 add chain=srcnat src-address=192.168.0.0/24 action=masquerade out-interface=WAN
```

Src-nat 操作

```
 add chain=srcnat src-address=192.168.0.0/24 action=src-nat to-address=10.5.8.109
out-interface=WAN
```

12.3 目标地址 nat

如果你想使用公网 IP 地址 10.5.8.200 访问本地地址 192.168.0.109，你应该使用 MikroTik 路由器的目的地址翻译特性。同样地，如果你允许本地服务器与公网 IP 进行通信，你就需要使用源地址翻译。

对公用接口添加公用 IP：

```
/ip address add address=10.5.8.200/24 interface=Public
```

添加允许外部网络访问本地服务器的规则：

```
/ip firewall nat add chain=dstnat dst-address=10.5.8.200 action=dst-nat \
       to-addresses=192.168.0.109
```

添加规则使本地服务器能够与外部网络通信，并将其源地址翻译为 10.5.8.200

```
/ip firewall nat add chain=srcnat src-address=192.168.0.109 action=src-nat \
       to-addresses=10.5.8.200
```

dst-nat 数据转移

通过使用 dst-nat 操作转移 IP 数据或端口到指定的主机上，如我们可以将内网所有访问 tcp/80 端口的数据转移到另外一个主机的 192.168.0.100

```
/ip firewall nat chain=dst-nat protocol=tcp  dst-port=80 action=dst-nat
to-address=192.168.0.100
```

dst-nat 数据重定向

Redirect 是改变目标 IP 地址或一个目标 IP 数据的端口，指定访问数据转移到本地。与 dst-nat 不同的是 Redirect 不需要指明“to-address”，一个将 tcp/80 端口重定向到本地的测试

```
    chain=dstnat  action=redirect protocol=tcp dst-port=80
```

1：1nat 实例

如果你想从公用 IP 子网 11.11.11.1/32 访问本地的 2.2.2.2/32，你应该使用目的地址翻译以及源地址翻译特性设置 **action=netmap**。

```
/ip firewall nat add chain=dstnat dst-address=11.11.11.1 \
       action=netmap to-addresses=2.2.2.2

/ip firewall nat add chain=srcnat src-address=2.2.2.2 \
       action=netmap to-addresses=11.11.11.1
```

12.4 连接状态

操作路径：***/ip firewall connection***

连接追踪用于维护连接状态信息，例如源目的 IP 地址和端口，连接状态，协议类型和超时。特定连接的状态包含：**estabilished** 意思即数据包是已知连接的一部分，**new** 意思为数据包开启了一个新连接，**related** 意为数据包开始了一个新连接，但与一个已存在连接想联系，如 FTP 数据传输或 ICMP 错误信息，**invalid** 意为数据包不属于任何一个已建立的连接。

注： 连接追踪是对本地产生的数据包在 **prerouting** 链或者 **output** 链完成的。

另一个不能被过高估计的连接追踪功能是 nat 对其的需要。你应该清楚除非你启用了连接追踪否则 NAT 是不能完成的，对 P2P 协议识别也一样。连接追踪也在进一步处理前会从碎片中收集 IP 包。

/ip firewall connection 状态列表包含的最大数连接是由路由器的初始物理内存大小决定的。因此，例如一个 64M RAM 的路由器可以容纳最多 65536 连接的信息，128M RAM 的路由器就可以增加到 130000 以上。因此请确定你的路由器配置了足够量的内存以便可以适宜地处理所有连接。

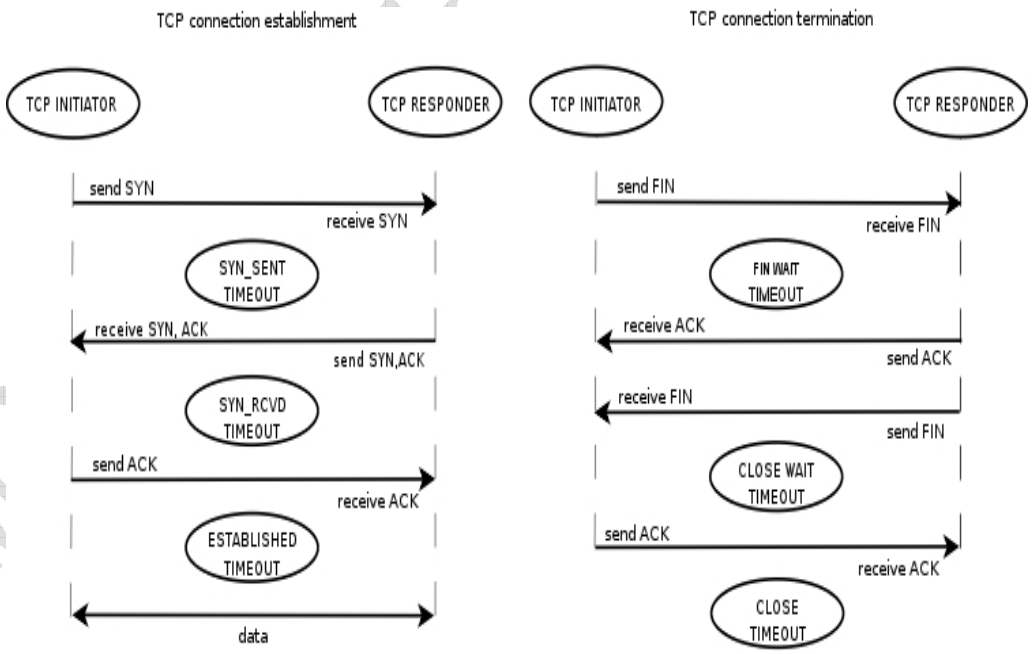
属性描述

connection-mark (*只读: 文本*) - mangle 中设置的连接标记
dst-address (*只读: IP address:port*) - 连接建立到的目的地址和端口
protocol (*只读: 文本*) – IP 协议名和序号
p2p (*只读: 文本*) – P2P 协议
reply-src-address (*只读: IP address:port*) - 从源地址和端口建立的响应连接
reply-dst-address (*只读: IP address:port*) - 连接建立到的目的地址和端口
src-address (*只读: IP address:port*) - 从源地址和端口建立的连接
tcp-state (*只读: 文本*) - TCP 连接状态
timeout (*只读: 时间*) - 直到连接超时的时间量
assured (*只读:* true | false) - 显示是否看到对该条登记的最后一个包的回应
icmp-id (*只读: 整型*) - 每个 ICMP 包都会在被发送时得到一个为其设定的 ID，并且当接收器收到了 ICMP 信息时，它会在新的 ICMP 信息内设定同样的 ID 以使发送器能识别回应并能够用适当的 ICMP 请求连接它。
icmp-option (*只读: 整型*) - ICMP 类型和代码域
reply-icmp-id (*只读: 整型*) - 包含已接收包的 ICMP ID
reply-icmp-option (*只读: 整型*) - 已接收包的 ICMP 类型和代码域
unreplied (*只读:* true | false) - 显示是否请求未被回应

连接跟踪

操作路径: ***/ip firewall connection tracking***

连接追踪提供了几个连接超时（timeout）。当特定的超时超过了相应的条目将会从连接状态列表中删除。下面的图描绘了典型的 TCP 连接建立和终端以及在这些处理过程中发生的 TCP 超时：



属性描述

count-curent (*只读: 整数*) - 在连接状态列表中记录的当前连接数
count-max (*只读: 整数*) - 取决于总内存量的连接状态列表，自动计算出最大连接数
enable (yes | no; 默认: **yes**) - 允许或禁止连接追踪，nat 被使用的情况下必须开启。

generic-timeout (*时间*; 默认: **10m**) - 连接列表中追踪既非 TCP 又非 UDP 包的条目的最大时间量将会在看到匹配此条目最后一个包之后存活
icmp-timeout (*时间*; 默认: **10s**) - 连接追踪条目将在看到 ICMP 请求后存活最的大时间量
tcp-close-timeout (*时间*; 默认: **10s**) – TCP 连接追踪条目在看到连接复位请求（RST）或来自连接释放初始化机连接终端请求确认通知（ACK）之后存活的最大时间
tcp-close-wait-timeout (*时间*; 默认: **10s**) – 当来自应答器的终端请求（FIN）之后连接追踪条目存活的最大时间
tcp-established-timeout (*时间*; 默认: **1d**) – 当来自连接初始化机的确认通知后连接追踪条目存活的最大时间
tcp-fin-wait-timeout (时间; 默认: **10s**) – 当来自连接释放初始化机的连接终端请求（FIN）后存后连接追踪条目存活的最大时间
tcp-syn-received-timeout (*时间*; 默认: **1m**) – 当匹配连接请求（SYN）之后连接追踪条目存活的最大时间
tcp-syn-sent-timeout (*时间*; 默认: **1m**) – 当来自连接初始化机的连接请求（SYN）后连接追踪条目存活的最大时间
tcp-time-wait-timeout (*时间*; 默认: **10s**) – 当紧随连接请求（SYN）的连接终端请求（FIN）之后或在看到来自连接释放初始化机的其他终端请求（FIN）之后连接追踪条目存活的最大时间
udp-timeout (*时间*; 默认: **10s**) – 当匹配此条目的最后一个包之后连接追踪条目存活的最大时间
udp-stream-timeout (*时间*; 默认: **3m**) - 在匹配此连接（连接追踪条目是确定的）的最后一个包的响应被看到之后连接追踪条目存活的最大时间。它用于增加对 H323，VoIP 等连接的超时。

*注:*最大超时值取决于在连接状态列表中的连接数量。如果在列表中连接数量大于：

- 连接的最大数量的 1/16，超时值将为 1 天
- 连接的最大数量的 3/16，超时值将为 1 小时
- 连接的最大数量的 1/2，超时值将为 10 分钟
- 连接的最大数量的 13/16，超时值将为 1 分钟

如果超时值超过了上面列出的值，那么将使用更小的值。如果连接追踪超时值小于数据包率，比如：在下一个包到达之前超时就过期了，那么 nat 和 statefull-firewalling 将停止工作。

注： tracking 功能被关闭，nat 功能也将会失效；如果你在不考虑启用 nat 功能情况，可以关闭掉 tracking。

第十三章 分类标记（Mangle）

mangle 允许对 IP 数据包做特殊的标记，mangle 是通过修改指定的 IP 数据包头字段，去标记 IP 数据包的特征 能标记端口、IP、协议、TCP 协议和相应的 IP 数据流。 Mangle 属于综合性功能，所以在路由、流量控制和其他相应功能中都会涉及到。

需要功能包: ***system***
需要等级: *Level1*
操作路径: ***/ip firewall mangle***
协议标准: IP

13.1 Mangle 介绍

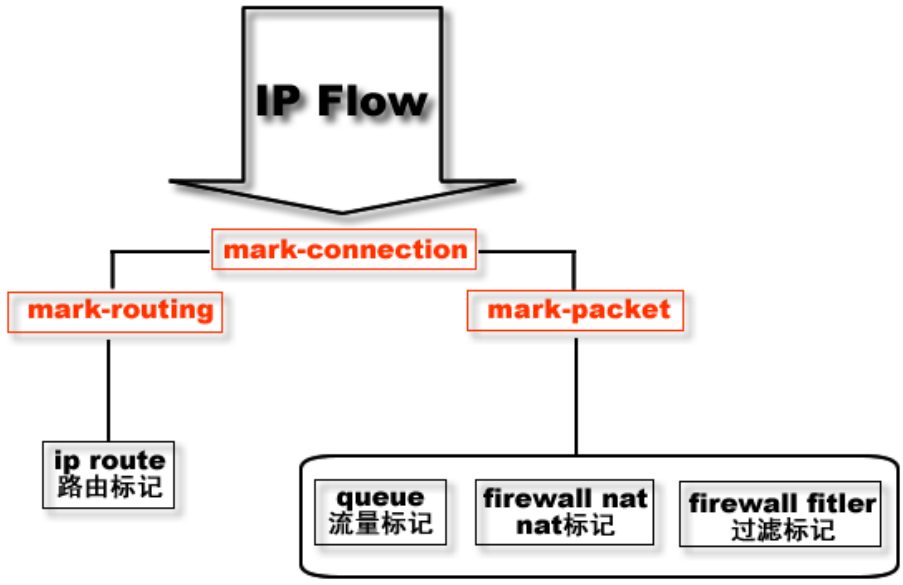
Mangle 是一种标记器，标记特殊的数据包等待将来处理。在 RouterOS 中许多其他的功能组件会使用到他，如 queue-trees 和 nat，他们识别到一个数据包了标记的便会做相应的处理。Mangle 标记仅存在于该路由器中，他们无法传输到网络中去。根据数据传输方式不同可以选择：

- **Prerouting：路由前，常用于标记策略和端口路由**
- **Input：进入路由器的数据**
- **Foreward：通过路由转发，用于修改 TTL、TCP-MSS 和流量控制规则**
- **Output：数据输出**
- **Prostrouting：路由后**

标记 IP 数据流的三种类型，这三种类型会在各种应用中多次出现，特别是 Queue 的流量控制和 ip route 的路由：

- **Mark-connection：标记所有 IP 流的连接**
- **Mark-packet：标记 IP 流中数据包**
- **Mark-routing：标记 IP 流中 IP 数据包的路由信息**

三种类型的关系，所有的在 IP 数据包传输前，首先需要通过建立 TCP/UDP 连接，进行传输。所以当数据通过 IP 流进入 Mangle 后，建立相应的连接标记，并从连接标记中提取数据包，做处理。图示如下：



13.2 Mangle 应用

Peer-to-Peer 传输标记

保证优质的网络连接，如 VoIP 和 HTTP 等为最优先级，将 P2P 的优先级设置为最低 RouterOS QOS 操作首先使用 mangle 标记不同类型的传输，然后把它们放入的 queues 做不同的限制。下面的事例是强迫 P2P 的总的传输不能超过 1Mbps，其他的传输连接则扩大连接带宽和优先级：

```
[admin@NAT] > /ip firewall mangle add chain=forward  p2p=all-p2p
action=mark-connection new-connection-mark=p2p_conn
[admin@NAT] > /ip firewall mangle add chain=forward connection-mark=p2p_conn
action=mark-packet new-packet-mark=p2p
[admin@NAT] > /ip firewall mangle add chain=forward  packet-mark=!p2p_conn
action=mark-packet new-packet-mark=other
[admin@NAT] > /ip firewall mangle print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=forward p2p=all-p2p action=mark-connection new-connection-mark=p2p_conn
```

```
 1   chain=forward connection-mark=p2p_conn action=mark-packet new-packet-mark=p2p

 2   chain=forward packet-mark=!p2p_conn action=mark-packet new-packet-mark=other
[admin@NAT] >
[admin@NAT] > /queue tree add parent=Public packet-mark=p2p limit-at=1000000
max-limit=100000000 priority=8
[admin@NAT] > /queue tree add parent=Local packet-mark=p2p limit-at=1000000
max-limit=100000000 priority=8
[admin@NAT] > /queue tree add parent=Public packet-mark=other  limit-at=1000000
max-limit=100000000 priority=1
[admin@NAT] > /queue tree add parent=Local packet-mark=other  limit-at=1000000
max-limit=100000000 priority=1
```

Mangle 限制 2 级代理

通过 mangle 限制 2 级代理，但对端口的 http 代理无效，我们可以指定 in-interface 或者指定目标数据到内往的 IP 地址，即 dst-address

```
[admin@MikroTik] /ip firewall mangle> add chain=forward out-interface=lan action
=change-ttl new-ttl=set:1
[admin@MikroTik] /ip firewall mangle>print chain=forward
Flags: X - disabled, I - invalid, D - dynamic
8   chain=forward action=change-ttl new-ttl=set:1 out-interface=lan
```

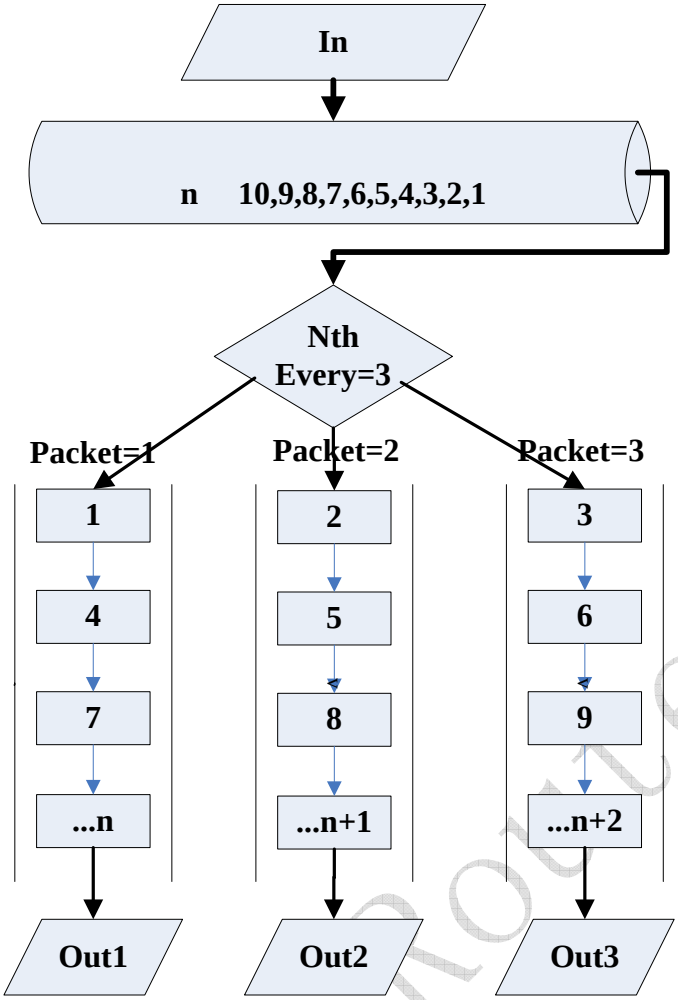
第十四章 RouterOS Nth

Nth 是 RouterOS 中对路由负载均衡一个重要的工具，不仅可以实现基于 IP 的负载均衡，还能实现对端口负载均衡和 nat 的有序指定访问

14.1 Nth 原理介绍

在 v3.0 中 Nth 工具做了一点修改，仅只有两个参数“every”和“packet”。每个规则都有自己的计数器。当规则收到数据包，当前规则的计数器会增加 1，如果计数器匹配值“every”与数据包匹配，计数器将重新设置为 0。 使用 Nth 我们可以将一串连接通过计数器分离，比如可以将连接分配为多个组，重新排列连接序列。

nth –匹配特定的第 N 次收到的数据包的规则。一个计数器最多可以计数 16 个数据包
Every – 匹配每 every 数据包，同时指定 **Counter**（计数器值）
Packet – 匹配给定的数据数，例如，Nth=3,1，匹配 3 个数据包的第 1 个

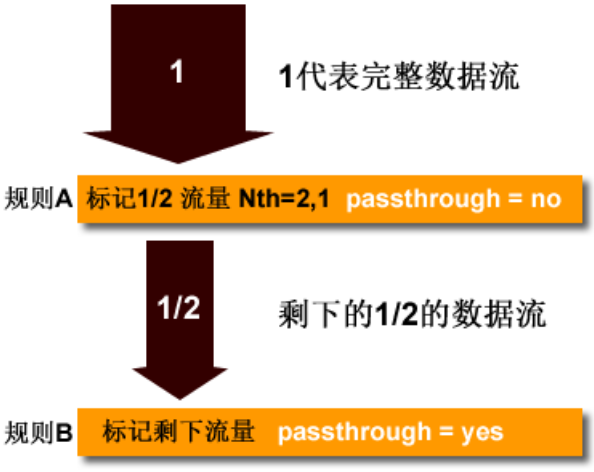


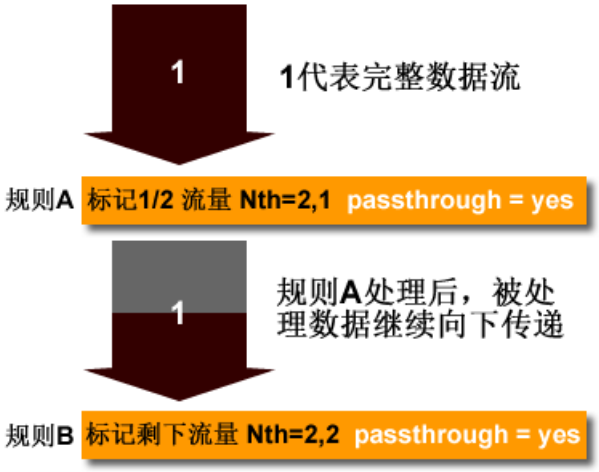
数据

上图，可以看到数据流从 1-n 的数据，被 Nth 分为 3 个计数器，并根据 Packet 重新排列数据流的队列。Nth 我们可以应用的范围，包括多线路的负载均衡、内网多台 ftp 访问、以及其他的应用。

14.2 Passthrough 对 Nth 的控制

实现相同的 Nth 结果时，改变 Passthrough 参数（Passthrough 为是否将该规则数据继续向下传递，**no** 为停止向下传递，**yes** 则相反，具体参考 Mangle 章节）会得到不同的规则配置，首先要知道 Mangle 标记捕获数据是先进先出算法，即从上往下执行，我们在配置 Mangle 的 Nth 规则，需要注意前后顺序。如我们把数据流标记为两个组，即一条为 1/2，另一条也为 1/2，把一个数据流看成“1”，而我们把可以通过两种方法配置：





当我们需要将数据流标记为 3 组时，即每条规则为 1/3。配置方法同样有两种，如下图

如同从上面的图上看到，使用和不使用 Passthrough 的区别，在于流量是否继续向下传递。

例如，有双线接入，并采用 Nth 的双线负载均衡。首先我们需要在 mangle 里标记连接，如果配置 Passthrough=no 参数，Nth 参数配置仅需要一条规则，即标记置 50%流量，首先我们需要标记连接：

```
/ip firewall mangle
add chain=prerouting new-connection-mark=AAA nth=2,1 action=mark-connection
passthrough=no;
```

抓取完前 50%的数据后，剩下的流量只需要做一个默认的标记剩下的数据即可。

```
add chain=prerouting new-connection-mark=BBB action=mark-connection
```

当变成 3 条线路时，第一条规则标记所有数据包并对比所有流量的 1/3，第二条规则标记剩下 2/3 数据包的 50%，第三条规则标记和对比所有剩下的数据包（所有数据包的 1/3）

```
/ip firewall mangle
add action=mark-connection chain=prerouting new-connection-mark=AAA nth=3,1
passthrough=no;
add action=mark-connection chain=prerouting new-connection-mark=BBB nth=2,1
passthrough=no;
add action=mark-connection chain=prerouting new-connection-mark=CCC ;
```

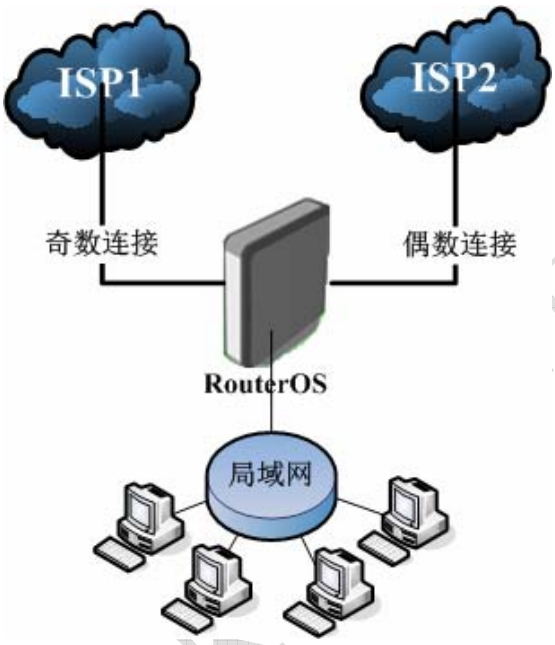
同样我们有的数据包并且每个规则对比每 3 个数据包。

```
/ip firewall mangle
add action=mark-connection chain=prerouting new-connection-mark=AAA nth=3,1
passthrough=yes;
```

```
add action=mark-connection chain=prerouting new-connection-mark=BBB nth=3,2
passthrough=yes;
add action=mark-connection chain=prerouting new-connection-mark=CCC nth=3,3
passthrough=yes;
```

14.3 Nth 在负载均衡的应用

下面我看一个实际的双线接入的 Nth 应用事例，假设我们有两条 ISP 的线路，我们通过 Nth 的方法实现负载均衡，让 2 条同样 ISP 线路达到合并带宽的作用。



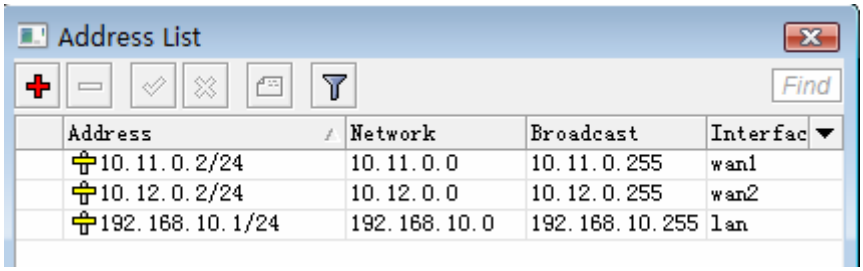
根据 Nth 的原理我们可以将来至内网的联接分为两组，即一组为奇数连接、一组为偶数连接，即奇数走一条线路，偶数走另外一条线路。因为我们定义的是连接状态为 new，即新建立的连接，对正常的访问没有任何影响，每个新建立所产生的后续数据都会按照原来的线路连接运行。

我们从所有的连接中，提取每次新建立的连接 connection=new，并对他们做 Nth 的标记，将这些连接中相关的奇数（odd）包和偶数（even）包分离开，并走两个不同的网关（ISP1 与 ISP2）出去。这样就能保持每次连接的持续性。

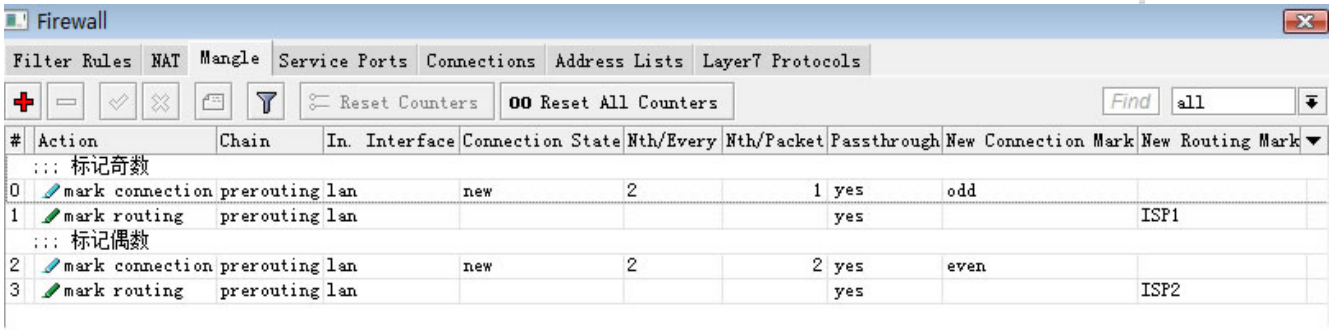
网络参数如下：

- wan1：ip 地址 10.11.0.2/24，网关 10.11.0.1
- wan2：ip 地址 10.12.0.2/24，网关 10.12.0.1
- lan：192.168.10.1/24

首先配置 IP



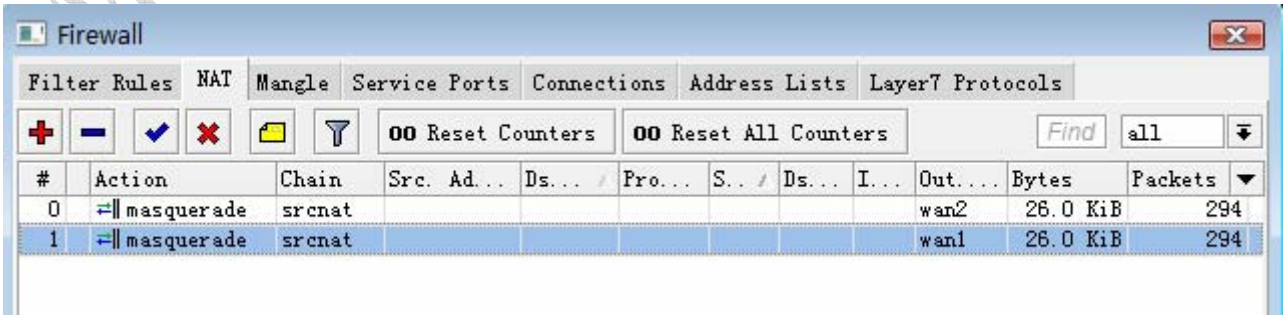
接下来在 ip firewall mangle 中标记奇数和偶数的 Nth，并配置路由标记，奇数 Nth 连接标记取名为 odd，偶数连接标记取名为 even，将奇数的路由标记取名为 ISP1，将偶数的路由标记取名为 ISP2，如下：



命令行配置如下：

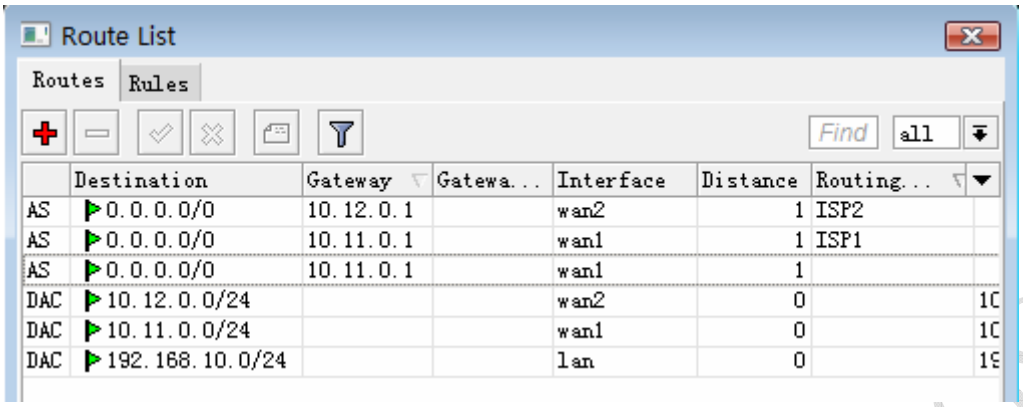
```
[admin@MikroTik] /ip firewall mangle> print
Flags: X - disabled, I - invalid, D - dynamic
 0 chain=prerouting action=mark-connection new-connection-mark=odd passthrough=yes
connection-state=new in-interface=lan nth=2,1

 1   chain=prerouting action=mark-routing new-routing-mark=ISP1 passthrough=yes
in-interface=lan connection-mark=odd

 2 chain=prerouting action=mark-connection new-connection-mark=even passthrough=yes
connection-state=new in-interface=lan nth=2,2

 3   chain=prerouting action=mark-routing new-routing-mark=ISP2 passthrough=yes
in-interface=lan  connection-mark=even
```

NAT 配置



路由配置

进入 ip route 中配置路由规则，配置 10.12.0.1 对应 ISP2 的路由标记，10.11.0.1 对应 ISP1 的路由标记，我们用 10.11.0.1 作为路由器本身的默认网关。



命令行配置如下

```
/ ip route
add gateway=10.11.0.1 routing-mark=ISP1
add gateway=10.12.0.1 routing-mark=ISP2
```

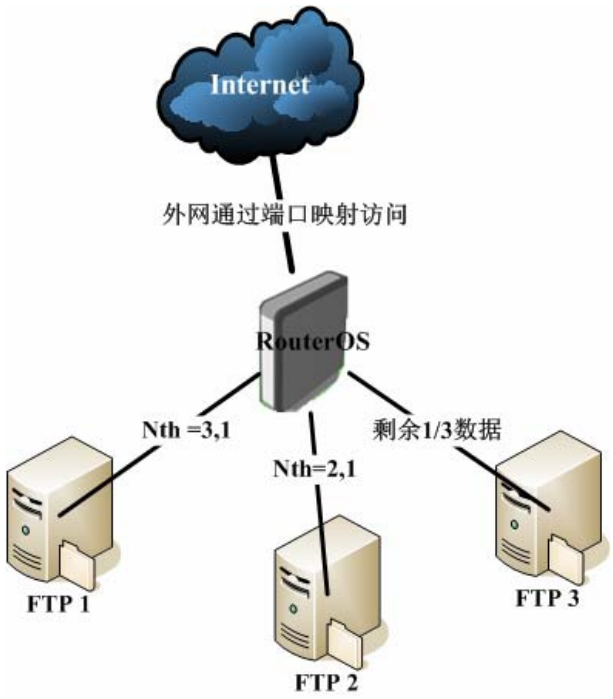
这样双线的 Nth 负载均衡就配置完成，建议这样的负载均衡使用在相同 ISP 的线路上，并且带宽接近。

注：在 Nth 时我们需要将 TCP 443 和 8443 端口指定到固定的一条线路，避免一些要求固定 IP 验证的网站，如网上银行等。

Nth 在最多支持 16 个计数器，如果我们允许接入 12 条相同带宽的线路，并采用 **passthrough=yes**，即 Nth 规则应是[every,packet]=[12,1]，[12,2]，[12,3]，[12,4]，[12,5]，[12,6]，[12,7]，[12,8]，[12,9]，[12,10]，[12,11]，[12,12]

14.4 Nth 在端口映射的应用

通过 Nth 的原理我可以实现一些特定的应用，比如应用于 FTP 服务的端口映射，当我们有大量信息需要向互联网共享时，可能我们一台 FTP 服务器无法承担所有的数据流量，我们可以通过建立多台服务器来分担流量，在不必修改 FTP 端口的情况下，通过 Nth 均衡分流数据到 3 台 ftp 服务器上，如下图内网的 3 个 FTP 服务器：



我们通过建立 3 条 nat 规则，区别 3 个不同的服务器连接，在 nat 中没有同时做 Passthrough 的选项，而且在 nat 规则中采用的是先进先出算法，所以我们只能采用先标记 1/3，在标记 1/2，最后标记剩下的数据的方法处理 3 条线路的均衡操作。

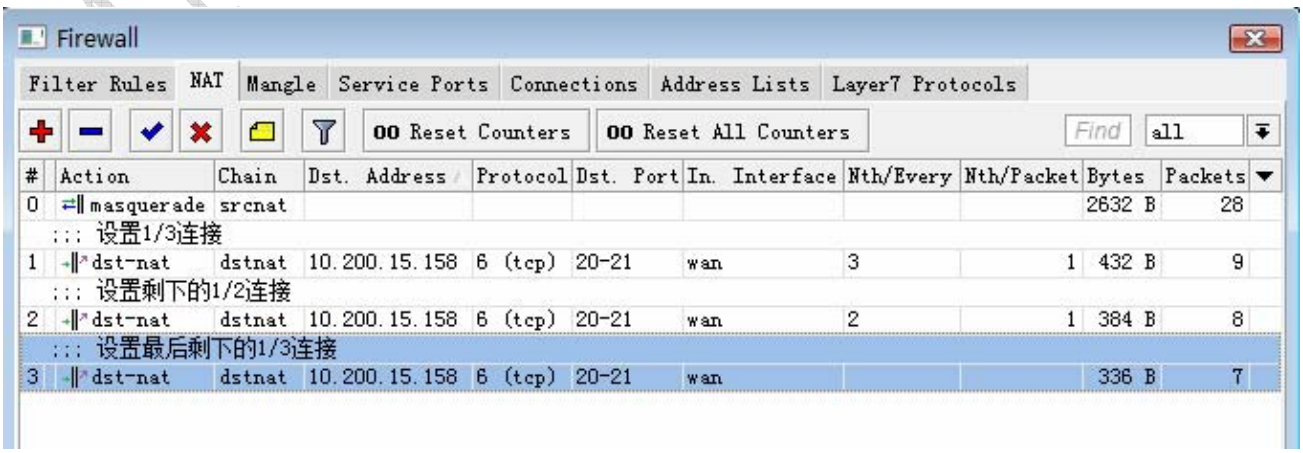
我们的网络环境如下

- Wan：ip 地址为 10.200.15.158/24 网关为 10.200.15.1
- Lan：ip 地址为 192.168.10.1/24
- 内网的 3 个 FTP 服务器的 IP 地址分别是 192.168.10.2，192.168.10.3，192.168.10.4

在配置完 IP 地址后，我们进入 ip firewall nat 配置 nat 规则， 首先我们需要配置基本的 nat 伪装规则，将内网的私有 IP 地址转换为公网 IP。

```
/ip firewall nat
add action=masquerade chain=srcnat disabled=no out-interface=wan
```

接着，设置端口映射，FTP 使用的是 TCP，20-21 端口，我们配置 3 条 nat 的 Nth 端口映射的规则，分别指向 192.168.10.2、192.168.10.3 和 192.168.10.4 三个服务器的 IP 地址：



通过命令行配置如下：标记前 1/3 的端口映射

```
add action=dst-nat chain=dstnat dst-address=10.200.15.158 dst-port=20-21
in-interface=wan nth=3,1 protocol=tcp to-addresses=192.168.10.2 to-ports=20-21
```

标记剩下 1/2 的端口映射

```
add action=dst-nat chain=dstnat dst-address=10.200.15.158 dst-port=20-21
in-interface=wan nth=2,1 protocol=tcp to-addresses=192.168.10.3 to-ports=20-21
```

标记最后 1/3 的端口映射

```
add action=dst-nat chain=dstnat dst-address=10.200.15.158 dst-port=20-21
in-interface=wan protocol=tcp to-addresses=192.168.10.4 to-ports=20-21
```

这样通过 Nth 分流的端口映射配置完成，这样的 Nth 操作仅适合于一次性提交和访问的数据连接。如果是带登陆验证的访问，不建议使用这种方式，会出现连接后在不同服务器上的重复认证。

第十五章 Bridge 网桥

支持以太网 MAC 等级桥接，EoIP（Ethernet over IP），Prism, Atheros 以及广播局域网。所有的 802.11a, 802.11b, and 802.11g 客户无线接口（**ad-hoc**, **infrastructure** 或 **station** 模式）都不支持这个因为 802.11 的限制。然而，在 Prism 和基于 Atheros 连接之间使用 WDS 特性（对基于卡片的 Atheros 和 Prism 芯片组）或 EoIP 进行桥接还是可能的。

为防止网络中的环路，你可以使用生成树协议（STP/RSTP）。这个协议也可以作为备分连接的配置。主要特征：

- 生成树协议(STP)
- 快速生成树协议（RSTP）
- 多重桥接口
- MAC 地址可以被实时监控
- 为路由器访问的 IP 地址分配
- 桥接口可以被过滤及网络地址翻译
- 支持基于桥数据包过滤器的桥路由

快速配置指南

把接口 **ether1** 和 **ether2** 放在一个桥里：

1. 添加一个桥接口，命名为 **MyBridge**：

```
/interface bridge add name="MyBridge" disabled=no
```

2. 把 **ether1 和 ether2** 添加到 **MyBridge** 接口：

```
/interface bridge port add interface=ether1 bridge=MyBridge
/interface bridge port add interface=ether2 bridge=MyBridge
```

规格

功能包需要: ***system***
认证需要: *Level3*
子目录需要: ***/interface bridge***
标准和技术: IEEE801.1D

类似以太网的网络（Ethernet, Ethernet over IP, IEEE802.11 in ap-bridge 或 bridge 模式, WDS, VLAN）可以通过使用 MAC 桥连接在一起。桥特性允许这些不同局域网的主机互连（使用 EoIP，如果任何种类的 IP 网络互连存在其中则地理分布式网络也可以被桥接起来）好像他们是连接在一个局域网中。由于桥是透明的，他们不会在追踪路由表中出现，并且没有实用程序可以使工作在一个局域网中主机和工作在另一个局域网的主机有区别如果这些局域网桥接起来了（由于局域网互连方式的不同，不同主机间的延迟和数据率会有不同）。

网络环路可能以复杂的拓扑形式出现（有意或无意的）。如果没有特殊的处理，环路将组织网络的正常工作，因为他们可能导致雪崩一样的数据包倍增。每一个桥都运行一个计算如何组织环路的算法。STP 允许桥之间进行通信，于是他们可以协商无环路的拓扑。所有其他可能形成环路的连接被当成备用，所以如果主连接失败其他的连接就可以取代他的位置。这个算法定期地互相交换配置信息（BPDU—Bridge Protocol Data Unit：桥协议数据单元），因此所有的桥都可以用网络拓扑中最新变化的信息进行更新。STP 选择负责网络配置的根桥，像关闭和打开其他桥端口的桥。根桥是拥有最低 ID 的桥。

15.1 网络桥配置

操作路径: ***/interface bridge***

为了把许多网络连接到一个桥上，必须建立一个桥接口（一会，所有需要的接口都应该像他的端口一样配置）。一个 MAC 地址将会被分配给岁有的桥接口（最小的 MAC 地址将会被自动选择）。

属性描述

ageing-time (*时间*; 默认: **5m**) - 一个主机信息可以被保存在桥数据库的时间
arp (disabled | enabled | proxy-arp | reply-only; 默认: **enabled**) - 地址解析协议设置
forward-delay (*时间*; 默认: **15s**) - 在桥接口初始化阶段（例如：在路由器启动或起用接口之后）桥正常工作之前监听/学习状态所用的时间
garbage-collection-interval (*时间*; 默认: **4s**) - 丢弃桥数据库中老的（过期的）主机词条的频率。无用存储单元收集过程消除比 **ageing-time** 属性定义的更老的词条。
hello-time (*时间*; 默认: **2s**) - 给其他桥发送 hello 包的频率
mac-address (*只读: MAC 地址*) – 接口的 MAC 地址
max-message-age (*时间*; 默认: **20s**) - 保留从其他桥接受 hello 信息的时间长短
mtu (整型; 默认: **1500**) - 最大传输单元
name (名称; 默认: **bridgeN**) - 桥接口的描述性名称
priority (整型: 0..65535; 默认: **32768**) - 桥接口优先级。STP 使用优先级参数决定如果最后两个端口形成了环路应保留哪个
stp (no | yes; 默认： **no**) - 是否启用生成树协议。桥环路仅在这个属性启用是才会被阻止。

添加并启用一个转发所有协议的桥接口：

```
[admin@MikroTik] interface bridge> add; print
Flags: X - disabled, R - running
 0  R name="bridge1" mtu=1500 arp=enabled mac-address=61:64:64:72:65:73 stp=no
     priority=32768 ageing-time=5m forward-delay=15s
     garbage-collection-interval=4s hello-time=2s max-message-age=20s
[admin@MikroTik] interface bridge> enable 0
```

端口设置

操作路径: ***/interface bridge port***

Port 用于定义桥接的从属网络接口，即那些网络接口归属于指定的桥。

属性描述

bridge (*名称*; 默认: **none**) – 那些接口被定义为 bridge 接口
none – *接口没有被定义到任何桥中*
interface (*只读: 名称*) - 接口名，包含在一个桥内
path-cost (*整型*: 0..65535; 默认: **10**) - STP 使用的用以决定最佳路径代价
priority (*整型*: 0..255; 默认: **128**) - 同一网络中相比较于其他接口的接口优先级

注: 从 V2.9.9 版本起，列表中的端口应被添加（add）而非设置(set)，请看下面的例子：

把 **ether1 和 ether2** 分配到已创建的桥 **bridge1** 中（V2.9.9 以前）：

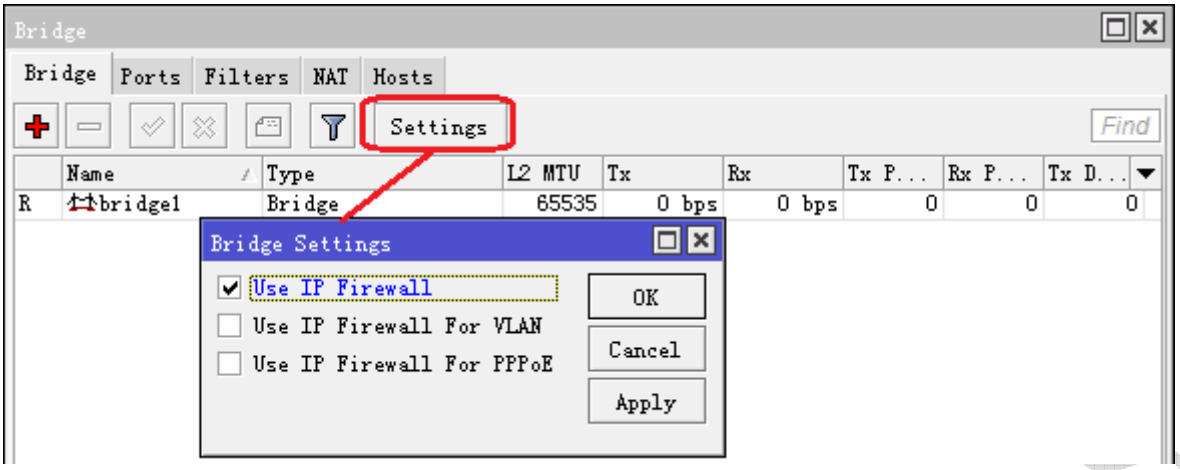
```
[admin@MikroTik] interface bridge port> set ether1,ether2 bridge=bridge1
[admin@MikroTik] interface bridge port> print
 # INTERFACE   BRIDGE PRIORITY PATH-COST    HORIZON
 0 ether1     bri... 0x80     10           none
 1 ether2     bri... 0x80     10           none
 2 wlan1      none     128     10             none
[admin@MikroTik] interface bridge port>
```

把 **ether1 和 ether2** 分到已创建的桥 **bridge1** 中（V2.9.9 后）：

```
[admin@MikroTik] interface bridge port> add interfae=ether1 bridge=bridge1
[admin@MikroTik] interface bridge port> add interfae=ether2 bridge=bridge1
[admin@MikroTik] interface bridge port> print
 # INTERFACE   BRIDGE PRIORITY PATH-COST     HORIZON
 0 ether1     bri... 0x80    10             none
 1 ether2     bri... 0x80     10             none
[admin@MikroTik] interface bridge port>
```

Bridge setting 参数

这个参数可以选择是否启用 ip firewall 的三层过滤规则，这个功能有别于 2.9 的桥接功能，如果关闭三层的 ip firewall 过滤，可以大大提高 RouterOS 的网桥转发率，特别在 WLAN 桥接和一些纯二层的网桥应用中非常有用。



当你需要开启三层的网桥过滤时，包括 ip firewall filter、mangle、nat 和 queue 流控等，就需要打开以下设置

```
[admin@MikroTik] /interface bridge settings>set use-ip-firewall=yes
[admin@MikroTik] /interface bridge settings>print
          use-ip-firewall: yes
   use-ip-firewall-for-vlan: no
  use-ip-firewall-for-pppoe: no
[admin@MikroTik] /interface bridge settings>
```

当然网桥由于打开了三层过滤，转发率会受到一定的影响，但你可以实现对三层数据的过滤和流控。通过启用三层过滤我们可以实现许多透明桥的流量整形，如 IP 地址流控、P2P 和基于 80 端口的 HTTP 流控等；也可以实现基于 IP 和协议端口的过滤。

15.2 桥接口查看

命令名：***/interface bridge monitor***

用于监听一个桥的当前状态。

属性描述

bridge-id (*文本*) – 桥 ID，以如下形式 bridge-priority，bridge-MAC-address
designated-root (文本) - 根桥的 ID
path-cost (*整型*) - 到根桥所需总代价
root-port (*名称*) -根桥连接的端口

监听一个桥：

```
[admin@MikroTik] interface bridge> monitor bridge1
       state: enabled
   current-mac-address: 00:00:00:00:00:00
         root-bridge: yes
      root-bridge-id: 0x8000.00:00:00:00:00:00
      root-path-cost: 0
          root-port: none
         port-count: 2
```

```
  designated-port-count: 0

[admin@MikroTik] interface bridge>
```

15.3 桥端口监测

命令名: ***/interface bridge port monitor***

属性描述

designated-port (*文本*) - 指定根桥端口
designated-root (*文本*) - 最靠近根桥的桥 ID
port-id (*整型*) - 端口 ID，代表端口优先级和端口号且是唯一的
status (disabled | blocking | listening | learning | forwarding) - 桥端口的状态：
  disabled - 端口被禁用。没有帧被转发，没有桥协议数据单元（BPDUs）被收到
  blocking - 端口不转发任何帧但监听 BPDU
  listening - the port does not forward any frames, but listens to them 端口不转发任何帧但监听
  learning - 端口不转发任何帧但学习 MAC 地址
  forwarding - 端口转发帧并学习 MAC 地址

监听一个桥端口：

```
[admin@MikroTik] interface bridge port> mo 0
             state: enabled
   current-mac-address: 00:00:00:00:00:00
          root-bridge: yes
       root-bridge-id: 0x8000.00:00:00:00:00:00
       root-path-cost: 0
           root-port: none
          port-count: 2
 designated-port-count: 0
-- [Q quit|D dump|C-z pause]
```

15.4 桥主机列表

命令名: ***/interface bridge host***

属性描述

age (*只读:时间*) - 从主机获得最后一个包开始的时间
bridge (只读: *名称*) - 属于词条（entry）的桥
local (只读: *标志*) - 主机词条是否是桥本身的
mac-address (只读: *MAC 地址*) - 主机 MAC 地址
on-interface (只读: *名称*) - 主机所连接的桥接的接口

获得活动的主机列表：

```
[admin@MikroTik] interface bridge host> print
```

```
Flags: L - local, E - external-fdb
  BRIDGE            MAC-ADDRESS       ON-INTERFACE      AGE
  bridge1           00:00:B4:5B:A6:58 ether1            4m48s
  bridge1           00:30:4F:18:58:17 ether1            4m50s
L bridge1           00:50:08:00:00:F5 ether1            0s
L bridge1           00:50:08:00:00:F6 ether2            0s
  bridge1           00:60:52:0B:B4:81 ether1            4m50s
  bridge1           00:C0:DF:07:5E:E6 ether1            4m46s
  bridge1           00:E0:C5:6E:23:25 prism1            4m48s
  bridge1           00:E0:F7:7F:0A:B8 ether1            1s
[admin@MikroTik] interface bridge host>
```

15.5 桥防火墙

操作路径: ***/interface bridge filter***, ***/interface bridge nat***, ***/interface bridge broute***

桥防火墙执行包过滤因此提供了用于管理数据流进，流出和流经桥的安全功能。

注：在桥接接口之间的数据包就像其他 IP 流一样，也要经过类属的**/ip firewall** 规则（但桥过滤器总是在 IP 过滤器/NAT 之前应用，除了在 IP 防火墙输出之后执行的 **output**）。这些规则可以同真实的物理接收/发送接口一起使用，也可以和简单对桥接在一起的接口划分的桥接口同时使用。

有三种桥过滤器列表：

- **filter** - 有三个预先设定的桥防火墙链表：
  - **input** -其目的地是桥（进入桥设备的数据包，无论什么情况下以本地桥 MAC 地址为目标的数据）。
  - **output** - 来自于桥（由桥设备本身处理发出的数据）
  - **forward** – 通过桥转发（即有桥设备转发到另外网络的数据）。
- **nat** - 桥网络地址翻译提供了改变遍历桥的数据包的源/目的 MAC 地址的方法。它有连条内置的链：
  - **scnat** - 用于在一个不同的 MAC 地址后“隐藏”一个主机或者一个网络。这个链适用于通过一个桥接口离开路由器的数据包
  - **dstnat** -用于把一些包重定向到另一个目的地址
- **broute** - 使一个桥变为一个桥路由器 — 一种在一些包上起路由作用而在其他包起桥作用的路由器。它有一个预定义链：**brouting**，当一个包进入一个受控接口后它便进行遍历（在“Bridging Decision”之前）。

注：桥的目标网络地址翻译在桥接判定之前执行。当需要涉及到三层过滤时或者流量控制，需要将桥的 **use-ip-firewall** 启用，否则三层过滤和流量控制将无法工作。

你可以在桥防火墙（filter, broute and NAT）中设置数据包标记，就像用 mangle 在 IP 防火墙中设置数据包标记一样。所以用桥防火墙设置的包标记可以在 IP 防火墙中使用，反之亦然。普通桥防火墙属性在这部分描述。一些在 nat,，broute 和 filter rules 之间有区别的参数将在后面的部分描述。

属性描述

802.3-sap (整型) - DSAP（目的文件服务访问点） 和 SSAP（源端业务接入点）是两个 1 字节域，它们识别使用链路层服务的网络协议实体。这些字节总是相等的。两个十六进制数字可以在这里指定以匹配 SAP 字节。
802.3-type (整型) - 以太网协议类型，放置在 IEEE 802.2 帧标题后面。仅当 802.3-sap 为 0xAA（SNAP——子网连接点标题）时才生效。例如：AppleTalk 可以由跟随在 0x8098 SNAP 类型码后面的 0xAA SAP 码说明。

arp-dst-address (*IP 地址*; 默认: **0.0.0.0/0**) - ARP 目的地址
arp-dst-mac-address (*MAC 地址*; 默认: **00:00:00:00:00:00**) - ARP 目的 MAC 地址
arp-hardware-type (*整型*; 默认: **1**) - ARP 硬件类型
arp-opcode (arp-nak | drarp-error | drarp-reply | drarp-request | inarp-request | reply | reply-reverse | request | request-reverse) - ARP opcode (数据包类型)
arp-nak – 消极 ARP 应答 (很少使用，主要在 ATM 网络中使用)
drarp-error – 动态 RARP 错误代码, saying that an IP address for the given MAC address can not be allocated 表明一个给定 MAC 地址的 IP 地址不能分配
drarp-reply – 动态 RARP 应答，带有一个主机临时地址分配
drarp-request - 动态 RARP 请求一个对给定 MAC 地址的临时 IP 地址
reply - 带有一个 MAC 地址的标准 ARP 应答
reply-reverse - 带有一个以分配 IP 地址的反向 ARP（RARP）应答
request - 向一个已知 IP 地址询问未知 MAC 地址的标准 ARP 请求
request-reverse - reverse ARP (RARP) request to a known MAC address to find out unknown IP 向已知 MAC 地址询问未知 IP 地址的凡响 ARP（RARP）请求(intended to be used by hosts to find out their own IP address 主机有意用来查明其本身 IP 地址，类似于 DHCP 服务)
arp-src-address (*IP 地址*; 默认: **0.0.0.0/0**) – ARP 源 IP 地址
arp-src-mac-address (*MAC 地址*; 默认: **00:00:00:00:00:00**) – ARP 源 MAC 地址
chain (文本) - 过滤器工作其中的桥防火墙链（内置或用户定义的）
dst-address (*IP 地址*; 默认: **0.0.0.0/0**) – 目的 IP 地址(仅当 MAC 协议设置为 IPv4 时)
dst-mac-address (*MAC 地址*; 默认: **00:00:00:00:00:00**) – 目的 MAC 地址
dst-port (整型: 0..65535) - 目标端口号或范围（仅对 TCP 或 UDP 协议）
in-bridge (名称) - 数据包进入的桥接口
in-interface (名称) - 数据包进入的物理接口（例如：桥端口）
ip-protocol (ipsec-ah | ipsec-esp | ddp | egp | ggp | gre | hmp | idpr-cmtp | icmp | igmp | ipencap | encap | ipip | iso-tp4 | ospf | pup | rspf | rdp | st | tcp | udp | vmtp | xns-idp | xtp) – IP 协议(仅当 MAC 协议设置为 IPv4)
ipsec-ah - IPsec AH 协议
ipsec-esp - IPsec ESP 协议
ddp - 数据报投递协议
egp - 外部网关协议
ggp – 网关-网关协议
gre - 通用路由压缩
hmp - 宿主监督协议
idpr-cmtp - idp 控制报文传输
icmp - 因特网控制报文协议
igmp - 因特网分组管理协议
ipencap - ip 压缩至 ip
encap - ip 压缩
ipip - ip 压缩
iso-tp4 - iso 传输协议类型 4
ospf - 开放式最短路径优先
pup - parc 通用包协议
rspf - 广播最短路径优先
rdp - 靠数据报协议
st - st 数据报模式
tcp - 传输控制协议
udp - 用户数据报协议
vmtp - 通用信息传输

xns-idp - xerox ns idp
xtp – xpress 传输协议
jump-target (名称) – 如果指定 **action=jump**，那么指定用户定义的防火墙链来处理数据包
limit (*整型*/时间{0,1},*整型*) - 以给定值限制包匹配率，有助于减少日志消息的总量
Count - 除非跟随在 **Time** 选项之后否则以包每秒（pps）衡量最大平均包率
Time - 指定包率测量的时间间隔
Burst - 要匹配的脉冲串中的包数量 8
log-prefix (文本) - 在日志信息之前定义用于打印的前缀
mac-protocol (整型 | 802.2 | arp | ip | ipv6 | ipx | rarp | vlan) - 以太网有效负载类型(MAC 等级协议)
mark-flow (*名称*) - marks existing flow
packet-type (broadcast | host | multicast | other-host) – MAC 帧类型:
broadcast – 广播 MAC 包
host -目的为桥本身的数据包
multicast – 多重 MAC 包
other-host - 定位到其他联合广播地址而非到桥本身的数据包
src-address (*IP 地址*; 默认: **0.0.0.0/0**) – 源 IP 地址(仅当 MAC 协议设置为 IPv4 时)
src-mac-address (*MAC 地址*; 默认: **00:00:00:00:00:00**) – 源 MAC 地址
src-port (整型: 0..65535) - 端口号或范围 (仅对 TCP 或 UDP 协议)
stp-flags (topology-change | topology-change-ack) - BPDU (网桥协议数据单元)标志。桥之间为阻止环路定期地互相交换名为 BPDU 的配置信息。
topology-change - 拓扑变化标志是当一个桥检测到端口状态改变时设置，它命令所有其他桥丢弃它们的主机列表并重新计算网络拓扑
topology-change-ack - 拓扑变化确认标志是作为通告数据包回应而设置的
stp-forward-delay (*time*: 0..65535) - forward delay timer 转发延迟计时器
stp-hello-time (*time*: 0..65535) - stp hello 数据包时间
stp-max-age (*time*: 0..65535) – 最大 STP 信息年龄
stp-msg-age (*time*: 0..65535) – STP 信息年龄
stp-port (*整型*: 0..65535) – stp 端口识别
stp-root-address (*MAC 地址*) – 根桥 MAC 地址
stp-root-cost (*整型*: 0..65535) – 根桥代价
stp-root-priority (*时间*: 0..65535) – 根桥优先级
stp-sender-address (*MAC 地址*) – stp 信息发射机 MAC 地址
stp-sender-priority (*整型*: 0..65535) – 发射机优先级
stp-type (config | tcn) - BPDU 类型
config – 配置 BPDU
tcn - 拓扑变化通告
vlan-encap (802.2 | arp | ip | ipv6 | ipx | rarp | vlan) -压缩在 VLAN 帧中的 MAC 协议类型
vlan-id (*整型*: 0..4095) – VLAN 识别域
vlan-priority (*整型*: 0..7) – 用户优先级域

*注：*仅当目的 MAC 地址为 01:80:C2:00:00:00/FF:FF:FF:FF:FF:FF (桥组地址)时, **stp** 匹配器才有效，同时 **stp** 应被启用。仅当 **mac-protocol** 为 **arp** 或 **rarp** 时 ARP 匹配器才有效。VLAN 匹配器仅对 **vlan** 以太网协议有效。IP 相关匹配器仅当 **mac-protocol** 被设置为 **ipv4** 时才有效

如果实际帧和 IEEE 802.2 和 IEEE 802.3 标准一致时，802.3 匹配器就会被询问（注意：它并不是在全世界网络使用的工业标准以太网帧格式）。这些匹配器对其他包会被忽视。

桥数据包过滤

操作路径: ***/interface bridge filter***

这部分描述的是桥数据包过滤器详细的过滤选项，在一般的防火墙描述中这部分通常都被省略掉了。

属性描述

action (accept | drop | jump | log | mark | passthrough | return; default: **accept**) - 如果数据包匹配了其中一个规则就采取动作：
accept - 接受包，无动作。例如：数据包通过而没有任何动作，并且没有其他规则会在相关列表/链中处理。
drop – 悄然地丢弃包(不发送 ICMP 拒绝信息)
jump - 跳转到由 jump-target 变量指定的链
log - 记录数据包
mark – 标记数据包以便后面使用
passthrough - 忽视这条规则并到下一个。除了对包计数外像一个被禁用的规则一样动作
return - 从跳转发生的地方回到前一个链
out-bridge (*名称*) - 流出桥的接口
out-interface (*名称*) - 数据包离开桥的接口

15.6 桥网络地址翻译 Bridge nat

操作路径: ***/interface bridge nat***

本部分描述了在一般防火墙描述中省略了的桥 nat 选项。

属性描述

action (accept | arp-reply | drop | dst-nat | jump | log | mark | passthrough | redirect | return | src-nat; 默认: **accept**) - 如果数据包匹配了其中一个规则就采取动作：
accept - 接受包，无动作。例如：数据包通过而没有任何动作，并且没有其他规则会在相关列表/链中处理。
arp-reply - 发送一个带有指定 MAC 地址的 ARP 应答(任何其他包都会被这条规则忽略，仅在 **dstnat** 链内有效)
drop – 悄然丢弃数据包 (不发送 ICMP 拒绝信息)
dst-nat - 改变一个包的目的 MAC 地址 (仅在 **dstnat** 链有效)
jump - 跳转到由 jump-target 变量指定的链
log - 记录数据包
mark – 标记数据包以便后面使用
passthrough - 忽视这条规则并到下一个。除了对包计数外像一个被禁用的规则一样动作
redirect - 把数据包重新定位到桥本身（仅在 **dstnat** 链中有效）
return - 从跳转发生的地方回到之前的链
src-nat - 改变包的源 MAC 地址 (仅在 **srcnat** 链中有效)
out-bridge (*名称*) - 流出桥接口
to-arp-reply-mac-address (*MAC 地址*) - 当选中 **action=arp-reply** 时，把源 MAC 地址加入以太网帧及 ARP 有效负载
to-dst-mac-address (*MAC 地址*) - 当选中 **action=dst-nat** 时，把目的 MAC 地址加入以太网帧
to-src-mac-address (*MAC 地址*) - 当选中 **action= src-nat** 时，把源 MAC 地址加入以太网帧

桥路路由

操作路径: ***/interface bridge broute***

这部分描述在一般防火墙描述省略了的桥路设施具体选项，桥路表应用于进入一个转发受控接口的每个包（例如：它不会工作在普通的接口，因为它们没有包含在桥里）。

属性描述

action (accept | drop | dst-nat | jump | log | mark | passthrough | redirect | return; 默认: **accept**) - action to undertake if the packet matches the rule, one of the:
如果数据包匹配了其中一个规则就采取动作：
accept - 由桥接代码决定对数据包做哪种处理
drop - 从桥接代码中提取数据包，使它看起来像来自一个非桥接的接口（不会在有其他桥判定或过滤被应用于这个包除非数据包被路由出到一个桥接的接口，这种情况下包将和其他路由包一样被正常处理）
dst-nat - 改变一个包的目的 MAC 地址（仅在 **dstnat** 链中有效）
jump - 跳转到由 jump-target 变量指定的链
log - 记录数据包
mark – 标记数据包以便后面使用
passthrough - 忽视这条规则并到下一个。除了对包计数外像一个被禁用的规则一样动作
redirect - 把数据包重新定位到桥本身（仅在 **dstnat** 链中有效）
return - 从跳转发生的地方回到之前的链
to-dst-mac-address (*MAC 地址*) - 当选中 **action=dst-nat** 时，把目的 MAC 地址加入以太网帧

故障分析

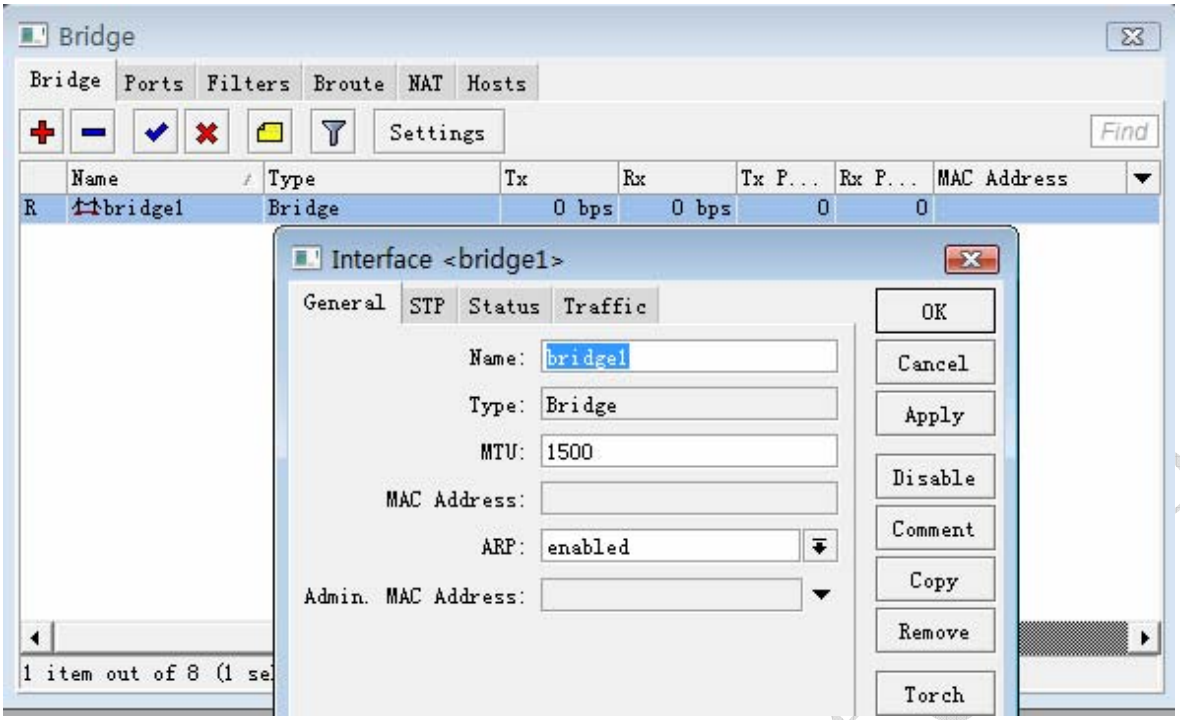
- **路由器显示我的规则不合法**
  - in-interface, in-bridge (或 in-bridge-port) 被指定，但并不存在这样的接口
  - 有一条 action=mark-packet 的动作，但没有 new-packet-mark
  - 有一条 action=mark-connection 的动作，但没有 new-connection-mark
  - 有一条 action=mark-routing 的动作，但没有 new-routing-mark

15.7 网桥应用事例

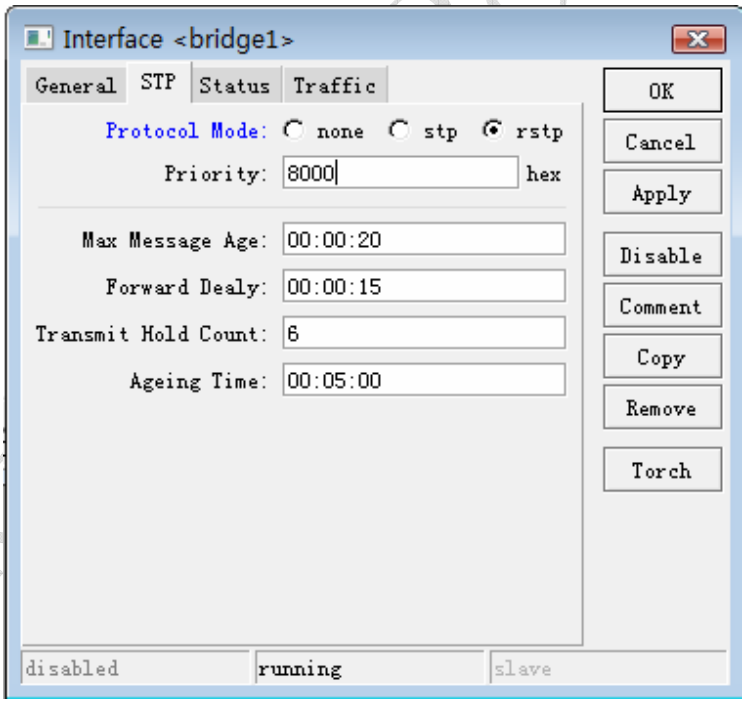
Bridge 实现二层端口隔离

RouterOS 具有 Bridge 的桥接功能，在配置多网口的情况下可以实现二层数据的转发，即可以实现交换机功能，加上 RouterOS 支持 birdge filter 的过滤，同样也支持对二层数据的管理，通过配置 Bridge 的防火墙规则实现多网口的端口隔离。

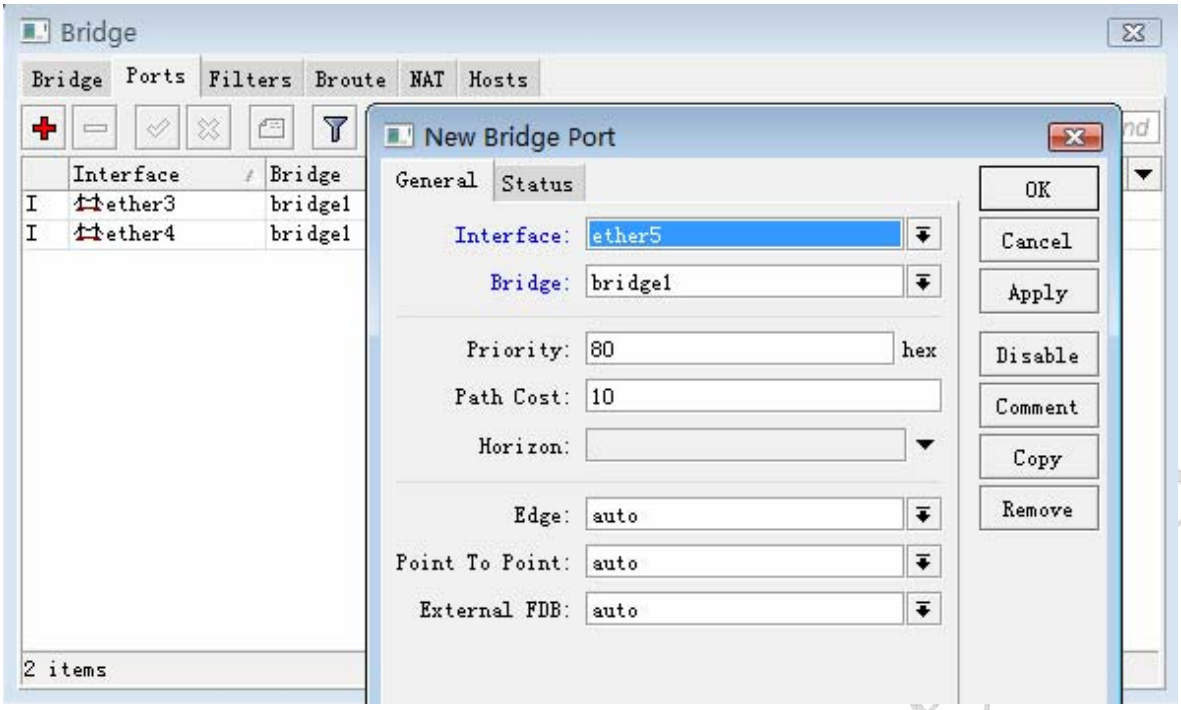
在这里我们通过 RB450 的操作为实例，配置二层端口隔离。首先我们在 Bridge 中添加一个网桥 bridge1：



在 bridge 中启用 rstp 快速生成树协议，防止二层的回环出现，同样也是支持二层的冗余功能，在这里我们选择 rstp：

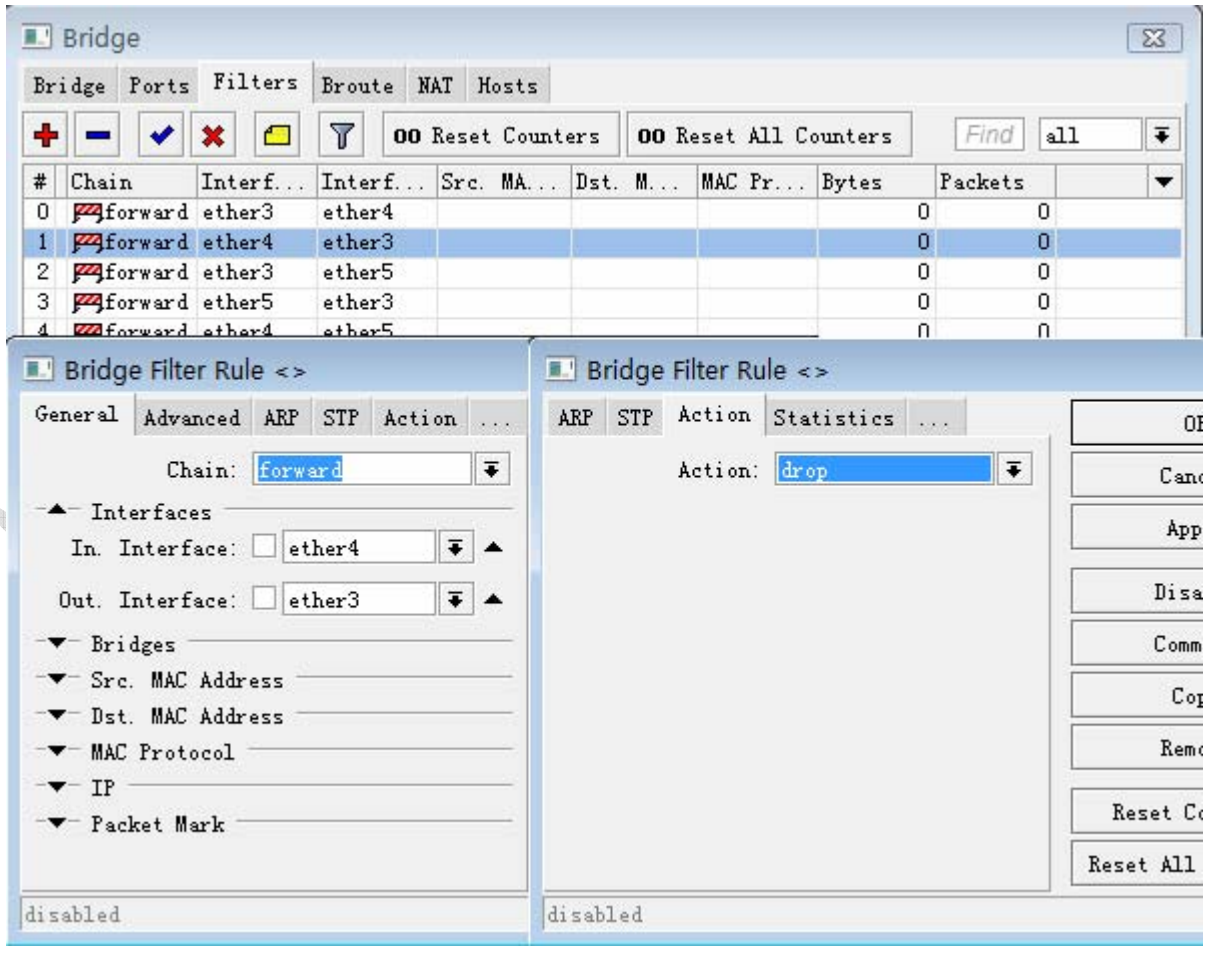


添加完桥接功能后，需要将对应的网卡添加入 bridge1 中，进入 Port 中设置，我们将 3 个网卡 ether3、ether4 和 ether5 一个一个添加到 bridge1 中：

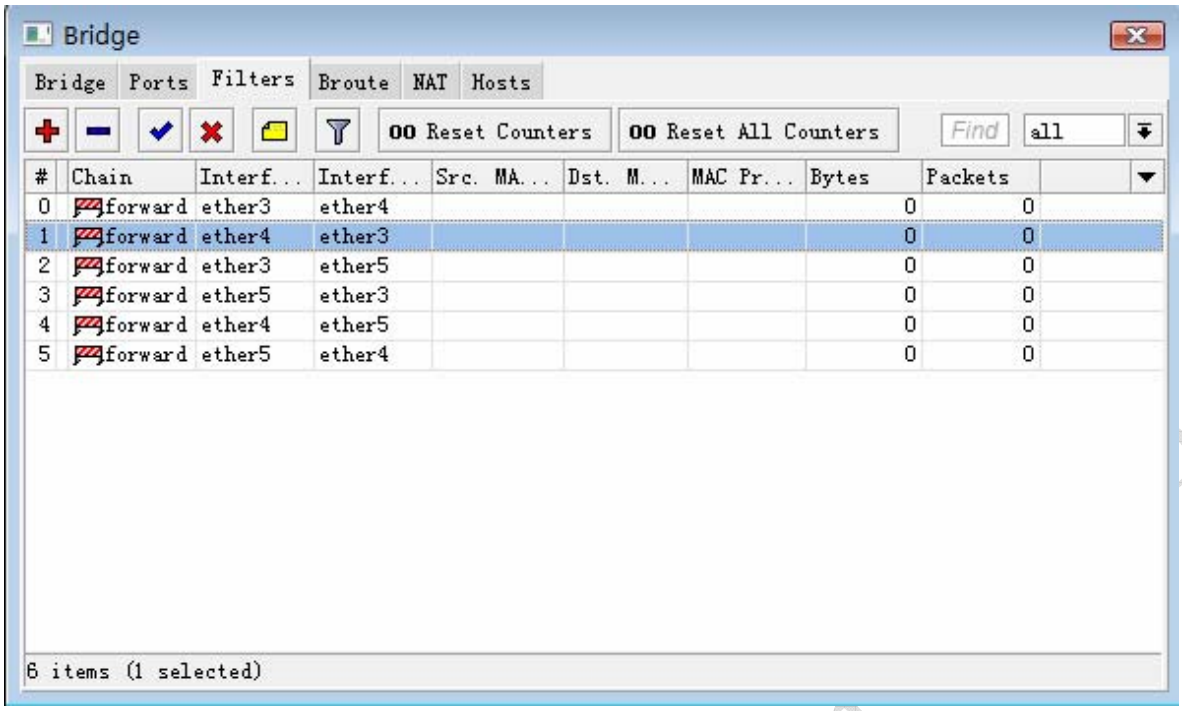


添加完每个端口后，现在 RB450 的 3 个以太网口，就完成了桥接的设置，这样 3 个口就实现了二层的交换功能。

这里我们禁止 ether3、ether4 和 ether5 进行通信， 我们进入 filter 中设置防火墙过滤规则，我们首先配置 ether3 与 ether4 的数据隔离我们在 interface 选项中设置 In-interface 和 Out-interface（In-interface 为数据进入的网口，Out-interface 为数据出去的网口，数据是双向传输的，两个接口需要做两条规则），然后选择 action 设置 action 参数为 drop，丢弃数据：



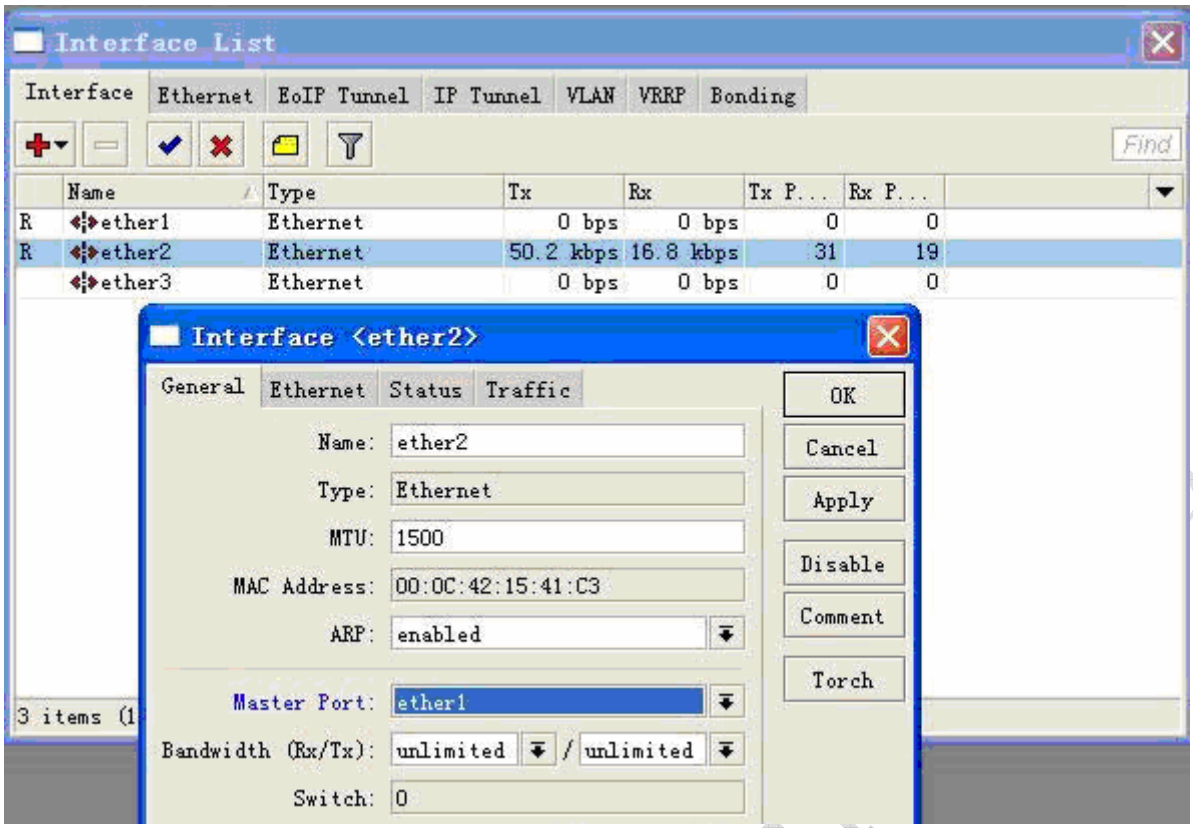
下面是设置好的状态：



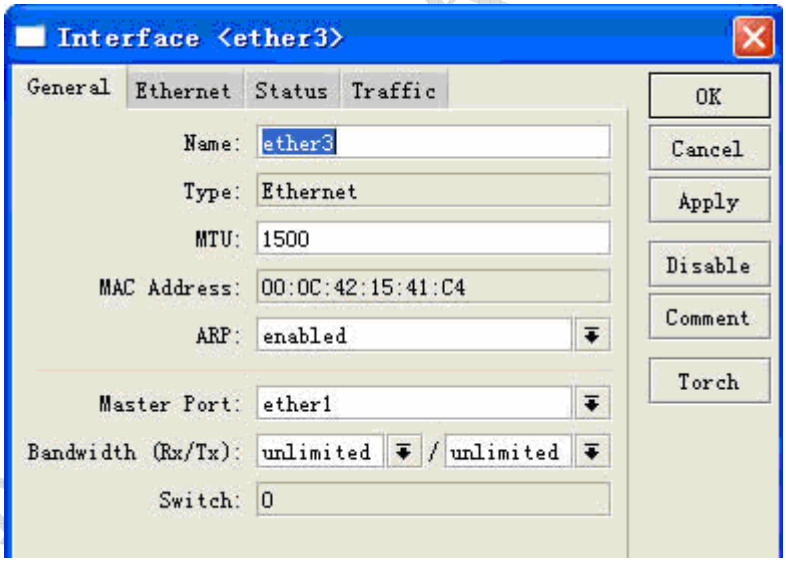
RouterBOARD 设置硬交换 Switch

随着 RouterOS 3.0 发布后，RouterBOARD 系列路由产品开始支持以太网口的硬件交换，如 RouterBOARD450 迷你路由器，5 个以太网口能设置为 5 个硬件交换口，即数据通过二层转发，不在经过 RouterOS 路由软件处理，完全和交换机转发相同。Switch 功能仅支持 RouterBOARD100 和 400 系列产品，需要 3.0 以上的软件版本支持。

下面我们用 RB433 为例，RB433 一共有 3 个以太网口，分别为 ether1、ether2 和 ether3，这里我们需要将三个网卡配置为交换口。设置硬件交换，需要将 1 个网口设置为主端口（Master port），其他口为从端口（Slave port），我们已 ether1 为 Master，其他网口为从端口。我们就只需要配置 ether2 和 ether3 的参数，配置 ether2 接口：



配置 ether3 接口



这样 ether1、ether2 和 ether3 设置为 switch 交换口，三个口可以多到数据的硬件转发，同时可以通过 interface 中的 Bandwidth 设置每个端口的带宽。

15.8 如何建立一个透明传输整形器

你想用在一个以太网中做一个 **MikroTik RouterOS** 透明传输整形器。你可以在两个网络中间加入。要达到这样 RouterOS™ 应该如下配置（这里假设为没有其他配置在整形器上，并且安装了两张以太网卡）：

1. 启用并命名以太网卡。连接到内部网络的网卡命令为 **int**，连接到上级路由器的网卡为 **ext** ：

