MikroTik

RouterOS v3

中文教程

成都网大科技有限公司

Chengdu NetAll Technongy co.Ltd

前 言

MikroTik RouterOS 是一种路由操作系统，并通过该软件将标准的 PC 电脑变成专业路由器，在软件的开发和应用上不断的更新和发展，软件经历了多次更新和改进，使其功能在不断增强和完善。特别在无线、认证、策略路由、带宽控制和防火墙过滤等功能上有着非常突出的功能，其极高的性价比，受到许多网络人士的青睐。MikroTik RouterOS 在具备现有路由系统的大部分功能，能针对网吧、企业、小型 ISP 接入商、社区等网络设备的接入，基于标准的 x86 构架的 PC。一台 586PC 机就可以实现路由功能，提高硬件性能同样也能提高网络的访问速度和吞吐量。完全是一套低成本，高性能的路由器系统。并且现在 MikroTik 开始走向系统化、专业化和多元化，在网络中的路由器、网桥、AP、认证系统和网络监控日志记录都可以使用到，几乎是一套整体的解决方案。

成都网大科技做为 **MikroTik RouterOS** 中国中国官方分销商，提供对 **RouterOS** 软件和硬件的销售、技术支持以及相关应用等 。

目 录

应用说明

主要特征

TCP/IP 协议组：

- **Firewall 和 NAT** – 包状态过滤；P2P 协议过滤；源和目标 NAT；对源 MAC、IP 地址、端口、IP 协议、协议（ICMP、TCP、MSS 等）、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…
- **路由** – 静态路由；多线路平衡路由；基于策略的路由(在防火墙中分类)；RIP v1 / v2, OSPF v2, BGP v4
- **数据流控制** – 能对每个 IP、协议、子网、端口、防火墙标记做流量控制；支持 PCQ, RED, SFQ, FIFO 对列；Peer-to-Peer 协议限制
- **HotSpot** – HotSpot 认证网关支持 RADIUS 验证和记录；用户可用即插即用访问网络；流量控制功能；具备防火墙功能；实时信息状态显示；自定义 HTML 登录页；支持 iPass；支持 SSL 安全验证；支持广告功能。
- **点对点隧道协议** – 支持 PPTP, PPPoE 和 L2TP 访问控制和客户端；支持 PAP, CHAP, MSCHAPv1 和 MSCHAPv2 验证协议； 支持 RADIUS 验证和记录；MPPE 加密；PPPoE 压缩；数据流控制；具备防火墙功能；支持 PPPoE 按需拨号。

- **简单隧道** – IPIP 隧道、EoIP 隧道 (Ethernet over IP)
- **IPsec** – 支持 IP 安全加密 AH 和 ESP 协议；
- **Proxy** – 支持 FTP 和 HTTP 缓存服务器；支持 HTTPS 代理；支持透明代理 ； 支持 SOCKS 协议； DNS static entries；支持独立的缓存驱动器；访问控制列表；支持父系代理。
- **DHCP** – DHCP 服务器；DHCP 接力；DHCP 客户端；多 DHCP 网络；静态和动态 DHCP 租约；支持 RADIUS。
- **VRRP** – 高效率的 VRRP 协议（虚拟路由冗余协议）
- **UPnP** – 支持即插即用
- **NTP** – 网络对时协议服务器和客户端；同步 GPS 系统
- **Monitoring/Accounting** – IP 传输日志记录；防火墙活动记录；静态 HTTP 图形资源管理。
- **SNMP** – 只读访问
- **M3P** – MikroTik 分包协议，支持无线连接和以太网。
- **MNDP** – MikroTik 邻近探测协议；同样支持思科的 CDP。
- **Tools** - ping; traceroute; bandwidth test; ping flood; telnet; SSH; packet sniffer; DDNS。

二层链接

- **Wireless** - IEEE802.11a/b/g wireless client 和访问节点（AP）；Nsetreme 和 Nstreme2 协议；无线分布系统(WDS)；虚拟 AP 功能；40 和 104 bit WEP; WPA pre-shared key 加密；访问控制列表；RADIUS 服务器验证；漫游功能(wireless 客户端)；接入点桥接功能。
- **Bridge** – 支持生成树协议（STP）；多桥接口；桥防火墙；MAC NAT 功能。
- **VLAN** - IEEE802.1q Virtual LAN，支持以太网和无线连接；多 VLAN 支持；VLAN 桥接。
- **Synchronous** - V.35, V.24, E1/T1, X.21, DS3 (T3)媒体类型； sync-PPP, Cisco HDLC，帧中继协议；ANSI-617d (ANDI or annex D)和 Q933a (CCITT or annex A) 帧中继 LMI 类型
- **Asynchronous** – 串型 PPP dial-in / dial-out；PAP, CHAP, MSCHAPv1 和 MSCHAPv2 验证协议；RADIUS 验证和记录；支持串口；modem 池支持 128 个端口。
- **ISDN** - ISDN dial-in / dial-out; PAP, CHAP, MSCHAPv1 和 MSCHAPv2 验证协议；RADIUS 验证和记录；Cisco HDLC, x75i, x75ui, x75bui 队列支持。

硬件要求

- **CPU 和主板** – 核心频率在 100MHz 或更高的单核心 i386 处理器，以及与兼容的主板。
- **RAM** – 最小 32 MiB，最大 1 GiB；推荐 64 MiB 或更高。
- **ROM** – 标准 ATA/IDE 接口、USB 接口和 SATA 接口(SCSI 不支持；RAID 控制器驱动不支持；) 最小需要 64 Mb 空间； Flash 和一些微型驱动器使用 ATA 接口能连接使用。

MIPS 硬件要求

- **支持系统** –RouterBOARD 500 (532, 512 和 511)与 RouterBOARD 100 (133、133c、150、192)
- **RAM** – 最小 16MiB
- **ROM** – 板载 NAND 驱动，最小 64Mb

PPC 硬件要求

- RouterBOARD1000、RouterBOARD600、RouterBOARD333

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。Web 配置提供了多数常用的功能上。 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：

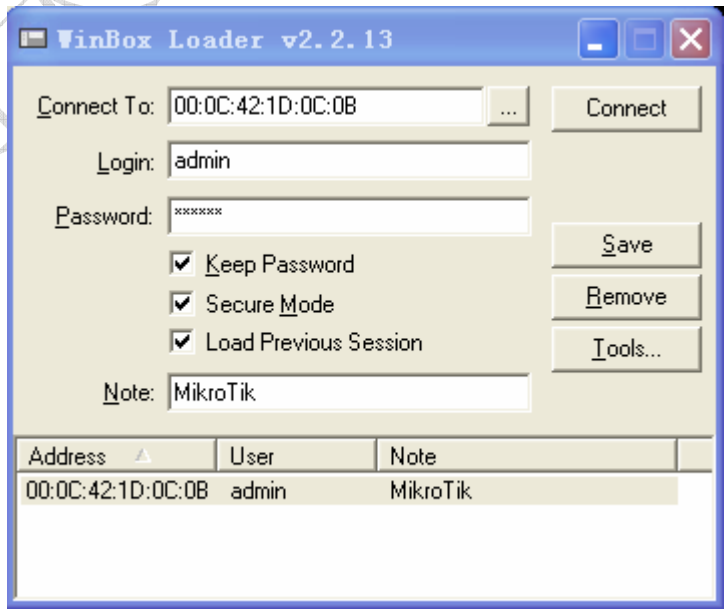
- **本地 teminal console** - PS/2 或 USB 键盘和 VGA 显示卡进行控制
- **Serial console** – 任何 (默认使用 COM1) RS232 异步串口，串口默认设置为 9600bit/s, 8 data bits, 1 stop bit, no parity, hardware (RTS/CTS) flow control。
- **Telnet** – telnet 服务默认运行在 TCP 端口 23
- **SSH** - SSH (安全 shell) 服务默认运行在 TCP 端口 22
- **MAC Telnet** - MikroTik 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。

文档版本: ***3.0***
应用于: ***RouterOS V3.0***

基本设置向导

登陆 RouterOS

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

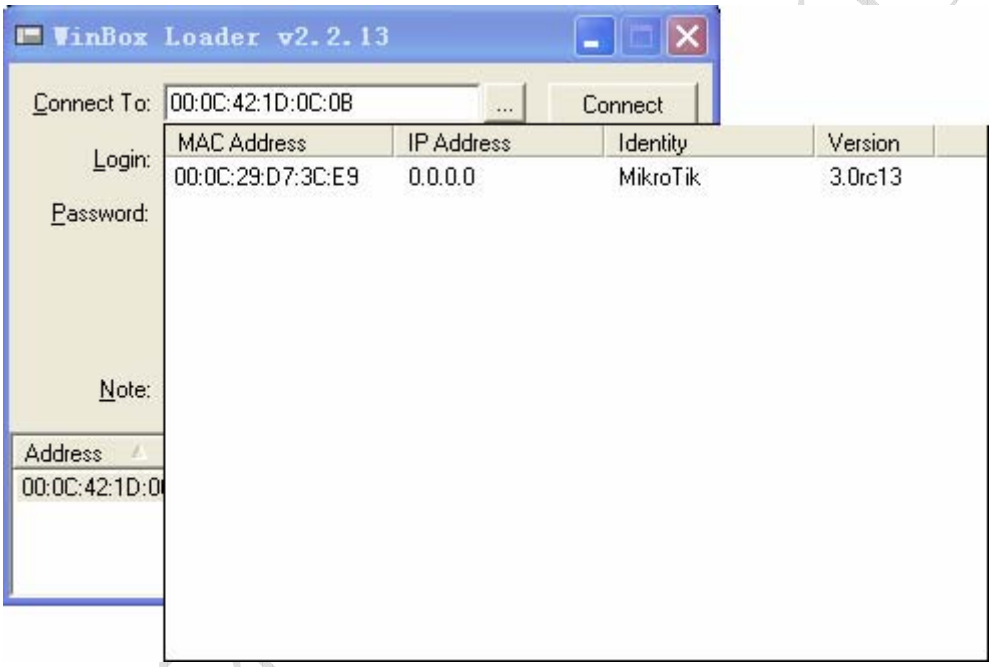
启用。注:在 Winbox 中嵌入了通过 MAC 地址连接路由器的功能，并内置了探测工具。 这样在管理员忘记或复位了路由器后，同样可以通过 MAC 登陆到 RouterOS 上，进行图形界面操作。

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:

- 

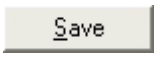
  搜索和显示 MNDP (MikroTik Neighbor Discovery Protocol) 或 CDP (Cisco Discovery Protocol) 设备。可以通过该功能键搜索同一子网内 MikroTik 和 Cisco 设备。并能通过 MAC 地址登陆到 MikroTik RouterOS 进行操作。



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。

- 

  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。

- 

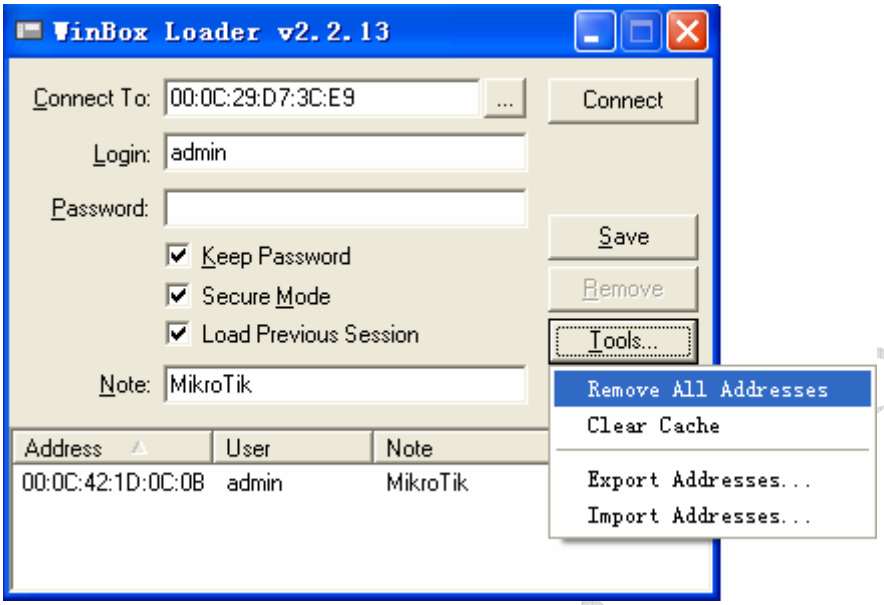
  保存当前连接列表（当需要运行它们时，只需双击）

- 

  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）

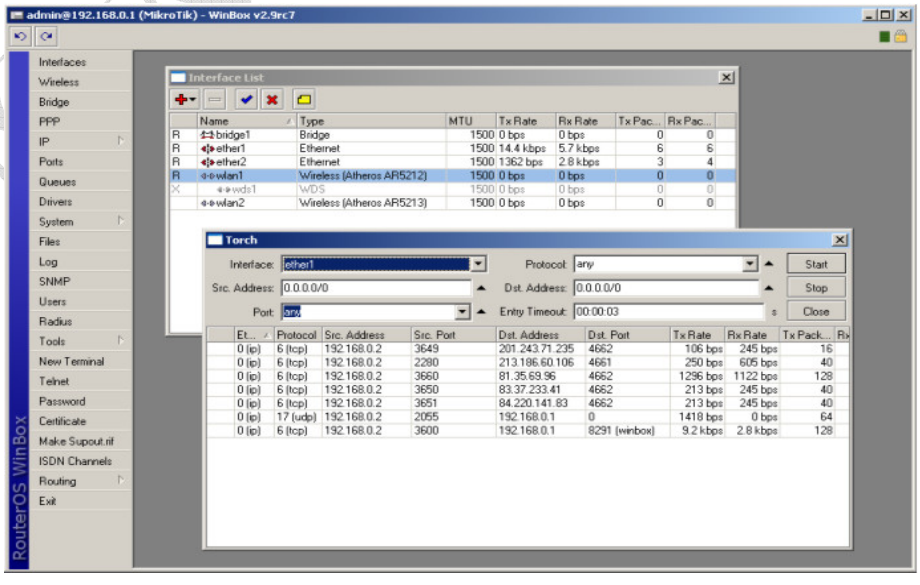
  提供保密并在 winbox 和 RouterOS 之间使用 TLS（Transport Layer Security）协议

- Keep Password（保存密码）

  保存密码到本地磁盘的文本文件中

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

路由器的 winbox 控制台：



Winbox 控制台使用 TCP8291 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 控制台的操作建议：

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

故障分析

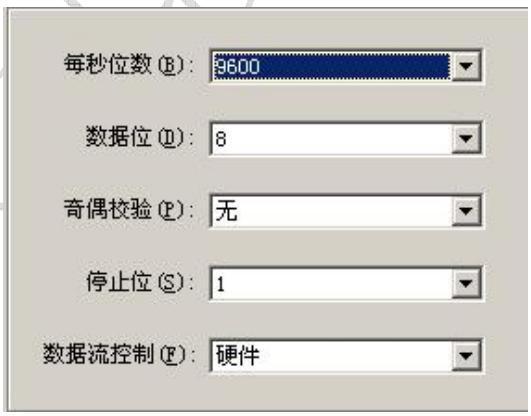
- **我能在 Linux 上运行 Winbox?**

  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。

同样也能用任何 PC 通过标准的 DB9 模式串口线连接到路由器，串口连接的默认设置为每秒位数：9600 bits/s (RouterBOARD 500 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：



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

用一条特殊的串口线连接串行接口（COM），串口线为 DB9 接口，线序排列如下：

路由(DB9f)	Signal	(DB9f)
1, 6	CD, DSR	4
2	RxD	3
3	TxD	2
4	DTR	1, 6
5	GND	5
7	RTS	8
8	CTS	7

RB100 系列、RB300、RB600 的串口线序如下：

DB9f	signal	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。

当登录到终端控制台后，会出现 RouterOS 的登录提示，第一次登录的时候用户名为“admin”密码为空，直接敲回车键进入，如下面的所示：

```
MikroTik v3.0
Login: admin
Password:
```

修改密码可以使用 **/password** 命令

```
[admin@MikroTik] > password
old password:
new password: ************
retype new password: ************
[admin@MikroTik] >
```

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.

MAC telnet 服务器

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.

实例

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.

实例

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

实例

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 地址扫描

操作路径: ***/tool mac-scan***

这个命令是在指定的网络上探测网络支持 MAC telnet 协议的设备。

(*name*) –扫描出的接口名字

MAC telnet 访问客户端

操作路径: ***/tool mac-telnet***

(*MAC address*) – 兼容设备的 mac 地址

实例

```
[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] >
```

添加软件功能包

基本安装仅有 **system** 功能能包，包括基本的 IP 路由和路由管理功能。可用通过添加功能包如：IP Telephony，OSPF，wireless 等，增加路由器的功能。你需要下载软件功能包，并上传到路由器

添加软件功能包，应该和当前的系统版本相同，如果不是相同的版本，功能包将不会被安装，上传功能包是通过 FTP 的方式，将功能包，放到路由器的 FTP 空间中，上传完后重启路由器，路由器将自动安装功能包。

终端控制向导

欢迎界面和命令提示

在登录后路由器后将会看到 RouterOS™ 欢迎界面和命令提示：

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




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

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>
[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， 如果输入的内容含糊不清，第二次键入 [Tab]就会给出存在的选项
/	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
"word1 word2"	Specifies a string of 2 words that contain a space

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
```

基本配置

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>
```

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：

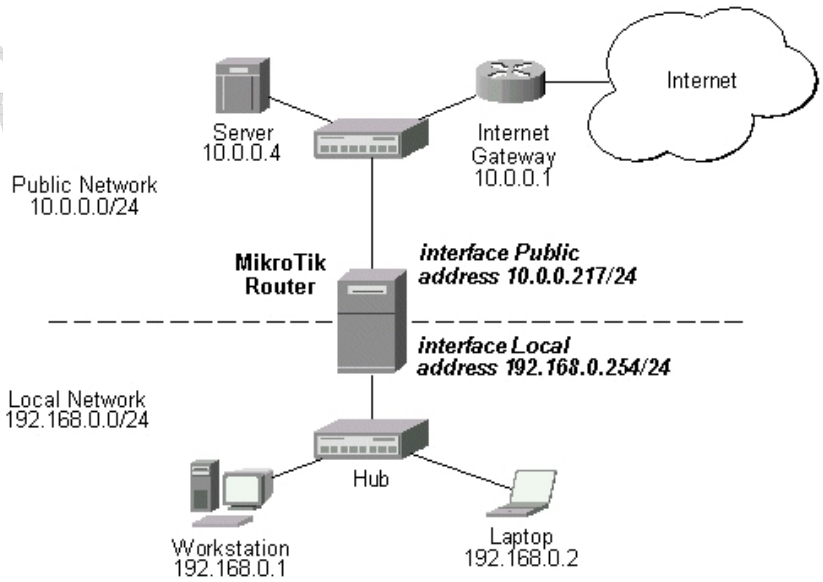
```
your choice: a
enable interface:
ether1  ether2  wlan1
enable interface: ether1
ip address/netmask: 10.1.0.66/24
#Enabling interface
/interface enable ether1
#Adding IP address
/ip address add address=10.1.0.66/24 interface=ether1 comment="added by setup"
 + a - add ip address
 * g - setup default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

基本事例

配置一个 RouteroS 分为三个步骤：
第一步：检查 Interface 上的网卡是否正确安装，然后在 ip address 中配置 IP 地址；
第二步：在配置好 IP 地址后，在 ip routes 中配置默认网关，配置完后检查是否到外网默认网关正常；
第三步：在 ip firewall nat 中配置 NAT 伪装，隐藏内部网络。

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（公网和本地网络）：

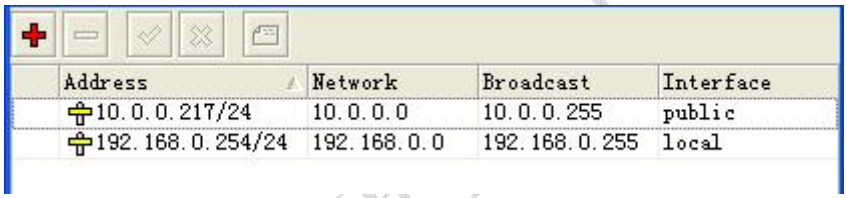
- 本地网络使用地址为：192.168.0.0 子网淹码 24-bit（255.255.255.0）。路由器的地址在这个网络中为 192.168.0.254
- ISP 的网络为 10.0.0.0 子网淹码 24-bit（255.255.255.0）。路由器的地址是在网络中为 10.0.0.217

通过下面的指令添加地址：

```
[admin@MikroTik] ip address> add address 10.0.0.217/24 interface Public
[admin@MikroTik] ip address> add address 192.168.0.254/24 interface Local
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS          NETWORK        BROADCAST      INTERFACE
  0   10.0.0.217/24     10.0.0.217     10.0.0.255     Public
  1   192.168.0.254/24   192.168.0.0    192.168.0.255   Local
[admin@MikroTik] ip address>
```

这里，子网淹码在 address 变量中指定，或者也可以通过在 netmask 变量中设置 255.255.255.0。网段和广播地址在输入时没有指定，这些可以由 RouterOS 自动计算出来。

请注意：在 IP 地址被分配到路由器的不同网卡上时，应属于不同的网络，下面是 winbox 中的设置情况：



查看路由

你可以看到两个带有动态 dynamic (D)和连接 connected (C)的路由，当地址添加后会在路由中自动添加动态路由：

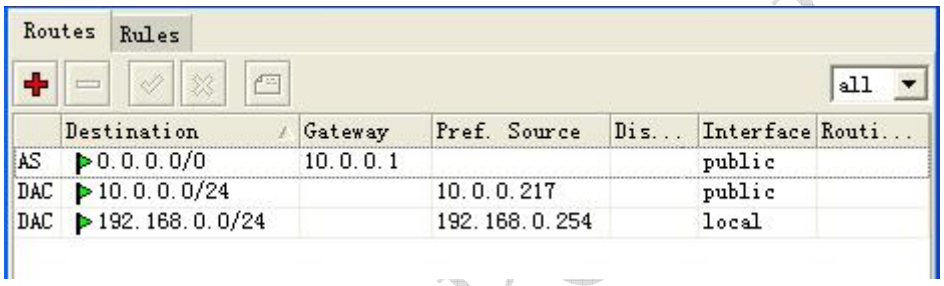
```
[admin@MikroTik] ip route> print
Flags: X - disabled, I - invalid, D - dynamic, J - rejected,
C - connect, S - static, R - rip, O - ospf, B - bgp
    #    DST-ADDRESS       G GATEWAY        DISTANCE INTERFACE
    0 DC 192.168.0.0/24     r 0.0.0.0         0       Local
    1 DC 10.0.0.0/24        r 0.0.0.0         0       Public
[admin@MikroTik] ip route> print detail
Flags: X - disabled, I - invalid, D - dynamic, J - rejected,
C - connect, S - static, R - rip, O - ospf, B - bgp
    0 DC dst-address=192.168.0.0/24 preferred-source=192.168.0.254
         gateway=0.0.0.0 gateway-state=reachable distance=0 interface=Local
    1 DC dst-address=10.0.0.0/24 preferred-source=10.0.0.217 gateway=0.0.0.0
         gateway-state=reachable distance=0 interface=Public
[admin@MikroTik] ip route>
```

添加默认路由

在下面的事例中将添加默认路由(destination 0.0.0.0 (any), netmask 0.0.0.0 (any)) 。在这个事例中 ISP 的网关是 10.0.0.1，通过 **Public** 接口

```
[admin@MikroTik] ip route> add gateway=10.0.0.1
[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 10.0.0.1       1       Public
   1 DC 192.168.0.0/24     r 0.0.0.0        0       Local
   2 DC 10.0.0.0/24        r 0.0.0.0        0       Public
[admin@MikroTik] ip route>
```

这里，默认路由被列入标号#0，同样我们看到，网关 10.0.0.1 能在接口'Public'通过。如果网关没有被正确的指定，'interface'变量值将会无法确定（unknown）。Winbox 添加后情况如下：



测试网络连接

从现在起，**/ping** 指令可以用来测试网络连接情况。

```
 [admin@MikroTik] ip route> /ping 10.0.0.4
10.0.0.4 64 byte ping: ttl=255 time=7 ms
10.0.0.4 64 byte ping: ttl=255 time=5 ms
10.0.0.4 64 byte ping: ttl=255 time=5 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 5/5.6/7 ms
[admin@MikroTik] ip route>
[admin@MikroTik] ip route> /ping 192.168.0.1
192.168.0.1 64 byte ping: ttl=255 time=1 ms
192.168.0.1 64 byte ping: ttl=255 time=1 ms
192.168.0.1 64 byte ping: ttl=255 time=1 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 1/1.0/1 ms
[admin@MikroTik] ip route>
```

如果路由器的地址 192.168.0.254 在 windows 工作站的 TCP/IP 协议中配置为默认网关，这时你就能 **ping** 通路由器

```
C:\>ping 192.168.0.254
```

```
Reply from 192.168.0.254: bytes=32 time=10ms TTL=253
Reply from 192.168.0.254: bytes=32 time<10ms TTL=253
Reply from 192.168.0.254: bytes=32 time<10ms TTL=253
C:\>ping 10.0.0.217
Reply from 10.0.0.217: bytes=32 time=10ms TTL=253
Reply from 10.0.0.217: bytes=32 time<10ms TTL=253
Reply from 10.0.0.217: bytes=32 time<10ms TTL=253
C:\>ping 10.0.0.4
Request timed out.
Request timed out.
Request timed out.
```

注： 你不能访问超过路由器的任何网络(10.0.0.0/24 的网络和 Internet)，你需要作下面的设置：

- 使用源地址翻译（masquerading)，通过 MikroTik 路由隐藏你的私有网络 192.168.0.0/24 (查看下面的信息)
- 在 ISP 的网关 10.0.0.1 上添加静态路由，指明到目标地址 192.168.0.0/24 通过 10.0.0.217 的主机，这时所有在 ISP 上的网络主机，包括服务器，将能连接到你的私有网络。

在设置路由时，你需要了解一些配置 TCP/IP 的网络知识，当你遇到配置网络安装困难时，我们建议你获取更多的网络技术知识。

下面我将讨论隐藏'hiding'私有的 LAN192.168.0.0/24 在 ISP 给的 10.0.0.217 的背后。

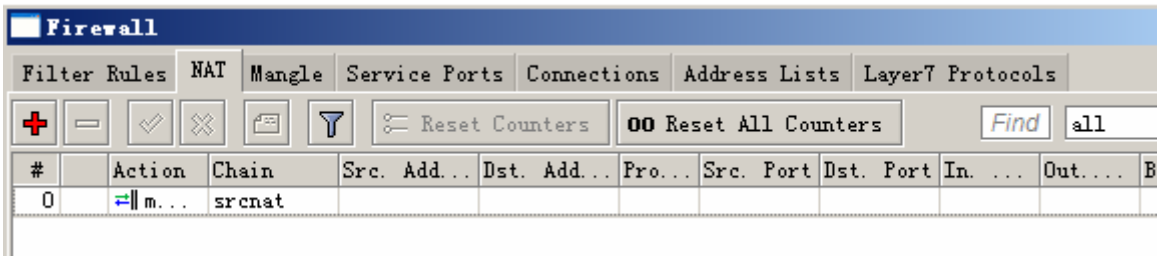
伪装的应用事例（Masquerading）

如果你想隐藏'hiding'私有的 LAN192.168.0.0/24 在 ISP 给的 10.0.0.217 的背后，你需要使用 RouterOS 的源地址翻译。伪装将改变源 IP 地址和数据包端口，即将 192.168.0.0/24 改为 10.0.0.217 去回应 ISP 的网络。

使用伪装时添加一条 NAT 规则在防火墙配置中，执行'masquerade',如下面：

```
[admin@MikroTik] ip firewall nat> add chain=srcnat action=masquerade out-interface=Public
[admin@MikroTik] ip firewall nat> print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=srcnat out-interface=Public action=masquerade
```

Winbox 添加后如下



注： 如果需要了解很多的 NAT 信息，建议参阅 NAT 说明文档。

以上是如何配置一个简单 RouterOS 网络的应用实例。

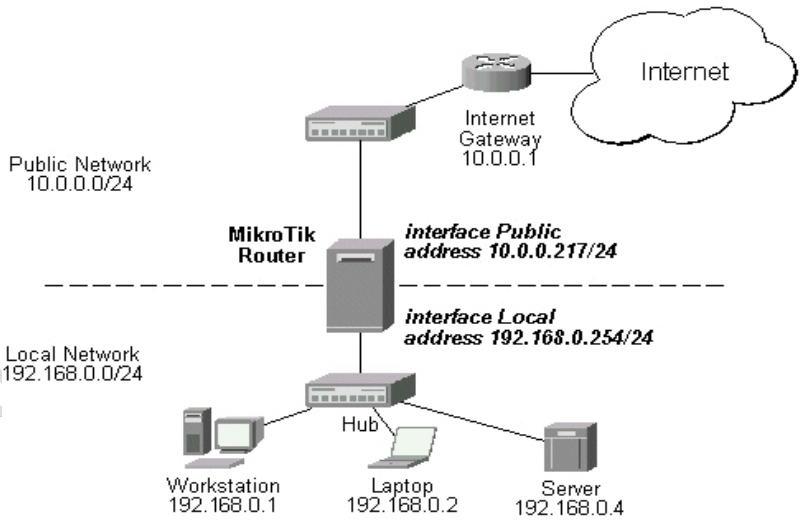
带宽管理事例

假设你想要限制所以的 LAN 内主机 192.168.0.88 的下行带宽为 128kbps 和上行带宽为 64kbps:

```
[admin@MikroTik] queue simple> add target-address=192.68.0.88 max-limit=64000/128000
interface=Local
[admin@MikroTik] queue simple> print
Flags: X - disabled, I - invalid, D - dynamic
 0   name="queue1" target-address=192.68.0.88 dst-address=0.0.0.0/0
     interface=Local queue=default priority=8 limit-at=0/0
     max-limit=64000/128000
[admin@MikroTik] queue simple>
```

NAT 端口映射事例

假如我们将以前的服务器从公网移动到私网中去，并想让其他的公网来访问我们的服务器，这时就需要用到 NAT：



现在服务器的地址是 192.168.0.4，并且我们在服务器上运行 80 端口监听 web 服务，我们想通过公网地址 10.0.0.217：80 端口访问该服务器，即我们就需要在 MikroTik 路由器上作静态的网络地址翻译(NAT) ,这样通过公网地址 10.0.0.217 端口 80 将数据传输到本地网络的 192.168.0.4:80，在目标地址上配置目标地址和端口：

```
[admin@MikroTik] ip firewall nat> add chain=dstnat action=dst-nat protocol=tcp
dst-address=10.0.0.217/32 dst-port=80 to-addresses=192.168.0.4
[admin@MikroTik] ip firewall nat> pr
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=dstnat dst-address=10.0.0.217/32 protocol=tcp dst-port=80
     action=dst-nat to-addresses=192.168.0.4 to-ports=0-65535
```

系统管理

基本信息

现在指导你是如何使用指令执行下面的功能：

- 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

MikroTik RouterOS 将配置备份为二进制文件，通过 FTP 访问路由器下载备份文件，并可以通过备份文件还原路由器设置。T

MikroTik RouterOS 通过导出配置可用打印出配置信息到终端控制的屏幕上或生成文本文件（脚本），同样使用 FTP 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电脑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。 在 **/system reset** 复位系统后，上传备份文件到 RouterOS 中，并通过 **/system backup** 中的 **load** 指令载入配置在还原系统配置

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
...
```