```
/interface set ether1,ether2 disabled=no
```

```
/interface set ether1 name=int
/interface set ether2 name=ext
```

2. 让我们假设 10.0.0.1 的 IP 地址是网关。那我们添加 IP 地址为 **10.0.0.2/24** 到相应的网卡上(以后你将需要这个地址远程配置整形器)，设置好后你可以通过 ping 来检查你的网关。如果不能通，你可以换一下网线（例如：将插在 ext 网卡上的线换到 int 上，看是否网卡设置反了）**注：**如果一个都没有工作，可能在网关上设置了防火墙策略或是地址绑定，先暂时删除它们再试一次。

```
/ip address add interface=ext address=10.0.0.2/24
```

3. 创建一个桥接口，并将两个物理网卡 **int** 和 **ext** 做桥接：

```
/interface bridge add name=bridge
/interface bridge port add interface=ext bridge=bridge
/interface bridge port add interface=int bridge=bridge
```

注：现在前面设置的 IP 地址应被改变到 **bridge** 接口上：

```
/ip address set [/ip address find] interface=bridge
```

现在你可以简单的添加期望的队列。**注：**你可以在队列中使用真实的网卡名称。例如，限制所有下载为 256Kbit/s 和所有上传为 128Kbit/s，仅需要添加两条队列就可以了：

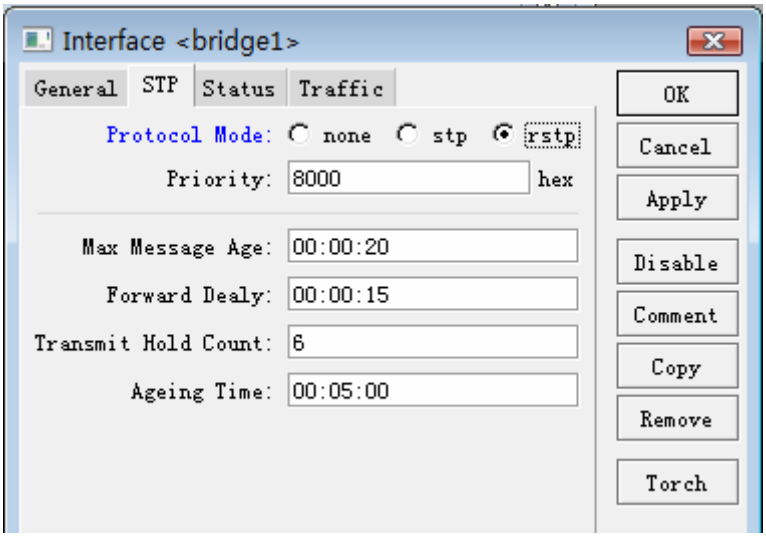
```
/queue simple add limit-at=131072 interface=ext
/queue simple add limit-at=262144 interface=int
```

15.9 通过 Bridge Filter 控制 MAC 地址

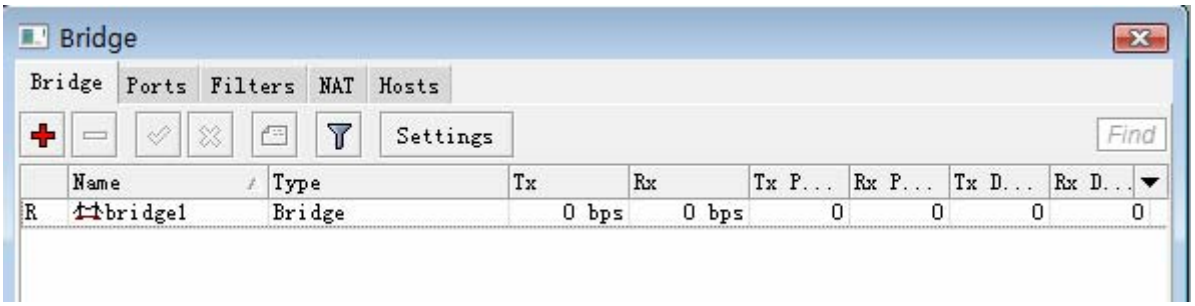
通过 bridge filter 控制 MAC 地址，如当我们把 RouterOS 设置为透明桥时，可以控制网络内的主机 MAC 通讯，这样我们可以从二层上控制客户端 PC。

我们通过 bridge 过滤 MAC 地址，必须启用 bridge，并指定相应网络接口到 bridge port 中，至少需要设定一个网络接口到 Port 中，设置 bridge 的操作如下

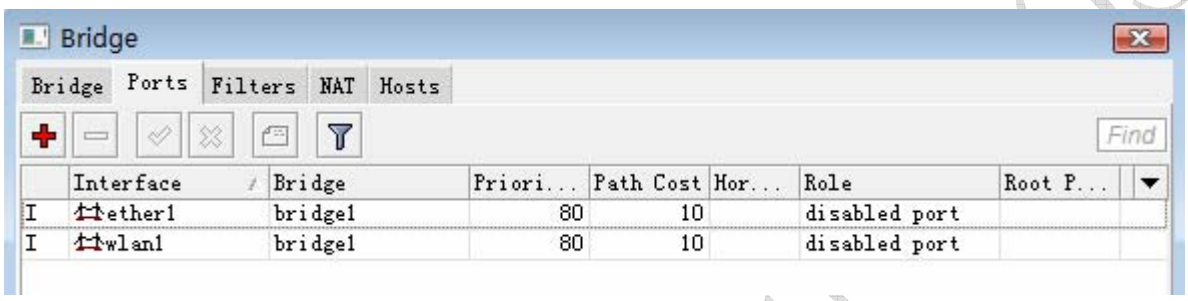
1、添加一个 bridge，默认的 bridge 的名称为 bridge1，并设置 RSTP（快速生成树协议）模式：



添加完 bridge 后，我们可以在 bridge 列表中查看到：



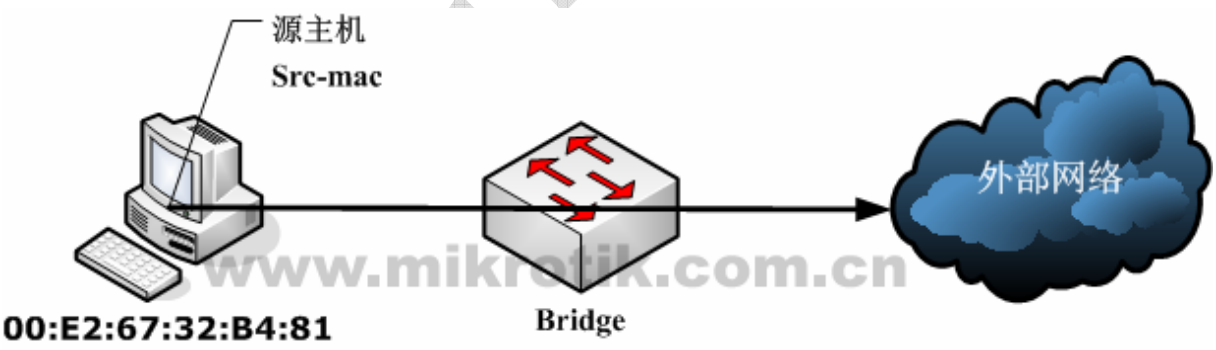
2、我们将 ether1 和 wlan1 网络接口添加到 bridge1 里，这样 2 个网络接口就实现了桥接功能



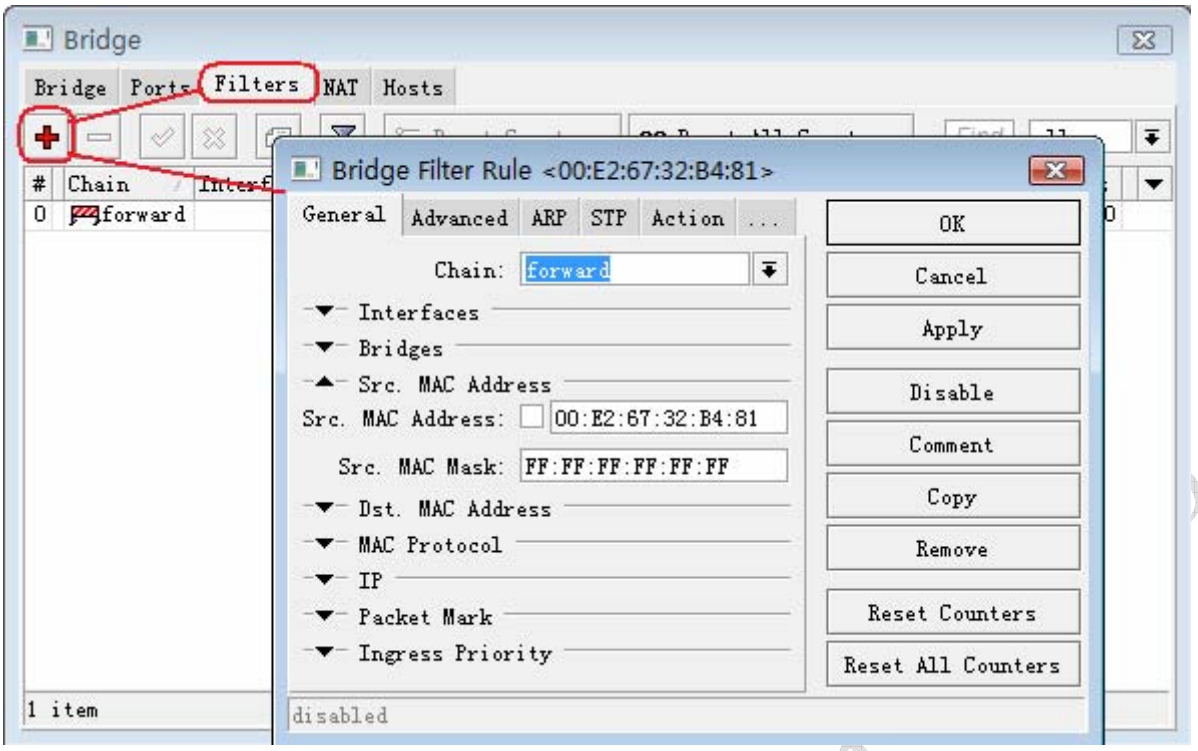
过滤源和目标 MAC 地址

1、源 MAC 地址过滤

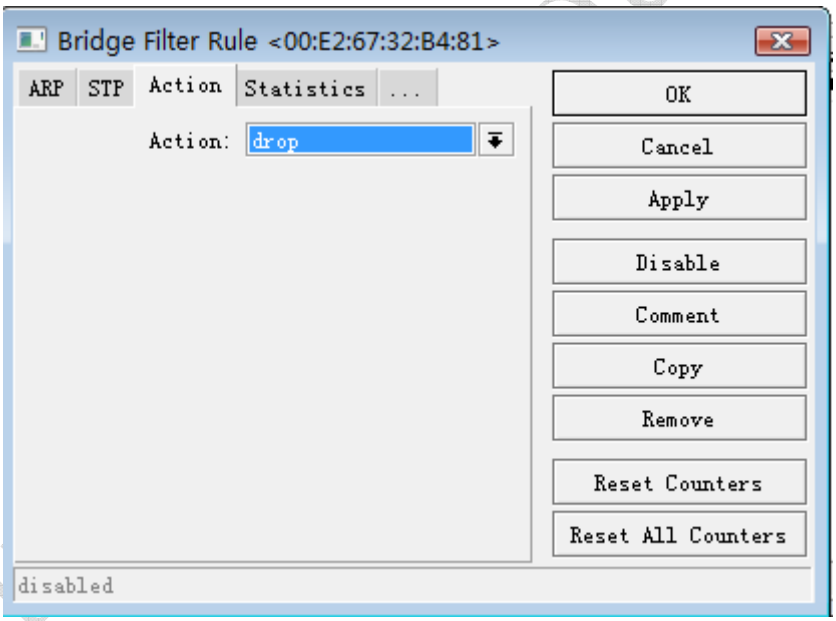
在设置完基本的 bridge 后，我们进入 bridge filter 中配置桥防火墙过滤，首先我们需要对指定的一台 PC 的 MAC：00:E2:67:32:B4:81 地址做过滤，不允许与 bridge 的外部网络连接，如下图：



这个 MAC 是发起源，选择 src-mac-address，由于这里拒绝访问 bridge 以外的网络，选择 chain=forward，，设定 action=drop。RouterOS Winbox 配置如下：



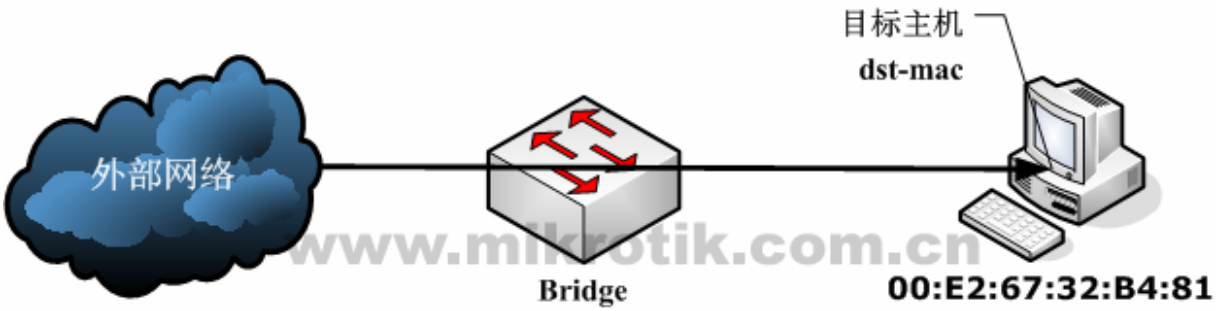
接下来选择 Action 为 drop，丢弃该 MAC 地址发出的数据：



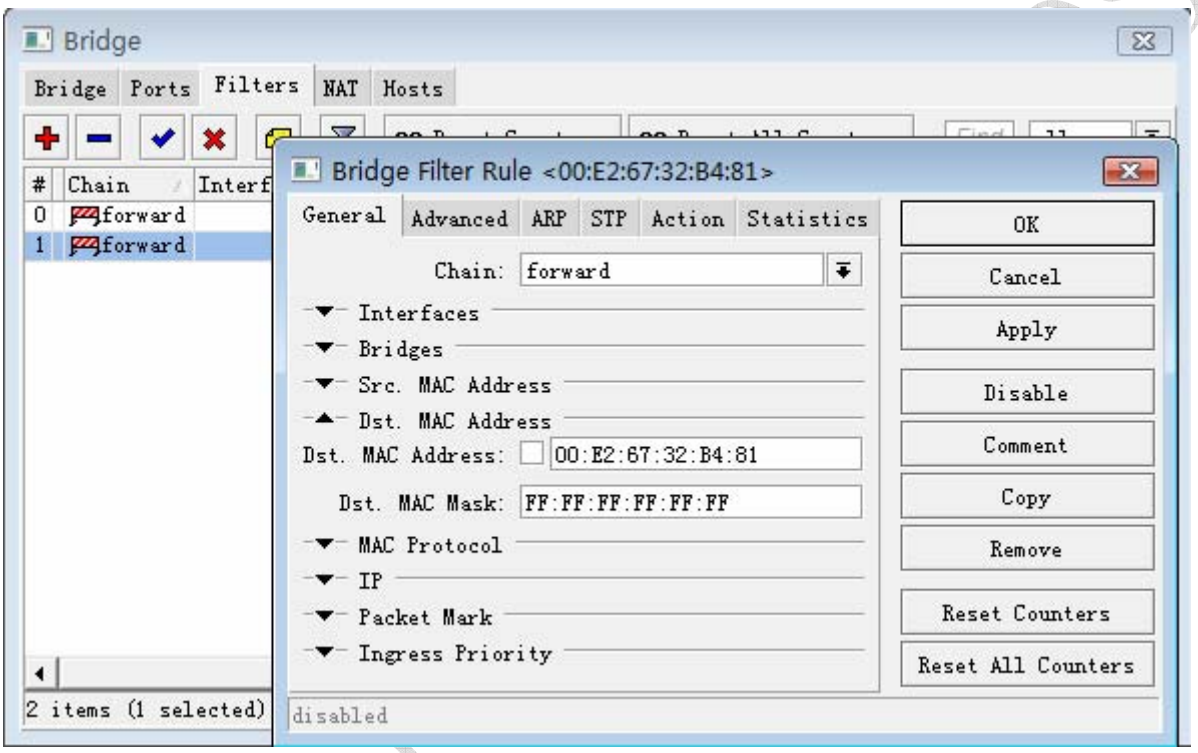
注意：我们设置 src-mac-address 时，后面跟着 MAC 掩码，这个掩码和我们 IP 层的子网掩码类是，只是 MAC 掩码是按照十六进制换算，十六进制的 FF 与 IP 掩码的 255 是相同，规定网络范围，因为这里是过滤一个台主机的 MAC 地址，所以我们设置 MAC 子网掩码为 FF:FF:FF:FF:FF:FF。

2、目标 MAC 地址

反过来从外网访问一个该主机，则是目标 MAC 过滤，只是之前我们设置的是 scr-mac-address，反过来填写目标的 MAC，即 dst-mac-address，我们还是用之前的 MAC 地址做事例

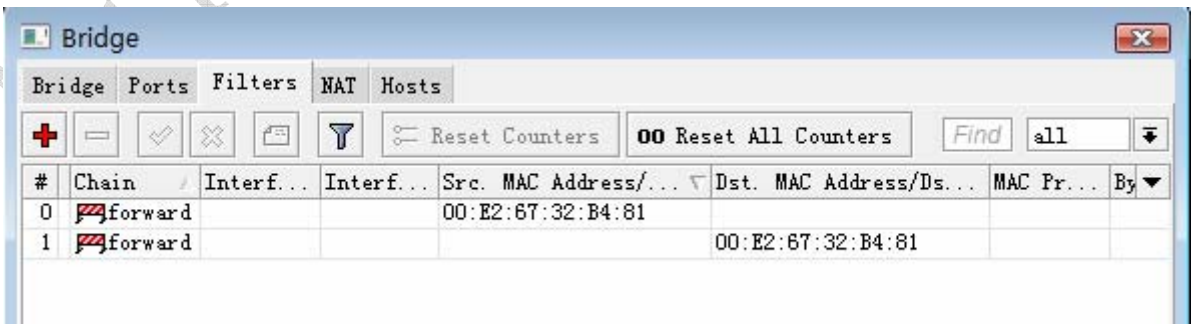


我们添加目标 MAC 地址过滤规则，选择 dst-mac-address=00:E2:67:32:B4:81，dst-mac-address 默认为全 FF。



Action 同样选择 drop，丢弃到该目标 MAC 的数据。

下面我们可以在 filter 中看到 2 条规则，分别是控制从源地址和目标地址的数据，这样设置后，我们可以理解为对 00:E2:67:32:B4:81 主机数据的双向过滤。



过滤指定厂商的 MAC 地址

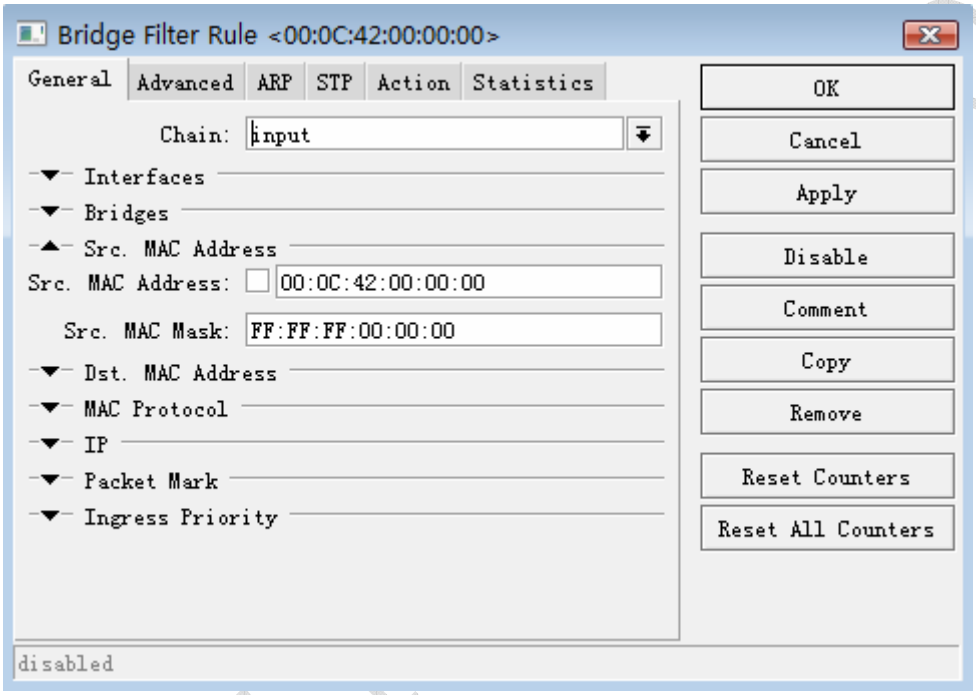
我们知道所有的网络设备都有一个 6 位的 MAC 地址，前 3 位为生产厂商标示，后 3 位为设备编号，当我们在做无线网桥的时候，只允许特定某一厂商的网卡连接到 RouterOS，可以通过 Bridge 的防火墙控制 MAC 地址，限制某一类的 MAC 不能连接到 RouterOS 设备，或者通过 RouterOS 设备。

例如，我们的一台 RouterBOARD 设备要求只能允许其他 RouterBOARD 的设备连接，可以通过 brigde fiter 控制，由于每个 RouterBOARD 的以太网卡 MAC 地址都是前 3 位都是以 00:0C:42 开头，我们只需要允许前 3 位 MAC 为 00:0C:42 的 MAC 通过就可以。

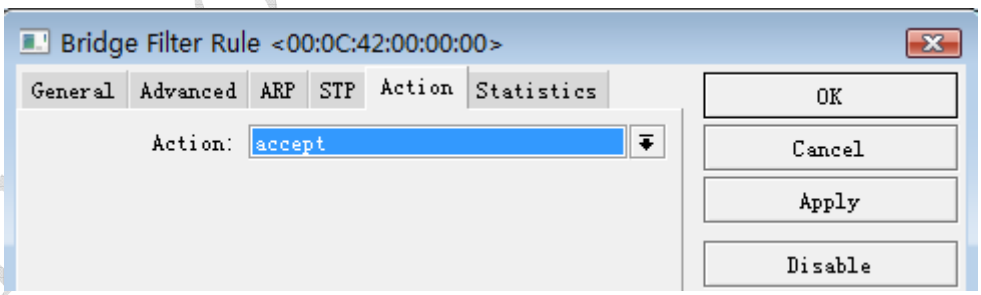
在设置为 bridge 的接口参数后，我们在 filter 中配置 2 条 input 规则，限制除了 MAC 地址前 3 位是 00:0C:42 能连接 RouterOS，其他的都拒绝掉。

根据 RouterOS 防火墙原理，分别需要设置两条规则，一条是接受 MAC 地址前 3 位是 00:0C:42 的 MAC 地址，第二条是丢弃其他所有的 MAC 数据。

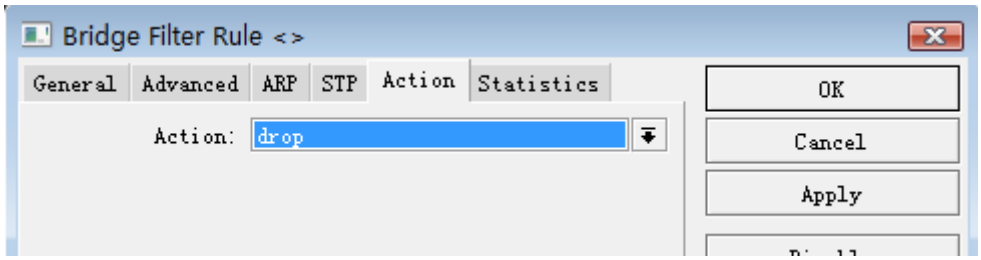
第一条规则，设置 src-mac-address=00:0C:42:00:00:00/src-mac-mask=FF:FF:FF:00:00:00



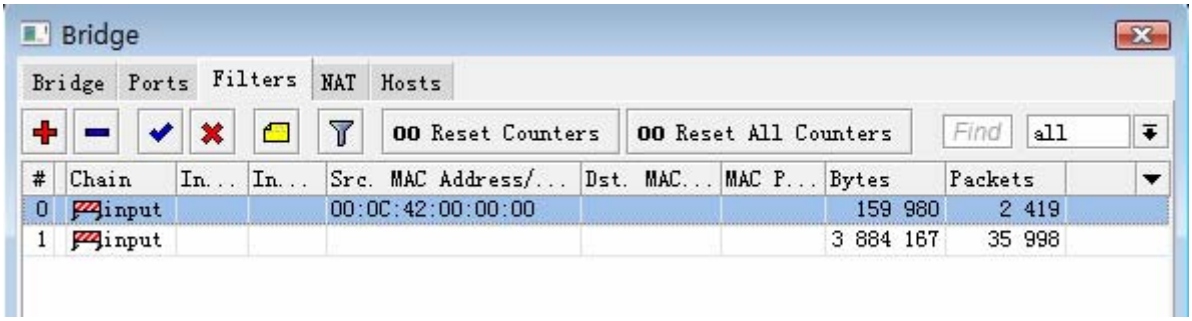
在 action 中选择 accpet 接受，数据通过



第二条规则，是丢弃其他所有的 MAC 数据



配置完成后，如下：



第十六章 虚拟路由冗余协议(VRRP)

虚拟路由冗余协议 Virtual Router Redundancy Protocol (VRRP)，MikroTik RouterOS VRRP 协议遵循 RFC2338。VRRP 协议是保证访问一些资源不会中断，即通过多台路由器组成一个网关集合，如果其中一台路由器出现故障，会自动启用另外一台。两个或多个路由器建立起一个动态的虚拟集合，每一个路由器都可以参与处理数据，这个集合最大不能超过 255 个虚拟路由器(可参考虚拟路由协议)。一般现在的路由器都支持该协议。

利用 VRRP 聚合功能提供高效的路由器运行方式，不在需要复杂的脚本 ping 监测

规格

需要功能包: ***system***
软件等级: *Level1*
操作路径: ***/interface vrrp***

许多 VRRP 路由器可用组成一个虚拟路由器集合。在一个网络中最大可用支持相同 VRID（虚拟路由 ID）255 个。每个路由器都必须设置一个优先参数，每个 VRRP 配置通一个虚拟的网卡绑定在一个真实的网卡上。VRRP 地址放入虚拟的 VRRP 网卡上。VRRP **Master** 状态显示为 **running** 标志，虚拟网卡上的地址被激活，其他属于 **backup**（即优先级低的 VRRP 路由）停止运行。

虚拟路由冗余协议是一种为路由提供高效率的路由选择协议。一个或多个 IP 地址可以分配到一个虚拟路由上，一个虚拟路由节点应该具备以下状态：

- **MASTER** 状态，一个节点回答所有的请求给相应请求的 IP 地址。仅只有一个 MASTER 路由器在虚拟路由中。每隔一段时间这个主节点发出 VRRP 广播包给所有 backup 路由器。
- **BACKUP** 状态, VRRP 路由器监视 Master 路由器的状态。它不会回答任何来至相应 IP 地址的请求，当 MASTER 路由器无法工作时（假设至少三次 VRRP 数据连接丢失），选择过程发生，新的 MASTER 会根据优先级产生。

注： VRRP 不能运行在 VLAN 接口上，VLAN 的接口 MAC 地址于与运行在物理网卡 MAC 地址是不同的 。

16.1 VRRP 路由

操作路径: ***/interface vrrp***

属性描述

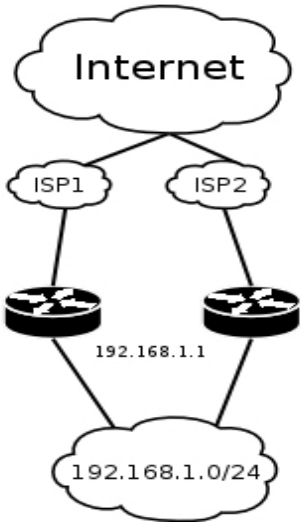
arp (disabled | enabled | proxy-arp | reply-only;默认: **enabled**) – 地址解析协议 Address Resolution Protocol
authentication (none | simple | ah; default: **none**) – 使用 VRRP 消息数据包的验证方式。
none – 没有证明
simple – 纯文本验证
ah – 验证头使用 HMAC-MD5-96 算法
backup (*只读: flag*) – 是否为备份状态
interface (*name*) – 运行接口的名称
interval (*整型*: 1..255; 默认 t: **1**) – VRRP 状态更新间隔秒钟。定义多少频率发送 VRRP 信息数据包。
mac-address (*MAC address*) – VRRP 的 MAC 地址 address。符合 RFC 协议，任何 VRRP 都应该只有唯一的 MAC 地址。
master (*只读: flag*) – 是否为 master 状态
mtu (*整型*; 默认: **1500**) – 最大传输单位
name (*name*) – VRRP 分配的名称
on-backup (*name*; 默认: **""**) – 当节点为 backup 状态执行的脚本
on-master (*name*; 默认: **""**) -当节点为 master 状态执行的脚本
password (*文本*; 默认: **""**) – 需要验证时的密码，不使用验证时可以被忽略。8 位字符长文本字符串（为纯文本验证方式）；16 位字符长文本字符串（为需要 128 位 key 的 AH 验证）
preemption-mode (yes | no; 默认: **yes**) – 是否启用优先模式。
no – 一个 backup 节点在当前的 master 失效之前，是不会选择 master ，即使该 backup 的优先高于当前 master 的级别
yes – 该节点总是拥有最高优先级。
vrid (*整型*: 0-255; 默认: **1**) – 虚拟路由的身份号(必须是在接口（interface）上是唯一的)
priority (*整型*: 1-255; 默认: **100**) – 当前节点的优先级(高的数值代表高的优先级)

注: 所有同一个集合的节点，必须使相同的 **vrid**, **interval**, **preemption-mode**, **authentication** 和 **password**. 第 255 的优先级被保留为真正的虚拟路由的主机 IP 地址。

添加一个 VRRP 事例在 ether1 的接口上，一个虚拟路由的 **vrid** 设置为 **1**，因为是虚拟路由的主机，所有优先级为 255：

```
[admin@MikroTik] interface vrrp> add interface=ether1 vrid=1 priority=255
[admin@MikroTik] interface vrrp> print
Flags: X - disabled, I - invalid, R - running, M - master, B - backup
 0   RM name="vrrp1" mtu=1500 mac-address=00:00:5E:00:01:01 arp=enabled
       interface=ether1 vrid=1 priority=255 interval=1 preemption-mode=yes
       authentication=none password="" on-backup="" on-master=""
[admin@MikroTik] ip vrrp>
```

16.2 简单的 VRRP 事例



VRRP 协议能被用于一个冗余的无缝 Internet 连接，让我们假设有 192.168.1.0/24 网络和我们需要提供高效的 Internet 连接。这个网络需要启用 NAT（VRRP 网络需要使用公网 IP，使用动态路由协议如 BGP 或 OSPF）。我们连接到两个不同的 ISP，且一个被设置为最优先(如，价格便宜或者速度更快的)..

这个事例讲解如何配置 VRRP 在两个路由器上。路由器必须初始化配置：网卡已被启用、每个网卡配置好了 IP 地址、路由表这种正确（至少一个默认路由）。 SRC-NAT 或 masquerading（伪装）应配置好。具体设置请参见相关的内容

我们将 192.168.0/24 的网络连接到名为 **local** 网卡的两台 VRRP 路由器上

配置 Master VRRP 路由器

首先我们应创建一个 VRRP 在这个路由器上。我们将使用 255 的优先值，该路由器将被设置为优先路由器

```
[admin@MikroTik] interface vrrp> add interface=local priority=255
[admin@MikroTik] interface vrrp> print
Flags: X - disabled, I - invalid, R - running, M - master, B - backup
 0   RM name="vrrp1" mtu=1500 mac-address=00:00:5E:00:01:01 arp=enabled
       interface=local vrid=1 priority=255 interval=1 preemption-mode=yes
       authentication=none password="" on-backup="" on-master=""
[admin@MikroTik] interface vrrp>
```

下一步，IP 地址应被添加到 VRRP 中

```
[admin@MikroTik] ip address> add address=192.168.1.1/24 interface=vrrp1
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.0.0.1/24        10.0.0.0        10.0.0.255      public
  1   192.168.1.2/24     192.168.1.0     192.168.1.255   local
  2   192.168.1.1/24     192.168.1.0     192.168.1.255   vrrp1
[admin@MikroTik] ip address>
```

配置 Backup VRRP 路由器

现在我们将创建一个低优先级的 VRRP 路由（我们可以使用默认值 **100**），因此路由器将优先选择 backup：

```
[admin@MikroTik] interface vrrp> add interface=local
[admin@MikroTik] ip vrrp> print
Flags: X - disabled, I - invalid, R - running, M - master, B - backup
 0    B name="vrrp1" mtu=1500 mac-address=00:00:5E:00:01:01 arp=enabled
       interface=local vrid=1 priority=100 interval=1 preemption-mode=yes
       authentication=none password="" on-backup="" on-master=""
[admin@MikroTik] interface vrrp>
```

现在我们添加同样的地址到备份 VRRP 路由中：

```
[admin@MikroTik] ip address> add address=192.168.1.1/24 interface=vrrp1
```

现在，当我们断开 master 路由器，在几秒钟后备份路由将选择 master 状态：

```
[admin@MikroTik] interface vrrp> print
Flags: X - disabled, I - invalid, R - running, M - master, B - backup
 0   RM name="vrrp1" mtu=1500 mac-address=00:00:5E:00:01:01 arp=enabled
       interface=local vrid=1 priority=100 interval=1 preemption-mode=yes
       authentication=none password="" on-backup="" on-master=""
[admin@MikroTik] interface vrrp>
```

第十七章 HotSpot 热点认证网关

17.1 HotSpot 介绍

HotSpot 是一种通过要求用户认证来访问某些网络资源的方法。用户可以使用几乎任何网页浏览器（HTTP 或 HTTPS 协议）登陆，所以他们不需要安装任何附加的插件。RouterOS 会自动计算正常运行时间以及每个客户使用的流量，并且也能把这个信息发送到 RADIUS 服务器。HotSpot 系统可以限制每个特定用户的比特率，总流量，运行时间以及涉及的其他参数。

Hotspot 热点 web 服务认证是一种友好的 web 方式的认证系统，在此种认证方式中，系统将自动要求未认证用户打开认证网页，验证通过后，便可连接到因特网，未认证用户无论输入任何一个网站地址，都会被强制到一个认证界面，要求用户进行认证。配置了 Walled Garden 特性后，允许用户不需要提前认证就可以访问一些网页。

获取地址

首先，一个客户必须先获得一个 IP 地址。它可以通过设置被静态 IP 地址，或者获取一个 DHCP 服务器的所分配的 IP 地址。如果需要的话，DHCP 服务器可以提供绑定分发 IP 地址到客户 MAC 地址的途径。

此外，HotSpot 服务器能自动分配给任何客户端的虚拟 IP 地址，分配来自 Hotspot 建立的 IP 池。这个特性对那些不愿意修改 IP（或不清楚，缺乏网络技术）。如果用户和 Hotspot 网关不在相同子网，Hotspot 通过 ARP 广播的方式强迫分配一个 IP 给用户，用户不会注意到这个转变（例如：在用户配置不会有任何改变），但路由器本身则看到完全不同（在 hotspot host 中可以看到被转化了的地址），这项技术叫做一对一 NAT，但它也以 RouterOS 2.8 版本中叫做的“即插即用”。

一对一 NAT 接收来自已连接网络接口的任何向内地址，并完成一个网络地址翻译。客户可以使用任何预先配置的地址（注意要求配置**网关**）。如果一对一 NAT 特性被设置为翻译一个客户的地址为一个公网 IP 地址，那么这个客户就甚至可以运行一个服务器或任何其他需要公网 IP 地址的服务。这个 NAT 将在数据包被路由器接收后就立即改变包的源地址。

注意，你使用一对一 NAT 时，必须在该接口上启用 **arp** 模式。

认证之前

当在一个接口上启用 HotSpot 时，系统自动配置对所有未登陆用户显示登陆页面。这个是通过添加动态目的 NAT 规则完成的，你可以在一个运行中的 HotSpot 系统上观察的到。这些规则是用来把未认证用户的所有 HTTP 及 HTTPS 请求重定向到 HotSpot servlet（认证过程，例如：登陆页面）。其他一些规则将在该章后面专门部分进行讲述。

在配置好 Hotspot 后，打开任何 HTTP 页面都会产生 HotSpot servlet 登陆页面（可以通过自行定义登陆页面），所有访问 Hotspot 网关以外的资源，都会跳转到登陆页面，因此必须在 HotSpot 网关配置一个合法的 DNS。

Walled Garden

有时希望对某些服务不要求认证（例如让客户不需要认证就访问公司的服务器），或者一些服务要求认证（例如，用户访问一个内部文件服务器或其他限制区域）。这些都可以通过 Walled Garden 系统实现。

当一个未登陆用户请求 Walled Garden 中允许的服务时，HotSpot 网关不会阻拦它，或者如果是 HTTP，就简单地把请求重定向到原来的目的（或定向到一个指定的父级代理）。

为了执行 Walled Garden 对 HTTP 请求的特性，专门设计了一个嵌入的 web 代理服务器，所有来自未认证用户的请求是从这个代理通过。注意嵌入的代理服务器还没有高速缓存功能。还要注意这个嵌入代理服务器是在 **system** 软件功能包里并不需要 **web-proxy** 功能包。它是在**/ip proxy** 下面配置的。

认证

现在有 5 种不同的认证方法。你可以同时使用一个或多个：

- **HTTP PAP** - 最简单的方法。显示 HotSpot 登陆页并以纯文本格式获取认证信息（如：用户名和密码）。注意当在网络传输时，密码是没有加密的。
- **HTTP CHAP** - 标准方式，在登陆页包含了 CHAP 询问。CHAP MD5 散列询问与用户密码一起使用来计算将被发送到 HotSpot 网关的字符串。散列结果（作为一个密码）与用户名一起通过网络发送到 HotSpot 服务器（所以，密码是从来不以纯文本格式通过 IP 网络发送的）。在客户端，MD5 算法通过 JavaScript applet 执行，所以如果一个浏览器不支持 JavaScript（比如，Internet Explorer 2.0 或一些 PDA 浏览器），将不能认证用户。可以允许未加密密码，即打开 HTTP PAP 认证方式被接受，但并不推荐使用这个特性（出于安全考虑）。
- **HTTPS** - 与 HTTP PAP 一样，但对加密传输使用了 SSL 协议。HotSpot 用户只发送没有附加散列的密码（注意没有必要担心纯文本密码在网络上的暴露，因为传输本身是加密的）。在另一种情况，HTTP POST 方法（如果不可能，那么用 HTTP GET 方法）用于向 HotSpot 网关发送数据。
- **HTTP cookie** - 在每次成功登陆之后，会有一个 cookie 发送到 web 浏览器，同时被添加到活动 HTTP cookie 列表。这个 cookie 将与存储在 HotSpot 网关的相比较，并仅当源 MAC 地址及随机生成的 ID 与存储在网关的相匹配。这个方法只可以与 HTTP PAP， HTTP CHAP 或 HTTPS 方法一起使用，不然的话没有其他方式可以产生 cookie。
- **MAC address** - 将用客户端的 MAC 地址与用户帐号同时作为用户名。

HotSpot 可以通过询问本地用户数据库或 RADIUS 服务器认证用户（本地数据库会被先询问，然后是 RADIUS 服务器）。如果通过 RADIUS 服务器认证 HTTP cookie，那么路由器将在 cookie 被第一次产生时发送相同的信息到服务器。如果认

证在本地完成，那么符合该用户的信息将会被调用， 否则将会调用 RADIUS 中的参数。如果要知道更多关于 RADIUS 服务器工作的信息，请参见其相应的 Radius 手册。

HTTP PAP 方法也使得通过请求页 /login?username=username&password=password。如果你想使用 telnet 连接登陆，准确的 HTTP 请求应该这样：**GET /login?username=username&password=password HTTP/1.0**

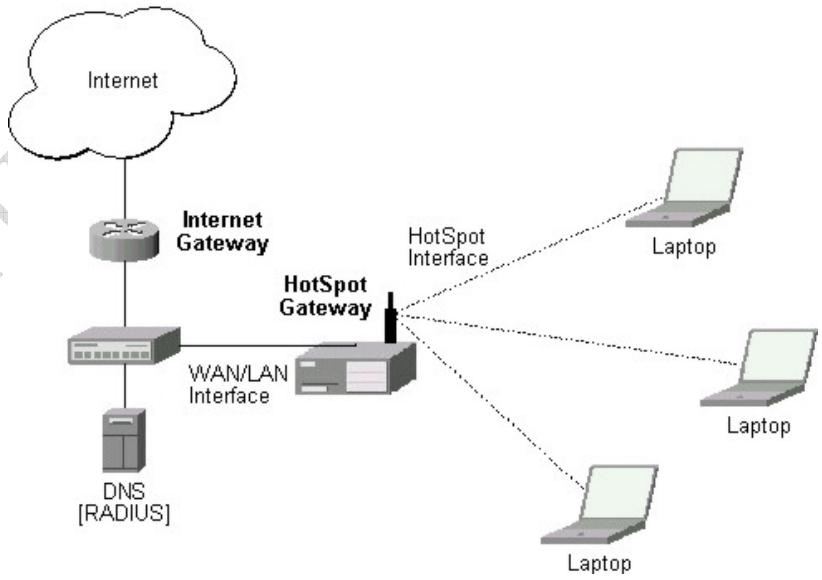
配置菜单

- **/ip hotspot** - HotSpot 上的特定界面(每个界面一个服务器)。HotSpot 服务器必须添加在这个目录中，HotSpot 系统才能够在一个界面上工作。
- **/ip hotspot profile** - HotSpot 服务器概要。影响 HotSpot 客户登陆过程的设置在这里进行。多个 HotSpot 服务器可以使用同样的概要信息。
- **/ip hotspot host** - 所有 HotSpot 接口上的活动网络主机的动态列表。在这里你可以找到 IP 地址与一对一 NAT 的绑定
- **/ip hotspot ip-binding** - 将 IP 地址绑定到主机 HotSpot 接口的规则
- **/ip hotspot service-port** - 一对一 NAT 地址翻译助手
- **/ip hotspot walled-garden** – HTTP 等级的 Walled Garden 规则（DNS 名， HTTP 请求子串)
- **/ip hotspot walled-garden ip** – IP 等级的 Walled Garden 规则 (IP 地址，IP 协议)
- **/ip hotspot user** –本地 HotSpot 系统用户
- **/ip hotspot user profile** – 本地 HotSpot 系统用户组规则
- **/ip hotspot active** - 所有已认证 HotSpot 用户的动态列表
- **/ip hotspot cookie** - 所有合法的 HTTP cookie 动态列表

下面是一个简单的 Hotspot 事例，HotSpot 网关应该至少有两个网络接口：

1. HotSpot 接口，用于连接 HotSpot 客户
2. LAN/WAN 接口，用于访问网络资源。例如：DNS 和 RADIUS 服务器应该可达

下面的图表显示了一个简单的 HotSpot 设置。



HotSpot 接口应该分配一个 IP 地址。物理网络连接应该建立在 HotSpot 用户的电脑和网关之间。它可以是无线（无线网卡需要在 AP 上注册），或者有线的（NIC 网卡需要连接到一个集线器或一个交换机）。

当 ISP 需要在有线或者无线网络中建立 Hotspot 热点认证系统，如：小区、酒店、机场和其他公共场所。一个普通的 Hotspot 网络建立在一个外网接口和一个内部网络接口下，我们需要对内网用户作认证上网。

注： 在 2.9 版本的 RouterOS Hotspot 功能包采用的是端口代理的方式连接，在启用 Hotspot 接口后 UpNp 即插即用功能自动开启，通过在/ip hotspot host 列表中可以查询相应的信息。

17.2 HotSpot 接口设置

操作路径：***/ip hotspot***

HotSpot 系统建立在一个独立的网络接口，你可以在不同的网络接口（以太网卡、无线网卡等）上配置不同的 HotSpot 服务器。

属性描述

addresses-per-mac (*整型* | unlimited; 默认: **2**) - 允许与特定 MAC 地址绑定的 IP 地址数量（降低一个 IP 模拟多个 MAC 的攻击）

unlimited - 每个 MAC 对应 IP 地址数量无限制

address-pool (*名称* | none; 默认: **none**) - 运行一对一 NAT 的 IP 地址池。你可以选择不使用一对一 NAT

none - 对这个 HotSpot 接口的客户不使用一对一 NAT

HTTPS (*只读: flag*) - HTTPS 服务是否在这个接口上实际在运行（它在这个服务器概要中设置，并且在路由器中输入了一个合法的认证）

idle-timeout (*时间* | none; 默认: **00:05:00**) - 对未认证客户的空闲超时时间（非活动的最大时间）。它用于探测客户没有使用外部网络（因特网），例如，没有收到来自某个客户的流量也没有流出路由器的流量。达到超时时间后，用户将被主机注销清除，用户所使用的地址也将被释放

none – 不切断空闲用户

interface (*名称*) - 运行 HotSpot 的接口

ip-of-dns-name (*只读: IP address*) - HotSpot 接口概要中设置的 HotSpot 网关 DNS 名称的 IP 地址

keepalive-timeout (*时间* | none;默认: **none**) - 对未认证客户的持活超时时间。用于探测客户的计算机是活动的并且是可达的。如果在这个期间探测失败，那么用户将被主机列表清除并且用户使用的地址也将被释放

none - 不切断不可达用户

profile (*名称*; 默认: **default**) - 接口的默认 HotSpot 概要

reset-html (*名称*) - 以原始的 HTML 文件重新覆盖已有的 HotSpot servlet。它用于你改变 servlet 之后且它不工作。

注： **addresses-per-mac** - 只有当地址池定义后，属性才能生效。

为了把 HotSpot 系统添加到本地接口，允许系统对每个客户进行一对一 NAT（来自 **HS-real** 地址池的地址将被用于 NAT）：

```
[admin@MikroTik] ip hotspot> add interface=local address-pool=HS-real
[admin@MikroTik] ip hotspot> print
Flags: X - disabled, I - invalid, S - HTTPS
 #   NAME                    INTERFACE    ADDRESS-POOL PROFILE IDLE-TIMEOUT
 0   hs-local                 local        HS-real      default 00:05:00
[admin@MikroTik] ip hotspot>
```

17.3 HotSpot 服务

操作路径：***/ip hotspot profile***

属性描述

dns-name (*文本*) - HotSpot 服务器的 DNS 名称。与 HotSpot 服务器名类似的 DNS 名。（它看起来像登陆页面位置）。这个名字会被自动地在 DNS 缓存中添加为一个静态 DNS。

hotspot-address (*IP address*; default: **0.0.0.0**) - HotSpot 服务器的 IP 地址

html-directory (*文本*; default: **""**) - 目录的名称（以 FTP 访问），它存储了 HTML servlet 页面（当改变路径时，如果路径不存在，默认页面会自动被复制到指定的目录中）

http-cookie-lifetime (*时间*; 默认: **3d**) - HTTP cookies 的有效时间

http-proxy (*IP 地址 s*; 默认: **0.0.0.0**) - HotSpot 服务器将作为一个代理服务器使用的对所有被通用代理系统打断并没在**/ip proxy direct** 列表中定义的代理服务器地址。如果没有特别指明，地址将在**/ip proxy** 下面的 **parent-proxy** 参数定义。如果这个也空缺，请求将被本地代理处理。

login-by (*多选项:* cookie | http-chap | http-pap | https | mac | trial; default: **cookie,http-chap**) - 使用的认证方法

cookie - 使用 HTTPcookie 认证，而不询问用户证明。以防客户没有 cookie，或者存储的用户名和密码对从上一次认证后不再合法，就将使用其他方法认证。可能仅和其他 HTTP 认证方法一同使用(HTTP-PAP, HTTP-CHAP 或 HTTPS)，因为第一次 cookie 是没有办法产生的。

http-chap - 对密码使用 MD5 散列算法的 CHAP 询问-回答的方法。这种方法很容易避免在一个不安全网络上发送清楚的文本密码。这个方法是默认的认证方法。

http-pap - 在网络中使用纯文本认证。请注意如果使用了这个方法，你的用户密码将在本地网络中暴露，所有可够侦听它们。

https - 使用加密了的 SSL 通道来传输用户与 HotSpot 服务器的通信。注意，为了使它能工作，必须对路由器输入一个合法的认证（参见认证管理的手册）。

mac - 试着先使用客户的 MAC 地址作为它的用户名。如果与本地用户数据库或 RADIUS 服务器匹配了，那么客户将不会被要求填写登陆表格就可以通过认证。

trial - 在一定时间内不会要求认证

radius-accounting (yes | no; 默认: **yes**) - 是否不时地在每个用户上发送 RADIUS 帐户管理信息（这个“不时”的时间是在 **radius-interim-update** 属性中定义的）

radius-interim-update (*time* | received; 默认: **received**) - 发送累计帐户报告的频率

0s – 与 **received** 相同

received - 使用接收自 RADIUS 服务器的任何值

rate-limit (*文本*; 默认: **""**) - 从路由器角度考虑以 **rx-rate[/tx-rate] [rx-burst-rate[/tx-burst-rate] [rx-burst-threshold[/tx-burst-threshold] [rx-burst-time[/tx-burst-time]]]]**格式表示的速率限制(其中 "rx" 是客户上传，"tx"是客户下载)。所有的速率都应该是带有‘k’(1,000s)或‘M’ (1,000,000s)的数字。如果 tx-rate 没有指定，rx-rate 和 tx-rate 一样。对于 tx-burst-rate 和 tx-burst-threshold 以及 tx-burst-time 也同理。如果 rx-burst-threshold 和 tx-burst-threshold 都没有指定 (但是 burst-rate 已指定)， rx-rate 和 tx-rate 将被做为 burst threshold 使用。如果 rx-burst-time 和 tx-burst-time 都没有指定，那么 1s 将会作为默认值使用。

smtp-server (*IP 地址*; 默认: **0.0.0.0**) - 默认 SMTP 服务器无条件地用于重定向

split-user-domain (yes | no; 默认: **no**) - 当用户名以"user@domain"或"domain\user"格式给出时，是否把用户名从域名中分离出来

ssl-certificate (*名称* | none; 默认: **none**) - 对 HTTPS 认证使用的 SSL 认证名。不用语其他认证方法

trial-uptime (*时间I时间*; 默认: **30m/1d**) - 仅当认证方式为询问时使用。

trial-user-profile (*名称*; 默认: **default**) - 仅当认证方法为询问时使用。指定询问用户将使用的用户概要

use-radius (yes | no; 默认: **no**) - 是否使用 RADIUS 认证 HotSpot 用户

*注：*如果 **dns-name** 属性没有指定，则 **hotspot-address** 将代替使用。如果 **hotspot-address** 也没有指定，那么将自动探测这两个值。如果启用了 RADIUS 验证，**/radius** 下的参数应正确配置。

属性描述

domain (*只读: 文本*) - 域名(如果从用户名中分离出来的话)
expires-in (*只读: 时间*) - cookie 合法存在的时间
mac-address (*只读: MAC 地址*) – 用户的 MAC 地址
user (*只读: 名称*) – 用户名

*注：*可以在相同的 MAC 地址上有多重的 cookie。例如，在同一台电脑上对没个 web 浏览器都可以有一个单独的 cookie。

Cookie 是可以过期的。默认的 cookie 合法时间为 3 天（72 小时），但对每个 HotSpot 服务是可以修改的，例如：

```
/ip hotspot profile set default http-cookie-lifetime=1d
```

获取合法 cookie 列表：

```
[admin@MikroTik] ip hotspot cookie> print
  # USER            DOMAIN          MAC-ADDRESS       EXPIRES-IN
  0 ex                            01:23:45:67:89:AB 23h54m16s
[admin@MikroTik] ip hotspot cookie>
```

17.4 Walled Garden 内院

HTTP 方式 Walled Garden

操作路径: ***/ip hotspot walled-garden***

Walled garden 是在允许未认证下访问某些资源，同样能用于需要认证访问的其他资源。例如：访问一些 HotSpot 服务提供商的基本信息或帐单选项。

这个目录只管理对 HTTP 和 HTTPS 协议的 Walled Garden。其他协议也可以包含进 Walled Garden，但要在其他地方配置（**/ip hotspot walled-garden ip**，参考本手册的下一部分）。

属性描述

action (allow | deny; 默认: **allow**) - 如果数据包和规则匹配则执行动作：
allow - 无需优先认证就允许访问页面
deny - 需要认证才能访问页面
dst-address (*IP 地址*) -目的 web 服务器的 IP 地址
dst-host (*wildcard*; 默认: **""**) - 目的 web 服务器的域名（这是一个通配符）
dst-port (*整型*; 默认: **""**) -客户发送请求的目的 TCP 端口
method (*文本*) – 请求的 HTTP 方法
path (*文本*; 默认: **""**) -请求的路径 (这是一个通配符)
server (*名称*) - 应用该规则的 HotSpot 服务器名
src-address (*IP 地址*) - 发送请求的用户 IP 地址

注： 通配符属性(**dst-host 和 dst-path**)匹配一个完整的串 （如：若设置为"example"，则它们不会匹配"example.com")。可用的通配符为'*' (匹配任意字符的任意数量)并且 '?' (匹配任何一个字符)。正则表达式也在这里接受，但如果属性做为一个正则表达式对待，那么它应该以图标(':')开始。

关于使用正则表达式：：

- **** 符号序列是用于在控制台输入****字符的
- **\.** 样式的意思为只是 **.** (在正则表达式单独的点表示任何符号)
- 显示在给出样式之前任何符号都不允许，我们在样式开始使用**^**符号
- 指定在给出样式之后任何符号都不允许，我们在样式结束的地方使用符号**$**

由于路由器不能解密请求，你也就不能对 HTTPS 请求使用 **path** 属性（也不应该使用——这就是 HTTPS 协议被创造的目的）。

允许未认证用户到 **www.example.com** 域**/paynow.html** 页面的请求：

```
[admin@MikroTik] ip hotspot walled-garden> add path="/paynow.html" \
\... dst-host="www.example.com"
[admin@MikroTik] ip hotspot walled-garden> print
Flags: X - disabled, D - dynamic
 0   dst-host="www.example.com" path="/paynow.html" action=allow
[admin@MikroTik] ip hotspot walled-garden>
```

IP 方式 Walled Garden

操作路径: ***/ip hotspot walled-garden ip***

这个目录管理类属 IP 请求的 Walled Garden。参见前面 HTTP 和 HTTPS 协议属性的部分（像实际的 DNS 名，HTTP 方法和在请求中使用的路径）。

属性描述

action (allow | deny; default: **allow**) -如果数据包和规则匹配则执行动作：
allow - 无需认证就允许访问页面
deny - 需要认证才能访问页面
reject -需要认证才能访问该页面，以防页面会被没有认证的 ICMP 拒绝信息访问，主机不可达将被产生
dst-address (*IP 地址*) -目的 web 服务器的 IP 地址
dst-host (*wildcard*; 默认: **""**) - 目的 web 服务器的域名（这是一个通配符）
dst-port (*整型*; default: **""**) -客户发送请求的目的 TCP 端口
protocol (*整型* | ddp egp encap ggp gre hmp icmp idpr-cmtp igmp ipencap ipip ipsec-ah ipsec-esp iso-tp4 ospf pup rdp rspf st tcp udp vmtp xns-idp xtp) - IP 协议名
server (*名称*) - 应用该规则的 HotSpot 服务器名
src-address (*IP 地址*) - 发送请求的用户 IP 地址

17.5 IP 绑定

操作路径: ***/ip hotspot ip-binding***

你可以静态地设置源 IP 地址（或 IP 网络）或源 MAC 地址的 NAT 翻译。你也可以允许一些地址绕过 HotSpot 认证（如：它们可以不必认证登陆就能访问外部资源），并阻止指定的地址认证登陆。

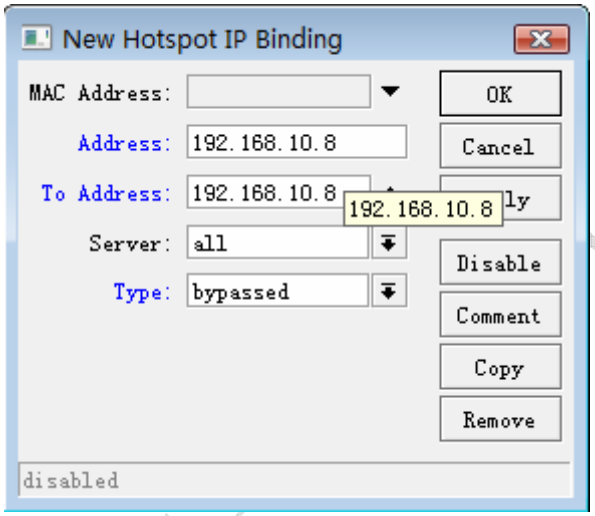
属性描述

address (*IP 地址* /子网掩码; 默认: **""**) - 源 IP 地址或客户网络
mac-address (*MAC 地址*; 默认: **""**) - 客户的源 MAC 地址

server (*名称*|all; 默认: **all**) - 客户将连接到的服务器名
to-address (*IP 地址*; 默认: **""**) - 把原始客户地址翻译成的 IP 地址。如果 **address** 属性是作为一个网络给定，那么这个将是翻译的开始地址（例如：第一个 **address** 被翻译为 **to-address**，**address+1** 翻译为 **to-address+1**，以此类推）
type (regular | bypassed | blocked) - 静态绑定条目类型
regular - 根据条目中设定的值进行一对一 NAT 翻译
bypassed – 绕过认证，即不需要通过 HotSpot 认证,扩音访问资源
blocked - 不会执行翻译，并且所有来自主机的数据包被丢弃

注： 这是一个有序列表，所以你可以把更详细的条目放在里表的顶部以超越比较低的普通条目。

例如，让 192.168.10.8 不通过认证，即可上网，并访问外部资源：



17.6 Hotspot 主机列表

操作路径: ***/ip hotspot host***

这个目录显示了所有连接到 HotSpot 服务下的活动主机，这个列表包含所有一对一 NAT 翻译。

属性描述

address (*只读: IP address*) - 客户的原始 IP 地址
authorized (*只读: flag*) - 客户是否成功地被 HotSpot 系统认证
blocked (*只读: flag*) - 如果访问在 walled-garden 中因为广告超时时间过期被阻止，则为真
bridge-port (*只读: 名称*) - 主机连接的真实物理接口。当 HotSpot 服务被放在一个桥接口以判定在桥中的主机实际的端口时，使用该值
bypass-hotspot (*只读: flag*) - 是否客户不需要 HotSpot 系统的认证
bytes-in (*只读: 整型*) - 路由器从客户接收的字节数
bytes-out (*只读: 整型*) - 路由器发送到客户的字节数
host-dead-time (*只读: 时间*) - 路由器没有从主机接收任何数据包（包括 ARP 回应，持活回应及用户流量）的时间。
idle-time (*只读: 时间*) – 闲置的时间
idle-timeout (*只读: 时间*) - 应用于用户的确切 **idle-timeout** 值。这个属性显示了用户空闲多久会被自动登出。
keepalive-timeout (*只读: 时间*) - 应用于用户的 **keepalive-timeout** 精确值。这个属性显示了用户的电脑在不可达状态多久会被自动登出
mac-address (*只读: MAC 地址*) - 实际的用户 MAC 地址

packets-in (*只读: 整型*) - 路由器接收客户的包数
packets-out (*只读: 整型*) - 路由器发送到客户的数据包数
server (*只读: 名称*) - 主机连接到的服务器名
static (*只读: flag*) - 翻译是否是来自静态 IP 绑定列表
to-address (*只读: IP 地址*) - 主机翻译成的原始 IP 地址
uptime (*只读: 时间*) - 用户的当前会话时间（如：用户在活动用户列表中已经多久了？）

命令描述

make-binding - 把可以个动态项目从这个列表复制到静态 IP 绑定列表
unnamed (*名称*) – 项目编号
comment (*文本*) - 产生客户对静态条目的评论
type (regular | bypassed | blocked) - 静态项目的类型

17.7 HotSpot 用户管理

主要对 Hotspot 的用户帐号、权限和用户参数分组进行管理。

操作路径: ***/ip hotspot user***
操作路径: ***/ip hotspot user profile***

热点用户管理用于普通用户分类设置，profile 用户组根据需要能将不同用户分类管理。

属性描述

address-pool (*名称* | none; 默认: **none**) - 用户用来分配 IP 的 IP 池名称。 这个就像 MikroTik RouterOS 早期版本的 **dhcp-pool** 一样工作。
none - 不向这个服务中的用户再分配 IP 地址
advertise (yes | no; 默认: **no**) - 是否对此服务启用强制广告弹出
advertise-interval (*多选项: time*; 默认: **30m,10m**) - 显示广告弹出之间间隔的设置。在列表完成后，最后一项值会后面所有广告使用
advertise-timeout (*time* | immediately never; 默认: **1m**) - 在使用 walled-garden 阻止网络访问之前等待广告显示的时间长度
advertise-url (*多选项: 文本*; 默认: **http://www.mikrotik.com/,http://www.routerboard.com/**) - 广告弹出显示的 URL 列表。这个列表是循环的，所以当到达最后一项时，下次显示的将是第一项
idle-timeout (*time* | none; 默认: **none**) - 授权用户空闲超时时间（未活动状态的最长时间）。它用于探测用户没有使用向外部网络或 hotspot 主机发送数据（如因特网），比如：没有任何流量从用户进入或从路由器流出。当达到超时时间，用户会被登出，丢出主机列表，用户使用的地址也会被清空，记录的会话时间也会由这个值减少。
none – 不切断空闲用户
incoming-filter (*名称*) - 应用于来自此服务用户向内数据包的防火墙链表的名称
incoming-packet-mark (*名称*) - 自动置于来自此服务每个用户所有数据包的包标记
keepalive-timeout (*time* | none; 默认值: **00:02:00**) - 授权客户的持续活超时时间。用于探测客户的电脑是在线。如果在这个期间检测失败，那么用户会被注销，用户使用的地址也会被清空。
none – 关闭次功能，不切断不在线用户
name (*名称*) - 服务参考名
on-login (*文本*; 默认: **""**) - 用户登入后运行的脚本名
on-logout (*文本*; 默认: **""**) -用户登出后运行的脚本名
open-status-page (always | http-login; 默认值: **always**) - 是否为授权用户显示状态页面使用 MAC 登入方法。如果你想放一些信息（例如：横幅或弹出窗口）在 alogin.html 页面将会很有用，这样所有的用户都可以看到它。

http-login - 如果 http 登入打开状态页面（包括 cookie 和 http 登入方法）
always - 如果 mac 登入打开 http 状态页面
outgoing-filter (*名称*) - 应用于此服务用户的向外流出的包的防火墙链表名称
outgoing-packet-mark (*名称*) - 自动设置在此概要每个用户的所有数据包的包标记
rate-limit (*文本*; 默认: **""**) – 从路由器角度来看的 **rx-rate[/tx-rate]** 格式的速率限制。 **[rx-burst-rate[/tx-burst-rate] [rx-burst-threshold[/tx-burst-threshold] [rx-burst-time[/tx-burst-time] [priority] [rx-rate-min[/tx-rate-min]]]]** (所以 "rx" 客户的上传，"tx"客户的下载)。所有速率必须以可选的'k' (1,000s) 或 'M' (1,000,000s)计算。如果 tx-rate 没有指定，则 rx-rate 和 tx-rate 一样。对于 tx-burst-rate 和 tx-burst-threshold 以及 tx-burst-time 也同理。如果 both rx-burst-threshold 和 tx-burst-threshold 都没有指定（但 burst-rate 指定了），那么 rx-rate 和 tx-rate 会作为脉冲串门限使用。如果 rx-burst-time 和 tx-burst-time 都没有指定，那么 1s 将设置为默认值。优先级从 1 到 8 取值，1 代表最高优先级，而 8 代表最地的。如果 rx-rate-min tx-rate-min 都没有指定那么 rx-rate 和 tx-rate 的值将被使用。rx-rate-min 和 tx-rate-min 的值不能超过 rx-rate 和 tx-rate。
session-timeout (*time*; 默认: **0s**) - session timeout (maximal allowed session time) for client. After this time, the user will be logged out unconditionally 把客户会话切断（最大允许的会话时间）。在这个时间过后，用户将会被无条件地登出。
0 – 不切断
shared-users (*整型*; 默认: **1**) - 同时登陆切使用同一个用户名的最大用户数量
status-autorefresh (*time* | none; 默认: **none**) – 热点 servlet 状态页面自动刷新间隔
transparent-proxy (yes | no; 默认: **yes**) - 是否对该概要授权用户使用透明的 HTTP 代理

*注：*当 idle-timeout 或者 session-timeout 到时，对该用户的连接会话将会被从 Hotspot 认证中注销，减少用户闲置对系统的超载。

操作路径: ***/ip hotspot user***

属性描述

address (*IP 地址*; 默认: **0.0.0.0**) - 静态 IP 地址。如果不是 0.0.0.0，那么客户将总是得到相同的 IP 地址。也就是说，对该用户只允许一个同时的登陆。任何一个已存在的地址都将使用嵌入的一对一 NAT 被这个地址取代。
bytes-in (*只读: 整型*) - 接收用户的总字节数
bytes-out (*只读: 整型*) - 发送给用户的总字节数
limit-bytes-in (*整型*; 默认: **0**) - 用户可以传输的最大字节数（例如：从接收到的字节数）
0 – 无限制
limit-bytes-out (*整型*; 默认: **0**) - 用户可以接收的最大自己数（例如：发送给用户的字节数）
0 – 无限制
limit-uptime (*时间*; 默认: **0s**) - 用户的总正常运行时间限制
0s – 无限制
mac-address (*MAC 地址*; 默认: **00:00:00:00:00:00**) - 静态 MAC 地址。如果不是 **00:00:00:00:00:00**，那么用户仅能从该 MAC 地址登陆
name (*名称*) - 用户名
packets-in (*只读: 整型*) - 接收到用户的最大包数量
packets-out (*只读: 整型*) - 发送给用户的最大包数
password (*文本*) - 用户口令
profile (*名称*; 默认值: **defualt**) – 用户资料
routes (*文本*) - 当用户连接上后将在热点网关注册的路由器。路由格式为“dst-address 网关 公制”（例如:“10.1.0.0/24 10.0.0.1 1”）。数个路由应用逗号分开指定。
server (*名称* | all; 默认: **all**) - 该用户允许登陆的服务器
uptime (*只读: time*) - 用户登陆的总时间

注： 如果 MAC 认证方法使用，客户的 MAC 地址可以被当作用户名使用（不需要口令）

字节限制是对每个用户的总限制（不像在**/ip hotspot active** 中的对每个会话的限制）。所以，如果一个用户已经下载了些东西，那么会话限制将显示总限制-（minus）已下载的。例如：如果对一个用户的下载限制为 100MB，并且用户已经下载了 30MB，那么在**/ip hotspot active** 中的登陆后会话下载限制将为 100MB - 30MB = 70MB。

如果一个用户达到了他的限制（bytes-in >= limit-bytes-in 或 bytes-out >= limit-bytes-out），他将再也不能登陆。如果用户通过本地用户数据库认证，那么每次他登出时统计就会被更新。意思是说如果一个用户现在登陆，那么统计现在也不会显示当前总的值。使用**/ip hotspot active** 子目录以查看当前用户会话的统计。

如果用户的 IP 地址被指定了，则仅允许一个同时的登陆。如果同一个认证在用户为激活时被再次使用，那么活动用户将自动被登出。

添加一个仅允许以 **01:23:45:67:89:AB** MAC 地址登陆的用户名和密码都为 **ex** 的用户，并限制 1 小时工作时间：

```
[admin@MikroTik] ip hotspot user> add name=ex password=ex \
\... mac-address=01:23:45:67:89:AB limit-uptime=1h
[admin@MikroTik] ip hotspot user> print
Flags: X - disabled
 #   SERVER    NAME                  ADDRESS      PROFILE   UPTIME
 0             ex                                 default  00:00:00
[admin@MikroTik] ip hotspot user> print detail
Flags: X - disabled
  0   name="ex" password="ex" mac-address=01:23:45:67:89:AB profile=default
      limit-uptime=01:00:00 uptime=00:00:00 bytes-in=0 bytes-out=0
      packets-in=0 packets-out=0
[admin@MikroTik] ip hotspot user>
```

17.8 Hotspot 在线用户

操作路径： ***/ip hotspot active***

现时用户列表显示当前已登陆了的用户。这里不能修改任何信息，除了使用 **remove** 命令将用户登出。

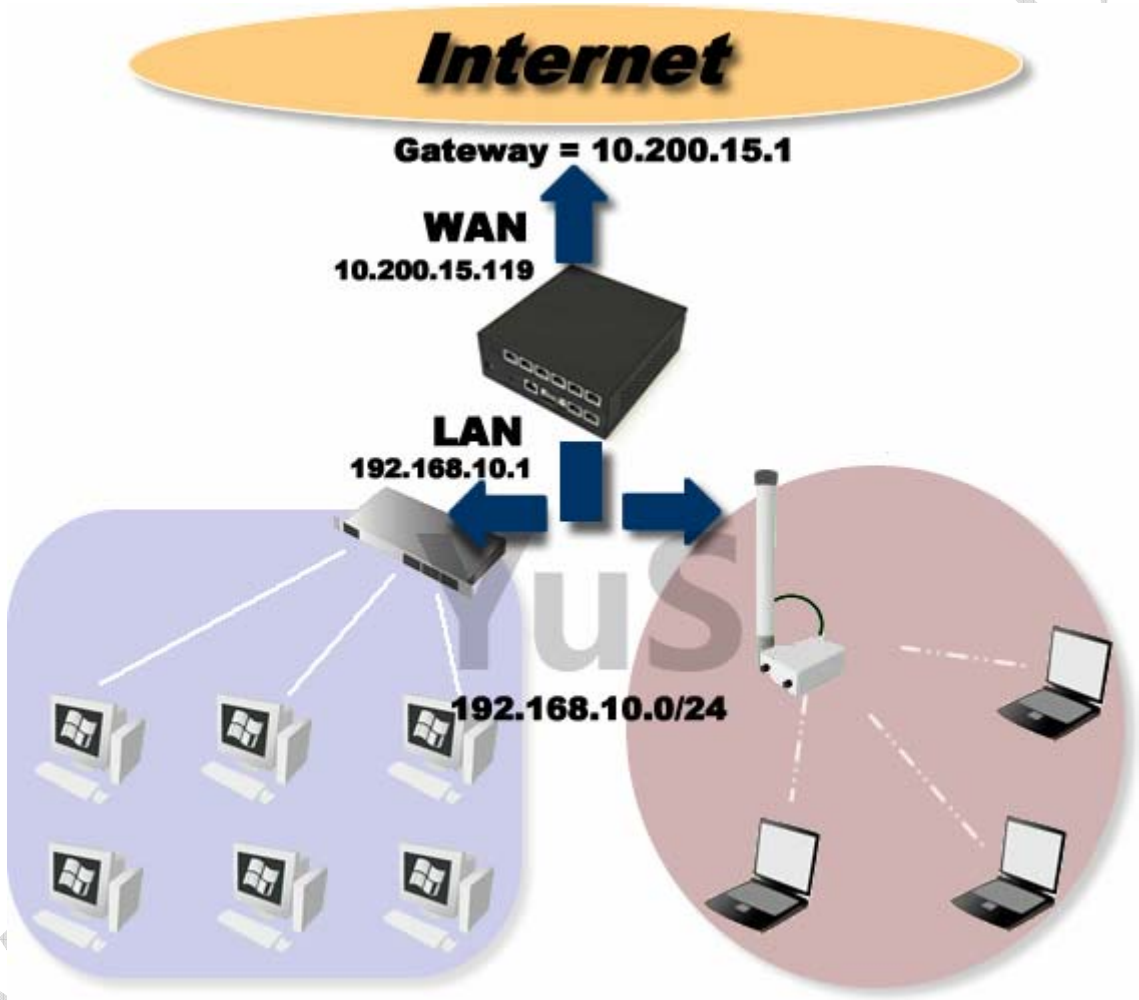
属性描述

address (*只读: IP 地址*) - 用户的 IP 地址
blocked (*只读: flag*) - 是否以广告将用户阻挡（例如：通常适当的广告未决）。
bytes-in (*只读: 整型*) - 路由器从客户收到的字节数
bytes-out (*只读: 整型*) - 由器发送到客户的字节数
domain (*只读: 文本*) - 用户范围（如果从用户名中分离出来）
idle-time (*只读: 时间*) - 用户被闲置的时间
idle-timeout (*只读: 时间*) -应用于该用户的 **idle-timeout** 精确值。这个属性显示他被自动登出的闲置时间
keepalive-timeout (*只读: 时间*) - 应用于该用户的 **keepalive-timeout** 精确值。该属性描述了用户的电脑不可达多久才会被自动登出
limit-bytes-in (*只读: 整型*) - 用户被允许发送给路由器的最大字节数
limit-bytes-out (*只读: 整型*) - 路由器被允许发送到客户的最大字节数
login-by (*多选项, 只读:* cookie | http-chap | http-pap | https | mac | trial) - 用户用的认证方法
mac-address (*只读: MAC 地址*) – 用户的实际 MAC 地址

packets-in (*只读: 整型*) - 路由器接受来自客户的包数量
packets-out (*只读: 整型*) - 路由器发送给客户的包数量
radius (*只读:* yes | no) - 用户是否通过 RADIUS 认证
server (*只读: 名称*) - 用户登陆所指定的服务器
session-time-left (*只读: time*) - 应用于该用户的 **session-time-left** 精确值。这个属性显示了用户在自动被注销前保持的登入状态时间
uptime (*只读: time*) -当前用户的会话时间 (例如：用户登入的时间)
user (*只读: 名称*) - 用户名

17.9 Hotspot 配置事例

我们根据下面的网络拓扑结构为例：

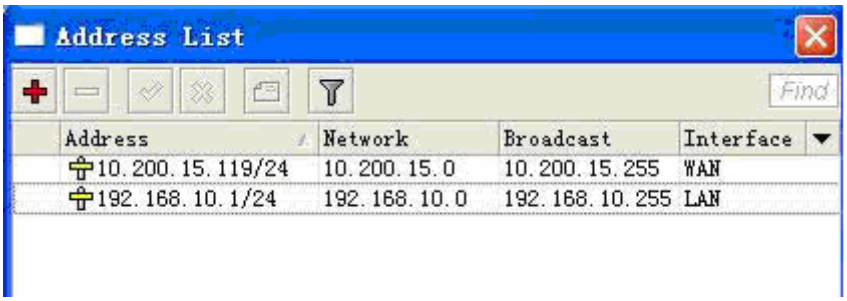


一个网关路由器的网络参数如下：

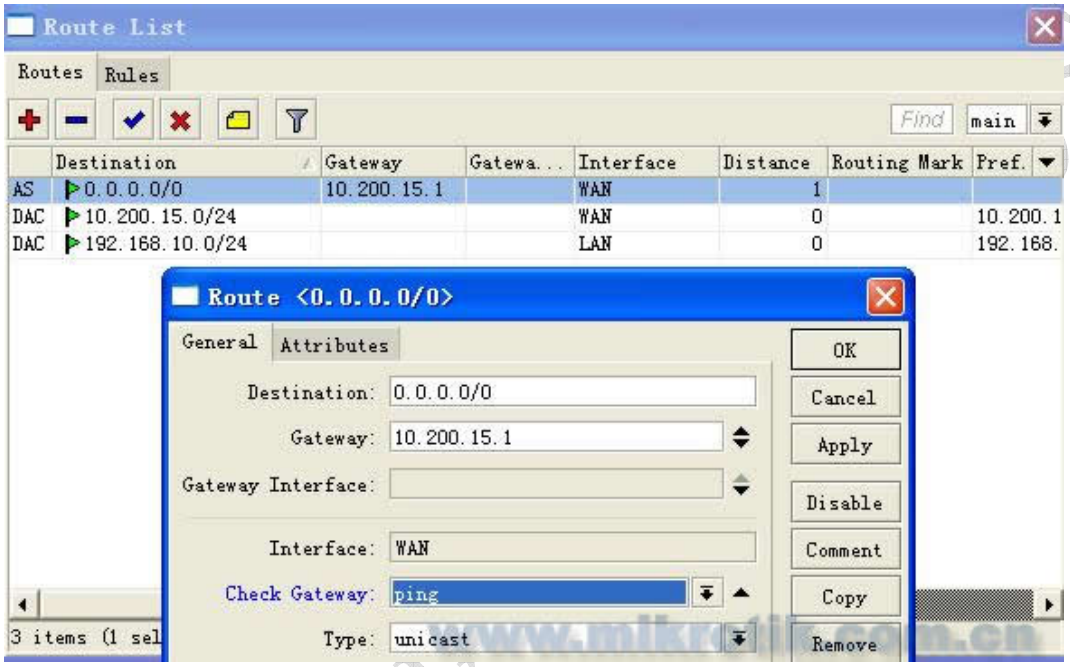
```
WAN 口对应外网 IP 为 10.200.15.119/24，网关为 10.200.15.1
LAN 口对应内网 IP 为 192.168.10.1/24
DNS：61.139.2.69
```

在根据这些参数我们需要先配置好 IP 地址、网关和 DNS，并打开 DNS 缓存等。

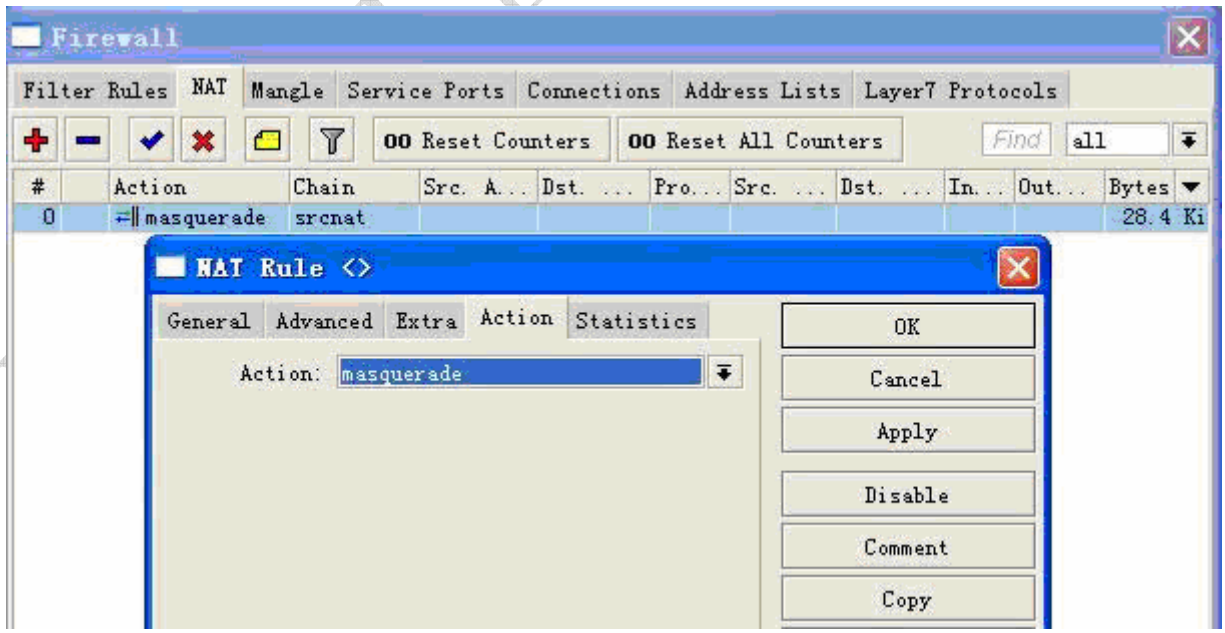
进入 ip address 配置 IP 地址：



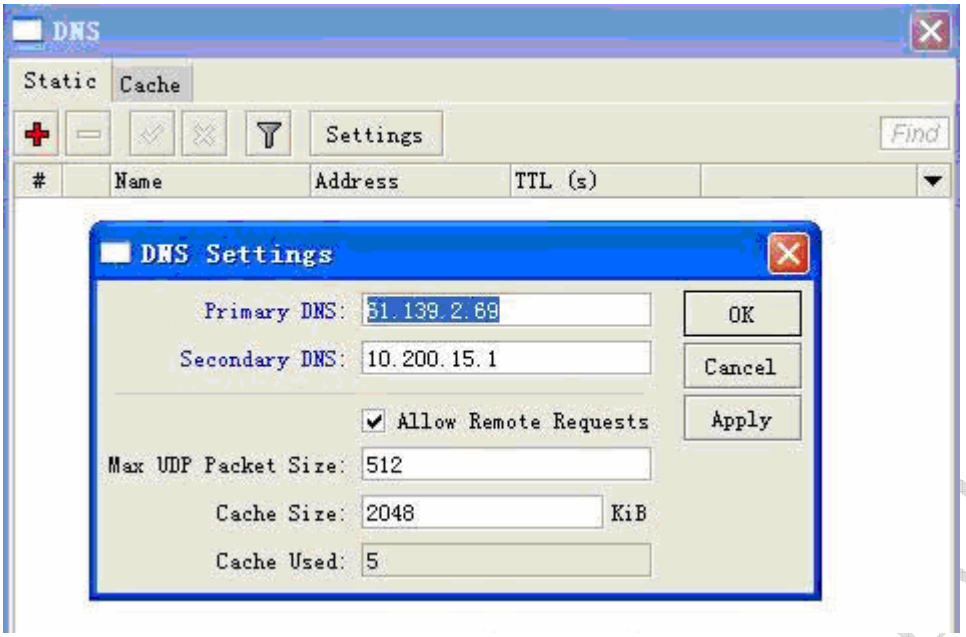
进入 ip route 配置网关



进入 ip firewall nat 设置好 NAT 伪装：



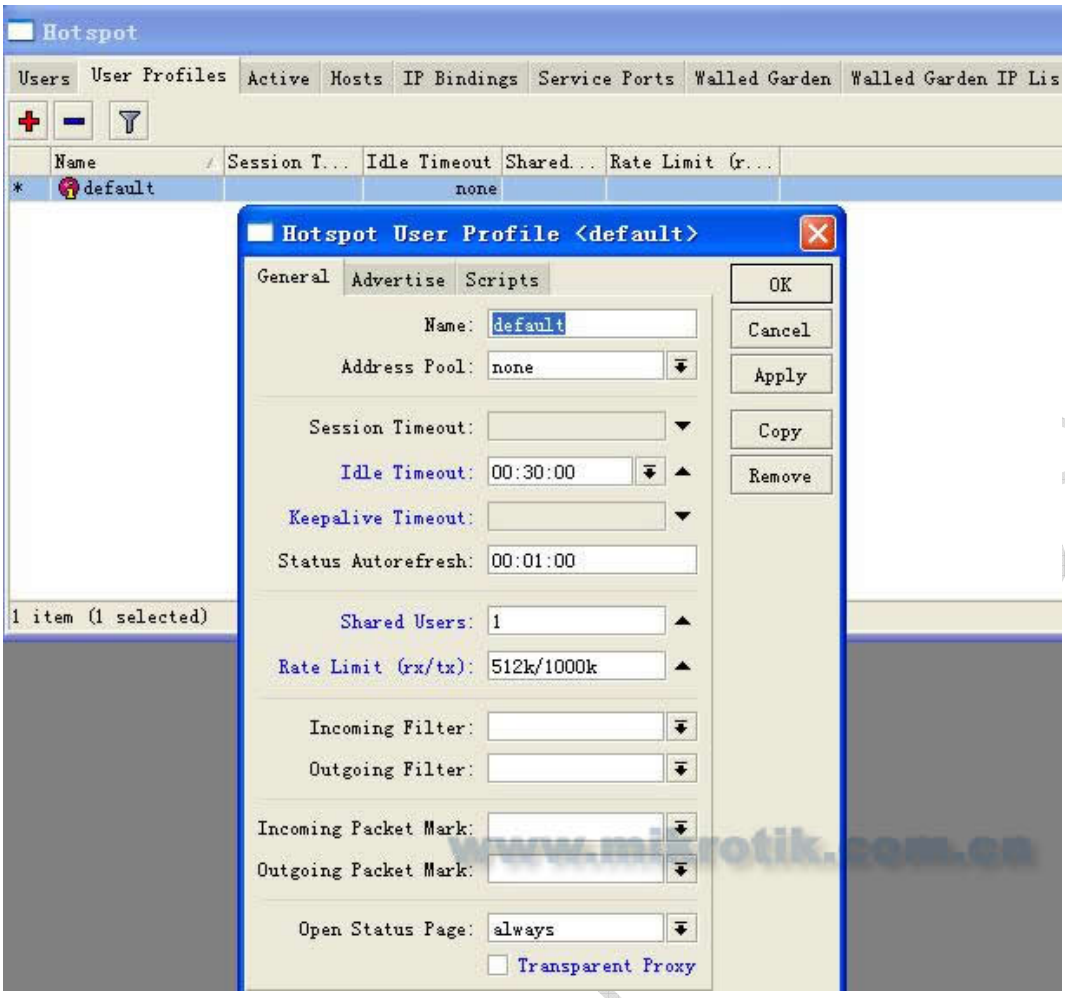
进入 ip dns 配置 DNS 缓存：



现在我们的基本参数已经配置完成，现在我们需要配置的 Hotspot 参数，配置 Hotspot 参数的基本流程是：

1、先进入 ip hotspot user profile 设置用户分组规则
2、然后在 ip hotspot user 添加用户的帐号
3、进入 ip hotspot server profile 配置服务器规则
4、在 ip pool 中分配 IP 地址段，根据需要启用 DHCP 服务
5、在 ip hotspot server 添加并启用 hotspot 服务

现在我们进入 ip hotspot，并配置 ip hotspot use profile

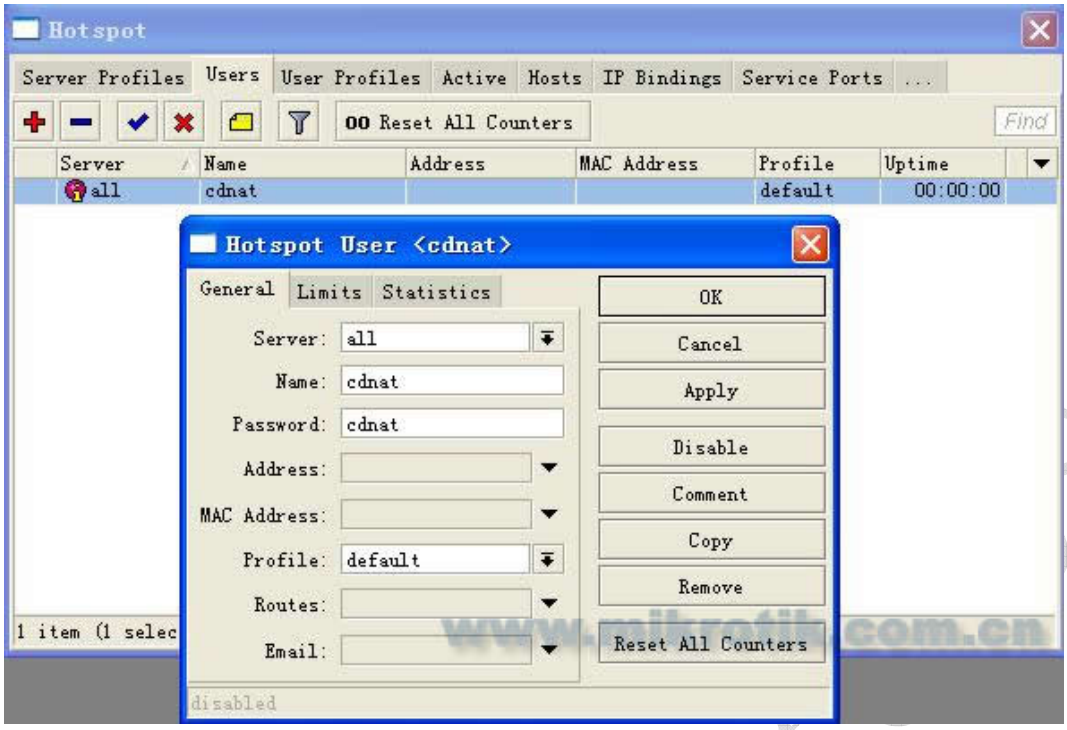


在 user profile 里面一般配置如下几个参数：

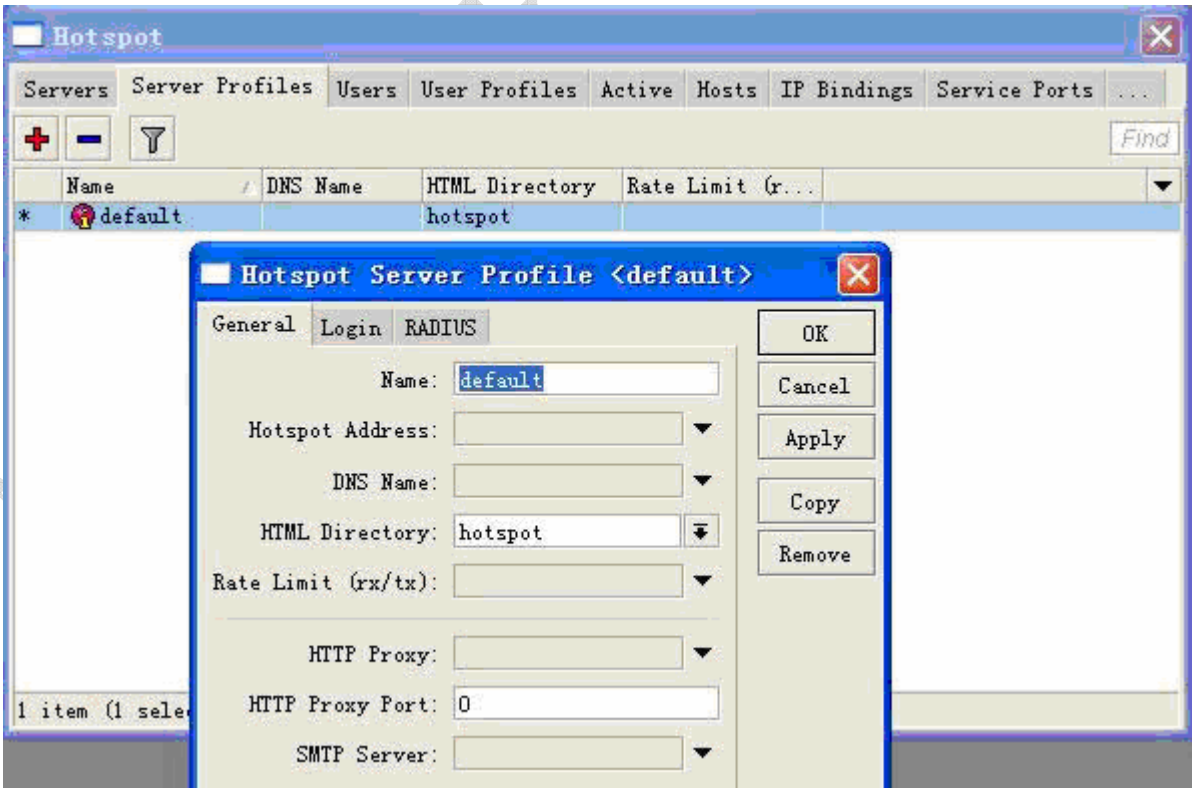
Idle-Timeout：用户在一定时间内没有任何流量发出后自动注销连接
Keepalive-Timeout：路由器主动通过 ICMP 探测主机是否在线，如果在一定时间为探测到自动注销连接（如果用户机开启防火墙，路由器无法探测到）
Shared-users：帐号的分享用户多少，默认为 1，即仅一个用户使用该帐号。
Rate-Limit：分配每个帐号带宽，格式为“上行 / 下行”
Transparent-proxy：透明代理功能是否开启，一般使用 Hotspot 认证建议不用打开此参数。
其他参数请参考具体 Hotspot 手册。

Address pool 这个是 DHCP 的地址池，给用户分配 IP，我们可以在 ip pool 中分配地址段，具体操作请参考 RouterOS 的 DHCP 操作。

在 user 配置用户登录帐号和密码，以及所属的 profile 类型：

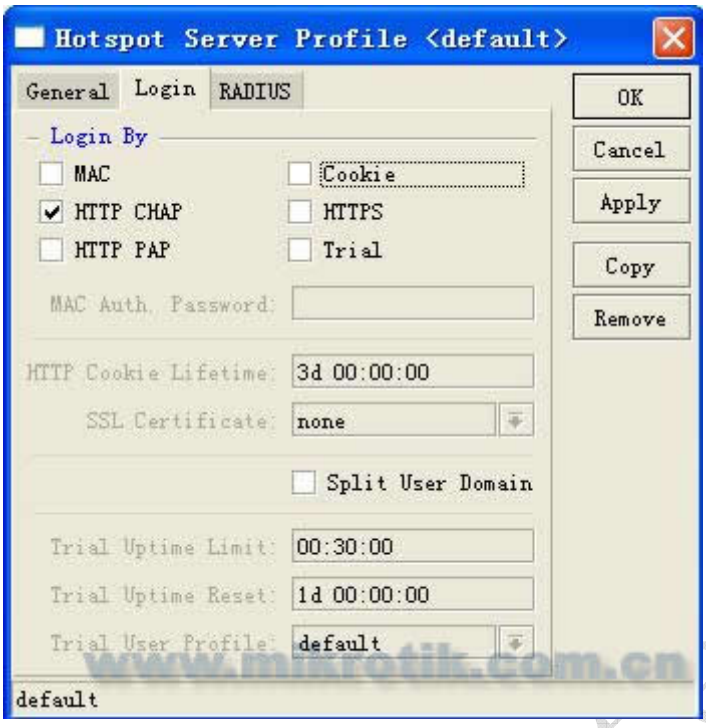


这里默认 server 服务器为 all，
Name 用户名：cdnat
Password：cdnat
Profile：用户组规则，这里选择我们之前设置的 default 规则
配置完用户规则后，进入 ip hotspot server profile，配置服务器器规则。



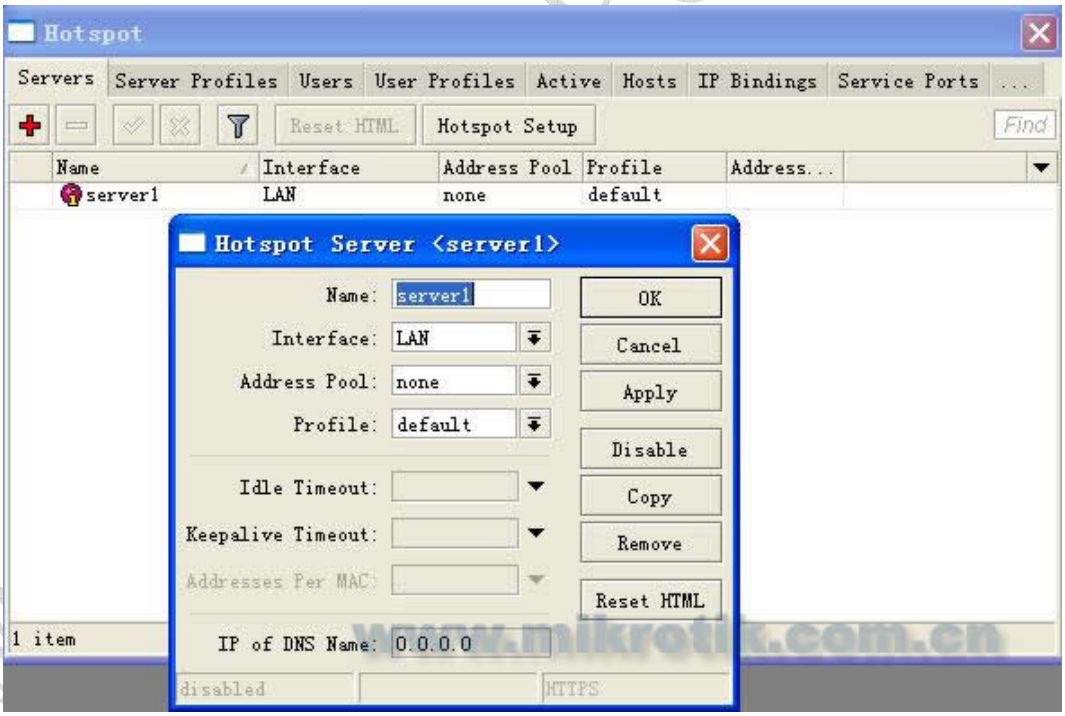
在 General 选项中选择 HTML Directory 为默认的 hotspot 文件路径，同时也可以选择自己定义的文件名路径。

配置 login 登录方式，一般只启用 http chap 即可，其他选项根据需要开启。



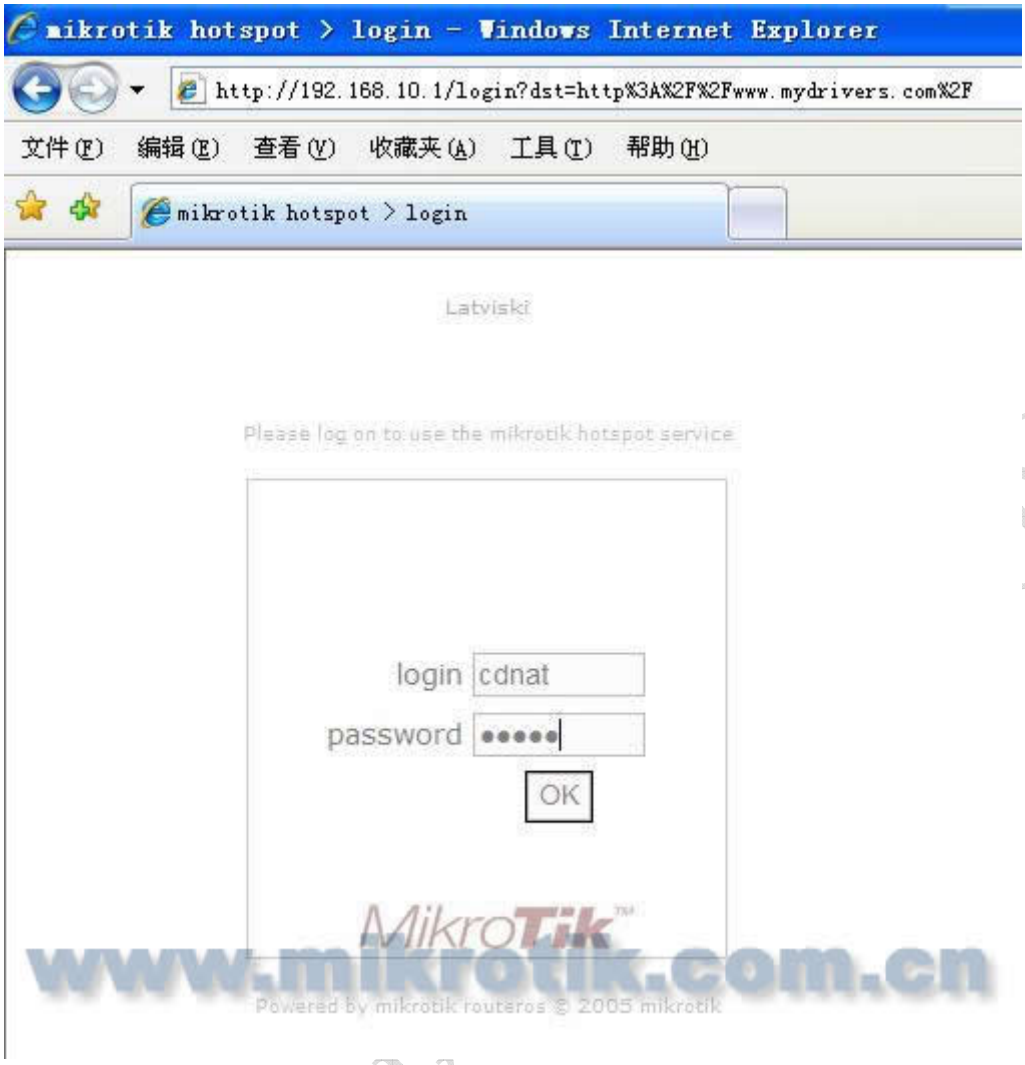
至于 Radius 根据需要开启。

配置完成以上参数后，最后我们启用 Hotspot 服务器：



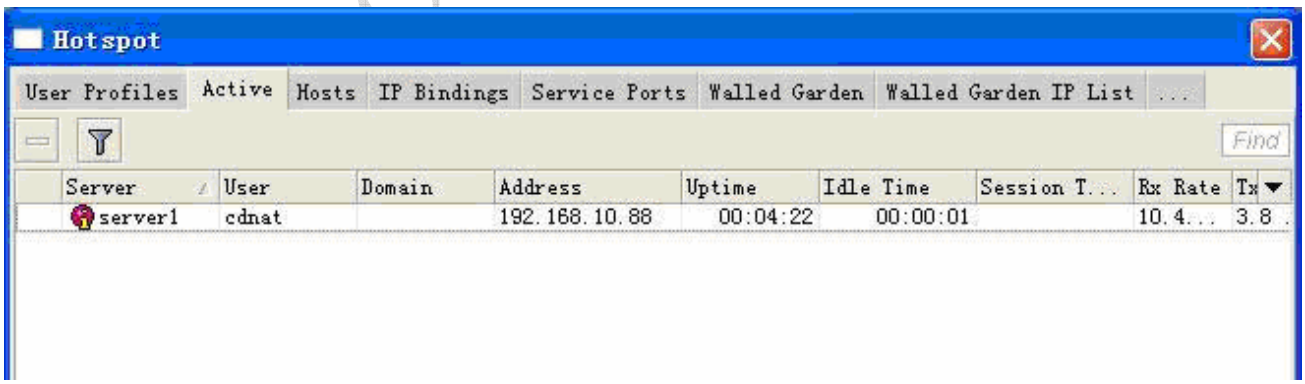
当启用完成后，所有对路由器或者外网访问都需要通过 web 认证，在用户没有认证的情况下，当用户随便输入一个网站都会跳转到认证页面。

如当输入 www.mydrivers.com 的网站，Hotspot 会强制用户的 web 页面跳转到认证页，如图：



用户输入帐号 cdnat 和密码 cdnat 后，点 ok 按钮即可通过认证，当认证通过后，页面自动跳转到 www.mydrivers.com 的网站。

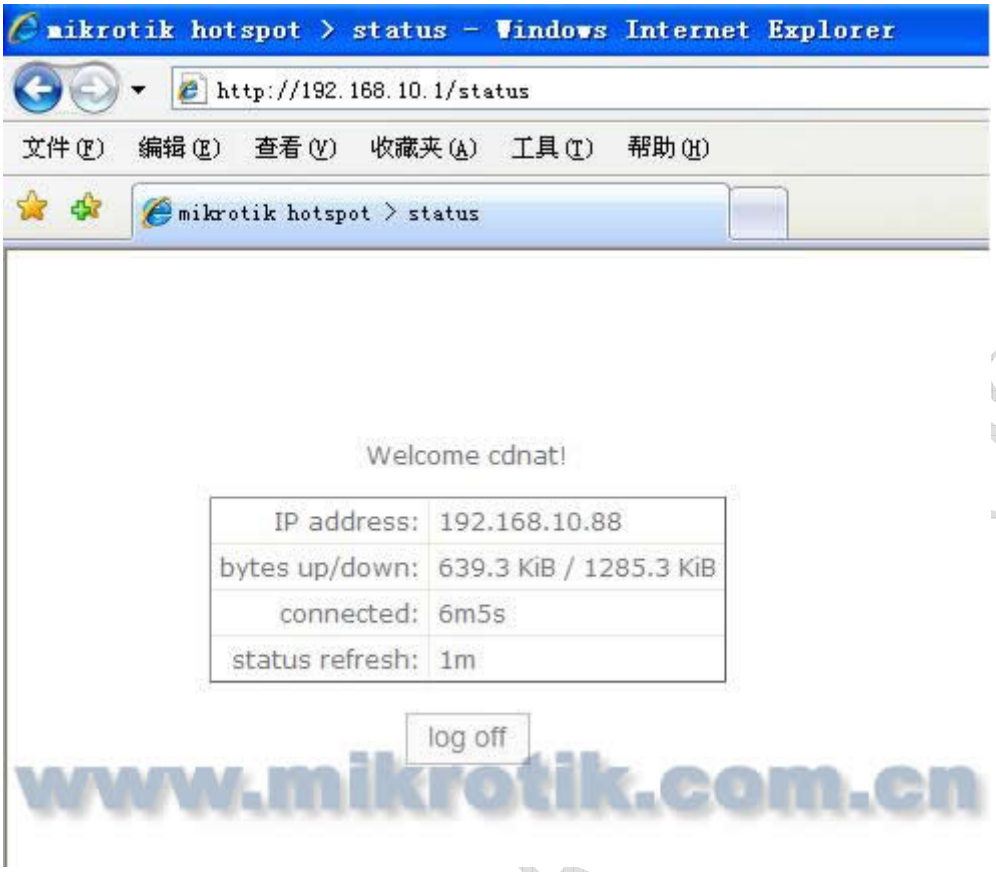
这时我们可以在 ip hotspot active 中看到用户登录的在线情况：



获取现时用户列表：

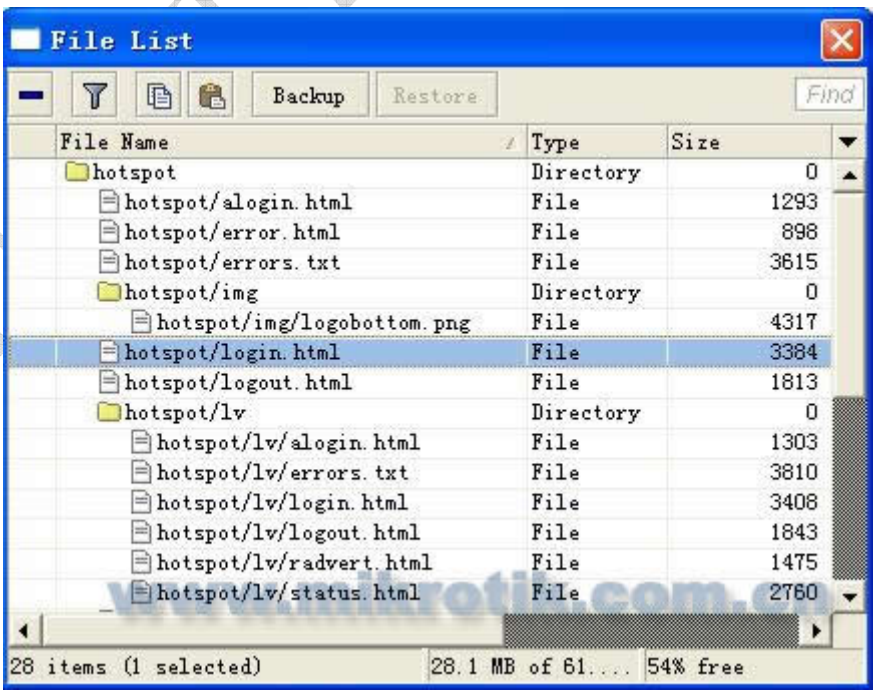
```
[admin@MikroTik] ip hotspot active> print
Flags: R - radius, B - blocked
 #    USER           ADDRESS         UPTIME      SESSION-TIMEOUT IDLE-TIMEOUT
 0    cdnat       192.168.10.88      4m17s       55m43s
[admin@MikroTik] ip hotspot active>
```

用户如果需要注销，通过输入 192.168.10.1 Hotspot 认证网关，点击 log off 退出登录页面



认证页面

Hotspot 的认证登录页面是开放式的，即可以通过 RouterOS 的 files 目录下找到这些文件，Hotspot 在 files 中的默认文件名“Hotspot”，



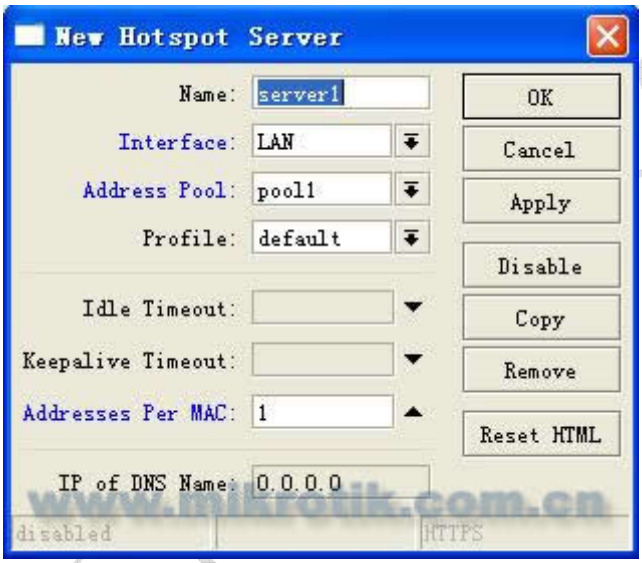
认证页面我们可以通过修改 login.html、logout.html 和 status.html 的 web 界面得到你想要的网页画面或者 log。

17. 10 Hotspot 即插即用功能

从 2.7 的版本就开始支持 upnp 的即插即用功能，即当用户和 Hotspot 认证服务器在同一局域网内，不管局域网用户设置任何的 IP 地址（前提是用户必须设置任意的 IP 地址、网关和 DNS）都可以被 Hotspot 认证服务器获取，并在 Hotspot 的 Host 中分配一个新的虚拟 IP 地址，并对用户作一对一的 NAT 转换。Hotspot 的即插即用方式分成适用于：流动性较强的公共场所，如机场、车站、公园，也可以应用到酒店和小区中。

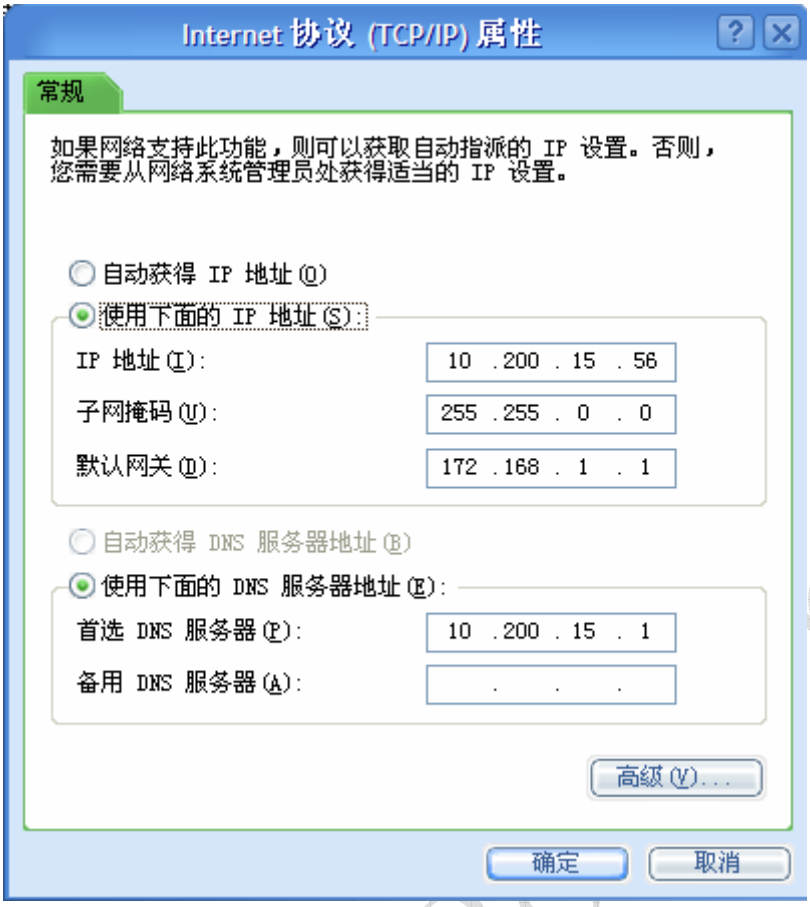
Hotspot 服务器会在同一局域网内发送 ARP 广播，告诉局域网内的所有主机自己的网关设备，并为在线的主机分配一个虚拟的 IP 地址，这样客户主机在没有配置正确的 IP 地址情况下也能连接到 Hotspot 网关服务器，并认证上网。

在 2.9 和 3.0 的 Hotspot 启用 server 服务后，即插即用功能默认是打开的，但配置 Hotstop 需要在 hotspot server 中将 address pool 的地址池设置好，如图：

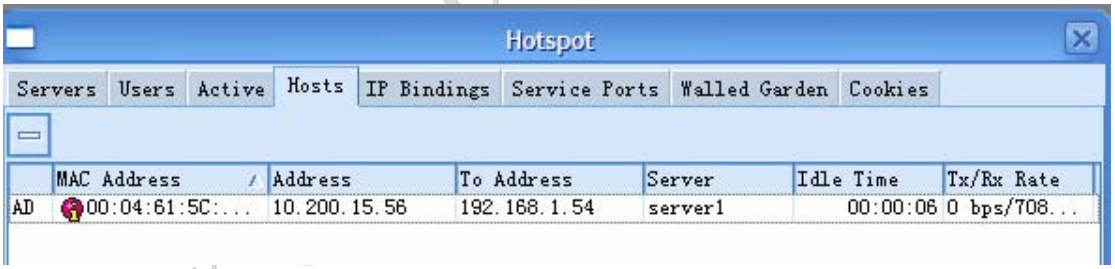


Addresses Per MAC 这个是每个 IP 对应的 MAC 地址，这里我们设置为 1，即一个 IP 对应一个 MAC 地址。

我们 windows 电脑的 IP 地址配置如下



在 Hotspot 的 host 列表中，我们可以看到，在同一局域网内的 windows 主机被 Hotspot 捕获后，自动为其分配 IP 地址，并做了对应关系



注： 如果 Hotspot 没有工作，可能的情况如下：

- 检查**/ip dns** 包含的合法 **DNS** 服务器，在命令或者 tools ping 中是否能解析**/ping www.mikrotik.com.cn** ，并确认 DNS 的缓存功能打开
- 确保连接追踪已经启用：**/ip firewall connection tracking set enabled=yes**

17.11 HotSpot 防火墙部分

除了在**/ip hotspot** 子目录本身的明显的动态规则（像主机及动态用户），一些附加的规则会在激活一个 HotSpot 服务时被添加到防火墙表中。不像 RouterOS 2.8 版本，只有相对较少的防火墙规则添加在防火墙中，因为主要的工作是有一对一 nat 算法完成的。

nat 规则

从**/ip firewall nat print dynamic** 命令你可以获取如下（在每条规则后跟有评注）：

```
0 D chain=dstnat hotspot=from-client action=jump jump-target=hotspot
```

把对数据包的所有 HotSpot 相关任务从 HotSpot 客户放到一个单独的链中：

```
1 D chain=hotspot protocol=udp dst-port=53 action=redirect to-ports=64872
2 D chain=hotspot protocol=tcp dst-port=53 action=redirect to-ports=64872
```

把所有 DNS 请求都重定向到 HotSpot 服务。64872 端口对所有 HotSpot 用户提供 DNS 服务。如果你想要 HotSpot 服务器也监听其他端口，在这里以同样方式添加规则，改变 **dst-port** 属性。

```
3 D chain=hotspot protocol=tcp dst-port=80 hotspot=local-dst action=redirect
   to-ports=64873
```

把所有 HTTP 登陆请求定向到 HTTP 登陆 servlet。64873 就是 HotSpot HTTP servlet 端口。

```
4 D chain=hotspot protocol=tcp dst-port=443 hotspot=local-dst action=redirect
   to-ports=64875
```

把所有 HTTPS 登陆请求定向到 HTTPS 登陆 servlet。64875 是 HotSpot HTTPS servlet 端口。

```
5 D chain=hotspot protocol=tcp action=jump hotspot=!auth jump-target=hs-unauth
```

所有其他的数据包除了 DNS 及来自未认证客户的登陆请求以外都应该通过 **hs-unauth** 链。

```
6 D chain=hotspot protocol=tcp action=jump hotspot=auth jump-target=hs-auth
```

来自认证用户的数据包通过 **hs-auth** 链

```
7 D ;;; www.mikrotik.com
    chain=hs-unauth dst-address=159.148.147.196 protocol=tcp dst-port=80
    action=return
```

首先在 **hs-unauth** 链中把所有影响 TCP 协议的东西都放到**/ip hotspot walled-garden ip** 子目录中。现在我们把 www.mikrotik.com 从重定向到登陆页面中排除。

```
8 D chain=hs-unauth protocol=tcp dst-port=80 action=redirect to-ports=64874
```

所有其他 HTTP 请求都被定向到监听 64874 的 Walled Garden 代理服务器。如果在**/ip hotspot walled-garden** 子目录有一个 HTTP 请求的 **allow** 条目，它将被转发到目的。否则，请求将会自动被重定向到 HotSpot 登陆 servlet（端口 64873）。

```
 9 D chain=hs-unauth protocol=tcp dst-port=3128 action=redirect to-ports=64874
10 D chain=hs-unauth protocol=tcp dst-port=8080 action=redirect to-ports=64874
```

默认设置的 HotSpot 假设只有这些端口才能用于 HTTP 代理请求。这两个条目用于“捕捉”客户到未知代理的请求。如：使的有可能让带有未知代理设置的客户与 HotSpot 系统能够一起工作。这个特性叫做“通用代理”。如果探测到一个客户正在使用某个代理服务器，系统将自动以 **http** hotspot 标志对数据包进行标记以便处理未知代理问题。注意已使用的端口（64874）与#8 规则中对 HTTP 请求的一样（所以 HTTP 和 HTTP 代理请求都由相同的代码处理）。

```
11 D chain=hs-unauth protocol=tcp dst-port=443 action=redirect to-ports=64875
```

HTTPS 代理监听 64875 端口

```
12 D chain=hs-unauth protocol=tcp dst-port=25 action=jump jump-target=hs-smtp
```

对 SMTP 协议的重定向也可以在 HotSpot 配置中定义。如果是这样，那么一个重定向规则将被放在 **hs-smtp** 链中。这个完成后以便带有未知 SMTP 配置的用户能通过服务提供商（你们的）的 SMTP 服务器发送邮件，而代替了用户在自己电脑配置的 SMTP 服务器。

```
13 D chain=hs-auth protocol=tcp hotspot=http action=redirect to-ports=64874
```

对认证用户提供 HTTP 代理服务。认证用户的请求可能需要透明的代理（“通用代理”技术以及广告特征）。**http** 标志会自动的放在被 HotSpot HTTP 代理探测到的服务器的 HTTP 代理请求（监听 64874 端口的）。这个完成后以便有代理设置的用户可以使用 HotSpot 网关代替用户在自己电脑上配置的代理服务器。这个标志也会被放在任何概要被配置为透明代理的用户所做的 HTTP 请求上。

```
14 D chain=hs-auth protocol=tcp dst-port=25 action=jump jump-target=hs-smtp
```

对授权用户提供 SMTP 代理（同#12 规则的一样）

包过滤规则

从**/ip firewall filter print dynamic** 命令，你可以获得：

```
 0 D chain=forward hotspot=from-client,!auth action=jump jump-target=hs-unauth
```

任何来自未认证且通过路由器的数据包都将被发送到 **hs-unauth** 链。**hs-unauth** 执行基于 IP 的 Walled Garden 过滤器。

```
 1 D chain=forward hotspot=to-client,!auth action=jump jump-target=hs-unauth-to
```

任何通过路由器到达客户的包都将被重定向到另一个叫做 **hs-unauth-to** 的链。这个链会拒绝到达客户的未认证请求。

```
2 D chain=input hotspot=from-client action=jump jump-target=hs-input
```

任何从客户到达路由器本身的包将重定向到另一个叫 **hs-input** 的链。

```
3 D chain=hs-input protocol=udp dst-port=64872 action=accept
4 D chain=hs-input protocol=tcp dst-port=64872-64875 action=accept
```

允许客户访问本地认证和代理服务。

```
5 D chain=hs-input hotspot=!auth action=jump jump-target=hs-unauth
```

所有其他来自未认证客户到路由器本身的数据流都将会与通过路由器的数据流一样的方式被处理。

```
6 D chain=hs-unauth protocol=icmp action=return
7 D ;;; www.mikrotik.com
    chain=hs-unauth dst-address=159.148.147.196 protocol=tcp dst-port=80
    action=return
```

不仅在 TCP 协议相关的 Walled Garden 条目被添加的 NAT 列表中，在包过滤器中 **hs-unauth** 链表也会添加在**/ip hotspot walled-garden ip** 目录中设置的东西。这就是为什么，尽管你只在 NAT 表中添加了一个条目却有两条规则的原因。

```
8 D chain=hs-unauth protocol=tcp action=reject reject-with=tcp-reset
9 D chain=hs-unauth action=reject reject-with=icmp-net-prohibited
```

任何没有被 Walled Garden 记录在表格上的都将被拒绝。注意拒绝 TCP 连接的 TCP 重启的使用。

```
10 D chain=hs-unauth-to action=reject reject-with=icmp-host-prohibited
```

用 ICMP 拒绝信息拒绝所有到达客户的包。

第十八章 PPPoE 配置

PPPoE 基于以太网的点对点协议(Point to Point Protocol over Ethernet)当前的 PPPOE 主要被 ISP 商用于 xDSL 和 cable modems 与用户端的连接，他们几乎与以太网一样。 PPPoE 是一种标准的点对点协议(PPP) 他们之间只是传输上的差异：PPPoE 使用 modem 连接来代替普通的以太网。一般来说, PPPoE 是基于与用户认证和通过分发 IP 地址给客户端。

RouterOS 能做一个的 RADIUS 客户端 – 你能使用一台 RADIUS 服务器去验证 PPPoE 的客户端和对他们计费

一个 PPPoE 连接由客户端和一个访问集线服务器组成，客户端可以是一个安装了 PPPoE 协议的 windows 电脑。 PPPoE 客户端和服务器能工作在任何以太网等级的路由器接口（interface） - wireless 802.11 (Aironet, Cisco, WaveLan, Prism, Atheros), 10/100/1000 Mbit/s Ethernet, RadioLan 和 EoIP (Ethernet over IP tunnel)都支持。

支持的连接

- MikroTik RouterOS PPPoE 客户端到任何 PPPoE 服务器(access concentrator)
- MikroTik RouterOS PPPoE 服务器(access concentrator)到多个 PPPoE 客户端 (客户端包括几乎所有的操作系统和大部分路由器)

多连接 PPP 协议支持 MP，提供 MRRU 协议（能够传输 1500 和大数据包）和基于 PPP 连接的桥接 bridging（使用桥接控制协议 BCP，能发送基于 PPP 连接的原始以太网帧）这样能在没有 EoIP 协议的支持下，设置桥接。

*注：*当 RADIUS 服务器验证一个用户 CHAP、MS-CHAPv1 或 MS-CHAPv2，RADIUS 戏院不会使用共享密码(shared secret)，仅验证回复(authentication reply)被使用。因此如果你有一个错误的共享密码，RADIUS 服务器将接受请求。你可以使用**/radius monitor** 命令查看 **bad-replies** 参数，无论什么时候客户在试图连接时这个值都会增加。

规格

需要功能包: ***ppp***
需要等级： *Level1 (限制 1 个连接)*，*Level3 (限制 200 个连接)*，*Level4 (限制 200 个连接)*，*Level5 (限制 500 个连接)*，*Level6 (无限制)*
操作路径： ***/interface pppoe-server***, ***/interface pppoe-client***
协议标准和技术： PPPoE (RFC 2516)
硬件要求: *PPPoE 服务器的需要增加 RAM 和提高 CPU 性能，每个连接使用 9KiB(如果限流被使用额外还需要增加 10KiB)，*

18.1 PPPoE Client 设置

操作路径: ***/interface pppoe-client***

属性描述

name (*名称*; 默认: **pppoe-out1**) –PPPoE 的接口名称
interface (*名称*) – 选择 PPPoE 服务器的接口使连接通过
mtu (*整型*; 默认: **1480**) – 最大传输单位..。最适合的 MTU 值（以避免以太网连接的 1500-byte，设置为 1480 以避免数据包的重复存储）
mru (*整型*; 默认: **1480**) – 最大接收单位。最适合的 MRU 值（以避免以太网连接的 1500-byte，设置为 1480 以避免数据包的重复存储）
user (*文本*; 默认: **""**) – 连接在 PPPoE 服务器的用户帐号。
password (*文本*; 默认: **""**) – 连接 PPPoE 服务器的用户密码
profile (名称) – 连接的默认策略
allow (*多选项:* mschap2, mschap1, chap, pap; default: **mschap2, mschap1, chap, pap**) – 客户端使用的验证协议
service-name (*文本*; 默认: **""**) – 在访问集线器上设定指定服务名（AC）
ac-name (*文本*; 默认: **""**) – 这条可以为空白，当客户端与任何一个访问集线器相连，会选取该服务名。
add-default-route (yes | no; 默认: **no**) – 是否添加动态默认路由。

dial-on-demand (yes | no; 默认: **no**) – 当连接唯一的 AC 时，传输数据产生，在断开连接，没有传输数据时，idle-timeout 将被设置。
use-peer-dns (yes | no; 默认: **no**) – 是否使用路由器的默认 DNS 给 ppp 的 DNS

注： 如果存在一条默认的 pppoe 的路由，**add-default-route** 将不会创建一个新的路由

在 **gig** 接口上添加和启用客户端客户端，连接 AC 提供的 **testSN** 服务名，使用的用户帐号 **john** 和密码 **password** ：

```
[admin@RemoteOffice] interface pppoe-client> add interface=gig \
\... service-name=testSN user=john password=password disabled=no
[admin@RemoteOffice] interface pppoe-client> print
Flags: X - disabled, R - running
 0  R name="pppoe-out1" mtu=1480 mru=1480 interface=gig user="john"
     password="password" profile=default service-name="testSN" ac-name=""
     add-default-route=no dial-on-demand=no use-peer-dns=no
```

监视 PPPoE 客户端

命令名称: ***/interface pppoe-client monitor***

属性描述

status (文本) – 客户端的状态
  Dialing – 拨号连接的情况
  Verifying password... – 确认连接到服务器，密码正在核对处理
  Terminated – 接口没有启用，或是另一端未建立连接
encoding (*文本*) – 在该条连接中使用加密和编码。
uptime (*时间*) – 连接时间显示为天、时、分、秒
service-name (*文本*) – 客户端连接的服务器名称
ac-name (*文本*) – 客户端已经连接到的 AC 名称
ac-mac (*MAC 地址*) – 客户端已经连接的访问集线器（AC）MAC 地址。

监视 **pppoe-out1** 连接情况：

```
[admin@MikroTik] interface pppoe-client> monitor pppoe-out1
        status: "connected"
        uptime: 10s
      encoding: "none"
  service-name: "testSN"
       ac-name: "10.0.0.1"
        ac-mac: 00:C0:DF:07:5E:E6

[admin@MikroTik] interface pppoe-client>
```

18.2 ADSL 拨号上网事例

```
ADSL 用户名: user@169
      密码: 1234
Service Name: CHN-Telecom
```

1：添加 PPPOE Clients

```
[admin@Router] interface pppoe-client>
[admin@Router] interface pppoe-client> add interface=ether1 mtu=1492 mru=1492
service-name=CHN-Telecom user= user@169 password=1234
add-default-route=yesuse-peer-dns=yes
[admin@ROUTER] interface pppoe-client> print
Flags: X - disabled, R - running
0  X name="pppoe-out1" mtu=1492 mru=1492 interface=ether1 user=user@169
    password=1234 profile=default service-name=CHN-Telecom ac-name=""
    add-default-route=yes dial-on-demand=no use-peer-dns=yes
```

PPPOE 拨号已经配置好，接下来将 ADSL MODEM 的网线连接好进行以下操作就可以连同了。

```
[admin@Router] interface pppoe-client>enable 0
[admin@Router] interface pppoe-client> monitor pppoe-out1
        status: "connected"
        uptime: 10s
      encoding: "none"
   service-name: "CHN-Telecom"
       ac-name: ""
        ac-mac: 00:C0:DF:07:5E:E6
```

之后还需在 ip firewall mangle 中添加一条规则：

```
[admin@Router] ip firewall mangle> add chain=forward protocol=tcp tcp-flags=syn
action=change-mss new-mss=1440
[admin@Router] ip firewall mangle> print
Flags: X - disabled, I – invalid
0   chain=forward protocol=tcp tcp-flags=syn action=change-mss
    new-mss=1440
```

最后如果你要起用 nat 功能，不要忘了在 ip firewall nat 设置 IP 伪装。

18.3 PPPoE Server 设置

操作路径: ***/interface pppoe-server server***

PPPoE server (access concentrator)支持在每一个接口上的多服务，需要设置不同的 service 名称 ，当前 PPPoE server 的吞吐量在一个 Celeron 600 CPU 测试达到 160 Mb/s ， 如果使用更高性能的 CPU，吞吐量将会程比例的增加。

service-name (*文本*) – PPPoE 服务名称

mtu (*整型*; 默认: **1480**) – 最大传输单位。最适合的 MTU 值（以避免以太网连接的 1500-byte，设置为 1480 以避免数据包的重复存储）

mru (*整型*; default: **1480**) – 最大接受单位。最适合的 MRU 值（以避免以太网连接的 1500-byte，设置为 1480 以避免数据包的重复存储）

mrru (*整型*: 512..65535; 默认: **disabled**) – 在连接中能被接收的最大数据包长度。如一个数据包比隧道的 MTU 值大时，将会被分割到多个数据包中，允许实际大小的 IP 或以太网的数据包发送到隧道。

authentication (多种选择：mschap2 | mschap1 | chap | pap; 默认: **mschap2, mschap1, chap, pap**) – 验证算法

keepalive-timeout – 定义时间周期(秒) 连接开始后路由器每秒钟会发出 keepalive 数据包。如果在设定的时间周期内没有传输和没有 keepalive 回应，客户端将会被认为失去连接。

one-session-per-host (yes | no; 默认: **no**) – 每次只允许一个主机对话连接 (MAC 地址被确定). 如果主机将试着去建立一个新的对话连接，旧的一个将会被关闭。

default-profile (*名称*; 默认: **default**) – 使用默认的策略配置

max-sessions (*整形*; 默认: **0**) – AC 能服务的最大客户端数量

0 – 没有限制

interface (*名称*) – 客户端连接的网卡接口

注： **keepalive-timeout** 值通常情况下设置为 **10**。如果你设置为 **0**，路由器将不会断开客户端，直到他们自己注销或是路由器重启该用户帐号才会断开。解决这个问题，**one-session-per-host** 属性需启用

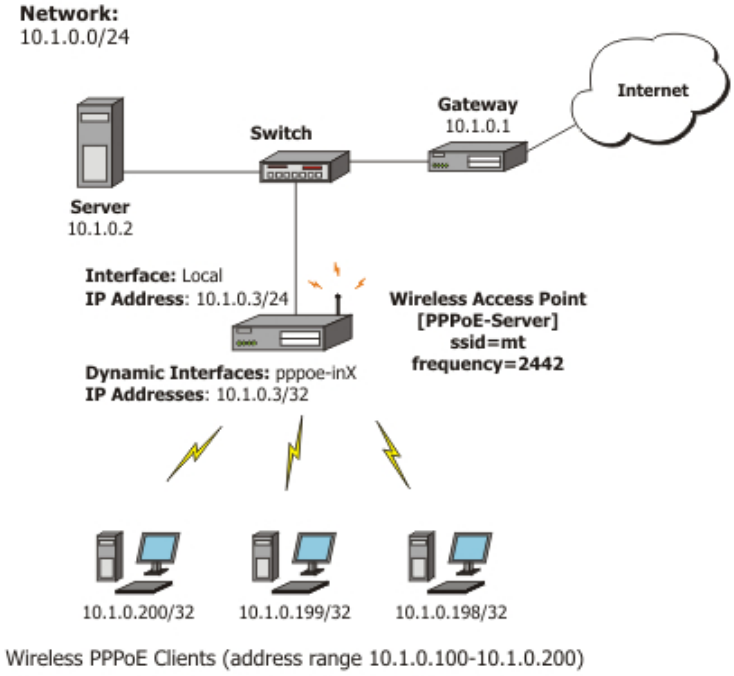
安全提示：请不要分配一个 IP 地址到 PPPoE 的物理网卡上，避免出现用户不通过 PPPoE 验证即可上网的情况。

明确的讲 MRRU 意思为基于单连接的 MP，该协议被为拆分大数据包为更小的。在 windows 下在网络属性下，设置按钮中打开“为单链路连接协商多重链接”MRRU 是强行设置为 1614。这个设置有益于超载线路 MTU 探测失败。且 MP 协议应在双方都被启用。

基于 802.11g 无线网络的 PPPoE 服务

在无线网络中，服务器可以设置在一个访问节点（Access Point）,任意一个 RouterOS 客户端或是 Windows 客户端都可以连接到访问节点的 PPPoE 认证。无线网卡的 MTU 可以设置为 1600，因此接口上的 MTU 设置为 1500，这可以充分利于 1500byte 传输数据包，并避免 MTU 比 1500 低出现的任何问题。

让我们考虑下面的配置，MikroTik 无线 AP 能使无线用户端通过验证后访问到本地的网络:



首先，需要配置无线网卡：

```
[admin@PPPoE-Server] interface wireless> set 0 mode=ap-bridge \
   frequency=2442 band=2.4ghz-b/g ssid=mt disabled=no
[admin@PPPoE-Server] interface wireless> print
Flags: X - disabled, R - running
 0    name="wlan1" mtu=1500 mac-address=00:01:24:70:53:04 arp=enabled
      disable-running-check=no interface-type=Atheros AR5211
      radio-name="000124705304" mode=station ssid="mt" area=""
      frequency-mode=superchannel country=no_country_set antenna-gain=0
      frequency=2412 band=2.4ghz-b scan-list=default rate-set=default
      supported-rates-b=1Mbps,2Mbps,5.5Mbps,11Mbps
      supported-rates-a/g=6Mbps,9Mbps,12Mbps,18Mbps,24Mbps,36Mbps,48Mbps,
                          54Mbps
      basic-rates-b=1Mbps basic-rates-a/g=6Mbps max-station-count=2007
      ack-timeout=dynamic tx-power=default tx-power-mode=default
      noise-floor-threshold=default periodic-calibration=default
      burst-time=disabled fast-frames=no dfs-mode=none antenna-mode=ant-a
      wds-mode=disabled wds-default-bridge=none wds-ignore-ssid=no
      update-stats-interval=disabled default-authentication=yes
      default-forwarding=yes default-ap-tx-limit=0 default-client-tx-limit=0
      hide-ssid=no security-profile=default disconnect-timeout=3s
      on-fail-retry-time=100ms preamble-mode=both
```

```
[admin@PPPoE-Server] interface wireless>
```

现在，配置以太网卡，添加默认 IP 地址和设置默认路由：

```
[admin@PPPoE-Server] ip address> add address=10.1.0.3/24 interface=Local
[admin@PPPoE-Server] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS          NETWORK        BROADCAST      INTERFACE
 0   10.1.0.3/24       10.1.0.0       10.1.0.255     Local
[admin@PPPoE-Server] ip address> /ip route
[admin@PPPoE-Server] ip route> add gateway=10.1.0.1
[admin@PPPoE-Server] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf
 #    DST-ADDRESS       G GATEWAY        DISTANCE INTERFACE
 0 ADC 10.1.0.0/24                             Local
 1 A S 0.0.0.0/0         r 10.1.0.1       1       Local
[admin@PPPoE-Server] ip route> /interface ethernet
[admin@PPPoE-Server] interface ethernet> set Local arp=proxy-arp
[admin@PPPoE-Server] interface ethernet> print
Flags: X - disabled, R - running
 #    NAME                              MTU   MAC-ADDRESS      ARP
 0  R Local                              1500  00:0C:42:03:25:53 proxy-arp
[admin@PPPoE-Server] interface ethernet>
```

添加 PPPoE server 到无线网卡上:

```
[admin@PPPoE-Server] interface pppoe-server server> add interface=wlan1 \
  service-name=mt one-session-per-host=yes disabled=no
[admin@PPPoE-Server] interface pppoe-server server> print
Flags: X - disabled
 0   service-name="mt" interface=wlan1 max-mtu=1480 max-mru=1480
     authentication=pap,chap,mschap1,mschap2 keepalive-timeout=10
     one-session-per-host=yes max-sessions=0 default-profile=default
[admin@PPPoE-Server] interface pppoe-server server>
```

最后，设置 PPPoE clients:

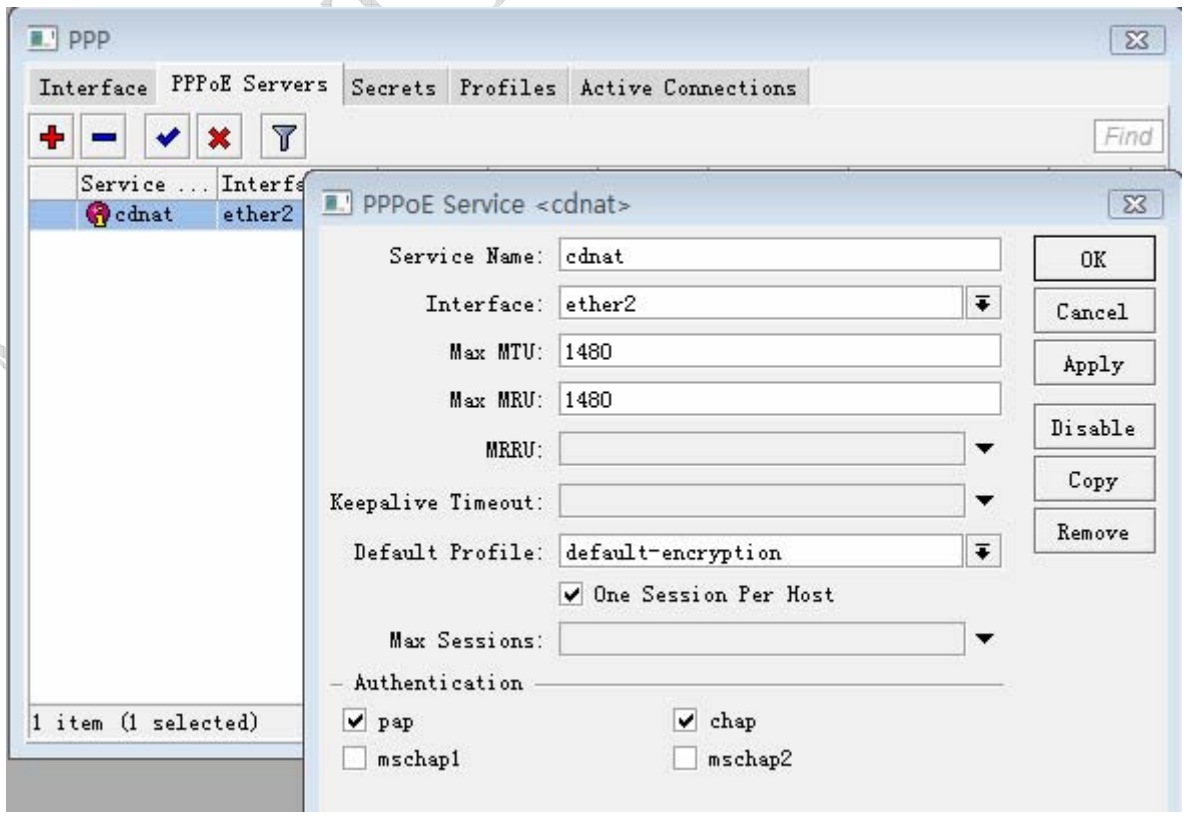
```
[admin@PPPoE-Server] ip pool> add name=pppoe ranges=10.1.0.100-10.1.0.200
[admin@PPPoE-Server] ip pool> print
 # NAME                                 RANGES
 0 pppoe                                10.1.0.100-10.1.0.200
[admin@PPPoE-Server] ip pool> /ppp profile
[admin@PPPoE-Server] ppp profile> set default use-encryption=yes \
  local-address=10.1.0.3 remote-address=pppoe
[admin@PPPoE-Server] ppp profile> print
```

```
Flags: * - default
 0 * name="default" local-address=10.1.0.3 remote-address=pppoe
     use-compression=no use-vj-compression=no use-encryption=yes only-one=no
     change-tcp-mss=yes

 1 * name="default-encryption" use-compression=default
     use-vj-compression=default use-encryption=yes only-one=default
     change-tcp-mss=default
[admin@PPPoE-Server] ppp profile> .. secret
[admin@PPPoE-Server] ppp secret> add name=w password=wkst service=pppoe
[admin@PPPoE-Server] ppp secret> add name=l password=ltp service=pppoe
[admin@PPPoE-Server] ppp secret> print
Flags: X - disabled
 #   NAME       SERVICE CALLER-ID PASSWORD  PROFILE          REMOTE-ADDRESS
 0   w          pppoe           wkst      default          0.0.0.0
 1   l          pppoe           ltp       default          0.0.0.0
[admin@PPPoE-Server] ppp secret>
```

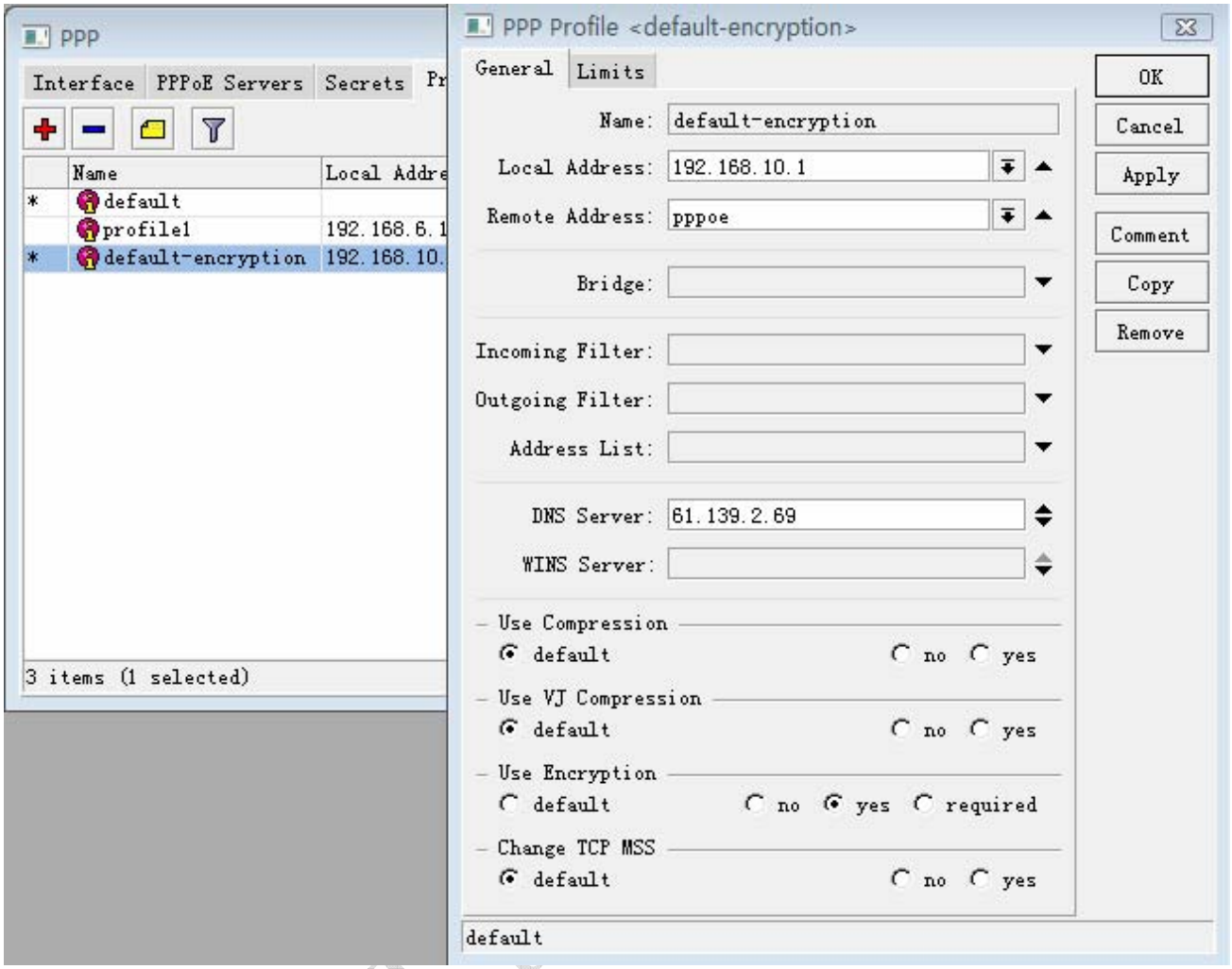
注: 在 Windows XP 中的 PPoE 客户端内建加密功能，但 RASPPPOE 没有。因此，如果计划不在支持比 Windows XP 老的 Windows 客户端，推荐在 **default** 规则配置把 **require-encryption** 值选择位 **yes**。在其他一些应用中，可以服务设置为接受为加密的数据。

18.4 Winbox 配置 PPPoE 服务

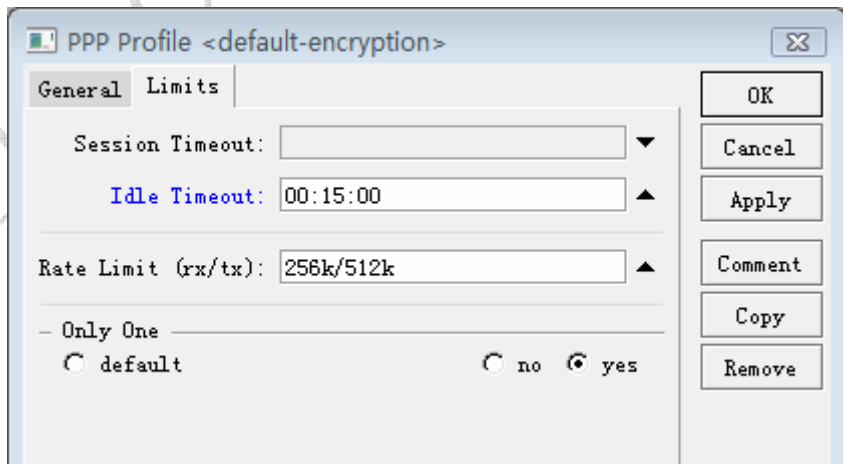
通过 Winbox 配置 PPPoE 服务器，这里我们首先通过进入 PPP 目录下的 PPPoE Server，配置 Service Name 为 cdnat，用于 PPPoE 服务器名，并把 PPPoE 服务指向 ether2 的网卡上，其他参数如图所示：



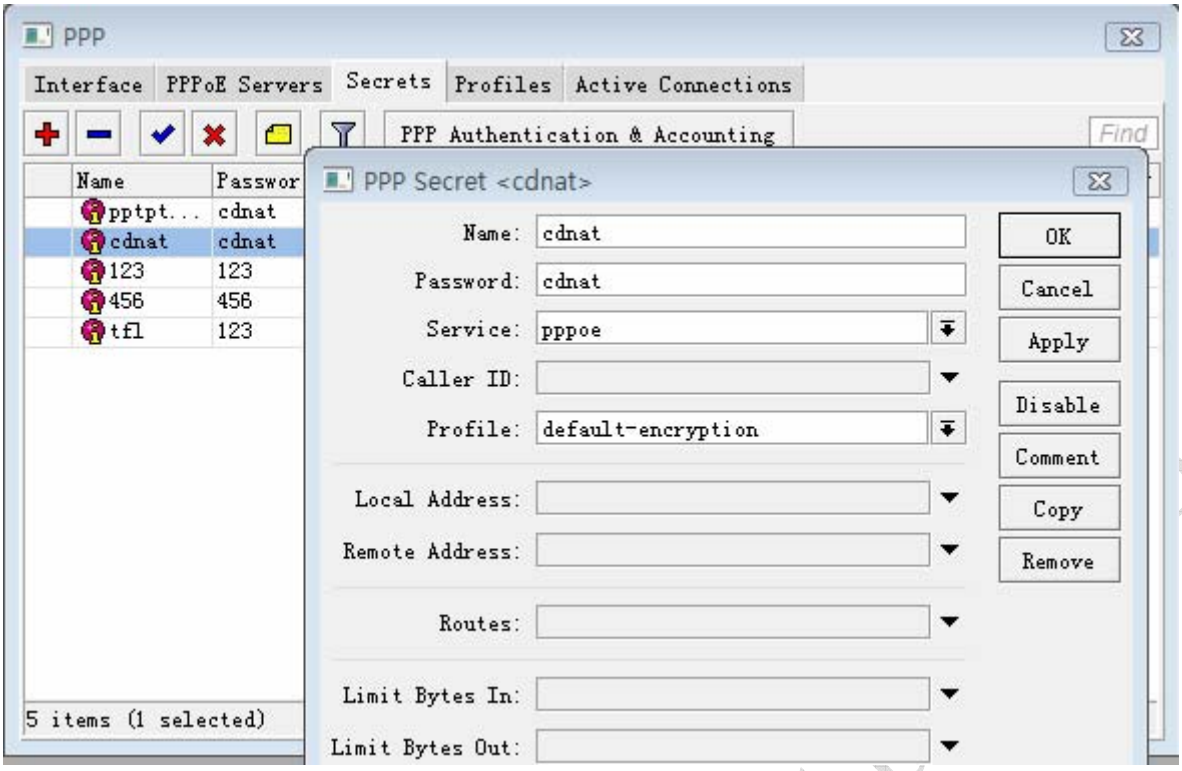
这里我们选择的是 default-encryption 的 profile 规则，所有我们需要进入 profiles 中配置该规则的参数，loacl-address 为本地路由器网关 IP，remote-address 则是远程客户端 IP 地址。这里我们设置 local-address 为 192.168.10.1，remote-address 添加在 ip pool 中设置好的地址池 pppoe，然后配置 DNS 参数，其他配置如图：



下面配置 Limits 参数：



这样用户的组规则配置完成，根据需要也可以增加其他的组规则到 profile 中。接下来配置每个用户信息，进入 ppp secrets 添加用户帐号：

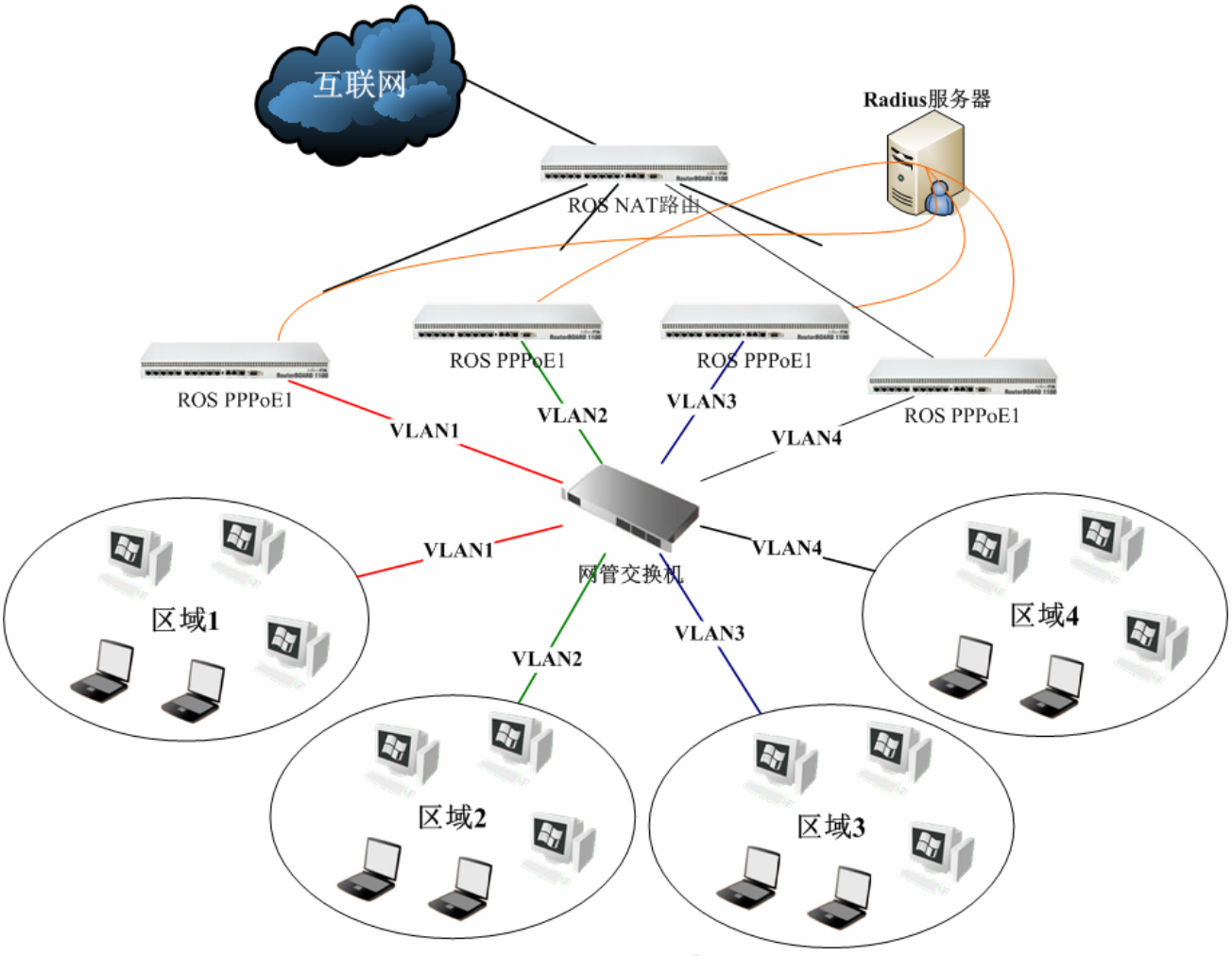


这里 Name 为用户帐号名，Password 为用户密码。Profile 选择刚才设置好的 default-encryption，根据情况也可以调用其它相应的 profile 规则。配置完用户的帐号和密码后，PPPoE 服务就可以启动。

18.5 大型 PPPoE 服务的综合应用

PPPoE 认证由于是基于 OSI 七层参考模型第二层运行，所以不会受 IP 层数据的影响，特别是 ARP 协议，这样可以避免现在比较常见的 ARP 病毒攻击。PPPoE 是让每一个用户在二层 MAC 地址间建立一个虚拟的隧道，即保证了数据的安全，又保证了稳定。比起通过 IP 方式认证的 Web 页面要稳定安全的多。在一些网吧为了避免 ARP 病毒的侵扰，也在网吧内部建立的 PPPoE 认证方式，避免 ARP 对网吧带来上网电脑频繁掉线问题。

现在几乎所有的人都在使用 WindowsXP 或以上的操作系统，这些操作系统都自带了 PPPoE 拨号软件，即用户不需要太复杂的操作，就可以建立一个虚拟拨号连接。下面是一个 PPPoE 认证系统的网络结构：



- 1、首先采用 RouterOS 作为接入路由器和外网防火墙，这里我们采用 RB1000 设备或者 RouterOS x86PC 系统作为外网接入路由器，可以实现多线路多运营商的路由器，并作为 nat 转换设备，减轻 PPPoE 认证服务器的压力。
- 2、在接入路由器下面，我们可以根据用户数量，建立多个 PPPoE 服务器，采用 PPPoE 集群服务器方式均衡用户，一般通过一台高性能的 PC，如双核的服务器支持 1000 个左右 PPPoE 认证用户同时在线，更高的 PC 配置可以获得更高的在线用户数量。
- 3、通过核心交换 VLAN 的 Trunk 连接用户层交换机，并分配每个用户连接那个 PPPoE 服务器，这样可以通过 VLAN 划分用户区域，隔离不必要的数据，减小广播风暴。用户接入可以通过以太网的有线连接，也可以通过无线的 AP 接入网络，连接方式灵活多样化。
- 4、所有 PPPoE 服务器都采用同一个 Radius 服务器，这样帐号便于管理，特别在多 PPPoE 的集群认证下有利于冗余的工作，在一台 PPPoE 服务器停机后，其余的设备可以接替工作。配置 Radius 服务器也可以分担 RouterOS 在账号管理的负荷。

故障分析

- **我能够连接到服务器，Ping 也能完全通过，但我仍然不能打开 web 页面？**

  确定你在路由器上指定了正确的 DNS 服务器（在**/ip dns** 或在**/ppp profile** 中的 **dns-server** 参数）

- **我能使 PPPoE 连接小点的数据包（例如 pings）**

  你需要改变所以经过 PPPoE 连接的 **mss** 数据包为 1440：