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被激活。**export** 同样也可以通过 **file** 生成脚本配置文件，可用 FTP 下载下来。

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>
```

制作一个仅一个项目的导出文件：

```
[admin@MikroTik] ip address> export file=address1 from=1
[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
1  address1.rsc                   script       201       dec/23/2003 13:22:57
[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***

在根目录使用**/import file_name** 指令还原指定的导出文件。这种还原是用于部分的配置丢失。

注：导入指令不可能导入收有的路由器配置只能导入部分的配置如，firewall rules 中的策略

指令描述

file=[filename] – 载入需要导入的路由器配置文件

例如：

使用下面的指令操作载入保存的配置文件：

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

复位

操作路径：***/system***

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，接口将会被禁用，在 **reset** 指令执行后路由器将会重起。

例如：

```
[admin@MikroTik] > system reset
Dangerous! Reset anyway? [y/N]: n
action cancelled
[admin@MikroTik] >
```

系统资源 resource

操作路径：***/system resource***

通过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、内存和硬盘使用情况。

事例：

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] >
```

IRQ 使用监测

命令路径: ***/system resource irq print***

显示当前 IRQ 使用情况

事例

```
[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

事例

```
[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 (*read-only: text*) – 设备编号
name (*read-only: text*) – USB 端口名称
speed (*read-only: integer*) – 该端口工作的带宽速度
vendor (*read-only: text*) – USB 设备销售商名称

事例

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 (*read-only: text*) – 设备类型
device (*read-only: text*) – 设备编号
device-id (*read-only: integer*) – 十六进制设备 ID
irq (*read-only: integer*) – 该设备使用的 IRQ 编号
memory (*read-only: integer*) – 该设备使用的内存长度
name (*read-only: text*) – 设备名称

vendor (*read-only: text*) – 设备销售商名称
vendor-id (*read-only: integer*) – 设备十六进制销售商

事例

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>
```

重启

操作路径：***/system reboot***

当升级或安装新软件功能包时需要重新启动路由器，在整个系统重启周期中执行功能包安装。

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。

事例

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

关机

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 30 秒（如果没有升级操作，通常最少需要 10 秒）才能安全关闭电源。

事例

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

路由器身份

操作路径：***/system identity***

通过命令可以查看路由身份名，这个身份名同样被用于 DHCP 客户端的“主机名（host name）”

事例

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] >
```

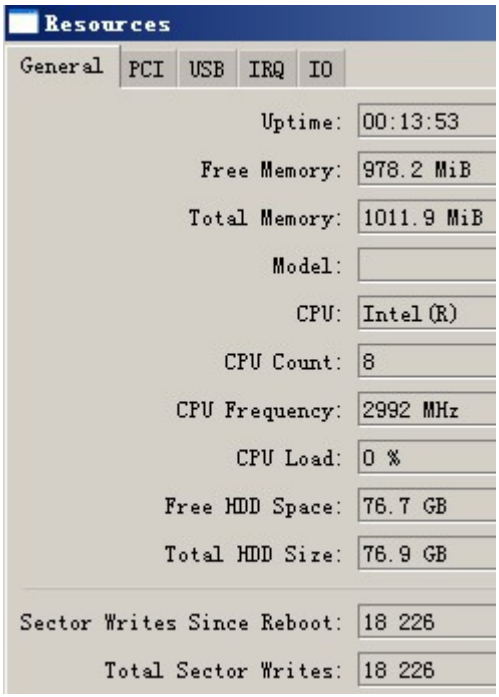
启用多 CPU

操作路径：***/system hardware***

如果你使用的是多 CPU，但你在启动 RouterOS v3 后，在***/system resource*** 中查看到只有一个 CPU 在运行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。

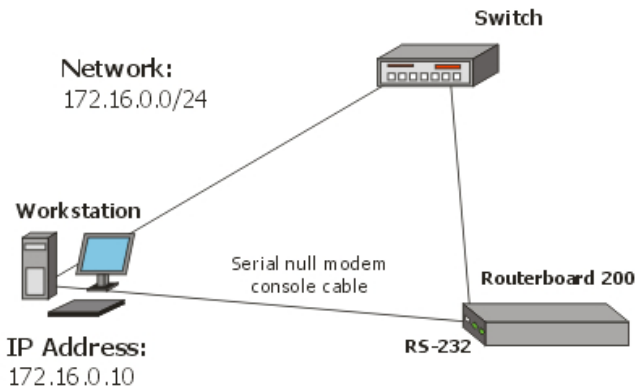


通过 NetInstall 安装和复位 RouterRoard

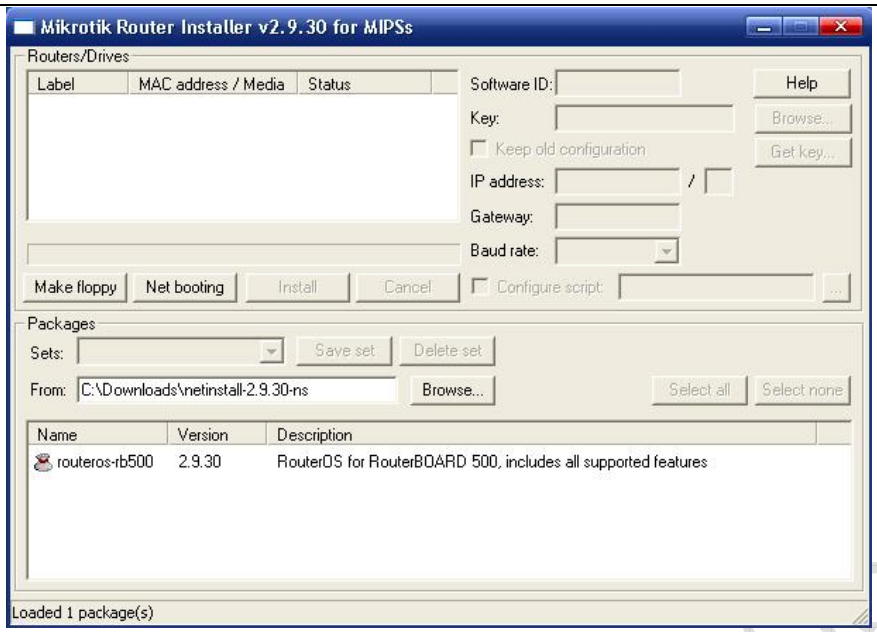
网络安装和复位 RouterRoard

这个事例将介绍如何一步一步在一个 RouterBoard 上安装软件，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(Hub)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。



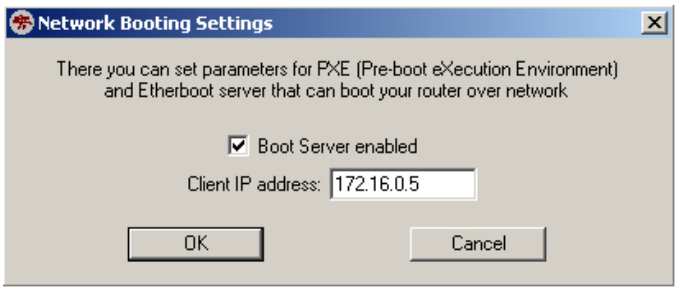
2. 在你的电脑上运行 NetInstall for MIPS 程序 ，确定软件包(*.npk 文件)在你本地磁盘上。NetInstall for MIPS:



3. 设置好 Windows 工作站的超级终端连接，每秒位数为 115200，其他参数为系统默认值：

4. 输入 Boot Server 客户端的 IP 地址。设置一个 IP 地址段，用于临时分配给 RouterBoard 的 IP 地址（该事例的地址为 **172.16.0.0/24**）。

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。



5. 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 532

CPU frequency: 330 MHz
  Memory size:  32 MB

Press any key within 2 seconds to enter setup.

RouterBOOT-1.13
What do you want to configure?
   d - boot delay
   k - boot key
   s - serial console
   o - boot device
   u - cpu mode
   f - try cpu frequency
   c - keep cpu frequency
   r - reset 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

```
 RouterBOOT-1.13
What do you want to configure?
   d - boot delay
   k - boot key
   s - serial console
   o - boot device
   u - cpu mode
   f - try cpu frequency
   c - keep cpu frequency
   r - reset configuration
   e - format nand
   g - upgrade firmware
   i - board info
   p - boot protocol
   t - do memory testing
   x - exit setup
your choice: o - boot device
```

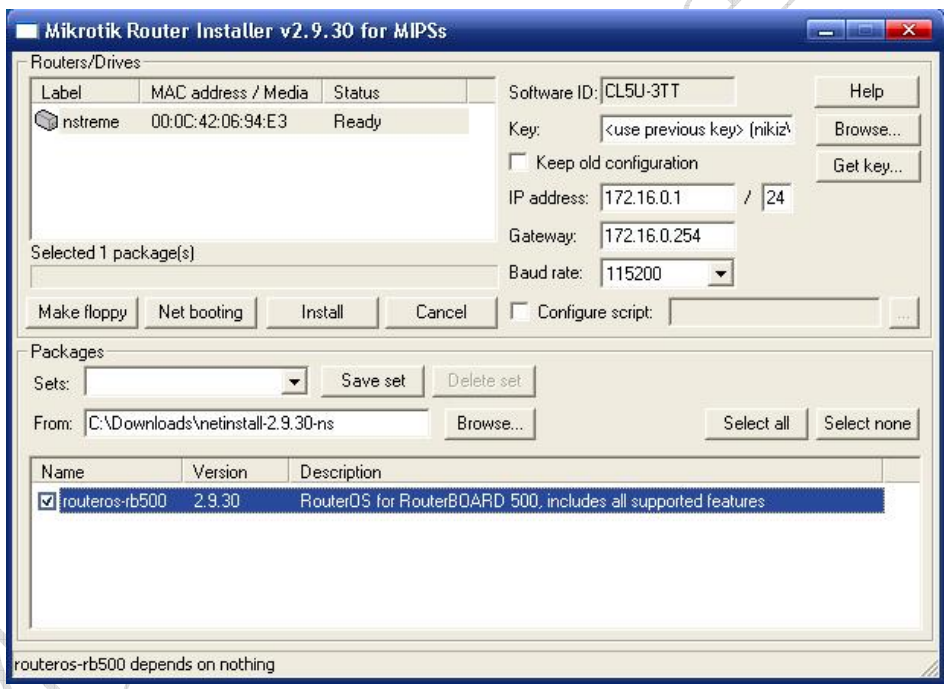
按“e”键，是选择从以太网卡引导 RouterBoard：

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

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。

6. 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 工作站，将会出现一个新的路由器列表，显示当前连接的 RouterBoard 设备。



连接完成后，需要选择安装的功能包或文件的路径，是否保留原来的配置，设置给路由器新的 IP 地址和网关。还有就是传输的波特率选择对应的 115200。

当完成设置后，就可以按 Install 键开始安装 RouterOS.

7. 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。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。

服务、协议及端口

基本信息

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。

修改服务设置

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

描述

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

接口设置（Interface）

基本信息

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。

接口状态

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：

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>
```

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

例如：

多接口监视：

```
[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]
```

以太网接口（Ethernet）

基本信息

MikroTik RouterOS支持各种以太网卡，完全支持的以太网卡型号在 Device Driver List可以找到 。

功能规格

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
硬件要求：*Not significant*

以太网接口配置

操作路径：***/interface ethernet***

属性描述

name (*名称*; 默认: **etherN**) – 分配接口名称,。
arp (disabled | enabled | proxy-arp | reply-only; 默认: **enabled**) – 地址解析协议
mtu (*整型*; 默认: **1500**) – 最大传输单位
disable-running-check (yes | no; 默认: **yes**) – 检测运行情况。
mac-address (只读: *MAC 地址*) – 以太网卡的介质访问地址
auto-negotiation (yes | no; 默认: **yes**) – 当启用，接口会获取最好的网络连接。
full-duplex (yes | no; 默认: **yes**) – 定义数据传输全双工
long-cable (yes | no; 默认: **no**) – 改变电缆传输的长度设置(只适用于 NS DP83815/6 卡). 电缆长度超过 50m，设置"long-cable=yes"
speed (10 Mbps | 100 Mbps | 1000 Mbps) – 设置以太网的数据传输速度，参数由以太网卡支持的最大数据传输率确定。

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
[admin@MikroTik] > interface enable ether1
[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
[admin@MikroTik] > interface ethernet
[admin@MikroTik] interface ethernet> print
Flags: X - disabled, R - running
 #    NAME                             MTU   MAC-ADDRESS       ARP
 0  R ether1                            1500  00:0C:42:03:00:F2 enabled
[admin@MikroTik] interface ethernet> print detail
Flags: X - disabled, R - running
 0  R name="ether1" mtu=1500 mac-address=00:0C:42:03:00:F2 arp=enabled
     disable-running-check=yes auto-negotiation=yes full-duplex=yes
     long-cable=no speed=100Mbps
[admin@MikroTik] interface ethernet>
```

接口状态监视

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：

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

IP 地址与 ARP

基本信息

下面的手册讨论 IP 地址管理和地址解析协议设置， IP 地址在连接其它网络的设备使用 TCP/IP 协议，依次在一个物理网络中连接两个设备，借助于地址解析协议和 ARP 地址。

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

IP 地址

操作路径：***/ip address***

在 IP 网络中，IP 地址是确认每个主机地址为目的，一个典型的 IP 地址(IPv4)由 4 个 8 位组成适当的路由寻址还需要子网掩码值，即那一段完整的 IP 地址位访问主机地址，那一段到网络地址。在大多数事例中，需要具体指名地址、掩码和接口参数。网络起始范围和广播地址能被自动计算出来。

能在一个接口上添加多个 IP 地址或在接口上不分配任何地址， 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, pptp, 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

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>
```

地址解析协议

操作路径：***/ip arp***

尽管 IP 数据包对话通过 IP 地址，但硬件地址必须使用实际的传输数据从一个主机到另一个，通过地址解析协议从 OSI 第 3 层解析第 2 层的 MAC 地址。 一个路由器会有一个当前 ARP 登记表，通常这个表是建立为动态，但为增强网络安全性，建立一个静态的即添加静态的 ARP。

属性描述

address (*IP 地址*) – 相应的 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 1024.

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 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：

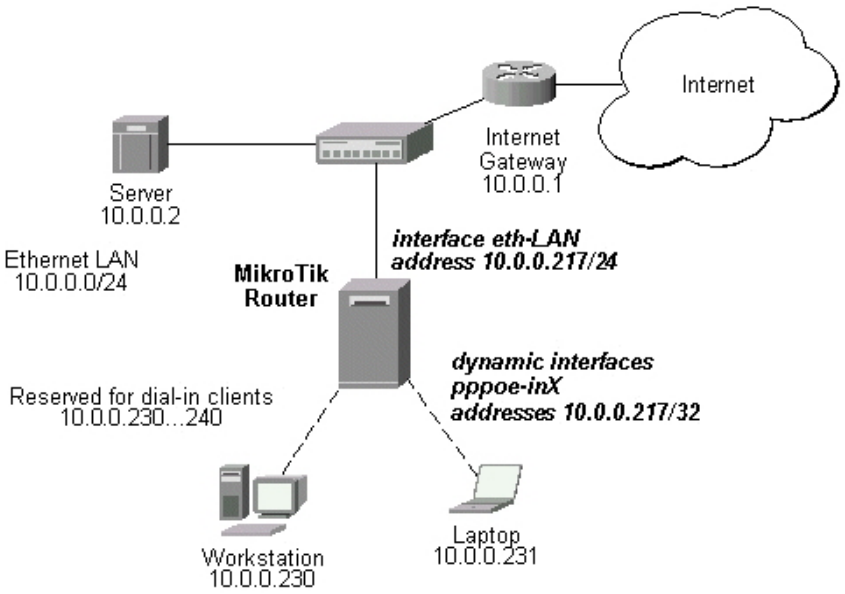
```
[admin@MikroTik] ip arp> /interface ethernet set ether2 arp=reply-only
[admin@MikroTik] ip arp> print
Flags: X - disabled, I - invalid, H - DHCP, D - dynamic
  #   ADDRESS         MAC-ADDRESS       INTERFACE
  0 D 10.5.7.242      00:A0:24:9D:52:A4 ether1
  1   10.10.10.10     06:21:00:56:00:12 ether2

[admin@MikroTik] ip arp>
```

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>
```

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：Reply-only

```
[admin@MikroTik] ip arp> /interface ethernet set LAN arp=reply-only
```

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

Proxy-ARP 有哪些优点?

最主要的一个优点就是能够在不影响其他 router 的路由表的情况下在网络上添加一个新的 router,这样使得子网的变化对主机是透明的

proxy ARP 应该使用在主机没有配置默认网关或没有任何路由策略的网络上

proxy-ARP 带来的哪些负面影响?

1.增加了某一网段上 ARP 流量

2.主机需要更大的 ARP table 来处理 IP 地址到 MAC 地址的映射

3.安全问题,比如 ARP 欺骗(spoofing)

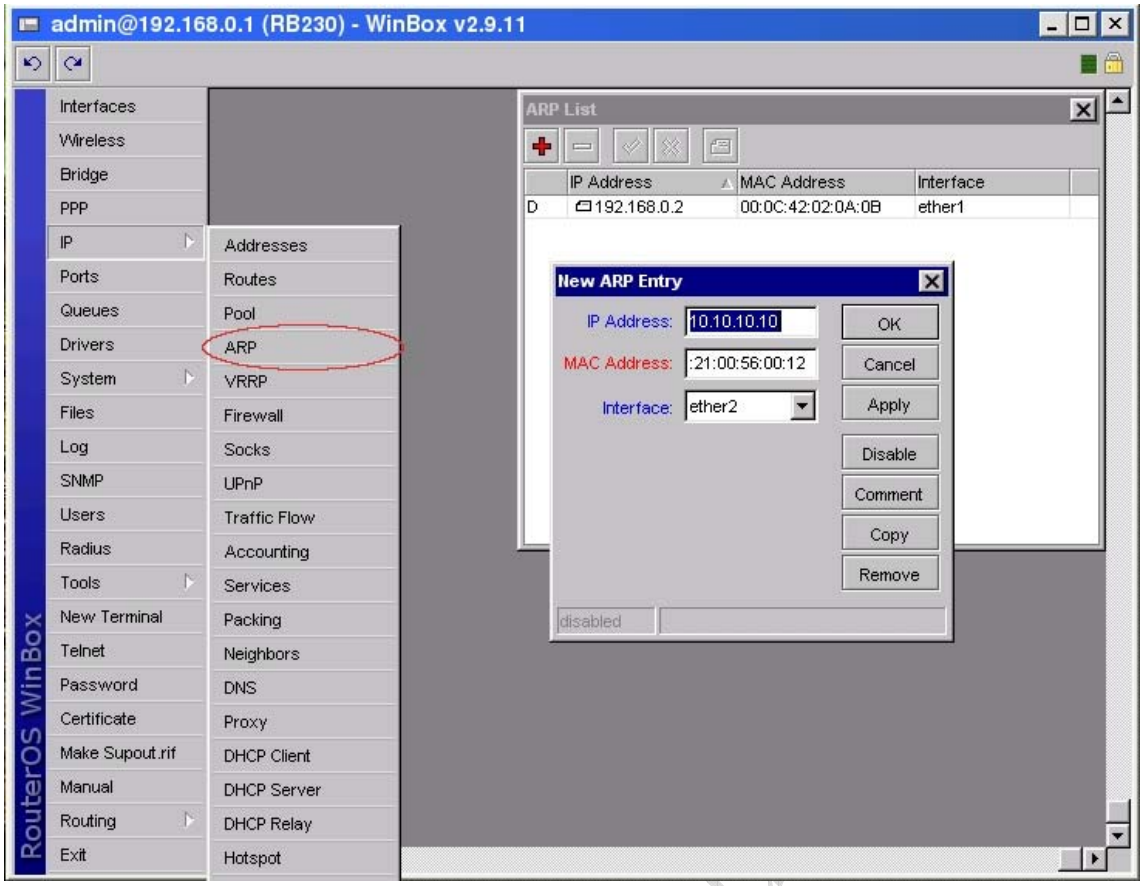
4.不会为不使用 ARP 来解析地址的网络工作

5.不能够概括和推广网络拓扑

如何使用 ARP 邦定

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 中添加一个静态主机的 ARP 记录。

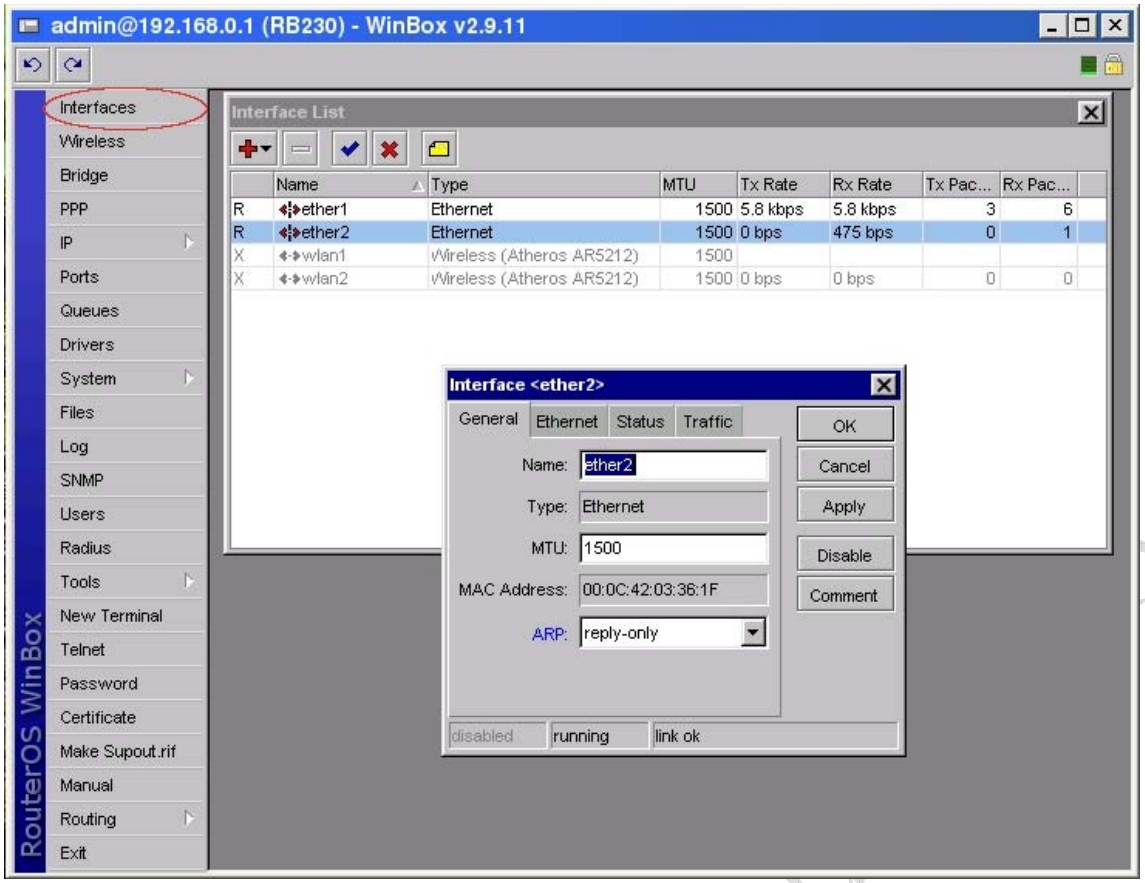


或者通过命令操作：

```
[admin@RB230] ip arp> add address=10.10.10.10 interface=ether2 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 ether2 interface 仅回应静态 ARP 的请求：



命令操作如下：

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

路由设置（Routes）

基本信息

下面的手册概述了 IP 路由的管理，ECMP（equal-cost multi-path）路由技术和策略路由等，具体使用哪一种路由方式，需要根据用户网络情况来选择。

操作规则

需要功能包: ***system***
软件等级: *Level1*
操作路径: ***/ip route***, ***/ip policy-routing***
技术标准: IP (RFC 791)

NAT RouterOS 有下列类型的路由：

- **动态连接路由** 是当在一个网卡上添加了 IP，会自动创建一个动态的路由。
- **静态路由** 是用户自定义将数据传输到指定的网络去的路由，这需要指定默认的网关。

当添加一个 IP 地址后，会自动创建一个动态的路由连接，你不需要手动添加连接路由器的路由配置，除非你使用一些路由协议（RIP 或 OSPF）你就需要定义静态路由到指定的网络，或指定默认网关。

ECMP (Equal Cost Multi-Path) 路由

当使用在到一个目标网络多于一个网关时，可以称为“Equal-Cost Multi-Path Routing”即将其作负载均衡。每一对新的源/目标 IP 会选择一个新的网关。例如，一个 FTP 仅使用一个连接，但当一个新连接到不同的服务器就会使用其他的连接。

- 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。

基于策略的路由

根据路由算法将一个数据包选择到期望的一个网关上，在 RouterOS 中操作过程如下：

- 标记期望的数据包，设置一个 **routing-mark**
- 为标记的数据包选择一个网关

路由

操作路径：***/ip route***

在路由子选项中，你可以配置静态路由、Equal Cost Multi-Path、Policy-Based Routing

注： 你能指定两个或更多的网关在路由中，而且你能重复一些路由的不同类型的参数多次设置到一个网关上。

事例

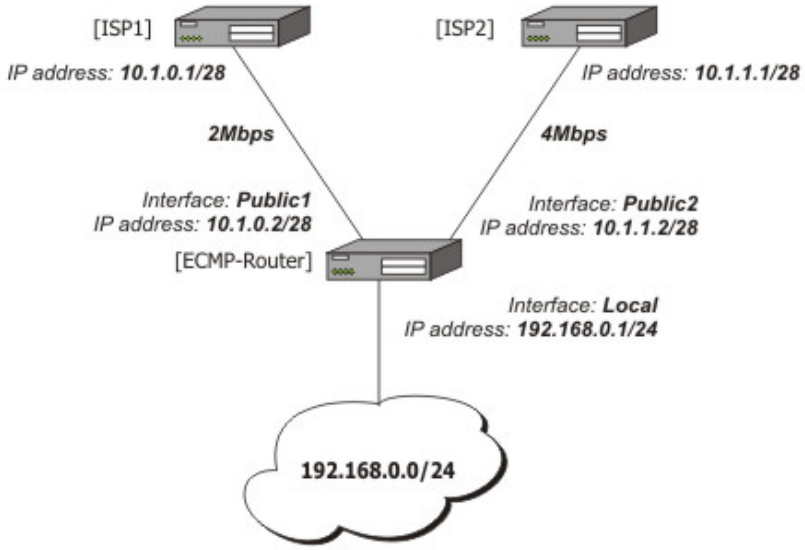
在一个路由器的两张网卡和两个 IP 地址中，添加两个静态路由到网络 10.1.12.0/24 和 0.0.0.0/0 (默认的目标的地址) ：

```
[admin@NAT] ip route> add dst-address=10.1.12.0/24 gateway=192.168.0.253
[admin@NAT] ip route> add gateway=10.5.8.1
[admin@NAT] 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@NAT] ip route>
```

应用事例

Equal Cost Multi-Path 静态多线路由

考虑下面的网络环境，所以的数据都是从一个网络 **192.168.0.0/24** 到两个网关 **10.1.0.1** 和 **10.1.1.1**。



注，ISP1 给我们的带宽是 2Mbps ， ISP2 是 4Mbps，因此我们想要一个 1:2 的传输比(1/3 从 **192.168.0.0/24** 的数据走 ISP1，2/3 的通过).