```
[admin@MT] interface pppoe-server server> set 0 max-mtu=1440 max-mru=1440
[admin@MT] interface pppoe-server server> print
Flags: X - disabled
 0   service-name="mt" interface=wlan1 max-mtu=1440 max-mru=1440
     authentication=pap,chap,mschap1,mschap2 keepalive-timeout=10
     one-session-per-host=yes max-sessions=0 default-profile=default
[admin@MT] interface pppoe-server server>
```

- **我的 windows PPPoE 客户端得到了来至 MikroTik PPPoE server 的 IP 地址和默认网关,但不能出 PPPoE server 并且不能连接外部网络.**

  PPPoE 服务器没有与客户端连接， 为 PPPoE 客户端的地址配置伪装（masquerading）或是确定你为客户端分配的地址段指定了正确的路由,或是你在以太网卡上启用了 Proxy-ARP (请看 IP 地址和地址解析协议 Address Resolution Protocol (ARP))

- **我的 Windows XP 不能连接到 PPPoE 服务器**

  你要在 XP 的 pppoe 客户端属性中指明"Service Name"。或是没有在 MikroTik PPPoE 服务器配置服务名（service name），这样你会得到“line is busy”错误或是系统显示"verifying password - unknown error"

- **我想要记录连接建立的日志**

  在**/system logging facility** 中配置日志信息并启用 ppp 日志类型

第十九章 PPTP

PPTP（点对点隧道协议）支持 IP 上的加密隧道。MikroTik RouterOS 工具包含对 PPTP 客户和服务器的支持。PPTP 隧道的基本应用：

- 因特网上的安全路由器-路由器隧道
- 连接（桥接）本地企业网或 LAN（当使用了 EoIP 时）
- 对移动或远程客户远程访问企业网/公司的 LAN（参见 Windows 的 PPTP 设置以获取更多信息）

每个 PPTP 连接都包含一个服务器和客户。MikroTik RouterOS 可能作为一个服务器或者客户工作——或者，对多种配置，它可以对某些连接是服务器而对其他连接是客户。例如，下面创建的客户可以连接到 Windows 2000 服务器，另一个 MikroTik Router，或另一个支持 PPTP 服务器的路由器。

快速设置向导

在两个 IP 地址为 **10.5.8.104**（PPTP 服务器）及 **10.1.0.172** （PPTP 客户）的 MikroTik 路由器之间创建一个 PPTP 隧道，参考下面的步骤：

- PPTP 服务器上的设置：
  1. 添加一个用户：

```
[admin@PPTP-Server] ppp secret> add name=jack password=pass local-address=10.0.0.1
remote-address=10.0.0.2
```

2. 启用 PPTP 服务器：

```
[admin@PPTP-Server] interface pptp-server server> set enabled=yes
```

- PPTP 客户的设置：
  1. 添加 PPTP 客户：

```
[admin@PPTP-Client] interface pptp-client> add user=jack password=pass
connect-to=10.5.8.104 disabled=no
```

规格

功能包要求: ***ppp***
等级要求: *Level1 (限制 1 个在线)* , *Level3 (限制 1 在线)* , *Level4 (限制 200 在线)*，*Level5（无限制）*
操作路径: ***/interface pptp-server***, ***/interface pptp-client***
标准与技术: PPTP (RFC 2637)

点对点隧道协议（PPTP）是一种支持多协议虚拟专用网络的网络技术。通过该协议，远程用户能够通过 windows 客户端或者路由器，以及其它装有点对点协议的系统安全访问公司网络，并能拨号连入本地 ISP，通过 Internet 安全链接到公司网络。

PPTP 包含了 PPP 认证及对每个 PPTP 连接的帐户管理。全部的认证和每个连接的帐户管理可以通过 RADIUS 客户或本地完成。支持 MPPE 40bit RC4 以及 MPPE 128bit RC4 加密。

PPTP 的连接采用的是 TCP 端口 1723 和 IP 协议 GRE（类属路由封装，IP 协议 ID 47）。PPTP 可以通过启用定为 TCP 端口 1723 和 47，注意让相应的路由器不会对这两个端口做防火墙过滤等操作，否则 PPTP 连接会失效。

19.1 PPTP 客户设置

操作路径: ***/interface pptp-client***

属性描述

add-default-route (yes | no; default: **no**) - 是否像使用默认路由器（网关）一样使用该客户连接到的服务器
allow (*多选项:* mschap2, mschap1, chap, pap; default: **mschap2, mschap1, chap, pap**) -允许客户用来认证的协议
connect-to (*IP address*) - PPTP 服务器连接到的 IP 地址
mru (*整型*; default: **1460**) - 最大接受单元。最优值是隧道工作的接口 MRU 减少 40（所以，1500 字节以太网连接设置 MRU 为 1460 以避免包的分割）
mtu (*整型*; default: **1460**) -最大传输单元。最优值是隧道工作的接口 MTU 减少 40（所以，1500 字节以太网连接设置 MTU 为 1460 以避免包的分割）
name (*名称*; default: **pptp-outN**) - 参考接口名
password (*文本*; default: **""**) -当登陆远程服务器时用户的密码

profile (*名称*; default: **default**) -当连接到远程服务器时使用的概要简介
user (*文本*) - 当登陆到远程服务器时使用的用户名

使用用户名为 **john** 密码为 **john**，设置 PPTP 名为 **test2** 的客户连接到 **10.1.1.12**PPTP 服务器并使用它作为默认网关：

```
[admin@MikroTik] interface pptp-client> add name=test2 connect-to=10.1.1.12 \
\... user=john add-default-route=yes password=john
[admin@MikroTik] interface pptp-client> print
Flags: X - disabled, R - running
  0 X  name="test2" mtu=1460 mru=1460 connect-to=10.1.1.12 user="john"
       password="john" profile=default add-default-route=yes

[admin@MikroTik] interface pptp-client> enable 0
```

属性描述

encoding (*文本*) -加密及编码（如果非对称，使用‘/’分隔）在该连接中使用
status (*文本*) - status of the client
Dialing – 试图进行连接
Verifying password... - 连接已建立到服务器，正在核实密码
Connected - 毋需解释的
Terminated -没有启用借口或另一端不能建立连接
uptime (*time*) - 以天，小时，分钟以及秒钟显示的连接时间
命令名：***/interface pptp-client monitor***

一个已建立连接的实例：

```
[admin@MikroTik] interface pptp-client> monitor test2
     uptime: 4h35s
   encoding: MPPE 128 bit, stateless
     status: Connected
[admin@MikroTik] interface pptp-client>
```

19.2 PPTP 服务器设置

操作路径：***/interface pptp-server server***

PPTP 服务器为每个连接的 PPTP 客户创建了一个动态的接口。PPTP 连接依靠你所有的证书登记从客户计数。Level1 证书允许一个 PPTP 客户，Level3 或 Level4 证书最多允许 200 客户，Level5 或 Level6 证书没有 PPTP 客户限制。

为了创建 PPTP 用户，你应该咨询 PPP secret 以及 PPP Profile 手册。也可以使用 MikroTik 路由器作为 RADIUS 客户来注册 PPTP 用户。

属性描述

authentication (*多选项:* pap | chap | mschap1 | mschap2; default: **mschap2**) - 认证算法
default-profile -默认概要信息

enabled (yes | no; default: **no**) - 定义 PPTP 服务器是否启用

keepalive-timeout (*time*; default: **30**) -定义路由器开始每秒发送持活时间数据包之后的时间段（以秒计算）。如果没有流量并且没有保持活动，在那段时间将出现反应（例如，2 * keepalive-timeout），没有反应的客户将被宣布为断开连接。

mru (*整型*; default: **1460**) - 最大接收单元。最优值是隧道工作的接口 MRU 减少 40（所以，1500 字节以太网连接设置 MRU 为 1460 以避免包的封装问题）

mtu (*整型*; default: **1460**) -最大传输单元。最优值是隧道工作的接口 MTU 减少 40（所以，1500 字节以太网连接设置 MTU 为 1460 以避免包的封装问题）

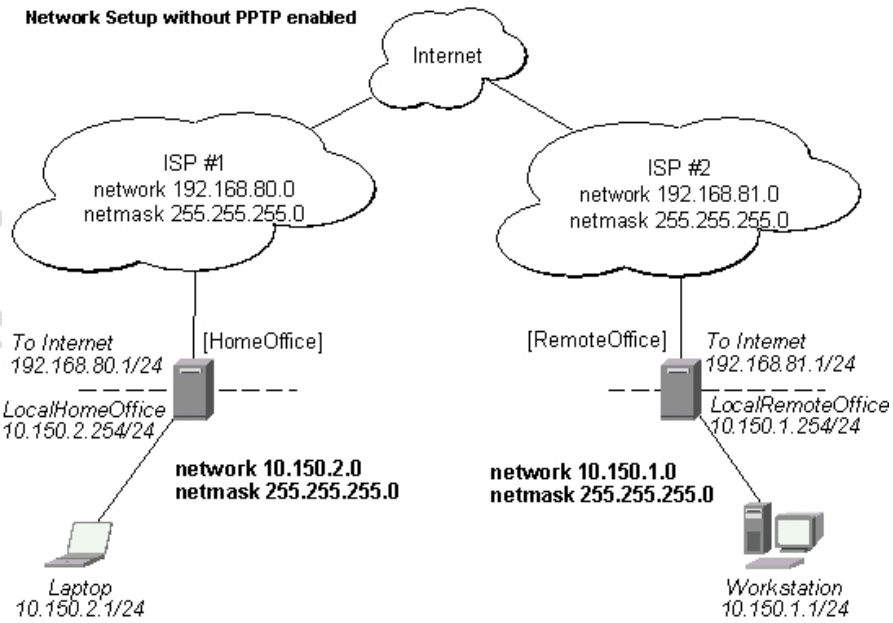
启用 PPTP 服务器：

```
[admin@MikroTik] interface pptp-server server> set enabled=yes
[admin@MikroTik] interface pptp-server server> print
            enabled: yes
                mtu: 1460
                mru: 1460
     authentication: mschap2,mschap1
   keepalive-timeout: 30
     default-profile: default
[admin@MikroTik] interface pptp-server server>
```

19.3 PPTP 应用实例

Router to Router 安全隧道实例

以下是一个使用互联网上的加密 PPTP 隧道连接两个企业网局域网的例子：



在这个例子中有两个不同地区办公室的路由器，需要让两个办公局域网的主机之间实现互访：

- **[HomeOffice]**

接口 LocalHomeOffice　10.150.2.254/24
接口 ToInternet　192.168.80.1/24

- **[RemoteOffice]**

接口 ToInternet　192.168.81.1/24
接口 LocalRemoteOffice　10.150.1.254/24

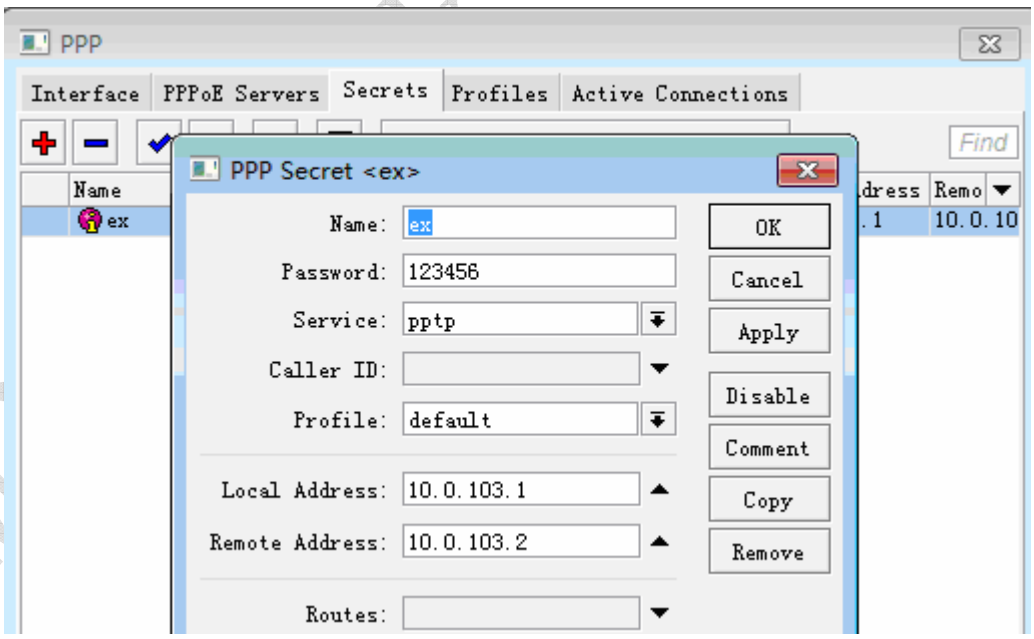
每个路由器连接到当地的 ISP，任何一个路由器可以通过互联网访问到对端的路由器。

HomeOffice 配置

在 HomeOffice 端建立 PPTP 服务器，首先我们进入/ppp secret 目录下添加客户端账号：

```
[admin@HomeOffice] ppp secret> add name=ex service=pptp password=123456
local-address=10.0.103.1 remote-address=10.0.103.2
[admin@HomeOffice] ppp secret> print detail
Flags: X - disabled
 0   name="ex" service=pptp caller-id="" password="123456" profile=default
     local-address=10.0.103.1 remote-address=10.0.103.2 routes==""

[admin@HomeOffice] ppp secret>
```

Winbox 操作如下：

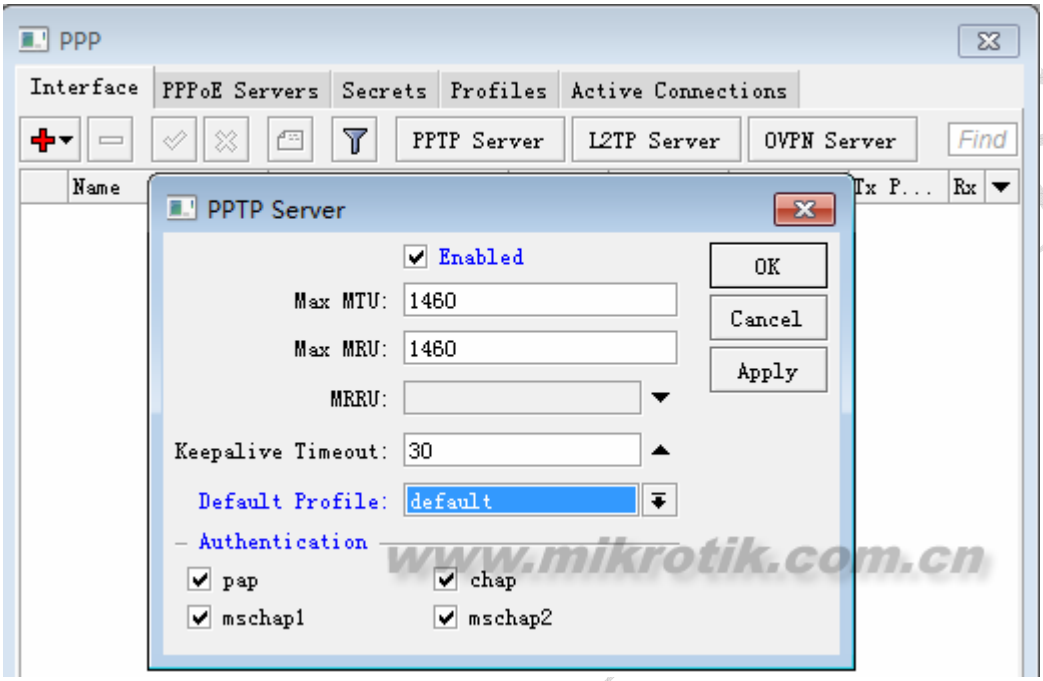


在 interface pptp-server server 目录下，启用 pptp 服务器：

```
[admin@HomeOffice] interface pptp-server server> set enabled=yes
[admin@HomeOffice] interface pptp-server server> print
          enabled: yes
              mtu: 1460
```

```
              mru: 1460
    authentication: mschap2
   default-profile: default
[admin@HomeOffice] interface pptp-server server>
```

Winbox 下配置进入 ppp 目录下启用 pptp server：

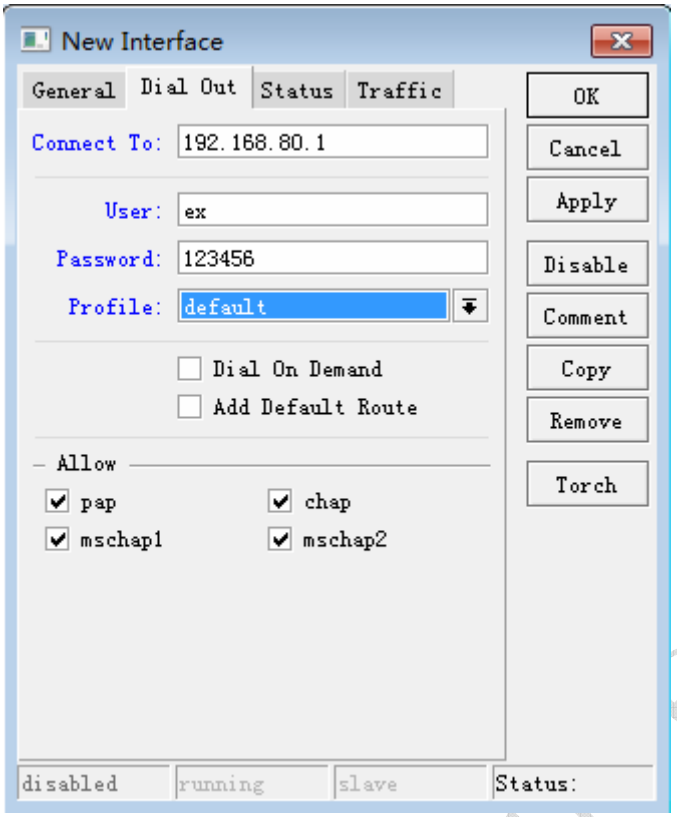


RemoteOffice 配置

在 RemoteOffice 路由器添加一个 PPTP 客户：

```
[admin@RemoteOffice] interface pptp-client> add connect-to=192.168.80.1 user=ex \
\... password=123456 disabled=no
[admin@RemoteOffice] interface pptp-client> print
Flags: X - disabled, R - running
  0  R name="pptp-out1" mtu=1460 mru=1460 connect-to=192.168.80.1 user="ex"
       password="123456" profile=default add-default-route=no

[admin@RemoteOffice] interface pptp-client>
```

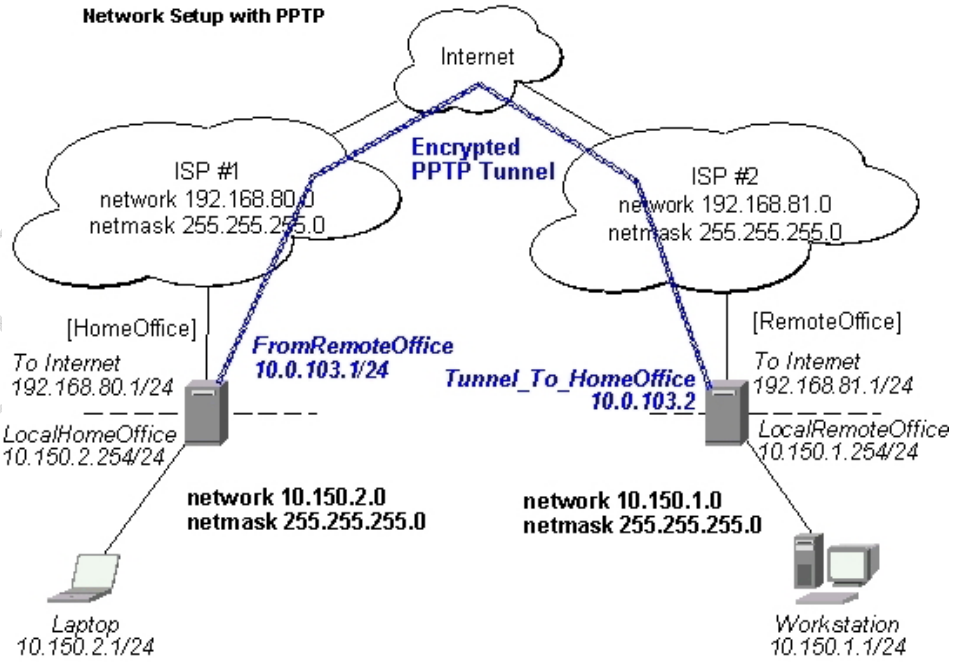
Winbox 在 interface 中添加 pptp-client



这样，一个 PPTP 隧道就在路由器之间创建好了。这个隧道就像在 IP 地址为 10.0.103.1 及 10.0.103.2 的路由器之间的三层点对点连接。

pptp 局域网的互访

pptp 隧道建立完成后，仅是路由器间可以互访，但两个企业间的局域网需要通过设置路由完成连接



为了在 PPTP 隧道上互访企业间本地网络，需要添加以下路由：

```
[admin@HomeOffice] > ip route add dst-address=10.150.1.0/24 gateway=10.0.103.2
```

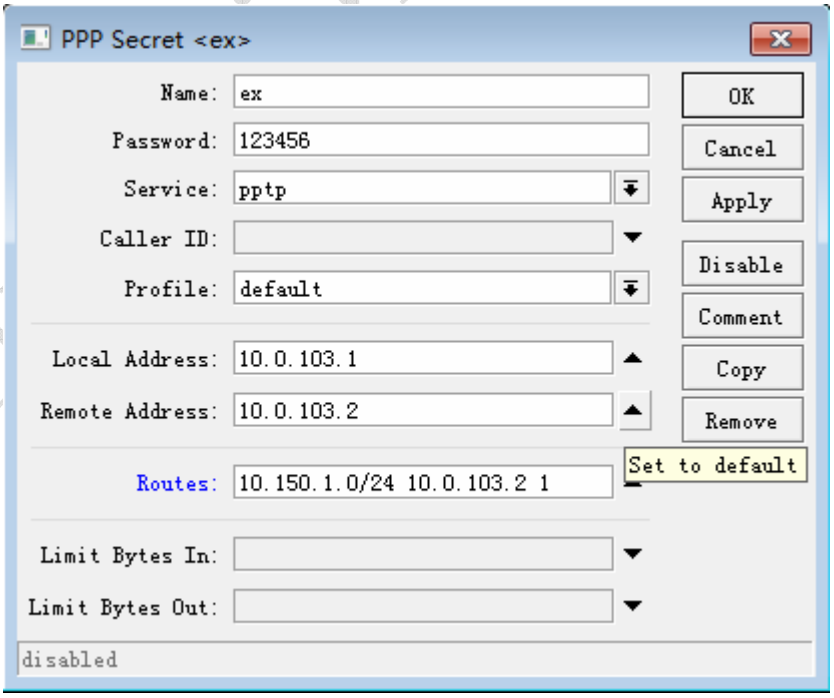
```
[admin@RemoteOffice] > ip route add dst-address=10.150.2.0/24 gateway=10.0.103.1
```

或者也可以在 PPTP 服务器（HomeOffice）上通过用户配置的 routes 参数完成，RemoteOffice 还是需要在 /ip route 中配置路由：

```
[admin@HomeOffice] ppp secret> print detail
Flags: X - disabled
 0   name="ex" service=pptp caller-id="" password="123456" profile=default
     local-address=10.0.103.1 remote-address=10.0.103.2 routes==""

[admin@HomeOffice] ppp secret> set 0 routes="10.150.1.0/24 10.0.103.2 1"
[admin@HomeOffice] ppp secret> print detail
Flags: X - disabled
 0   name="ex" service=pptp caller-id="" password="123456" profile=default
     local-address=10.0.103.1 remote-address=10.0.103.2
     routes="10.150.1.0/24 10.0.103.2 1"

[admin@HomeOffice] ppp secret>
```

目的路由：10.150.1.0/24
pptp 的网关：10.0.103.2
Distance 路径： 1

Winbox 中修改 routes 参数



测试 PPTP 隧道连接：

```
[admin@RemoteOffice]> /ping 10.0.103.1
10.0.103.1 pong: ttl=255 time=3 ms
```

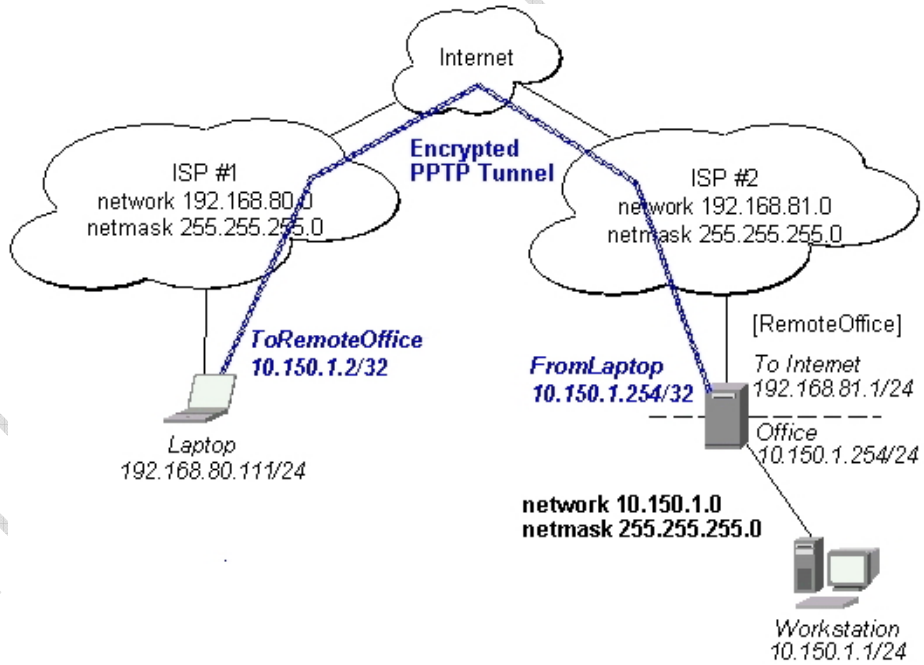
```
10.0.103.1 pong: ttl=255 time=3 ms
10.0.103.1 pong: ttl=255 time=3 ms
ping interrupted
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 3/3.0/3 ms
```

测试通过 PPTP 隧道到 LocalHomeOffice 接口的连接：

```
[admin@RemoteOffice]> /ping 10.150.2.254
10.150.2.254 pong: ttl=255 time=3 ms
10.150.2.254 pong: ttl=255 time=3 ms
10.150.2.254 pong: ttl=255 time=3 ms
ping interrupted
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 3/3.0/3 ms
```

19.4 通过 PPTP 隧道连接终端客户

下面的例子显示了通过终端电脑与远程办公网络进行 PPTP 加密隧道通信，如外地出差的同时，通过笔记本电脑连接会公司的网络进行远程信息管理和查询



这个例子中的路由器：

- **[RemoteOffice]**

  接口 ToInternet　192.168.81.1/24
  接口 Office　10.150.1.254/24

在 PPTP 服务器上设置用户帐号：

```
[admin@RemoteOffice] ppp secret> add name=ex service=pptp password=123456
local-address=10.150.1.254 remote-address=10.150.1.2
[admin@RemoteOffice] ppp secret> print detail
Flags: X - disabled
 0   name="ex" service=pptp caller-id="" password="123456" profile=default
     local-address=10.150.1.254 remote-address=10.150.1.2 routes==""

[admin@RemoteOffice] ppp secret>
```

启用 pptp 服务：

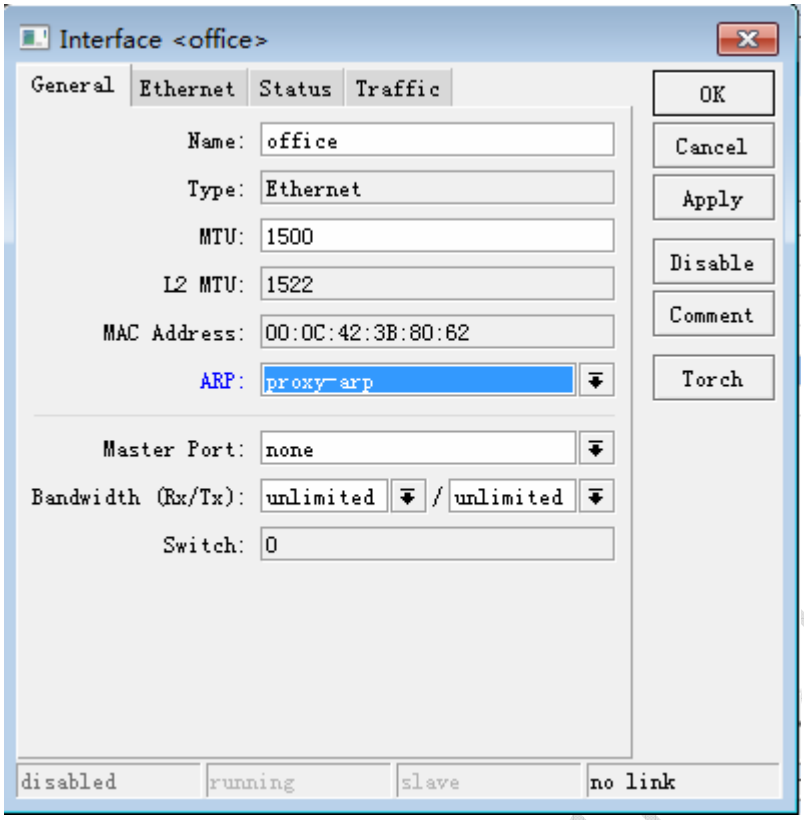
```
[admin@RemoteOffice] interface pptp-server server> set enabled=yes
[admin@RemoteOffice] interface pptp-server server> print
          enabled: yes
              mtu: 1460
              mru: 1460
    authentication: mschap2
   default-profile: default
[admin@RemoteOffice] interface pptp-server server>
```

在笔记本电脑访问回公司后，需要访问内部网络资源，需要配置规则才能确保通常，有两种方法：

局域网连接方法 1：代理 ARP 必须在'Office'接口上启用，这样可以通过代理 arp 访问，但有个缺点是内外的 DHCP 服务可能会受到影响：

```
[admin@RemoteOffice] interface ethernet> set Office arp=proxy-arp
[admin@RemoteOffice] interface ethernet> print
Flags: X - disabled, R - running
 #    NAME               MTU   MAC-ADDRESS      ARP
 0  R ToInternet          1500  00:30:4F:0B:7B:C1 enabled
 1  R Office              1500  00:30:4F:06:62:12 proxy-arp
[admin@RemoteOffice] interface ethernet>
```

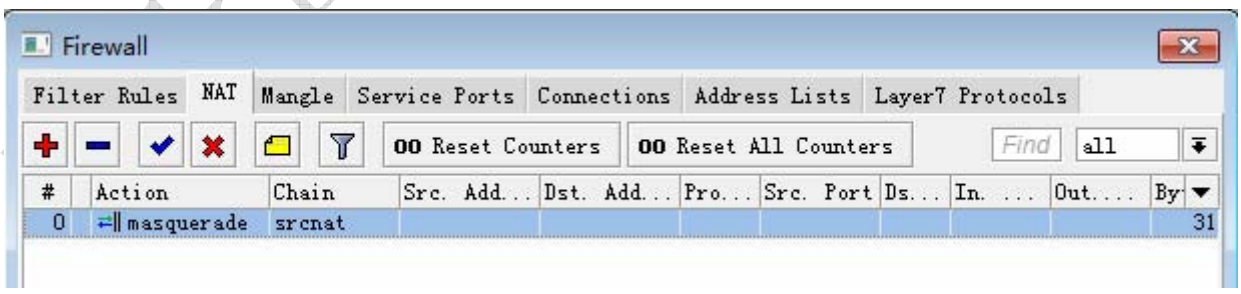
在 winbox 中进入 interface 目录下，选择 office 接口设置 arp 为 proxy-arp



局域网连接方法 2：通过 nat 设置 masquerade，规则要求对所有来访数据进行伪装，这样保证内外网通过转换通信

```
[admin@RemoteOffice] /ip firewall nat> add chain=srcnat action=masquerade
[admin@RemoteOffice] /ip firewall nat> print
Flags: X - disabled, R - running
  Flags: X - disabled, I - invalid, D - dynamic
 0   chain=srcnat action=masquerade
[admin@RemoteOffice] interface ethernet>
```

在 winbox 中添加 masquerade 规则：



Windows 的PPTP 设置

对 Windows NT，2000，98SE 以及 98 支持 PPTP 客户。Windows 98SE，2000，以及 ME 包括 Windows 设置中的支持或者自动安装 PPTP。对 95，NT，及 98，安装需要从 Microsoft 下载。很多 ISP 都制作了帮助页面以帮助客户进行 Windows PPTP 安装。

PPTP (VPN)安装的简单说明及客户设置 - Windows 98SE

如果 VPN（PPTP）套件已经安装，选择'Dial-up Networking' 和 'Create a new connection'。创建一个 VPN 的选项应该选择。如果没有 VPN 选项，那么按照下面的安装说明进行。当询问 VPN 服务器主机名或 IP 地址时，输入路由器的 IP 地址。双击'new'图标并输入正确的用户名和密码（必须在路由器或用于认证的用户数据库中）。

连接的设置在选择了'connect'按钮后需要 9 秒钟。建议把连接属性进行编辑以便'NetBEUI', 'IPX/SPX compatible'，及'Log on to network'为未选择的。连接的设置时间为在'connect'按钮选择后 2 秒钟。

为了安装 Windows 98SE 的 VPN 套件，从'Start'主目录中选择'Setting'。选择'Control Panel'，选择'Add/Remove Program'，选择'Windows setup'标签，选择 'Communications'软件安装以及'Details'。在软件列表的底部选择'Virtual Private Networking'安装。

故障分析

- **我使用了防火墙但我不能建立 PPTP 建立**

  确定 TCP 连接到 1723 端口可以通过你的两个站点。而且，TCP 协议 47 应该通过。

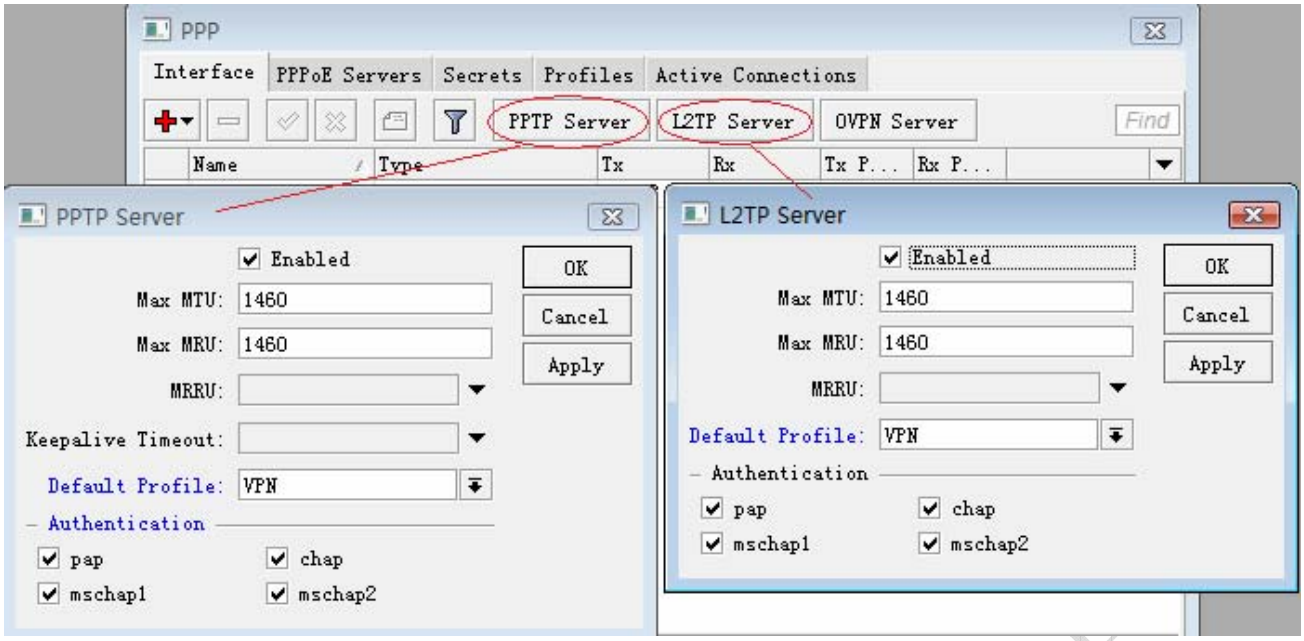
第二十章 PPTP 与 L2TP 服务

PPTP 和 L2TP 都使用 PPP 协议对数据进行封装，然后添加附加包头用于数据在互联网络上的传输。尽管两个协议非常相似，但是仍存在以下几方面的不同：

PPTP 要求互联网络为 IP 网络。L2TP 只要求隧道媒介提供面向数据包的点对点的连接。PPTP 只能在两端点间建立单一隧道。L2TP 支持在两端点间使用多隧道。使用 L2TP，用户可以针对不同的服务质量创建不同的隧道。L2TP 可以提供包头压缩。当压缩包头时，系统开销（overhead）占用 4 个字节，而 PPTP 协议下要占用 6 个字节。L2TP 可以提供隧道验证，而 PPTP 则不支持隧道验证。但是当 L2TP 或 PPTP 与 IPSEC 共同使用时，可以由 IPSEC 提供隧道验证，不需要在第 2 层协议上验证隧道

20.1 同时建立 PPTP 和 L2TP 服务器

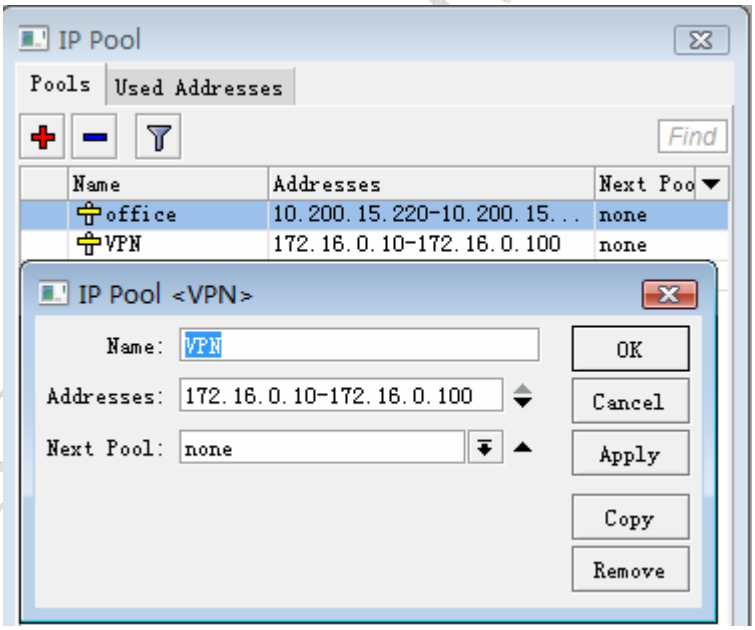
首先我看一下 PPTP 和 L2TP 的建立，同样是在 PPP 的目录下，只是选择的服务不同，一个是 PPTP 服务，一个是 L2TP 服务：



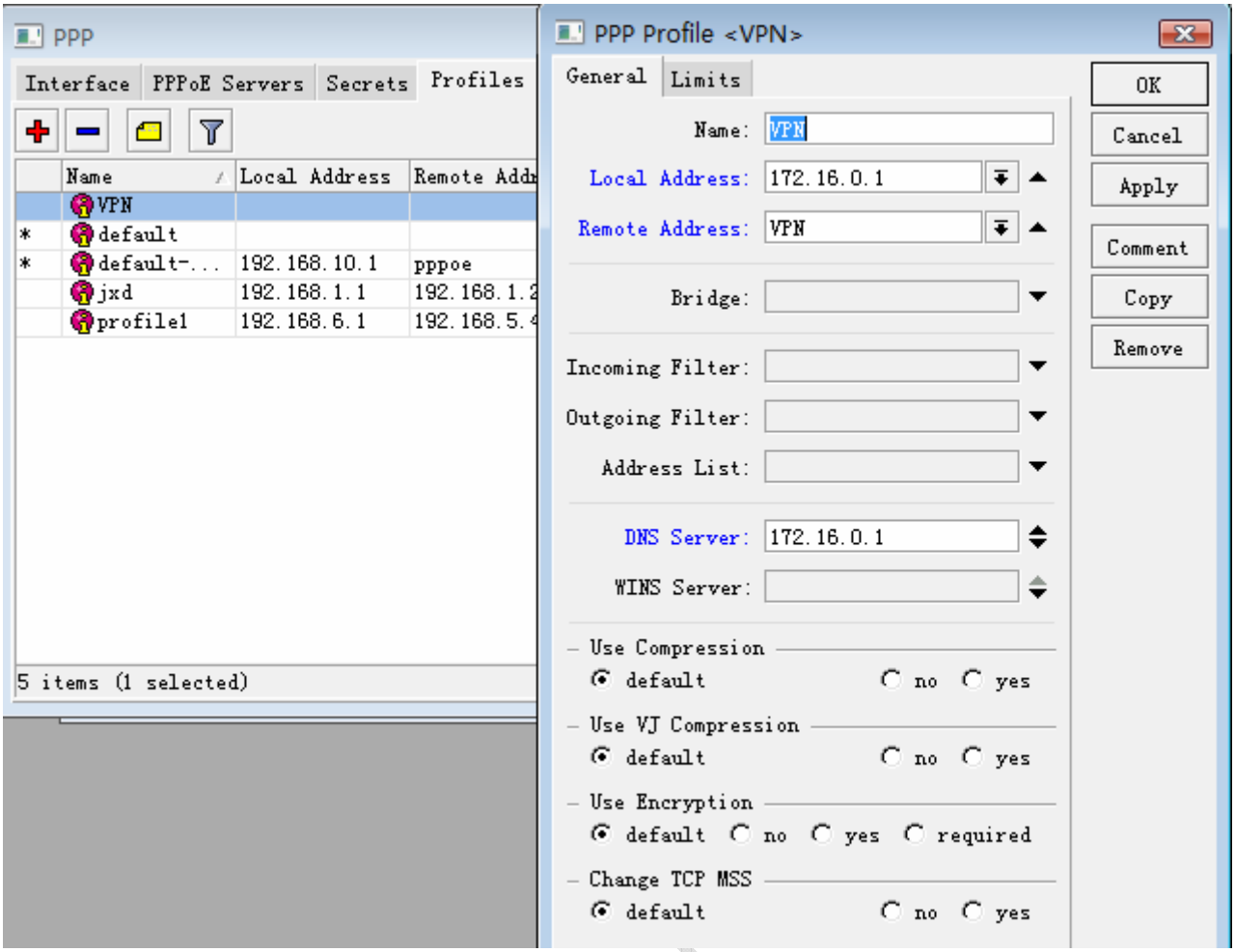
这里选择的 Profile 类型完全相同，都为 VPN 应用，Autentication 认证方式也可以选择相同方式。

这里我们举一个实例，我们建立了一个主机的 VPN 服务，同时启用 PPTP 和 L2TP 方式，分配远程 IP 为 172.16.0.10-172.16.0.100 的地址池，我用 172.16.0.11 做为 VPN 隧道的本地 IP。

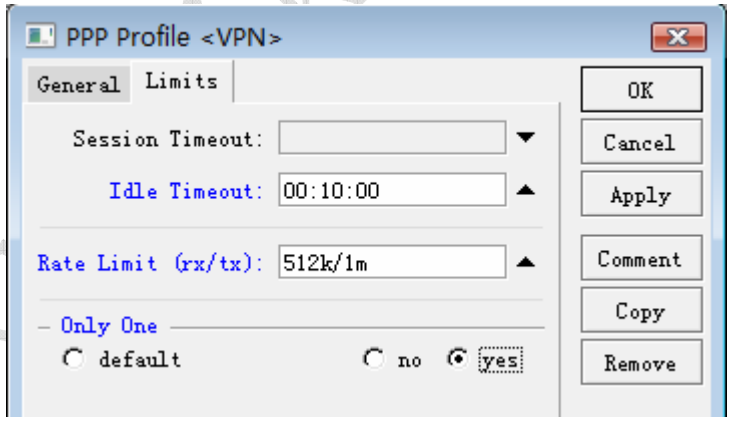
首先我进入 ip pool 中配置地址池：



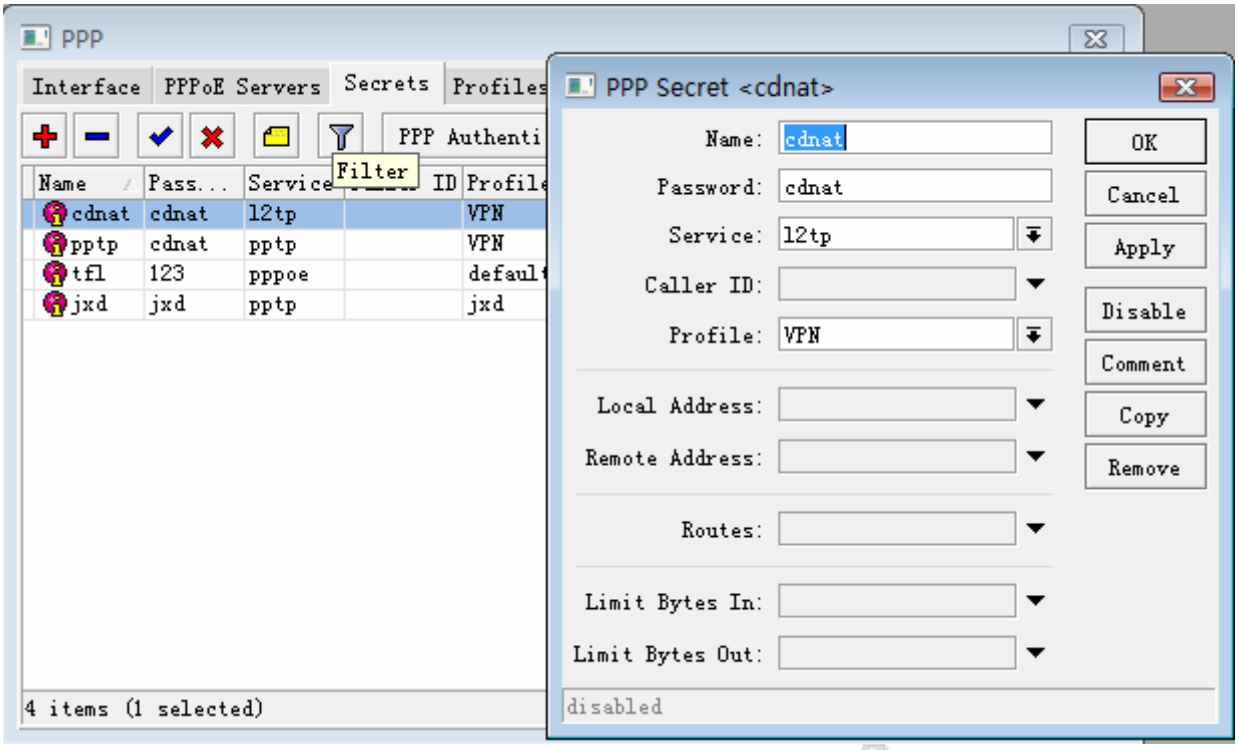
配置好地址池后，在 PPP Profiles 中添加用户组规则，这里我们添加一个组规则取名 VPN，配合本地 IP 地址 172.16.0.1，在远程 remote-address 种配置之前添加号的地址池 VPN，设置 DNS 为 172.16.0.1,其他配置参数如下：



在 limits 选项中，配置相应的 Idle-timeout（空闲超时时间）、Rate-limit（带宽）和 Only-one（帐号是否唯一性）：



在 PPP secret 中配置用户帐号信息，service 参数用于选择，启用服务器的类型，这里我们添加了 cdnat 和 pptp 两种类型的帐号，分别对应 L2TP 和 PPTP 登陆方式，profile 类型选择 VPN。



配置完成后，我们便可以通过 PPTP 或者 L2TP 连接 RouterOS 的 VPN 服务，在 windows 下可以通过 PPTP 的方式直接连接 RouterOS 的 VPN 服务，但 L2TP 不行，因为 windows 要求 L2TP 进行 IPsec 的加密方式连接，这里我们可以通过修改 windows 注册表连接

L2TP 的 Windows 注册表修改

L2TP 修改注册表，缺省的 Windows XP L2TP 传输策略不允许 L2TP 传输不使用 IPSec 加密。可以通过修改 Windows XP 注册表来禁用缺省的行为，手工修改：

1) 进入 Windows XP 的“开始”“运行”里面输入“Regedt32”，打开“注册表编辑器”，定位“HKEY_Local_Machine \ System \ CurrentControl Set \ Services \ RasMan \Parameters ”主键。

2) 为该主键添加以下键值：

- 键值：ProhibitIpSec
- 数据类型：reg_dword
- 值：1

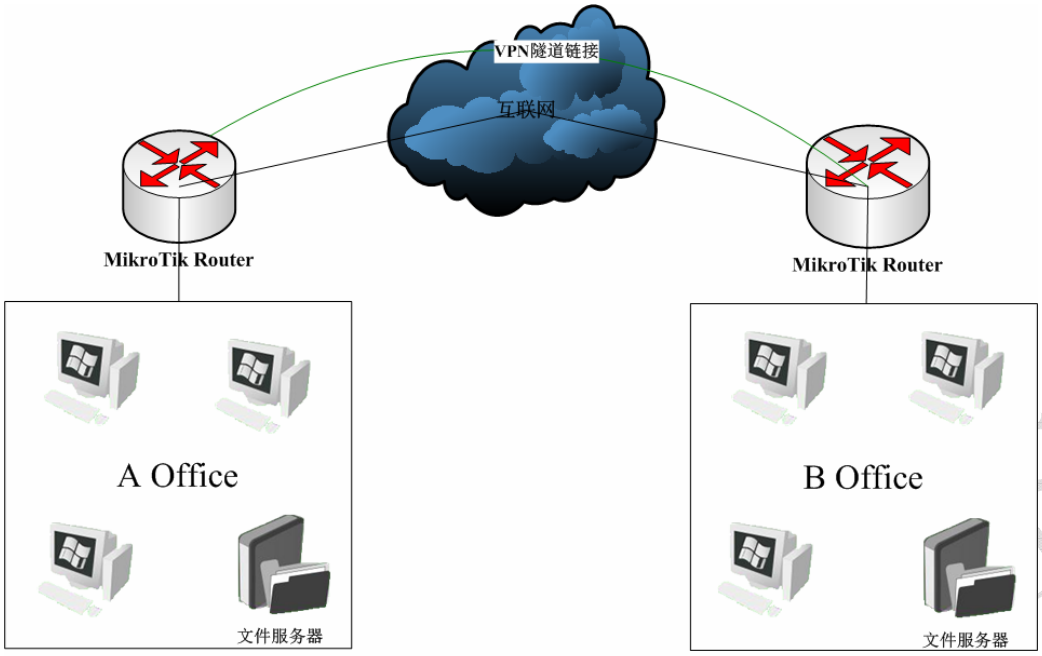
修改后即可通过 windows 正常连接到 L2TP 服务。

20.2 VPN 的几种应用方式

MikroTik 的 VPN 系统支持多种方式的应用，能实现企业对等访问、企业与多分支点、多点与移动办公和 VPN 数据转移等多种 VPN 连接方式。在 VPN 数据转移方面用处最多的方式是在 VoIP 方面，因为受某些 ISP 网络的限制，使得正常的 VoIP 通讯收到影响，所以可以通过 VPN 的方式实现数据的转移。

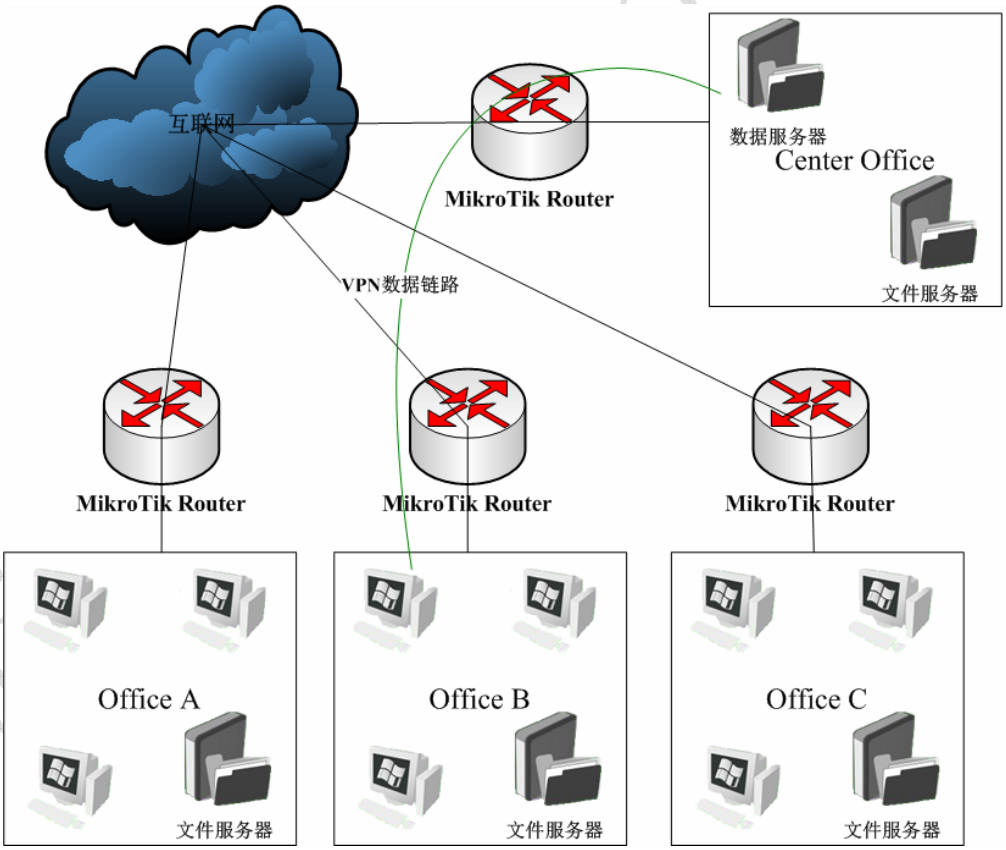
企业间对等互访

1、



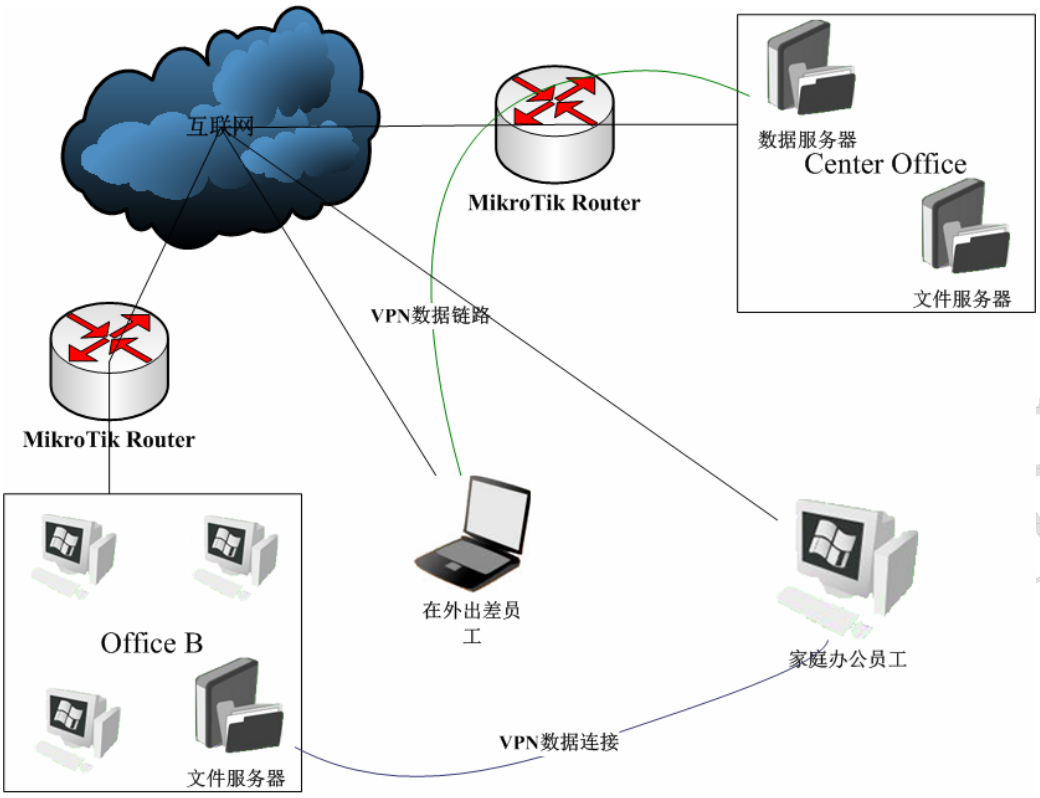
通过 VPN 隧道协议如 PPTP、L2TP 或者 IPIP 可以建立一个对等的隧道，使得两个办公室能互相访问公司数据和文件。这种方式我们首先建立 PPP 服务器，并给客户端分配帐号和固定 IP 地址、建立 PPP 的拨号和分配 IP 地址，最后设置两个远程局域网的 IP 地址路由（操作可以参考 PPTP 章节）

企业与分支点总分连接



我们可以使用 PPTP 或者 L2TP 等隧道协议，建立多个客户端账号，使多个分公司能链接到总公司的中心服务器，并能通过公司管理各个点的数据和信息互联。这样总公司做到对分公司的有效管理，可以即时发布信息到各个分公司。并能实现数据的安全传输。

企业移动办公与综合应用



RouterOS 支持普通用户的 PPTP 和 L2TP 的 windows 的拨号连接，所以对于在外出差和家庭办公的用户就可以方便的链接到总部的中心服务器和分支点的文件服务器，特别对需要即时处理问题的公司员工最为适合。这种方式也适用于建立 VoIP 数据的转移和企业内部的 IP 语言通讯。

第二十一章 Open VPN

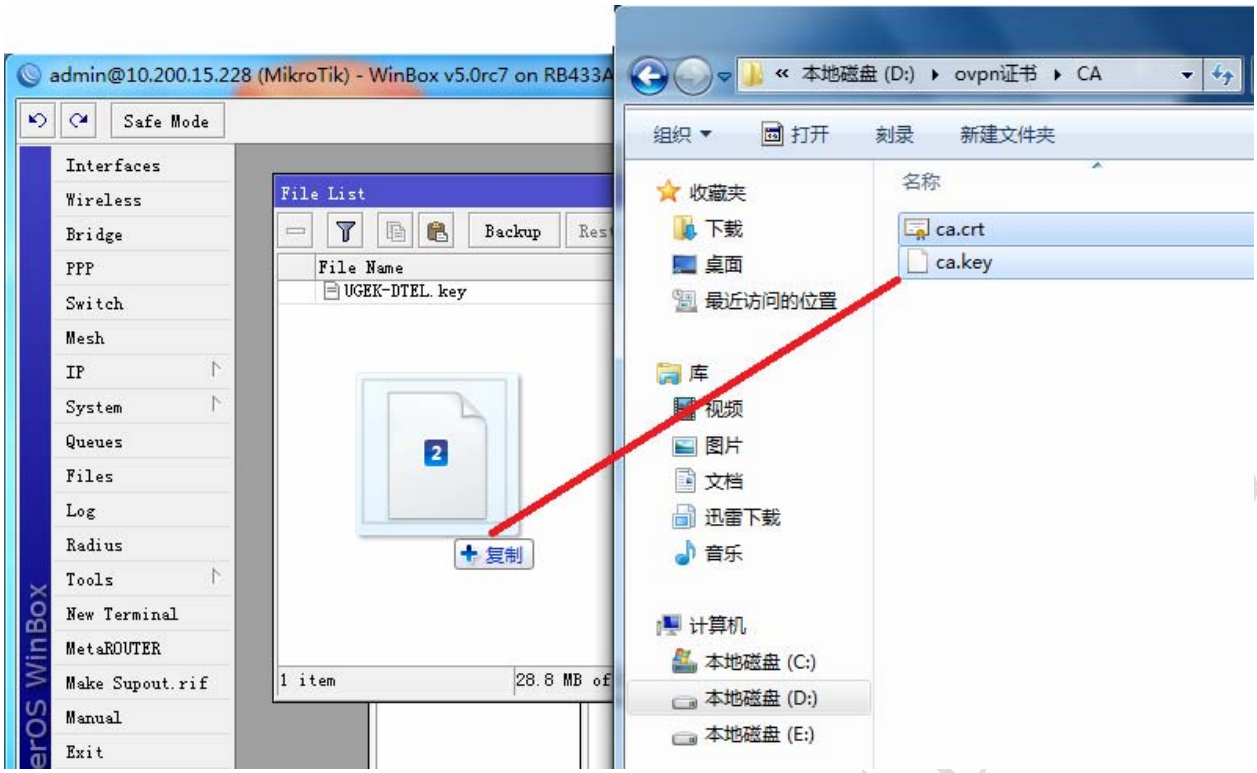
OpenVPN 已经被转移到各种平台，包括 Linux 和 Windows，**RouterOS** 在 v3.x 支持 OpenVPN ，你需要通过安装和启用 ppp 功能包。在 RouterOS 平台对 OpenVPN 仅支持 tcp 方式，udp 方式不支持。

在 **Windows** 你需要另外的 GUI，系统中允许你在 windows 系统中安装客户端，你可以到 OpenVPN GUI 下载 http://www.openvpn.se/download.html .

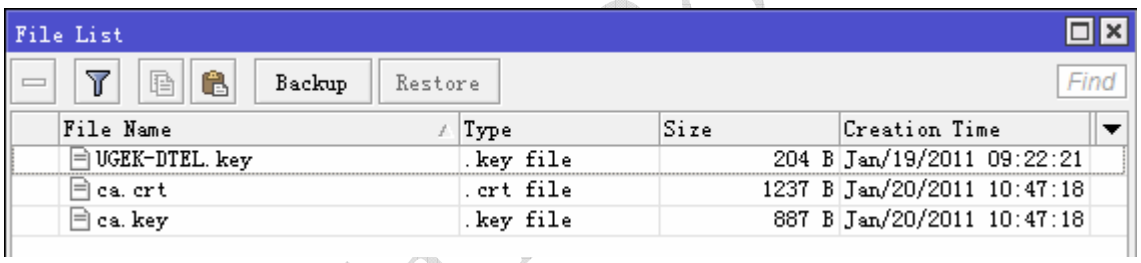
OpenVPN 要求与 SSL 证书一起工作，你需要生成一个证书，一般可以通过 linux 安装 OpenVPN 功能包后生成，或者通过 http://cacert.org 网站申请

21.1 OVPN 配置

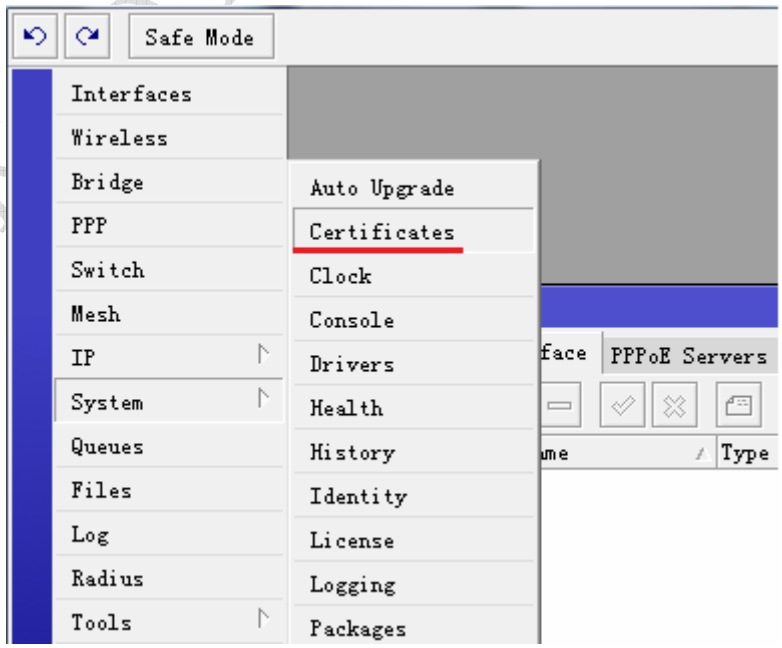
OVPN 服务器配置，首先要导入证书，否则 OVPN 与客户端是无法连接的，我们可以从网上下载已有的证书或者自己通过已经安装 OVPN 的 linux 系统生成新的证书，这里我们根据已有的证书进行操作，如下图：

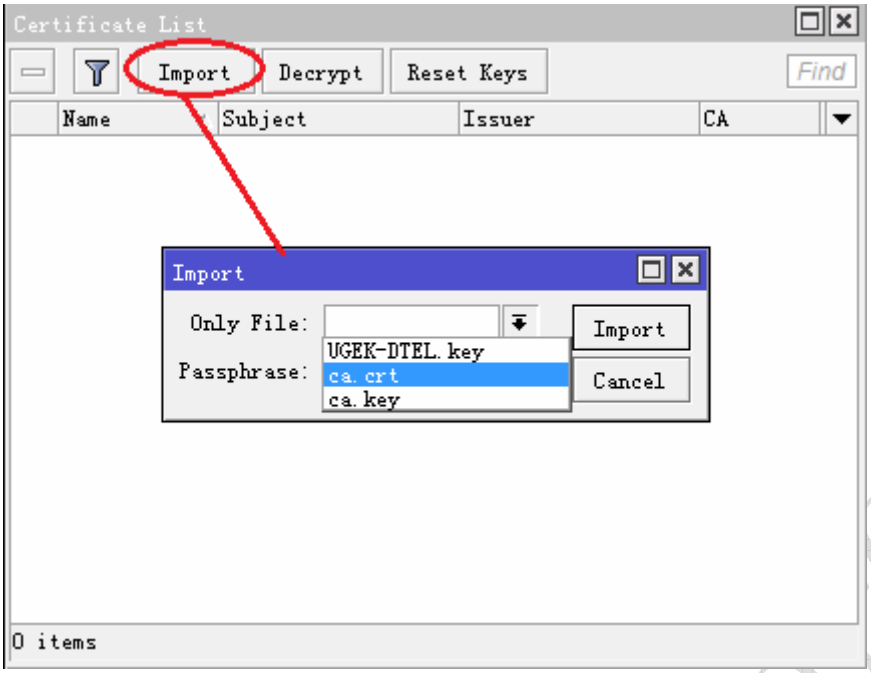


导入完成后，在 file list 可以看到证书

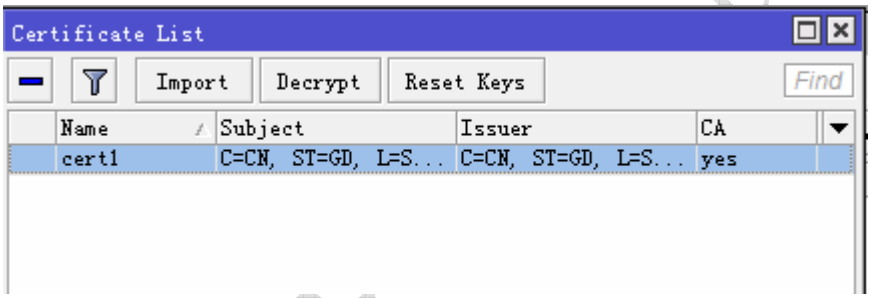


选择 system certificates 证书，并导入 crt 和 key 文件

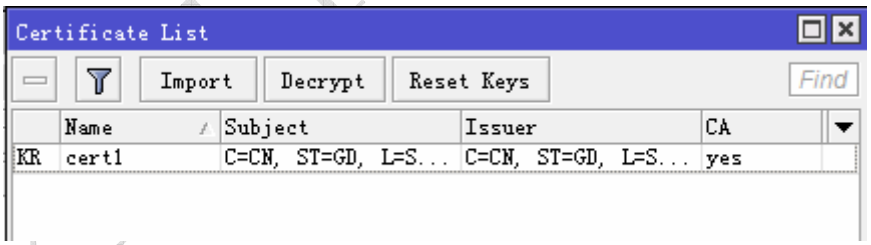




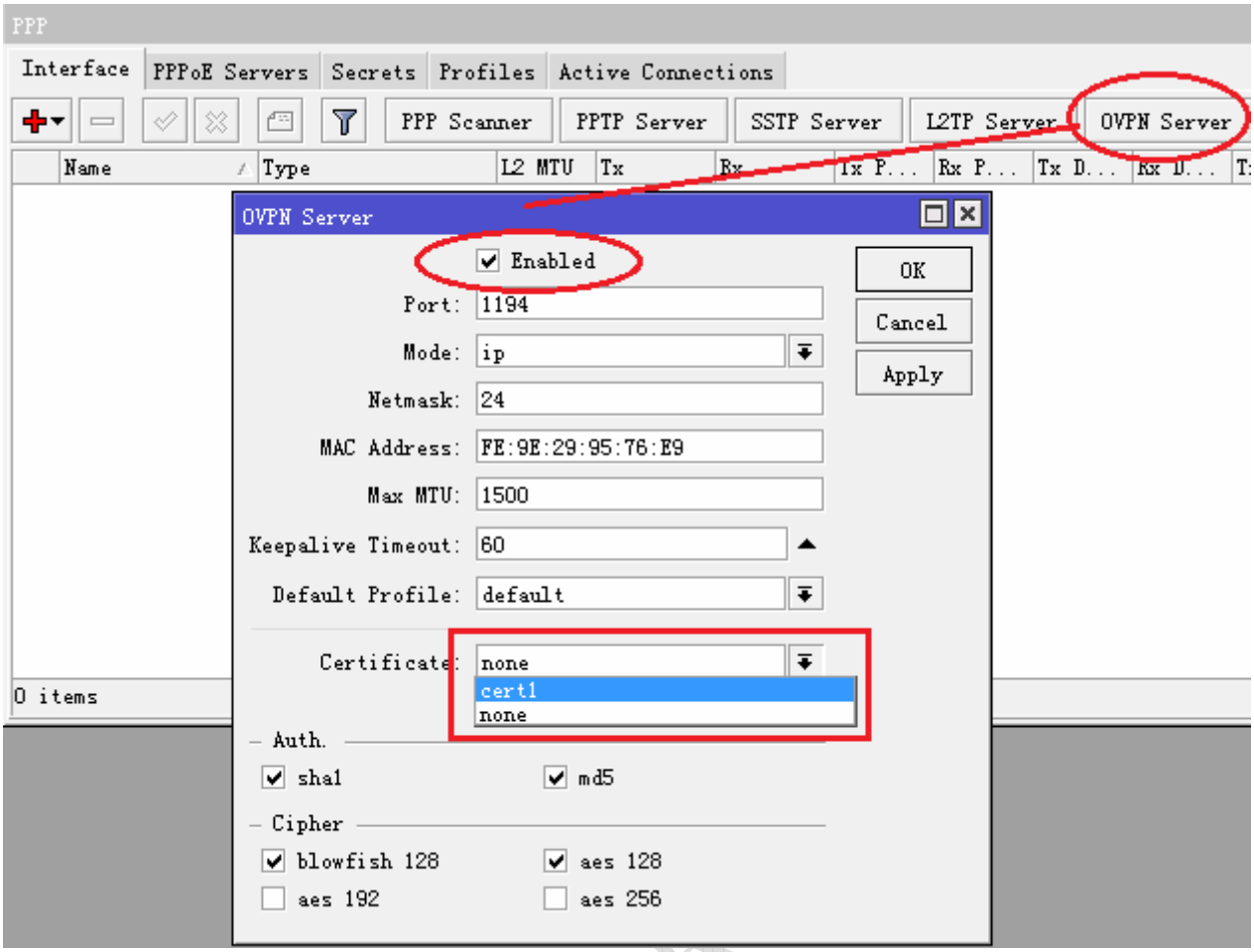
点击 import 导入 ca.crt 文件



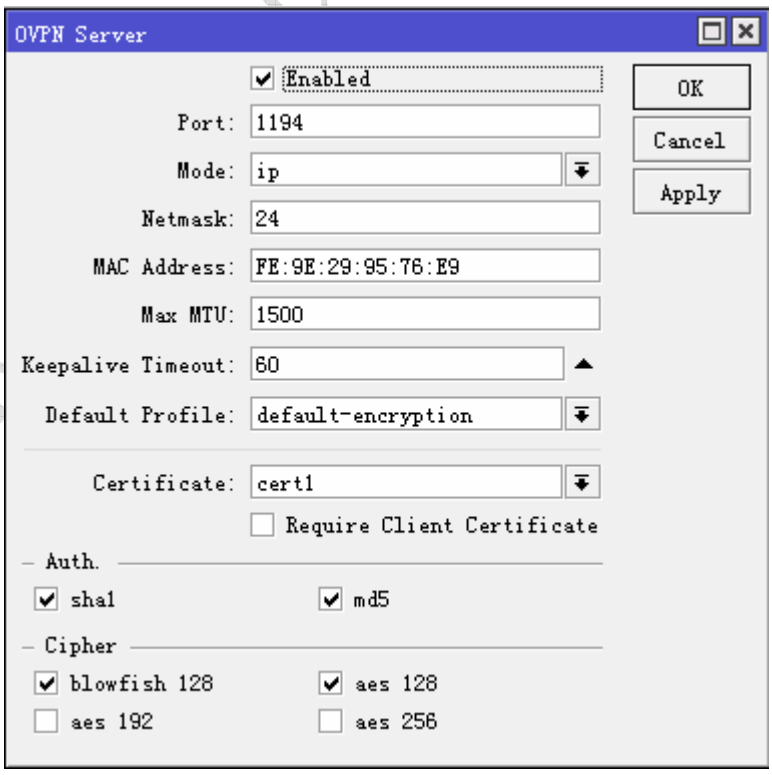
用同样的方式，导入 ca.key 文件，在当前目录下的项目出现了 KR 的前缀，表示已经导入 key 并运行



剩下的步骤就是启用 OVPN 服务，选择 enable 启用 OVPN 服务，并选择 certificate 为 cer1，其他参数默认配置，当然端口你可以自己选择，这里默认是 1194

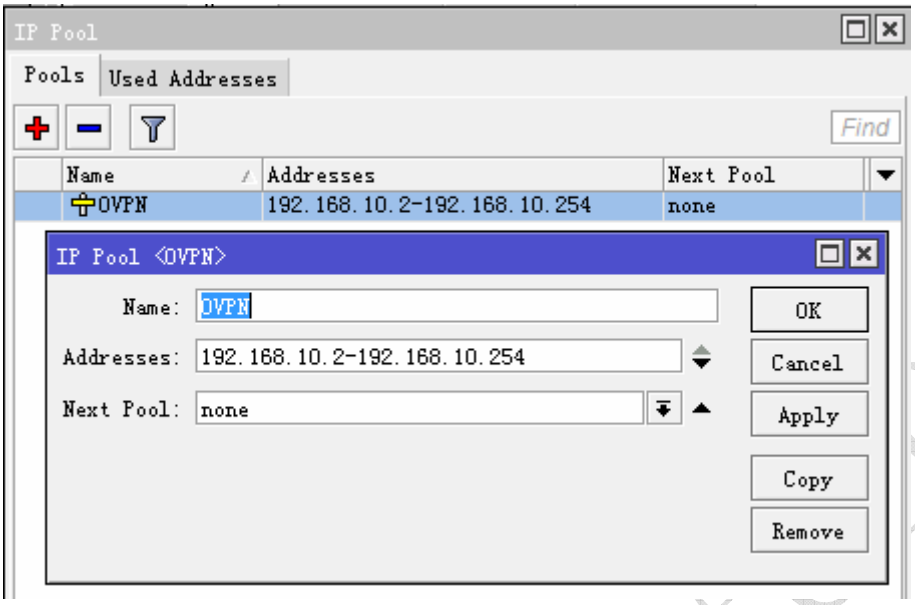


配置完成如下：

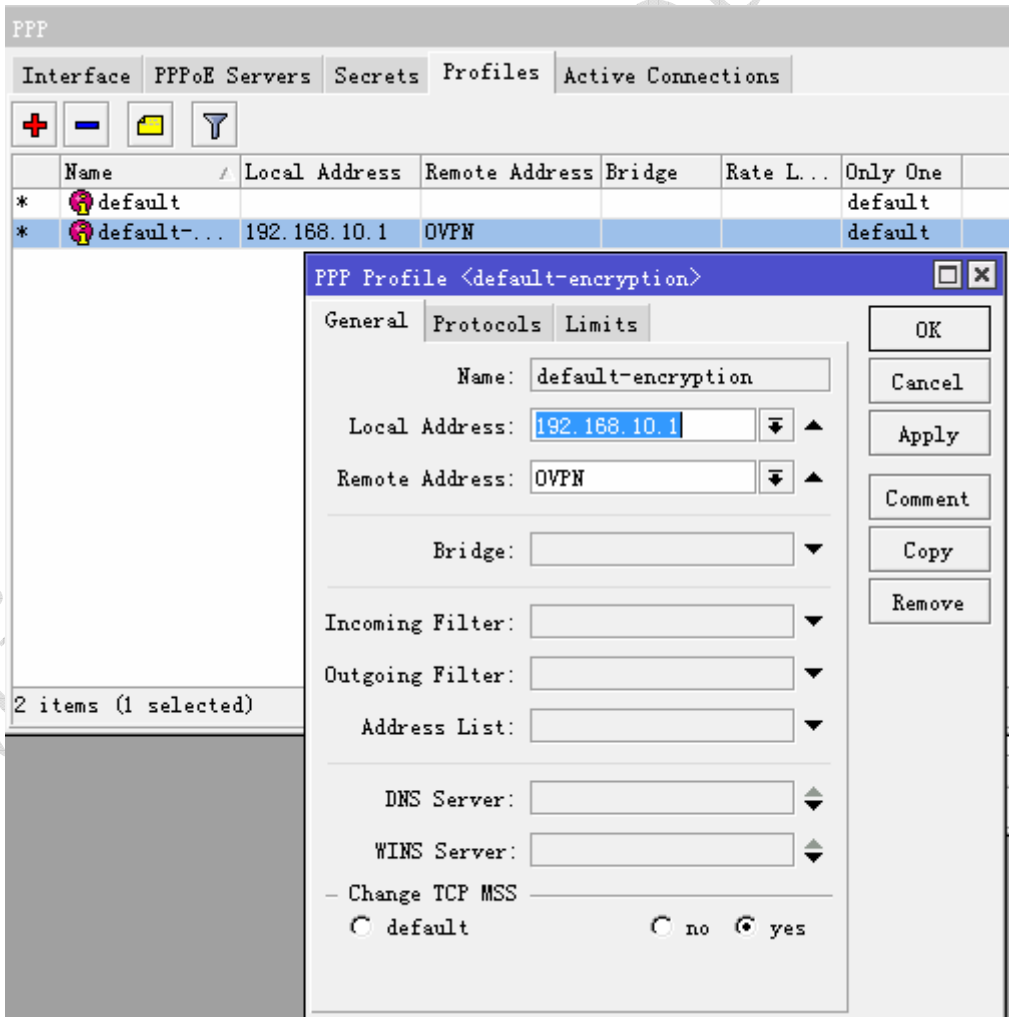


注意：require-client-certificate 这里如果选择上了，客户端同样也要导入相同的证书

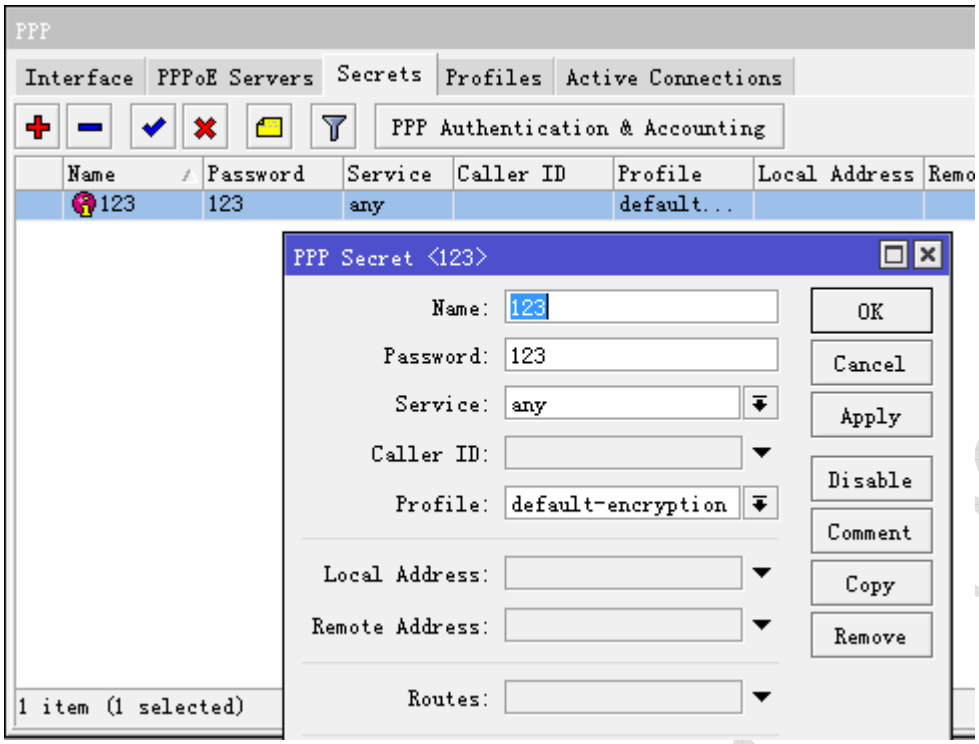
OVPN 服务器配置完成后，配置相应的规则，需要配置地址池 192.168.10.2-192.168.10.254，用于分配给用户的 IP 地址



进入 ppp profile 设置规则 local-address=192.168.10.1 和 remote-address=OVPN

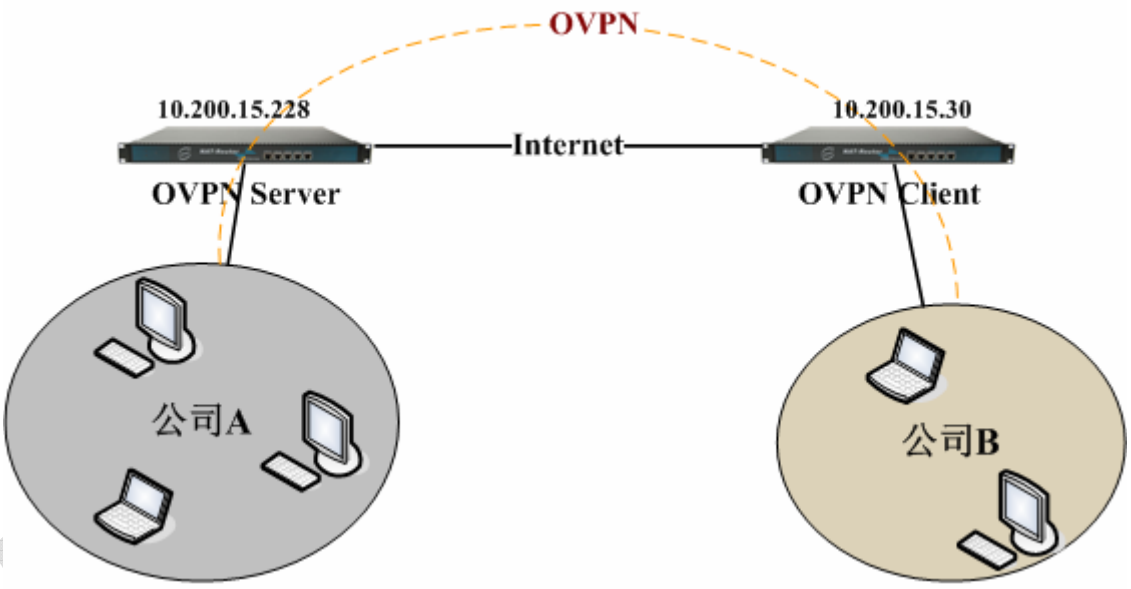


进入/ppp secret 添加用户账号



到此 OVPN 服务器配置完成

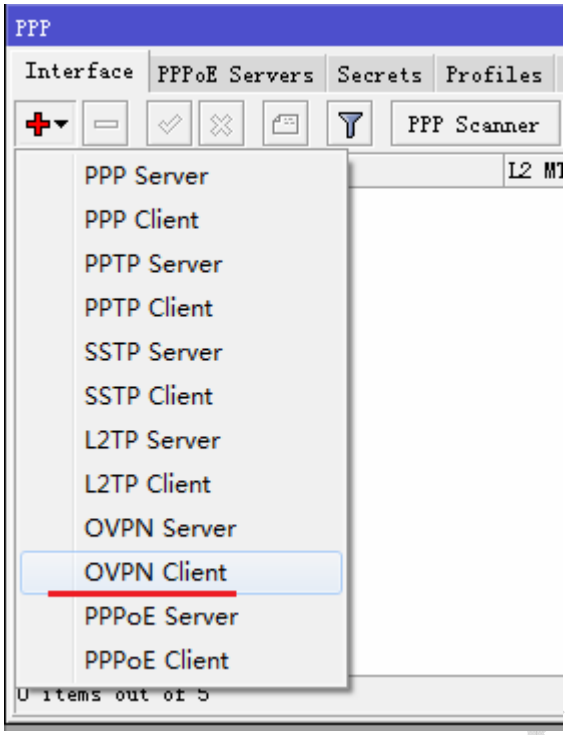
接下来我们需要在另外一台路由器配置 OVPN 客户端，如下面的拓扑图，将 2 台路由建立 OVPN 连接：



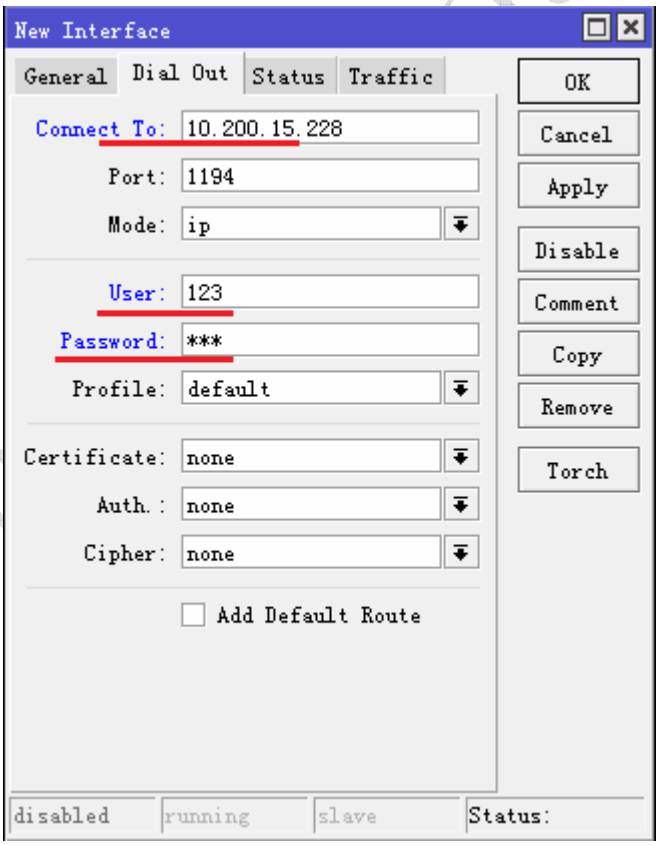
根据以上拓扑图我们假设以下网络环境：

OVPN Server 的 WAN 地址是 10.200.15.228，
OVPN Client 的 WAN 地址是 10.200.15.30

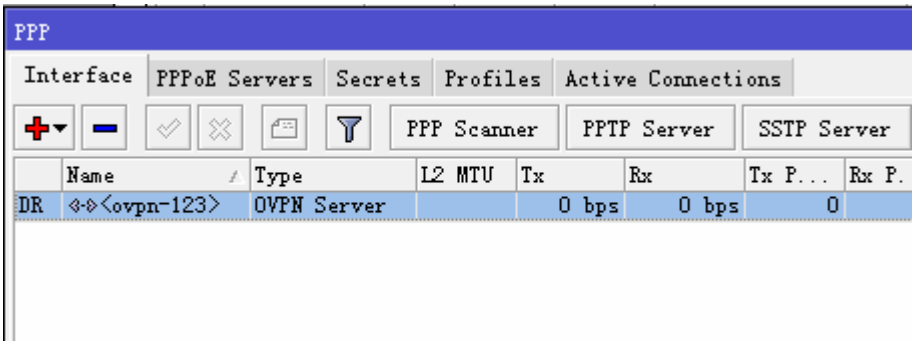
通过这两个 WAN 地址建立 OVPN 的互联，我们在 OVPN Client 建立 ovpn-client 拨号，打开 PPP， 在 interface 里点加号，选择 ovpn-client



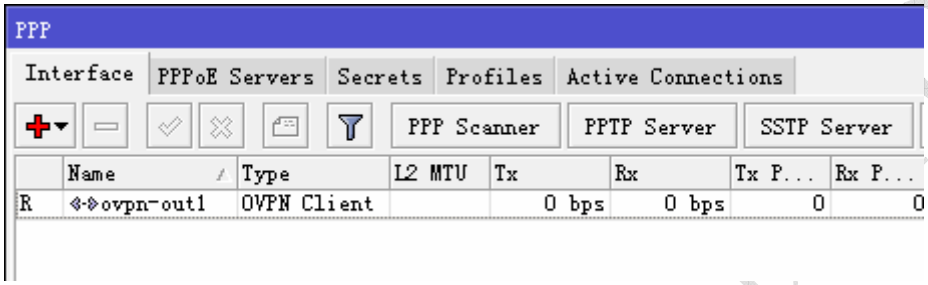
打开后，选择 Dial-out，设置 Connect-to=10.200.15.228，user=123，password=123，其他参数默认



确定后，连接 OVPN 服务器，服务器连接状态如下，显示 DR 前缀

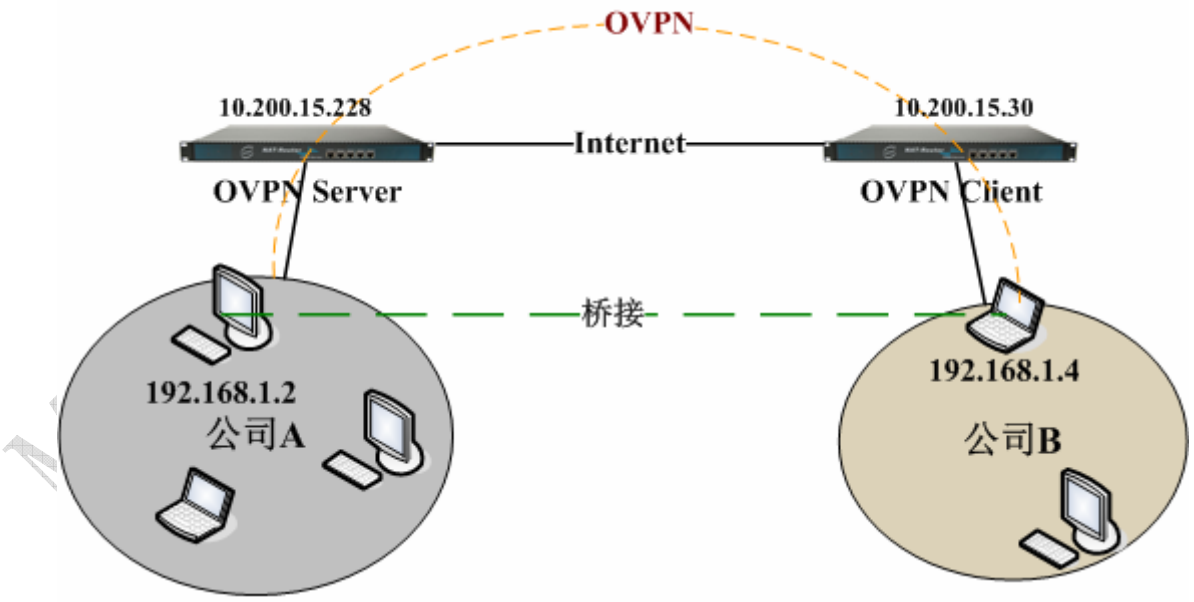


OVPN Client 显示前缀 R，表示连接成功

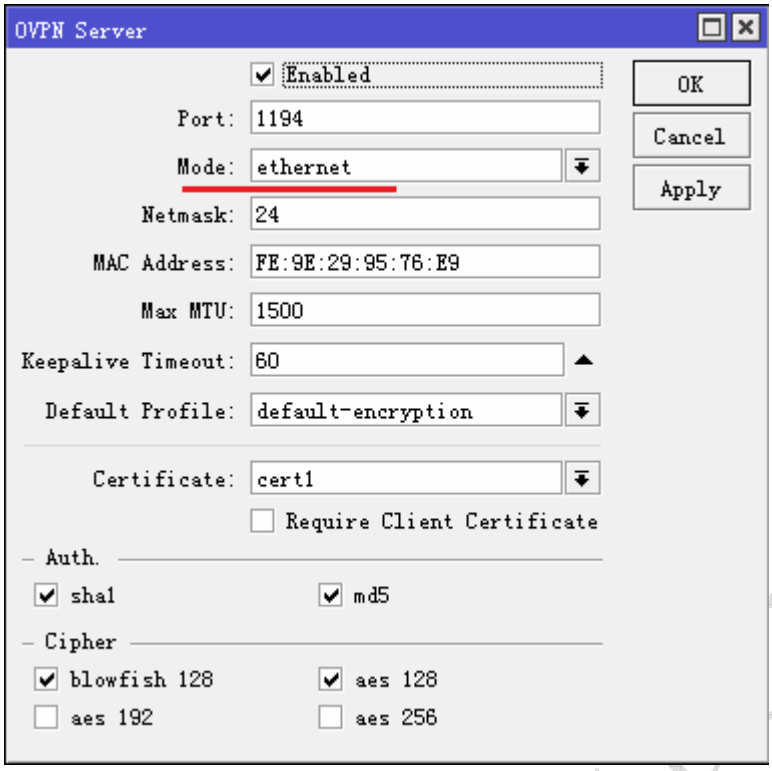


21.2 OVPN bridge 模式

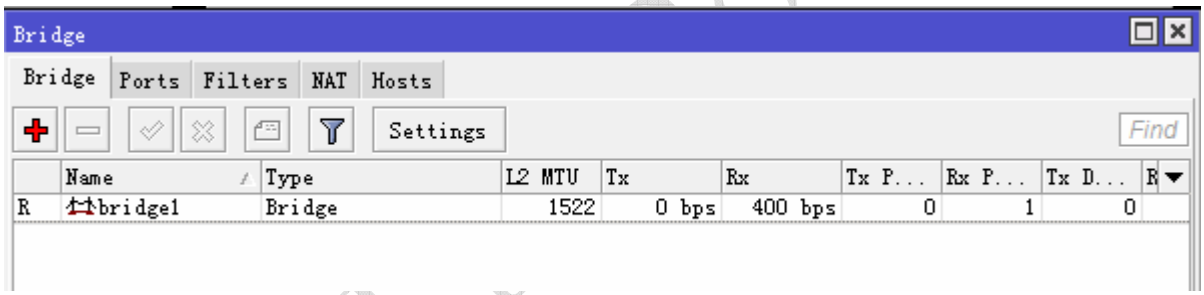
通过 RouterOS 提供的隧道 bridge 功能和 OVPN 的 ethernet 模式实现将两个远程网络建立二层的隧道透传连接，我们依然用之前的网络结构，如下图：



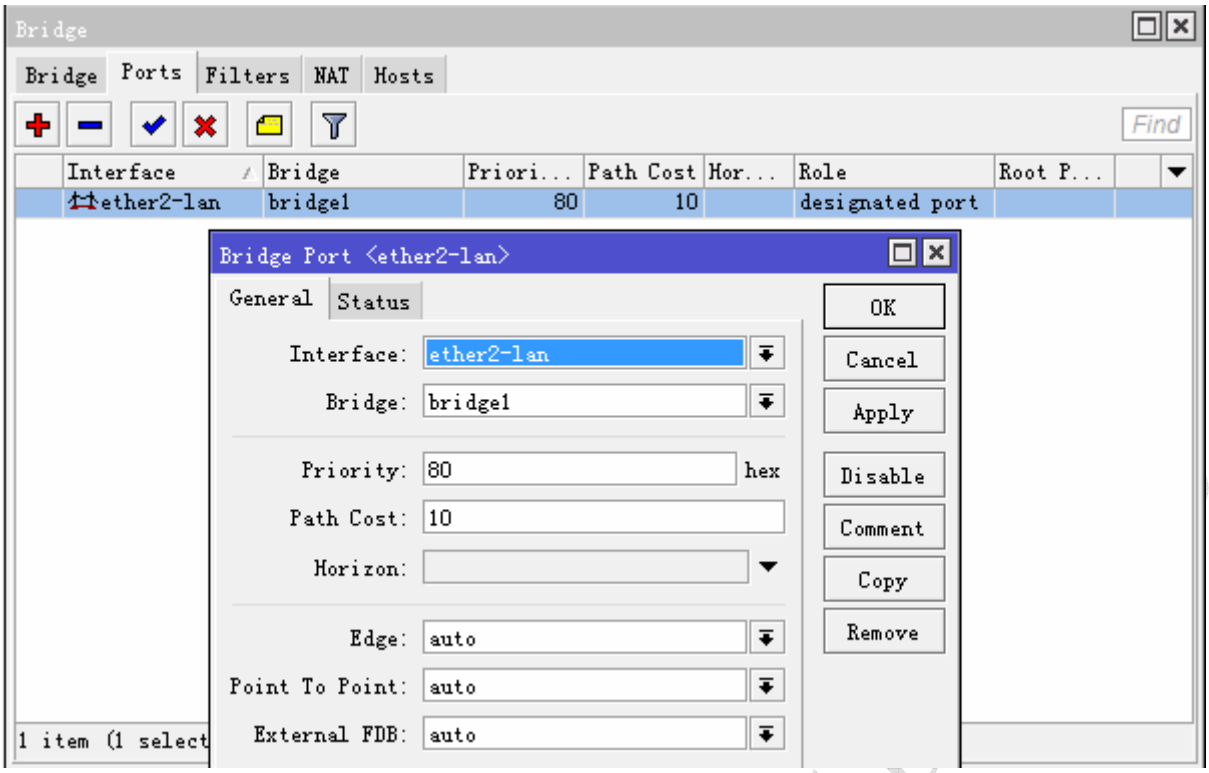
这里我们需要将 OVPN Server 和 OVPN Client 的模式修改为 ethernet



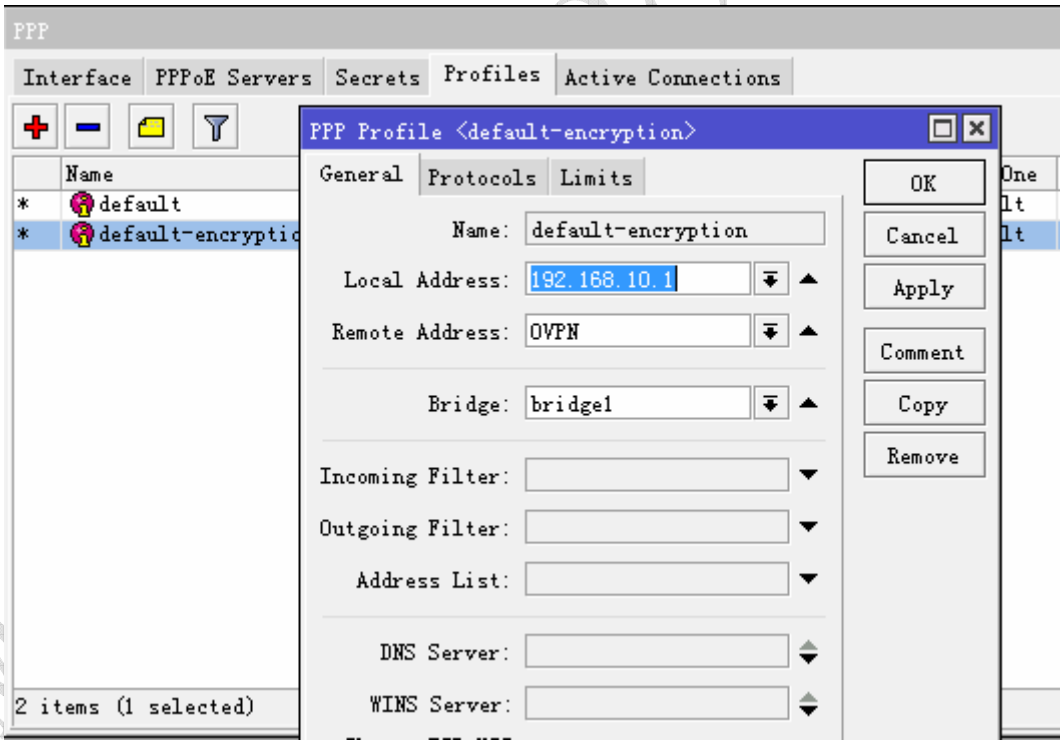
在 bridge 中添加一个桥配置



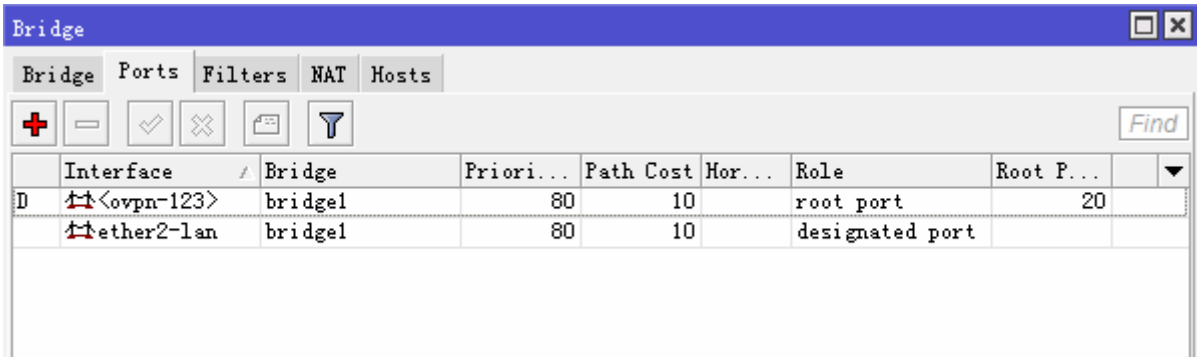
进入 ports 选择，将 ether2-lan 口添加入桥里



回到 ppp profile 里设置 default-encryption 的 bridge 选项设置为刚才我们添加的 bridge1



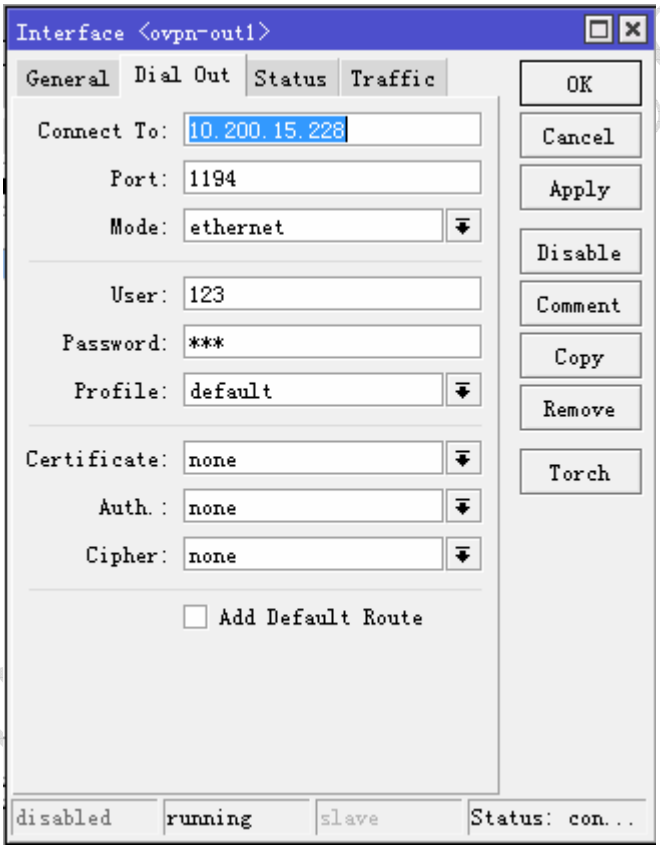
设置完成后，我们建立 OVPN Client 与 OVPN Server 的连接，在 bridge 的 port 里，我们可以看到 123 账号的连接在 port 被自动添加



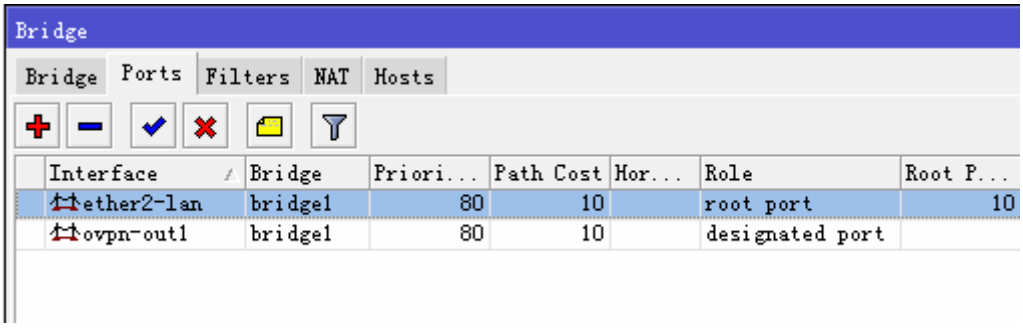
这样 OVPN Server 的 ether2-lan 内网口和 ovpn-123 隧道在同一个桥里

OVPN Client 设置

修改 ovpn-out1 的 mode 为 ethernet 模式



在 OVPN Client 端同样配置桥接，进入 bridge 后添加一个桥取名为 bridge1，并在 port 里将内网网卡 ether2-lan 和， ovpn-out1 接口手动添加到 bridge1 里，如下图



这样的配置，将两个远程的网络桥接在一个广播域内，即大家都在一个局域网内，这样的方式相对于 EoIP 隧道更安全，但需要考虑到数据加密会产生一定的开销

注意：在做 nat 伪装规则时，需要避免桥接隧道被转换，需要设置 nat 规则（包括 Server 和 Client 路由器的 nat 规则）

```
/ip firewall nat add out-interface=ether1-wan action=masquerade
```

第二十二章 SSTP 介绍

Secure Socket Tunneling Protocol (SSTP)方式是基于 SSL3.0 通道传输 PPP 隧道，使用 TCP 端口 443 的 SSL 允许 SSTP 通过所有防火墙和代理服务器

新的 SSTP 协议的支持，并没有完全否决 PPTP 及 L2TP 在微软产品所组成的解决方案中的作用，当企业使用基于 PPTP 和 L2TP 的 VPN 解决方案时，这种协议仍是被常用来解决或是提升企业网络安全性。但两者的数据包通过防火墙、NAT、WEB PROXY 时却都有可能发生一些连线方面的问题。

PPTP 数据包通过防火墙时，防火墙需被设定成同时充许 TCP 连接以及 GRE 封装的数据通过，但大部分 ISP 都会阻止这种封包，从而造成连线的问题;而当你的机器位于 NAT 之后，NAT 亦必需被设定成能转发 GRE 协议封装的数据包。否则就会造成只能建立 PPTP 的 TCP 连接，而无法接收 GRE 协议封装的数据包;WEB PROXY 是不支持 PPTP 协议的。

L2TP OVER IPSEC 的情况和此类似，需要在防火墙上充许 IKE 数据和 ESP 封装的数据同时通过，否则也会出现连接问题。且 WEB PROXY 也是不支持 L2TP OVER IPSEC 协议。

因此 SSTP 在透传方面由于前两种 VPN，新的 VPN 隧道工作在 SSL3.0 通道，因此可以绕过任何的过滤器和代理，因为 SSTP 工作在 TCP 的 443 端口，任何过滤器都会看作正常的传输并通过。

SSTP 客户端

操作路径： /interface sstp-client

add-default-route (*yes* | *no*; 默认: **no**)是否添加 SSTP 远程地址的默认路由
authentication (*mschap2* | *mschap1* | *chap* | *pap*; 默认: **mschap2, mschap1, chap, pap**)启用验证方试
certificate (*字符* | *none*; 默认: none) 证书
comment (*字符*; 默认: “”)注释
connect-to (*IP:Port*; 默认: **0.0.0.0:443**)远端 SSTP 服务器地址
dial-on-demand (*yes* | *no*; 默认: **no**)根据需求拨号
disabled (*yes* | *no*; 默认: **yes**)是否禁用该接口，默认是被禁用
keepalive-timeout (整型 | *禁用*; 默认: **60**)
max-mru (*整型*; 默认: **1500**)最大接收单位
max-mtu (整型; 默认: **1500**)最大传输单位
mrru (*disabled* | *整型*; 默认: **disabled**)在连接被认可的最大数据包长度。如果一个数据大于隧道的 MTU 值，将会被拆分多个数据包，允许最大长度的 IP 或者以太网数据包发送到隧道
name (*字符*; 默认:)说明接口名称
password (字符; 默认: *""*)用于验证的密码
profile (*name*; Default: **default-encryption**)被使用的 PPP profile
proxy (*IP:Port*; 默认: **0.0.0.0:443**)HTTP 代理服务的地址和端口

user（字符；默认：“”）用于验证的账号名

这个事例示范如何设置 SSTP 客户端，连接服务器 10.1.101.1，用户名“sstp-test”，密码“123”

```
[admin@MikroTik]  /interface sstp-client>add user=sstp-test password=123 \
\... connect-to=10.1.101.1 disabled=no
[admin@MikroTik] /interface sstp-client> print
Flags: X - disabled, R - running
 0  R name="sstp-out1" max-mtu=1500 max-mru=1500 mrru=disabled connect-to=10.1.101.1:443
     user="sstp-test" password="123" proxy=0.0.0.0:443 profile=default
     certificate=none keepalive-timeout=60 add-default-route=no dial-on-demand=no
     authentication=pap,chap,mschap1,mschap2
```

SSTP 服务器

操作路径: /interface sstp-server

这路径显示接口为每个已连接的 SSTP 客户端 ，创建一个接口是为建立到指定服务器的每条隧道，这里在 PPTP 服务器的配置有两种类型

- 静态接口是如果需要一个固定参照详细接口名（如防火墙规则或者其他任何规则）被管理而创建一个特别用户接口
- 动态接口是，无论任何时候当一个用户连接，且用户名没有匹配任何静态接口（或者这个用户账号已经登录，同时不能有 2 个独立相同的隧道出现），这时接口将会被自动添加到列表

当一个用户连接动态接口会出现，用户退出或断开后会消失，因此如果你要为用户提供固定的规则，需要创建一个静态接口，否则建立动态配置

注：在两个状态的 PPP 用户必须配置正确，静态接口不能替代 PPP 基本配置参数和规则

服务器配置

操作路径: /interface sstp-server server

authentication（*pap* | *chap* | *mschap1* | *mschap2*; 默认：pap,chap,mschap1,mschap2）服务器将使用的验证模式
certificate（*名称*; 默认：none)SSTP 将使用的证书的名称。必须使用证书，否则 SSTP 服务器将不能运行
default-profile（*名称*; 默认：default)选择 Profile 规则
enabled（*yes* | *no*; 默认：no)定义是否启用 SSTP 服务
keepalive-timeout（整型 | *disabled*; 默认：60)定义路由发送 Keepalive 数据包时钟周期（以秒为单位），如果没有数据，并且没有在指定的时钟周期回应，这些没有回应的用户将会被注销
max-mru（*整型*; 默认：1500)最大接收单位，
max-mtu（*整型*; 默认：1500)最大传输单位
mrru(*禁用* | *整型*; 默认：disabled)在连接被认可的最大数据包长度。如果一个数据大于隧道的 MTU 值，将会被拆分多个数据包，允许最大长度的 IP 或者以太网数据包发送到隧道
require-client-certificate（*yes* | *no*; 默认：no）如果设置为 yes，这时服务器将检查客户端的证书是否属于与服务器的证书同一证书链

启用 SSTP 服务器你需要导入证书，之后你可以配置服务器

```
[admin@MikroTik] /interface sstp-server server> set certificate=server
[admin@MikroTik] /interface sstp-server server> set enabled=yes
[admin@MikroTik] /interface sstp-server server> print
                 enabled: yes
                    port: 443
                 max-mtu: 1500
                 max-mru: 1500
                    mrru: disabled
         keepalive-timeout: 60
           default-profile: default
              certificate: server
 require-client-certificate: no
           authentication: pap,chap,mschap1,mschap2
[admin@MikroTik] /interface sstp-server server>
```

Monitor 命令能查看在客户端和服务器的隧道运行状态：

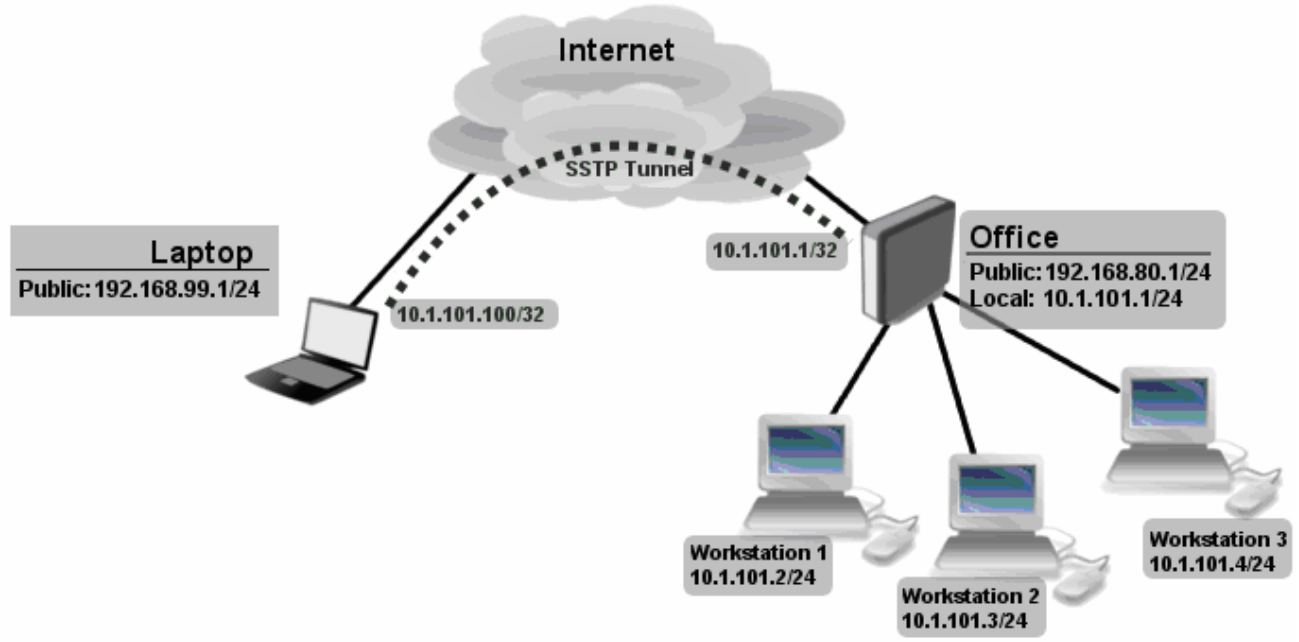
```
[admin@dzeltenais_burkaans] /interface sstp-server> monitor 0
    status: "connected"
    uptime: 17m47s
 idle-time: 17m47s
      user: "sstp-test"
 caller-id: "10.1.101.18:43886"
       mtu: 1500
```

只读属性

status ()当前 SSTP 状态，值显示为“connected”表明隧道连接已经建立
uptime (*时间*)从隧道建立开始经过的时间
idle-time (*时间*)在隧道上最后一次活动经过的时间
user (*字符*)隧道建立的用户名称
mtu (*整型*)使用的 MTU
caller-id (*IP:ID*)连接者的身份，一般以 IP 地址显示

22.1 端到服务器连接

下面事例显示了如何使用 SSTP 加密隧道通过一个电脑连接到远端办公网络，SSTP 服务器分配给电脑一个相同远程办公网络的 IP 地址(不需要通过桥接的 EoIP 隧道)



Office 路由通过 ether1 连接到互联网，内网主机连接到 ether2，笔记本电脑通过互联网连接到 office 路由的公网 IP 地址 (在我们的事例中 IP 地址为 192.168.80.1).

注：在你开始配置 SSTP 前，你需要建立服务器证书，并导入路由器（证书可以通过 windows2008 生成，具体操作可以通过网上查询，证书导入方式和 OVPN 一样）

第一步创建一个用户

```
[admin@RemoteOffice] ppp secret> add name=Laptop service=sstp password=123
local-address=10.1.101.1 remote-address=10.1.101.100
[admin@RemoteOffice] ppp secret> print detail
Flags: X - disabled
 0   name="Laptop" service=sstp caller-id="" password="123" profile=default
     local-address=10.1.101.1 remote-address=10.1.101.100 routes==""

[admin@RemoteOffice] ppp secret>
```

注意 SSTP 的 local-address 与路由器本地网卡地址相同，并且 remote-address 与本地内网属同一地址段 (10.1.101.0/24).

下一步，启用 SSTP 服务器，并在笔记本电脑建立客户端

```
[admin@RemoteOffice] /interface sstp-server server> set certificate=server
[admin@RemoteOffice] /interface sstp-server server> set enabled=yes
[admin@RemoteOffice] /interface sstp-server server> print
                 enabled: yes
                    port: 443
                 max-mtu: 1500
                 max-mru: 1500
                    mrru: disabled
         keepalive-timeout: 60
           default-profile: default
```

```
              certificate: server
 require-client-certificate: no
           authentication: pap,chap,mschap1,mschap2

[admin@RemoteOffice] /interface sstp-server server>
```

SSTP 客户端连接到路由器的公网 IP 地址，这个事例 IP 地址是 192.168.80.1.

注意，根据你的操作系统不同配置你的 SSTP 客户端，当前支持 SSTP 的系统有 windows2008,windows vista 和 vista sp1，其他操作系统不一定支持

检查 SSTP 客户端是否连接

```
[admin@RemoteOffice] /interface sstp-server> print
Flags: X - disabled, D - dynamic, R - running
 #    NAME     USER        MTU       CLIENT-ADDRESS    UPTIME   ENCODING
 0  DR <sstp-... Laptop    1500      10.1.101.18:43886 1h47s

[admin@RemoteOffice] /interface sstp-server>monitor 0
    status: "connected"
    uptime: 1h45s
 idle-time: 1h45s
      user: "Laptop"
 caller-id: "192.168.99.1:43886"
       mtu: 1500
```

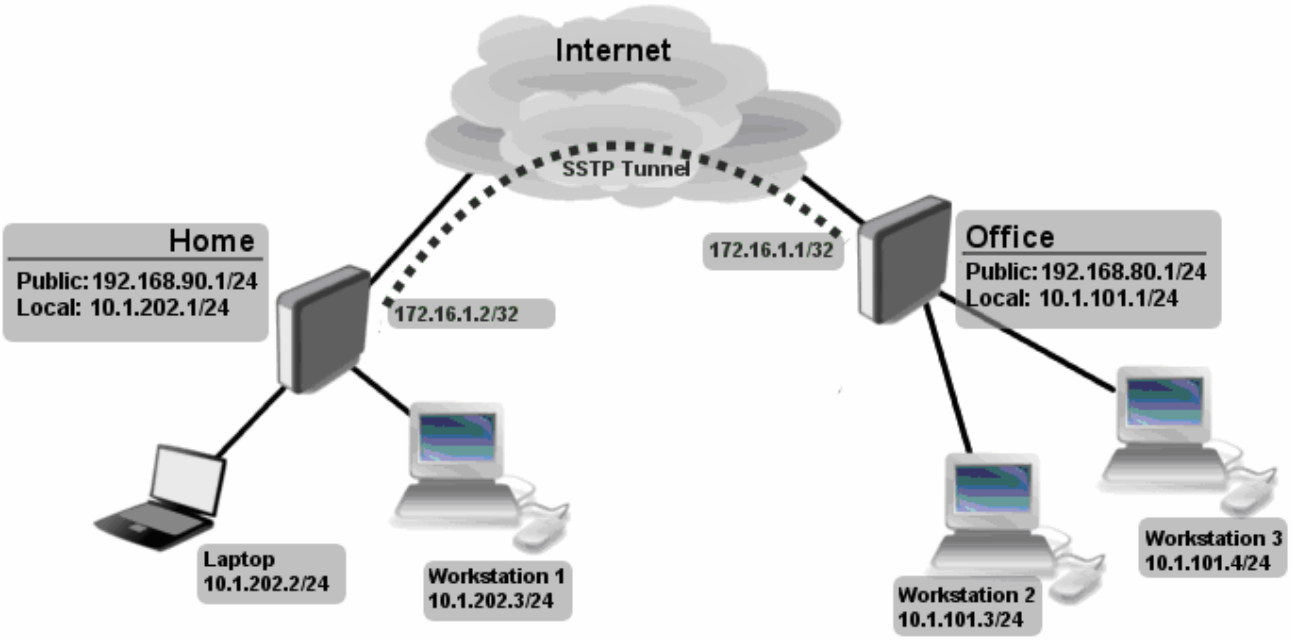
这里（当 SSTP 客户端连接成功），如果你想从笔记本电脑 ping 内网主机，将显示 timeout，因为笔记本电脑将不能从内网主机获取 ARP，解决方法是设置内网网卡的 arp 为 proxy-arp

```
[admin@RemoteOffice] /interface ethernet> set ether2 arp=proxy-arp
[admin@RemoteOffice] /interface ethernet> print
Flags: X - disabled, R - running
 #    NAME              MTU   MAC-ADDRESS       ARP
 0  R ether1            1500  00:30:4F:0B:7B:C1 enabled
 1  R ether2            1500  00:30:4F:06:62:12 proxy-arp
[admin@RemoteOffice] interface ethernet>
```

在 proxy-arp 启用后，将能成功 ping 通过内网主机

22.2 点对点的 SSTP

下面一个事例两个企业内网采用基于 SSTP 隧道连接，考虑下面的网络：



Office 和 Home 路由器连接互联网通过 ether1，内网主机连接到 ether2，两个内网他们不在同一广播域，如果两个内网需要在同样广播域内，你需要使用 BCP，并桥接 SSTP 隧道与本地网卡

首先创建一个用户账号

```
[admin@RemoteOffice] /ppp secret> add name=Home service=sstp password=123
local-address=172.16.1.1 remote-address=172.16.1.2 routes="10.1.202.0/24 172.16.1.2 1"
[admin@RemoteOffice] ppp secret> print detail
Flags: X - disabled
 0   name="Home" service=sstp caller-id="" password="123" profile=default
     local-address=172.16.1.1 remote-address=172.16.1.2 routes=="10.1.101.0/24 172.16.1.1
1"

[admin@RemoteOffice] /ppp secret>
```

注意：无论什么时候我们设置 SSTP 客户端添加路由，如果这个选项没有设置，你将需要指定静态路由配置在两个路由器上

接下来启用 SSTP 服务器在 Office 路由器，并在 Home 路由器配置 SSTP 客户端

```
[admin@RemoteOffice] /interface sstp-server server> set certificate=server
[admin@RemoteOffice] /interface sstp-server server> set enabled=yes
[admin@RemoteOffice] /interface sstp-server server> print
                 enabled: yes
                    port: 443
                 max-mtu: 1500
                 max-mru: 1500
                    mrru: disabled
         keepalive-timeout: 60
           default-profile: default
              certificate: server
 require-client-certificate: no
```

```
            authentication: pap,chap,mschap1,mschap2
```

在 Home 路由器配置 SSTP 客户端

```
[admin@Home] /interface sstp-client> add user=Home password=123 connect-to=192.168.80.1
disabled=no
[admin@Home] /interface sstp-client> print
Flags: X - disabled, R - running
 0  R name="sstp-out1" max-mtu=1500 max-mru=1500 mrru=disabled connect-to=192.168.80.1:443
      user="Home" password="123" proxy=0.0.0.0:443 profile=default certificate=none
      keepalive-timeout=60 add-default-route=no dial-on-demand=no
      authentication=pap,chap,mschap1,mschap2
[admin@Home] /interface sstp-client>
```

现在我们我们需要添加静态路由在 Home 路由器上，用于查找在 Office 路由后的网络

```
[admin@Home] /ip route> add dst-address=10.1.101.0/24 gateway=172.16.1.1
```

在隧道建立后你可以 ping 通远端的网络。

第二十三章 EoIP 隧道

EoIP（Ethernet over IP）隧道是一个建立在两个路由器的 IP 传输层之间的以太网隧道协议，是 MikroTik RouterOS 的自由协议。EoIP 接口表现的类似以太网传输，当路由器的桥接功能被启用后，所有的以太网数据流量（所有的以太网协议）将被桥接就如同在两个路由器（启用了桥接功能）之间有物理交换机接口和光纤收发器一样。

有 EoIP 接口的网络设置：

- 可以在因特网上桥接 LAN
- 可以在加密的隧道桥接 LAN
- 可以在 802.11b 'ad-hoc'无线网络上桥接 LAN

快速设置向导

在 IP 地址为 **10.5.8.1** 和 **10.1.0.1** 的两个路由器之间做 EoIP 隧道：

1. 在 IP 地址为 **10.5.8.1** 的路由器上添加一个 EoIP 接口并设置它的 MAC 地址：

```
/interface eoip add remote-address=10.1.0.1 tunnel-id=1 mac-address=00-00-5E-80-00-01
disabled=no
```

2. 在 IP 地址为 **10.1.0.1** 的路由器上添加一个 EoIP 接口并设置它的 MAC 地址：

```
/interface eoip add remote-address=10.5.8.1 tunnel-id=1 mac-address=00-00-5E-80-00-02
disabled=no
```

现在你可以从同一子网添加 IP 地址以创建 EoIP 接口。

规格

功能包要求: ***system***
等级要求: *Level3*
操作路径: ***/interface eoip***

EoIP 接口应用在有 IP 层连接，EoIP 通道可以在 IPIP 隧道，PPTP 128bit 加密隧道，PPPoE 连接或任何传输 IP 的连接上运行。具体属性：

- 每个上运行隧道接口可以与一个有相同“隧道 ID”的相应接口配置的远程路由器相连接
- EoIP 接口就好象接口列表下的以太网接口一样
- 这个接口支持以太网接口的所有特征。IP 地址及其他隧道可以在这个接口上运行
- EoIP 协议封装以太网帧在 GRE（IP 协议号 47）数据包中，并把它们发送到 EoIP 隧道的远程端
- EoIP 隧道的最大计数为 65536

注： WDS 在很大程度上比 EoIP 快（最多达可达到 10-20%，在 RouterBOARD 500 系统上），所以推荐在可能时使用 WDS。

23.1 EoIP 配置

操作路径: /interface eoip
arp (disabled | enabled | proxy-arp | reply-only; default: **enabled**) -地址解析协议
mac-address (*MAC 地址*) - EoIP 接口的 MAC 地址。你可以自由的使用从 **00-00-5E-80-00-00** 到 **00-00-5E-FF-FF-FF** 范围的 MAC 地址
mtu (*整型*; default: **1500**) -最大传输单元。默认值提供了最大的兼容性
name (*名称*; 默认: **eoip-tunnelN**) – 隧道接口名
remote-address - EoIP 隧道 IP 地址的另一端——必须是 MikroTik 路由器
tunnel-id (*整型*) – 隧道身份值

注： **tunnel-id** 是一种识别隧道的方法。在同一个路由器上不应该有相同 **tunnel-id** 的隧道。在参与的两个路由器的 **tunnel-id** 必须是平等的。

mtu 必须设置为 1500 以消除隧道内的数据包分段存储（它允许类似以太网络的透明桥接，因此有可能在隧道上传输满长度的以太网帧）。

当桥接 EoIP 隧道时，推荐对每个隧道设置唯一的 MAC 地址以使桥接算法正常工作。对于 EoIP 接口你可以使用从 **00-00-5E-80-00-00** 到 **00-00-5E-FF-FF-FF** 范围的 MAC 地址，IANA 就是为这些情况保留的。或者，你可以设置第一字节的第二位来标记地址为由网络管理员指定的本地管理的地址，并使用任何 MAC 地址，你只需要确定它们在连接到一个桥的主机之间是唯一的。

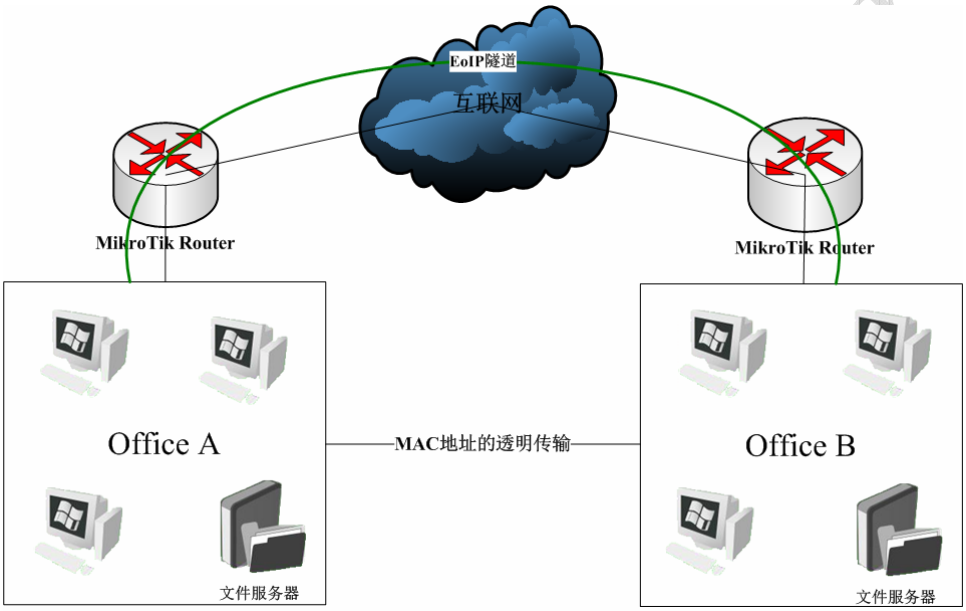
添加并启用名为 **to_mt2** 连接到 **10.5.8.1** 路由器的 EoIP 隧道，指定 **tunnel-id** 为 **1**：

```
[admin@MikroTik] interface eoip> add name=to_mt2 remote-address=10.5.8.1 \
\... tunnel-id 1
[admin@MikroTik] interface eoip> print
Flags: X - disabled, R - running
  0 X  name="to_mt2" mtu=1500 arp=enabled remote-address=10.5.8.1 tunnel-id=1
```

```
[admin@MikroTik] interface eoip> enable 0
[admin@MikroTik] interface eoip> print
Flags: X - disabled, R - running
  0 R  name="to_mt2" mtu=1500 arp=enabled remote-address=10.5.8.1 tunnel-id=1