路由器的 IP 地址：

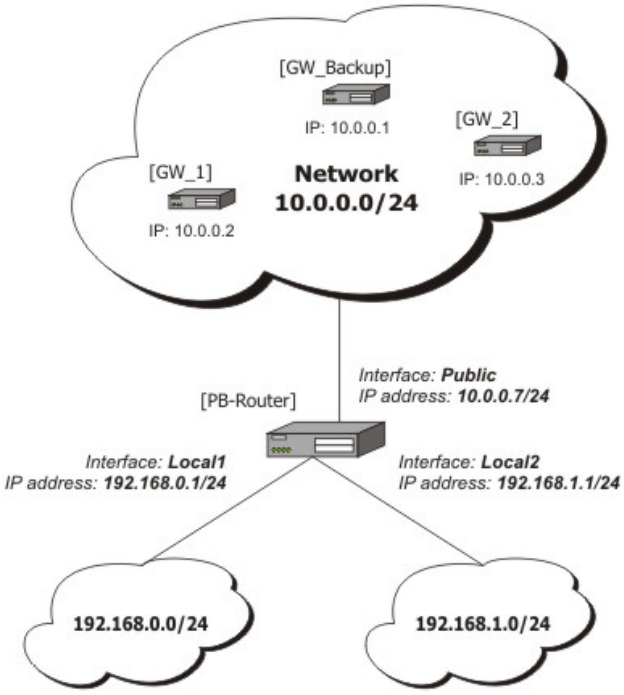
```
[admin@ECMP-Router] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS          NETWORK        BROADCAST      INTERFACE
 0   192.168.0.254/24   192.168.0.0    192.168.0.255   Local
 1   10.1.0.2/28       10.1.0.0       10.1.0.15      Public1
 2   10.1.1.2/28       10.1.1.0       10.1.1.15      Public2
[admin@ECMP-Router] ip address>
```

添加默认路由，一个 ISP1 和两个 ISP2，这样我们得到了 1:3 的比例:

```
[admin@ECMP-Router] ip route> add gateway=10.1.0.1,10.1.1.1,10.1.1.1
[admin@ECMP-Router] 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 10.1.0.0/28                              Public1
 1 ADC 10.1.1.0/28                              Public2
 2 ADC 192.168.0.0/24                            Local
 3 A S 0.0.0.0/0         r 10.1.0.1               Public1
                       r 10.1.1.1              Public2
                       r 10.1.1.1              Public2
[admin@ECMP-Router] ip route>
```

Policy-Based 策略路由

这个事例将介绍如何定义数据包的路由，策略设置如下：从 **192.168.0.0/24** 的数据包通过网关 10.0.0.2，从 **192.168.1.0/24** 的数据包通过网关 10.0.0.3，如果 GW_1 没有应答 ping 的检测，192.168.0.0/24 使用 GW_Backup ，如果 GW_2 没有应答 ping 的检测，同样使用 GW_Backup 替换 192.168.1.0/24GW_2 网关。



IP 地址配置：

```
[admin@PB-Router] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS           NETWORK        BROADCAST      INTERFACE
 0   192.168.0.1/24     192.168.0.0     192.168.0.255   Local1
 1   192.168.1.1/24     192.168.1.0     192.168.1.255   Local2
 2   10.0.0.7/24       10.0.0.0       10.0.0.255     Public
[admin@PB-Router] ip address>
```

1. 标记从 192.168.0.0/24 网段的数据包为 **new-routing-mark=net1**，和从 192.168.1.0/24 网段的数据包为 **new-routing-mark=net2**：

```
[admin@PB-Router] ip firewall mangle> add src-address=192.168.0.0/24 \
\... action=mark-routing new-routing-mark=net1 chain=prerouting
[admin@PB-Router] ip firewall mangle> add src-address=192.168.1.0/24 \
\... action=mark-routing new-routing-mark=net2 chain=prerouting
[admin@PB-Router] ip firewall mangle> print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=prerouting src-address=192.168.0.0/24 action=mark-routing
     new-routing-mark=net1

 1   chain=prerouting src-address=192.168.1.0/24 action=mark-routing
     new-routing-mark=net2
[admin@PB-Router] ip firewall mangle>
```

2. 从 192.168.0.0/24 网段的数据包指给网关 GW_1 (10.0.0.2)，数据包从网络 192.168.1.0/24 指给网关 GW_2 (10.0.0.3)，并使用相应的数据包标记。如果 GW_1 或 GW_2 连接失败 fails (没有回应 ping)，路由会把各自的数据包给 GW_Backup (10.0.0.1)：

```
[admin@PB-Router] ip route> add gateway=10.0.0.2 routing-mark=net1 \
\... check-gateway=ping
[admin@PB-Router] ip route> add gateway=10.0.0.3 routing-mark=net2 \
\... check-gateway=ping
[admin@PB-Router] ip route> add gateway=10.0.0.1
[admin@PB-Router] 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 10.0.0.0/24       10.0.0.7                             Public
 1 ADC 192.168.0.0/24    192.168.0.1                           Local1
 2 ADC 192.168.1.0/24    192.168.1.1                           Local2
 3 A S 0.0.0.0/0                      r 10.0.0.2              Public
 4 A S 0.0.0.0/0                      r 10.0.0.3              Public
 5 A S 0.0.0.0/0                      r 10.0.0.1              Public
[admin@PB-Router] ip route>
```

网关掉线

大多简单的方法是通过使用 netwatch 做监测。这里我们使用每间隔 5 秒钟 ping 一次路由器的默认网关（2.2.2.2），如果没有回应，我们将选择备用网关（3.3.3.1）：

```
/system script add name=down source={/ip route \
{... set [/ip route find dst-address=0.0.0.0] gateway 3.3.3.1}
/system script add name=up source={/ip route \
{... set [/ip route find dst-address=0.0.0.0] gateway 2.2.2.1}
/tool netwatch add host=2.2.2.2 interval=5s up-script=up down-script=down
```

负载均衡处理

现在有两条 Internet 线路接入，我们需要同时使用两条线路并作负载均衡。因此我们需要在路由器的默认路由上添加两个默认网关：

```
/ip route add gateway=1.1.1.1,2.2.2.1
```

下面是通过脚本合理的根据负载均衡的使用情况调整线路，保证在某条线路出现故障的时候其他线路能正常工作：

```
/system script add name=fo source={
  :local R1
  :local R2
  :if ([/tool netwatch get R1 status]=up) do={:set R1 1.1.1.1}
  :if ([/tool netwatch get R2 status]=up) do={:set R2 2.2.2.1}
```

```
   /ip route set [/ip route find dst-address=0.0.0.0/0] \
  gateway=($R1 . , . $R2)
  }
/tool netwatch add comment=R1 host=1.1.1.1 interval=5s up-script=fo \
  down-script=fo
/tool netwatch add comment=R2 host=2.2.2.1 interval=5s up-script=fo \
  down-script=fo
```

通过修改脚本使其能使用三个或多个网关，例如：如我们第三个网关的地址是 3.3.3.1，脚本设置为：

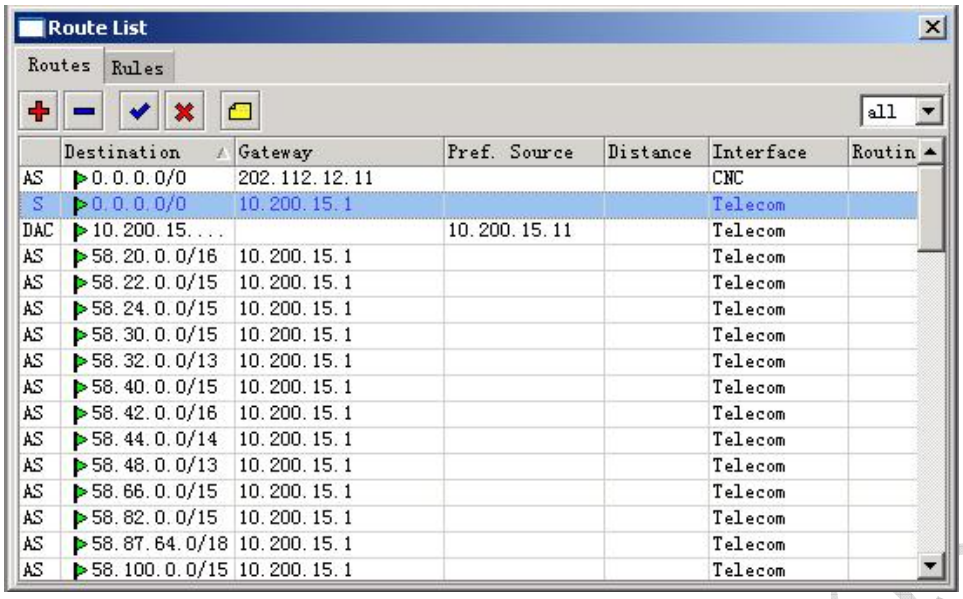
```
/system script add name=fo source={
  :local R1
  :local R2
  :local R3
  :if ([/tool netwatch get R1 status]=up) do={:set R1 1.1.1.1}
  :if ([/tool netwatch get R2 status]=up) do={:set R2 2.2.2.1}
  :if ([/tool netwatch get R3 status]=up) do={:set R3 3.3.3.1}
   /ip route set [/ip route find dst-address=0.0.0.0/0] \
  gateway=($R1 . , . $R2 . , . $R3)
  }
/tool netwatch add comment=R1 host=1.1.1.1 interval=5s up-script=fo \
  down-script=fo
/tool netwatch add comment=R2 host=2.2.2.1 interval=5s up-script=fo \
  down-script=fo
/tool netwatch add comment=R3 host=3.3.3.1 interval=5s up-script=fo \
  down-script=fo
```

如何实现其中一条线路掉线后，自动切换到另一条线路

RouterOS 2.9 中路由规则增加的两点功能：

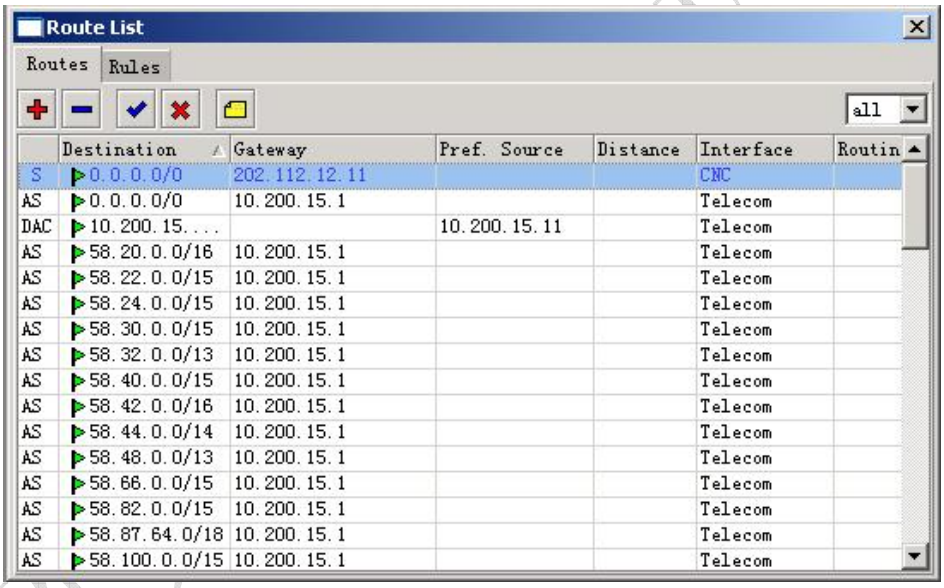
1、在 RouterOS 2.9 路由规则中增加了 check-gateway 的功能，能检测到网关的线路状态，如果网关无法探测到，便认为网关无法连接，会自动禁止访问网关的数据通过，check-gateway 功能的探测时间为 10s 一个周期。

2、在 RouterOS 2.9 中具备了对缺省网关的判断，在 RouterOS 2.9 的任何一个路由表中只能存在一个缺省网关，即到任何目标地址为 0.0.0.0/0，没有做路由标记（routing-mark）的规则，如果存在另一个缺省网关则认为是错误，路由将不予以执行。如下图：



	Destination	Gateway	Pref. Source	Distance	Interface	Routin
AS	0.0.0.0/0	202.112.12.11			CNC	
S	0.0.0.0/0	10.200.15.1			Telecom	
DAC	10.200.15....		10.200.15.11		Telecom	
AS	58.20.0.0/16	10.200.15.1			Telecom	
AS	58.22.0.0/15	10.200.15.1			Telecom	
AS	58.24.0.0/15	10.200.15.1			Telecom	
AS	58.30.0.0/15	10.200.15.1			Telecom	
AS	58.32.0.0/13	10.200.15.1			Telecom	
AS	58.40.0.0/15	10.200.15.1			Telecom	
AS	58.42.0.0/16	10.200.15.1			Telecom	
AS	58.44.0.0/14	10.200.15.1			Telecom	
AS	58.48.0.0/13	10.200.15.1			Telecom	
AS	58.66.0.0/15	10.200.15.1			Telecom	
AS	58.82.0.0/15	10.200.15.1			Telecom	
AS	58.87.64.0/18	10.200.15.1			Telecom	
AS	58.100.0.0/15	10.200.15.1			Telecom	

从上图我们可以看到，所有访问电信的 IP 段从 10.200.15.1 出去，其他的数据走网通的缺省网关出去，在我们可以这些网关的前缀都为“AS”，即确定的静态路由，而在第二排可以看到蓝色一行，他也是一个缺省网关，但因为一个路由表中只能存在一个缺省网关，所有前缀为“S”即静态但不确定的网关，被认为位非法的。如果当 202.112.12.12.11 网关断线，则 10.200.15.1 会自动启用，变为缺省路由，实现现在的切换，如下：



	Destination	Gateway	Pref. Source	Distance	Interface	Routin
S	0.0.0.0/0	202.112.12.11			CNC	
AS	0.0.0.0/0	10.200.15.1			Telecom	
DAC	10.200.15....		10.200.15.11		Telecom	
AS	58.20.0.0/16	10.200.15.1			Telecom	
AS	58.22.0.0/15	10.200.15.1			Telecom	
AS	58.24.0.0/15	10.200.15.1			Telecom	
AS	58.30.0.0/15	10.200.15.1			Telecom	
AS	58.32.0.0/13	10.200.15.1			Telecom	
AS	58.40.0.0/15	10.200.15.1			Telecom	
AS	58.42.0.0/16	10.200.15.1			Telecom	
AS	58.44.0.0/14	10.200.15.1			Telecom	
AS	58.48.0.0/13	10.200.15.1			Telecom	
AS	58.66.0.0/15	10.200.15.1			Telecom	
AS	58.82.0.0/15	10.200.15.1			Telecom	
AS	58.87.64.0/18	10.200.15.1			Telecom	
AS	58.100.0.0/15	10.200.15.1			Telecom	

当 202.112.12.11 断线后，check-gateway 在 10s 一个周期后探测到，并将 10.200.15.11 设置为缺省路由，如果 202.112.12.11 正常后，系统也将会将 202.112.12.11 设置为缺省路由，因为他是先于 10.200.15.1 添加入路由表中。

光纤和 ADSL 静态路由的实例

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：

添加脚本方式请，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，即可完成！

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
.....
```

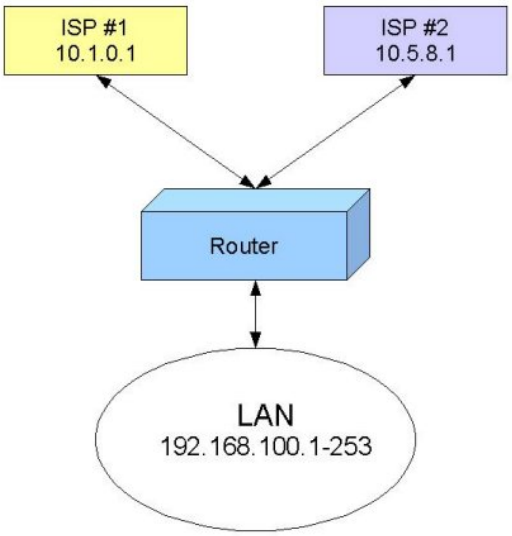
为保证一条线路断线时，到其他目标地址能正常连接，在 **/tool netwatch** 中设置主机网关监控（具体设置参考 Network 监控），并配置脚本编译。

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}
```

双线应用案例

这是一个典型的通过一个路由器并使用两条 ISP 线路接入的环境（比如都是两条电线的 ADSL 或者 LAN 接入）：



当然，你可以选择负载均衡！这里有多种方法可以选择，只是根据你的环境，选择最适合你解决方案。

基于用户端 IP 地址的策略路由

如果你有很多的主机地址，你可以通过 IP 地址将他们分组。这时，指定源 IP 地址，发送的传输通过 ISP1 或者 ISP2 的网关出去。 让我们假设终端电脑的网络地址段为 192.168.100.0/24，IP 分配如下：

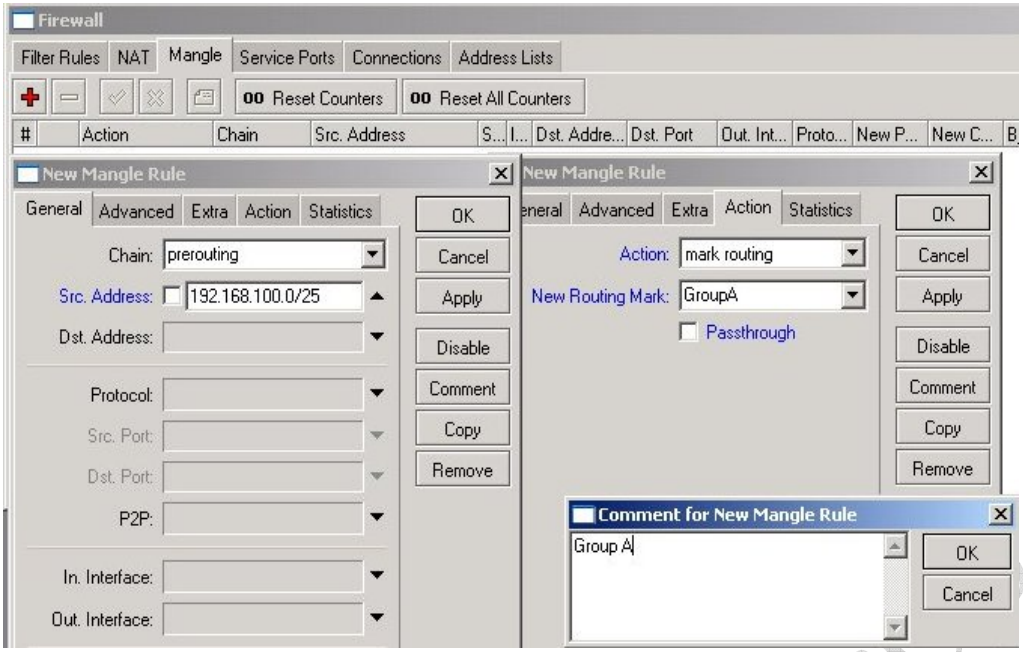
- 192.168.100.1-127 分配到 A 组
- 192.168.100.128-253 分配到 B 组
- 192.168.100.254 路由器本地 IP 地址（即内网的网关）

现在，我们通过子网划分的方式，将终端电脑进行分组：

- A 组为 192.168.100.0/25，地址范围：192.168.100.0-127
- B 组为 192.168.100.128/25，地址范围：192.168.100.128-255

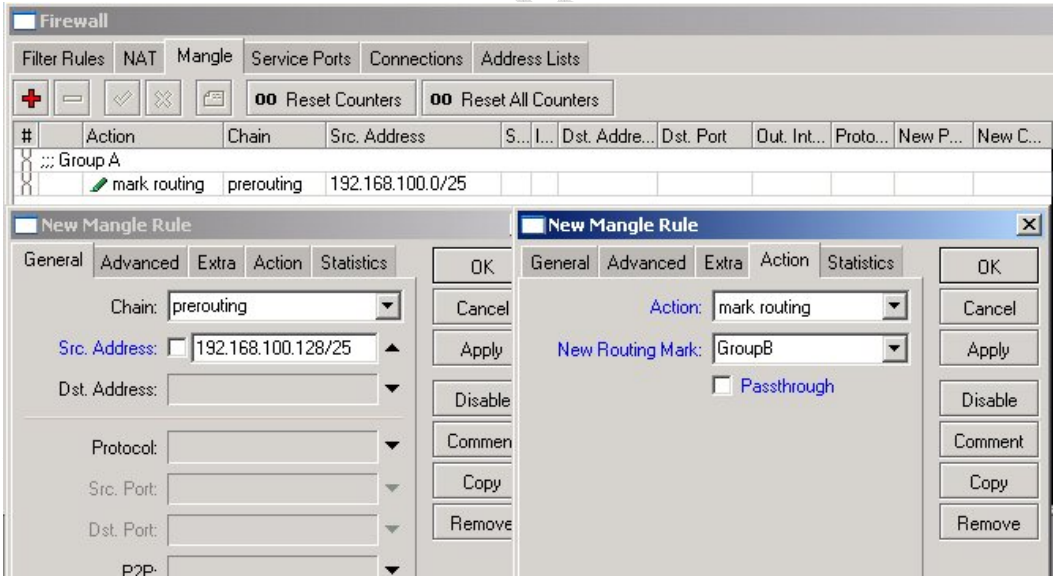
如果你不能理解，请你查阅 TCP/IP 的相关教材或通过网上查找相关的子网划分资料！我们需要添加两个 ip firewall mangle 的规则，标记来至 A 组和 B 组终端电脑的数据包。

```
定义 A 组：
链表为 chain=prerouting，源地址：src-address=192.168.100.0/25
操作为 Action=mark routing 并定义新的路由标记 GroupA.
```



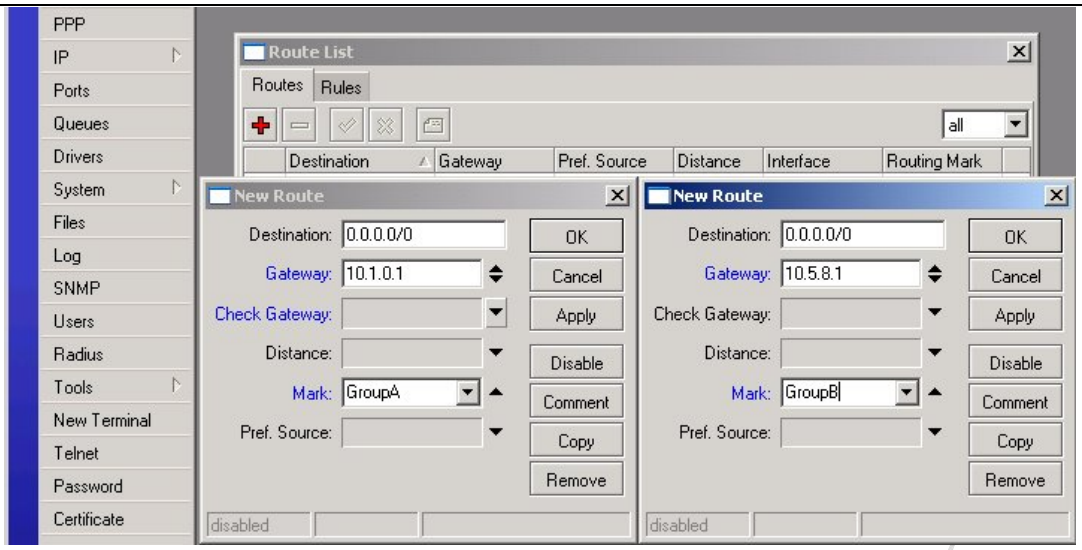
最好做一个注释，以便以后便于你自己或者别人查看和处理。

定义 B 组：
链表为 chain=prerouting，源地址：src-address=192.168.100.128/25
操作为 Action=mark routing 并定义新的路由标记 GroupB



所有来至终端电脑的 IP 传输都通过路由标记为 GroupA 或者 GroupB。这样我们可以标记到路由表中（routing table）。

下面，我们需要定义两个默认路给相应的路由标记和网关：



到这里，如果你没有对路由器做 NAT 的伪装，请在**/ip firewall nat** 里添加 src- Address=**192.168.100.0/24** action=**masquerade,** 在终端电脑上测试一下跟踪路由是否正确定义两个分组的默认路由：

A 组测试如下情况：

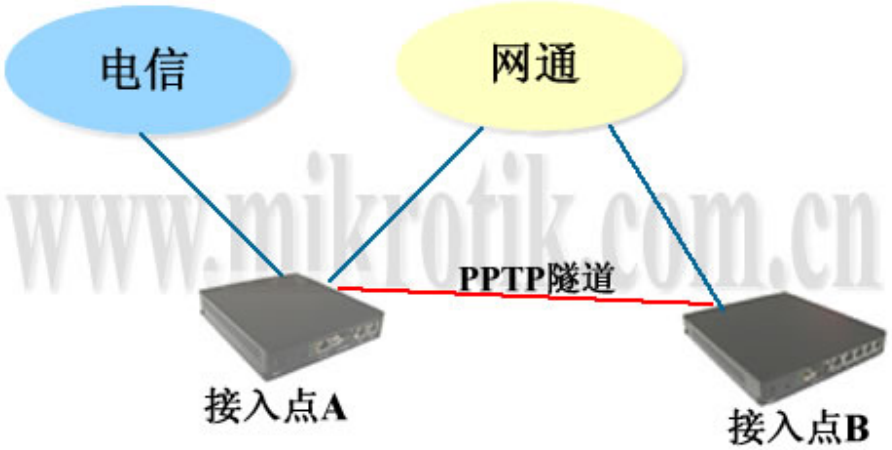
```
C:\>tracert -d 8.8.8.8
Tracing route to 8.8.8.8 over a maximum of 30 hops

  1    2 ms    2 ms    2 ms  192.168.100.254
  2   10 ms    4 ms    3 ms  10.1.0.1
  ...
```

B 组测试如下情况：

```
C:\>tracert -d 8.8.8.8
Tracing route to 8.8.8.8 over a maximum of 30 hops

  1    2 ms    2 ms    2 ms  192.168.100.254
  2   10 ms    4 ms    3 ms  10.5.8.1
  ...
```

PPTP 借线操作

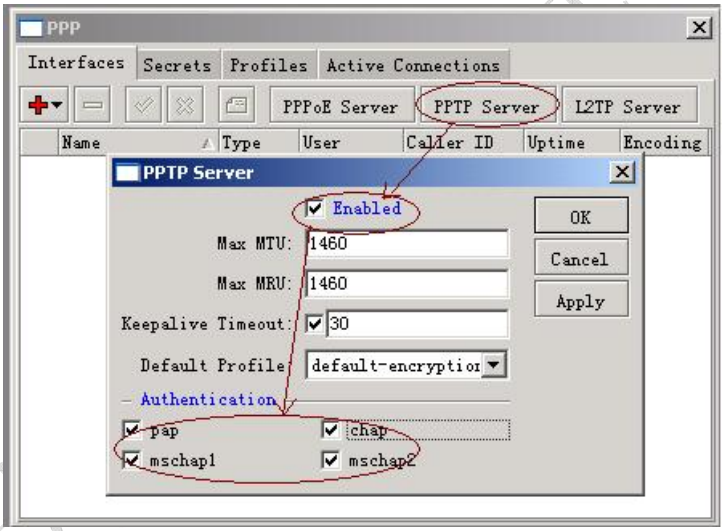
假设一个接入点 A 有电信和网通两条线路，并做了以网通为主，电信为静态路由策略设置。而另一个接入点 B 接入了网通的线路，并且想通过 PPTP 隧道的方式借用接入点 A 的电信线路，现在看下面的图例



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。

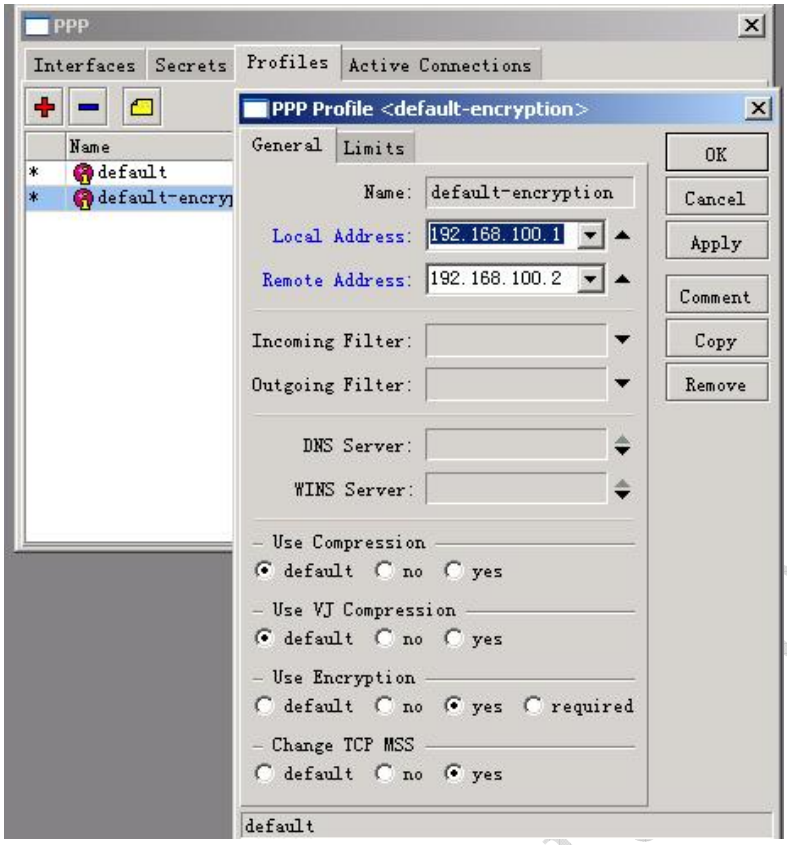
配置 PPPTP-Server

在接入点 A 启用 PPTP-Server，并设置密码传输的加密类型：



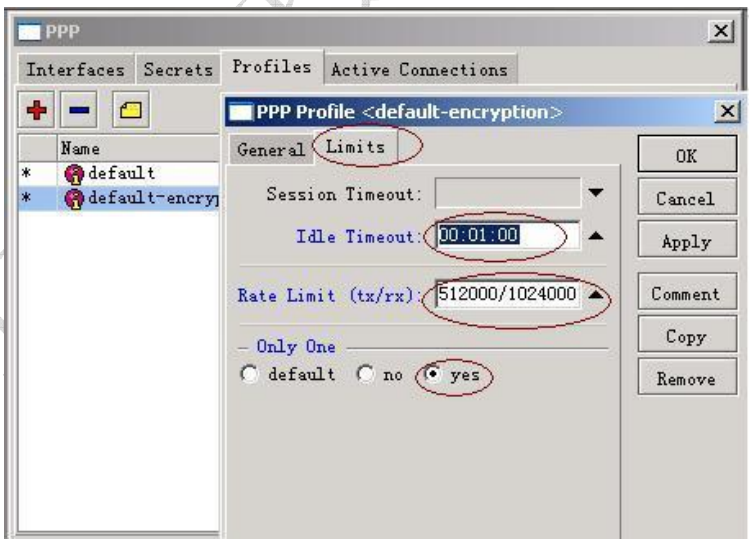
在这里 Default-Profile 我们采用 default-encryption，同样你也可以在 PPTP-Server 的 profiles 中创建自己的规则。Keepalive-Timeout 是 PPTP-Server 主动使用 ICMP 协议探测客户端是否在线，如果客户端使用了防火墙或禁止 ICMP 探测，那无法探测到客户端，Server 就会主动断开该客户端的连接，这个设置需要用户自己根据网络情况判断。

设置 Profile 定义客户和主机的访问地址：



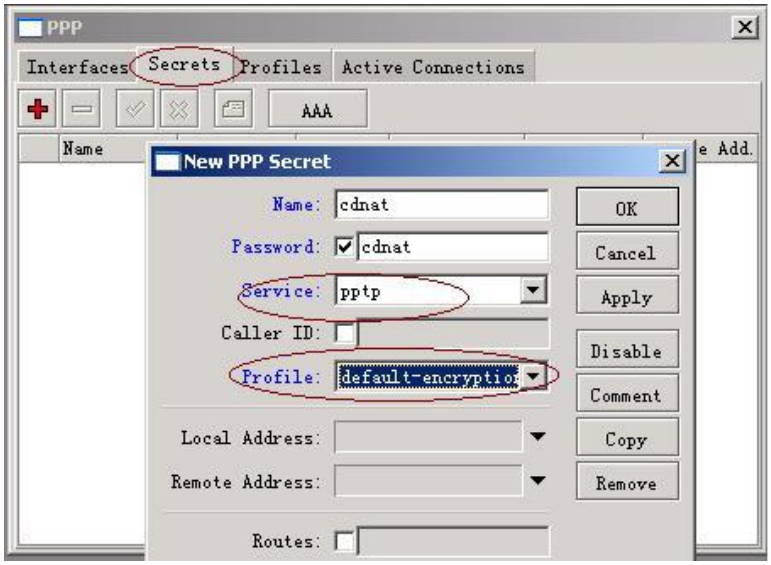
在这里我们给 PPTP-Server 分配的 IP 地址为 192.168.100.1(local-address)，给客户端分配的地址为 192.168.100.2(remote-address)。分配 IP 地址也可以通过账号设置 Secrets 进行，在这里我们只有一个客户端所有可以直接通过 profile 中的规则设置，如果有多个客户端也可以通过/ip pool 中的地址池做 DHCP 的分配。

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。

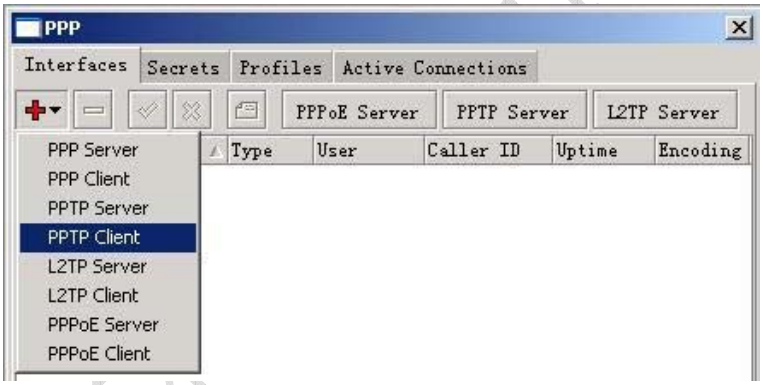
设置客户端的账号密码：



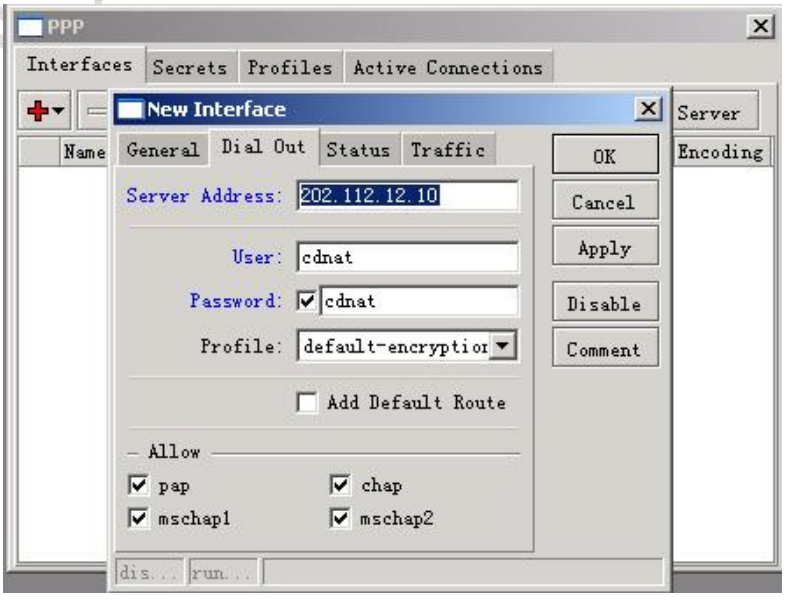
进入 secret 设置账号和密码以及相关信息，设置好 name 和 password 后，选择 service 服务类型为 pptp，profile 规则为 default-encryption。这样 PPTP-Server 就已经设置完成。