[admin@MikroTik] interface eoip>
```

23.2 EoIP 应用实例

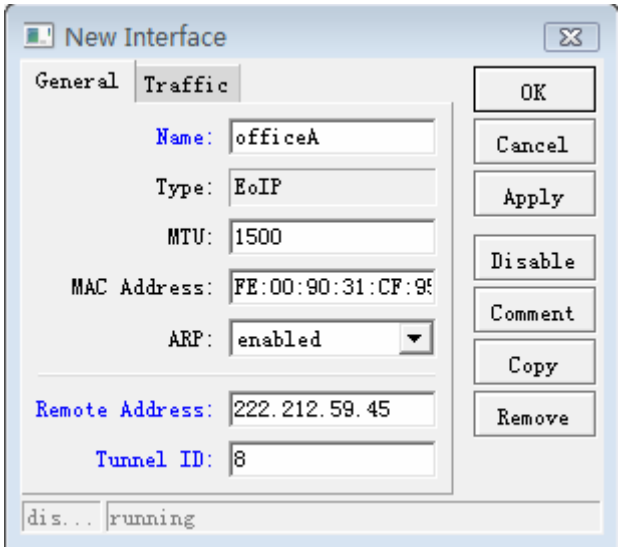
这里我们假设有两个异地的办公点，officeA 和 officeB，我们通过 Eoip 隧道将他们连接起来，建立 2 层的安全隧道通信，如下图：



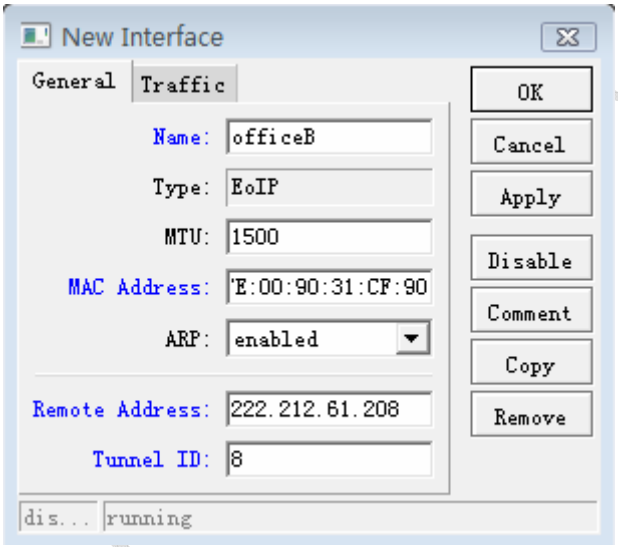
网络参数：

- OfficeA 的 IP 地址为 222.212.61.208，桥接分配地址 10.0.0.1
- OfficeB 的 IP 地址是 222.212.59.45，桥接分配地址 10.0.0.2

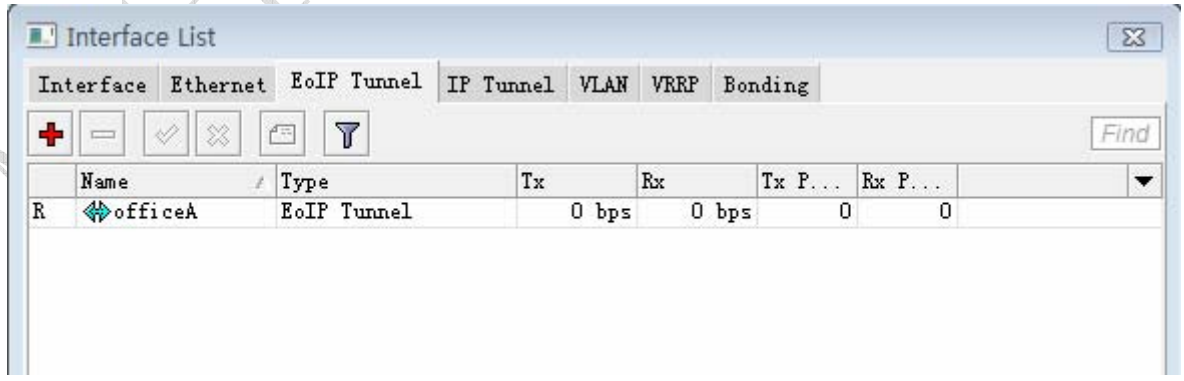
1、这里我们将配置两个 Eoip 隧道的 Tunnel ID 为 8，首先在 Interface 里建立 Eoip 隧道



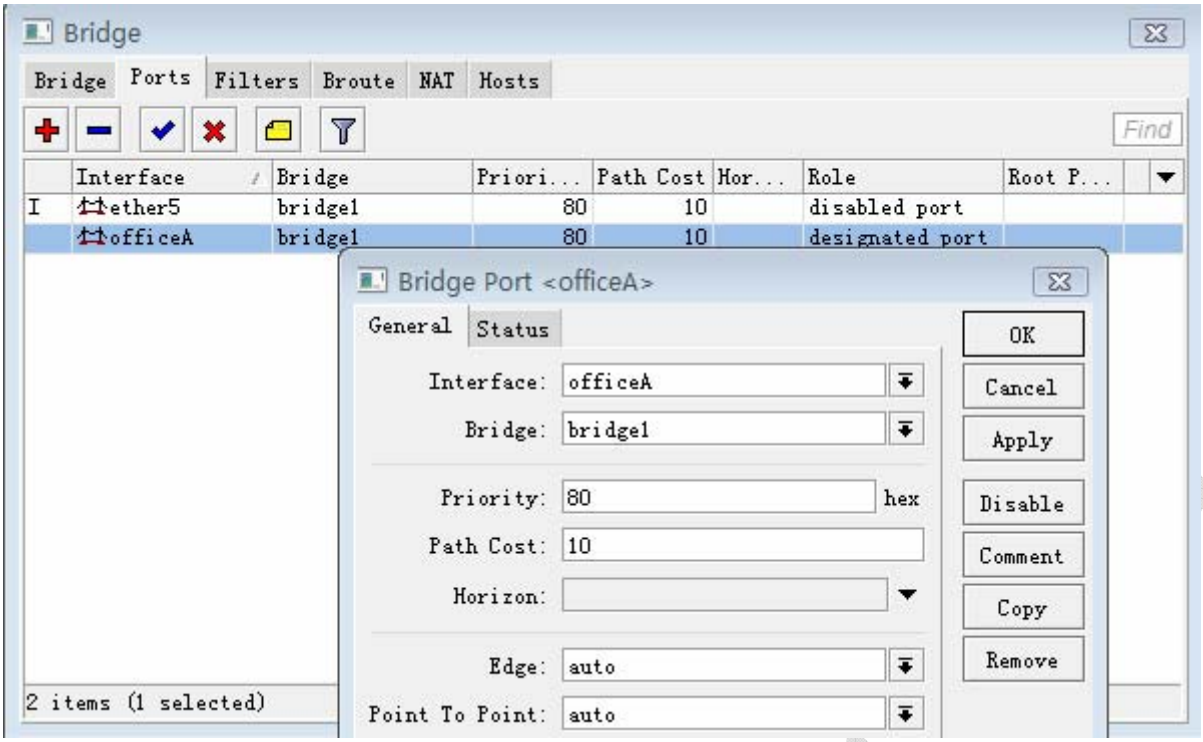
建立 officeB 的 Eoip 隧道：



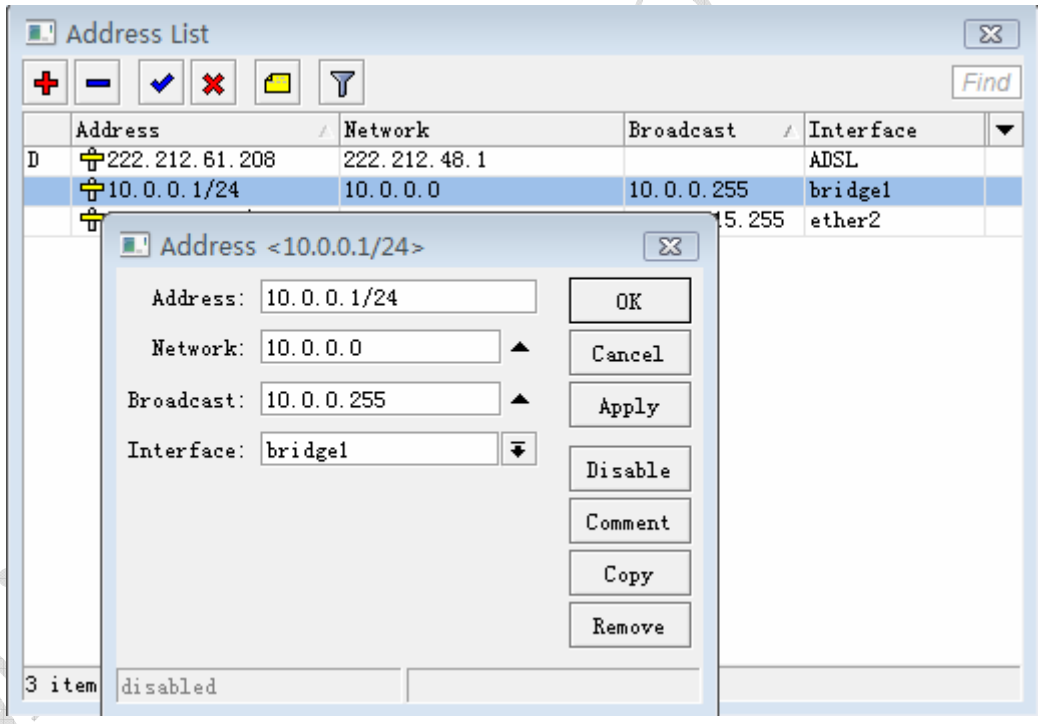
2、接下来，OfficeA 和 OfficeB 的配置基本相同，现在我们来看 officeA 配置，在 interface 中的 Eoip Tunnel 中可以看到 Eoip 隧道连接成功：



进入 bridge，并在 bridge 中添加一个 bridge1 的桥，然后并在 port 中将 ether5 网卡和建立 Eoip 隧道的 officeA 绑定到 Bridge1 中：



3、绑定完桥后，进入 ip address，设置桥 IP 地址为 10.0.0.1/24，同样在 OfficeB 的路由器则设置为 10.0.0.2/24：



注： 在做 NAT 规则的时候，特别是伪装，需要指明伪装的端口，如果默然伪装，将会把 EoIP 隧道隐藏，使其二层透穿出现问题，为了避免影响 EOIP 的连接，要选择 out-interface 为 WAN 口。

故障分析

- **路由器可以相互之间 ping 通但 EoIP 隧道依然不能正常工作！**

  检查 EoIP 接口的 MAC 地址——两台通信的路由器它们的 MAC 不应该一样！

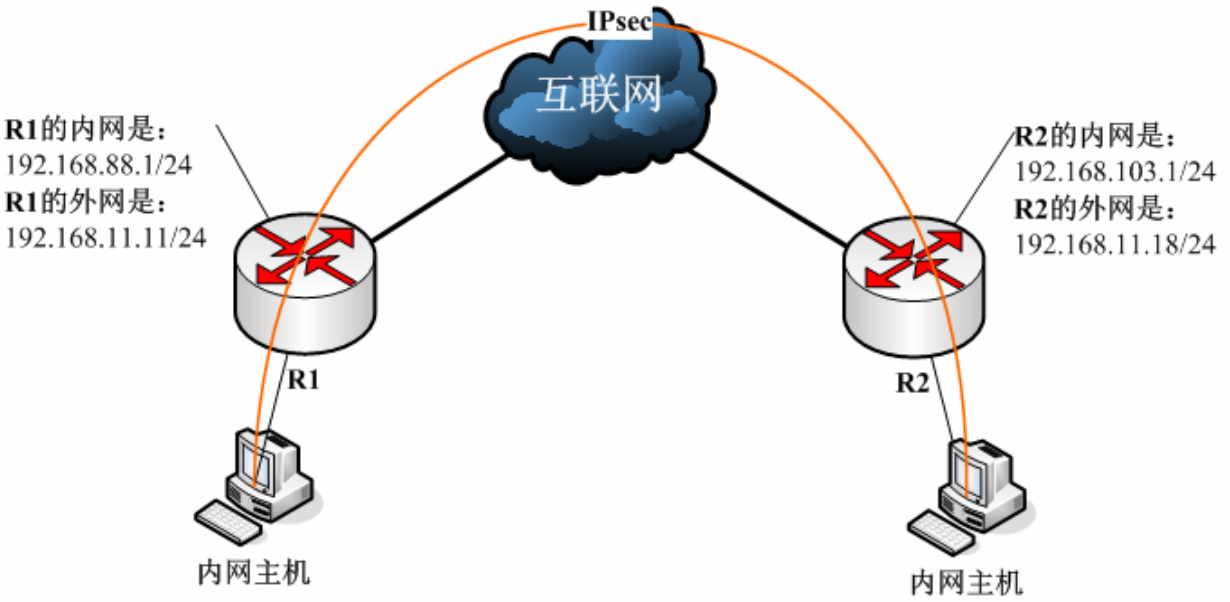
第二十四章 IPSec 配置

IPSec 作为新一代网络安全协议，为网络传输提供了安全保证，使端到端的数据保密成为可能，是互联网上的新一代安全标准。提供包括访问控制、无连接的完整性、数据源认证、抗重放 (replay)保护、保密和有限传输保密性在内的服务，服务基于 IP 层并对 IP 及上层协议进行保护。服务的实施通过两种通信安全协议：认证头（AH）和封装安全负载（ESP）以及 Internet 密钥交换（IKE）协议来达到这些目标。

IP AH 协议提供数据源认证、无连接的完整性和可选的抗重放服务。ESP 协议提供数据保密性，有限的数据流保密性、数据源认证、无连接的完整性及抗重放服务。IKE 协议用于协商 AH 和 ESP 协议所使用的密码算法，并将算法所需的必备密钥放在合适的位置。IPSec 有两种模式：传输模式和隧道模式。它们都是对外出的数据包添加 IPSec 头进行加密和认证，而对于接收的 IPSec 数据包作解密认证处理和适当的转发传送。

24.1 IPSec 配置实例

以下是一个使用 RouterOS 建立的 IPsec VPN 案例，网络拓扑图：

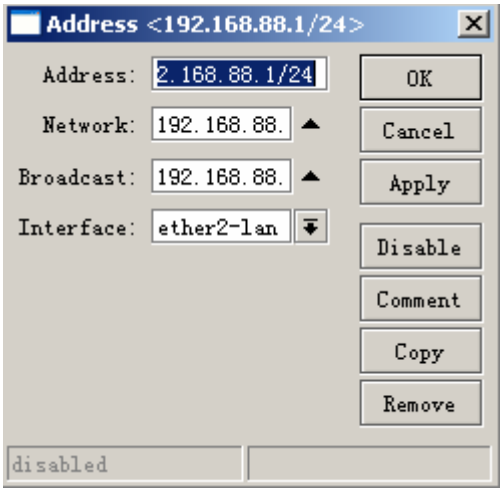


需要 IPsec VPN 互联的网络环境：

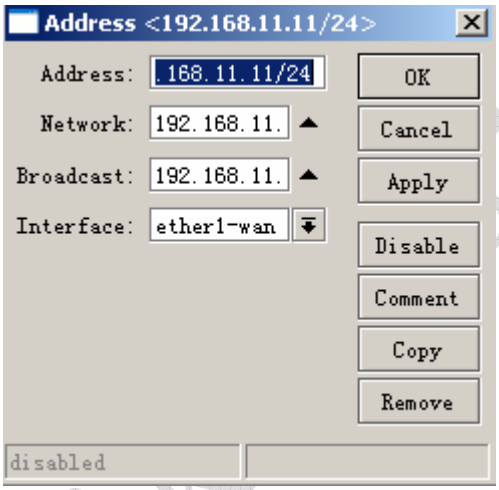
192.168.88.1/24--***R1***---192.168.11.11/24 **----互联网----** 192.168.11.18/24---***R2***---192.168.103.1/24

R1 配置

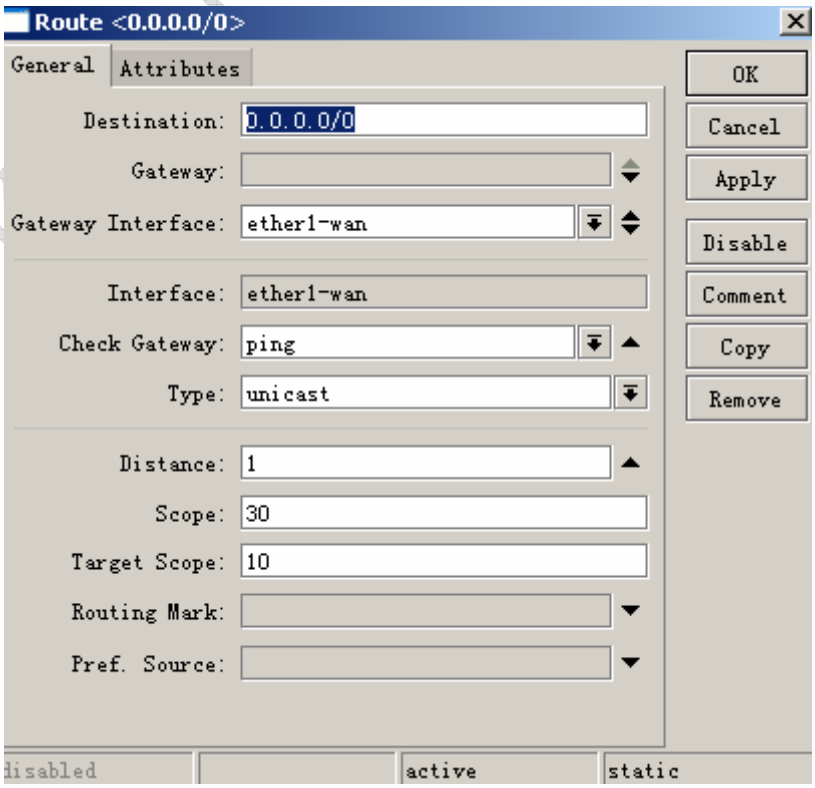
进入 ip address 里面添加内网接口地址：



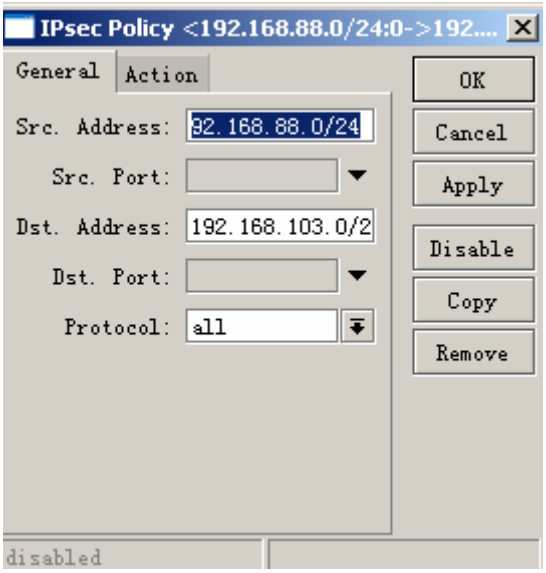
再添加外网接口地址：



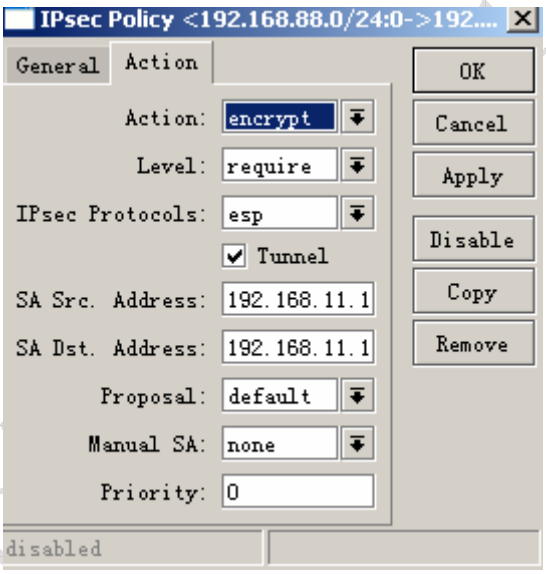
进入 ip routes 里面添加网关出口：



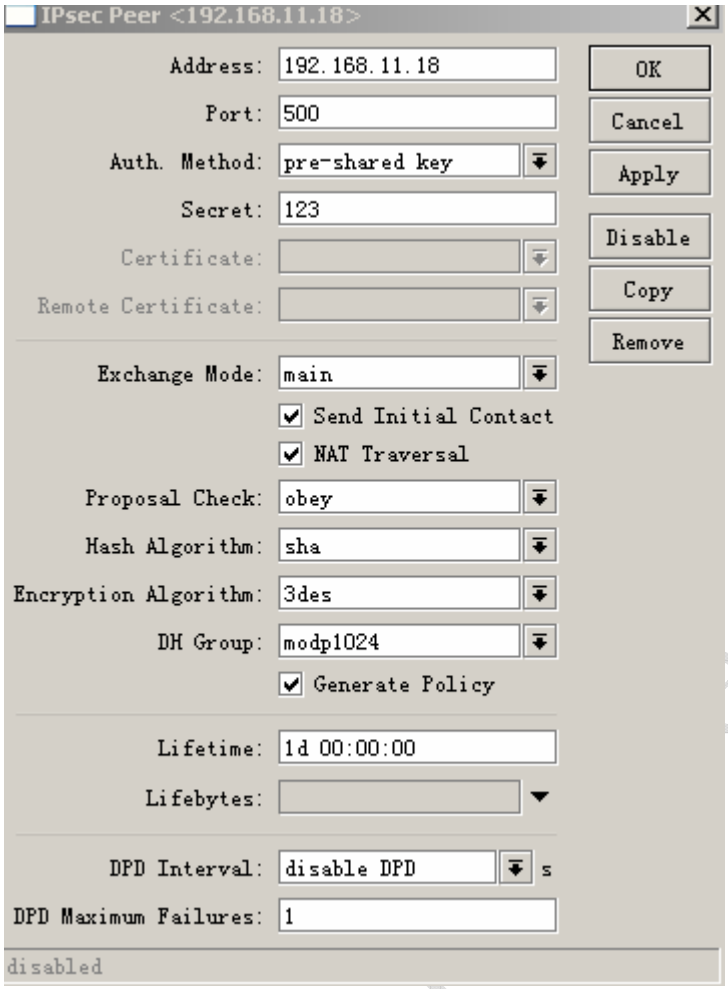
进入 ip ipsec 里面 policies 的 general 选项添加内网的源地址和需要做 ipsec 的对端内网地址。



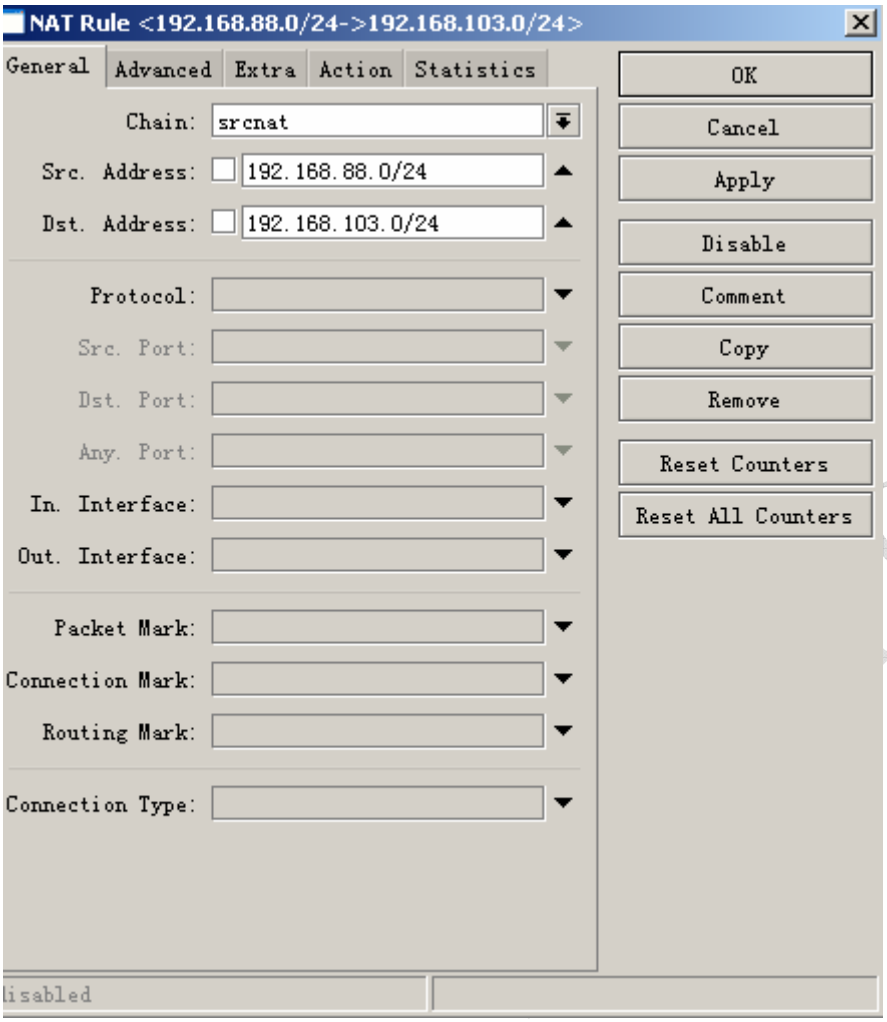
再在 action 选项里面添加源外网地址和对端外网地址和开启 tunnel 隧道协议



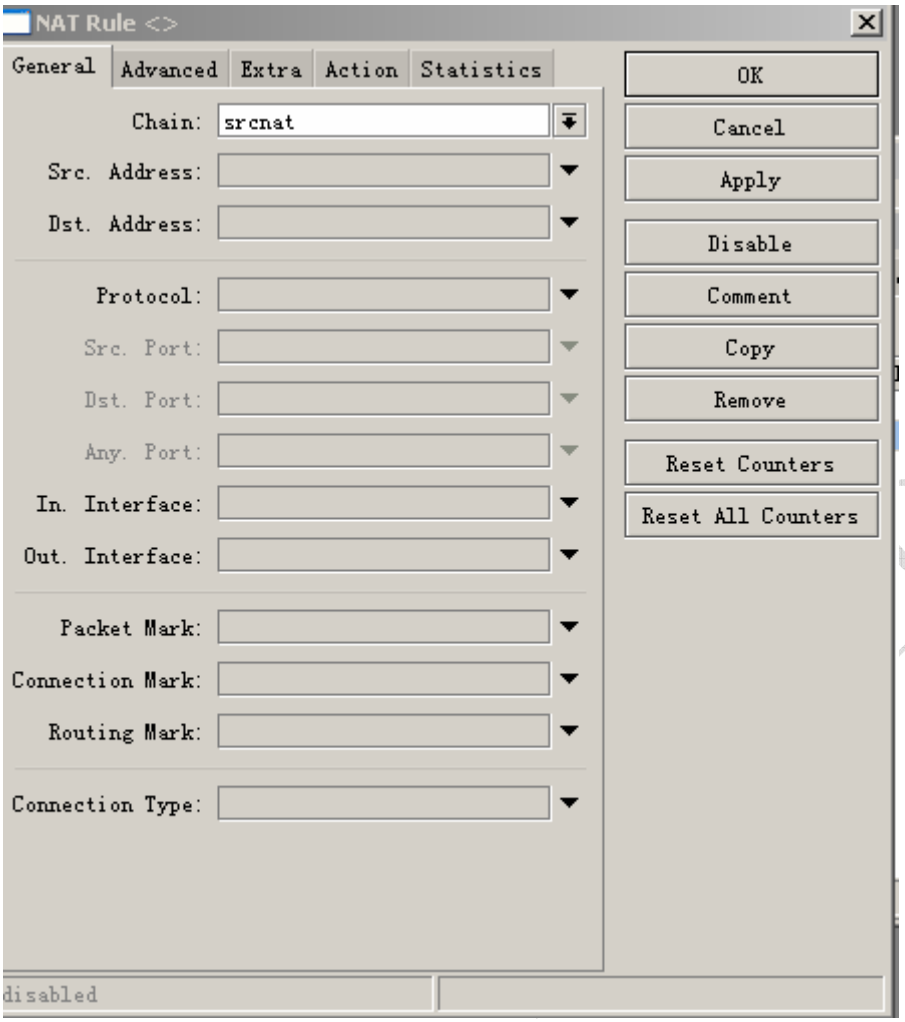
再在 ip ipsec 里的 peers 标签里添加目标外网 ip 地址和 secert 密码：



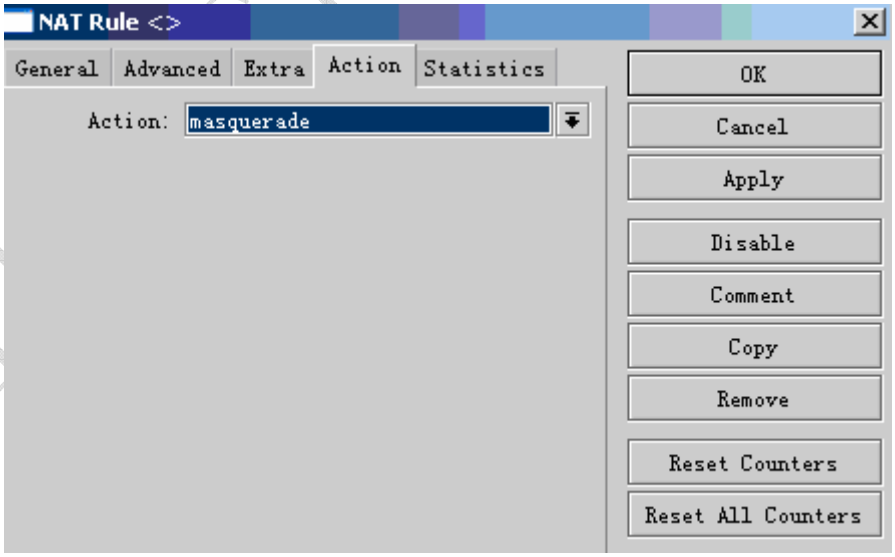
再在 ip firewall 里面的 nat 标签建立源内网地址和对端内网地址：



在建立 nat 的转换 chain 选择 srcnat：



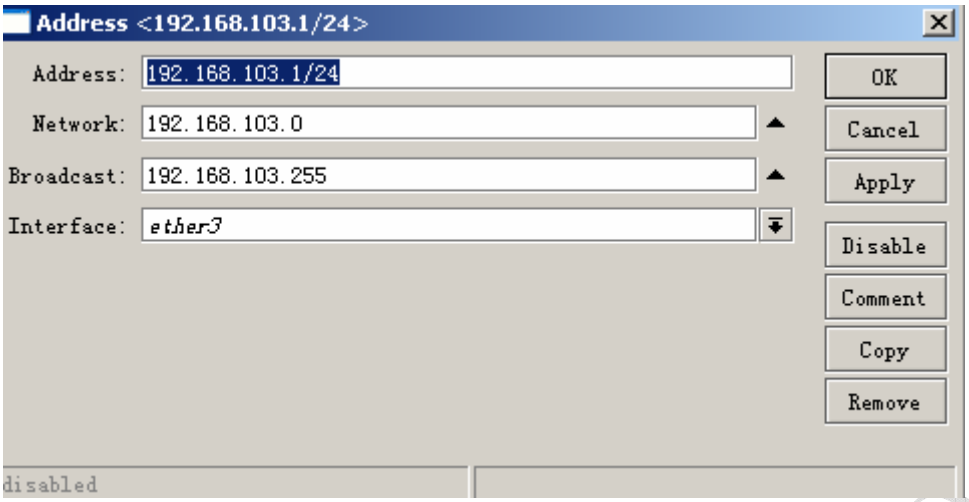
再在 action 里面选择 masquerade：



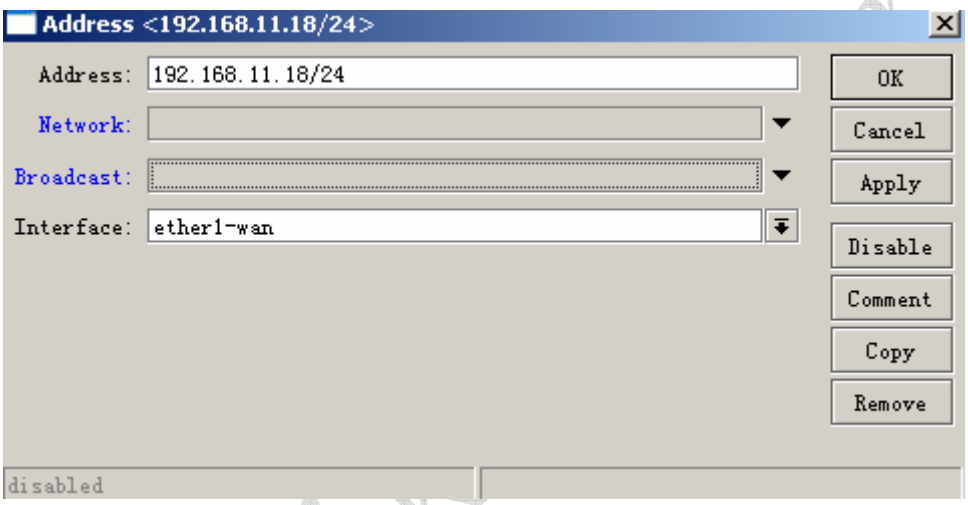
以上就是 R1 在 winbox 里面的配置过程。R1 已经配置完成现在就 R2 了。

R2 配置

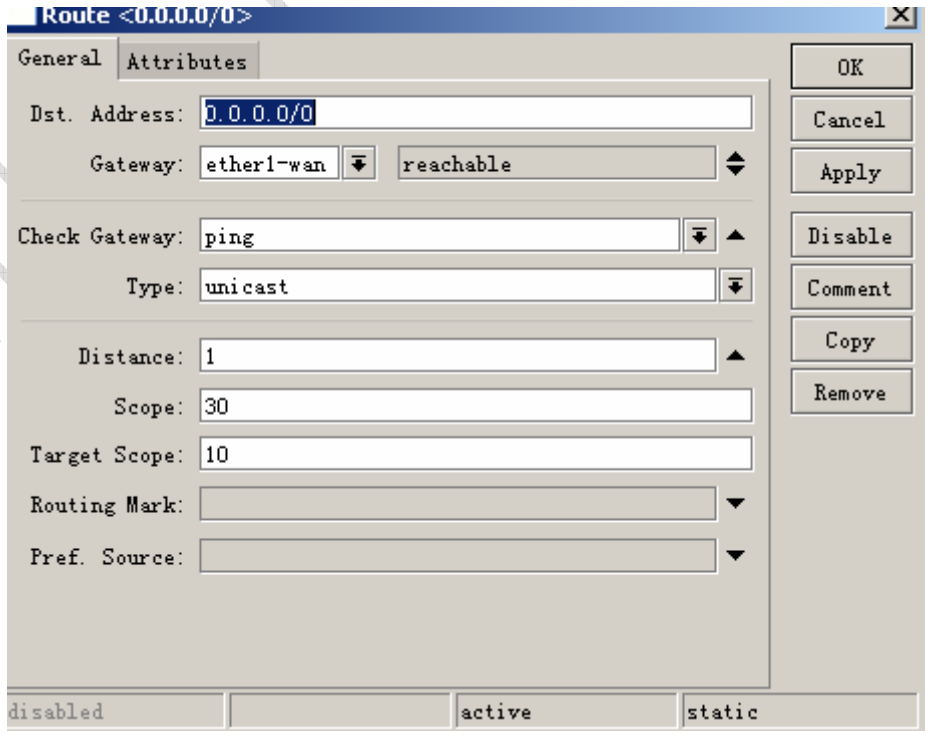
进入 ip address 里面添加内网接口地址：



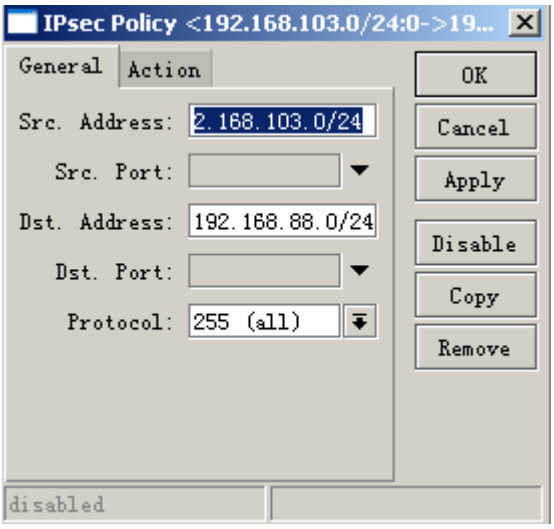
再添加外网接口地址：



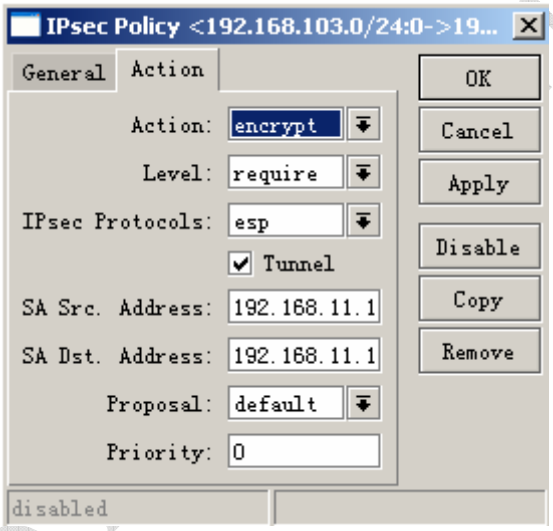
进入 ip routes 里面添加网关出口：



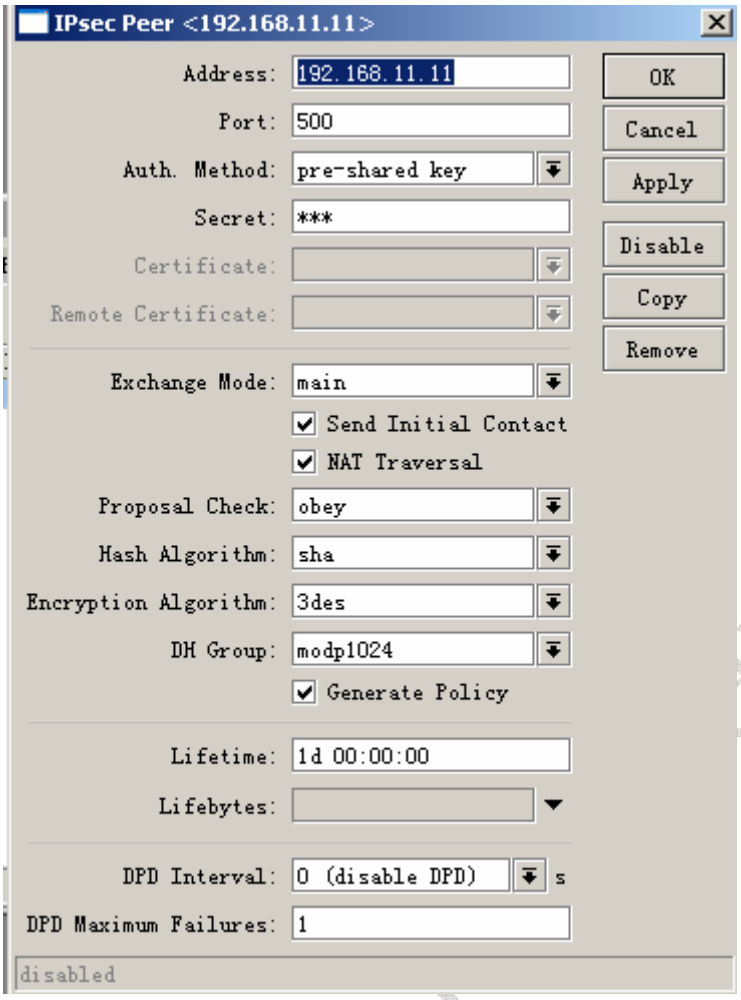
进入 ip ipsec 里面 policies 的 general 选项添加内网的源地址和需要做 ipsec 的对端内网地址:



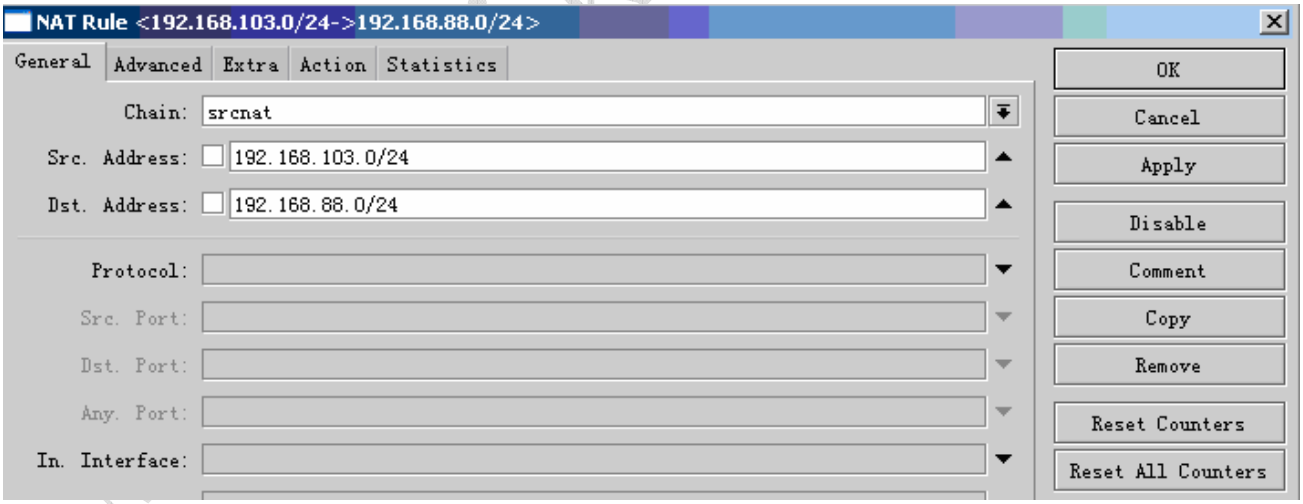
再在 action 选项里面添加源外网地址和对端外网地址和开启 tunnel 隧道协议:



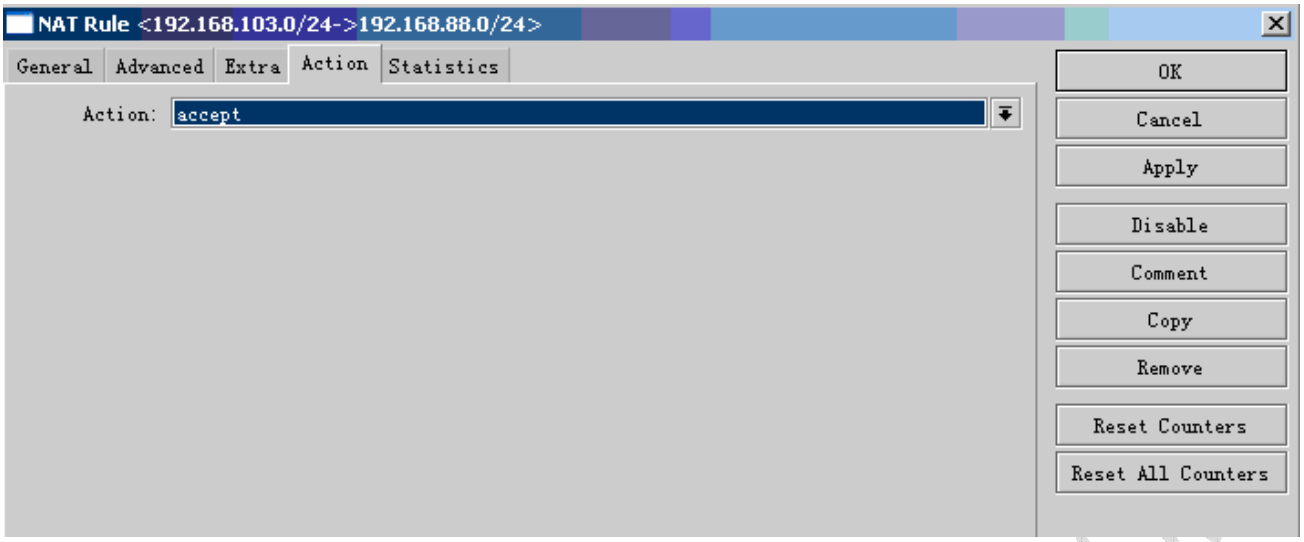
再在 ip ipsec 里的 peers 标签里添加对端外网 ip 地址和 secert 密码:



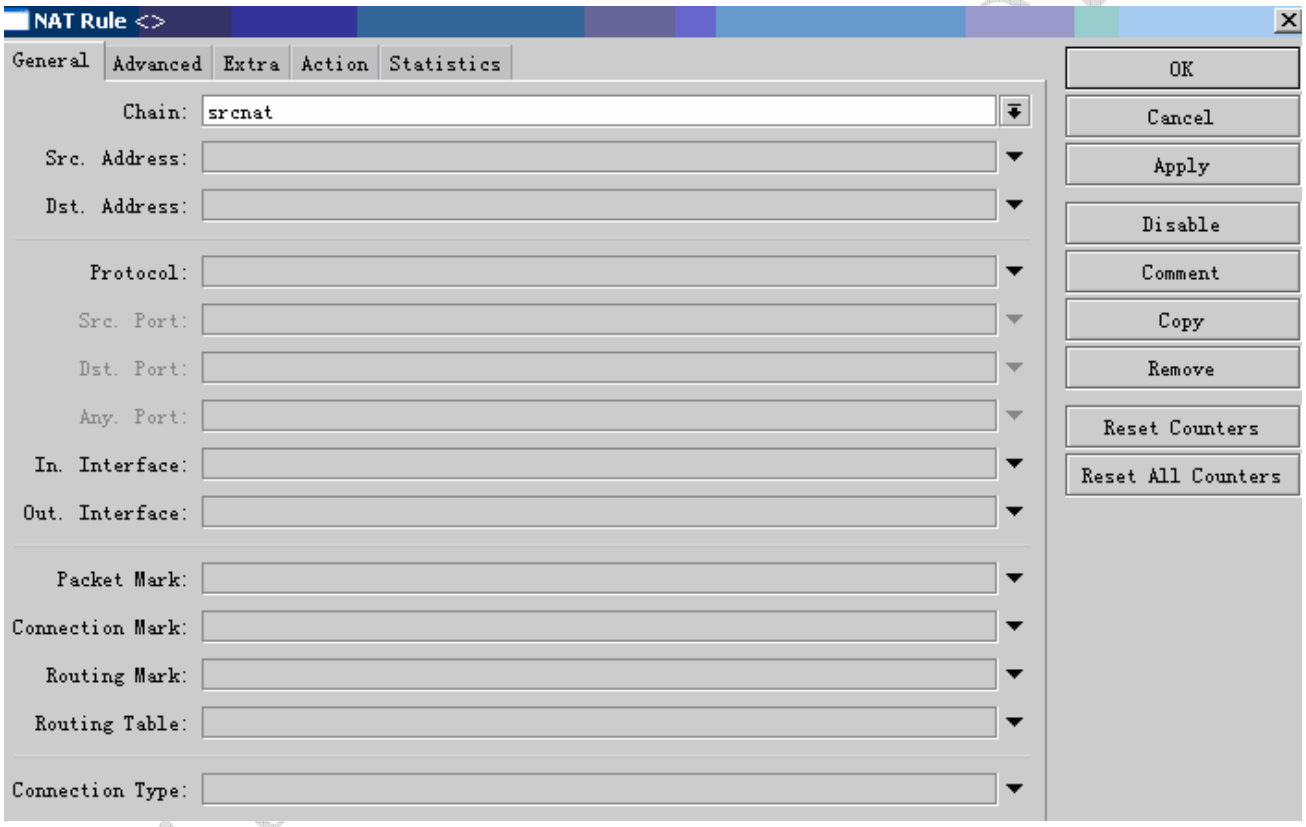
再在 ip firewall 里面的 nat 标签建立源内网地址和对端内网地址：



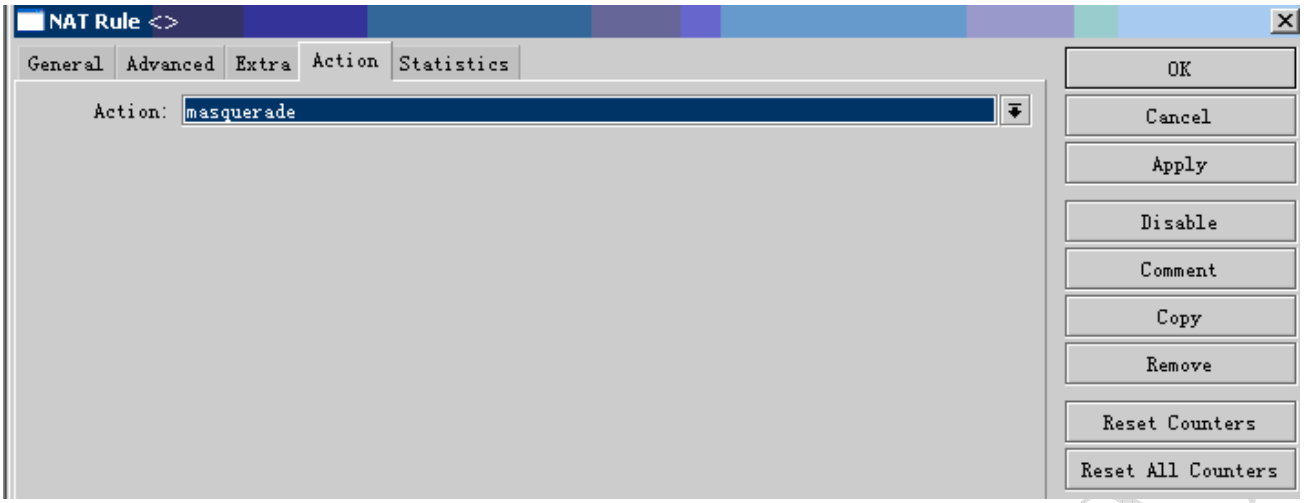
再在 action 里面选择 accept：



在建立 nat 的转换:

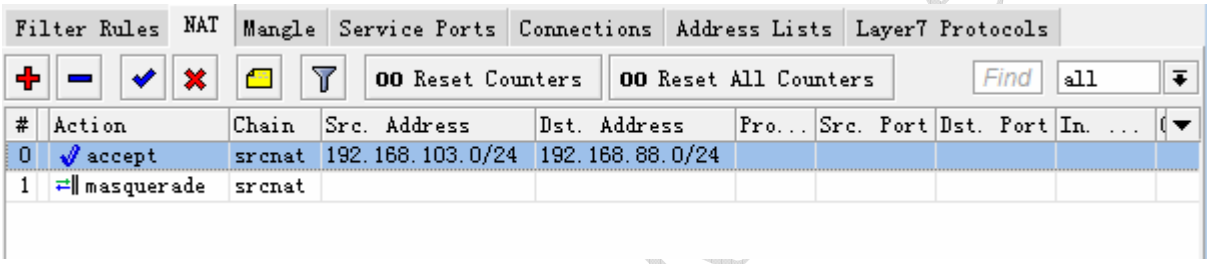


再在 acion 里面选择 masquerade：



以上就是 R2 的配置过程。

注意：NAT 规则的配置的上下顺序，accept 规则需在 masquerade 伪装规则前：

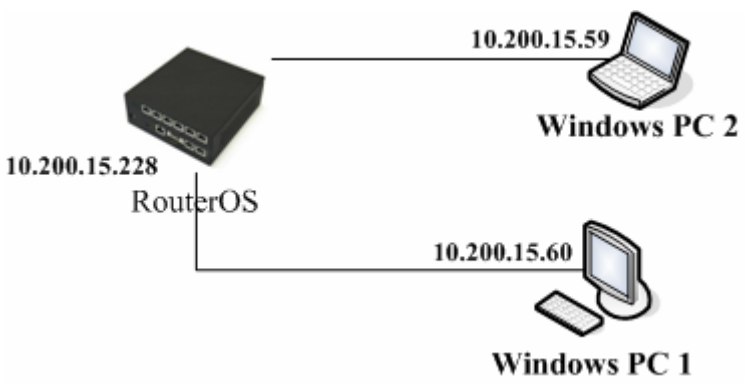


24.2 Windows L2TP/IPsec 连接

Microsoft Windows XP/Vista/win7 内建了 PPTP 客户端和和 L2TP/IPSec 客户端。PPTP 链接是不要 IPsec 加密的，而 windows 的 L2TP/IPsec 默认要求建立 IPsec 链接后，才能进行 L2TP 的拨号连接，这样的解决方法在早期采用的是修改 windows 的注册表，将 windows 默认的 IPsec 连接值修改并关闭，相对于终端客户操作繁琐，且安全性降低。为了能正常让 windows 的 L2TP/IPsec 与 RouterOS 连接，我们可以配置 RouterOS 启用 IPsec。

Windows 建立 L2TP/IPSec 连接，首先要求连接到对端的 IPSes，在 IPSec 建立完成后在允许 L2TP 连接，也就是 IPSec 连接在先，L2TP 其后，所以我们首先配置 IPSec 连接

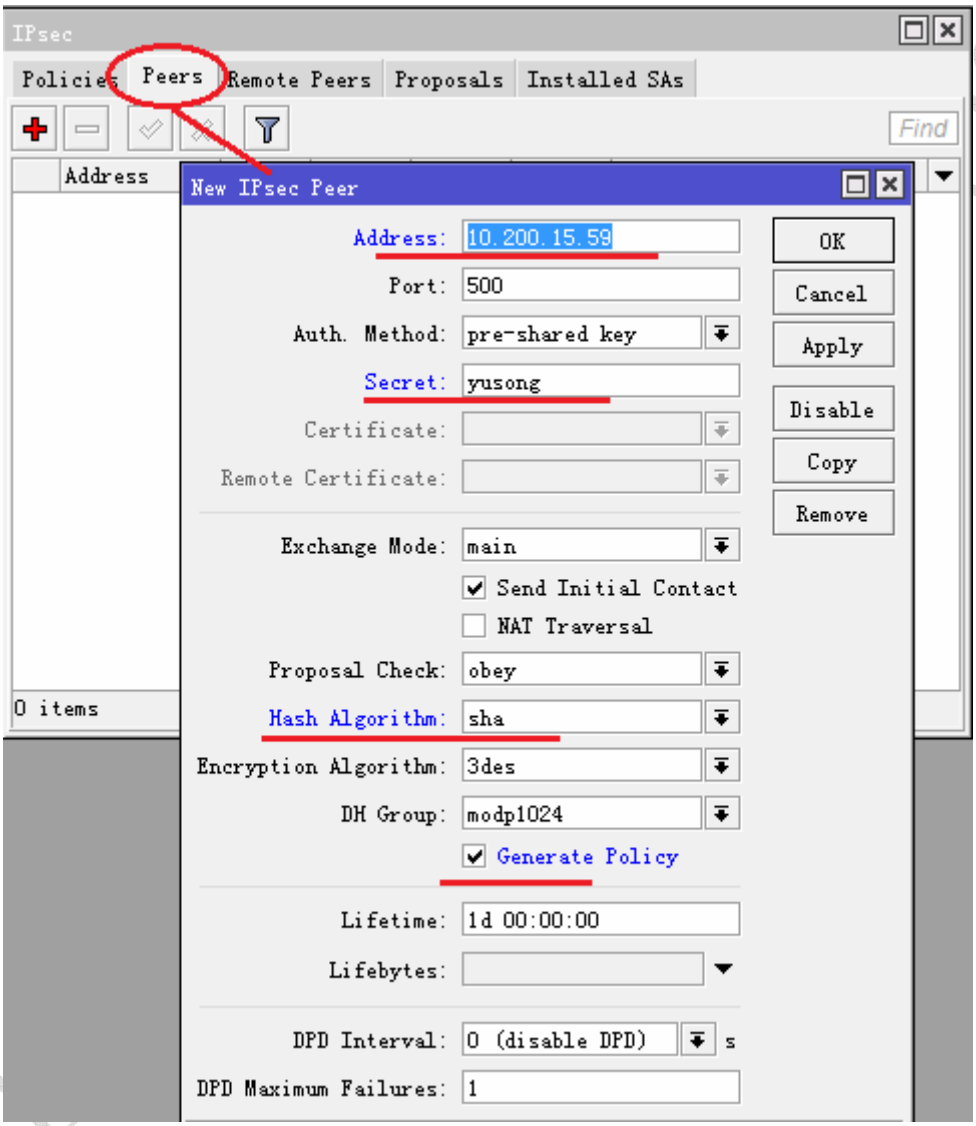
我们先确定一下网络结构：



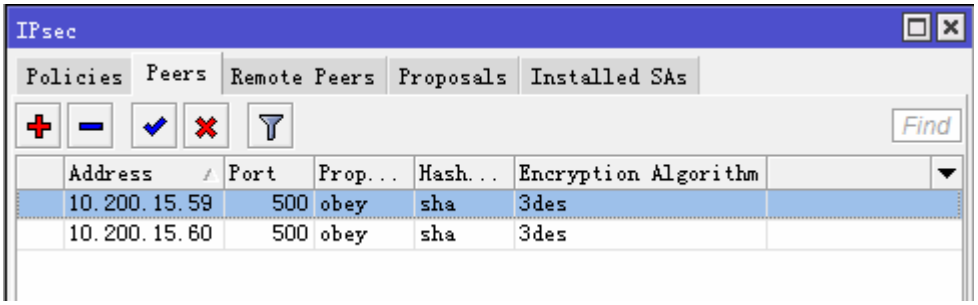
这里 RouterOS 的 IP 地址是 10.200.15.228，两台 PC 的 IP 地址分别是 10.200.15.59,和 10.200.15.60。两台 PC 的 IP 地址必须是固定，以便 IPsec 连接成功。在这个拓扑图里要求所有的地址是能被访问到的，即非 nat 转换的地址（也非 L2TP 隧道分配的 IP 地址）。

IPSec 配置

首先要将 IPsec 指向对端的 windows PC 的 IP 地址（非 L2TP 分配 IP 地址），进入/ip ipsec 菜单下（确定安装 security 功能包），选择 peer 标签，设置 address 为 PC 的 IP 地址，secret 设置共享密钥 yusong，Hash-algorithm 选择 sha，generate-policy 勾上，其他默认。



添加 10.200.15.60 的 peer 规则




```
/ip ipsec peer add address=10.200.15.59:500 auth-method=pre-shared-key \
```

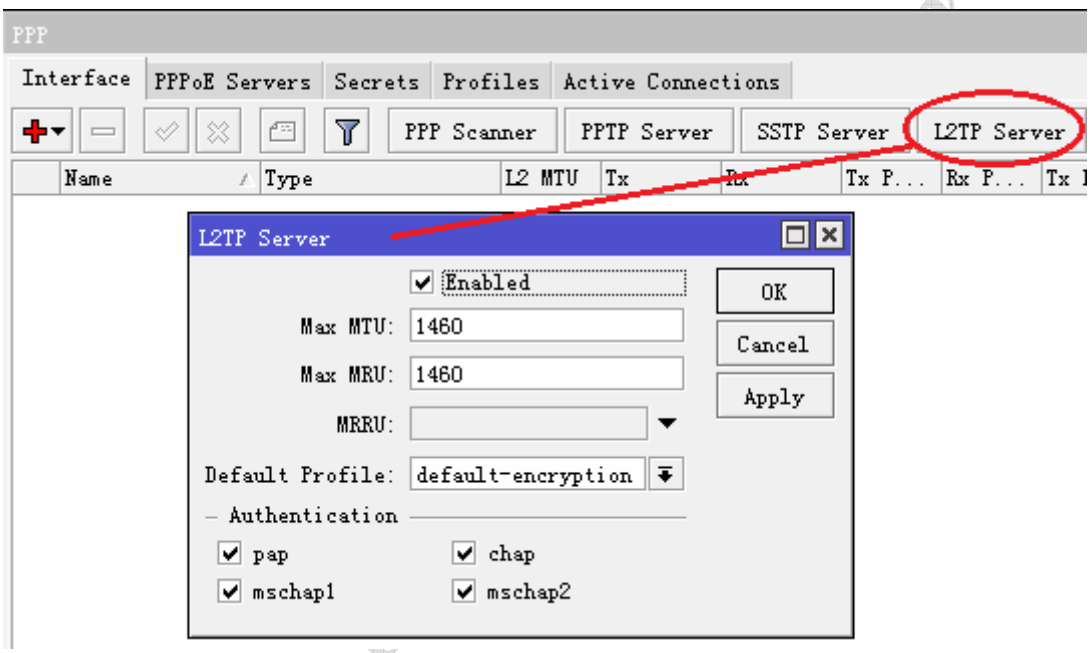
```
   secret=yusong hash-algorithm=sha enc-algorithm=3des generate-policy=yes
/ip ipsec peer add address=10.200.15.60:500 auth-method=pre-shared-key \
   secret=yusong hash-algorithm=sha enc-algorithm=3des generate-policy=yes
```

添加 IPSec peer 设置，

- **address=10.200.15.59** 是你的 windows 电脑的网卡实际地址。
- **:500** 端口号;
- **hash-algorithm=sha** 和 **enc-algorithm=3des** 是 windows 上的默认配置；
- **generate-policy=yes** 自动产生 IPSec 策略

RouterOS 配置

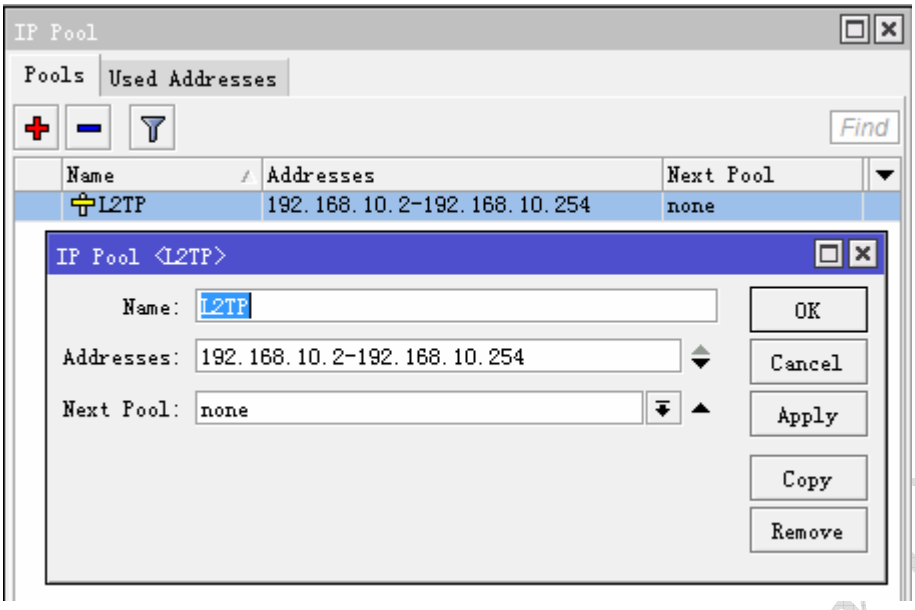
首先我们配置 RouterOS 的 L2TP 服务器，这个配置和普通的 PPTP 配置一样，在 PPP 里启用 L2TP 服务



命令行配置，记住这里的路径不同：

```
/ interface l2tp-server server set enabled=yes
```

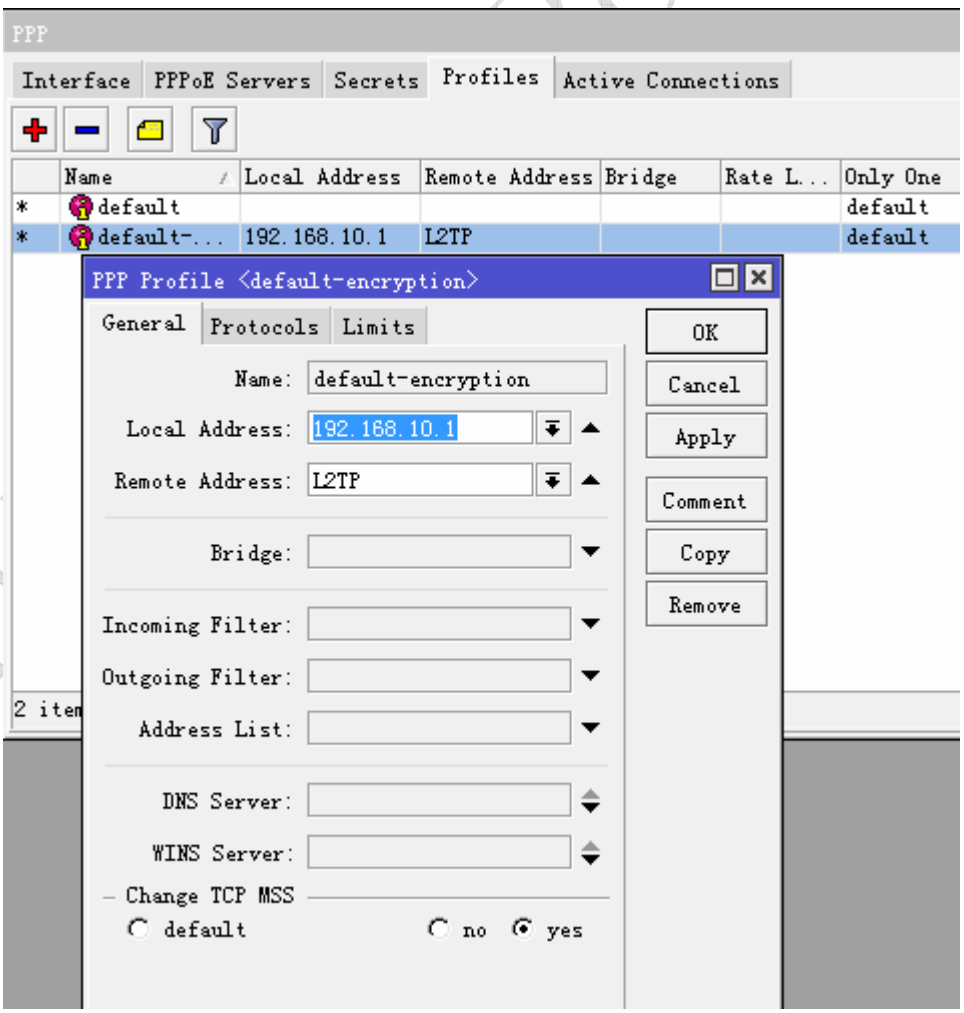
进入 ip pool 设置分配给用户的地址池：



命令操作如下：

```
/ip pool add name=L2TP ranges=192.168.10.2-192.168.10.254
```

进入/ppp profile 配置 default-encryption 的规则:

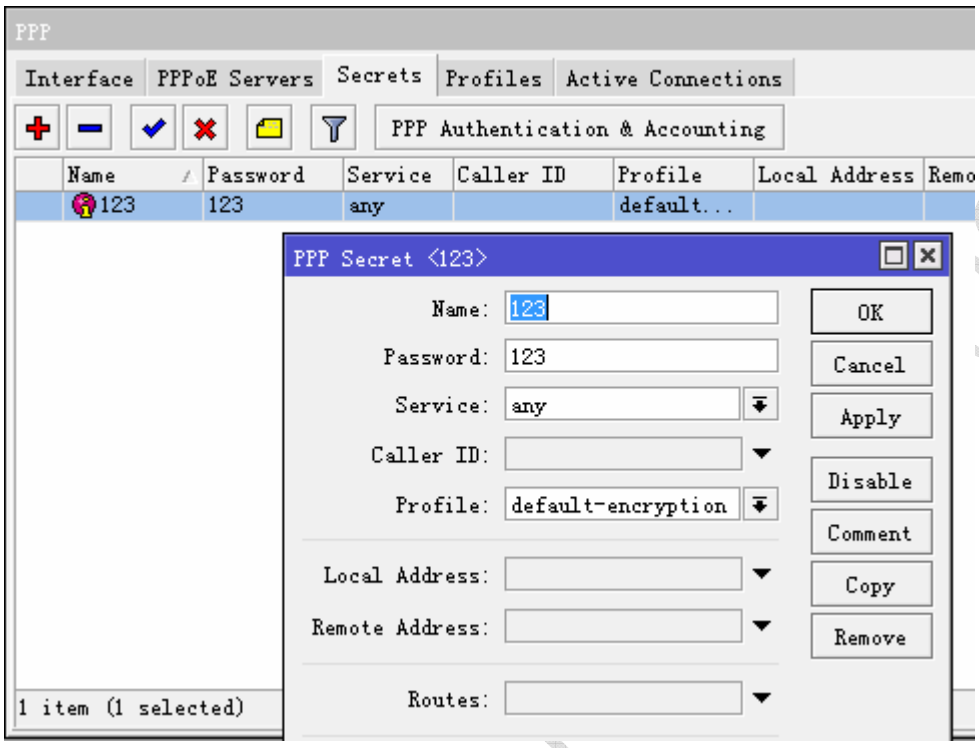


DNS 和 limit 选项里的 rate-limit、only one 参数根据需要设置，这里就不多讲解。

命令行配置

```
/ppp profile> set 1 local-address=192.168.10.1 remote-address=L2TP
```

进入/ppp secret 添加用户账号



命令行配置

```
/ ppp secret add name=123 password=123 profile=default-encryption
```

到这里 L2TP 服务器配置完成。

Windows 配置

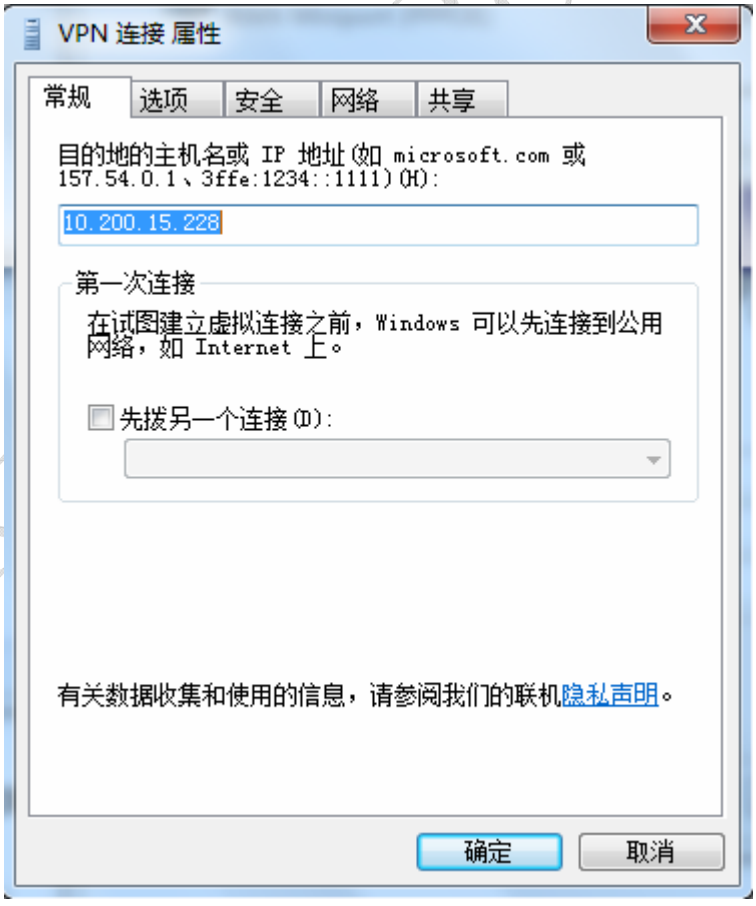
Windows 配置包含 2 个部分，第一个部分添加新的网络连接，第二个部分调整 IPSec 设置

Win7 配置步骤：

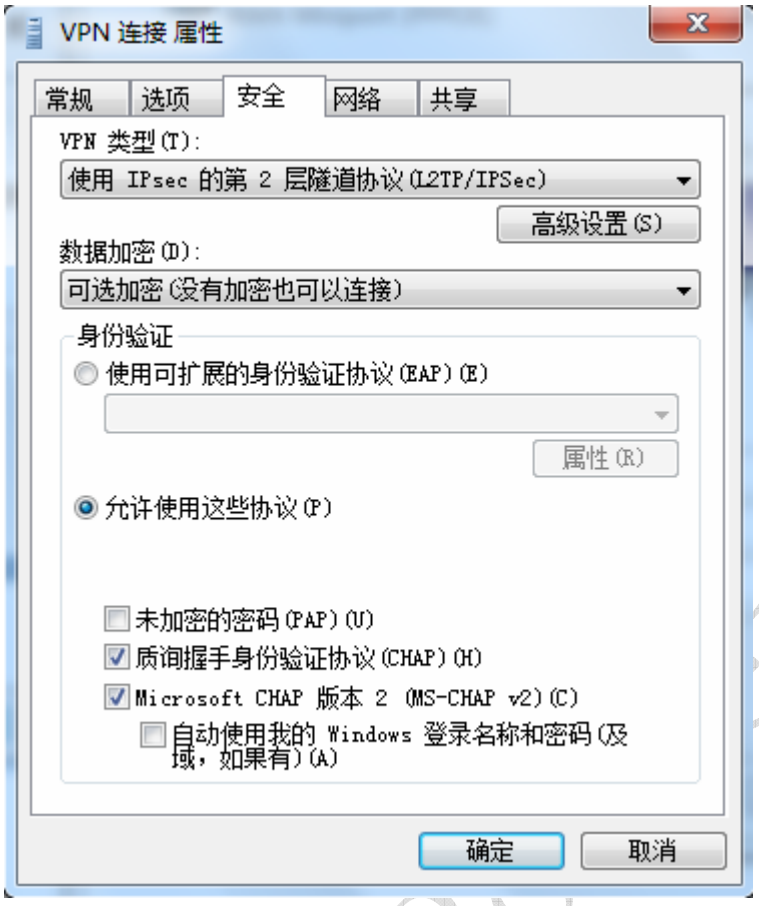
- 点开始菜单;
- 控制面板\网络和 Internet\网络和共享中心
- 设置新的连接或网络;
- 添加一个 VPN 连接，
- 目的地的主机或域名填写 10.200.15.228（具体操作跟着步骤走，不详细说明）

接下来我们需要配置 VPN 连接的属性

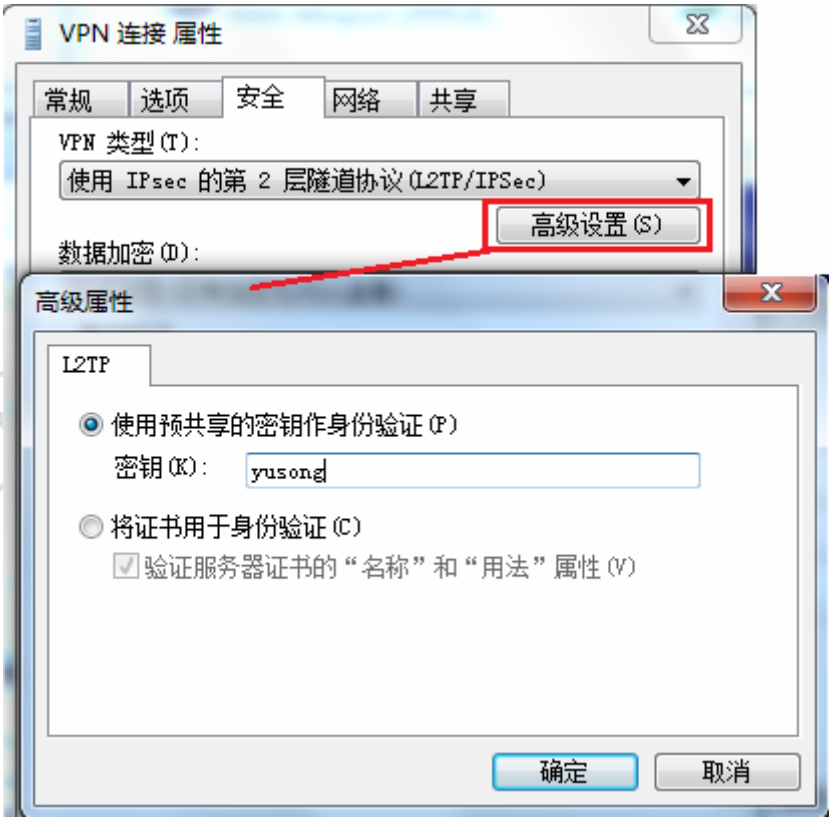
确定主机地址是 10.200.15.228



选择 VPN 类型为使用 ipsec 的第 2 层隧道协议（L2TP/IPSec）

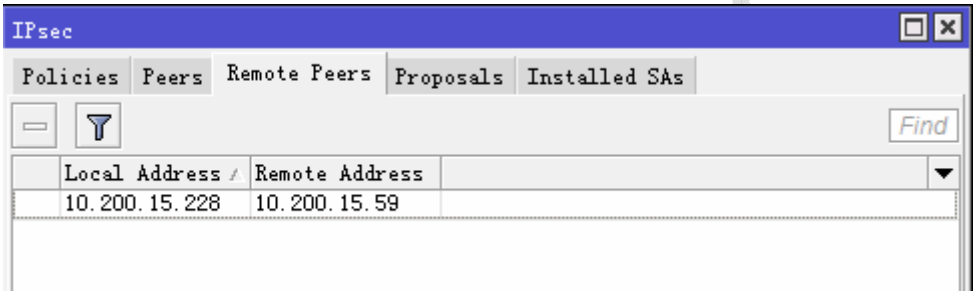


选择高级设置，并设置使用预共享的密钥作身份验证：输入相同的密钥：yusong

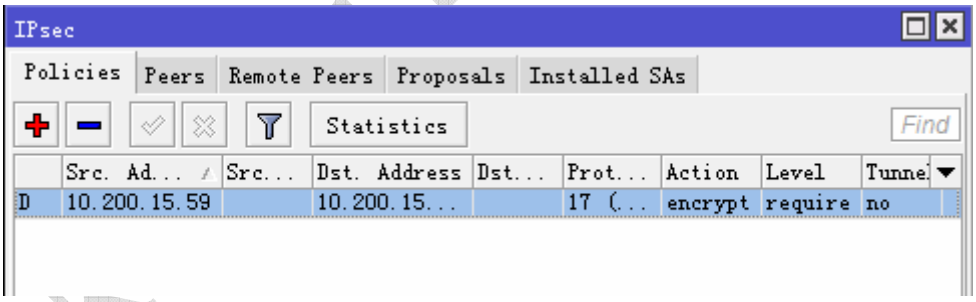


配置完成后，输入账号 123，密码 123，连接

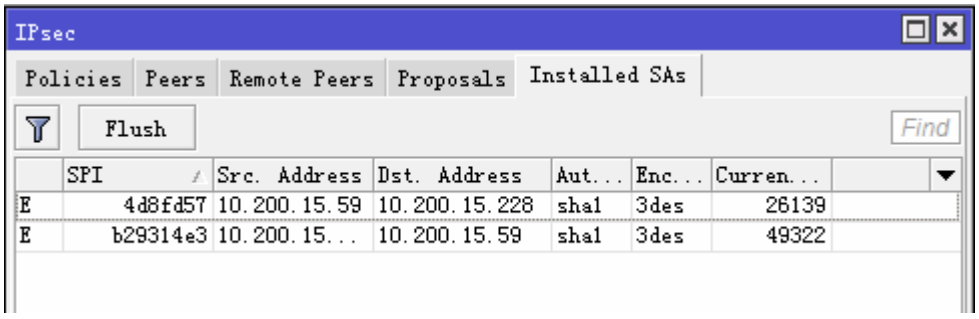
连接完成后，在 remote peers 可以看到连接的 IP 地址



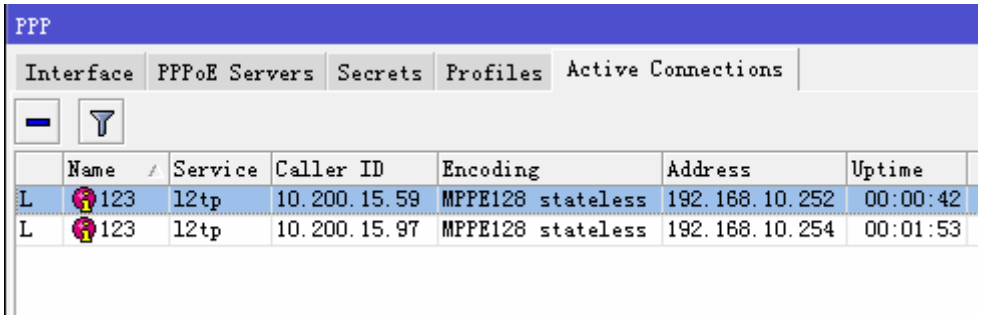
Policies 策略会被自动添加



Installed SAs 状态，注意当你的 L2TP 注销后，可能会出现 Installed SAs 状态没有清楚，第二次重播可能需要使用 Flush 清空状态



PPP 里的 active 状态



第二十五章 Bonding

Bonding 是通过汇聚多个接口到一个虚拟的链接上，这种方式可以获得更高的带宽或提供失效转移接管。Bongding 操作必须用于二层链路层，不支持三层 IP 层的应用。

25.1 Bonding 基本操作

让我们假设每个路由器有 2 张网卡（Router1 和 Router2）并且我们想在两个路由器之间得到最大的传输速率。通过 bonding 配置可以让该设想成为可能。如下配置：

1. 确定你没有 IP 地址在相应的接口，这将被从属到 bonding 接口上！
2. 在 Router1 上添加 **bonding** 接口：

```
[admin@Router1] interface bonding> add slaves=ether1,ether2
```

在 Router2 上添加：

```
[admin@Router2] interface bonding> add slaves=ether1,ether2
```

3. 添加地址到 bonding 接口上：

```
[admin@Router1] ip address> add address=172.16.0.1/24 interface=bonding1
[admin@Router2] ip address> add address=172.16.0.2/24 interface=bonding1
```

4. 在 Router1 上测试链接：

```
[admin@Router1] interface bonding> /pi 172.16.0.2
172.16.0.2 ping timeout
172.16.0.2 ping timeout
172.16.0.2 ping timeout
172.16.0.2 64 byte ping: ttl=64 time=2 ms
172.16.0.2 64 byte ping: ttl=64 time=2 ms
```

注意：bonding 接口需要几秒钟时间的连通时间。

规格

需要功能包: ***system***
需要等级: *Level1*
操作路径: ***/interface bonding***

提供了最佳的失效转移管理，你需要指定 **link-monitoring** 参数:

- MII (媒体独立接口 Media Independent Interface) type1 or type2 - 媒体独立接口是一个在操作系统与 NIC 之间的理论层，探测连接是否运行（执行可以通过其他功能实现，但在我们的事例中这个是非常重要的）。
- ARP – 地址解析协议（通过 **arp-interval** 时间）检测连接状态。

link-monitoring 被用于检测是否连接。

属性描述

arp (disabled | enabled | proxy-arp | reply-only; 默认: **enabled**) – 接口的地址解析协议
disabled – 接口不使用 ARP
enabled – 接口使用 ARP
proxy-arp – 接口使用 ARP 代理功能
reply-only – 接口将只回应/ip arp 的静态 MAC 地址
arp-interval (*time*; 默认: **00:00:00.100**) – 通过定义多少毫秒监测 ARP 请求。
arp-ip-targets (*IP 地址*; 默认: **""**) – IP 目标地址，如果 **link-monitoring** 被设置 **arp** 目标 IP 地址将会被监视。你也可以指定多个 IP 地址。
down-delay (*时间*; 默认: **00:00:00**) – 如果一个连接失效被探测到，bonding 接口通过 **down-delay** 时间禁用配置。
lacp-rate (1sec | 30secs; 默认: **30secs**) – 连接聚合控制协议速率是指定多久将 bonding 端的 LACPDUs 进行交换。被用于确定是否连接或进行其他变化。LACP 试着适应这些变化并提供失效管理。
link-monitoring (arp | mii-type1 | mii-type2 | none; 默认: **none**) – 连接监视是否使用 (是否设置启用)
arp – 使用地址解析协议，探测远程地址是否到达。
mii-type1 – 使用 MII type1 协议确认连接状态。连接状态探测依赖设备驱动。如果 bonding 显示状态为 up，但运行时并未启动，说明该卡可能不支持 bonding 功能。
mii-type2 – 使用 MII type2 探测连接状态（被用于如果 **mii-type1** 不支持 NIC）
none – 没有任何模式监测，如果一个连接失效，不会被关闭（但没有传输通过）。
mac-address (*只读: MAC address*) – bonding 接口的 MAC 地址
mii-interval (*时间*; 默认: **00:00:00.100**) – 多久监测一次连接失效（此参数被用于在 **link-monitoring** 设置为 **mii-type1** 或 **mii-type2**）
mode (802.3ad | active-backup | balance-alb | balance-rr | balance-tlb | balance-xor | broadcast; 默认: **balance-rr**) – 接口绑定模式，如下：
802.3ad - IEEE 802.3ad 动态连接聚合，提供容错和负载平衡。在这个模式下，接口被聚合到一个组里，每个 slave 共享同样的速度。如果你在两个 bonding 路由器之间使用一个交换机，必须确定这个交换机支持 IEEE 802.3ad。**active-backup** – 提供连接备份。在同一时间仅一个 slave 可以运行。如果一个失效，另外一个 slave 自动连接。
balance-alb – 自适应负载均衡。该模式包含 **balance-tlb**，通过接收传输负载均衡。设备驱动应支持设置 MAC 地址，不需要指定的交换机支持
balance-rr – 轮询负载均衡。在 bonding 接口里 Slaves 将依次序的传输和接收。提供负载均衡和容错
balance-tlb – 输出传输同分布式方式分配负荷到当前的每个 slave 上，传入数据被接收通过当前 salve。如果接收 slave 失败，

这时另外一个 slave 带走实效的 MAC 地址。不需要任何特殊的交换机支持

balance-xor – 为传输使用 XOR 策略。仅提供失效管理，但不支持负载均衡

broadcast – 同样的数据在所有接口广播一次。这样提供失效容错，但在一些慢的机器上降低了传输吞吐量。

mtu (*整型*: 68..1500; 默认: **1500**) – 最大传输单元，单位 btyes

name (*名称*) – bonding 接口的名称

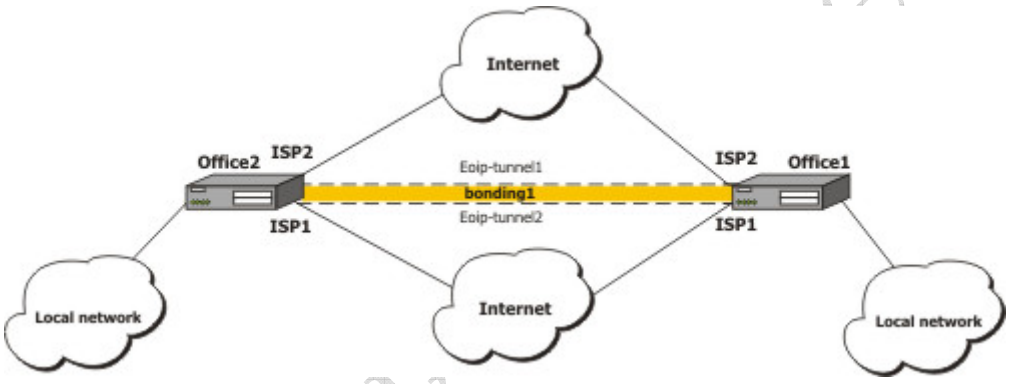
primary (*名称*; 默认: **none**) – 接口被涩混浊主要的输出媒体。如果主接口失效，从属接口会被自动启用。该参数仅能使用于 **mode=active-backup**

slaves (*名称*) – 至少 2 个 ethernet 接口被用于 bonding 接口

up-delay (*时间*; 默认: **00:00:00**) – 如果一个链路已经连接，bonding 接口被 **up-delay** 时间禁用，在这个时间过后 bonding 接口启用。

25.2 基于两个EoIP 隧道的Bonding

假设你需要通过 MikroTik 路由器配置以下的网络设置，你有 2 个办公室，并同时接入了相同的 2 个 ISP 线路，你想绑定 2 条线路，得到双倍的贷款速度，并提供失效管理。



两个路由器直接通过 2 个 ISP 连接到 Internet，并配置这两个路由器连接上网。

- 配置 **office1** 路由器：

```
[admin@office1] > /interface print
Flags: X - disabled, D - dynamic, R - running
 #    NAME                                      TYPE           MTU
 0  R isp1                                      ether          1500
 1  R isp2                                      ether          1500

[admin@office1] > /ip address print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS           NETWORK        BROADCAST       INTERFACE
 0   1.1.1.1/24        1.1.1.0        1.1.1.255       isp2
 1   10.1.0.111/24     10.1.0.0       10.1.0.255      isp1
```

配置 **Office2** 的路由器

```
[admin@office2] interface> print
Flags: X - disabled, D - dynamic, R - running
 #    NAME                                      TYPE           MTU
```

```
 0  R isp2                                          ether          1500
 1  R isp1                                          ether          1500
[admin@office2] interface> /ip add print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS           NETWORK        BROADCAST      INTERFACE
 0   2.2.2.1/24        2.2.2.0        2.2.2.255      isp2
 1   10.1.0.112/24     10.1.0.0       10.1.0.255     isp1
```

- 通过 EoIP 隧道连接，实现一个虚拟的二层网络链接，用于 bonding 的连接（由于 bonding 基于二层链路层的链路聚合，所以必须使用 2 层接口）。先配置 **Office1** 通过 ISP1 连接的 EoIP 隧道：

```
[admin@office1] > interface eoip add remote-address=10.1.0.112 tunnel-id=2
\... mac-address=FE:FD:00:00:00:04
[admin@office1] > interface eoip print
Flags: X - disabled, R - running
0  R name="eoip-tunnel2" mtu=1500 mac-address==FE:FD:00:00:00:04 arp=enabled
\... remote-address=10.1.0.112 tunnel-id=2
```

在 **Office2** 路由器上配置 ISP1 线路的 EoIP

```
[admin@office2] > interface eoip add remote-address=10.1.0.111 tunnel-id=2
\... mac-address=FE:FD:00:00:00:02
[admin@office2] > interface eoip print
Flags: X - disabled, R - running
0  R name="eoip-tunnel2" mtu=1500 mac-address=FE:FD:00:00:00:02 arp=enabled
\... remote-address=10.1.0.111 tunnel-id=2
```

在 **Office1** 路由器上配置 ISP2 的 EoIP 隧道

```
[admin@office1] > interface eoip add remote-address=2.2.2.1 tunnel-id=1
\... mac-address=FE:FD:00:00:00:03
[admin@office1] interface eoip> print
Flags: X - disabled, R - running
 0  R name="eoip-tunnel1" mtu=1500 mac-address=FE:FD:00:00:00:03 arp=enabled
    remote-address=2.2.2.1 tunnel-id=1

1  R name="eoip-tunnel2" mtu=1500 mac-address=FE:FD:00:00:00:04 arp=enabled
 remote-address=10.1.0.112 tunnel-id=2
```

在 **Office2** 路由器上配置 ISP2 的 EoIP 隧道

```
[admin@office2] > interface eoip add remote-address=1.1.1.1 tunnel-id=1
\... mac-address=FE:FD:00:00:00:01
[admin@office2] interface eoip> print
Flags: X - disabled, R - running
```

```
 0  R name="eoip-tunnel1" mtu=1500 mac-address=FE:FD:00:00:00:01 arp=enabled
      remote-address=1.1.1.1 tunnel-id=1

 1  R name="eoip-tunnel2" mtu=1500 mac-address=FE:FD:00:00:00:02 arp=enabled
      remote-address=10.1.0.111 tunnel-id=2
```

- 设置 Bonding，在 **Office1**

```
 [admin@office1] interface bonding> add slaves=eoip-tunnel1,eoip-tunnel2
[admin@office1] interface bonding> print
Flags: X - disabled, R - running
 0  R name="bonding1" mtu=1500 mac-address=00:0C:42:03:20:E7 arp=enabled
slaves=eoip-tunnel1,eoip-tunnel2 mode=balance-rr primary=none  link-monitoring=none
arp-interval=00:00:00.100 arp-ip-targets=""  mii-interval=00:00:00.100 down-delay=00:00:00
up-delay=00:00:00  lacp-rate=30secs
[admin@office1] ip address> add address=3.3.3.1/24 interface=bonding1
[admin@office1] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS           NETWORK        BROADCAST      INTERFACE
 0   1.1.1.1/24        1.1.1.0        1.1.1.255      isp2
 1   10.1.0.111/24     10.1.0.0       10.1.0.255     isp1
 2   3.3.3.1/24        3.3.3.0        3.3.3.255      bonding1
```

在 **Office2** 上配置

```
[admin@office2] interface bonding> add slaves=eoip-tunnel1,eoip-tunnel2
[admin@office2] interface bonding> print
Flags: X - disabled, R - running
 0  R name="bonding1" mtu=1500 mac-address=00:0C:42:03:20:E7 arp=enabled
      slaves=eoip-tunnel1,eoip-tunnel2 mode=balance-rr primary=none
      link-monitoring=none arp-interval=00:00:00.100 arp-ip-targets=""
      mii-interval=00:00:00.100 down-delay=00:00:00 up-delay=00:00:00
      lacp-rate=30secs
[admin@office2] ip address> add address=3.3.3.2/24 interface=bonding1
[admin@office2] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS           NETWORK        BROADCAST      INTERFACE
 0   2.2.2.1/24        2.2.2.0        2.2.2.255      isp2
 1   10.1.0.112/24     10.1.0.0       10.1.0.255     isp1
 2   3.3.3.2/24        3.3.3.0        3.3.3.255      bonding1
[admin@office2] ip address> /ping 3.3.3.1
3.3.3.1 64 byte ping: ttl=64 time=2 ms
3.3.3.1 64 byte ping: ttl=64 time=2 ms
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 2/2.0/2 ms
```

关于无线 bonding 设置请参阅本人编写的 RouterOS 无线手册

第二十六章 VLAN

VLAN 是基于 802.1Q VLAN 协议。它允许你在单个以太网或无线接口上拥有多个虚拟 LAN，给予了高效分割 LAN 的能力。它最多可以支持 4095 个 VLAN 接口，每个以太网的每个接口都有唯一的 VLAN ID。很多路由器，包括 Cisco 或华为，以及很多二层交换机也都支持。

VLAN 是一个允许终端用户如同物理连接到一个隔离 LAN 一样相互通信的逻辑分组，独立于网络的物理配置。VLAN 支持添加新的安全尺度并对允许当在不相关用户间逻辑地维持分隔时共享一个物理网络收取开支。

规格

功能包要求: ***system***
等级要求: *Level1 (limited to 1 vlan)* , *Level3*
子目录要求: ***/interface vlan***
标准与技术: VLAN (IEEE 802.1Q)

VLAN 是一个简单的对一套交换机端口进行分组以形成一个逻辑网络的方法。在一个交换机内这是一个简单的逻辑配置。当 VLAN 延伸到多个交换机时，内部交换机连接就成为主干，在它上面数据包会被标记以指明它们属于哪个 VLAN。

你可以使用 MikroTik RouterOS (也可以是 Cisco IOS 和 Linux)来标记这些数据包也可以用来接受并路由标记了的包。

由于 VLAN 工作于 OSI 的第二层，它可以作为另一个没有任何显示的网络接口使用。VLAN 成功地通过以太网桥（对 MikroTik RouterOS 桥你应该设置 **forward-protocols** 为 **ip**, **arp** 以及 **other**；对其他桥也应该有类似设置）。

你可以在无线连接上传输 VLAN 并把多个 VLAN 接口放在一个无线接口上。注意 VLAN 不是一个全隧道协议（例如，它没有附加域来传输发送者和接收者的 MAC 地址），相同的限制适用于 VLAN 上的桥接也适用于普通的无线接口桥接。换句话说，当无线客户参与放置在无线接口的 VLAN 时，就没有可能使放置在一个无线接口站模式的 VLAN 与其他任何接口进行桥接。

当前支持的以太网接口

这是一个 VLAN 经过测试并能工作的网络接口列表。注意也存在很多其他支持 VLAN 的接口，但它们并没有被检测。

- Realtek 8139
- Intel PRO/100
- Intel PRO1000 server adapter
- National Semiconductor DP83816 based cards (RouterBOARD200 onboard Ethernet, RouterBOARD 24 card)
- National Semiconductor DP83815 (Soekris onboard Ethernet)
- VIA VT6105M based cards (RouterBOARD 44 card)
- VIA VT6105
- VIA VT6102 (VIA EPIA onboard Ethernet)

这是一个 VLAN 经过测试并能工作的网络接口列表，**但不支持大数据包（>1496 字节）**：

- 3Com 3c59x PCI
- DEC 21140 (tulip)

26.1 VLAN 配置

操作路径: ***/interface vlan***

属性描述

arp (disabled | enabled | proxy-arp | reply-only; 默认: **enabled**) – 地址解析协议设置
  disabled – 接口不使用 ARP 协议
  enabled - 接口使用 ARP 协议 I
  proxy-arp – 接口将成为 ARP 代理
  reply-only - 接口将只对源于它本身 IPD 地址的请求回应，但邻居 MAC 地址将仅从**/ip arp** 静态设置表收集。
interface (*名称*) - VLAN 网络的物理接口
mtu (*整型*; 默认: **1500**) – 最大传输单元
name (*名称*) - 参考接口名
vlan-id (*整型*; 默认: **1**) - 虚拟 LAN 用于区别 VLAN 的标识符或标记。必须在一个 VLAN 中对所有电脑是平等的。

注： MTU 必须像在以太网接口那样设置为 1500 字节。但这样也可能不能与一些不支持接受/传输满长度带有 VLAN 标题的以太网数据包的以太网卡一起工作（1500 字节数据+4 字节 VLAN 标题+14 字节以太网标题）。这种情况下使用 MTU1496，但要注意如果较长的数据包要在接口发送的话这会引起数据包的分割。同时要记得如果路径 MTU 搜索在源和目的间不能正常工作，MTU1496 可能引起一些问题。

实例： 在接口 **ether1** 添加并启用名为 **test** 且 **vlan-id**=1 的 VLAN：

```
[admin@MikroTik] interface vlan> add name=test vlan-id=1 interface=ether1
[admin@MikroTik] interface vlan> print
Flags: X - disabled, R - running
  #    NAME                MTU  ARP      VLAN-ID INTERFACE
  0 X  test                1500 enabled   1      ether1
[admin@MikroTik] interface vlan> enable 0
[admin@MikroTik] interface vlan> print
Flags: X - disabled, R - running
  #    NAME                MTU  ARP      VLAN-ID INTERFACE
  0  R test                1500 enabled   1      ether1
[admin@MikroTik] interface vlan>
```

26.2 VLAN 应用事例

我们假设我们有两个或更多连接到 hub 的 MikroTik RouterOS 路由器。在 VLAN 将被创建的连到物理网络的接口是 **ether1**（它只是为了例子简单化才需要，不是必须的）。

要通过 VLAN 连接电脑它们就必须物理上连接并且唯一的 IP 地址应该分配给它们以便它们可以互相 ping 通。然后分别在它们创建 VLAN 接口：