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：

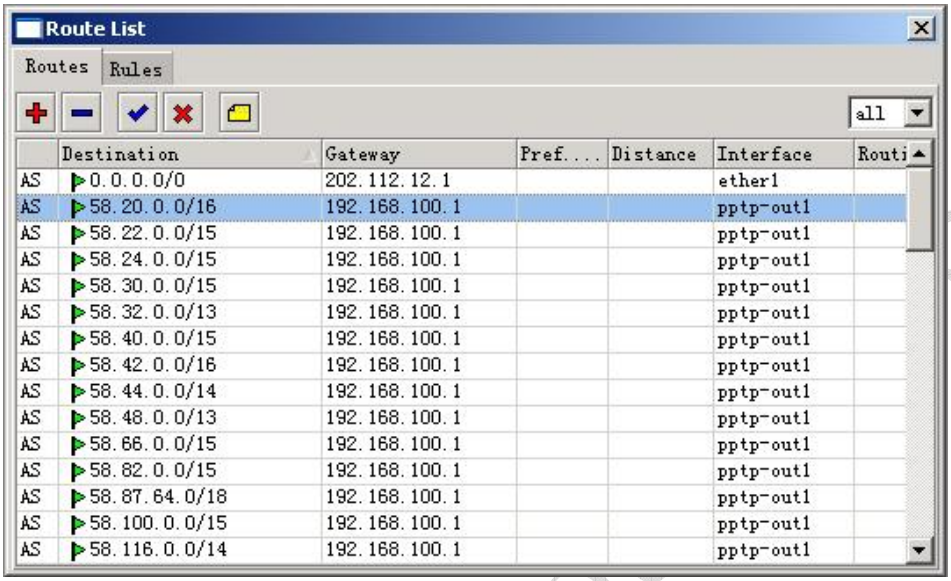


设置账号和密码分别为 cdnat，设置完成后，便可以与接入点 A 的 PPTP-Server 连接。

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。

RouterOS v3 NTH 的负载均衡

在 v3.0 中 NTH 工具做了一点修改。仅只有两个参数“every”和“packet”。

在 RouterOS v3 中如何工作

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被增加 1，如果计数器匹配值“every”与数据包匹配且计数器将设置为 0。

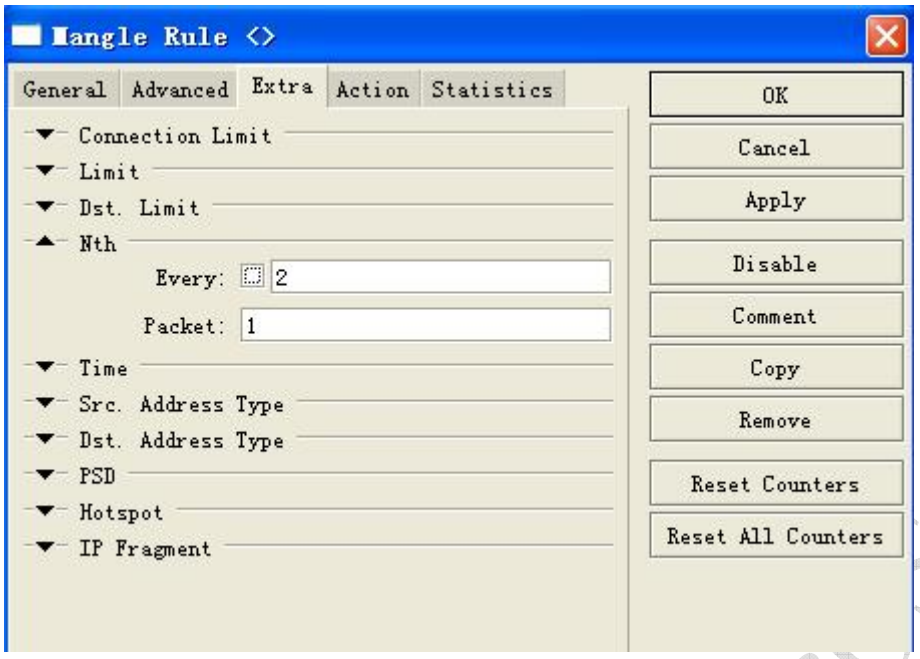
如果 passthrough 没有设置将标记如下：

第一条规则 nth=2,1 规则将对比每两个数据包，所有流量的 50%被该规则对比。
第二条规则设置相同，但如果 passthrough=no 将对比剩下 25%的流量，因为 3.0 捕获流量只需要一条规则，不像 2.9。

事例

现在配置 50%流量仅需要一条规则：

```
/ip firewall mangle
add action=mark-packet chain=prerouting new-packet-mark=AAA nth=2,1 passthrough=no;
```



之后剩下的流量只需要做一个默认的标记即可。如果多余一条规则的标记，这时有两种方式：

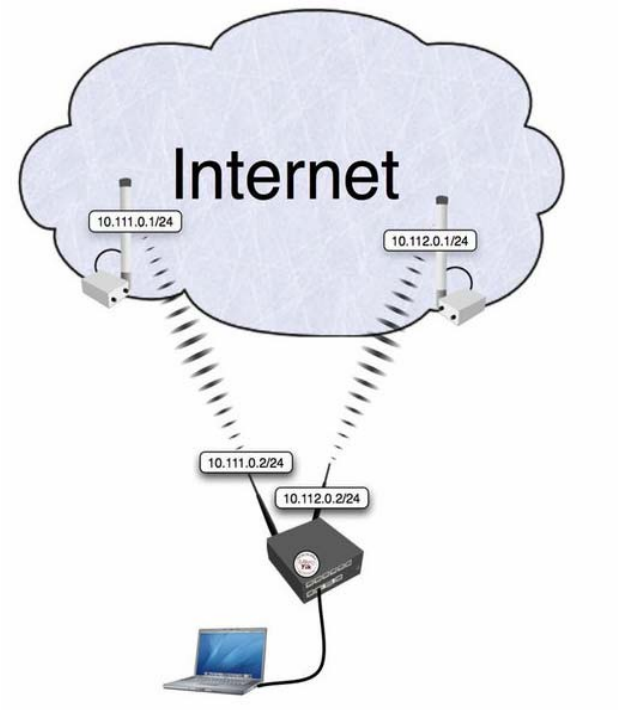
第一条规则查看所有数据包并对比所有流量的 1/3，第二条规则查看剩下 2/3 数据包的 50%，第三条规则查看和对比所有剩下的数据包（所有数据包的 1/3）

```
/ip firewall mangle
add action=mark-packet chain=prerouting new-packet-mark=AAA nth=3,1 passthrough=no;
add action=mark-packet chain=prerouting new-packet-mark=BBB nth=2,1 passthrough=no;
add action=mark-packet chain=prerouting new-packet-mark=CCC ;
```

所有规则能查看所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-packet chain=prerouting new-packet-mark=AAA nth=3,1 passthrough=yes;
add action=mark-packet chain=prerouting new-packet-mark=BBB nth=3,2 passthrough=yes;
add action=mark-packet chain=prerouting new-packet-mark=CCC nth=3,3 passthrough=yes;
```

这个例子是基于 RouterOS3.0 负载均衡的改进版本。请看下列拓扑图：



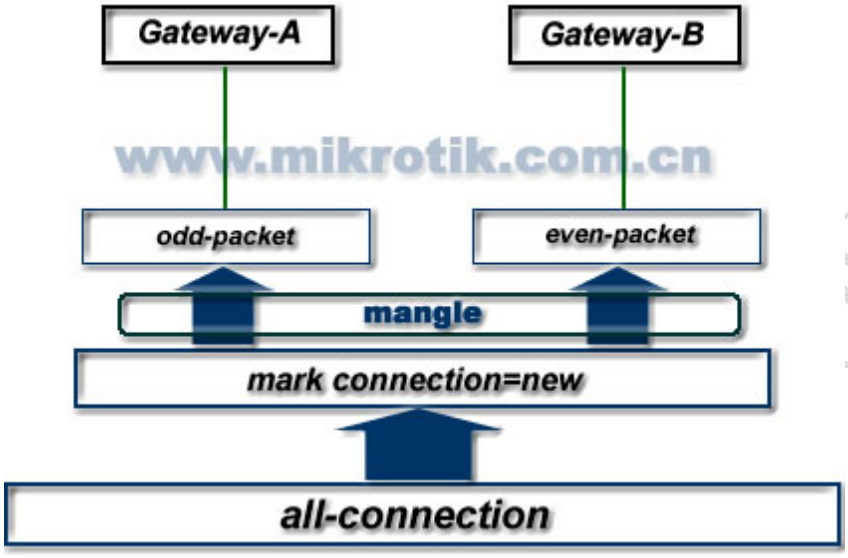
配置代码：

```
/ ip address
add address=192.168.0.1/24 network=192.168.0.0 broadcast=192.168.0.255 interface=local
add address=10.111.0.2/24 network=10.111.0.0 broadcast=10.111.0.255 interface=wlan2
add address=10.112.0.2/24 network=10.112.0.0 broadcast=10.112.0.255 interface=wlan1

/ ip firewall mangle
0    chain=prerouting action=mark-connection new-connection-mark=odd
     passthrough=no connection-state=new in-interface=local nth=2,1

 1   chain=prerouting action=mark-connection new-connection-mark=even
     passthrough=no connection-state=new in-interface=local

 2   chain=prerouting action=mark-routing new-routing-mark=odd
     passthrough=yes connection-mark=odd

 3   chain=prerouting action=mark-routing new-routing-mark=even
     passthrough=yes connection-mark=even

/ ip firewall nat
add chain=srcnat connection-mark=odd action=src-nat to-addresses=10.111.0.2 \
    to-ports=0-65535
add chain=srcnat connection-mark=even action=src-nat to-addresses=10.112.0.2 \
    to-ports=0-65535

/ ip route
add dst-address=0.0.0.0/0 gateway=10.111.0.1 scope=255 target-scope=10 routing-mark=odd
add dst-address=0.0.0.0/0 gateway=10.112.0.1 scope=255 target-scope=10 routing-mark=even
```

```
add dst-address=0.0.0.0/0 gateway=10.112.0.1 scope=255 target-scope=10
```

原理与命令讲解

这里我们来看看持续的负载均衡的基本原理：



通过上面的图，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（GatewayA 与 GatewayB）出去。这样就能保持每次连接的持续性。

首先我们通过对每段代码做分析，理解他们是怎么运行

IP 配置：

```
/ ip address
add address=192.168.0.1/24 network=192.168.0.0 broadcast=192.168.0.255 interface=Local
add address=10.111.0.2/24 network=10.111.0.0 broadcast=10.111.0.255 interface=wlan2
add address=10.112.0.2/24 network=10.112.0.0 broadcast=10.112.0.255 interface=wlan1
```

路由器的两个 WAN 口地址分别是 10.111.0.2/24 和 10.112.0.2/24，LAN 口的地址是 192.168.0.1/24，内网网卡命名为 LOCAL

Mangle 配置

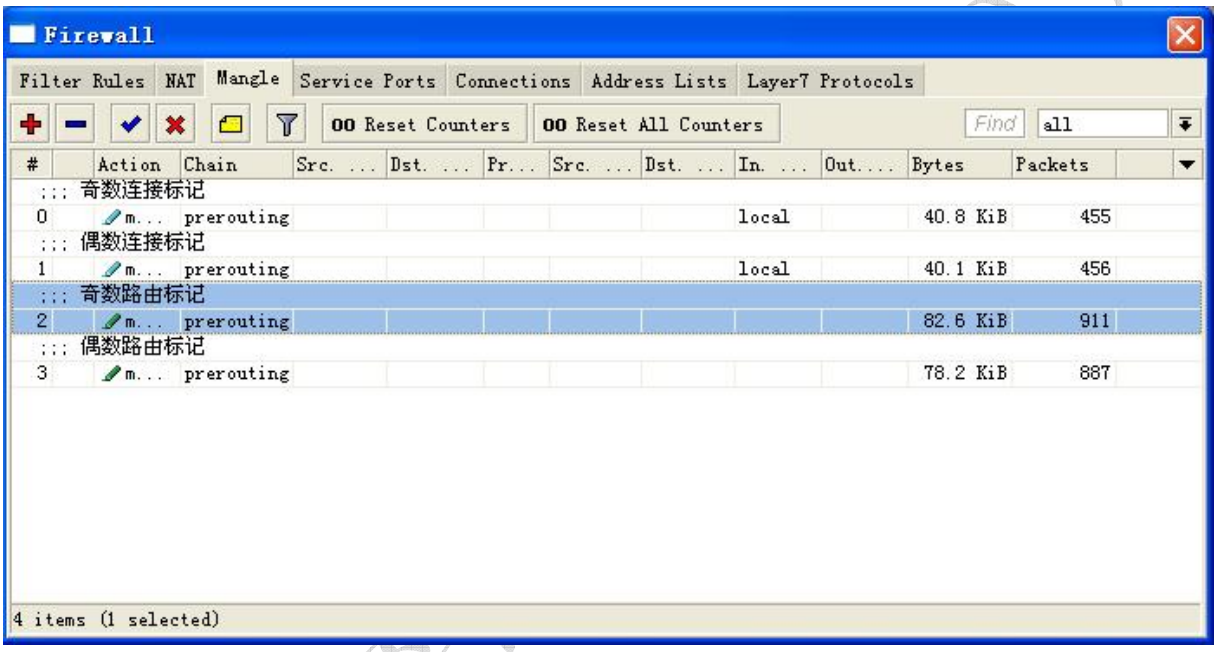
下面是通过 nth 来分配用户连接会话：

```
/ ip firewall mangle
0   chain=prerouting action=mark-connection new-connection-mark=odd
    passthrough=no connection-state=new in-interface=local nth=2,1

 1  chain=prerouting action=mark-connection new-connection-mark=even
    passthrough=no connection-state=new in-interface=local
```

首先，每隔一个包建立一个新的会话，并用“odd”做标记，因此所有属于同一会话的连续的数据包将被放到标记为 odd 的连接中，注意，因为是 v3 版本只有是双线的两组规则，所以在这里我们只需要标记一条 NTH 规则，剩下的连接则被最后一条规则获取，并标记为 even。

```
/ ip firewall mangle
2   chain=prerouting action=mark-routing new-routing-mark=odd
    passthrough=yes connection-mark=odd

 3   chain=prerouting action=mark-routing new-routing-mark=even
    passthrough=yes connection-mark=even
```

接着我们将这些标记好的连接，分别设置到相应的路由标记（new-routing-mark）中。



NAT 配置

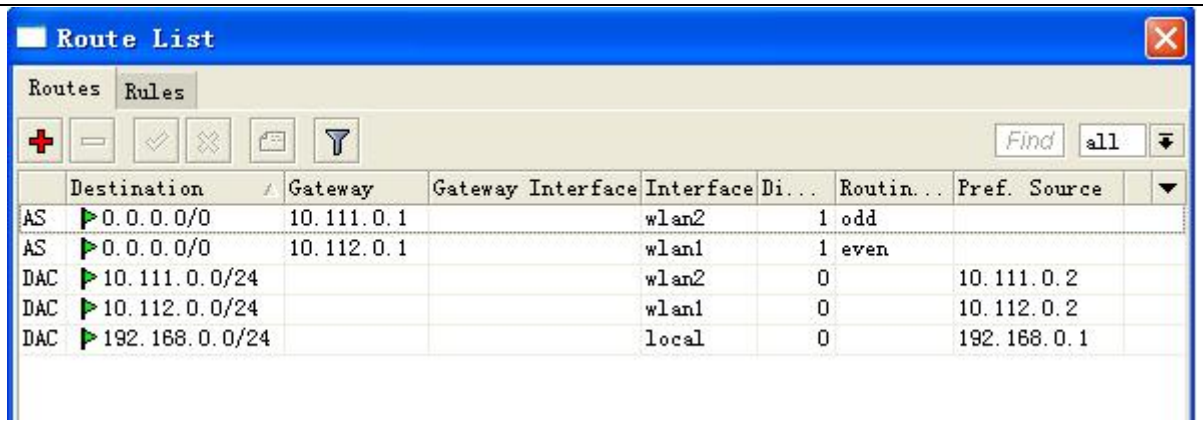
```
/ ip firewall nat
add chain=srcnat connection-mark=odd action=src-nat to-addresses=10.111.0.2 \
   to-ports=0-65535
add chain=srcnat connection-mark=even action=src-nat to-addresses=10.112.0.2 \
   to-ports=0-65535
```

被标记为 ODD 的数据 NAT 为 10.111.0.2，以 EVEN 为标记的数据 NAT 为 10.112.0.2

路由配置

```
/ ip route
add dst-address=0.0.0.0/0 gateway=10.111.0.1 scope=255 target-scope=10 routing-mark=odd
add dst-address=0.0.0.0/0 gateway=10.112.0.1 scope=255 target-scope=10 routing-mark=even
```

被标记为 ODD 的数据用 10.111.0.1 为网关，同样，被标记为 EVEN 的数据从 10.112.0.1 这个网关出去。




```
/ ip route
add dst-address=0.0.0.0/0 gateway=10.112.0.1 scope=255 target-scope=10
```

最后，没有做任何标记的数据从 10.112.0.2 这个网关出去，也是给路由器一个默然网关。

Network 监控

基本信息

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
Hardware usage: *Not significant*

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>
```

DHCP 设置

DHCP-Client 设置

操作路径: ***/ip dhcp-client***

MikroTik RouterOS DHCP-client 在一个以太网卡上启用，client 将接受一个地址、子网掩码、默认网关和两个 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的动态项目，如果 DHCP-client 被禁用或没有更新一个地址，动态路由将自动删除。

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**) – 是否接受 DHCP 服务器的 DNS 的设置(将会添加到/ip dns 中)

命令描述

renew – 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。

事例

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

```
[admin@MikroTik] ip dhcp-client> set enabled=yes interface=ether1
[admin@MikroTik] ip dhcp-client> print
           enabled: yes
         interface: ether1
         host-name: ""
         client-id: ""
   add-default-route: yes
       use-peer-dns: yes
[admin@MikroTik] ip dhcp-client>
```

DHCP-Server 基本向导设置

指令名称：***/ip dhcp-server setup***

向导内容

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>
```

图形显示（Graphing）

基本信息

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***

属性

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。

基本选项

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。

例如

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 Property Description

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 Graphing

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

例如

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

例如

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 Graphing

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

例如

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>
```

系统监测 System Watchdog

基本信息

系统监督的特征是当系统软件万一出现错误会重启.

规格

功能包需求: ***system***
等级需求: *Level1*
操作路径: ***/system watchdog***
技术与标准:
硬件使用: *Not significant*

硬件监督管理

操作路径: ***/system watchdog***

描述

当一个 IP 地址没有响应或者系统已被锁定这个菜单允许配置当前系统. 软件监督计时器是用来提供上一次的选择记录, 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 的硬件监督设备来说它能在任何情况下重启

属性描述

auto-send-supout (yes | no; default: **no**) –帮助文件是自动产生它能通过邮件发送
automatic-supout (yes | no; default: **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 (*time*; default: **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 (*text*; default: **""**) –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** 开启其功能
send-email-to (*text*; default: **""**) – 邮件地址是向帮助文件发送
send-smtp-server (*text*; default: **""**) – 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** 开启其功能
watch-address (*IP address*; default: **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; default: **no**) – 是否重启取决于在一段时间馁系统无法响应

实例

万一系统崩溃系统产生的帮助文件并且自动通过 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>
```

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。

属性

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。

注意！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:

注意 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

实例

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 (*time*; 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 (name | string; 默认: **""**) - 脚本源代码
duration (*time*; 默认: **0s**) - 测试时长
0s – 测试时间没有被限制
interval (*time*: 20ms..5s; 默认: **1s**) – 报告间隔时间（秒钟计算）

local-tx-speed (*integer*; 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 (*integer*: 40..64000) – 本地 UDP 发送最大数据包

password (*text*; 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 (*integer*; 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 (*integer*: 40..64000) – 远端 UDP 发送最大数据包

user (*name*; 默认: **""**) - 远程用户名

实例

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>
```

Torch (即时通信监听)

基本信息

规格

功能包要求: ***system***
等级要求: *Level1*
子目录要求: ***/tool***
标准与技术: none
硬件使用: *Not significant*

描述

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.

Torch 命令

操作路径: **/tool torch**

特性描述

(*name*) – 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 (*time*) - 屏幕输出暂停的立即时间
port (name | integer) – 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).

实例

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>
```

分类标记（Mangle）

基本信息

mangle 允许对 IP 数据包做特殊的标记，mangle 是通过修改一些 IP 数据包头的字段，去标记 IP 数据包的特征。

需要功能包: ***system***
需要等级: *Level1*
操作路径: ***/ip firewall mangle***
协议标准: IP

属性

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。

应用事例

下面是一些使用标记的事例：

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
```

防火墙过滤（Firewall Filte）

基本信息

Firewall 对经过和进入路由器数据包进行过滤，因此能管理数据流，为网络提供安全功能。为网络提供地址翻译功能，能阻止未经过认证的访问直接连接网络，并能过滤路由器向外发出的数据。

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
```

需要功能包: ***system***
需要等级: *Level1 (P2P filters limited to 1)* , *Level3*
操作路径: ***/ip firewall filter***
技术标准: IP, RFC2113

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) byte
  - 数据包内容 packet content
  - rate at which packets arrive and sequence numbers
  - 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，一个策略是告诉路由器如何处理一个 IP 数据包决定，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。

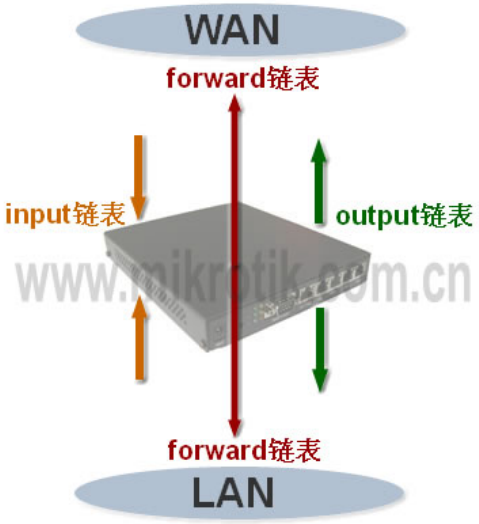
过滤功能有三个默认的数据链（chains）：**input**, **forward** 和 **output** 他们分别负责从哪里进入路由器的、通过路由器转发的与从路由器发出的数据。用户也可用自定义添加链，当然这些链没有默认的传输配置，需要在三条默认的链中对 **action=jump** 策略中相关的 **jump-target** 进行配置。

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：

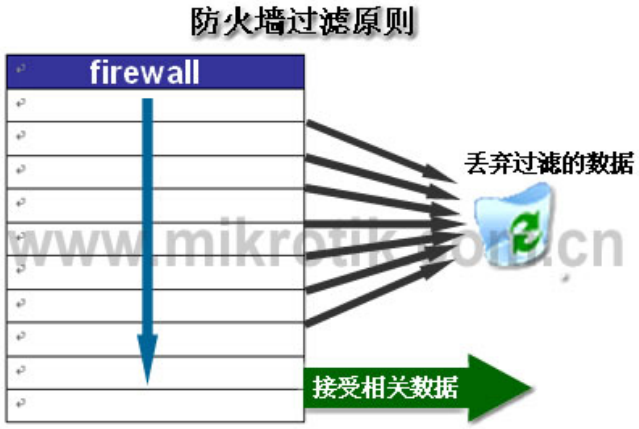
- **input** – 用于处理进入路由器的数据包，即数据包目标 IP 地址是到达路由器一个接口的 IP 地址，经过路由器的数据包不会在 input-chains 处理。
- **forward** – 用于处理通过路由器的数据包
- **output** – 用于处理源于路由器并从其中一个接口出去的数据包。

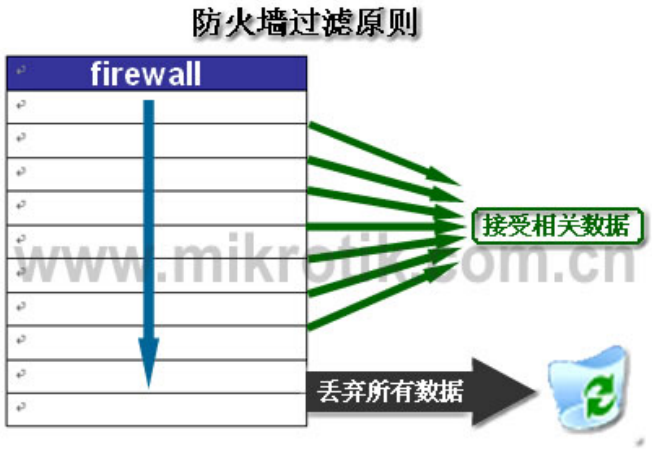
他们具体的区别如下：



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：

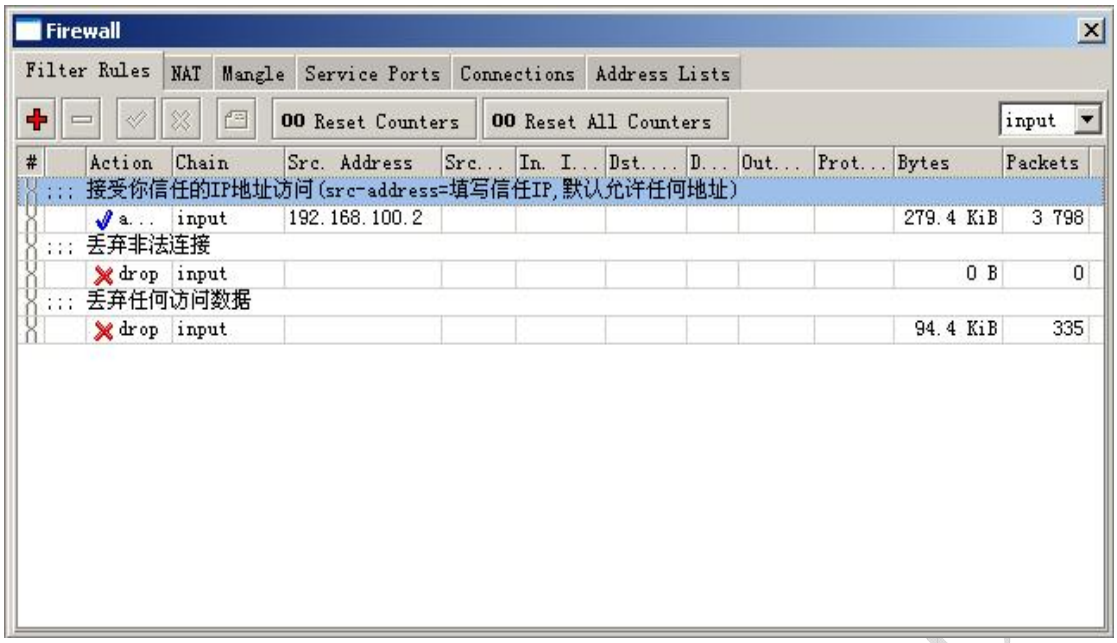




现在我来看事例中的防火墙规则：

防火墙规则

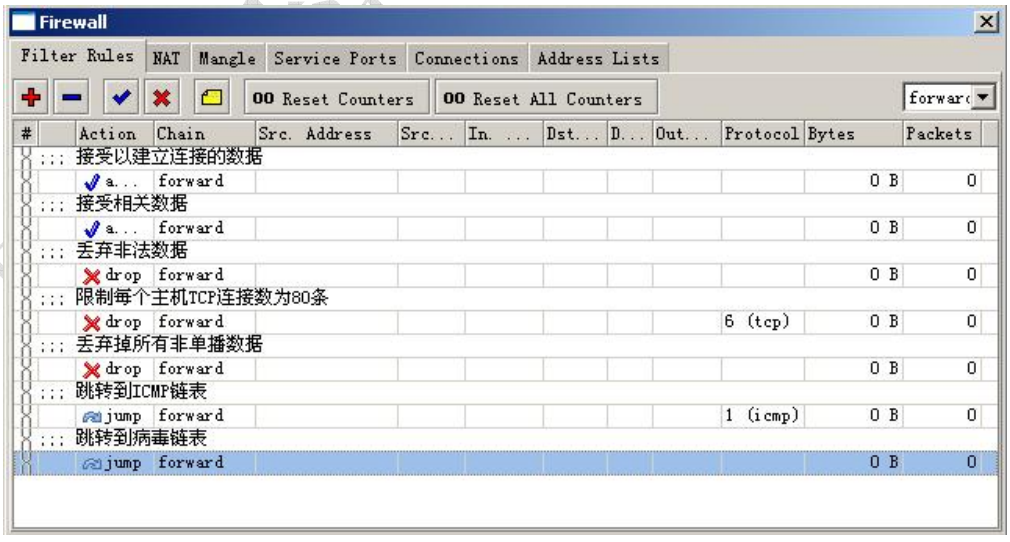
我先从 input 链表开始，这里是对所有访问路由的数据进行过滤和处理：



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

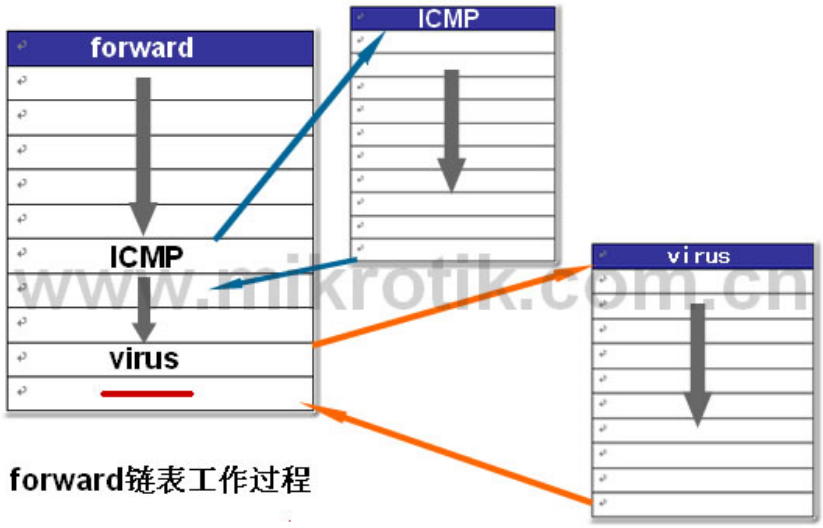


forward 链表，一共有 7 条规则，包括两个跳转到自定义链表 ICMP 和 virus 链表：

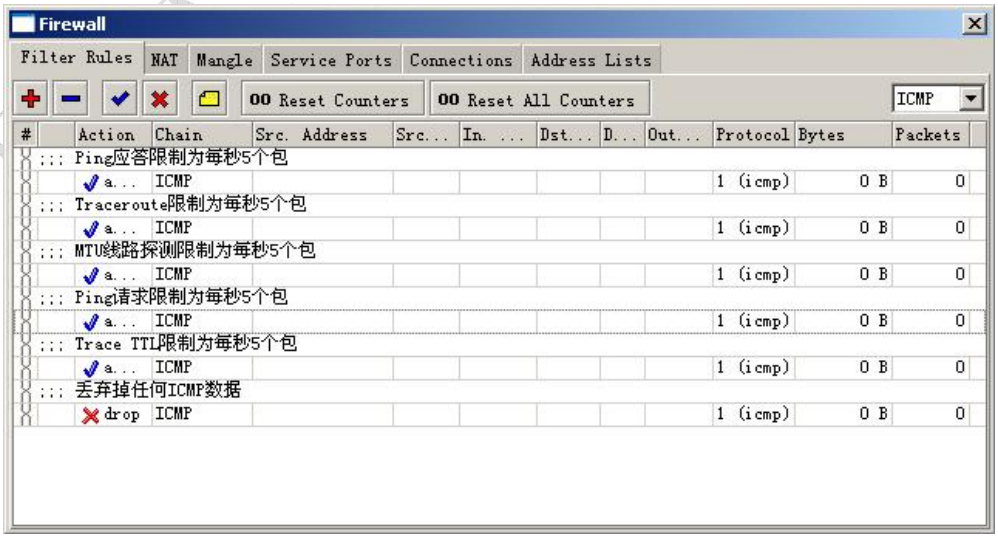
```
0  ;;;  接受已建立连接的数据
     chain=forward connection-state=established action=accept
 1  ;;;  接受相关数据
     chain=forward connection-state=related action=accept
```