```
[admin@MikroTik] interface vlan> add name=test vlan-id=32 interface=ether1
[admin@MikroTik] interface vlan> print
Flags: X - disabled, R - running
  #    NAME              MTU  ARP      VLAN-ID INTERFACE
  0  R test              1500 enabled   32     ether1
[admin@MikroTik] interface vlan>
```

如果接口成功的创建，那么它们都能够运行。如果电脑没有正确的连接（通过不再传输或转发 VLAN 包的网络设备），则两个或者一个接口不能运行。当接口运行时，IP 地址可以分配给 VLAN 接口。

在 Router 1 上:

```
[admin@MikroTik] ip address> add address=10.10.10.1/24 interface=test
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS          NETWORK        BROADCAST      INTERFACE
  0   10.0.0.204/24     10.0.0.0       10.0.0.255     ether1
  1   10.20.0.1/24      10.20.0.0      10.20.0.255    pc1
  2   10.10.10.1/24     10.10.10.0     10.10.10.255   test
[admin@MikroTik] ip address>
```

在 Router 2 上:

```
[admin@MikroTik] ip address> add address=10.10.10.2/24 interface=test
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS          NETWORK        BROADCAST      INTERFACE
  0   10.0.0.201/24     10.0.0.0       10.0.0.255     ether1
  1   10.10.10.2/24     10.10.10.0     10.10.10.255   test
[admin@MikroTik] ip address>
```

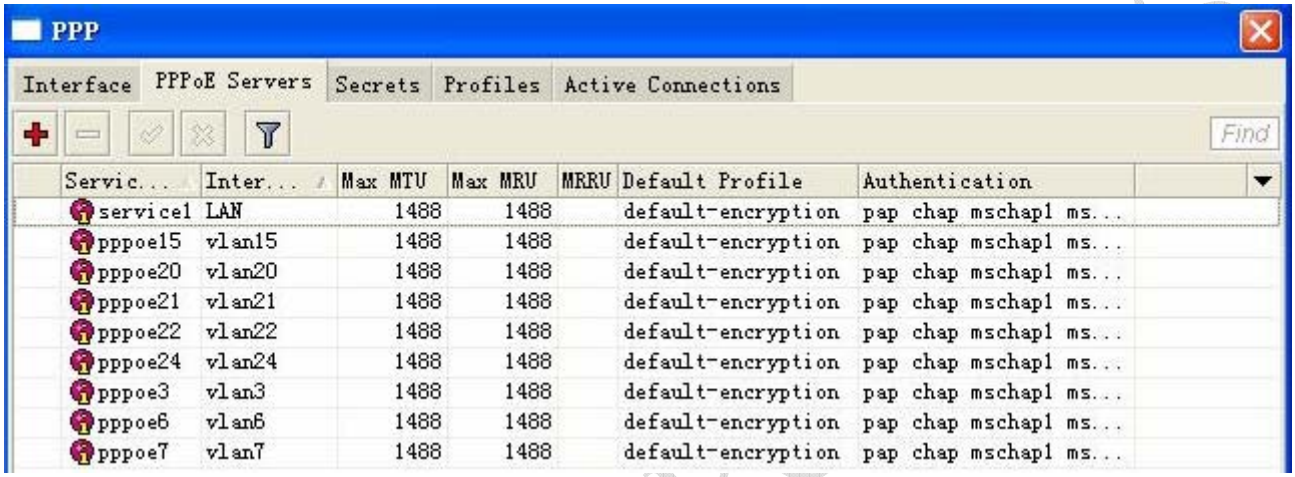
如果设置得正确，那么从 Router 1 可以 ping 通 Router 2，反之亦然：

```
[admin@MikroTik] ip address> /ping 10.10.10.1
10.10.10.1 64 byte pong: ttl=255 time=3 ms
10.10.10.1 64 byte pong: ttl=255 time=4 ms
10.10.10.1 64 byte pong: ttl=255 time=10 ms
10.10.10.1 64 byte pong: ttl=255 time=5 ms
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 3/10.5/10 ms
[admin@MikroTik] ip address> /ping 10.10.10.2
10.10.10.2 64 byte pong: ttl=255 time=10 ms
10.10.10.2 64 byte pong: ttl=255 time=11 ms
10.10.10.2 64 byte pong: ttl=255 time=10 ms
10.10.10.2 64 byte pong: ttl=255 time=13 ms
4 packets transmitted, 4 packets received, 0% packet loss
```

```
round-trip min/avg/max = 10/11/13 ms
[admin@MikroTik] ip address>
```

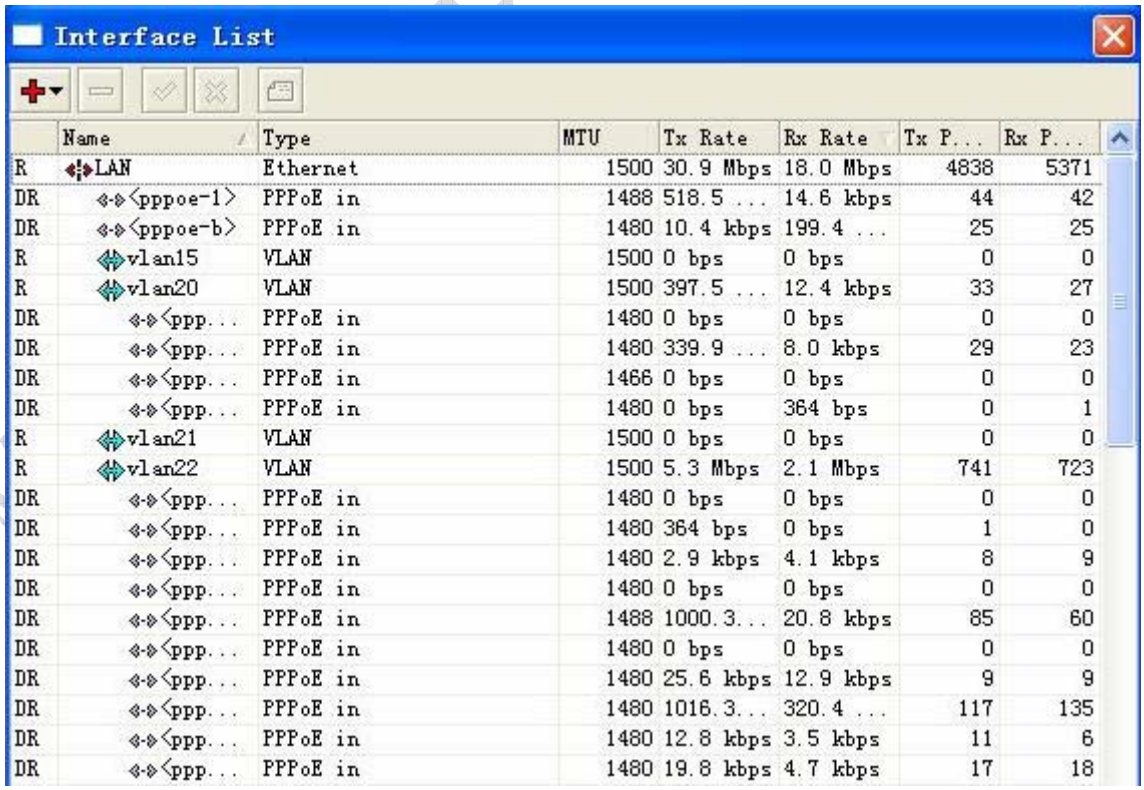
26.3 多 VLAN 下的 PPPoE 服务

在大型局域网络中，会建立多个 VLAN track 隧道，而 PPPoE 服务只能运行在一个局域网中，如果有多个 VLAN 网络，这样给个 VLAN 下的客户就是相互独立的，这样就只能在 RouterOS 中多建立几个基于 VLAN 的 PPPoE 服务器，如下图，建立多个 VLAN 后，在 PPPoE-Server 中对每个 VLAN 建立一个 PPPoE 服务：



Servic...	Inter...	Max MTU	Max MRU	MRRU	Default Profile	Authentication
service1	LAN	1488	1488		default-encryption	pap chap mschap1 ms...
pppoe15	vlan15	1488	1488		default-encryption	pap chap mschap1 ms...
pppoe20	vlan20	1488	1488		default-encryption	pap chap mschap1 ms...
pppoe21	vlan21	1488	1488		default-encryption	pap chap mschap1 ms...
pppoe22	vlan22	1488	1488		default-encryption	pap chap mschap1 ms...
pppoe24	vlan24	1488	1488		default-encryption	pap chap mschap1 ms...
pppoe3	vlan3	1488	1488		default-encryption	pap chap mschap1 ms...
pppoe6	vlan6	1488	1488		default-encryption	pap chap mschap1 ms...
pppoe7	vlan7	1488	1488		default-encryption	pap chap mschap1 ms...

下面是在 interface 中独立 VLAN 下的 PPPoE 运行情况：



	Name	Type	MTU	Tx Rate	Rx Rate	Tx P...	Rx P...
R	LAN	Ethernet	1500	30.9 Mbps	18.0 Mbps	4838	5371
DR	<pppoe-1>	PPPoE in	1488	518.5 ...	14.6 kbps	44	42
DR	<pppoe-b>	PPPoE in	1480	10.4 kbps	199.4 ...	25	25
R	vlan15	VLAN	1500	0 bps	0 bps	0	0
R	vlan20	VLAN	1500	397.5 ...	12.4 kbps	33	27
DR	<ppp...	PPPoE in	1480	0 bps	0 bps	0	0
DR	<ppp...	PPPoE in	1480	339.9 ...	8.0 kbps	29	23
DR	<ppp...	PPPoE in	1466	0 bps	0 bps	0	0
DR	<ppp...	PPPoE in	1480	0 bps	364 bps	0	1
R	vlan21	VLAN	1500	0 bps	0 bps	0	0
R	vlan22	VLAN	1500	5.3 Mbps	2.1 Mbps	741	723
DR	<ppp...	PPPoE in	1480	0 bps	0 bps	0	0
DR	<ppp...	PPPoE in	1480	364 bps	0 bps	1	0
DR	<ppp...	PPPoE in	1480	2.9 kbps	4.1 kbps	8	9
DR	<ppp...	PPPoE in	1480	0 bps	0 bps	0	0
DR	<ppp...	PPPoE in	1488	1000.3...	20.8 kbps	85	60
DR	<ppp...	PPPoE in	1480	0 bps	0 bps	0	0
DR	<ppp...	PPPoE in	1480	25.6 kbps	12.9 kbps	9	9
DR	<ppp...	PPPoE in	1480	1016.3...	320.4 ...	117	135
DR	<ppp...	PPPoE in	1480	12.8 kbps	3.5 kbps	11	6
DR	<ppp...	PPPoE in	1480	19.8 kbps	4.7 kbps	17	18

第二十七章 Web 代理

MikroTik RouterOS 支持下面的代理服务器功能：

- 常规 HTTP 代理
- 透明代理。可以同时透明代理和常规代理
- 源、目的、URL 及请求方法的访问列表
- 缓存访问列表(指定哪些对象需要缓存，哪些不需要)
- 直径访问列表(指定哪些资源应该直接访问，哪些需要通过其他代理服务器)
- 日志功能

设置 1GiB 的 web 缓存，并通过 8000 端口监听，操作如下：

```
[admin@MikroTik] ip proxy> set enabled=yes port=8000 max-cache-size=1048576
[admin@MikroTik] ip proxy> print
               enabled: yes
           src-address: 0.0.0.0
                  port: 8000
          parent-proxy: 0.0.0.0
      parent-proxy-port: 0
           cache-drive: system
    cache-administrator: "webmaster"
        max-cache-size: 1048576KiB
         cache-on-disk: no
  max-client-connections: 600
  max-server-connections: 600
        max-fresh-time: 3d
   serialize-connections: no
      always-from-cache: no
        cache-hit-dscp: 4
[admin@MikroTik] ip proxy>
```

记住保护你的代理，被未验证的用户所访问，这样会变为一个开放的代理。同样你需要设置目标 NAT 启用透明代理功能：

```
[admin@MikroTik] ip firewall nat> add chain=dstnat protocol=tcp dst-port=80 action=redirect
to-ports=8000
[admin@MikroTik] ip firewall nat>
```

规格

功能包需求：：***web-proxy***
许可等级：*Level3*
操作路径：***/ip proxy（winbox：ip web-proxy）***
技术标准：HTTP/1.0, HTTP/1.1, FTP
硬件需求：需要内存和硬盘空间*(具体情况下面的属性)*

这个服务履行代理 HTTP 以及 HTTP 代理（对 FTP，HTTP 及 HTTPS 协议）请求。Web 代理通过存储被请求的因特网对象，以起到网页缓存功能的作用，例如，通过在一个网络数据产生的站点更接近接受者的系统上的 HTTP 及 FTP 协议数据的可用数据。这里“更接近”指的是增加的路径可靠度，或速度或者两者都有。Web 浏览器可以使用本地代理缓存来加快访问并减少带宽消耗。

当设置代理服务时，确定它只为你的客户服务，而不是误用为继电器。请阅读访问列表部分的安全注意。

注意保持 web 代理一直运行，即使当你想使用它作为像 HTTP 及 FTP 防火墙（例如，拒绝访问 mp3 文件）或把请求透明地重定向到外部代理时也没有缓存，这样做会是很有用处的。

属性描述

cache-administrator (*文本*; default: **webmaster**) – 显示在代理错误页面的管理员 e-mail
cache-drive (system | *name*; default: **system**) -指定用于存贮缓存对象的目标磁盘机。你可以使用控制台完成来查看可用驱动器列表
cache-only-on-disk (yes | no; default: **yes**) -是否在描述磁盘上缓存目录的内存中创建的数据库。这样会减少内存消耗，但会影响速度
enabled (yes | no; default: **no**) - 代理服务器是否启用
max-disk-cache-size (none | unlimited | *整型*: 0..4294967295; default: **none**) -指定最大磁盘缓存大小，以 kb 计算
max-fresh-time (*时间*; default: **3d**) - 存贮缓存对象的最大时间。一个目标的合法时间一般是由对象本身定义的，但以防太长，你可以覆盖最大值
maximal-client-connecions (*整型*; default: **1000**) -客户接受的最大连接数（任何更多的连接都将被拒绝）
maximal-server-connectons (*整型*; default: **1000**) - 到服务器的最大连接数（任何更多来自客户的连接都将被挂起知道一些服务器连接结束）
max-object-size (*整型*; default: **2000KiB**) - 大于指定长度的对象将不会保存在磁盘上。以 kb 计算。如果你想获得一个更高的比特命中率，你应该增加该值（一个 2MiB 对象撞击代表 2048 个 1KiB 撞击）。如果你更想增加速度而不是接生带宽，你应该把这个值设的低一些
max-ram-cache-size (none | unlimited | *整型*: 0..4294967295; default: **none**) -指定最大 RAM 缓存大小，以 kb 计算
parent-proxy (*IP address*:*port*; default: **0.0.0.0:0**) - 把所有请求定向到的 IP 地址及其他 HTTP 代理端口（异常会在"direct access"列表中定义）
0.0.0.0:0 – 没有使用父级代理
port (*port*; default: **8080**) -代理服务器将监听的 TCP 端口。这个会在所有想使用该服务器作为 HTTP 代理的客户上定义。透明（对客户使用零配置）代理设置可以通过使用目的 NAT 特性在 IP 防火墙重定向 HTTP 请求到该端口完成
src-address (*IP address*; default: **0.0.0.0**) - Web 代理将使用这个地址连接父级代理或 web 站点
0.0.0.0 – 合适的 **src-address** 将会自动从路由列表中取出

注： 这个 web 代理监听所有路由器 IP 地址列表中包含的 IP 地址。

在端口 8000 上启用代理：

```
[admin@MikroTik] ip proxy> set enabled=yes port=8000
[admin@MikroTik] ip proxy> print
                  enabled: yes
              src-address: 0.0.0.0
                     port: 8000
            parent-proxy: 0.0.0.0:0
             cache-drive: system
```

```
       cache-administrator: "dmitry@mikrotik.com"
       max-disk-cache-size: none
        max-ram-cache-size: 100000KiB
        cache-only-on-disk: yes
  maximal-client-connections: 1000
  maximal-server-connections: 1000
          max-object-size: 2000KiB
           max-fresh-time: 3d
[admin@MikroTik] ip proxy>
```

27.1 访问列表

操作路径: ***/ip proxy access***

访问列表像普通防火墙规则一样配置。规则从顶到底的处理。第一条匹配的规则指定对连接做何处理。一共有 6 个指定匹配显示的分类器。如果没有指定其中任何一个，那么特定规则将与每一条连接进行匹配。

如果连接被一条规则匹配，该规则的 **action** 属性就指定是否连接应被允许。如果特定连接没有匹配任何规则，那么它将被允许。

属性描述

action (allow | deny; default: **allow**) -指定通过或拒绝已匹配的包
dst-address (*IP address/netmask*) - IP 包的目的地址
dst-host (*wildcard*) - IP 地址或用于连接目标服务器的 DNS 名（这是一个在指定端口与到特定网址路径之前写在他的浏览器的字符串）
dst-port (*port*{1,10}) - 包到达的列表或端口范围
hits (*只读: 整型*) - 被规则修正的请求数
local-port (*port*) -指定包接受的 web 代理端口。这个值应该匹配 web 代理监听的其中一个端口
method (any | connect | delete | get | head | options | post | put | trace) -用于请求的 HTTP 方法（参见本文档最后面的 HTTP 方法部分）
path (*wildcard*) -在目标服务器中的被请求页面名（例如，特定网页的名称或不含它存在的服务器名称的文档）
redirect-to (*文本*) - 以防访问被该规则拒绝，用户应被重定向到这里指定的 URL
src-address (*IP address/netmask*) - IP 包的源地址

注： 统配符属性（**dst-host** 和 **dst-path**）匹配一个完整的字符串（例如，如果设置为"example"，则他们不会匹配"example.com"）。可用的统配符'*'（匹配任何数量的任何字符）以及'?'（匹配任何一个字符）。这里也接受常规表达，但是如果属性被当作常规表达处理，那就应该以冒号(':')开始。

在常规表达式中的低命中：

- **** 符号顺序用于在控制台中输入****字符
- **\.** 样式仅表示 **.** (在常规表达式中单独一个点表示任何符号)
- 表示在给定样式之前不允许任何符号，我们在样式的开头使用**^**符号
- 指定在给定样式之后不允许任何符号，我们在样式的结尾使用**$**
- 输入**[** or **]**符号，你可以用反斜杠对它们转义

强烈建议拒绝所有 IP 地址除了在路由器之后的那些，因为代理仍然可以访问你的 internal-use-only（企业网）web 服务器。在 Firewall Manual 中查询如果保护你的路由器。

27.2 直接访问列表

操作路径: ***/ip proxy direct***

如果指定了 **parent-proxy** 属性，就很可能告诉代理服务器是否尝试通过请求到父级代理或通过直接连接到被请求的服务器以解决问题。直接访问列表就像前一章节描述的代理访问列表一样管理，除了 **action** 参数。

属性描述

action (allow | deny; 默认: **allow**) -指定对已匹配包的动作
  allow -总是直接绕过父级路由器解决匹配的请求
  deny - 通过父级代理以解决匹配请求。如果没有指定则这个与 **allow** 的效果相同
dst-address (*IP address/netmask*) - IP 包的目的地址
dst-host (*wildcard*) - 用于连接到目标服务器的 IP 地址或 DNS 名（这是在指定特定网页到达的端口与路径之前用户写在他的浏览器中的字符串）
dst-port (*port*{1,10}) -包到达的列表或端口范围
hits (*只读: 整型*) -被规则修正过的请求数
local-port (*port*) -指定包接受的 web 服务器端口。这个值应该与 web 代理监听的其中一个匹配
method (any | connect | delete | get | head | options | post | put | trace) - 用于请求中的 HTTP 方法(参见本文档最后的 HTTP 方法部分)
path (*wildcard*) - 目标服务器中的被请求页面名（例如，特定 web 页面名或不含它存在的服务器名的文档）
src-address (*IP address/netmask*) - IP 包的源地址

注： 不像访问列表，直接代理访问列表有与 **deny** 相等价的默认动作。当没有规则指定或一个特定请求没有匹配任何规则时发生。

27.3 缓存管理

操作路径: ***/ip web-proxy cache***

缓存访问列表指定哪个请求（域、服务器、页面）应该由 web 代理本地缓存，而哪个不用。这个列表与 web 代理访问列表完全一样地执行。

属性描述

action (allow | deny; 默认: **allow**) - 指定对已匹配包的动作
  allow – 允许请求缓存对象
  deny – 不允许请求缓存对象
dst-address (*IP 地址*/子网掩码) - IP 包的目的地址
dst-host (*wildcard*) - 用于连接到目标服务器的 IP 地址或 DNS 名（这是在指定特定网页到达的端口与路径之前用户写在他的浏览器中的字符串）
dst-port (端口{1,10}) -包到达的列表或端口范围
hits (*只读: 整型*) - 被规则修正过的请求数
local-port (端口) -指定包接受的 web 服务器端口。这个值应该与 web 代理监听的其中一个匹配

method (any | connect | delete | get | head | options | post | put | trace) - 用于请求中的 HTTP 方法(参见本文档最后的 HTTP 方法部分)
path (*wildcard*) - 目标服务器中的被请求页面名（例如，特定 web 页面名或不含它存在的服务器名的文档）
src-address (*IP 地址/子网掩码*) - IP 包的源地址

27.4 代理监视

命令名: ***/ip proxy monitor***

这个命令显示代理服务器的一些状态

属性描述

cache-used (*只读: 整型*) - 用于缓存的磁盘空间
hits (*只读: 整型*) – 在缓存中找到并开始被服务的请求数
hits-sent-to-clients (*只读: 整型*) – 由缓存服务的数据量
ram-cache-used (*只读: 整型*) – 用于存贮缓存的 RAM 空间
received-from-servers (*只读: 整型*) – 从其他服务器接受的数据量
requests (*只读: 整型*) – 已处理的请求量
sent-to-clients (*只读: 整型*) – 发送到该代理服务器客户的数据量
status (*只读: 文本*; default: **stopped**) – 显示代理服务器的状态信息
  stopped – 代理被禁用且没有运行
  rebuilding-cache – 代理被启用并运行，存在的缓存被核实
  running -代理被启用并运行
  stopping - 代理关闭（最大 10s）
  clearing-cache – 代理停止，缓存文件被删除
  creating-cache – 代理停止，缓存目录结构被创建
  dns-missing – 代理被启用，但没有运行因为未知的 DNS 服务器（你应该在**/ip dns** 指定）
  invalid-address -代理被启用，但没有运行，因为非法的地址（你应该改变地址或端口）
  invalid-cache-administrator -代理被启用，但没有运行因为非法的缓存管理员的 e-mail 地址
  invalid-hostname – 代理被启用，但没有运行因为非法的主机名（你应该设置一个合法的主机名）
  error-logged – 主机没有运行因为未知的错误。这个错误会被日志标记为系统错误。请发把错误、描述以及如何发生的送给我们
  reserved-for-cache (整型) – 最大缓存大小，可以访问 web 代理
total-ram-used (*只读: 整型*) – 用于代理的总 RAM 大小
uptime (*只读: 时间*) - 代理最近一次启动后的时间

27.5 连接列表

操作路径: ***/ip proxy connections***

这个目录包含代理存储的当前连接的列表。

属性描述

dst-address (*只读: IP 地址*) – 连接的 IP 地址
protocol (*只读: 文本*) – 协议名
rx-bytes (*只读: 整型*) - 客户接收的字节量

src-address (*只读: IP 地址*) – 连接源发站的 IP 地址
state (*只读:* closing | connecting | converting | hotspot | idle | resolving | rx-header | tx-body | tx-eof | tx-header | waiting |) – 打开连接的状态
closing - 数据传输完成，连接正在最终完成
connecting - 建立 toe 连接
hotspot – 检查是否 hotspot 认证允许继续（对 hotspot 代理）
idle - 闲置状态
resolving – 分辨服务器的 DNS 名
rx-header – 接受 HTTP 标题
tx-body – 传输 HTTP 正文给客户
tx-eof - 写组块端(当传换为分组的回应)
tx-header – 传输 HTTP 标题给客户
waiting – 等待来自同等体的传输
tx-bytes (*只读: 整型*) - 由客户发送的字节数

27.6 缓存插页

操作路径: ***/ip proxy inserts***

这个目录显示存储在缓存中的对象的统计数据(缓存插页)

属性描述

denied (*只读: 整型*) - 被缓存列表拒绝的插页数
errors (*只读: 整型*) – 磁盘或其他系统相关的错误数量
no-memory (*只读: 整型*) – 由于没有足够内存而没有存贮的对象数量
successes (*只读: 整型*) - 成功缓存插页的数量
too-large (*只读: 整型*) – 过大而不能存储的对象数量

27.7 缓存查检

操作路径: ***/ip proxy lookups***

这个目录相识从缓存读取的对象的统计数据（缓存查检）

属性描述

denied (*只读: 整型*) - 被访问列表拒绝的请求数
expired (*只读: 整型*) -在缓存中发现的过期请求数，这样将会被外部服务器请求
no-expiration-info (*只读: 整型*) - 接受没有能与请求相比较的信息的页面的有条件请求
non-cacheable (*只读: 整型*) - 来自外部服务器的无条件请求数（由于它们的缓存被缓存访问列表拒绝了）
not-found (*只读: 整型*) -没有在缓存中发现的请求数，这样将被一个外部服务器请求（或者是父级代理，如果进行过相应的配置）
successes (*只读: 整型*) - 在缓存中发现的请求数

补充工具

操作路径: ***/ip proxy***

web 代理有附加的命令来处理用于缓存目的非系统驱动器和从严重的文件系统错误中恢复代理。

check-drive - 检查非系统缓存驱动器的错误
clear-cache - 删除存在缓存并建立新缓存目录
format-drive -格式化非系统缓存驱动器并为容纳缓存做准备

27.8 HTTP 方式

OPTIONS

这个方法是一个关于客户与 **Request-URI** 定义的服务器之间的链上的可用通信选项信息的请求。这个方法允许客户决定选项以及（或）与没有初始化任何资源检索的资源相关的需求。

GET

这个方法检索 **Request-URI** 定义的任何星系。如果 **Request-URI** 设计数据处理过程然后 **GET** 方法的回应应该包含处理产生的数据而不是处理过程的源代码，除非源是这个处理的结果。

如果请求信息包含一个 **If-Modified-Since**，**If-Unmodified-Since**，**If-Match**， **If-None-Match** 或 **If-Range** 标题字段，那么 **GET** 方法就会成为*有条件的* **GET**。有条件的 **GET** 方法用于通过指定该实体的传送应该仅在有条件标题字段描述的环境下发生来减少网络流量。

如果请求信息包含 **Range** 标题字段，那么 **GET** 方法就会成为一个*部分* **GET**。这个部分 **GET** 方法通过只请求不传送已被客户包含的数据的实体的一部分来减少不必要的网络使用。

当且仅当达到 HTTP 缓存要求时一个 **GET** 请求的回应为可缓存的。

HEAD

这个方法共享 **GET** 方法的所有特征除了服务器不一定必须在回应中返回一个信息体。它检索蕴涵在导致广泛应该于测试合法的超文本连接，可访问性，及最近修改的请求中的实体的元信息。

HEAD 请求的回应可能以这样的途径仍为可缓存的：包含于响应的信息可能用于更新先前缓存的被 **Request-URI** 识别的实体。

POST

这个方法需要起源服务器接受包含在请求中的实体，就像 **Request-URI** 定义的新的下级资源一样。

POST 方法执行的实际动作由起源服务器判定，并且通常是依赖 **Request-URI** 的。

POST 方法的回应不可缓存，除非回应包含合适的 **Cache-Control** 或 **Expires** 标题字段。

PUT

这个方法需要被包含的实体存储在提供的 **Request-URI** 中。如果另一个实体存在于指定的 **Request-URI** 中，被包含的实体应该被认为是存在于起源服务器上的新版本。如果 **Request-URI** 没有指向一个存在的资源，那么起源服务器应该创建一个有 URI 的资源。

如果请求通过了一个缓存并且 **Request-URI** 识别了一个或多个当前缓存的实体，那么这些实体就应作为过时的处理。这个方法的回应不可缓存。

TRACE

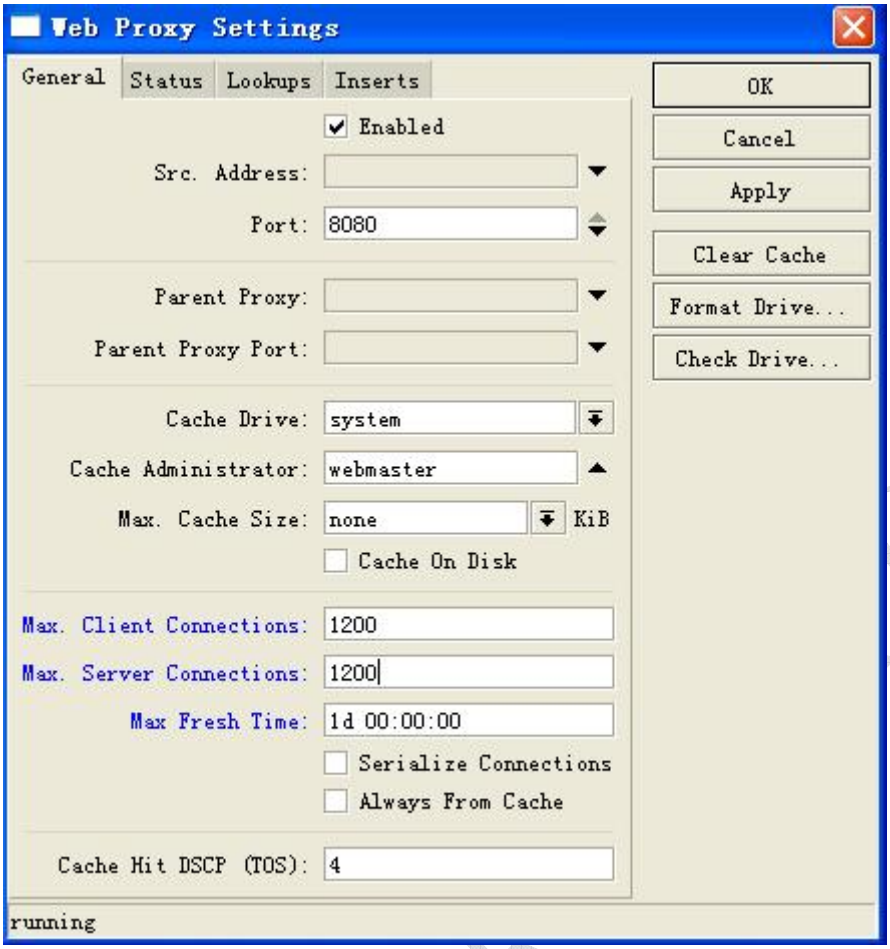
这个方法调用一个远程的，请求信息的应用层循环。最终的请求接受者应该把接受到的信息作为一个 200（OK）回应的实体正文反射给客户。最终接受者是起源服务器或者第一个代理或者在请求中接受 0 值的 **Max-Forwards** 的网关。一个 **TRACE** 请求不一定要包含一个实体。

27.9 Web 代理应用事例

通过使用 web-proxy 禁止网站和禁止下载

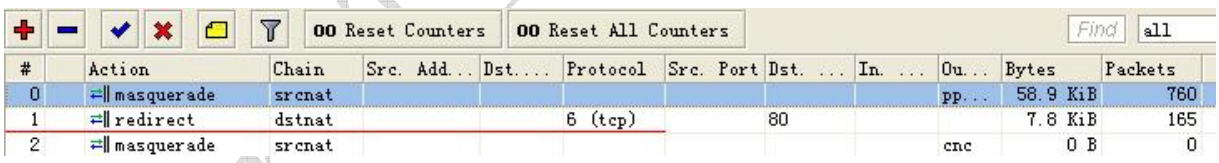
首先配置 web-proxy，配置参数如下：

```
[admin@MikroTik] /ip proxy> prin
              enabled: yes
          src-address: 0.0.0.0
                 port: 8080
         parent-proxy: 0.0.0.0
     parent-proxy-port: 0
          cache-drive: system
   cache-administrator: "webmaster"
       max-cache-size: none
        cache-on-disk: no
 max-client-connections: 1200
 max-server-connections: 1200
       max-fresh-time: 1d
  serialize-connections: no
     always-from-cache: no
       cache-hit-dscp: 4
```



现在，设置透明传输数据重定向，将所有访问 80 端口的数据重定向到 web-proxy 的 8080 端口上：

```
/ip firewall nat
chain=dstnat protocol=tcp dst-port=80 action=redirect to-ports=8080
```



确定你路由器本地的 Proxy 没有打开代理，并禁止外网通过路由器代理上网：

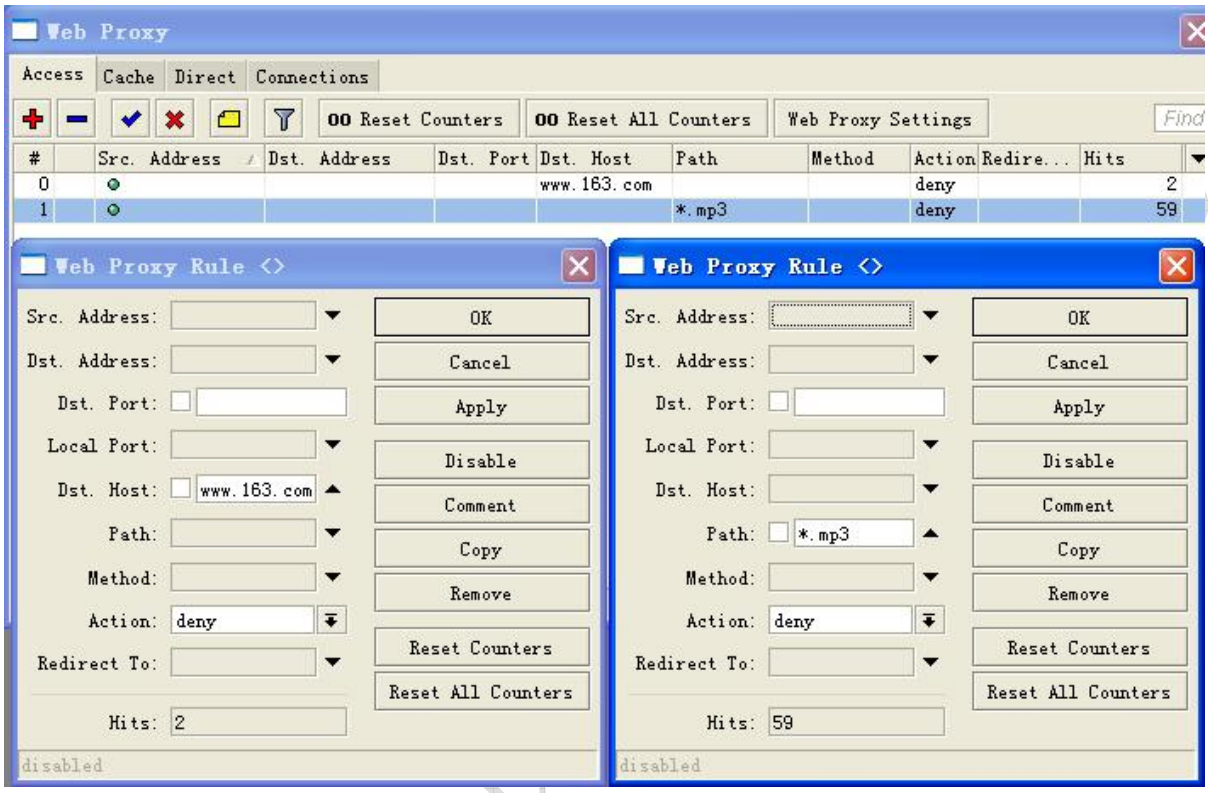
```
/ip firewall filter
chain=input in-interface=<Your WAN Port> src-address=0.0.0.0/0 protocol=tcp dst-port=8080
action=drop
```

设置禁止访问网站，该设置将禁止访问 http://www.163.com

```
/ip proxy access
dst-host=www.163.com action=deny
```

我们可用阻止文件如“.mp3, .exe, .dat, .avi”的下载。

```
/ip proxy access
path=*.exe action=deny
path=*.mp3 action=deny
path=*.zip action=deny
path=*.rar action=deny
```



同样我可用阻止所有含“mail”的关键字链接

```
/ip proxy access
dst-host=:mail action=deny
```

第二十八章 MetaRouter

RouterOS v3.30 前是 Xen 虚拟机，但在 v3.30 后用 KVM 替代了 Xen 虚拟机，RouterOS v4.0 现在支持 2 种不同的虚拟化技术分别是：MetaRouter 和 KVM，

Metarouter

MetaRouter 是 MikroTik 开发的，当前仅支持 RouterBOARD 400 系列(mips-be)，也只能创建 RouterOS 的虚拟机。 MikroTik 计划添加更多的功能到 MetaRouter 中，因此新的硬件支持将会添加到 MetaRouter 中，甚至会超过 Xen 的功能。

Xen

Xen 是基于 Linux Xen 虚拟机项目，应用于当前的 RouterOS x86 系统（PC），Xen 虚拟机能创建不同的操作系统。

KVM

Kernel-based Virtual Machine（KVM）提供虚拟基于 x86 的技术。为 RouterOS 基于 PC 主机提供完善的虚拟技术，KVM 要求支持 CPU 虚拟技术，如 Intel 的 VT-x 或者 AMD-V 等技术。运行要求虚拟机至少分配 16MB 内存，有足够的硬盘空间运行相应的镜像文件。镜像文件不能在创建后被增加，并且大小只能是你导入创建时的镜像不变。

28.1 虚拟化技术的应用

下面是一些虚拟机的可行方案（一些方案现在只指出 Xen，但 MetaRouter 将会添加更多的功能）：

数据管理中心

- 加强了一些路由器的硬件平台
- 加强路由器的服务，并更高等级的服务器如 VOIP 交换在同一台设备上
- 使用客户机上的一个路由器为定制功能，例如日志记录、LDAP 或者传统网络
- 冗余路由器更加简单和便宜

托管中心

- 通过 RouterOS 的虚拟化技术配合各种网络功能，如各种服务 Mail、Http、Ftp 等
- 提供虚拟路由器的 VPN 解决方案，这样能使网络管理员拥有自己的路由器，在高速骨干网络建立各种隧道或者 VPN 访问系统

无线 ISP 客户端

- 设置两个独立的路由器，并设置 WISP 的无线控制，并交由以太网端的客户进行控制

多客户端（例如办公楼）

- 分布在多个点的客户从一个骨干以太网连接(有线或无线)，让每个客户能控制自己的独立的虚拟路由器，并配置自己的办公的路由。

网络规划与测试

- 建立一个虚拟的网络在一台设备上，相同环境的可对一个网络测试计划配置，进行微调，起到到在实验室的作用，而不需要在外假设，通过脚本和 The Dude 网络管理起模拟和监测网络

28.2 MetaRouter 介绍

MetaRouter 是 RouterOS 从 4.0beta1 和 3.21 版本开始新增加的功能，当前 MetaRouter,只能用于 RB400 系列，用于创建虚拟机，在以后会有更多的硬件平台增加此功能。每一个 Metarouter 是使用设备相同的资源，建立独立的 RouterOS 系统。每一个 Metarouter 至少需要 16M 的 RAM。16M 是绝对最小的值，建议为每一个 Metarouter 使用更大的 RAM。

当前可以创建 8 个 Metarouter 虚拟机，将来新的版本会增加到 16 个。在主设备上，你可以创建 8 个虚拟接口连接到 MetaRouter，唯一可以增加接口的方式只能通过 VLAN。现在 MetaRouter 虚拟机还不能支持外部存储设备。

MetaRouter 功能常用于允许客户或者低特权用户访问自己的“路由”，并根据他们需要自己配置参数，这样不需要另外一个真实的路由器。例如：一个 ISP 能创建一个虚拟路由器，允许特定的用户通过以太接口访问，并定义他们自己的防火墙规则，但只有又不会影响主设备的运行

在/metarouter 目录下给出了一下命令：

- add – 允许你创建一个新的虚拟路由器
- print – 通过列表显示当前所有虚拟路由器
- enable – 启用一个虚拟路由器
- disable – 禁用一个虚拟路由器
- console – 访问一个虚拟路由器的控制台
- interface – 映射相应的网络接口

创建一个 MetaRouter

```
[admin@RB_Meta] /metarouter> add name=mr0 memory-size=32 disk-size=32000  disabled=no
 [admin@RB_Meta] /metarouter> print
 Flags: X - disabled
 #   NAME              MEMORY-SIZE DISK-SIZE    USED-DISK    STATE
 0   mr0               16MiB      0kiB         377kiB       running
```

- **name**: 虚拟路由器的名称
- **memory-size**: 分配给虚拟路由器的 RAM 大小
- **disk-size**: HDD 的容量，通过 KB 分配给虚拟路由器 （如果设置为 **0**, 容量默认为动态分配) *
- **used-disk**: 当前使用的硬盘空间 currently used disk space
- **state**: MetaRouter 运行的状态

注意：MetaRouter 在使用的动态 HDD 空间时，启用代理功能会占用你所有的 HDD 存储！

默认配置

如果你添加一个新的 MetaRouter 没有指定任何参数，默认会添加动态的 HDD 长度，和 16M 的 RAM：

```
[admin@RB_Meta] /metarouter> add name=mr1
[admin@RB_Meta] /metarouter> print
Flags: X - disabled
 #   NAME              MEMORY-SIZE DISK-SIZE    USED-DISK    STATE
 1   mr1               16MiB      0kiB         3kiB         running
```

添加接口

首先需要添加一个新的接口到你的虚拟路由器上，这个操作在 Interface 目录完成，Interface 命令如下面：

```
 [admin@MikroTik] /metarouter> interface add
```

```
comment    disabled       dynamic-mac-address  type     virtual-machine
copy-from  dynamic-bridge  static-interface     vm-mac-address
```

我们添加一个接口：

```
[admin@MikroTik] /metarouter> interface add virtual-machine=mr1 type=dynamic
```

在物理路由器的 interface 出现一个虚拟接口：

```
 [admin@MikroTik] > /interface print
 Flags: D - dynamic, X - disabled, R - running, S - slave
 #    NAME                                      TYPE           MTU
 8  R  ether9                                    ether          1500
 9  R  test                                     bridge         1500
10 DR  vif1                                     vif            1500
```

连接虚拟机

连接你的虚拟机，使用 console 命令:

```
/metarouter console 0
```

你可以看到你最新添加的虚拟接口：

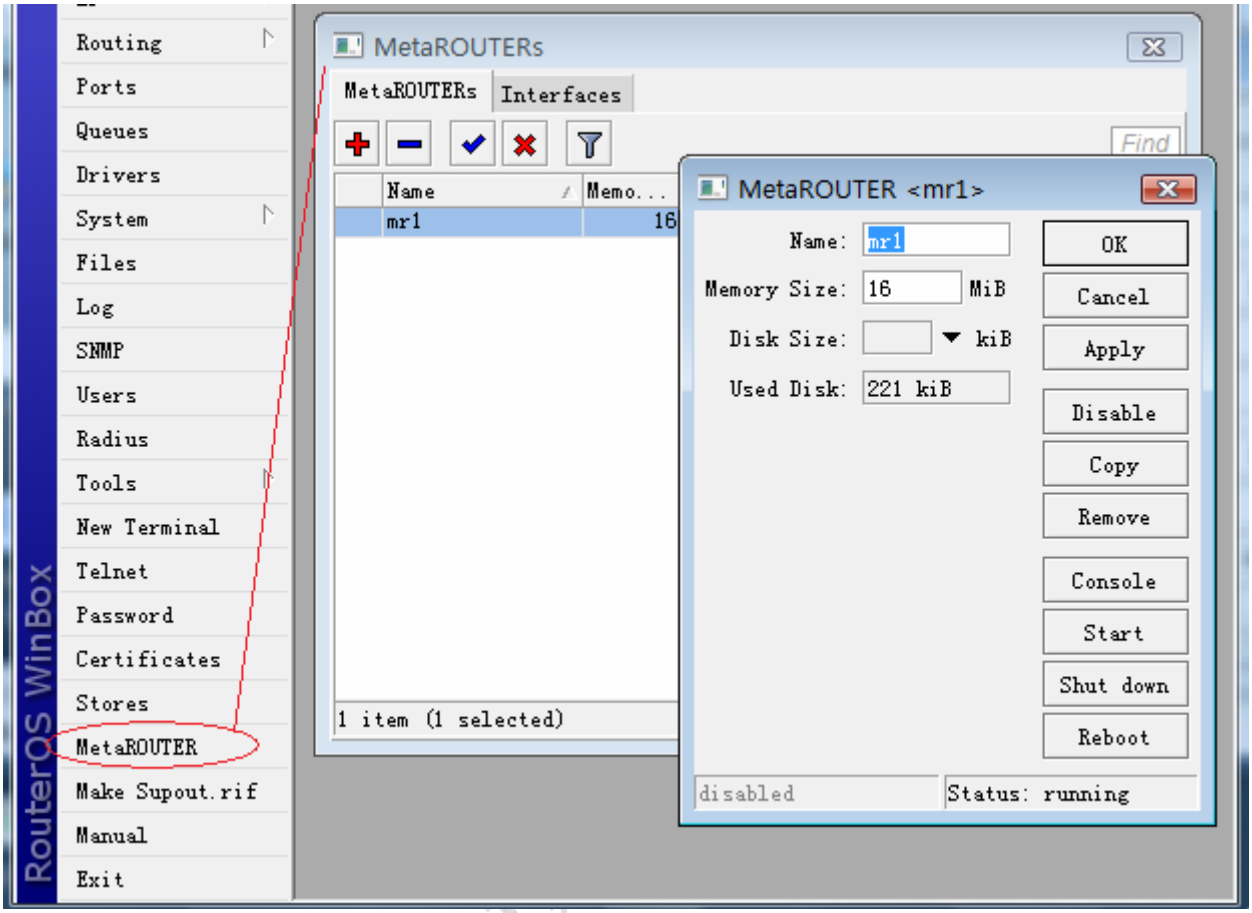
```
[admin@mr0] > interface print
 Flags: D - dynamic, X - disabled, R - running, S - slave
 #    NAME                                      TYPE           MTU
 0  R  ether1                                    ether          1500
```

从 MetaRouter 的虚拟机控制台断开，按 CTRL + A 和 Q 退回到物理路由器：

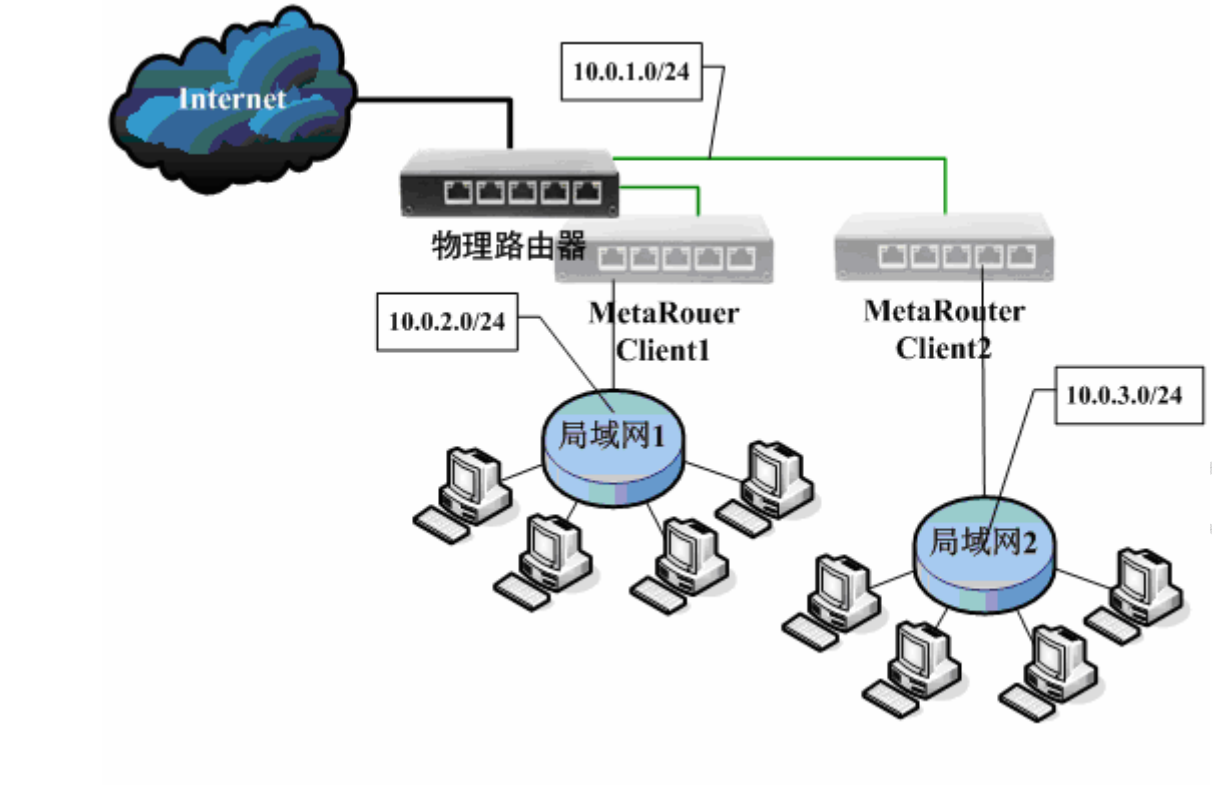
```
[admin@MikroTik] >
[Q - quit connection]     [B - send break]
[A - send Ctrl-A prefix]   [R - autoconfigure rate]

 Q

 Welcome back!
```

28.3 MetaRouter 事例

现在你看到之前添加的虚拟接口在物理路由器的 interface 目录中显示为 vif1，当然在 metarouter 的接口中显示为 ether1，你可以在 2 个接口上配置 IP 地址，并连接网络。创建一个 bridge 在允许传输的物理接口和虚拟接口上。下面是 winbox 操作界面

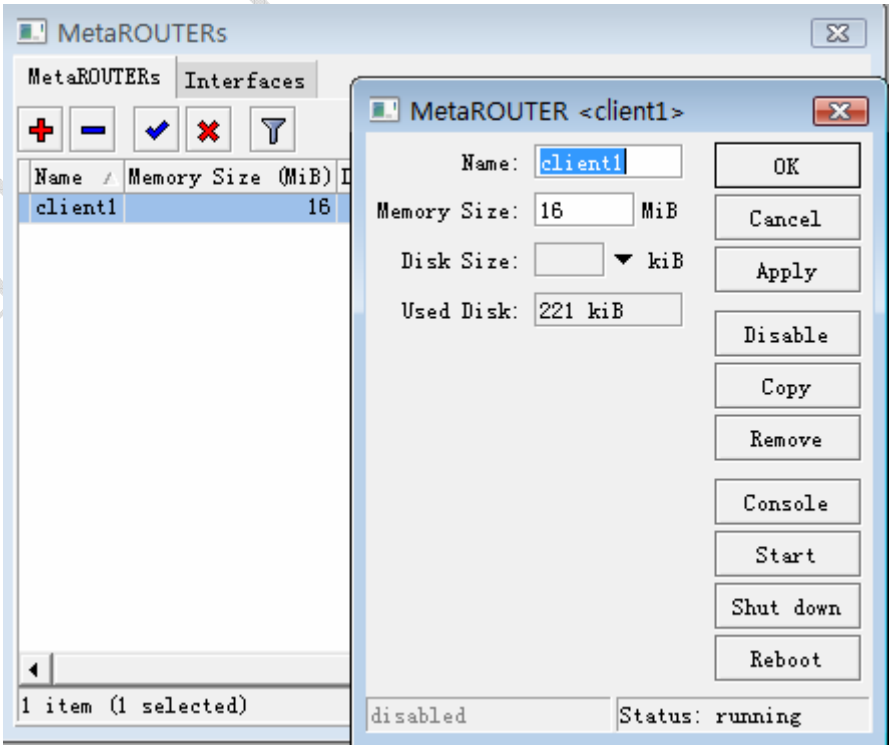


这个事例将介绍如何配置 MetaRouter 功能，为局域网内部独立 RouterOS 虚拟路由，基于 RB450 配置 MetaRouter，我们将建立两个 Client 虚拟路有，并管理两个不同的局域网。目的是让两个个局域网的管理员可以管理自己的路由器（MetaRouter），并根据自己的需要配置他们自己的防火墙、流量控制和 nat 规则，当然他们不能连接物理路由器（没有分配权限）：



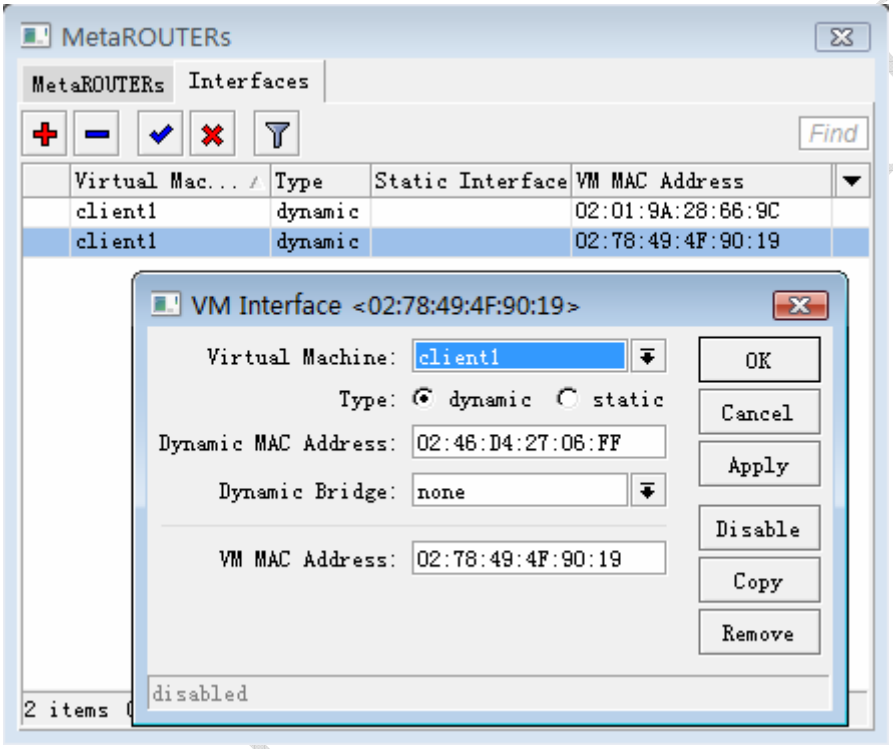
1.给客户添加一个 MetaRouter：

```
[admin@CDNAT] /metarouter> add name=client1 memory-size=16
[admin@CDNAT] /metarouter> print
Flags: X - disabled
 #   NAME                MEMORY-SIZE DISK-SIZE     USED-DISK     STATE
 0   client1               16MiB       0kiB          221kiB      running
[admin@CDNAT] /metarouter>
```



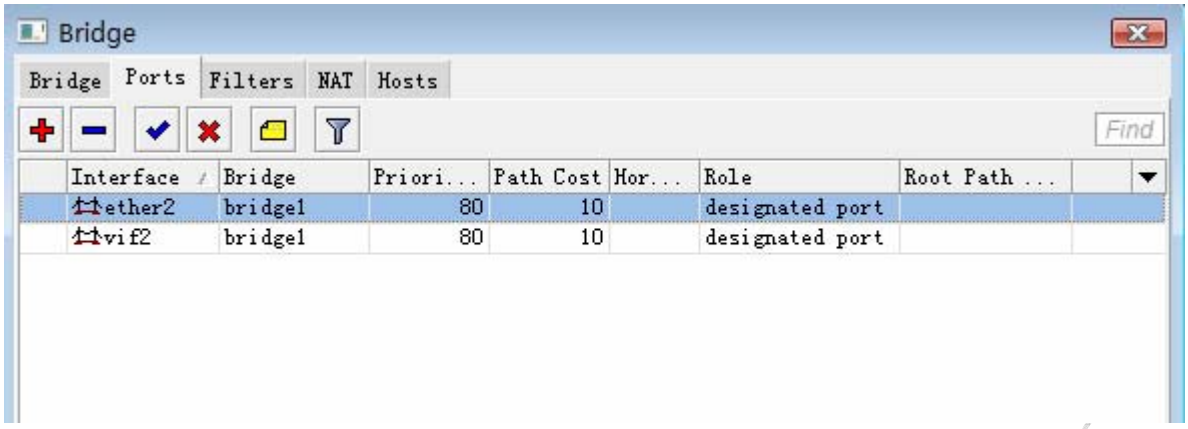
2. 添加 MetaRouter 虚拟机的接口：

```
[admin@CDNAT] /metarouter interface> add virtual-machine=client1
[admin@CDNAT] /metarouter interface> add virtual-machine=client1
[admin@CDNAT] /metarouter interface> print
Flags: X - disabled, A - active
 #   VIRTUAL-MACHINE                          TYPE      VM-MAC-ADDRESS
 0 A client1                                  dynamic   02:01:9A:28:66:9C
 1 A client1                                  dynamic   02:78:49:4F:90:19
[admin@CDNAT] /metarouter interface>
```



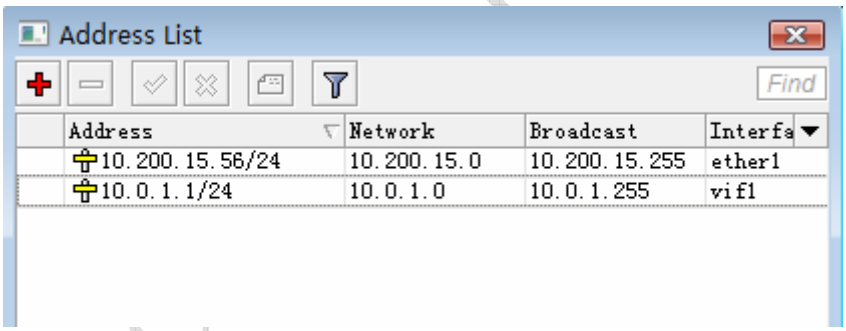
3. 创建一个桥接口，将 MetaRouter 接口与以太网接口桥接，这里我们将 vif2 的虚拟接口放入内网的桥中，用于客户端通过物理接口连接(使用桥接目的是将虚拟路由的内网接口，通过桥接透穿到真实的网络中)：

```
[admin@CDNAT] /interface bridge> add
[admin@CDNAT] /interface bridge> print
Flags: X - disabled, R - running
 0  R name="bridge1" mtu=1500 arp=enabled mac-address=00:00:00:00:00:00 protocol-mode=none
priority=0x8000 auto-mac=yes admin-mac=00:00:00:00:00:00 max-message-age=20s
forward-delay=15s transmit-hold-count=6 ageing-time=5m

[admin@CDNAT] /interface bridge port> add interface=ether2 bridge=bridge1
[admin@CDNAT] /interface bridge port> add interface=vif2 bridge=bridge1
[admin@CDNAT] /interface bridge port> print
Flags: X - disabled, I - inactive, D - dynamic
 #    INTERFACE        BRIDGE              PRIORITY  PATH-COST  HORIZON
 0    ether2           bridge1               0x80      10         none
 1    vif2             bridge1               0x80      10         none
```



4. 为新的 MetaRouter 接口添加 IP 地址，ether1 作为物理外网连接（假设我们已经配置好物理路由器的网络连接），vif1 用于连接 MetaRouter 主机系统（vif1 可以认为是一个 lan 接口连接）：

```
[admin@CDNAT] /ip address> add address=10.0.1.1/24 interface=vif1
[admin@CDNAT] /ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS             NETWORK        BROADCAST      INTERFACE
 0   10.200.15.56/24     10.200.15.0    10.200.15.255    ether1
 1   10.0.1.1/24         10.0.1.0       10.0.1.255       vif1
[admin@CDNAT] /ip address>
```



5. 进入 metarouter 控制平台，通过 console 命令：

```
[admin@CDNAT] /metarouter> console client1

[Ctrl-A is the prefix key]

Starting...
Starting services...

MikroTik 3.22
MikroTik Login: admin
Password:

[admin@MikroTik] > /sys identity set name=Client1
```

6. 配置 metarouter 的参数，设置以太网接口名称，让客户明白设备的连接情况：

```
 [admin@Client1] /interface ethernet> print
Flags: X - disabled, R - running, S - slave
 #    NAME                MTU    MAC-ADDRESS        ARP
 0 R  ether1              1500  02:49:E8:55:8E:E8   enabled
 1 R  ether2              1500  02:16:16:90:EF:0E   enabled
[admin@Client1] /interface ethernet> set 0 name=wan
[admin@Client1] /interface ethernet> set 1 name=lan
[admin@Client1] /interface ethernet> print
Flags: X - disabled, R - running, S - slave
 #    NAME                 MTU   MAC-ADDRESS        ARP
 0 R  wan                1500  02:49:E8:55:8E:E8  enabled
 1 R  lan                1500  02:16:16:90:EF:0E  enabled
[admin@Client1] /interface ethernet>
```

为外网和内网接口配置 IP 地址

```
[admin@Client1] /ip address> add address=10.0.1.2/24 interfae=wan
[admin@Client1] /ip address> add address=10.0.2.1/24 interface=l
[admin@Client1] /ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS           NETWORK        BROADCAST      INTERFACE
 0   10.0.1.2/24       10.0.1.0       10.0.1.255     wan
 1   10.0.2.1/24       10.0.2.0       10.0.2.255     lan
```

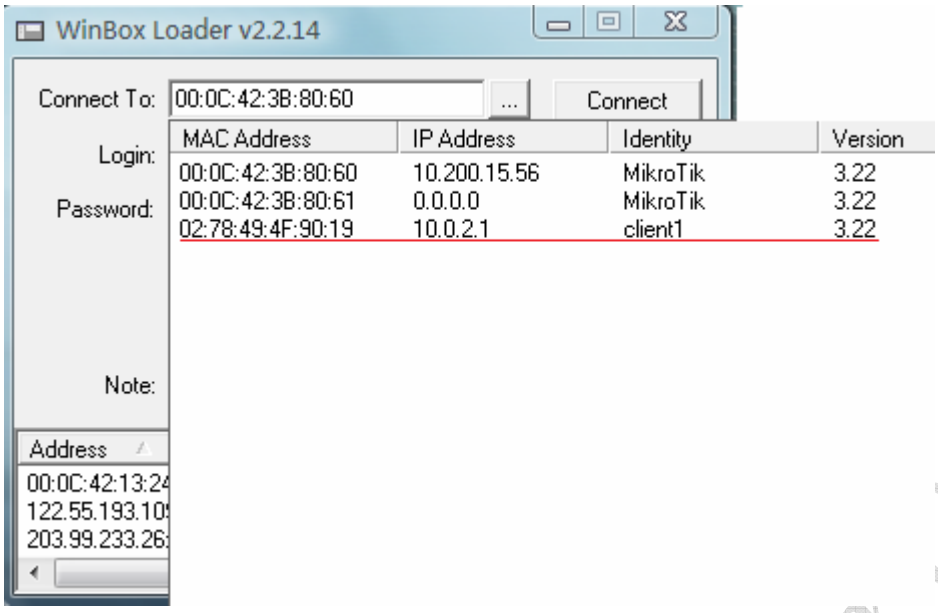
添加默认网关

```
[admin@Client1] /ip route> add gateway=10.0.1.1
[admin@Client1] /ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #    DST-ADDRESS       PREF-SRC    G GATEWAY          DISTANCE  INTERFACE
 0 A S  0.0.0.0/0      r 10.0.1.1                       1         wan
 1 ADC  10.0.1.0/24      10.0.1.2                       0         wan
 2 ADC  10.0.2.0/24      10.0.2.1                       0         lan
[admin@Client1] /ip route>
```

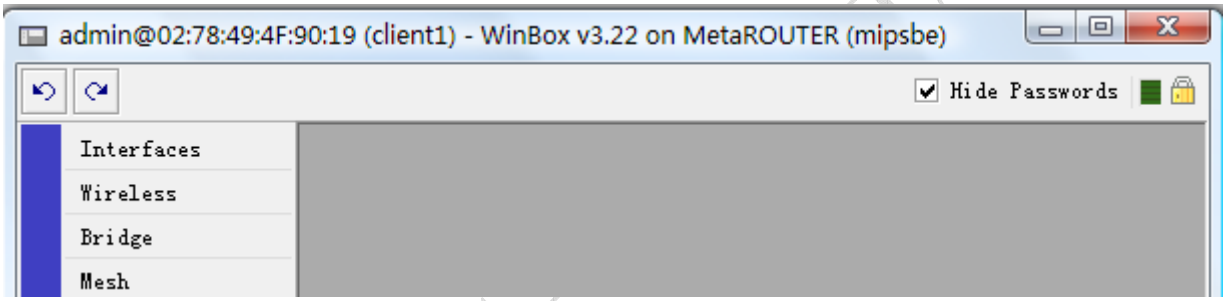
配置 nat 转换

```
[admin@Client1] /ip firewall nat> add action=masquerade out-interface=wan chain=srcnat
```

配置完成后，我们可以通过局域网的 winbox 扫描到配置好的 metarouter

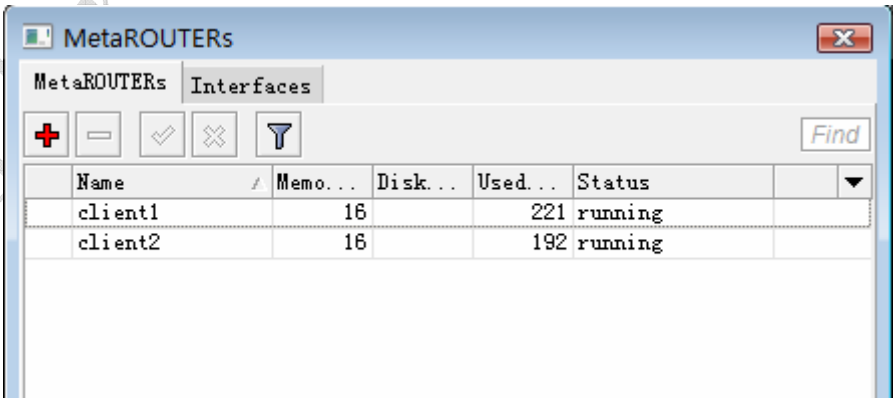


通过连接后，在 winbox 中显示 MetarRouter 信息



这样我们可以通过局域网内部，连接虚拟的 RouterOS 上网。这样我们可以让局域网 1 的管理进入 client1 的 MetaRotuer 配置自己的网络参数。

我们在用同样的方法建立 10.0.3.0/24 网络的第二个 MetaRouter 在 RouterBOARD 上，只要硬件性能允许，为不同客户提供多个自主路由器管理：



第二十九章 Log 日志管理

不同的系统事件和状态信息都能被 RouterOS 的 log 记录下，日志能被存储到本地路由器的内存或者文件中。也可以通过发送 email 或者通过运行在远程的 syslog 下载程序存储存储到其他的系统硬盘上。

RouterOS 中的日志有不同的分组或者项目，日志来至于每个项目运行状态，可通过配置进行每个组或项目的记录。局部日志文件能存储到内存中（内存记录为默认并会显示到**/log** 目录下，在重启或者断电后日志会丢失），以及远程记录等。

操作路径: ***/system logging***

属性描述

action (*名称*; 默认: **memory**) – 用户可选择在**/system logging action** 指定操作的类型
prefix (*文本*) – 本地日志前缀
topics (info | critical | firewall | keepalive | packet | read | timer | write | ddns | hotspot | l2tp | ppp | route | update | account | debug | ike | manager | pppoe | script | warning | async | dhcp | notification | pptp | state | watchdog | bgp | error | ipsec | radius | system | web-proxy | calc | event | isdn | ospf | raw | telephony | wireless | e-mail | gsm | mme | ntp | open | ovpn | pim | radvd | rip | sertcp | ups; 默认: **info**) – 指定日志组或者日志信息类型

在 logging 中通过记录 firewall 产生的日志信息，存储到本地缓存中。

```
[admin@MikroTik] system logging> add topics=firewall action=memory
[admin@MikroTik] system logging> print
 Flags: X - disabled, I - invalid
 #   TOPICS                                            ACTION PREFIX
 0   info                                              memory
 1   error                                             memory
 2   warning                                           memory
 3   critical                                          echo
 4   firewall                                          memory
[admin@MikroTik] system logging>
```

29.1 Logging 执行

操作: ***/system logging action***

属性描述

disk-lines (整型; 默认: **100**) – 在日志文件存储到硬盘的记录数量(仅在 action 设置为 **disk**)
disk-stop-on-full (yes | no; 默认: **no**) – 是否在 disk-lines 数量达到后停止存储日志信息
email-to (*名称*) – 发送到指定的 email 地址 (仅在 action 设置为 **email**)
memory-lines (*整型*; 默认: **100**) – 在本地缓存记录的数量(仅在 action 设置为 **memory**)
memory-stop-on-full (yes | no; 默认: **no**) -是否在 memory-lines 数量达到后停止存储日志信息
name (*名称*) – 一个 action 操作的名称
remember (yes | no; 默认: **yes**) – 是否保存日志信息， 其中尚未显示在控制台的 (仅在 action 设置为 **echo**)

remote (*IP address*:*port* ; 默认: **0.0.0.0:514**) – 远程日志服务器的 IP 地址和 UDP 端口(仅在 action 设置为 **remote**)
target (disk | echo | email | memory | remote; 默认: **memory**) – 记录存储设备或目标
disk – 日志记录到硬盘
echo – 日志显示在控制台屏幕上
email – 日志通过 email 发送
memory – 日志被存储到本地内存
remote – 日志发送到远端服务主机

注： 你不能删除或重命名默认 action 规则

添加一个新的 action 取名为 long，将日志记录到本地内存，在内存中的记录为 1000 条，这样在**/log** 中会显示 1000 条记录，用于查看很多的信息：

```
[admin@MikroTik] system logging action> add name=long \
\... target=memory memory-lines=50 memory-stop-on-full=yes
[admin@MikroTik] system logging action> print
Flags: * - default
 #   NAME                          TARGET REMOTE
 0 * memory                         memory
 1 * disk                           disk
 2 * echo                           echo
 3 * remote                         remote 0.0.0.0:514
 4   long                          memory
[admin@MikroTik] system logging action>
```

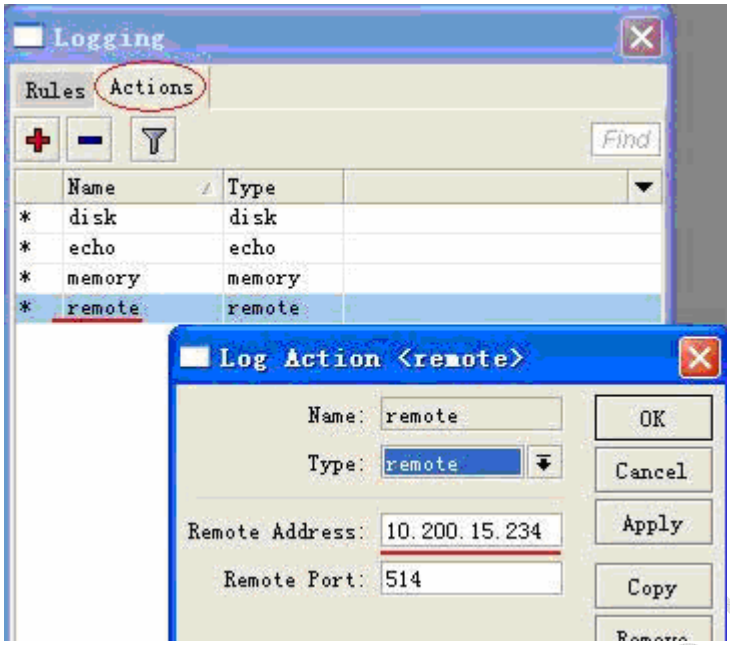
通过 ip firewall filter 记录所有访问 80 端口，并在 log 中添加前缀“80port”的相关信息：

```
[admin@MikroTik] /ip firewall filter>  add chain=forward protocol=tcp dst-port=80
 action=log log-prefix=80port
[admin@MikroTik] /ip firewall filter> print
Flags: X - disabled, I - invalid, D - dynamic

 0   chain=forward action=log protocol=tcp dst-port=80 log-prefix="80port"
```

29.2 使用 Dude 管理器记录系统日志

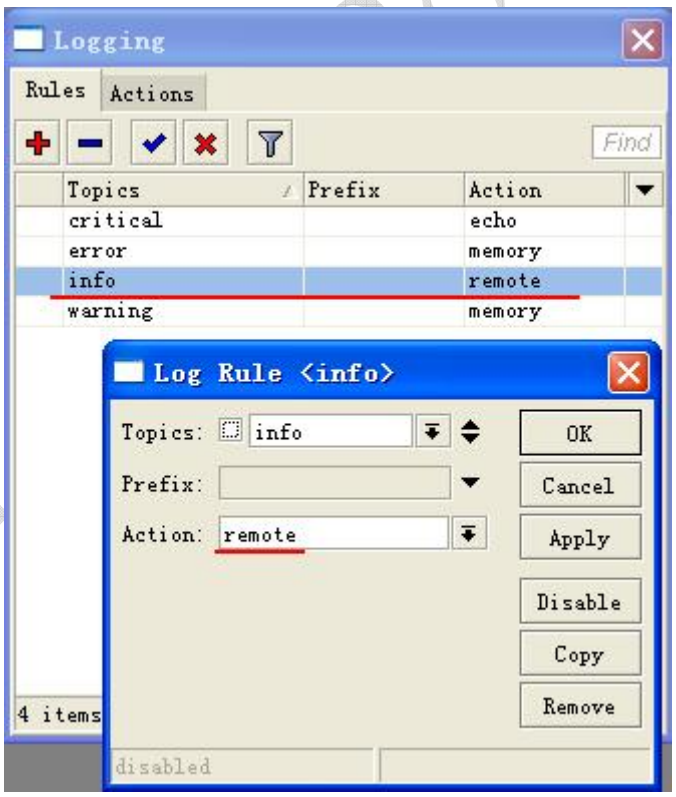
在 MikroTik 提供的 syslog 软件，用于记录 RouterOS 的系统日志信息，但这个软件只能记录 1000 条，不能做长时间记录和定期存储。在新版本的 The Dude 网络管理软件中增加了系统日志记录和下载的功能，这个我们可以通过 The Dude 管理器对我们需要的 RouterOS 日志信息进行记录和管理。

这里我们使用的是 The Dude 3.0beta8 的版本，首先我们需要进入 RouterOS 的 system logging 配置系统日志的远程记录参数：

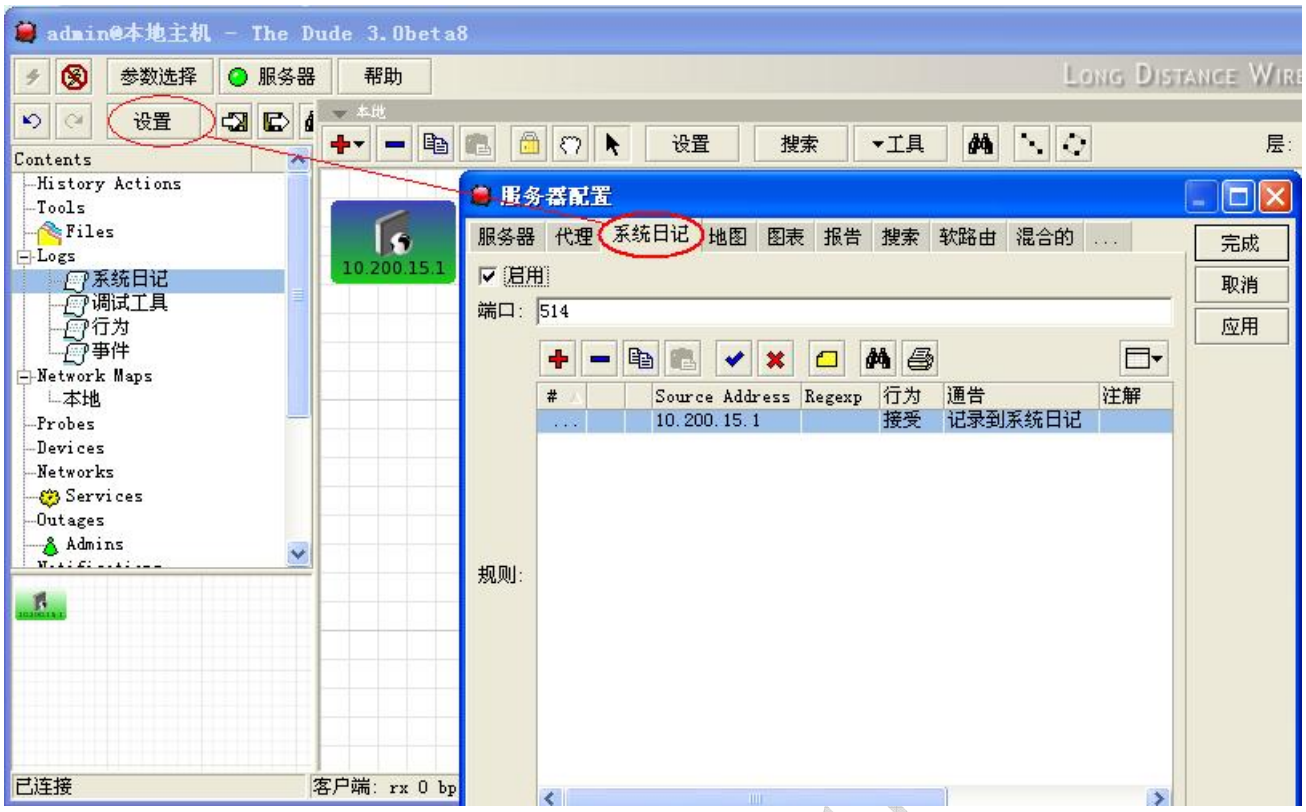


进入 system logging 后配置 actions 里的 remote 参数，将 Remote Address 配置为指定的系统信息接受的 The Dude 服务器 IP 地址。

然后将需要记录的日志信息，设置为 remote：

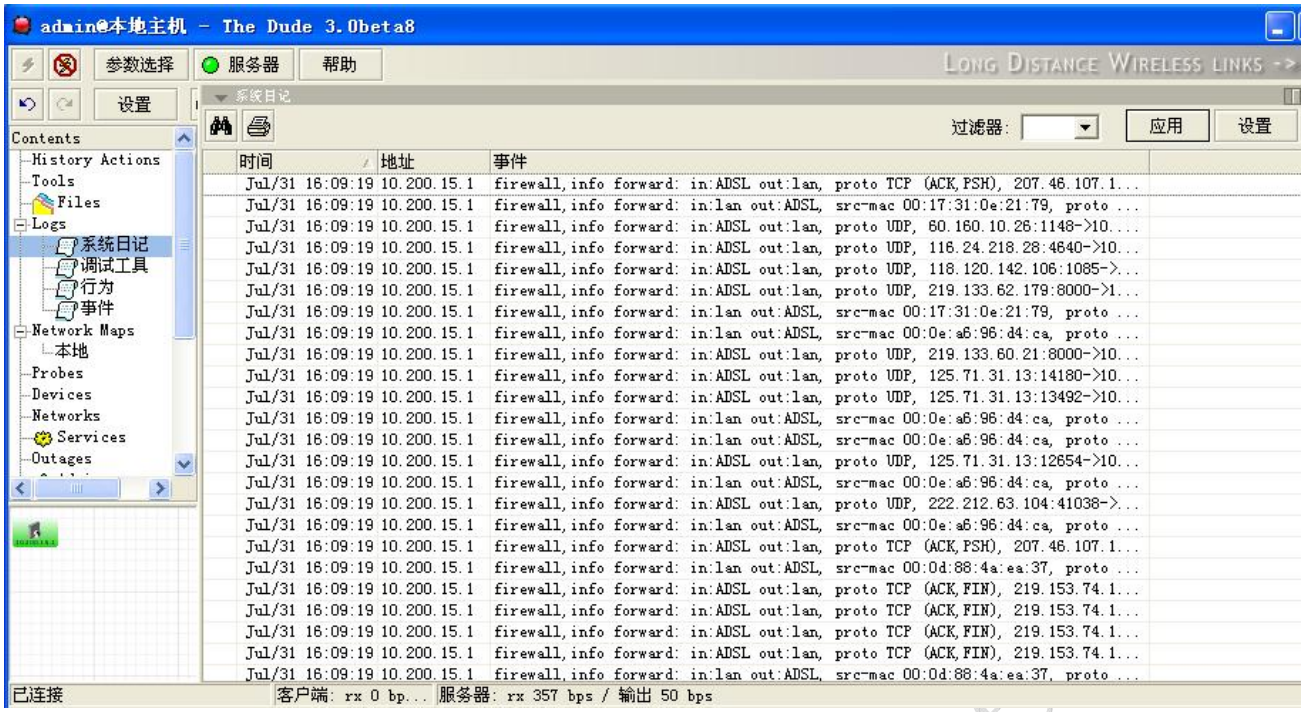


下面是在 10.200.15.234 的 The Dude 网络管理器上配置系统日志信息，进入“设置”，选择“系统日记”，并配置相应的端口和 IP 地址。



配置 The Dude 系统日志记录参数：

配置完成后，我们在 The Dude 管理器的 log 下系统日记看到接受到 IP 地址为 10.200.15.1 的 RouterOS 的日志：



在 The Dude 的 log 记录中，有一个“设置”选项，可以配置日志记录的存储参数：

这里我们设置日志记录存储的文件名字、产生新的日志文件时间间隔等参数。

通过 The Dude 网络管理软件，可以方便的纪录每台 RouterOS 的日志信息和情况，同时达到监控的目的。这样能对你的网络进行综合的管理和监控，分析网络运行情况和状态，为你及时对网络环境进行处理和改造提供及时的信息。

29.3 Log 信息

操作路径: ***/log***

用于显示**/system logging** 记录的日志信息

属性描述

message (*只读: 文本*) – 信息文本
time (*只读: 文本*) – 事件的日期和时间
topics (*只读: 文本*) – 项目信息的从属

查看本地日志：