```
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：

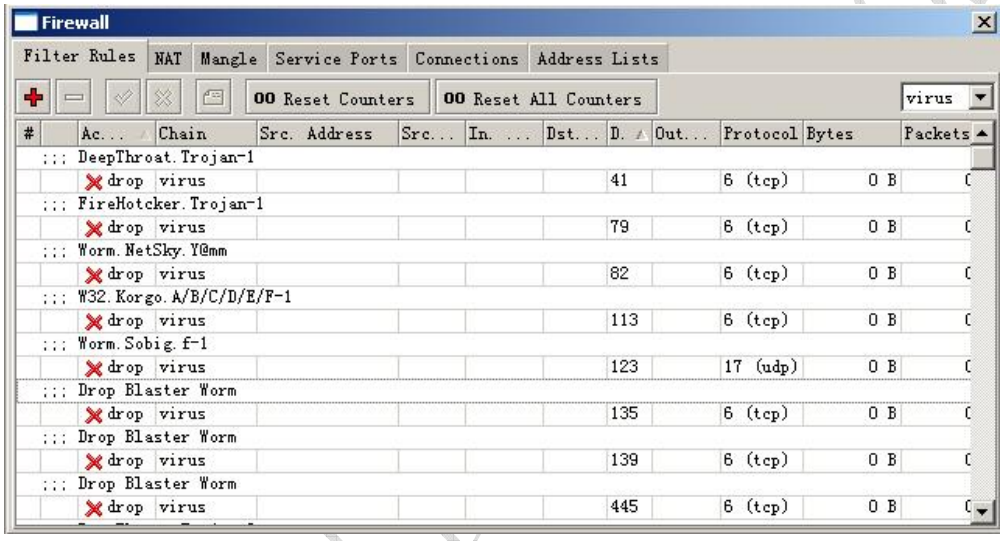


ICMP 链表操作过程：

```
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
```

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：



保护你的 RouterOS 路由器

保护你的路由器，你不应仅改变 admin 的密码，还要对数据包的过滤。在所数据包到达目标是路由器都会先被防火墙的 input 数据链处理。注意：input 数据链不会影响中转通过路由器的数据包。

```
/ ip firewall filter
add chain=input connection-state=invalid action=drop
       comment="Drop Invalid connections"
add chain=input connection-state=established action=accept \
       comment="Allow Established connections"
add chain=input protocol=udp action=accept \
       comment="Allow UDP"
add chain=input protocol=icmp action=accept \
       comment="Allow ICMP"
add chain=input src-address=192.168.0.0/24 action=accept \
       comment="Allow access to router from known network"
add chain=input action=drop comment="Drop anything else"
```

过滤应用

保护你的 RouterOS 路由器

保护你的路由器，你不应仅改变 admin 的密码，还要对数据包的过滤。在所数据包到达目标是路由器都会先被防火墙的 input 数据链处理。注意：input 数据链不会影响中转通过路由器的数据包。

```
/ ip firewall filter
add chain=input connection-state=invalid action=drop
       comment="Drop Invalid connections"
add chain=input connection-state=established action=accept \
       comment="Allow Established connections"
add chain=input protocol=udp action=accept \
       comment="Allow UDP"
add chain=input protocol=icmp action=accept \
       comment="Allow ICMP"
add chain=input src-address=192.168.0.0/24 action=accept \
       comment="Allow access to router from known network"
add chain=input action=drop comment="Drop anything else"
```

保护客户的网络

保护客户端的网络，我们需要检查所有通过路由器传输的数据和有害的数据块，外面将建立 icmp、tcp、udp 数据链（chains）的检测，我们将丢弃掉所有有害的数据包：

```
/ip firewall filter
add chain=forward protocol=tcp connection-state=invalid \
       action=drop comment="drop invalid connections"
add chain=forward connection-state=established action=accept \
       comment="allow already established connections"
add chain=forward connection-state=related action=accept \
       comment="allow related connections"
```

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"
```

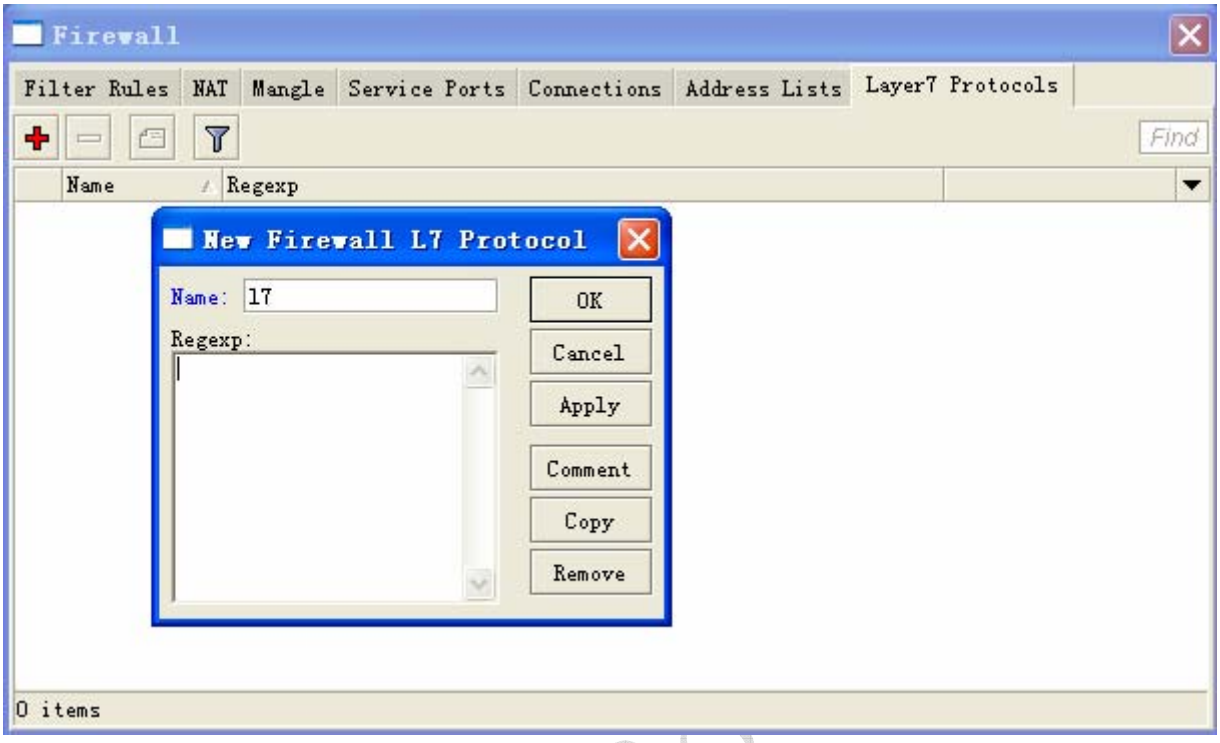
在 icmp-chain 允许相应需要的 icmp 连接：

```
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"
```

MikroTik RouterOS 3.0 7 层协议过滤脚本得应用

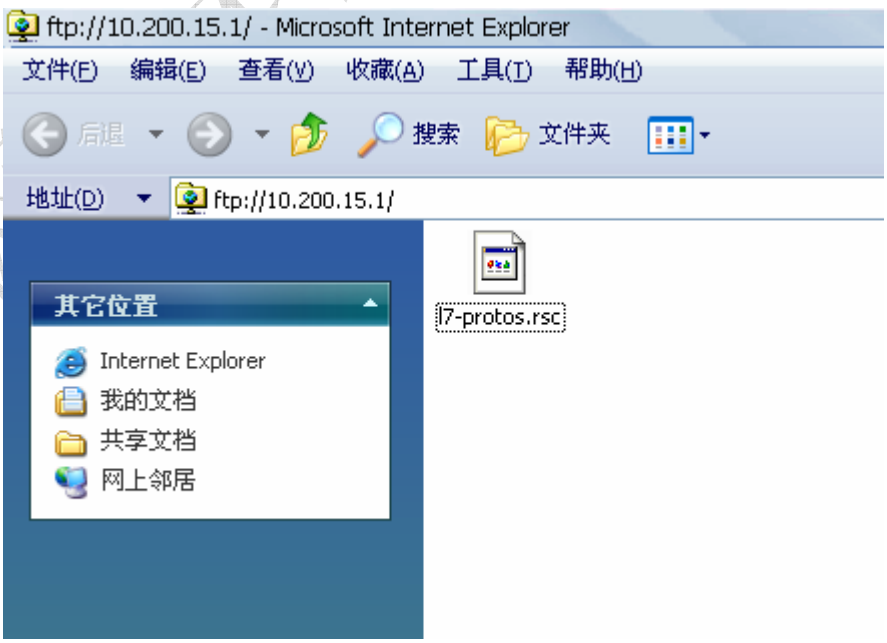
RouterOS V3.0 在防火墙中增加了一个新得功能——7 层协议过滤。针对一些应用程序如 skype、QQ、MSN、魔兽世界.....网络程序做限制和过滤。下面介绍一下具体方法的使用：

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：



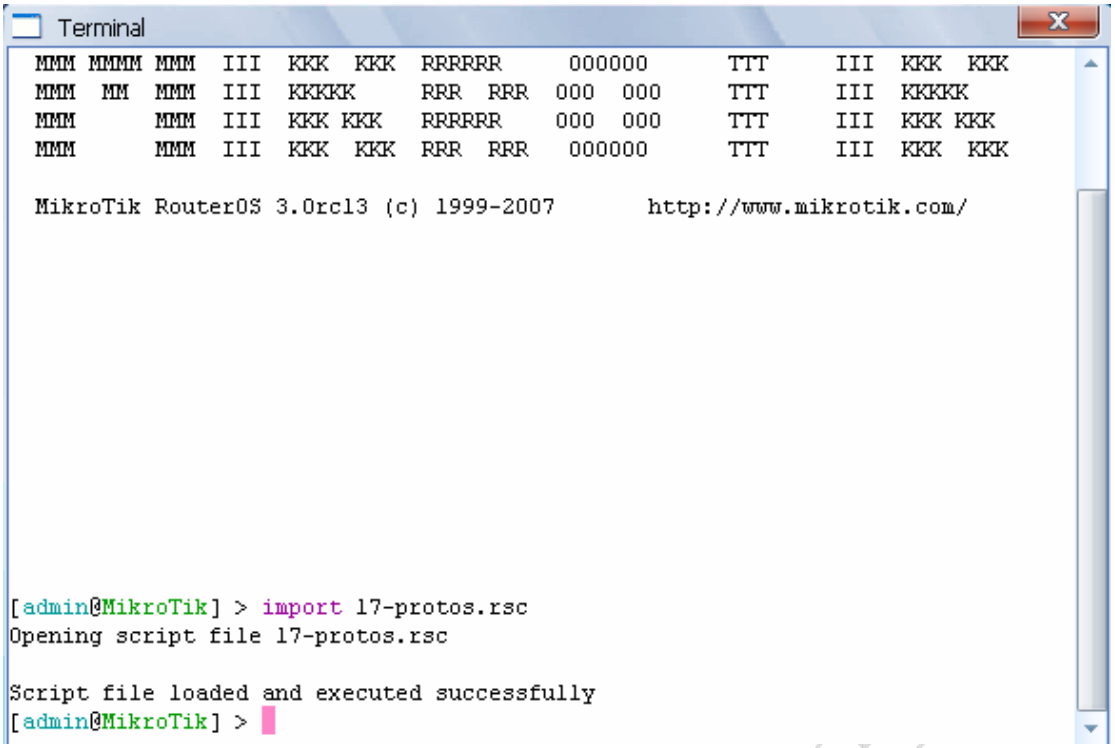
7 层协议通过 Regexp 脚本编写相应应用程序的过滤代码，Regexp 可以通过网上搜索相关资料了解。在这里我们已经提供了一些常用程序的 7 层协议脚本：

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。



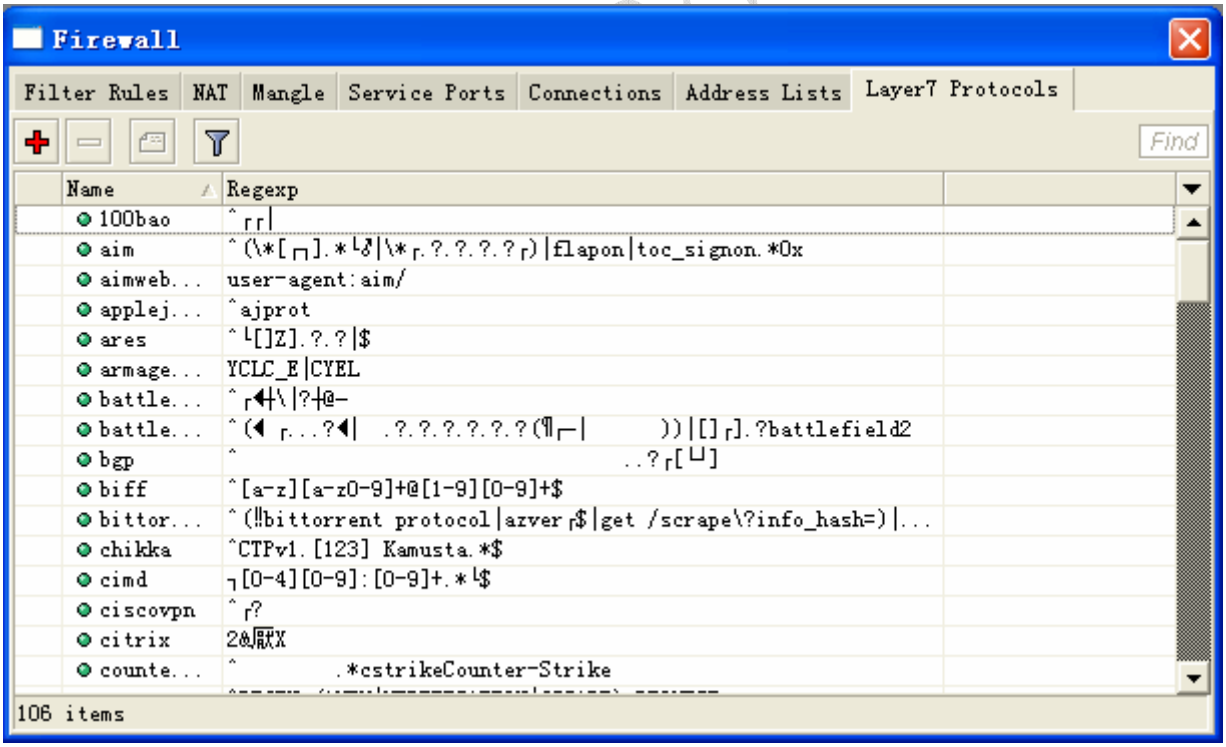
之后我们在命令行(Terminal)中导入 7 层协议脚本：

用 import 17-protos.rsc 命令来导入脚本

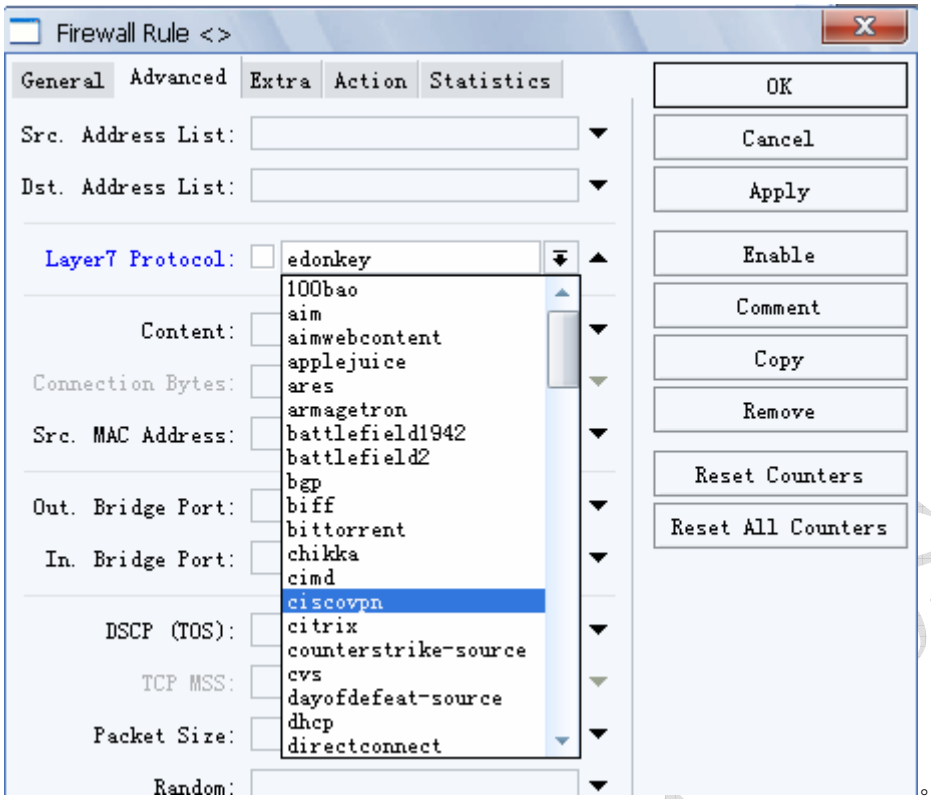


当系统提示 Script file loaded and executed successfully，说明脚本成功导入。

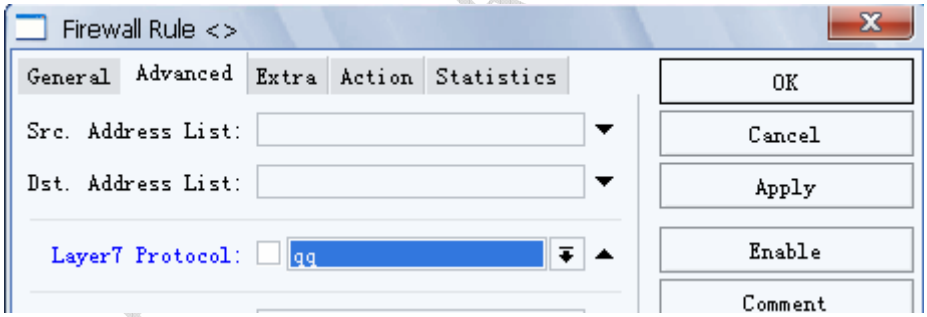
导入脚本后，我们可以在 Layer7 Protocols 中看到



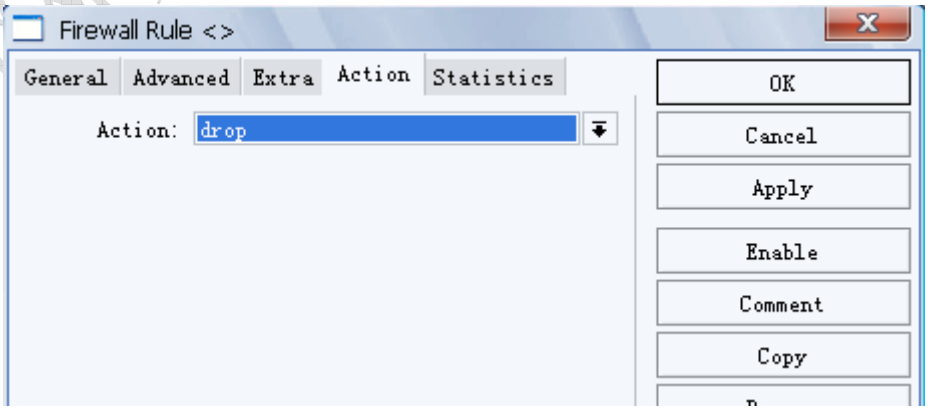
导入后，我们就可以在 ip firewall 中通过 Layer7 Protocols 参数调用，并做相应的规则处理，下面是一个在防火墙得 Filter Rules 里面调用 L7 脚本



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。



因为要禁止登陆 QQ，所以 drop 掉数据。



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。

连接追踪

操作路径: ***/ip firewall connection***

描述

连接追踪设计到维护连接状态信息的能力，例如源目的 IP 地址和端对，连接状态，协议类型和超时。完成连接追踪的防火墙以“stateful”知晓并内在地比那些只完成简单“stateless”包处理的安全的多。

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或属于一个没有在两个方向都看到的连接，**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。

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 状态列表能包含的最大数的连接是由最初路由器的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，但 128M RAM 的路由器就可以增加到 130000 以上。

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

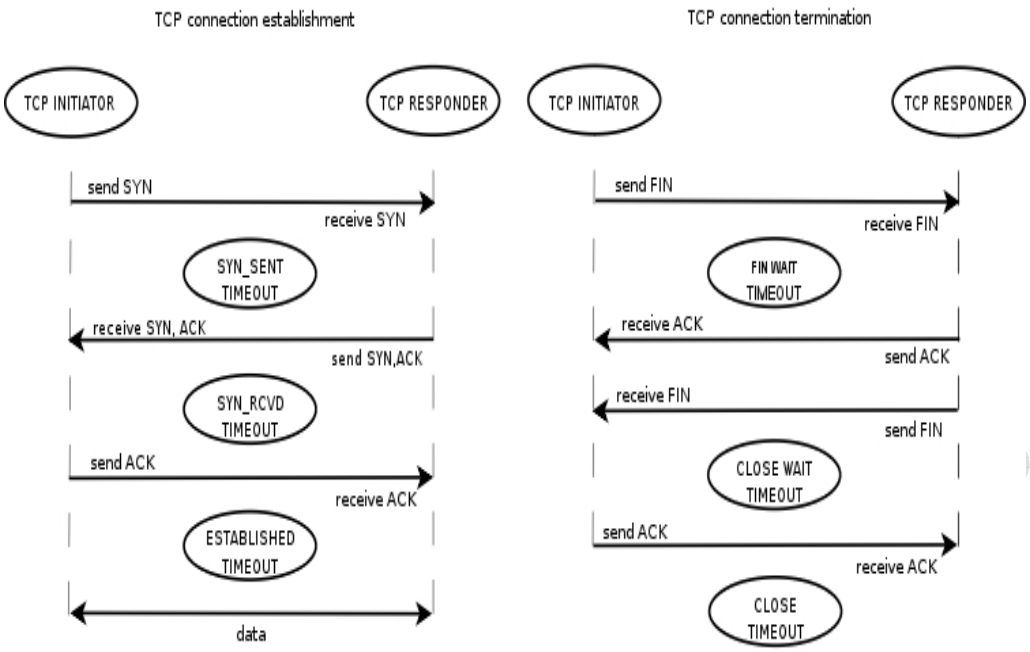
connection-mark (*read-only: text*) - mangle 中设置的连接标记
dst-address (*read-only: IP address:port*) - 连接建立到的目的地址和端口
protocol (*read-only: text*) – IP 协议名和序号
p2p (*read-only: text*) – P2P 协议
reply-src-address (*read-only: IP address:port*) - 从源地址和端口建立的响应连接
reply-dst-address (*read-only: IP address:port*) - 连接建立到的目的地址和端口
src-address (*read-only: IP address:port*) - 从源地址和端口建立的连接
tcp-state (*read-only: text*) - TCP 连接状态
timeout (*read-only: time*) - 直到连接超时的时间量
assured (*read-only*: true | false) - 显示是否看到对该条登记的最后一个包的回应
icmp-id (*read-only*: *integer*) - 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 (*read-only: integer*) - ICMP 类型和代码域
reply-icmp-id (*read-only: integer*) - 包含已接收包的 ICMP ID
reply-icmp-option (*read-only: integer*) - 已接收包的 ICMP 类型和代码域
unreplied (*read-only:* true | false) - 显示是否请求未被回应

连接超时

操作路径: ***/ip firewall connection tracking***

描述

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 (*read-only: integer*) - 在连接状态列表中记录的当前连接数

count-max (*read-only: integer*) - 取决于总内存量的连接状态列表能包含的最大连接数

enable (yes | no; default: **yes**) - 允许或禁止连接追踪

generic-timeout (*time*; default: **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 (*time*; default: **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 (*time*; default: **10s**) - 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 (*time*; default: **10s**) - 在看到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 (*time*; default: **1d**) - 在看到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 (*time*; default: **10s**) - 在看到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 (*time*; default: **1m**) - 在看到匹配连接请求（SYN）之后连接追踪条目存活的最大时间

tcp-syn-sent-timeout (*time*; default: **1m**) - 在看到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 (*time*; default: **10s**) - 在看到紧随连接请求（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 (*time*; default: **10s**) - 在看到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 (*time*; default: **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: 最大超时值却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。

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

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。

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

DMZ 配置事例

属性描述

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。

事例：

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
```

```
1  Chain=dst-nat dst-address=192.168.0.3 action=dst-nat to-dst-address=10.1.0.2
[admin@gateway] ip firewall nat>
```

带宽控制（Queue）

基本信息

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** - 随机公平序列
- **RED** - Random Early Detect
- **PCQ** – 每次连接队列
- **HTB** - 标记令牌桶

规格

功能包要求: ***system***
认证要求: *Level1 (limited to 1 queue) ， Level3*
操作路径: ***/queue***
标准和技术: None
硬件使用: *Significant*

描述

服务质量(QoS)意思是说路由器应该优先考虑并形成网络流通量。 QoS 并非是只关于限流的，它更多的是与提供优良品质的服务相关的。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
- 对固定的时间间隔应用队列
- 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

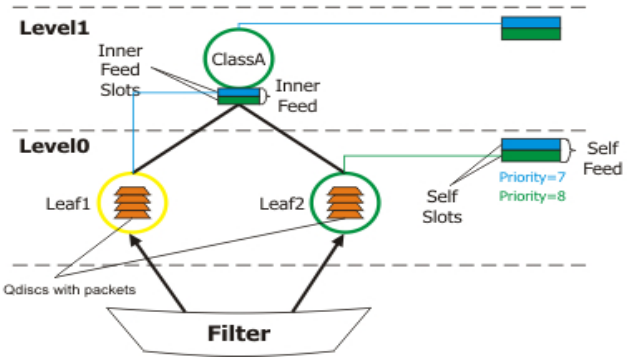
RouterOS 对实际接口增加了三个虚拟接口:

- **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)

HTB 介绍

HTB(等级标记存储桶)是一个对不同类型流量应用不同处理的等级排序规则。通常地，我们可以对一个接口设置唯一一个队列，但是在 RouterOS 队列是附属于主 HTB 的并且一些属性可以由父级队列继承而来。例如，我们可以对一个工作组设置一个最大数据率然后在这个工作组的成员中分配这个大小的流量。

HTB qdisc 细节描述:



HTB 术语:

- **queuing discipline (qdisc)** - 一个保存并维护一个队列的数据包的算法。它指定了向外数据包的顺序(即队列规则可以对包重新排序)。 Qdisc 也决定在没有空间的时候哪个包需要被丢弃。
- **filter** - 一个对包分类的程序。过滤器负责分类数据包以使它们能被发送到响应的队列规则。
- **level** - 在等级表中的类的位置。
- **inner class** - 拥有一个或多个附属子类的类。内部类不存贮任何数据包但是他们做流量成型。这个类也不拥有它自己的优先级。
- **leaf class** - 有一个父类或没有任何子类的类。叶子类总在等级表的 0 等级。没个叶子类都有一个队列规则附属于它。
- **self feed** - 一个代表了来自等级表中其等级的所有激活类数据包的出口的对象。它有 8 个自馈送槽组成。
- **self slot** - 一个相应于每个特定等级的自馈送(self feed)单元。所有的类，在同一等级活跃的，带有同一优先级的都附属于用来发送数据包一个自馈送槽(self slot)
- **active class (at a particular level)** - 附属于给定等级的类。
- **inner feed** - 类似于自馈送对象，呈现在每个内部类上都包含内部自馈送槽
- **inner feed slot** - 类似于自馈送槽。每个内部馈送都包含代表一个优先等级的内部槽

每个类有一个父类并可能有一个或多个子类。没有子类的类被放在维护队列的等级 0，并被叫做"叶子类"。

每个在等级表中的类都可以优先考虑并形成流通量。在 RouterOS 中有 2 个主要用于成型和优先的参数:

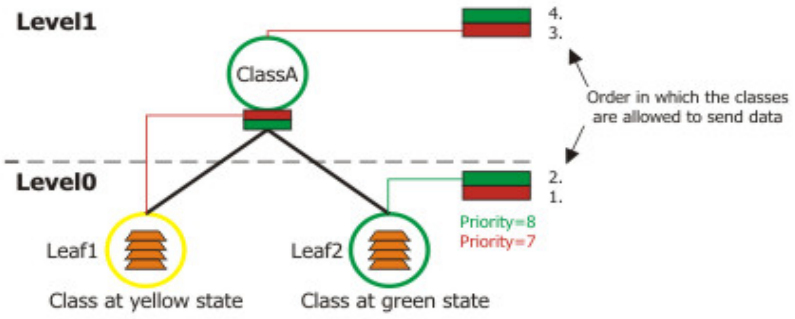
- **limit-at** - 对类承诺的数据率(CIR)
- **max-limit** - 最大允许类达到的数据率(MIR)
- **priority** - 在同等级中类操作的顺序(8 级是最低级，1 是最高级)

每一个 HTB 类都可以根据其占用的数据率处于 3 种状态之一:

- **green** - 表示实际数据率小于或等于 **limit-at** 的类。在这种状态下，类附属于它等级中的相应优先度的自馈送槽并且不论它父类有什么样的限制它都会被满足其 limit-at 的限制。例如：如果我们有一个 **limit-at=512000** 的叶子类而它的父类有 **max-limit=limit-at=128000** 的限制，此时这个类将得到 512kbps。
- **yellow** - 表示实际数据率大于或等于 **limit-at** 的类。这种情况下，类将被附属于其父类内部馈送相应优先度的内部馈送槽，依次地，也可能被附属于其父类这一优先等级的内部槽（以防父类也是 yellow），或者附属于这一优先度的它本身等级的自馈送槽（以防父类是 green）。当过度到这个状态时，类从它的等级的自馈送"断开"并"连接"到其父类的内部馈送。
- **red** - 表示实际数据率超过 **max-limit** 的类。这种类不能从其父类借用速率。

优先级

当一个叶子类想发送一些流量时（因为它们是可以容纳数据包的唯一的类），HTB 就会检查它的优先级。检查将从最高的优先级和最低等级开始，直到最高等级时达到最低优先级：



正如你在图中所看到的，在 green 状态的叶子类总比那些外借的优先级高，因为他们的优先级是更低的等级（等级 0）。在图中，**Leaf1** 总仅会在 **Leaf2** 之后被调用，尽管 **Leaf2**（7）有比 **Leaf1**（8）更高的优先级。

以防相同优先级和相同状态，HTB 使用 round robin 算法调用类。

HTB 实例

下面是一些关于 HTB 如何工作的实例。

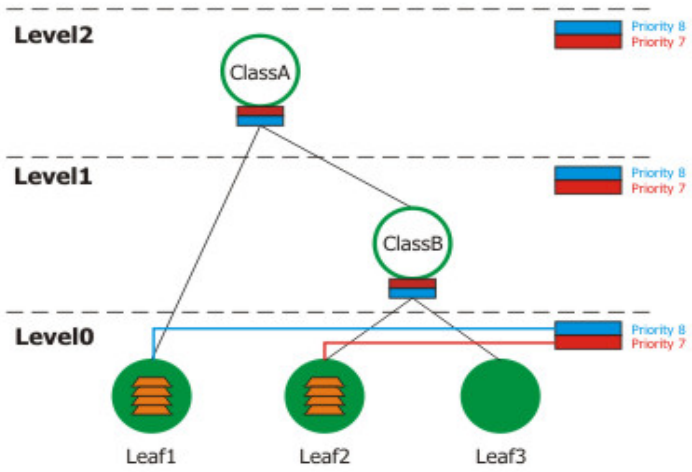
想象如下的情况——我们有 3 种不同的流，在 **/ip firewall mangle** (packet_mark1， packet_mark2 and packet_mark3)标记过，现在建立 HTB 等级制：

```
[admin@MikroTik] queue tree> add name=ClassA parent=Local max-limit=2048000
[admin@MikroTik] queue tree> add name=ClassB parent=ClassA max-limit=1024000
[admin@MikroTik] queue tree> add name=Leaf1 parent=ClassA max-limit=2048000 \
\。。。 limit-at=1024000 packet-mark=packet_mark1 priority=8
[admin@MikroTik] queue tree> add name=Leaf2 parent=ClassB max-limit=1024000 \
\。。。 limit-at=256000 packet-mark=packet_mark2 priority=7
[admin@MikroTik] queue tree> add name=Leaf3 parent=ClassB max-limit=1024000 \
\。。。 limit-at=768000 packet-mark=packet_mark3 priority=8
[admin@MikroTik] queue tree> print
Flags: X - disabled， I - invalid
 0   name="ClassA" parent=Local packet-mark="" limit-at=0 queue=default
     priority=8 max-limit=2048000 burst-limit=0 burst-threshold=0
     burst-time=0s

 1   name="ClassB" parent=ClassA packet-mark="" limit-at=0 queue=default
     priority=8 max-limit=1024000 burst-limit=0 burst-threshold=0
     burst-time=0s

 2   name="Leaf1" parent=ClassA packet-mark=packet_mark1 limit-at=1024000
     queue=default priority=8 max-limit=2048000 burst-limit=0
     burst-threshold=0 burst-time=0s

 3   name="Leaf2" parent=ClassB packet-mark=packet_mark2 limit-at=256000
     queue=default priority=7 max-limit=1024000 burst-limit=0
```

```
    burst-threshold=0 burst-time=0s

 4  name="Leaf3" parent=ClassB packet-mark=packet_mark3 limit-at=768000
    queue=default priority=8 max-limit=1024000 burst-limit=0
    burst-threshold=0 burst-time=0s
[admin@MikroTik] queue tree>
```

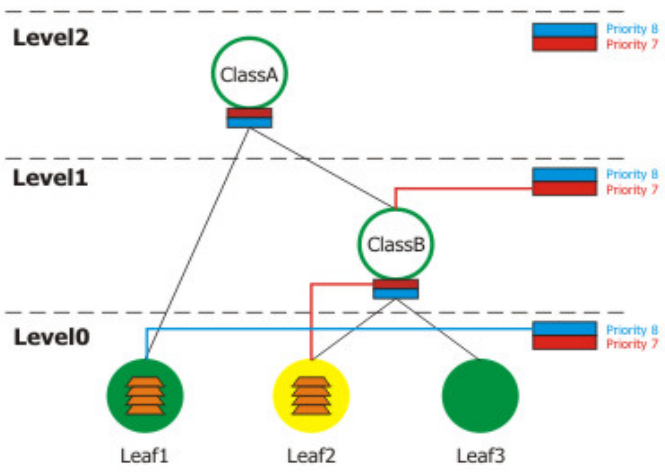
现在让我们使用 HTB 等级制描述一些情况：

1. 想象一种当有数据包到达 Leaf1 和 Leaf2 的情况。正是由于这样，Leaf1 把自己以 priority=8 附属于这个等级（等级 0）的自馈送槽，Leaf2 以 priority=7 附属于自馈送槽。Leaf3 没有数据包发送于是没有任何动作。



这是一个很简单的情况：在等级 0 上有活跃的类（Leaf1 and Leaf2）并且他们都处于 green 状态，他们处理的顺序就按照他们的优先级——首先，我们先处理 Leaf2 然后才是 Leaf1。

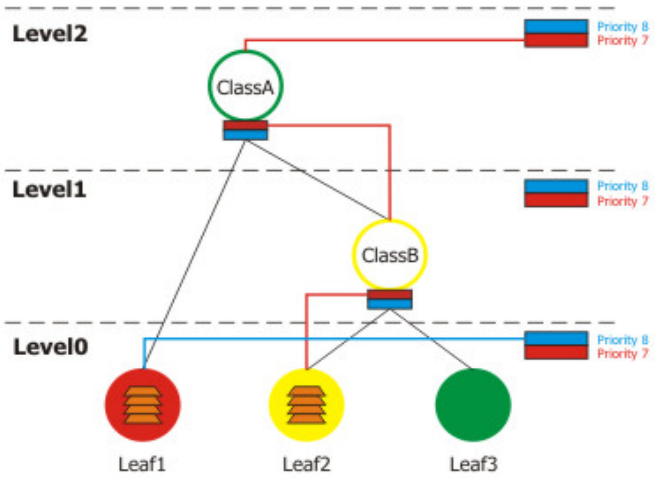
2. 现在我们假设 Leaf2 要发送大于 256kbps 速率的数据包，因此它将附属到其父类（Class B）的内部馈送，然后递归地以 priority=7 附属到等级 1 的自馈送槽。Leaf1 继续保持在 green 状态——它要发送数据包但不会快于 1Mbps。Leaf3 仍然没有包发送。



这是一个非常有意思的情况因为 Leaf1 有比 Leaf2 更高的优先级（当它还处于 green 状态是），尽管我们要为它配置一个比 Leaf2 更低的优先级（8）。这是因为 Leaf2 已经从它的等级 0 的自馈送断开并连接了其父类（Class B）

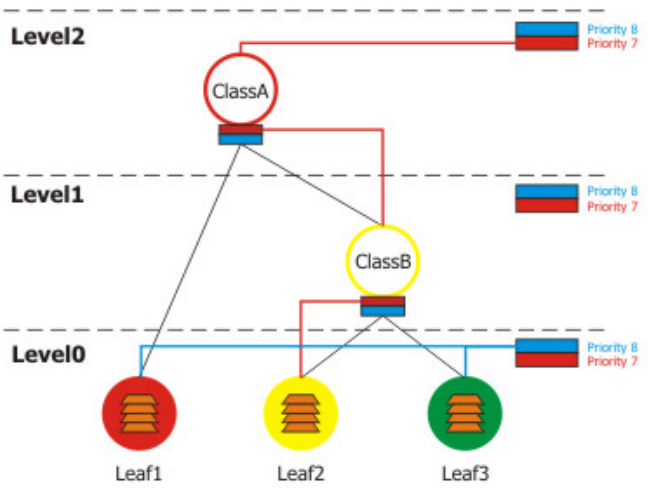
等级 1 的自馈送。因此，Leaf2 的优先级已经变成了等级 1。记住首先，我们调用那些在低等级有高优先级的类，然后再继续到下一个等级，等等。

3. 考虑这种情况：Leaf1 达到了它的 max-limit 并变为 red 状态，Leaf2 使用大于 1Mbps（小于 2Mbps），于是它的父类 Class B 就要借用 Class A 并且变为 yellow 状态。Leaf3 仍旧没有包发送。



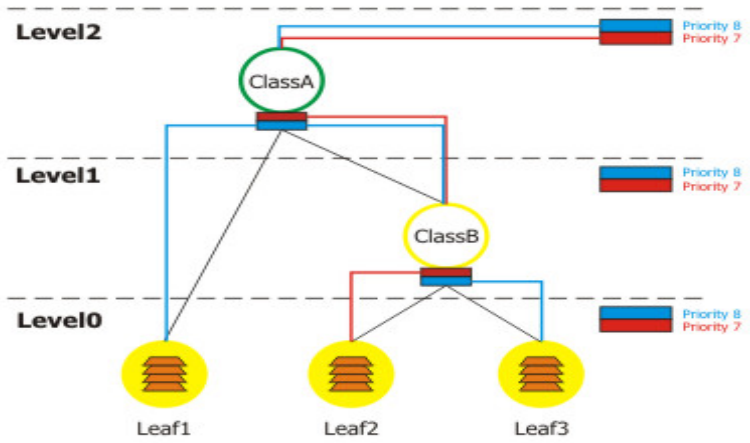
这种假设表明 Leaf1 达到了它的 max-limit，且不能从其父类（Class A）借用。Leaf2 等级达到了等级 2 并从 Class B 借用，递归地 Class B 必须从 Class A 借用因为它没有足够可用的速率。由于 Leaf3 没有包发送，唯一发送包的类就只有 Leaf2。

4. 假设 Leaf2 从 Class B 借用，Class B 从 Class A 借用，但 Class A 达到了它的 max-limit（2Mbps）。



在这种情况下 Leaf2 在 yellow 状态下，但它再也不能借用了（由于 Class B 再也不能从 Class A 借用）。

5. 最后，让我们看看如果 Leaf1， Leaf2， Leaf3 和 Class B 都在 yellow 状态， Class A 在 green 状态会发生什么。

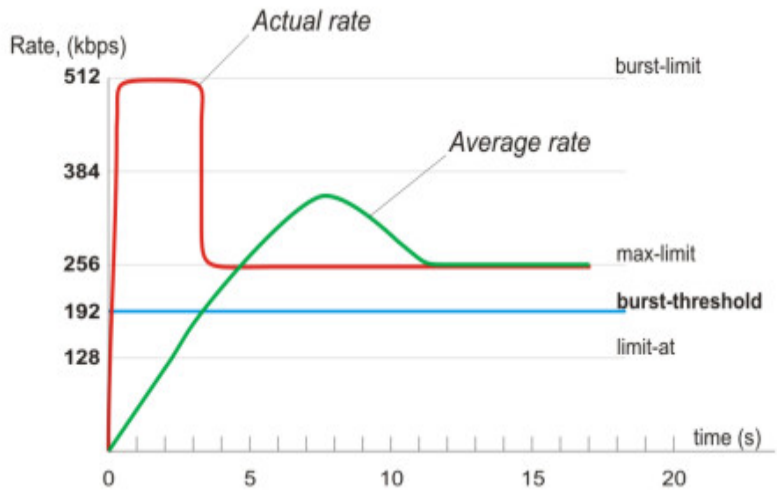


Leaf1 从 ClassA 借用，Leaf2 和 Leaf3 从 ClassB 借用，ClassB 也从 ClassA 借用。现在所有的优先级都“移动到了”等级 2。所以 Leaf2 有最高的优先级并会被首先调用。由于 Leaf2 和 Leaf3 在相同的等级（2）上有相同的优先级（8），使用 round robin 算法将可以调用他们。

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)。

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 and interface queue。如果是指向路由器的它将穿国 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。

队列类型

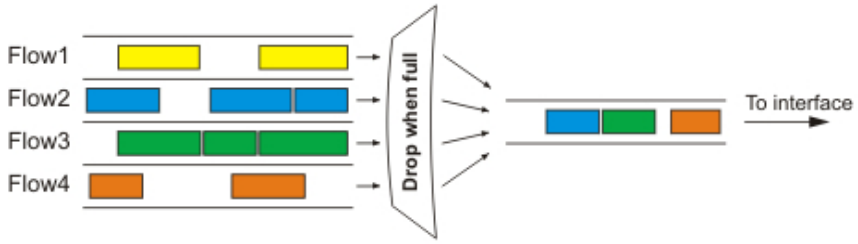
操作路径：***/queue type***

描述

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。

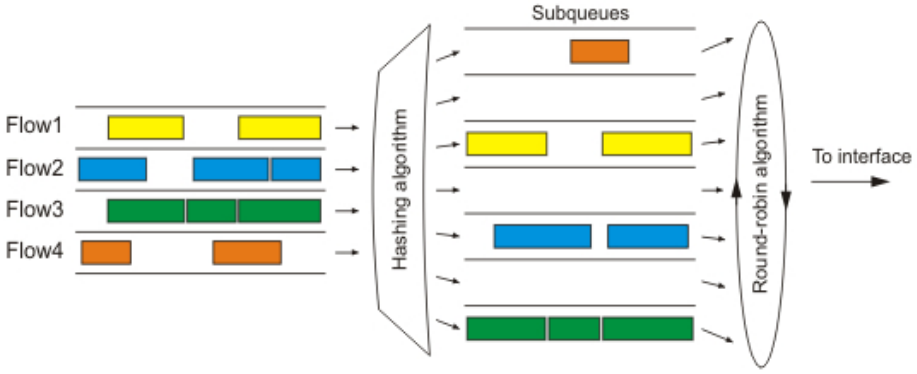


如果你的连接不拥塞的话请使用 FIFO 队列规则。

SFQ

随机公平排序（SFQ）不会限制流量。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。

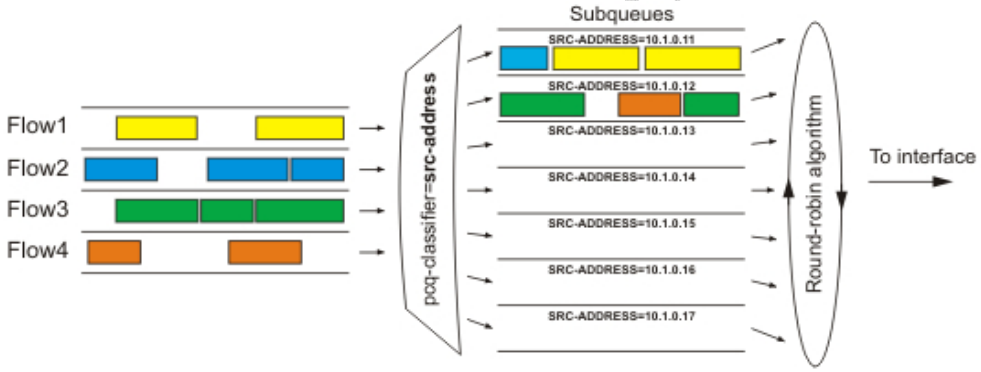


整个 SFQ 队列可以容纳 128 个数据包并且对这些包有 1024 个子队列可用。对拥挤的连接使用 SFQ 可以保证一些连接不至于空等待（starve）。

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。



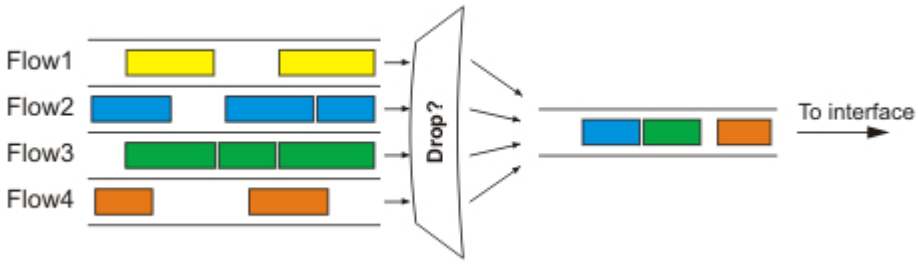
如果你以 **src-address** 对包分类那么所有带有不同源 IP 地址的包将被集合在不同的子队列中。现在你可以使用 **pcq-rate** 参数对每一个子队列进行限制或均衡。或许最重要的部分是决定我们到底应该把这个队列附属到哪个接口上。如果我们把它依附在本地接口上，那么所有来自公网接口的数据流都将以 src-address（很可能这不是我们想要的）地址分组；相反地如果我们把它依附到公共接口，所有来自我们客户的数据都会以 src-address 分组——于是我们可以很容易的限制或者均衡客户的上载。

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

Random Early Detection （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。



主要地，RED 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 (*integer*; default: **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 - Bytes First-In-First-Out
pcq - Per Connection Queue
pfifo - Packets First-In-First-Out
red - Random Early Detection
sfq - Stohastic Fairness Queuing
name (*name*) - 队列类型相关名称
pcq-classifier (dst-address | dst-port | src-address | src-port; default: **""**) - 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 (*integer*; default: **50**) - 可以容纳一个单个 PCQ 子队列的包的数目
pcq-rate (*integer*; default: **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 (*integer*; default: **2000**) - 可以容纳整个 PCQ 队列的包的数目
pfifo-limit (*integer*) - PFIFP 队列可以容纳包的最大数目
red-avg-packet (*integer*; default: **1000**) - 被 RED 用来对平均队列长度计算
red-burst (*integer*) - 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 (*integer*) - 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 (*integer*) - 以字节计算。数据包标记概率最高的平均队列长度
red-min-threshold (*integer*) - 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 (*integer*; default: **1514**) - 在一个 round-robin 循环中从子队列发出的字节数
sfq-perturb (*integer*; default: **5**) - 以秒计时。指定改变 SFQ 的散列算法的频率

Queue interface 接口队列

操作路径: ***/queue interface***

如果想要从一个接口发送数据包，那么即使你不想对流量进行限制也必须把它们入队列。这里你可以指定将被用于传送数据的队列类型。

注意如果其他队列应用于一个特定的包那么这些设置就没有用了。

属性描述

interface (*read-only: name*; default: **name of the interface**) - 接口名称
queue (*name*; default: **default**) - 接口使用的队列类型

事例

设置无线接口使用 **wireless-default** 队列：

```
[admin@MikroTik] queue interface> set 0 queue=wireless-default
[admin@MikroTik] queue interface> print
 # INTERFACE QUEUE
 0 wlan1    wireless-default
[admin@MikroTik] queue interface>
```

Simple Queue 简单队列

限制数据率的 IP 地址和/或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。它们具有有用的整体特征：

- P2P 流量队列
- 在选定时间间隔应用队列规则
- 优先级
- 从 */ip firewall mangle* 使用多重包标记
- 形成双向流量（对上传和下载的总量的限制）

属性描述

burst-limit (*integer/integer*) - 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 (*integer/integer*) - 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 (*integer/integer*) - 用于计算平均数据率。以 in/out（目标上传/下载）的形式。
direction (none both upload download) - 流量方向，受队列影响
none - the queue is effectively inactive
both - the queue limits both target upload and target download
upload - the queue limits only target upload， leaving the download rates unlimited
download - the queue limits only target download， leaving the upload rates unlimited
dst-address (*IP address/netmask*) - 要匹配的目标地址
dst-netmask (*netmask*) - *dst-address* 的掩码
interface (*text*) - 队列应用的对象端口。
limit-at (*integer/integer*) – 该队列以 in/out（目标上传/下载）的形式约定的数据率
max-limit (*integer/integer*) - 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 (*text*) - 队列的描述性名称
p2p (any | all-p2p | bit-torrent | blubster | direct-connect | edonkey | fasttrack | gnutella | soulseek | winmx) - 压迫匹配的 P2P 流量类型
all-p2p - match all P2P traffic
any - match any packet (i。e。， do not check this property)
packet-marks (*name*; default: **""**) - **/ip firewall** 中的数据包标记
mangle 更多数据包标记使用逗号(",")搁开。
parent (*name*) - 父队列在等级制度中的名称。只能是其他简单队列
priority (*integer*: 1..8) -队列的优先级。 1 是最高级的，8 是最低的

queue (*name/name*; default: **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 address/netmask*) - 限制目标 IP 地址（源地址）。使用多地址用逗号分搁开

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

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

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

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

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

total-max-limit (*integer*) - *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 (*name*) - **global-total** 队列的队列规则

Queue tree 队列树

操作路径: ***/queue tree***

当你想使用基于协议，端口，IP 地址群等的复杂数据率分配时，你应使用队列树。首先你应该在**/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 (*integer*) - 当脉冲串激活时可以达到的最大数据率

burst-threshold (*integer*) - 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*。

burst-time (*integer*) - 用于计算平均数据率。

flow (*text*) - 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 (*integer*) - 这个队列的约定数据率

max-limit (*integer*) - 在有足够带宽可用的情况下可大到的数据率

name (*text*) - 队列的描述性名称

parent (*text*) - 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。

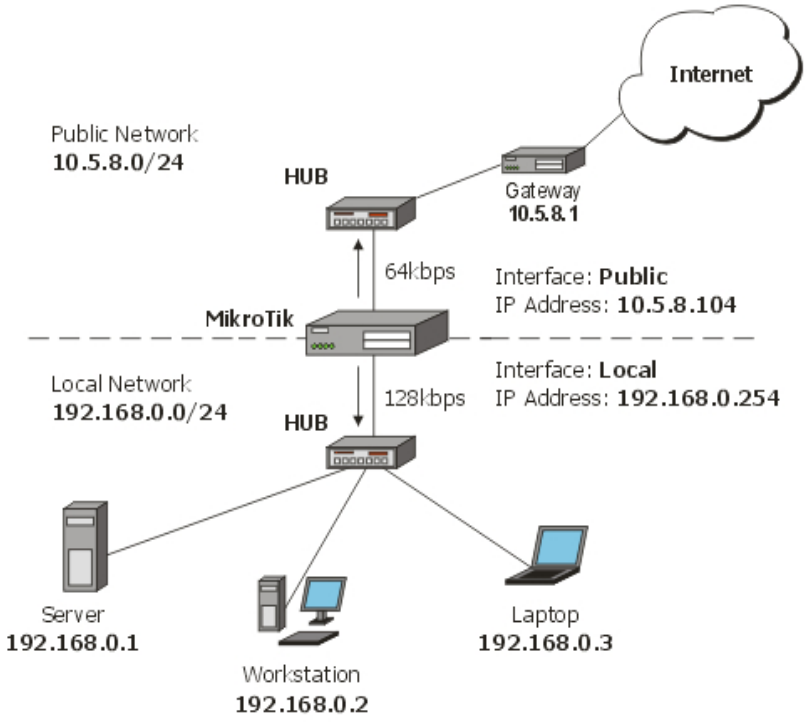
priority (*integer*: 1..8) - 队列的优先级。 1 是最高级的， 8 最低的。

queue (*text*) - 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叶子队列。

应用实例

128Kibps/64Kibps 仿真举例

假设我们想要仿真一个 128kps 下载且 64kps 上传的连接 IP 网络 192.168.0.0/24 线路。网络的基本设置如



为解决这个问题，我们使用简单队列。MikroTik router 的 IP 地址：

```
 [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>
```

路由器：

```
[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>
```

为网络 192.168.0.0/24 的客户端添加一个限制下载流量为 128kbps 上传流量 64kbps 的简单队列规则，使用接口 **Local**。

```
[admin@MikroTik] queue simple> add name=Limit-Local interface=Local \
\. target-address=192.168.0.0/24 max-limit=65536/131072
[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
```

```
     interface=Local parent=none priority=8 queue=default/default
     limit-at=0/0 max-limit=65536/131072 total-queue=default
[admin@MikroTik] queue simple>
```

参数 **max-limit** 削减了最大可用带宽。从客户的角度看，**65536/131072** 代表他们将会得到 131072bps 的下载和 65536bps 的上传流量。参数 **target-addresses** 定义了规则应用的目标网络（或者被逗号分捆开的网络）。

现在看一下通信负载：

```
[admin@MikroTik] interface> monitor-traffic Local
  received-packets-per-second: 7
      received-bits-per-second: 68kbps
       sent-packets-per-second: 13
          sent-bits-per-second: 135kbps

[admin@MikroTik] interface>
```

或许你不想让服务器受到任何限制，如果这样的话就添加一个没有任何限制的队列（**max-limit=0/0** 代表没有任何限制）并把它移到列表的表头位置：

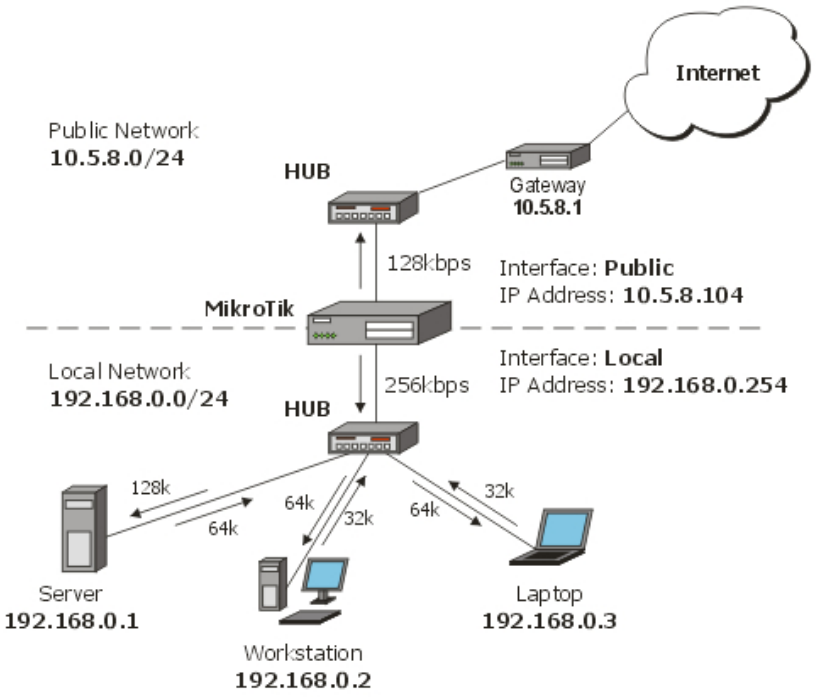
```
[admin@MikroTik] queue simple> add name=Server target-addresses=192.168.0.1/32 \\...
interface=Local
[admin@MikroTik] queue simple> print
Flags: X - disabled, I - invalid, D - dynamic
 0   name="Limit-Local" target-addresses=192.168.0.0/24 dst-address=0.0.0.0/0
     interface=Local parent=none priority=8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

队列树伪装实例

在前一个例子中我们设置了 128kbps 下载和 64kbps 上传流量的本地网络。在这里例子中我们将为本地网络配置 256kbps 下载（服务器使用 128kbps，工作站使用 64kbps 其他的使用 64kbps）和 128kbps 的上传流量（同理，各自分别为

64/32/32kbps）。另外，如果还有闲余带宽就把在用户中平均共享。例如：如果我们关掉便携式电脑就把它的 64kbps 下载和 32kbps 的上传共享给服务器和工作站。

当使用伪装时，你必须用 **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** 动作。



首先，标记服务器的下载和上传流量。对第一个规则我们标记向外的连接，对第二个规则，标记所有属于这个连接的数据包：

```
[admin@MikroTik] ip firewall mangle> add src-address=192.168.0.1/32 \
\... action=mark-connection new-connection-mark=server-con chain=prerouting
[admin@MikroTik] ip firewall mangle> add connection-mark=server-con \
\... action=mark-packet new-packet-mark=server chain=prerouting
[admin@MikroTik] ip firewall mangle> print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=prerouting src-address=192.168.0.1 action=mark-connection
     new-connection-mark=server-con

 1   chain=prerouting connection-mark=server-con action=mark-packet
     new-packet-mark=server
[admin@MikroTik] ip firewall mangle>
```

对于便携式电脑和工作站同样地：

```
[admin@MikroTik] ip firewall mangle> add src-address=192.168.0.2 \
\... action=mark-connection new-connection-mark=lap_works-con chain=prerouting
[admin@MikroTik] ip firewall mangle> add src-address=192.168.0.3 \
\... action=mark-connection new-connection-mark=lap_works-con chain=prerouting
[admin@MikroTik] ip firewall mangle> add connection-mark=lap_works-con \
```

```
\... action=mark-packet new-packet-mark=lap_work chain=prerouting
[admin@MikroTik] ip firewall mangle> print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=prerouting src-address=192.168.0.1 action=mark-connection
     new-connection-mark=server-con

 1   chain=prerouting connection-mark=server-con action=mark-packet
     new-packet-mark=server

 2   chain=prerouting src-address=192.168.0.2 action=mark-connection
     new-connection-mark=lap_works-con

3   chain=prerouting src-address=192.168.0.3 action=mark-connection
     new-connection-mark=lap_works-con

4   chain=prerouting connection-mark=lap_works-con action=mark-packet
    new-packet-mark=lap_work
[admin@MikroTik] ip firewall mangle>
```

如你所见，我们标记了我们用相同的流标记了属于便携电脑和工作站的连接。

在 **/queue tree** 中添加一条限制服务器的下载和上传的规则：

```
[admin@MikroTik] queue tree> add name=Server-Download parent=Local \
\... limit-at=131072 packet-mark=server max-limit=262144
[admin@MikroTik] queue tree> add name=Server-Upload parent=Public \
\... limit-at=65536 packet-mark=server max-limit=131072
[admin@MikroTik] queue tree> print
Flags: X - disabled, I - invalid
 0   name="Server-Download" parent=Local packet-mark=server limit-at=131072
     queue=default priority=8 max-limit=262144 burst-limit=0
    burst-threshold=0 burst-time=0s
 1   name="Server-Upload" parent=Public packet-mark=server limit-at=65536
     queue=default priority=8 max-limit=131072 burst-limit=0
     burst-threshold=0 burst-time=0s
[admin@MikroTik] queue tree>
```

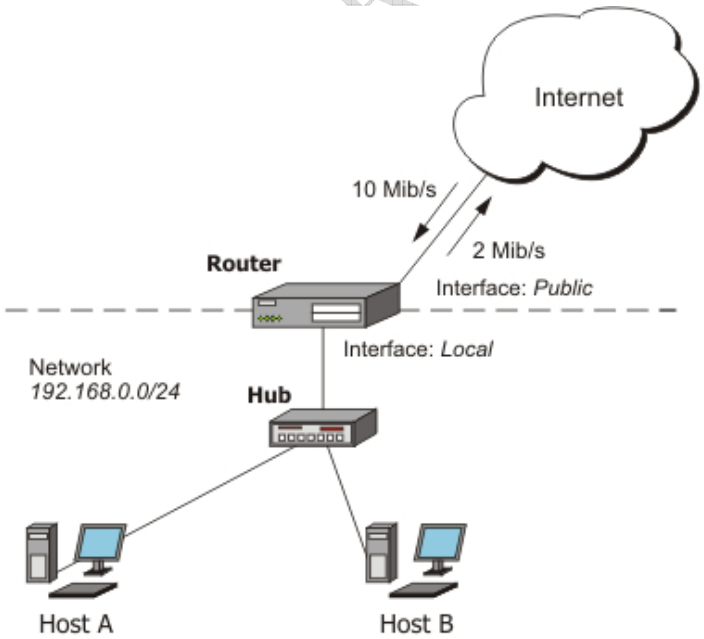
对于便携电脑和工作站有相类似的配置：

```
[admin@MikroTik] queue tree> add name=Laptop-Wkst-Down parent=Local \
\... packet-mark=lap_work limit-at=65535 max-limit=262144
[admin@MikroTik] queue tree> add name=Laptop-Wkst-Up parent=Public \
\... packet-mark=lap_work limit-at=32768 max-limit=131072
[admin@MikroTik] queue tree> print
Flags: X - disabled, I - invalid
 0   name="Server-Download" parent=Local packet-mark=server limit-at=131072
     queue=default priority=8 max-limit=262144 burst-limit=0
```

```
    burst-threshold=0 burst-time=0s

 1  name="Server-Upload" parent=Public packet-mark=server limit-at=65536
    queue=default priority=8 max-limit=131072 burst-limit=0
    burst-threshold=0 burst-time=0s

 2  name="Laptop-Wkst-Down" parent=Local packet-mark=lap_work limit-at=65535
    queue=default priority=8 max-limit=262144 burst-limit=0
    burst-threshold=0 burst-time=0s

 3  name="Laptop-Wkst-Up" parent=Public packet-mark=lap_work limit-at=32768
    queue=default priority=8 max-limit=131072 burst-limit=0
    burst-threshold=0 burst-time=0s
[admin@MikroTik] queue tree>
```

在用户中平均共享带宽

这个例子说明了如何均等地在网络 192.168.0.0/24 的活动用户中共享 10Mbps 的下载和 2Mbps 的上传流量。如果 **Host A** 正在使用 2Mbps 下载，**Host B** 则得到会 8Mbps，反之亦然。 There might be situations when both hosts want to use maximum bandwidth (10 Mibps)， then they will receive 5 Mibps each， the same goes for upload 也可能出现两个主机都想使用最大带宽（10Mbps）的情况，那么他们将会每个得到 5Mbps，上传亦同理。这个设置对于 2 个用户以上的依然成立。



首先，用 **users** 标记来自本地网络 192.168.0.0/24 的流量：

```
/ip firewall mangle add chain=forward src-address=192.168.0.0/24 \
   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
```

现在我们将添加 2 个新的 PCQ 类型。首先 **pcq-download** 将用目标地址把所有流量分组。由于我们将把这个队列类型附属到本地接口，所以它将为每个下载到网络 192.168.0.0/24 的目标地址（用户）产成一个动态的队列。第二个类型叫做 **pcq-upload**，将会用源地址把流量分组。我们将把这个队列附属到外网接口，于是它会为每个从本地网络 192.168.0.0/24 上传到因特网的用户产生一个动态队列。

```
/queue type add name=pcq-download kind=pcq pcq-classifier=dst-address
/queue type add name=pcq-upload kind=pcq pcq-classifier=src-address
```

最后，为下载流量生成一个队列：

```
/queue tree add name=Download parent=Local max-limit=10240000
/queue tree add parent=Download queue=pcq-download packet-mark=users
```

对于上传流量：

```
/queue tree add name=Upload parent=Public max-limit=2048000
/queue tree add parent=Upload queue=pcq-upload packet-mark=users
```

注意！如果你的 ISP 不能保证一个固定量的流量，你可以仅使用一个队列用于上传一个用于下载直接附属于因特网。

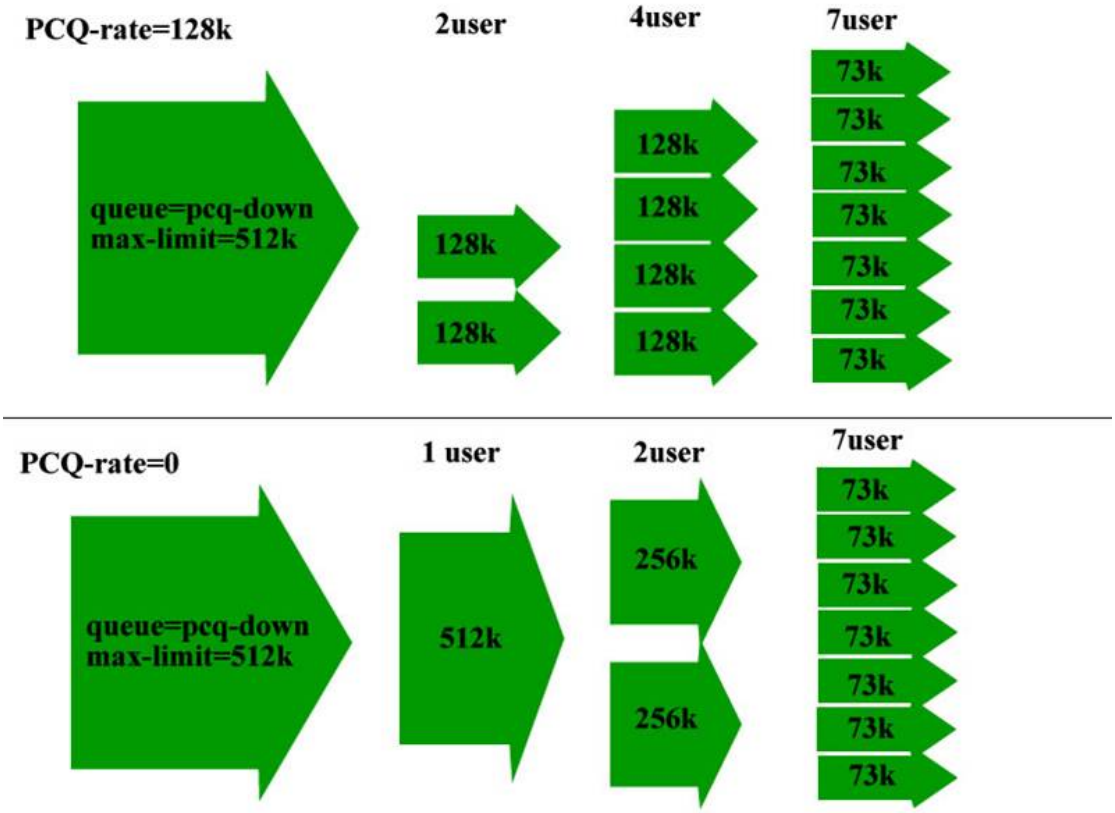
```
/queue tree add parent=Local queue=pcq-download packet-mark=users
/queue tree add parent=Public queue=pcq-upload packet-mark=users
```

注：如果你不想通过 mangle 和队列树控制流量，你可以直接在简单队列做设置，具体设置如下：

```
/queue simple add queue=pcq-upload/pcq-download target-addresses=192.168.0.0/24
```

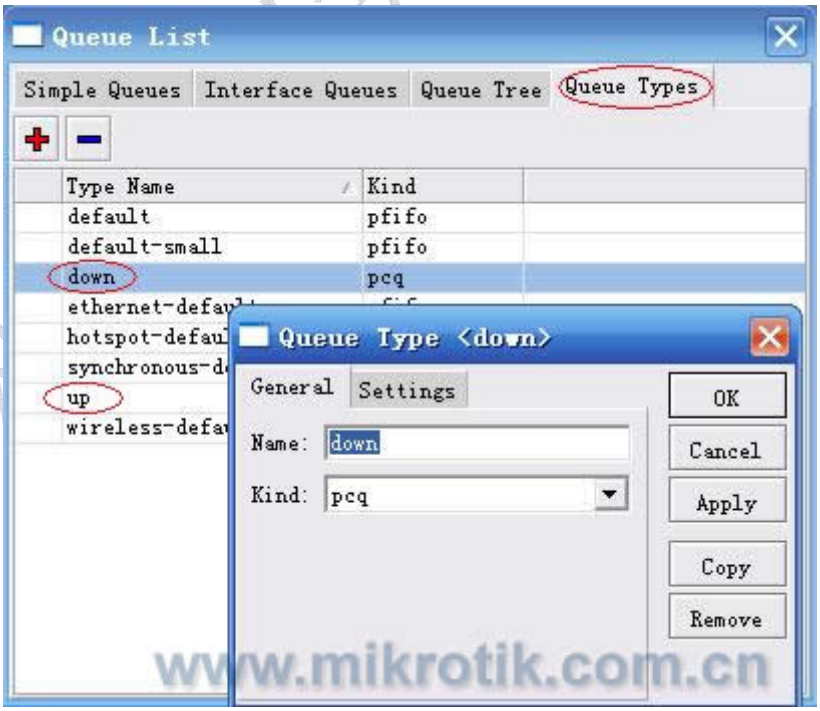
如何实现 RouterOS 的动态流量控制

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：

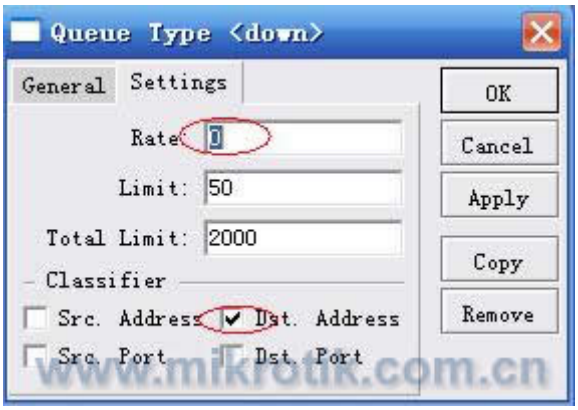


通过上图，我们可以看到当 PCQ 的速率设定为 128k 的时候，平均每个用户将会得到同样的带宽 128k，当上网高峰期的时候 PCQ 才会做二次流量分配，如果 PCQ 的速率在开始就设定为 0k，这样在一个用户的时候就可以得到全部带宽，之后是 2 个用户平均分配，依次类推，但最后带宽会控制在 73k 的范围内，控制最小使用带宽，保证用户正常使用。

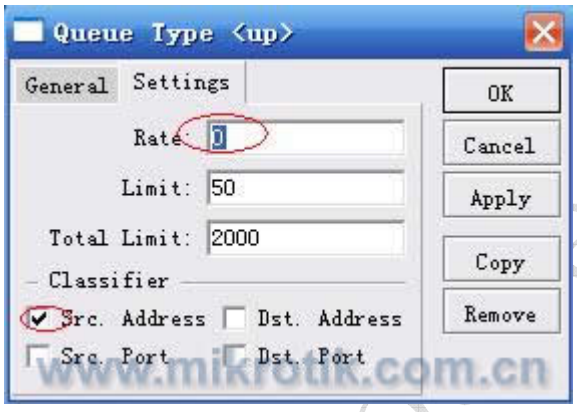
首先进入 Queue Type 中配置 PCQ 的上行和下行：



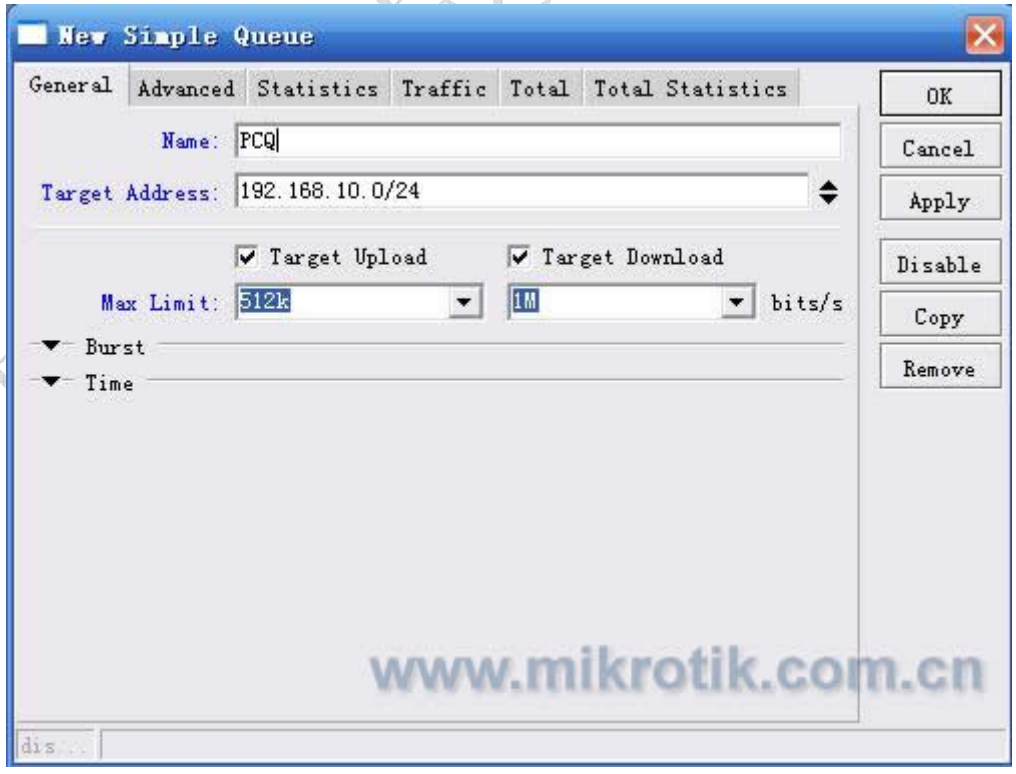
在配置 PCQ 的速率的时候将 rate=0，即每个用户不用配置流量速率，下面是 down 即下行的配置：



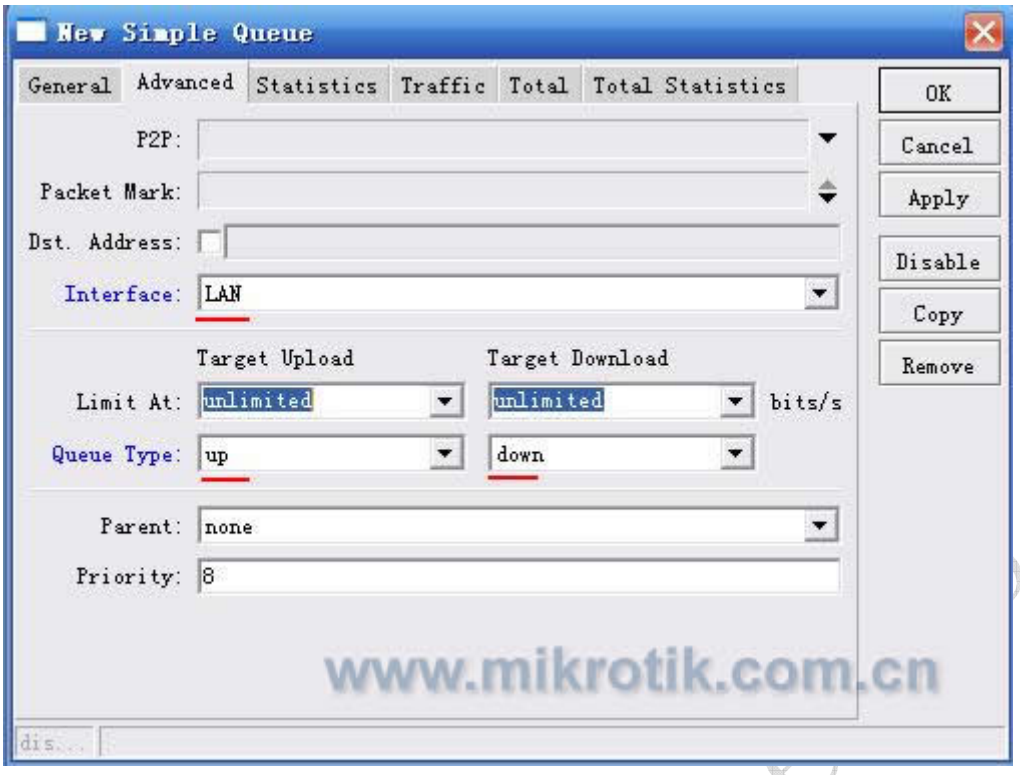
同样在上行配置如下：



在配置好 Queue Type 后我们进入 Simple Queue 中配置流量控制规则，这里我们的总出口带宽假设为 1M，上行带宽为 512k，内网地址段为 192.168.10.0/24：



接下来配置 Interface 和 Queue Type，选择上行和下行的 PCQ 类型分别为 Up 和 Down：



这样 PCQ 的动态流量控制就设定完成了，这样就能实现根据用户数占用流量来动态分配带宽，这样能达到带宽的有效分配和利用。

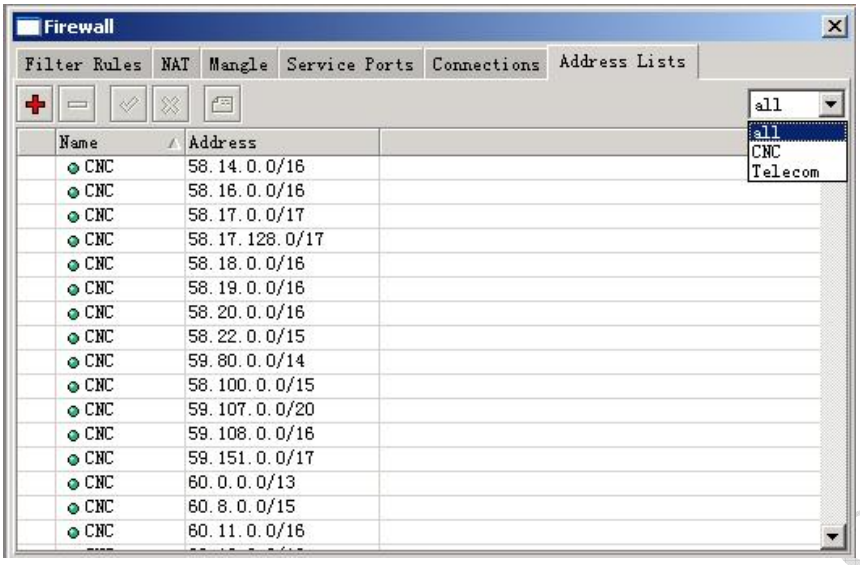
如果配置到电信网通的流量控制

对于电信和网通的IP地址段是已知，那么我们可以通过通过地址标记来实现对这些地址的流量控制，首先我们将电信和网通的地址段导入RouterOS的address-list中（可以在 www.mikrotik.com.cn下载到）

通过 import 命令，导入地址列表：

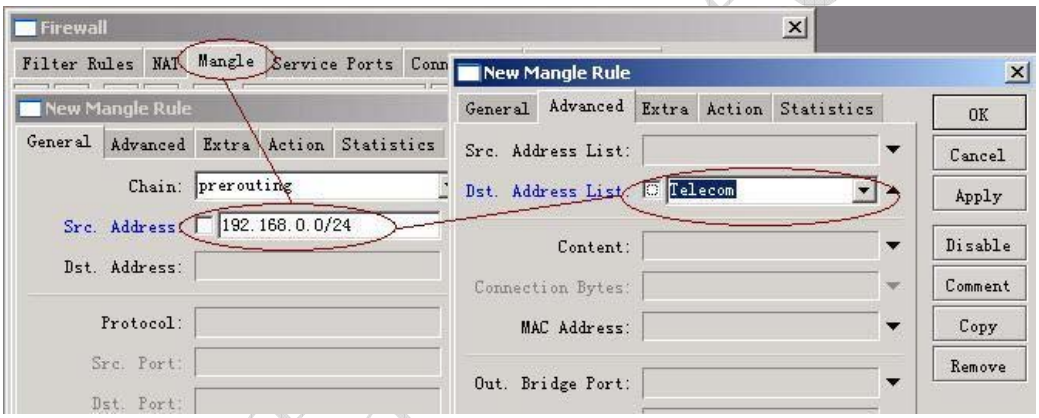
```
MikroTik RouterOS 2.9.34 (c) 1999-2006       http://www.mikrotik.com/

Terminal vt102 detected, using multiline input mode
[admin@CDNAT] > import cnc.rsc
Opening script file cnc.rsc
Script file loaded and executed successfully
[admin@CDNAT] > import tel.rsc
Opening script file tel.rsc
Script file loaded and executed successfully
[admin@CDNAT] >
```

导入后我们可以在/ip firewall address-list 中找到：



配置数据标记 mangle

进入/ip firewall mangle 设置，这里我们定义访问电信的流量控制，我们的内网地址段为 192.168.0.0/24，所有这里我们配置源地址 src-address=192.168.0.0/24。在 mangle 中先标记连接，然后在从连接中提取数据包：

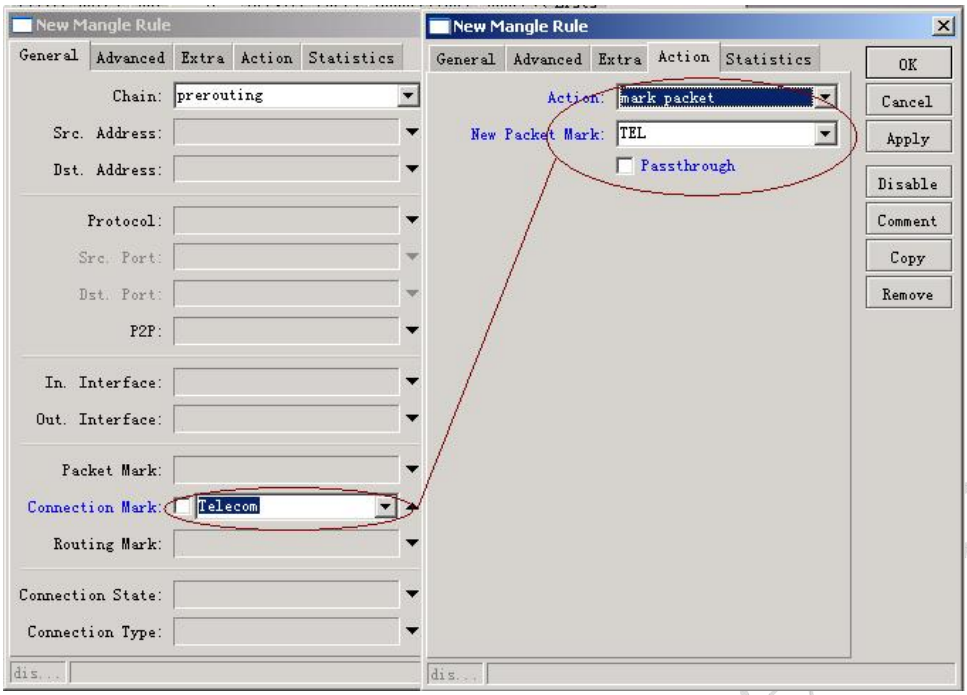


定义标记类型：

源代码：

```
/ ip firewall mangle
add chain=prerouting src-address=192.168.0.0/24 dst-address-list=Telecom
action=mark-connection  new-connection-mark=Telecom passthrough=yes comment=""
disabled=no
```

现在从标记的连接 Telecom 中提取数据包：

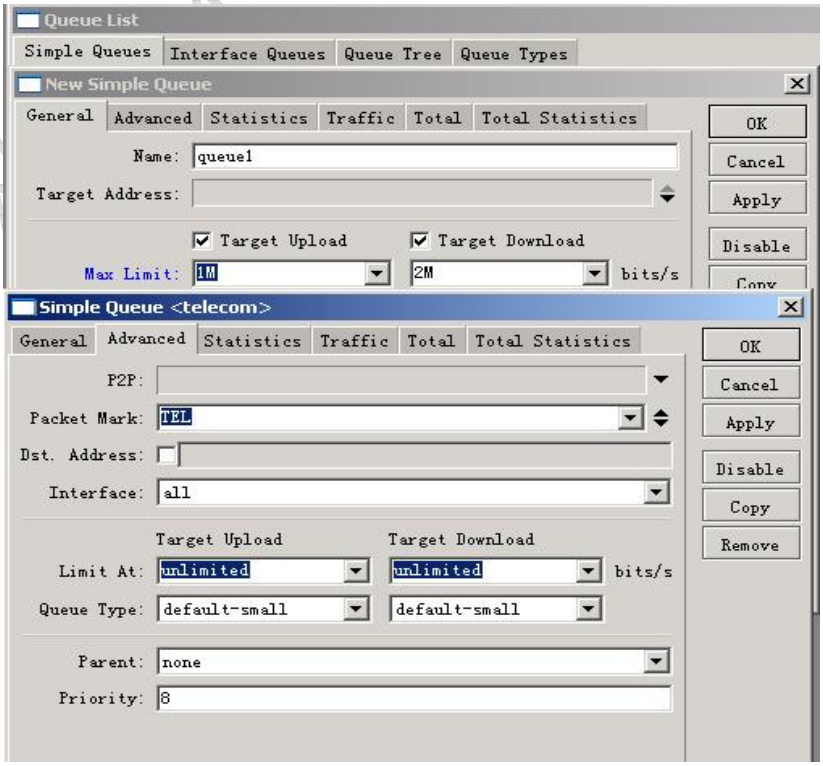


源代码：

```
/ ip firewall mangle
add chain=prerouting connection-mark=Telecom action=mark-packet
new-packet-mark=TEL passthrough=no comment="" disabled=no
```

配置 simple queue

现在我们进入/queue simple 对列中配置流控规则，在这里我们把到电信的带宽控制在 1M 上行和 2M 下行



源代码：

```
/ queue simple
add name="telecom" dst-address=0.0.0.0/0 interface=all parent=none packet-marks=TEL
direction=both priority=8 queue=default-small/default-small limit-at=0/0
max-limit=1000000/2000000 total-queue=default-small disabled=no
```

这样对电信的带宽控制便完成，控制网通带宽同样的

N A T

基本信息

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 (number of rules limited to 1) ， 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

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(NAT 路由器)起的就是在数据包进/出局域网时重写 IP 地址的作用。

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; default: **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 (*name*) -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 (*time*; default: **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*) - 定义一个具有特定规则的链。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。如果输入与一个已定义好的链名不匹配，那么一个新的链将被生成。

- **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 (*text*) - 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 (*integer-integer*) - 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 (*integer*, *netmask*) - 限制每个地址或地址群的连接限度。

connection-mark (*name*) - 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 (*text*) - 文本数据包必须按顺序排列以与匹配规则

dst-address (*IP address*/*netmask* | *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 (*name*) - 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 (*integer*/*time*{0,1},*integer*, 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 (*integer*: 0..65535-*integer*: 0..65535{*}) - 目的端口数或范围

hotspot (*multiple choice:* 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 (*integer*:*integer*) - 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 (*integer*/*time*{0，1}，*integer*) - 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 (*text*) - 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 (*integer*, *integer*: 0..15，*integer*{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 (*integer*: 0..65535-*integer*: 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 | *integer*) - 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 (*integer，time，integer，integer*) - 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 (*integer*) - 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 address/netmask* | *IP address-IP address*) - 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 (*integer*: 0..65535-*integer*: 0..65535{*}) - 源端口数或范围

tcp-mss (*integer*: 0..65535) - 与 IP 包的 TCP MSS 值匹配

time (*time-time*，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 (*integer*: 0..65535-*integer*: 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)

NAT 应用

在这部分中一些 NAT 的应用及实例将被讨论。

基本 NAT 配置

假设我们要做这样的一个路由器：

- 用一个地址把专用局域网“隐藏”再其“后面”
- 对本地服务器提供一个公用 IP
- 创造 1：1 的网络地址映像

源 NAT 实例（伪装）

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（伪装）。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，必须向防火墙配置中添加一个带有“伪装”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）。

目的 NAT 实例

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。200/32 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
```

1：1NAT 实例

如果你想从公用 IP 子网 11.11.11.0/24 访问本地的 2.2.2.0/24，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=11.11.11.1-11.11.11.254 \
       action=netmap to-addresses=2.2.2.1-2.2.2.254

/ip firewall nat add chain=srcnat src-address=2.2.2.1-2.2.2.254 \
       action=netmap to-addresses=11.11.11.1-11.11.11.254
```

DNS 与 DNS 缓存

基本信息

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。

功能说明

需要功能包: ***system***
需要等级: *Level1*
操作路径: ***/ip dns***
标准与技术协议: DNS

Client配置与缓存设置

操作路径: ***/ip dns***

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 159.148.60.2：

```
[admin@MikroTik] ip dns> set primary-dns=159.148.60.2
[admin@MikroTik] ip dns> print
    resolve-mode: remote-dns
     primary-dns: 159.148.60.2
   secondary-dns: 0.0.0.0
[admin@MikroTik] ip dns>
```

缓存状态

操作路径: ***/ip dns cache***

属性描述

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

静态 DNS

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***

指令属性

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

事例

```
[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>
```

Bridge 网桥

基本信息

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）。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)
- 多重桥接口
- Bridge associations on a per-interface basis
- 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 set ether1,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
硬件使用: *Not significant*

描述

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。根桥是拥有最低桥 ID 的桥。

桥接口配置

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 (*read-only: 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。

实例

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***

子目录用于使接口守制于一个特殊的桥接口。

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 (*read-only: 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：

实例

把 **ether1 和 ether2** 分到已创建的桥 **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 起）：

```
[admin@MikroTik] interface bridge port> add 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
[admin@MikroTik] interface bridge port>
```

注意现在已经再也没有了 **wlan1** 接口了，因为它是作为桥端口添加进去的。

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 (*text*) – 桥 ID，以如下形式 bridge-priority，bridge-MAC-address
designated-root (*text*) - 根桥的 ID
path-cost (*integer*) - 到根桥所需总代价
root-port (*name*) -根桥连接的端口

实例

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>
```

桥端口监测

命令名：***/interface bridge port monitor***

属于一个桥接口的统计

属性描述

designated-port (*text*) - 指定根桥端口
designated-root (*text*) - 最靠近根桥的桥 ID
port-id (*integer*) - 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

实例

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]
```

桥主机列表

命令名：***/interface bridge host***

属性描述

age (*read-only: 时间*) - 从主机获得最后一个包开始的时间
bridge (*read-only: 名称*) - 属于词条（entry）的桥
local (*read-only: 标志*) - 主机词条是否是桥本身的
mac-address (*read-only: MAC 地址*) - 主机 MAC 地址
on-interface (*read-only: 名称*) - 主机所连接的桥接的接口

实例

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>
```

桥防火墙基本描述

规格

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** - 有三个预定义链的桥防火墙：
  - **input** -其目的是桥的过滤器包（包括将被路由的那些数据包，因为无论怎么说他们都是以桥 MAC 地址为目标的）。
  - **output** - 来自于桥的过滤器包（包括那些被正常路由的数据包）
  - **forward** - 将被桥接的过滤器包（注意：这条链不适用于路由通过路由器的数据包，仅适用于在同一桥的端口间遍历的数据包）。
- **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
- **broute** - 使一个桥变为一个桥路器——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：桥目的网络地址翻译在桥接判定之前执行。