```
[admin@MikroTik] > log print
 TIME                MESSAGE
 dec/24/2003 08:20:36 log configuration changed by admin
 dec/24/2003 08:20:36 log configuration changed by admin
 dec/24/2003 08:20:36 log configuration changed by admin
 dec/24/2003 08:20:36 log configuration changed by admin
 dec/24/2003 08:20:36 log configuration changed by admin
 dec/24/2003 08:20:36 log configuration changed by admin
-- [Q quit|D dump]
```

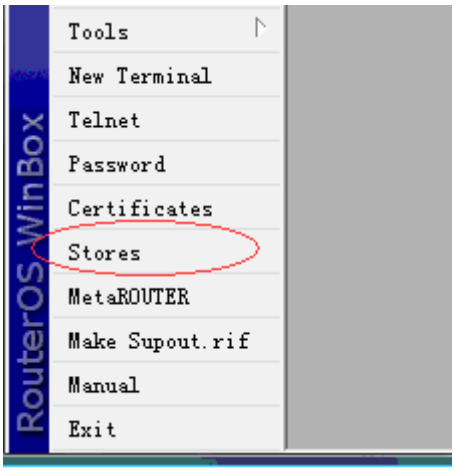
监视系统日志：

```
[admin@MikroTik] > log print follow
 TIME                MESSAGE
 dec/24/2003 08:20:36 log configuration changed by admin
 dec/24/2003 08:24:34 log configuration changed by admin
 dec/24/2003 08:24:51 log configuration changed by admin
 dec/24/2003 08:25:59 log configuration changed by admin
 dec/24/2003 08:25:59 log configuration changed by admin
 dec/24/2003 08:30:05 log configuration changed by admin
 dec/24/2003 08:30:05 log configuration changed by admin
 dec/24/2003 08:35:56 system started
 dec/24/2003 08:35:57 isdn-out1: initializing...
 dec/24/2003 08:35:57 isdn-out1: dialing...
 dec/24/2003 08:35:58 Prism firmware loading: OK
 dec/24/2003 08:37:48 user admin logged in from 10.1.0.60 via telnet
-- Ctrl-C to quit. New entries will appear at bottom.
```

第三十章 RouterOS Store 功能

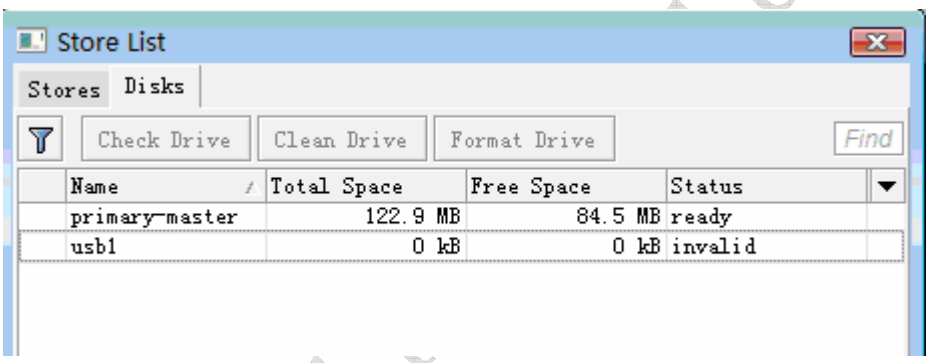
RouterOS 在 3.15 后增加了 store 存储功能，支持各种本地系统存储和外部设备存储，主要应用于 Web-proxy、User-Manager 和 The Dude 数据存储，在 3.23 后由于 RouterOS 支持 log 日志的本地存储，所以 Store 的应用有所增加。

除了 RouterOS 使用本地系统盘存储外，我们可以在 PC 或者 RouterBOARD 上增加各种存储设备，比如 RouterBOARD 可以选择 CF/MircoSD 方式存储，而 PC 可以选择增加硬盘、U 盘等方式。我们进入 winbox 后可以选择 store 目录，进入存储管理，

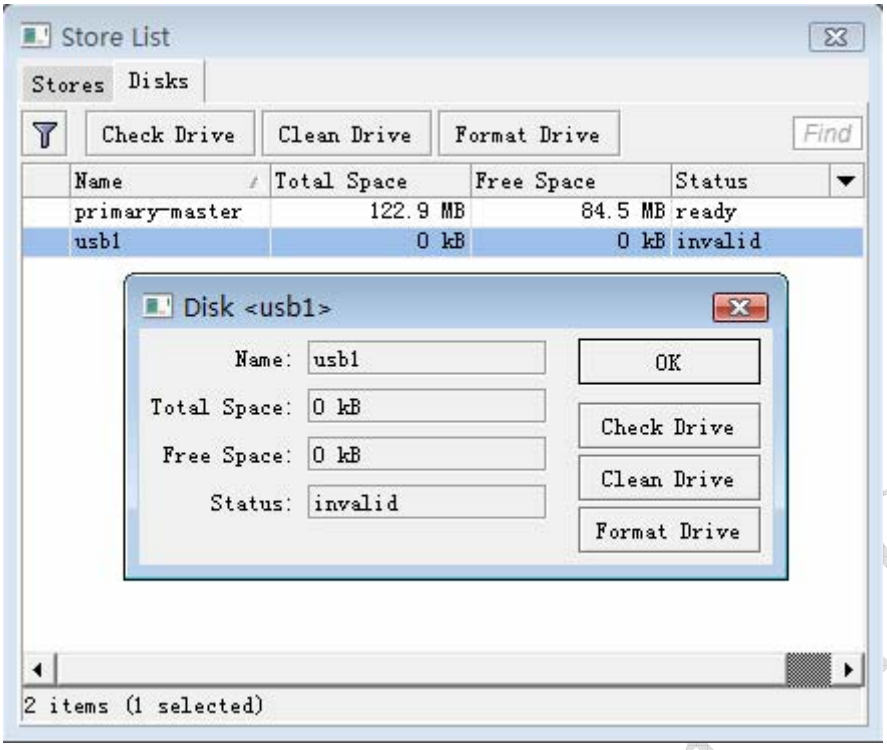


30.1 RouterOS 使用 U 盘扩展存储

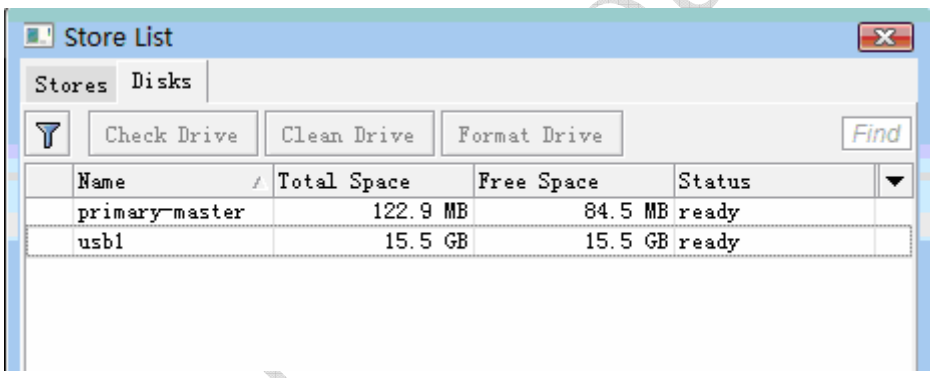
这里我们通过使用 U 盘来演示，在 PC 上增加外部存储的操作，我们将 1 个 16G 的 U 盘，插入 RouterOS PC 的 USB 接口，这个时候，我们可以在 Store 的 Disk 目录中找到 usb1 的硬盘信息：



当前状态为 invalid，即无法识别，因为 RouterOS 的硬盘分区和我们常用的 U 盘分区不同，所以我们需要选择 usb1，对 U 盘做格式化操作，选择 Format Driver 的选项



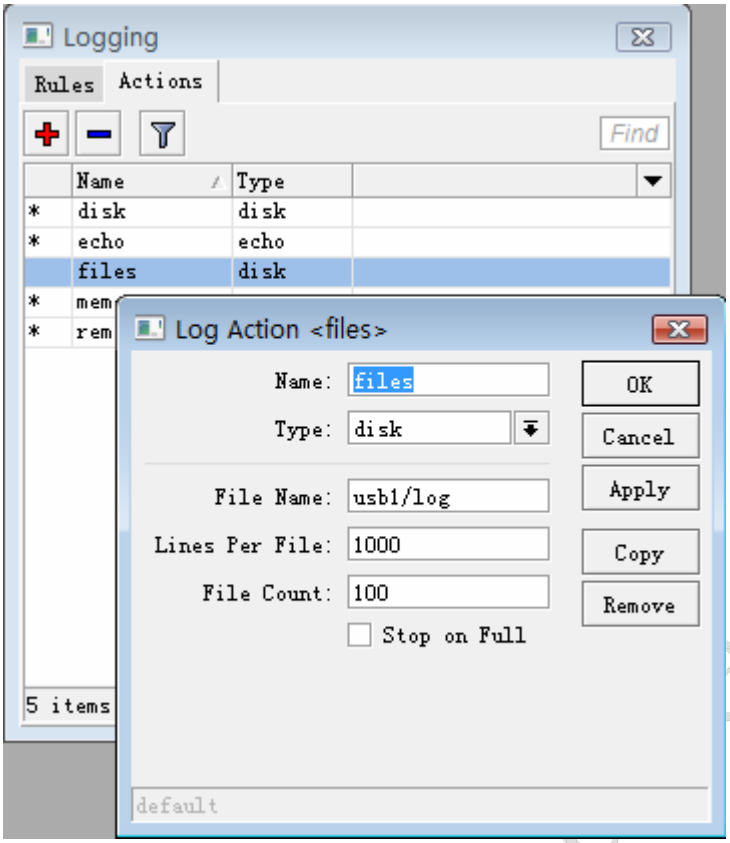
在格式化完成后，我们可以看到当前的 usb1 状态为 ready，能够正常识别到容量和空闲存储空间：



30.2 存储 log 日志信息

在 RouterOS3.23 后增加了可以将 log 日志存储到 RouterOS 上的存储设备里，由于本地系统存储空间有限，我们可以通过外部存储的 U 盘扩展，这里我们通过我们可以使用日志记录

首先我进入 system logging 配置 Action，并新建立一个 files 规则，并定义存储方式 type 为 disk：

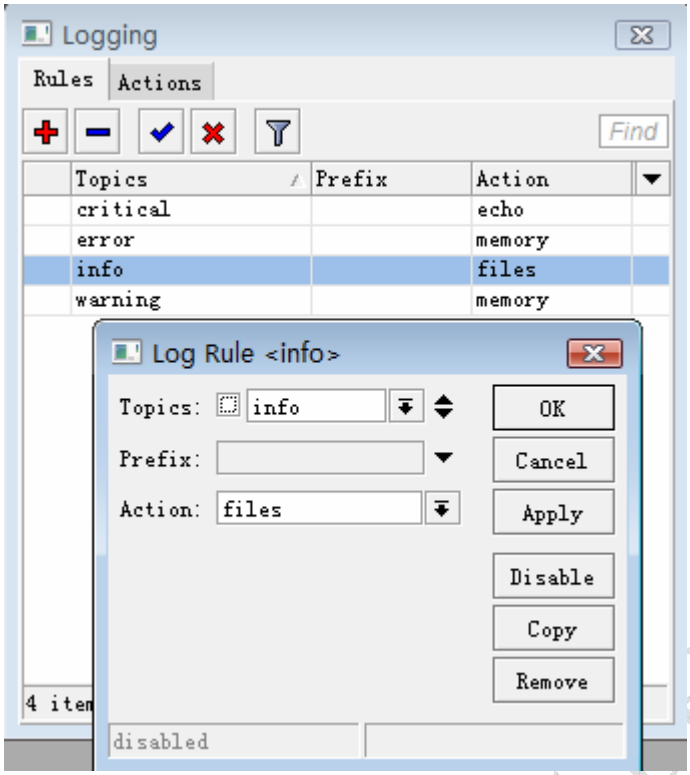


Disk 类型几个参数如下：

- **Type：**log 日志记录方式，这里我们选择 disk
- **File Name：**文件存储的路径，如果是 usb1 的 U 盘，我给的路径是 usb1/log
- **Lines Per File：**每个文件记录多少条信息
- **File Count：**log 日志一共建立多少个文件，如果日志记录超出文件数量，将会从 log0 号从头开始记录并覆盖原来的文件
- **Stop on Full：**当 log 建立的文件到达后，停止向文件写入 log 日志

注意，文件名建立的原则，即**<filename>.0.txt**，**<filename>.1.txt**，**<filename>.n.txt** 的顺序建立，文件的大小可以自行定义。

设置 logging 的 info（信息记录）为 files 操作，即记录到 usb1 中



Topics	Prefix	Action
critical		echo
error		memory
info		files
warning		memory

4 item

Log Rule <info>

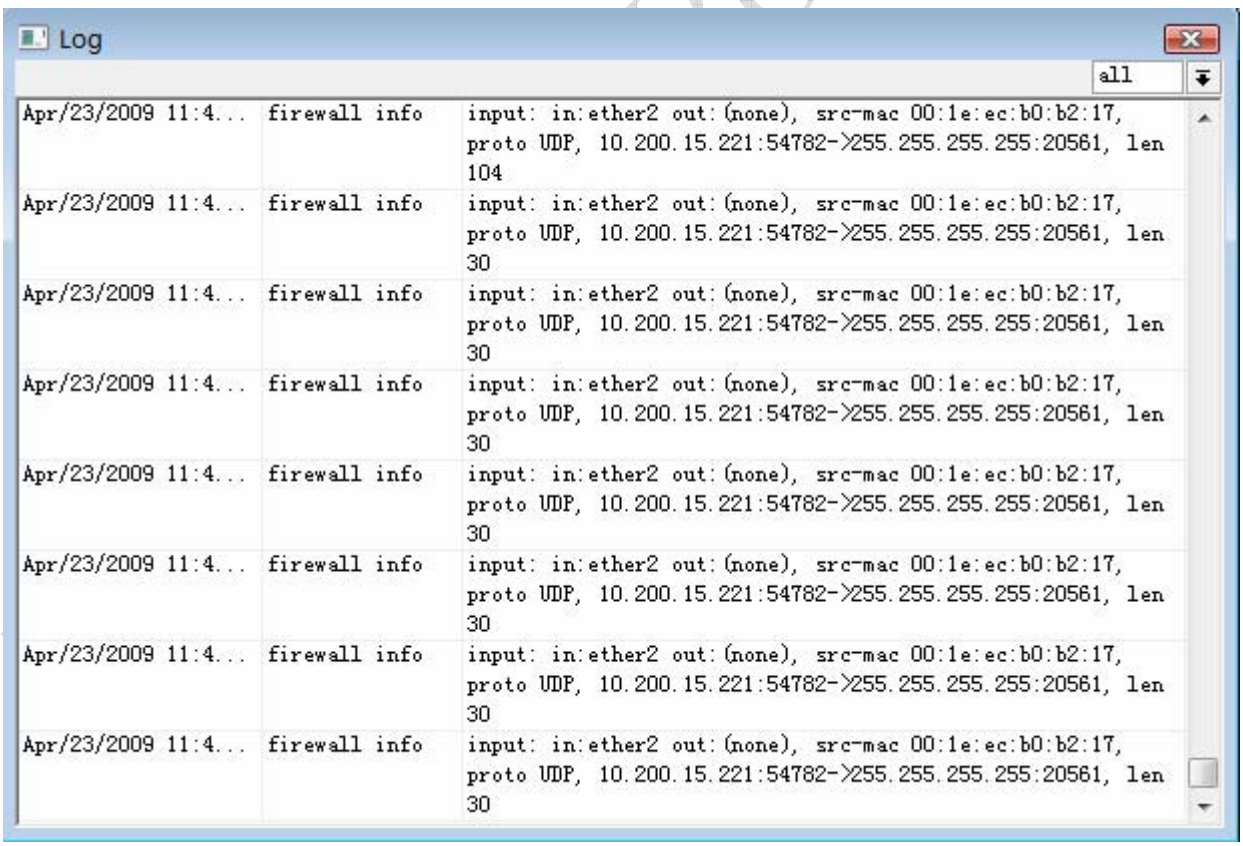
Topics: info

Prefix:

Action: files

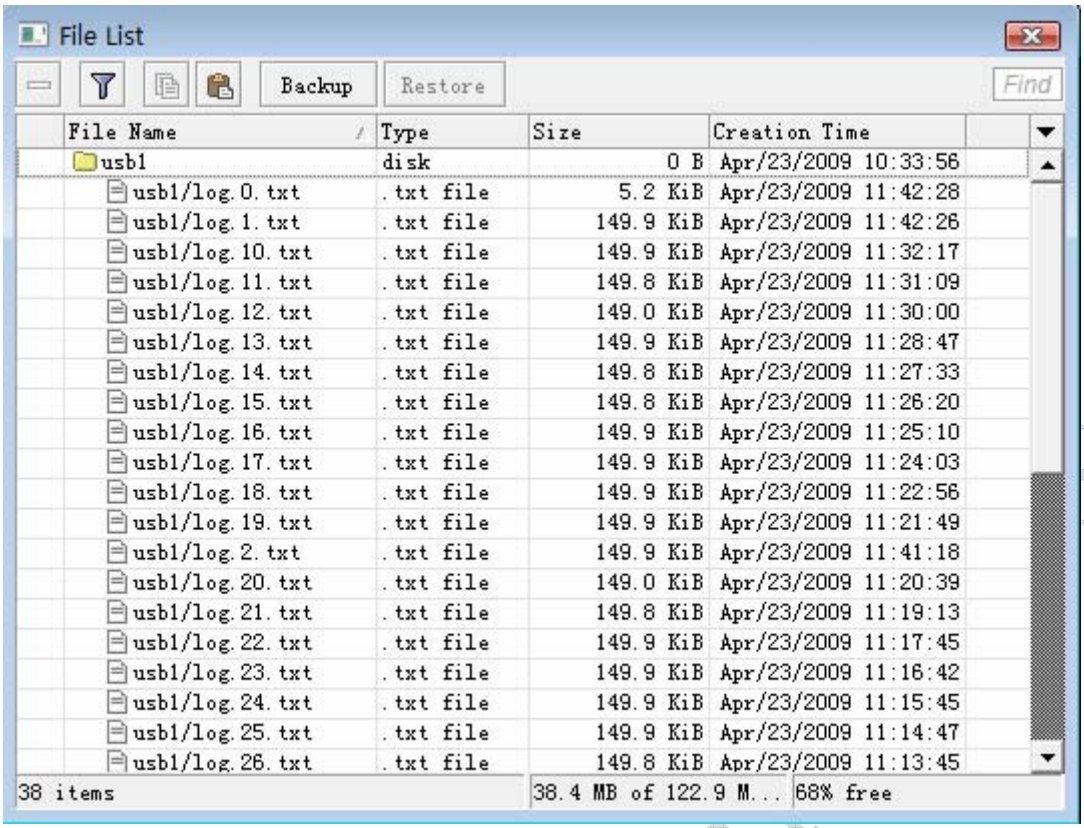
disabled

我们可以看到在 log 中记录的防火墙信息



```
Apr/23/2009 11:4...  firewall info  input: in:ether2 out:(none), src-mac 00:1e:ec:b0:b2:17,
                                    proto UDP, 10.200.15.221:54782->255.255.255.255:20561, len
                                    104
Apr/23/2009 11:4...  firewall info  input: in:ether2 out:(none), src-mac 00:1e:ec:b0:b2:17,
                                    proto UDP, 10.200.15.221:54782->255.255.255.255:20561, len
                                    30
Apr/23/2009 11:4...  firewall info  input: in:ether2 out:(none), src-mac 00:1e:ec:b0:b2:17,
                                    proto UDP, 10.200.15.221:54782->255.255.255.255:20561, len
                                    30
Apr/23/2009 11:4...  firewall info  input: in:ether2 out:(none), src-mac 00:1e:ec:b0:b2:17,
                                    proto UDP, 10.200.15.221:54782->255.255.255.255:20561, len
                                    30
Apr/23/2009 11:4...  firewall info  input: in:ether2 out:(none), src-mac 00:1e:ec:b0:b2:17,
                                    proto UDP, 10.200.15.221:54782->255.255.255.255:20561, len
                                    30
Apr/23/2009 11:4...  firewall info  input: in:ether2 out:(none), src-mac 00:1e:ec:b0:b2:17,
                                    proto UDP, 10.200.15.221:54782->255.255.255.255:20561, len
                                    30
Apr/23/2009 11:4...  firewall info  input: in:ether2 out:(none), src-mac 00:1e:ec:b0:b2:17,
                                    proto UDP, 10.200.15.221:54782->255.255.255.255:20561, len
                                    30
Apr/23/2009 11:4...  firewall info  input: in:ether2 out:(none), src-mac 00:1e:ec:b0:b2:17,
                                    proto UDP, 10.200.15.221:54782->255.255.255.255:20561, len
                                    30
```

在 file list 中可以看到 usb1 中建立的 log 文件，文件以 txt 文本文件存储

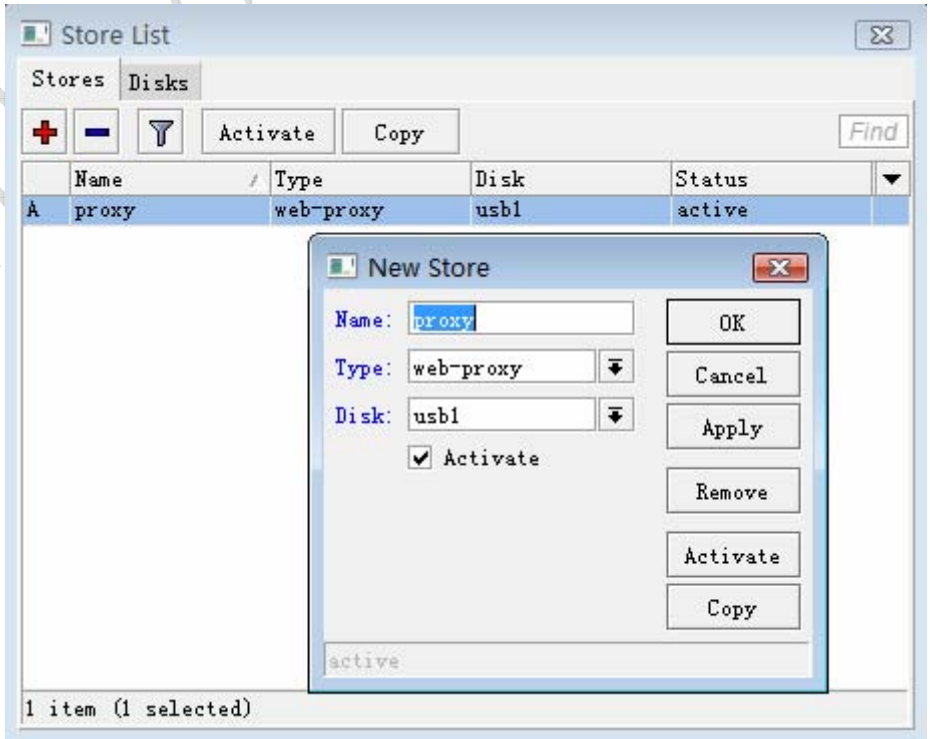


注意，当在记录大量的日志信息，使用本地存储设备写入数据时，会出现 CPU 占用较大的情况，需要注意合理分配你的系统资源，建议尽量做远程日志记录的存储。

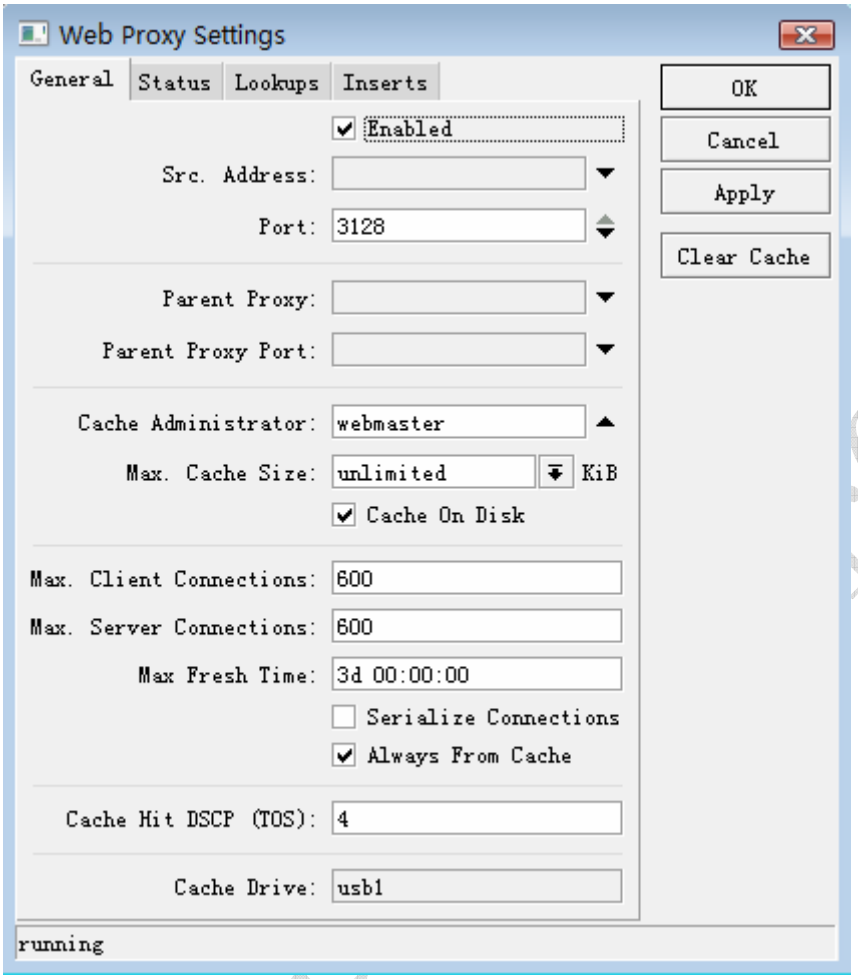
30.3 Web-Proxy 使用 U 盘存储

一些特殊网络环境，可能会用到 Web 缓存功能，需要将访问过的静态页面缓存到硬盘中，做二次访问。由于系统盘空间有限，我们可以使用 U 盘做为网页数据的外部存储。

下面在 Store 目录下添加一个名 Proxy 的规则，选择类型为 web-proxy，指定硬盘为 usb1

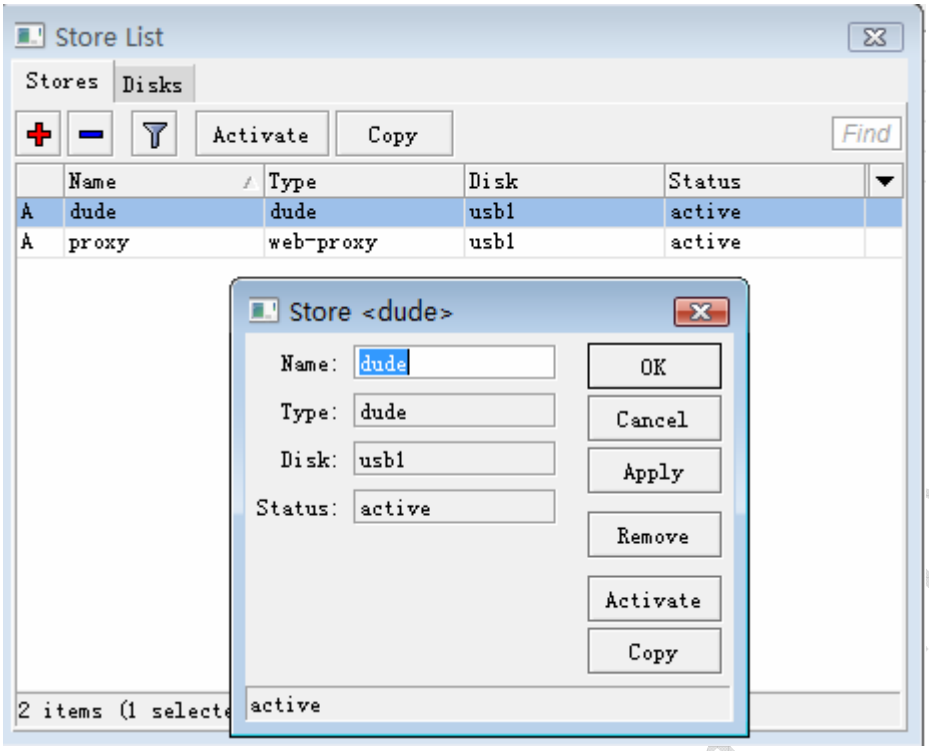


设置 Web-Proxy 的配置，这里可以看到 Cache Drive 会根据 Store 的配置，调用 usb1 的外部存储



30.4 Store 的其他应用

Store 可以建立 The Dude 网络管理器的拓扑结构图的数据存储，如下图



Store 功能也可以用户存储 User-Manager 的数据信息，建立 User-Manager 的数据库。

第三十一章 IP 访问日志记录

IP 访问日志记录，用于内向外或者外向内发送的所有连接（包括源地址、目标地址、数据包和字节）都会被记录下来。同时当启用 Hotspot 和 PPP 认证时，账号也随之被记录到相应的连接中。在 RouterOS 中主要应用于记录内网与外网之间的访问日志，以便对网络中的所有数据作记录进行检查和分析，或者出于安全考虑为以后非法连接提供依据。

规格

功能包要求: ***system***
认证等级: *Level1*
操作路径: ***/user***, ***/ppp***, ***/ip accounting***, ***/radius***
硬件使用: 传输记录需要根据记录内容大小增加内存

30.1 IP 访问记录

操作路径: ***/ip accounting***

当每个包通过路由器时，匹配 IP 数据包源和目的地址会成对的在访问列表中并且这个对的流量会增加。PPP，PPTP，PPPoE，ISDN，以及 HotSpot 客户的流量也可以在每个用户的基础上计算。数据包的数量和字节的数量都会被计算。如果没有与之前的 IP 或用户对匹配，那么新的记录将被添加到里表中。

属性描述

enabled (yes | no; 默认: **no**) - 是否启用了本地 IP 访问记录日志

account-local-traffic (yes | no; 默认: **no**) - 是否计算来自/到达路由器的流量访问
threshold (*整型*; 默认: **256**) - 在管理列表中的 IP 对的最大数量（最大值为 8192）

当临界值限制达到时，没有新的 IP 对将被添加到管理列表中。在管理列表中没有计算的每个包都将被添加到 **uncounted** 计数器。启用 IP 访问管理：

```
[admin@MikroTik] ip accounting> set enabled=yes
[admin@MikroTik] ip accounting> print
              enabled: yes
  account-local-traffic: no
            threshold: 256
[admin@MikroTik] ip accounting>
```

30.2 IP 访问快照

操作路径: ***/ip accounting snapshot***

当数据收集的快照做好后，管理列表会被清空并且新的 IP 对与流量数据会被添加近来。更经常的数据会被收集。

属性描述

bytes (*只读: 整型*) - 字节总数，以条目匹配
dst-address (*只读: IP address*) - 目的 IP 地址
dst-user (*只读: 文本*) – 接受者的名称(如果可应用)
packets (*只读: 整型*) - 包的总数，以这个条目匹配
src-address (*只读: IP address*) - 源 IP 地址
src-user (*只读: 文本*) – 发送者的名称（如果可用）

注: 仅当用户通过一个 PPP 隧道连接到路由器或被 HotSpot 认证时才显示用户名。在获取快照之前，列表是空的。

取一个新 IP 访问的快照：

```
[admin@MikroTik] ip accounting snapshot> take
[admin@MikroTik] ip accounting snapshot> print
 # SRC-ADDRESS     DST-ADDRESS     PACKETS    BYTES      SRC-USER    DST-USER
 0 192.168.0.2     159.148.172.197 474        19130
 1 192.168.0.2     10.0.0.4        3          120
 2 192.168.0.2     192.150.20.254  32         3142
 3 192.150.20.254  192.168.0.2     26         2857
 4 10.0.0.4        192.168.0.2     2          117
 5 159.148.147.196 192.168.0.2     2          136
 6 192.168.0.2     159.148.147.196 1          40
 7 159.148.172.197 192.168.0.2     835        1192962
[admin@MikroTik] ip accounting snapshot>
```

30.3 Web 获取 IP 访问信息

操作路径: ***/ip accounting web-access***

web 页面报告似的使用标准的 Unix/Linux wget 工具收集流量数据并存储到文件或者使用 MikroTik 的日志下载软件。如果 web 报告启用且 web 页面被查看，那么当连接起始为 web 页面时，**snapshot** 将被生成。**Snapshot** 将在 web 页面上显示。被有 wget 工具 http 连接使用的 TCP 协议保证任何一点的流数据都不会丢失。**Snapshot** 图像将在来自 wget 的连接被初始化时生成。Web 浏览器或 wget 可以连接到 URL：**http://routerIP/accounting/ip.cgi**

属性描述

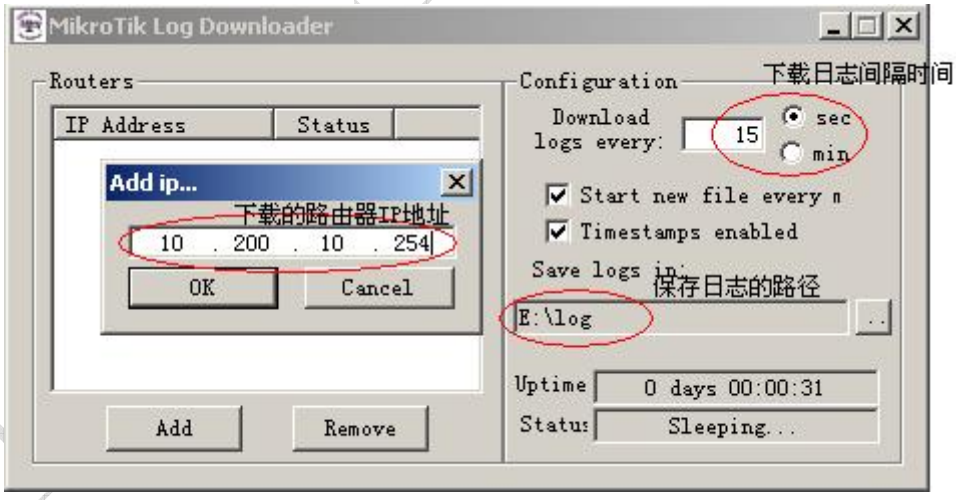
accessible-via-web (yes | no; 默认: **no**) – 是否 snapshot 通过 web 可用
address (*IP 地址*/子网掩码; 默认: **0.0.0.0**) -允许存取 snapshot 的 IP 地址范围

仅启用来自 **192.168.10.10** 主机对流量日志的 web 访问：

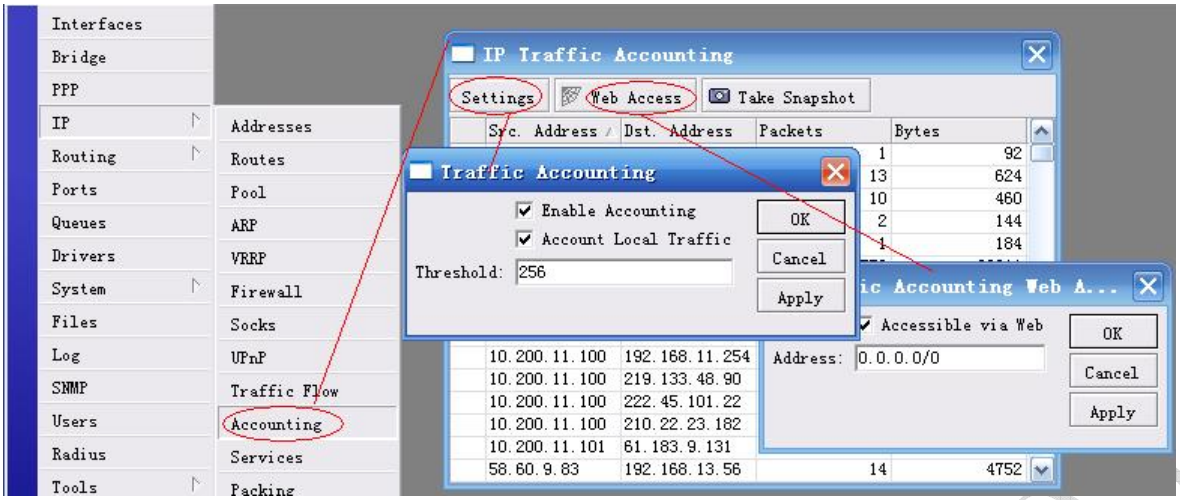
```
[admin@MikroTik] ip accounting web-access> set accessible-via-web=yes \
\... address=192.168.10.10/32
[admin@MikroTik] ip accounting web-access> print
   accessible-via-web: yes
              address: 192.168.10.10/32
[admin@MikroTik] ip accounting web-access>
```

下面是通过 Log Downloader 和 winbox 操作的事例

首先打开 Log Downloader 程序，如图所示，添加需要记录 RouterOS 的日志的 IP 地址，并配置相应的参数：



然后在 RotuerOS 打开，并启用日志的远程记录，在 ip accounting 中设置：



注意：该软件只能通过 RouterOS 的 web 端口记录，即 web 端口默认必须是 80。

第三十二章 Scheduler（计划任务）

设定的计划任务，并通过时间安排执行相应的脚本操作.

规格

功能包需求: ***system***
等级需求: *Level1*
操作路径: ***/system scheduler***

32.1 计划任务配置

计划任务列表能触发脚本执行，在指定的时间段或者是在指定的时间间隔。

属性描述

interval (*时间*; 默认: **0s**) - 脚本执行的间隔时间，脚本反复执行在一个指定的时间间隔
name (*名称*) – 任务名
on-event (*名称*) – 脚本执行名。通过调用**/system script** 里的脚本规则名称
run-count (*只读: 整型*) – 监视脚本使用数，这个计数器记录当每个脚本执行一次，计数器便增加 1
start-date (*日期*) – 开始脚本执行的日期
start-time (*时间*) – 开始脚本执行的时间
startup – 默认在系统启动 3 秒后执行脚本.

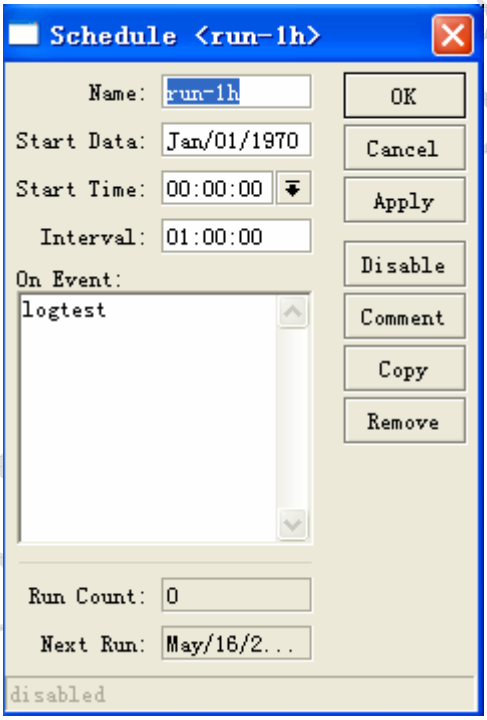
注： 重启路由器时将重置 run-count 计数器。

如果计划表选项里面对 start-time 设置了 startup，则在控制台开启后 3 秒运行。这意味着所有的脚本设置为 start-time=startup 和 interval=0，当路由器启动就会被执行。

事例 1： 我们添加一个任务执行系统日志记录测试，并间隔 1 小时执行一次，这个操作为 **logtest**：

```
[admin@MikroTik] system script> add name=logtest source=:log info “test”
[admin@MikroTik] system script> print
    0   name="script1" owner="admin"
    policy=ftp,reboot,read,write,policy,test,winbox,password,sniff
    last-started=may/16/2008 21:32:51 run-count=3 source=:log info " test"
[admin@MikroTik] system script> .. scheduler
[admin@MikroTik] system scheduler> add name=run-1h interval=1h on-event=logtest
[admin@MikroTik] system scheduler> print

Flags: X - disabled
 #   NAME      ON-EVENT  START-DATE  START-TIME INTERVAL    RUN-COUNT
 0   run-1h    logtest   mar/30/2004 06:11:35   1h         0
[admin@MikroTik] system scheduler>
```

Schedule 在 Winbox 的配置如下：



事例 2： 另外一个例子是添加 2 个脚本改变带宽设置队列规则”cust0”，每天上午 9 点限制为 64kb/s，下午 5 点限制为 128kb/s。这个队列的规则、脚本和计划任务如下(**注**：在 2.9 种 cust0 是不需要加双引号的，但在 3.0 中需要注明字符串，要加上双引号”cust0”) ：

```
[admin@MikroTik] queue simple> add name=Cust0 interface=ether1 \
\... target-address=192.168.0.0/24 limit-at=64000
[admin@MikroTik] queue simple> print
Flags: X - disabled, I - invalid
  0   name="Cust0" target-address=192.168.0.0/24 dst-address=0.0.0.0/0
      interface=ether1 limit-at=64000 queue=default priority=8 bounded=yes

[admin@MikroTik] queue simple> /system script
[admin@MikroTik] system script> add name=start_limit source={/queue simple set \
```

```
\... “Cust0” limit-at=64000}
[admin@MikroTik] system script> add name=stop_limit source={/queue simple set \
\... “Cust0” limit-at=128000}
[admin@MikroTik] system script> print
  0 name="start_limit" source="/queue simple set “Cust0” limit-at=64000"
    owner=admin run-count=0

  1 name="stop_limit" source="/queue simple set “Cust0” limit-at=128000"
    owner=admin run-count=0

[admin@MikroTik] system script> .. scheduler
[admin@MikroTik] system scheduler> add interval=24h name="set-64k" \
\... start-time=9:00:00 on-event=start_limit
[admin@MikroTik] system scheduler> add interval=24h name="set-128k" \
\... start-time=17:00:00 on-event=stop_limit
[admin@MikroTik] system scheduler> print
Flags: X - disabled
  #   NAME     ON-EVENT  START-DATE  START-TIME INTERVAL           RUN-COUNT
  0   set-64k   start... oct/30/2008 09:00:00   1d                0
  1   set-128k  stop_... oct/30/2008 17:00:00   1d                0
[admin@MikroTik] system scheduler>
```

事例 3： 下面的例子安排了一个通过电子邮件发送每周备份路由器配置信息的脚本：

```
[admin@MikroTik] system script> add name=e-backup source={/system backup
save name=email; /tool e-mail send to="root@host.com" subject=([/system
{... identity get name] . " Backup") file=email.backup}
[admin@MikroTik] system script> print
  0 name="e-backup" source="/system backup save name=ema... owner=admin
    run-count=0

[admin@MikroTik] system script> .. scheduler
[admin@MikroTik] system scheduler> add interval=7d name="email-backup" \
\... on-event=e-backup
[admin@MikroTik] system scheduler> print
Flags: X - disabled
  #   NAME     ON-EVENT START-DATE  START-TIME INTERVAL           RUN-COUNT
  0   email-... e-backup oct/30/2008 15:19:28   7d               1
[admin@MikroTik] system scheduler>
```

不要忘记去设置电子邮件参数，即 SMTP 服务的配置，操作路径**/tool e-mail** 例如（**注：**建议是自己的 SMTP 服务器，一些正规网站的邮件服务器可能会将发送信息屏蔽）：

```
[admin@MikroTik] tool e-mail> set server=159.148.147.198 from=SysAdmin@host.com
[admin@MikroTik] tool e-mail> print
   server: 159.148.147.198
     from: SysAdmin@host.com
```

```
[admin@MikroTik] tool e-mail>
```

事例 4

下面的例子是从午夜 12 点到正午 12 点的每个小时里把“x”加进日志中:

```
[admin@MikroTik] system script> add name=enable-x source={/system scheduler
{... enable x}
[admin@MikroTik] system script> add name=disable-x source={/system scheduler
{... disable x}
[admin@MikroTik] system script> add name=log-x source={:log info “x”}
[admin@MikroTik] system script> .. scheduler
[admin@MikroTik] system scheduler> add name=x-up start-time=00:00:00 \
\... interval=24h on-event=enable-x
[admin@MikroTik] system scheduler> add name=x-down start-time=12:00:00
\... interval=24h on-event=disable-x
[admin@MikroTik] system scheduler> add name=x start-time=00:00:00 interval=1h \
\... on-event=log-x
[admin@MikroTik] system scheduler> print
Flags: X - disabled
  #   NAME     ON-EVENT START-DATE  START-TIME INTERVAL           RUN-COUNT
  0   x-up     enable-x oct/30/2008 00:00:00   1d                0
  1   x-down    disab... oct/30/2008 12:00:00   1d                0
  2   x        log-x    oct/30/2008 00:00:00   1h                0
[admin@MikroTik] system scheduler>
```

第三十三章 RouterOS 常用工具

33.1、Netwatch 监控

Netwatch 工具通过 ping 监控网络中的主机，并能通过状态的改变产生定义的事件。

规格

需要功能包: ***advanced-tools***
等级: *Level1*
操作路径: ***/tool netwatch***
协议标准: none

Netwatch 监控的是在网络上的主机状态。 通过在列表中指定 IP 地址，并发送间隔的 ICMP 的 ping 探测和执行控制脚本。在主机状态改变时根据 netwatch 的情况下命令。

属性描述

down-script (*名称*) – 当一个主机的状态从 **unknown** 或 **up** 改变为 **down**。
host (*IP 地址*; 默认: **0.0.0.0**) – 需要监视的主机 IP 地址
interval (*时间*; 默认: **1s**) – ping 间隔时间。
status (*只读:* up | down | unknown) – 显示主机的当前状态
up – 主机状态为 up
down – 主机状态为 down
unknown – 在列表项目属性被改变后或是项目被启用或禁用
timeout (时间; 默认: **1s**) – 每个 ping 的 timeout 值。在这个时钟周期内没有收到来至主机的回应，将认为该主机为 **down**
up-script (*名称*) -当一个主机的状态从 **unknown** 或 **down** 改变 **up**

事例

这个事例将运行脚本 gw_1 或 gw_2 根据网关的状态来修改默认网关：

```
[admin@MikroTik] system script> add name=gw_1 source={/ip route set
{... [/ip route find dst 0.0.0.0] gateway 10.0.0.1}
[admin@MikroTik] system script> add name=gw_2 source={/ip route set
{.. [/ip route find dst 0.0.0.0] gateway 10.0.0.217}
[admin@MikroTik] system script> /tool netwatch
[admin@MikroTik] tool netwatch> add host=10.0.0.217 interval=10s timeout=998ms \\...
up-script=gw_2 down-script=gw_1
[admin@MikroTik] tool netwatch> print
Flags: X - disabled
  #   HOST           TIMEOUT               INTERVAL              STATUS
  0   10.0.0.217     997ms                 10s                   up
[admin@MikroTik] tool netwatch> print detail
Flags: X - disabled
  0   host=10.0.0.217 timeout=997ms interval=10s since=feb/27/2003 14:01:03
      status=up up-script=gw_2 down-script=gw_1

[admin@MikroTik] tool netwatch>
```

让我们来看上面的例子，如果网关变为无法到达改变默认路由。有两个脚本，当主机状态改变为 **up** 脚本"gw_2"执行一次。在这个事例中，相当于进入控制台执行下面的命令：

```
[admin@MikroTik] > /ip route set [/ip route find dst 0.0.0.0] gateway 10.0.0.217
```

/ip route find dst 0.0.0.0 命令是返回在路由表中 **dst-address** 值为 **0.0.0.0** 的参数，通常这种值为默认路由。用于代替**/ip route set** 命令后的第一个变量

当主机状态改变为 **down** 脚本"gw_1"执行一次。如下面：

```
[admin@MikroTik] > /ip route set [/ip route find dst 0.0.0.0] gateway 10.0.0.1
```

如果 10.0.0.217 地址无法到达，改变默认网关。

下面是另一个事例，无论什么时候 10.0.0.215 主机断线，发送 e-mail 通知到你指定的邮箱：

```
[admin@MikroTik] system script> add name=e-down source={/tool e-mail send
{... from="rieks@mt.lv" server="159.148.147.198" body="Router down"
{... subject="Router at second floor is down" to="rieks@latnet.lv"}
[admin@MikroTik] system script> add name=e-up source={/tool e-mail send
{... from="rieks@mt.lv" server="159.148.147.198" body="Router up"
{.. subject="Router at second floor is up" to="rieks@latnet.lv"}
[admin@MikroTik] system script>
[admin@MikroTik] system script> /tool netwatch
[admin@MikroTik] system netwatch> add host=10.0.0.215 timeout=999ms \
\... interval=20s up-script=e-up down-script=e-down
[admin@MikroTik] tool netwatch> print detail
Flags: X - disabled
  0   host=10.0.0.215 timeout=998ms interval=20s since=feb/27/2003 14:15:36
      status=up up-script=e-up down-script=e-down

[admin@MikroTik] tool netwatch>
```

33.2、图形显示（Graphing）

Graphing 是一个监视工具，用于监视 RouterOS 在一段时期内不同参数的情况。

需要功能包: ***system***, ***routerboard****(optional)*
等级需要: *Level1*
操作路径: ***/tool graphing***
Graphing 工具可以显示的图形为：

- Routerboard 健康状态 (电压和温度)
- 资源使用 (CPU，内存和硬盘使用 Disk usage)
- 通过 Interfaces 的传输情况
- simple queues 中的传输情况

Graphing 由两部分构成- 第一部分是收集数据信息，另一部分在一个 Web page 中显示数据访问图形的地址为 **http://[Router_IP_address]/graphs/** 或是通过浏览 RouterOS 的默认网页进入。

在路由器中数据收集每间隔 5 分钟，但保存到系统驱动中是每隔一个 **store-every** 时间，当重起路由后，显示的信息在重起前为最后一次存储到磁盘中的数据。

RouterOS 每一个项目产生四种图标 generates four graphics for each item:

- "Daily" Graph (5 Minute Average)
- "Weekly" Graph (30 Minute Average)
- "Monthly" Graph (2 Hour Average)
- "Yearly" Graph (1 Day Average)

从一个网络去访问每个图形，可以通过 **allow-address** 指定这个网络的访问项目。

操作路径: ***/tool graphing***

属性描述

store-every (5min | hour | 24hours; 默认: **5min**) – 多长时间将信息存储到系统驱动上。

存储信息到系统驱动上为每小时：

```
/tool graphing set store-every=hour
[admin@NAT] tool graphing> print
   store-every: hour
[admin@NAT] tool graphing>
```

健康情况

操作路径: ***/tool graphing health***

这个子项目提供关于 RouterBoard 的电压和温度的信息，但你必须安装 **routerboard** 功能包和使用 RouterBoard：

属性描述

allow-address (*IP 地址*/掩码; 默认: **0.0.0.0/0**) – 运行访问图形显示的网络地址段
store-on-disk (yes | no; 默认: **yes**) – 是否将信息存储到系统驱动上，如果选择为'no' ，这些信息将存储到 RAM 中，重起后回丢失

接口图表

操作路径: ***/tool graphing interface***

显示有多少流量传输在一段时期内通过了一个 interface

属性描述

allow-address (*IP 地址*/掩码; 默认: **0.0.0.0/0**) -运行访问图形显示的网络地址段，被允许的地址可以试着打开 **http://[Router_IP_address]/graphs/**，如果没有允许将无法看到
interface (名称; 默认: **all**) – interface 的名称
store-on-disk (yes | no; 默认: **yes**) -是否将信息存储到系统驱动上，如果选择为'no' ，这些信息将存储到 RAM 中，重起后回丢失

仅 **192.168.0.0/24** 的网段监视通过 **ether1** 的传输情况，并将信息写入到磁盘中:

```
[admin@NAT] tool graphing interface> add interface=ether1
allow-address=192.168.0.0/24 store-on-disk=yes
[admin@NAT] tool graphing interface> print
Flags: X - disabled
 #   INTERFACE ALLOW-ADDRESS     STORE-ON-DISK
 0   ether1    192.168.0.0/24    yes
```

```
[admin@NAT] tool graphing interface>
```

带宽 Graphing

操作路径：***/tool graphing queue***

在这个子选项中尼可以指定一个队列**/queue simple** 到图形显示中去。

属性描述

allow-address (*IP 地址*/掩码；默认：**0.0.0.0/0**) -运行访问图形显示的网络地址段，被允许的地址可以试着打开 **http://[Router_IP_address]/graphs/**，如果没有允许将无法看到
allow-target (yes | no；默认：**yes**) – 允许在**/queue simple target-address** 中那些 IP 段访问 graphing web
simple-queue (*名称*；默认：**all**) – 要监测 的 simple queue 名称
store-on-disk (yes | no；默认：**yes**) -是否将信息存储到系统驱动上，如果选择为'no'，这些信息将存储到 RAM 中，重起后回丢失

添加一个 simple queue 到图形列表，simple-queue 名称为 **queue1**，限制访问网段，并存储相关信息到磁盘中：

```
[admin@NAT] tool graphing queue> add simple-queue=queue1 allow-address=192.168.0.0/24
store-on-disk=yes
```

资源图表

操作路径：***/tool graphing resource***

提供路由器在一段时期内资源使用情况：

- CPU usage
- Memory usage
- Disk usage

属性描述

allow-address (*IP 地址*/掩码；默认：**0.0.0.0/0**) -运行访问图形显示的网络地址段，被允许的地址可以试着打开 **http://[Router_IP_address]/graphs/**，如果没有允许将无法看到
store-on-disk (yes | no；默认：**yes**) -是否将信息存储到系统驱动上，如果选择为'no'，这些信息将存储到 RAM 中，重起后回丢失

添加允许监视者的 IP 地址段为 **192.168.0.0/24** ：

```
[admin@NAT] tool graphing resource> add allow-address=192.168.0.0/24 store-on-disk=yes
[admin@NAT] tool graphing resource> print
Flags: X - disabled
 #   ALLOW-ADDRESS     STORE-ON-DISK
 0   192.168.0.0/24    yes
[admin@NAT] tool graphing resource>
```

33.3、Bandwidth-text 带宽测试

带宽测试用于监测远程 MikroTik 路由器的吞吐量（有线或无线），从而去发现网络瓶颈。

协议属性

TCP 测试使用 TCP 协议标准，根据 TCP 算法得出有多少包延迟，被丢弃和其他 TCP 算法特性。关于内部速度设定和状态分析请查看 TCP 协议。吞吐量的统计是用来计算整个 TCP 数据流的大小。TCP 内部链接的大小和使用没有包含在吞吐量的统计中。因此当在测算吞吐量时，这个统计并不像 UDP 协议一样可靠。

UDP 测试发送的数据包的数量是接收方当前所收到包的数量的 110%或更多。要得到链接的最大吞吐量，数据包要设置最大 MTU 为 1500 字节。这并不是 UDP 协议标准所要求的。 通过这样设置，便可以得到近似最大吞吐量。

注：Bandwidth Test 会使用所有可获得的带宽(by default)，并做可能冲击网络的使用性。

Bandwidth Test 比较占用资源。如果需要测试路由器的真实吞吐量，你应该运行 bandwidth test 通过所测路由器。这样做你需要三台路由器相链接：Bandwidth 服务器，测试路由器（Testing Router）和 Bandwidth 客户端：

注：如果用 UDP 协议，那么 Bandwidth Test 所测的数据是 IP header+UDP header+UDP。如果用 TCP 协议，那么 Bandwidth Test 所测的数据仅为 TCP 数据。（不包含 TCP 数据报头和 IP 数据报头）。

Server 配置

操作路径：***/tool bandwidth-server***

属性描述

allocate-udp-ports-from – 分配 UDP 端口
authenticate (yes | no; 默认: **yes**) – 通信要求验证客户端（通过账号和密码）
enable (yes | no; 默认: **no**) – 为客户端启用连接
max-sessions – bandwidth-test 最大的客户端连接数

Bandwidth 服务器:

```
[admin@MikroTik] tool bandwidth-server> print
               enabled: yes
          authenticate: yes
  allocate-udp-ports-from: 2000
           max-sessions: 10
[admin@MikroTik] tool>
```

查看会话连接

```
[admin@MikroTik] tool> bandwidth-server session print
  # CLIENT         PROTOCOL DIRECTION USER
  0 35.35.35.1     udp      send      admin
  1 25.25.25.1     udp      send      admin
  2 36.36.36.1     udp      send      admin
[admin@MikroTik] tool>
```

开启没有客户端的 **bandwidth-test** 服务器

```
[admin@MikroTik] tool bandwidth-server> set enabled=yes authenticate=no
[admin@MikroTik] tool bandwidth-server> print
               enabled: yes
          authenticate: no
  allocate-udp-ports-from: 2000
          max-sessions: 10
[admin@MikroTik] tool>
```

Client 配置

操作路径: ***/tool bandwidth-test***

属性描述

(*IP address*) - 目标主机 IP 地址
assume-lost-time (*时间*; 默认: **0s**) – 设定如果 Bandwidth Server 无响应多久后丢弃连接
direction (receive / transmit / both; 默认: **receive**) - 测试方式
do (名称 | string; 默认: **""**) - 脚本源代码
duration (*时间*; 默认: **0s**) - 测试时长
0s – 测试时间没有被限制
interval (*时间*: 20ms..5s; 默认: **1s**) – 报告间隔时间（秒钟计算）
local-tx-speed (*整型*; 默认: **0**) – 本地发送最大速率(bits per second)
0 – 没有速率限制
local-udp-tx-size (*整型*: 40..64000) – 本地 UDP 发送最大数据包
password (*文本*; 默认: **""**) – 测试的密码
protocol (udp | tcp; 默认: **udp**) – 使用的网络协议
random-data (yes | no; 默认: **no**) – 如果随即数据设置为 yes，Bandwidth 测试数据包的有效载荷，将有不可随机数据流，使连接利用数据压缩，将不会扭曲结果(如果较低性能的 CPU，**random-data** 应设置为 **no**)
remote-tx-speed (*整型*; 默认: **0**) – 远端接收测试的最大速率(bits per second)
0 – 没有速率限制
remote-udp-tx-size (*整型*: 40..64000) – 远端 UDP 发送最大数据包
user (*名称*; 默认: **""**) - 远程用户名

在 10.0.0.211 主机上运行 15 秒发送和接收 **1000**-byte UDP 数据包的带宽测试，用户名为 admin.

```
[admin@MikroTik] tool> bandwidth-test 10.0.0.211 duration=15s direction=both
\... size=1000 protocol=udp user=admin
             status: done testing
```

```
             duration: 15s
           tx-current: 3.62Mbps
   tx-10-second-average: 3.87Mbps
      tx-total-average: 3.53Mbps
           rx-current: 3.33Mbps
   rx-10-second-average: 3.68Mbps
      rx-total-average: 3.49Mbps
[admin@MikroTik] tool>
```

33.4、Torch (即时通信监听)

即时通信监听被称为 torch 它是用来监视正在运行的一个接口的通信情况. 你可以监视通过协议名、源地址、目的地址、端口来分类监视通信情况. Torch 能显示出你已经关闭和发送接受的每个数据流的情况.

操作路径: ***/tool torch***

属性描述

(*名称*) – 用于监视的接口名
dst-address (*IP address/netmask*) – 目的地址和子网掩码是用来通信，任意的目的地址是: 0.0.0.0/0 .
freeze-frame-interval (*时间*) - 屏幕输出暂停的立即时间
port (名称 | 整型) – 端口的名
protocol (any | any-ip | ddp | egp | encap | ggp | gre | hmp | icmp | idpr-cmtp | igmp | ipencap | ipip | ipsec-ah | ipsec-esp | iso-tp4 | ospf | pup | rdp | rspf | st | tcp | udp | vmtp | xns-idp | xtp) – 协议名
  any - 任何以太网和网络协议
  any-ip – 任何网络协议
src-address (*IP address/netmask*) – 源地址和子网掩码是用来进行通信，所有源地址是: 0.0.0.0/0

注: 如果规定了一个特殊的端口,仅有 tcp 和 udp 协议将被过滤，这就是说协议包含 any any-ip tcp udp.除了上行和下行，你已经用命令指定输出(例如,你将得到协议镞仅是以防万一如果协议被明确指出).

下面的例子是利用 telnet 协议监视通过 **ether1** 接口的通信情况:

```
[admin@MikroTik] tool> torch ether1 port=telnet
 SRC-PORT                 DST-PORT                 TX        RX
 1439                     23 (telnet)              1.7kbps    368bps

[admin@MikroTik] tool>
```

IP 协议通过 ether1 接口所显示的情况:

```
[admin@MikroTik] tool> torch ether1 protocol=any-ip
 PRO.. TX        RX
 tcp   1.06kbps   608bps
 udp   896bps    3.7kbps
 icmp  480bps    480bps
```

```
 ospf  0bps      192bps

[admin@MikroTik] tool>
```

IP 协议作用于 10.0.0.144/32 这台主机连接 ether1 接口所显示的情况：

```
[admin@MikroTik] tool> torch ether1 src-address=10.0.0.144/32 protocol=any
 PRO.. SRC-ADDRESS     TX        RX
 tcp   10.0.0.144     1.01kbps   608bps
 icmp  10.0.0.144     480bps     480bps

[admin@MikroTik] tool>
```

tcp/udp 协议通过 ether1 接口所显示的情况：

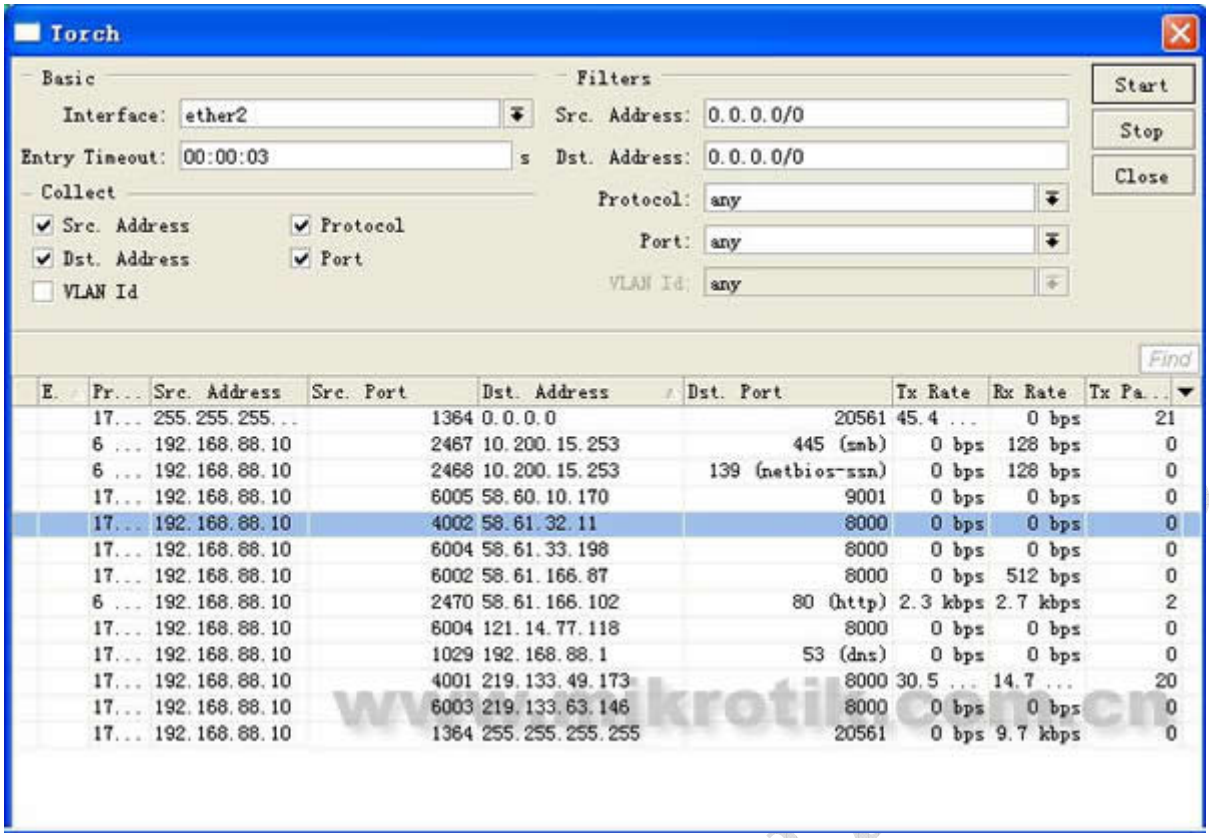
```
[admin@MikroTik] tool> torch ether1 protocol=any-ip port=any
 PRO.. SRC-PORT               DST-PORT               TX        RX
 tcp   3430                   22 (ssh)              1.06kbps   608bps
 udp   2812                   1813 (radius-acct)     512bps    2.11kbps
 tcp   1059                   139 (netbios-ssn)      248bps    360bps
[admin@MikroTik] tool>
```

禁止 QQ 连接到服务器

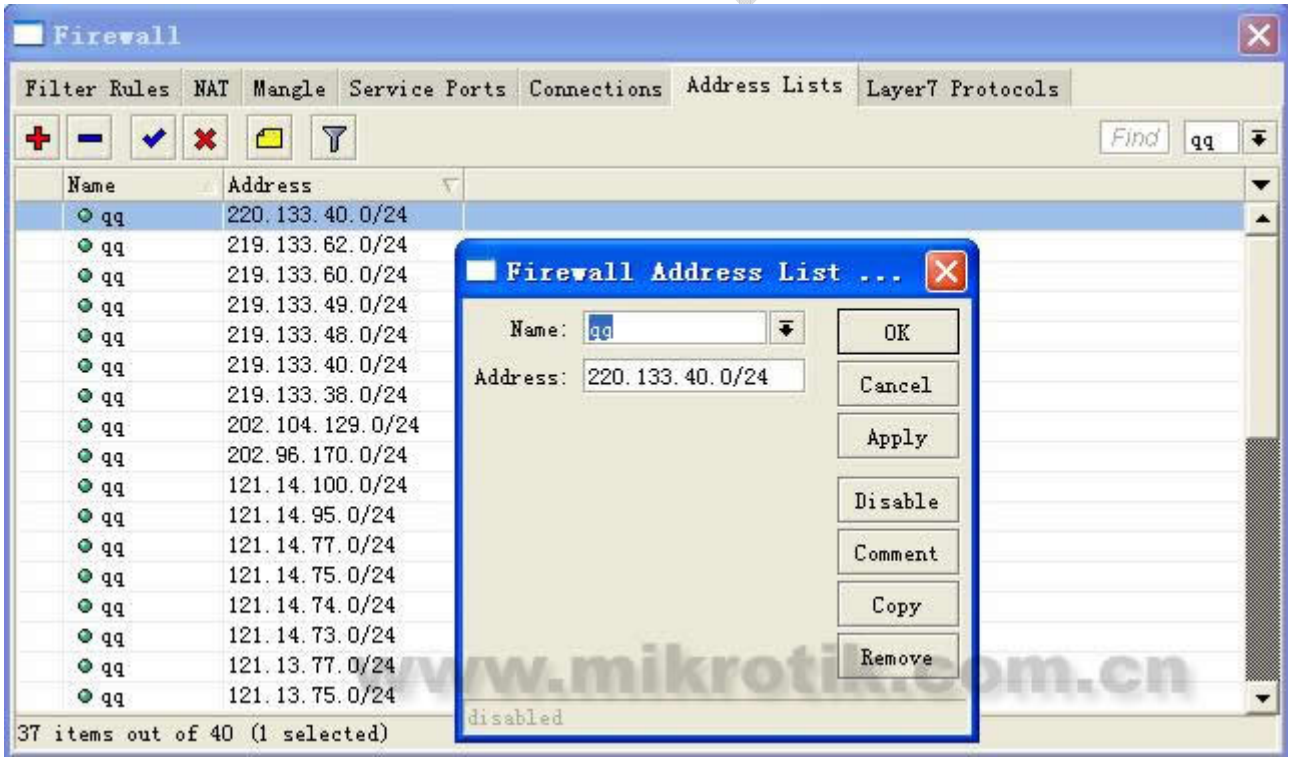
通过端口禁止 QQ 连接是没有作用的，因为 QQ 可以更改连接端口，最好的办法是通过禁止 QQ 连接相应的 QQ 服务器，实现对 QQ 的上网连接。

首先我需要通过 RouterOS 的工具查找每次 QQ 连接服务器的 IP 地址，在这里我们使用的是 RouterOS 自带的 torch 工具，Torch 是用于监测相应网卡的数据连接状态、协议、端口和流量的工具，在/tool 中可以找到 Torch。

我们需要监测 interface 为内网网卡，QQ 连接通常使用 8000 端口连接服务器，我们通过查看 dst-port 为 8000 的连接，当 8000 端口无法连接时，QQ 会使用 80 端口，如下图：



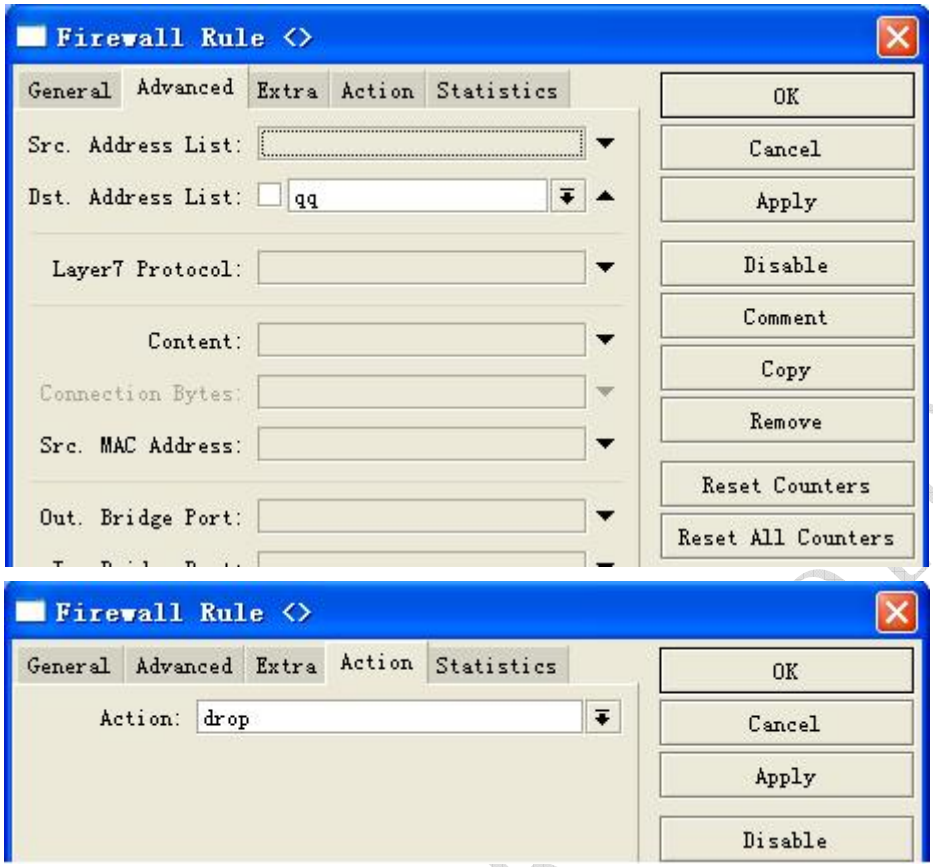
当看到 8000 端口的连接的 dst-address，可以判断为 QQ 的服务器，我通过 ip firewall address-list 填写 QQ 服务器的 IP 地址，比如 220.133.40.11 是 QQ 服务器地址，而添加到 address-list 名字取为 qq，填写 address：220.133.40.0/24，用于规则调用：



当在添加完 QQ 服务器 IP 地址后，在 ip firewall filter 的 forward 链表添加过滤规则：

```
/ip firewall filter add chain=forward dst-address-list=qq action=drop
```

Winbox 操作如下：



通过 torch 得到 QQ 服务器的 IP 地址，需要反复测试，直到 QQ 不能连接到服务器为止。

33.5 E-mail 发送工具

E-mail 工具非常有用，允许从路由发送 e-mail，这个工具被用于备份配置和导出网络管理信息，Email 工具用于纯验证和 TLS 加密，其他模式不支持

操作路径: /tool e-mail

属性

这个子菜单下可以设置 SMTP 服务器

from (*字符*; 默认: <>)显示接收者的名称或者 email 地址。
password (*字符*; 默认: *""*)SMTP 服务器要求验证的密码
server (*IP:Port*; 默认: **0.0.0.0:25**)SMTP 服务器的 IP 地址和端口
username (*字符*; 默认: *""*)SMTP 服务器要求使用的用户名。

Email 发送使用一些命令/tool e-mail send

发送命令采用下面参数:

body (*字符*; 默认:)邮件的实际信息
file (*字符*; 默认:) Email 的附件文件名称

from（*字符*; 默认: ）名称或者 email 地址，.
password（*字符*; 默认: ）密码用于 SMTP 服务的验证
server（*IP:Port*; 默认: ） IP 地址和 SMTP 服务器端口
subject（*字符*; 默认: ）邮件标题.
tls（*yes*/*no*; 默认: **yes**）是否使用 TLS 加密
to（*字符*; 默认: ）目的 emial 地址
user（*字符*; 默认: ）用于 SMTP 验证的用户名

基本事例

这个事例将说明如何导出配置，并每隔 24 小时发送 email

1. 配置 SMTP 服务器

```
[admin@MikroTik] /tool e-mail> set server=10.1.1.1:25 from="router@mydomain.com"
```

2. 添加新的脚本，并取名称“export-send”

```
/export file=export
/tool e-mail send to="config@mydomain.com" subject="$[/system identity get name]  export)
\
body="$[/system clock get date] configuration file" file=export.rsc
```

3.添加计划任务（scheduler），并设置运行 export-send 脚本

```
/system scheduler
add on-event="export-send" start-time=00:00:00 interval=24h
```

查看下面的接口列表：

```
[admin@MikroTik] interface> print
Flags: X - disabled, D - dynamic, R - running
 #    NAME                      TYPE           RX-RATE    TX-RATE    MTU
 0  R ether1                     ether           0          0         1500
 1  R bridge1                    bridge          0          0         1500
 2  R ether2                     ether           0          0         1500
 3  R wlan1                      wlan            0          0         1500
[admin@MikroTik] interface>
```

进入/interface bridge 桥接配置，添加一个桥：

```
[admin@MikroTik] /interface bridge> add
[admin@MikroTik] /interface bridge> prin
Flags: X - disabled, R - running
 0  R name="bridge1" mtu=1500 arp=enabled mac-address=00:00:00:00:00:00
     protocol-mode=none priority=0x8000 auto-mac=yes
     admin-mac=00:00:00:00:00:00 max-message-age=20s forward-delay=15s
     transmit-hold-count=6 ageing-time=5m
[admin@MikroTik] /interface bridge>
```

E-mail 在 winbox 下的操作路径：