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 (*integer*) - 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 (*integer*) - 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 address*; default: **0.0.0.0/0**) - ARP 目的地址
arp-dst-mac-address (*MAC address*; default: **00:00:00:00:00:00**) - ARP 目的 MAC 地址
arp-hardware-type (*integer*; default: **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 address*; default: **0.0.0.0/0**) – ARP 源 IP 地址

arp-src-mac-address (*MAC address*; default: **00:00:00:00:00:00**) – ARP 源 MAC 地址

chain (*text*) - 过滤器工作其中的桥防火墙链（内置或用户定义的）

dst-address (*IP address*; default: **0.0.0.0/0**) – 目的 IP 地址(仅当 MAC 协议设置为 IPv4 时)

dst-mac-address (*MAC address*; default: **00:00:00:00:00:00**) – 目的 MAC 地址

dst-port (*integer*: 0..65535) - 目标端口号或范围（仅对 TCP 或 UDP 协议）

in-bridge (*name*) - 数据包进入的桥接口

in-interface (*name*) - 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 (*name*) – 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 (*integer*/*time*{0,1},*integer*) - 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 (*text*) - 在日志信息之前定义用于打印的前缀

mac-protocol (*integer* | 802.2 | arp | ip | ipv6 | ipx | rarp | vlan) - 以太网有效负载类型(MAC 等级协议)

mark-flow (*name*) - 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 address*; default: **0.0.0.0/0**) – 源 IP 地址(仅当 MAC 协议设置为 IPv4 时)

src-mac-address (*MAC address*; default: **00:00:00:00:00:00**) – 源 MAC 地址
src-port (*integer*: 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 (*integer*: 0..65535) – stp 端口识别
stp-root-address (*MAC address*) – 根桥 MAC 地址
stp-root-cost (*integer*: 0..65535) – 根桥代价
stp-root-priority (*time*: 0..65535) – 根桥优先级
stp-sender-address (*MAC address*) – stp 信息发射机 MAC 地址
stp-sender-priority (*integer*: 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 (*integer*: 0..4095) – VLAN 识别域
vlan-priority (*integer*: 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 (*name*) - 流出桥的接口
out-interface (*name*) - 数据包离开桥的接口

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; 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。
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 (*name*) - 流出桥接口
to-arp-reply-mac-address (*MAC address*) - 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 address*) - 当选中 **action=dst-nat** 时，把目的 MAC 地址加入以太网帧
to-src-mac-address (*MAC address*) - 当选中 **action= src-nat** 时，把源 MAC 地址加入以太网帧

桥路设施

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; default: **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 address*) - 当选中 **action=dst-nat** 时，把目的 MAC 地址加入以太网帧

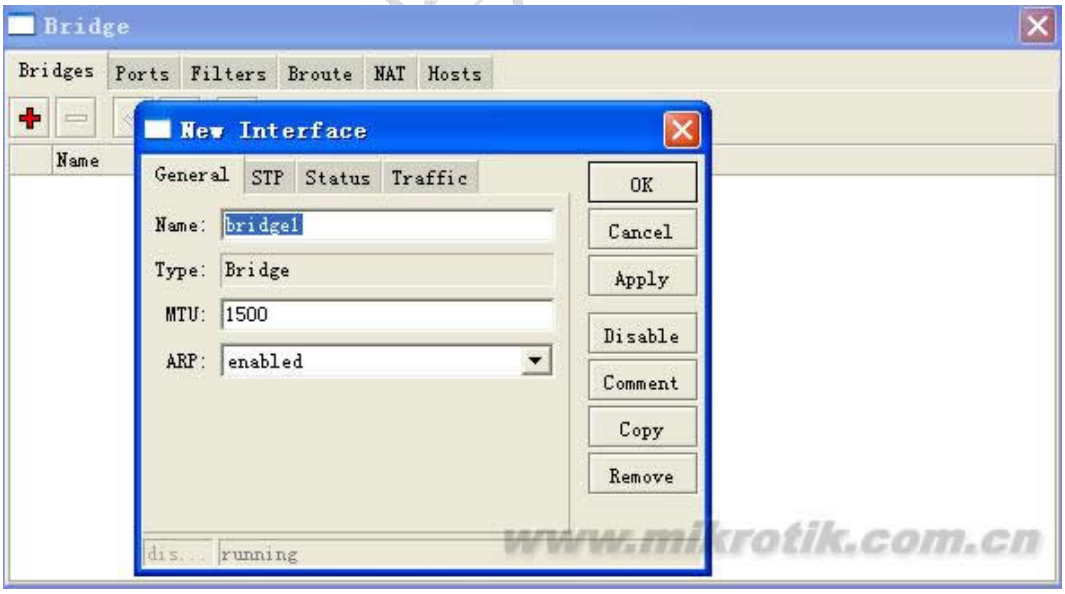
故障分析

- **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

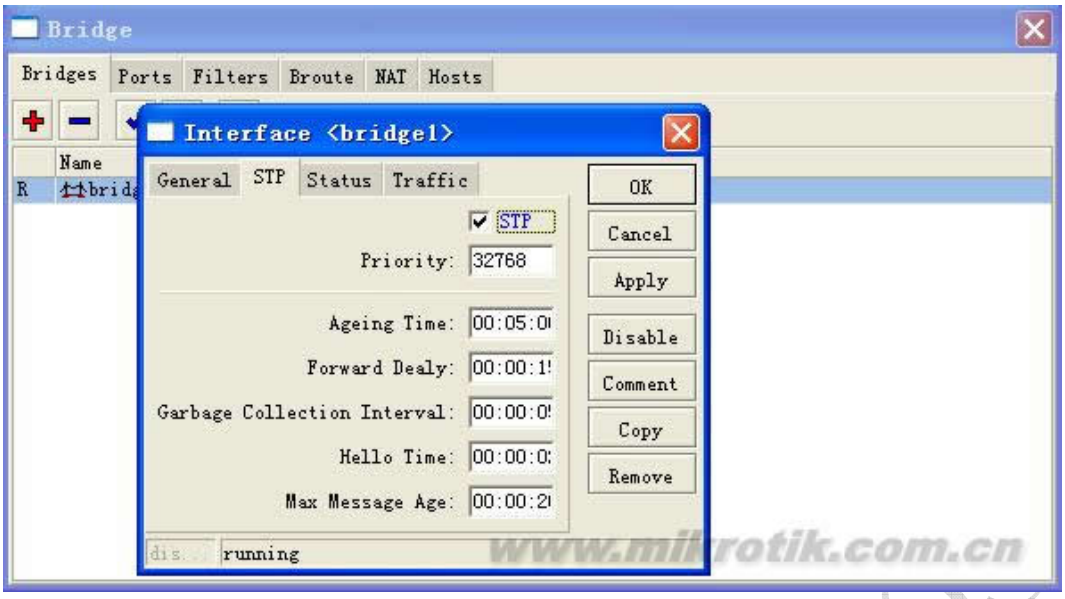
Bridge 实现二层端口隔离

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 RB150 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：



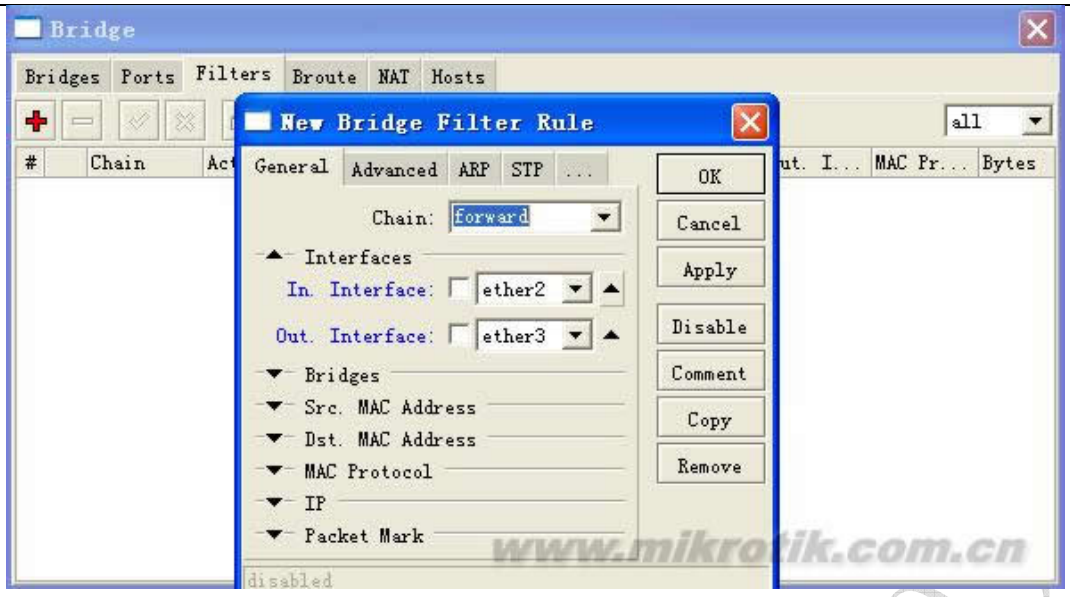
在 RouterOS 同样支持 STP（Spring Tree Protocol）生成树协议，防止二层的回环出现，同样也是支持二层的冗余功能，在这里我们将 STP 打上勾：



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，我们将五个网卡一个一个添加到 bridge1 中：

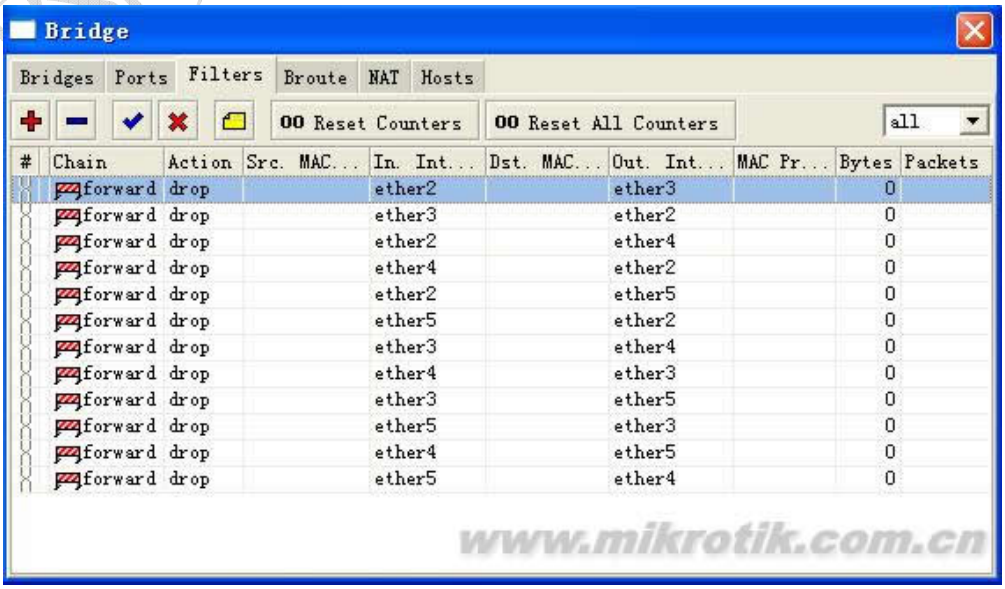
添加完每个端口后，现在 RB150 的 5 个以太网口，就完成了桥接的设置，这样 5 个口就实现了二层的交换功能。

现在我们需要让 ether1 为上联口，即 ether1 能与 ether2、ether3、ether4 和 ether5 进行通信，但 ether2、ether3、ether4 和 ether5 之间是被隔离。 我们进入 filter 中设置防火墙过滤规则，我们首先配置 ether2 与 ether3 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）：



设置好对应的端口后，丢弃他们之间的数据：

因为数据是双向的，上面这条规则是隔离的从 ether2 到 ether3 的数据，但无法封闭 ether3 到 ether2 的数据，所以我们还要做一条反方向的规则，即两个口之间需要做两条规则：



如何建立一个透明传输整形器

属性描述

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
/ping 10.0.0.1
```

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
```

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***
相关协议和标准: VRRP, AH, HMAC-MD5-96 within ESP and AH

属性

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 。

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 (*read-only: flag*) – 是否为备份状态
interface (*name*) – 运行接口的名称
interval (*integer*: 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 (*read-only: flag*) – 是否为 master 状态
mtu (*integer*; 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)

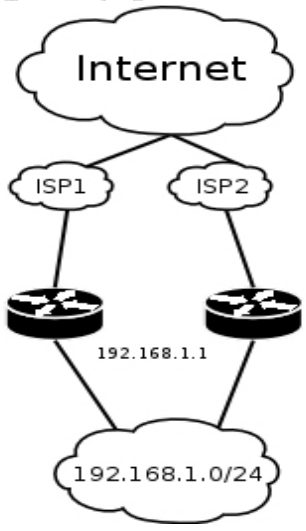
*注：*所有同一个集合的节点，必须使相同的 **vrid**, **interval**, **preemption-mode**, **authentication** 和 **password**.

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>
```

一个简单的 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
```

测试

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>
```

HotSpot 热点认证网关

HotSpot 介绍

HotSpot 是一种通过认证用户来访问某些网络资源的方法。它并不提供流量加密。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软件。网关会计算正常运行时间以及每个客户使用的流量，并且也可以把这个信息发送到 RADIUS 服务器。HotSpot 系统可以限制每个特定用户的比特率，总流量，征程运行时间以及在这个文档涉及的其他参数。

HotSpot 是通过把在一个本地网络（以访问因特网）提供认证作为目标，也可以用于认证来自外网的访问以使用本地资源。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就有了可能。

获取地址

首先，一个客户必须先获得一个 IP 地址。它可以被静态地设置在一个客户上，或者来自一个 DHCP 服务器的分配。如果需要的话，DHCP 服务器可以提供绑定发出的 IP 地址到客户 MAC 地址的途径。

此外，HotSpot 服务器可能会自动地并透明地改变任何客户的 IP 地址，分配一个来自已选 IP 池的合法未使用的地址。这个特性对那些不愿意（或不允许，没有完全资格或其他）改变他们网络设置的移动客户的网络访问提供了可能。用户不会注意到这个转变（例如：在用户配置方面不会有任何改变），但路由器本身则看到完全不同（与每个客户上实际的设置不同）的发送自客户（firewall mangle 表可以“看到”这些转化了的地址）数据包上的 IP 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“通用客户”为人所知。

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于是数据可以被路由通过标准的 IP 网络。客户可以使用任何预先配置的地址。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（它就像更早些完成的源网络地址翻译一样，因此甚至是通常不变地“看到”已接收包的 firewall mangle 列表都只能“看到”翻译后的地址）。

注意 **arp** 模式必须在你使用一对一 NAT 的接口上启用。

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）。其他一些规则也有设计，我们将在该手册后面专门部分进行讲述。

在最普通的设置中，打开任何 HTTP 页面都会产生 HotSpot servlet 登陆页面（可以广泛地自行定义，随后将进行描述）。由于普通用户的行为是通过他们的 DNS 名打开网页，所有一个合法的 DNS 配置必须在 HotSpot 网关本身设定。

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访问你们公司的服务器），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 配置中允许的服务时，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）。当一个用户登陆后，不会受他的影响。

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，所以来自未认证用户的所有请求真正是从这个代理通过的。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，那么符合那个用户的情况将会使用， 否则（如果 RADIUS 响应不包含该用户）默认的概要将用于设置在 RADIUS 访问接受信息中没有设置的参数默认值。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 成为可能。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**

授权

系统将自动探测并对客户所发出的请求重定向到路由器中内嵌的代理服务器上。

认证可以授权给一个传递与本地数据库类似配置选项的 RADIUS 服务器。对任何需要认证的用户，RADIUS 服务器先要进行询问，如果没有收到回应，则会检查本地数据库。RADIUS 服务器会发送一个与标准一致的认证改变请求以改变先前接受的参数。

用户管理

HotSpot 系统内部执行帐目管理，你不需要做任何特别的事情使之工作。每个用户帐目信息都会被发送到一个 RADIUS 服务器。

配置菜单

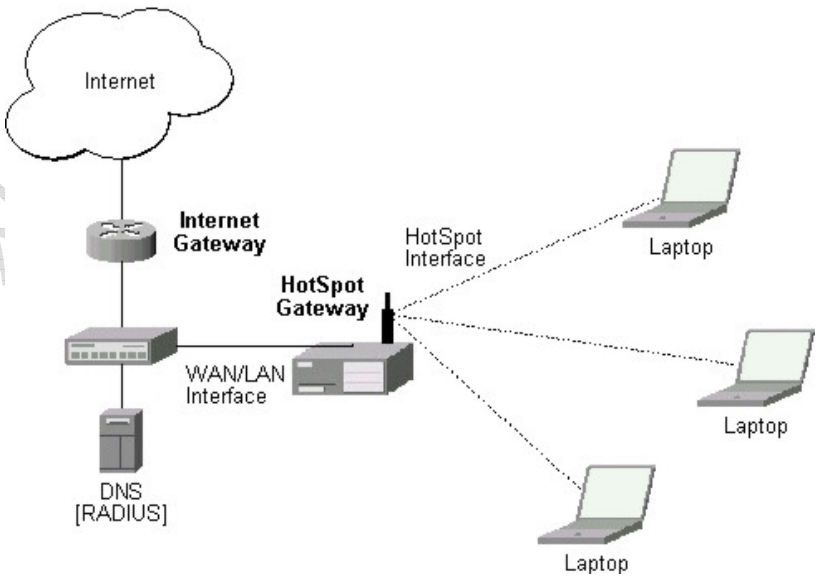
- **/ip hotspot** - HotSpot 上的特定界面(每个界面一个服务器)。HotSpot 服务器必须添加在这个目录中，HotSpot 系统才能够在一个界面上工作。
- **/ip hotspot profile** - HotSpot 服务器概要。影响 HotSpot 客户登陆过程的设置在这里进行。多个 HotSpot 服务器可以使用同样的概要信息。
- **/ip hotspot host** - 所有 HotSpot 接口上的活动网络主机的动态列表。在这里你可以找到 IP 地址与一对一 NAT 的绑定
- **/ip hotspot ip-binding** - 将 IP 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

描述

MikroTik 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。



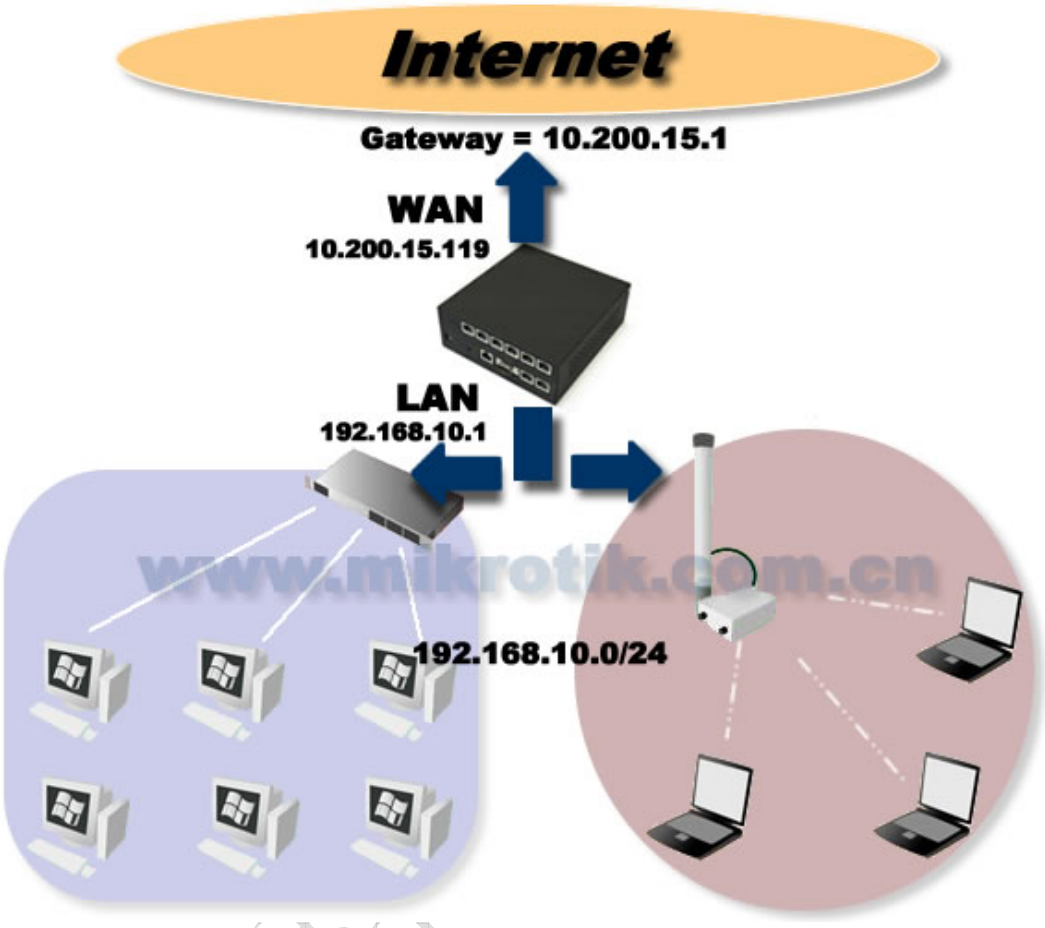
HotSpot 接口应该分配一个 IP 地址。物理网络连接应该建立在 HotSpot 用户的电脑和网关之间。它可以是无线（无线网卡需要在 AP 上注册），或者有线的（NIC 网卡需要连接到一个集线器或一个交换机）。

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。

Hotspot winbox 配置事例

我们根据下面的网络拓扑结构

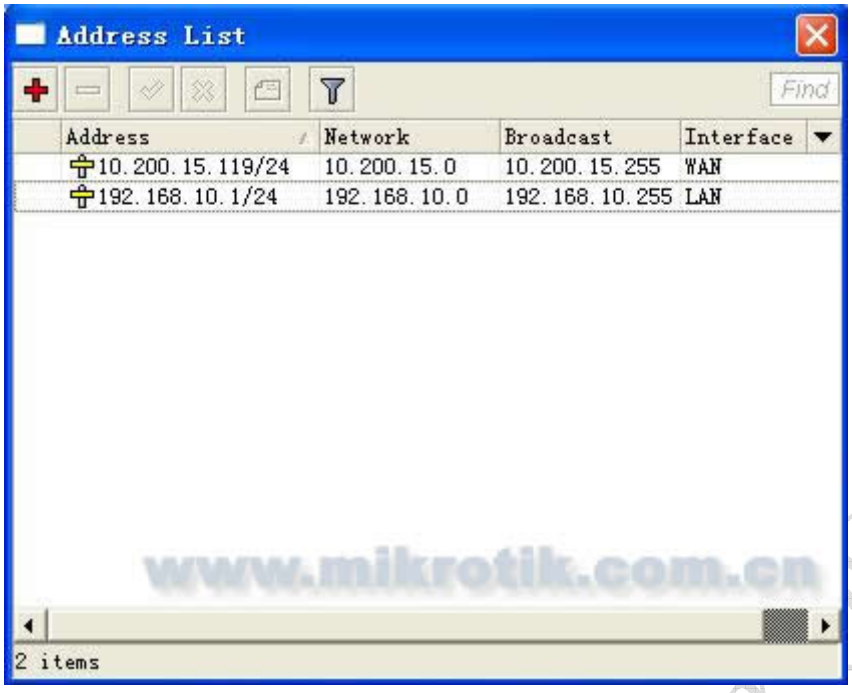


一个网关路由器的网络参数如下：

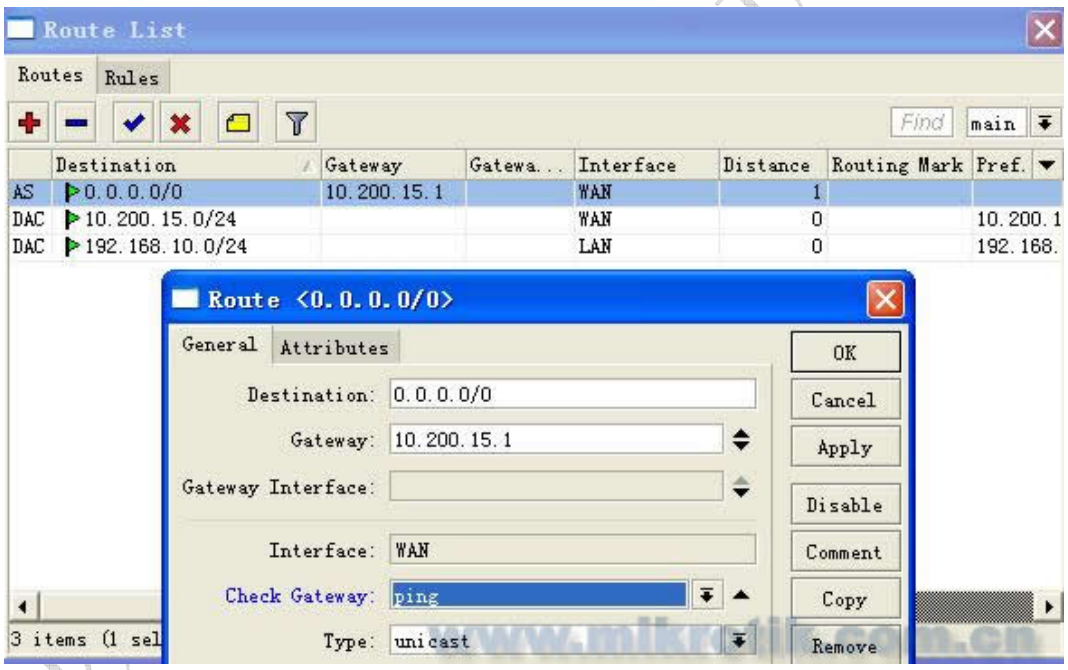
```
WAN 口对应外网 IP 为 10.200.15.119/24，网关为 10.200.15.1
LAN 口对应内网 IP 为 192.168.10.1/24
DNS：61.139.2.69
```

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。

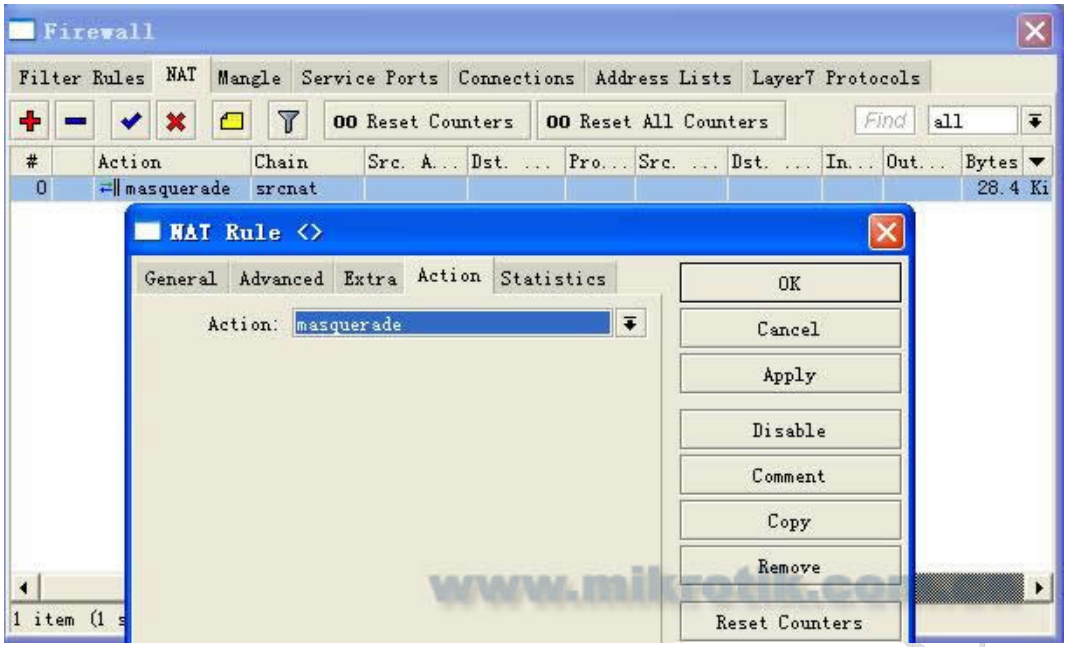
进入 ip address 配置 IP 地址：



进入 ip route 配置网关



进入 ip firewall nat 设置好 NAT 伪装：



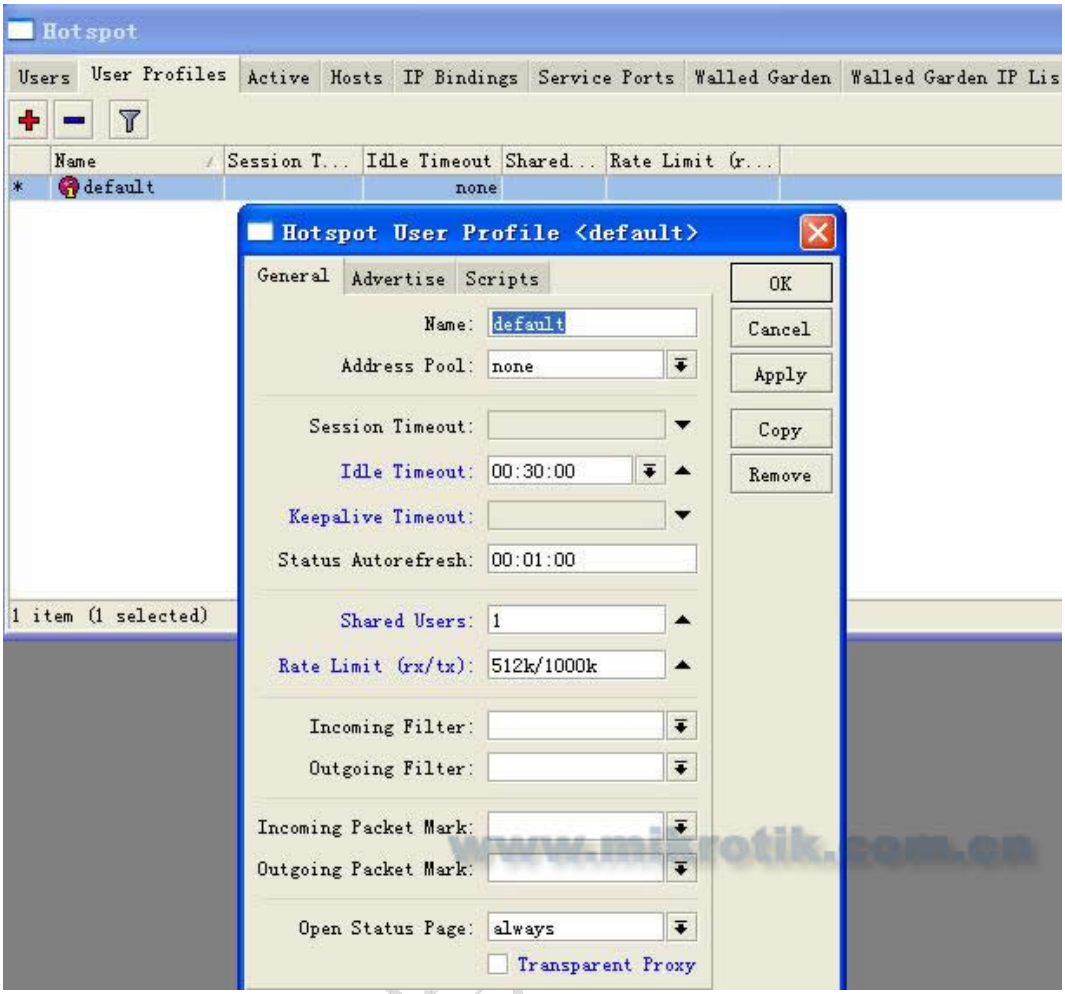
进入 ip dns 配置 DNS 缓存：

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

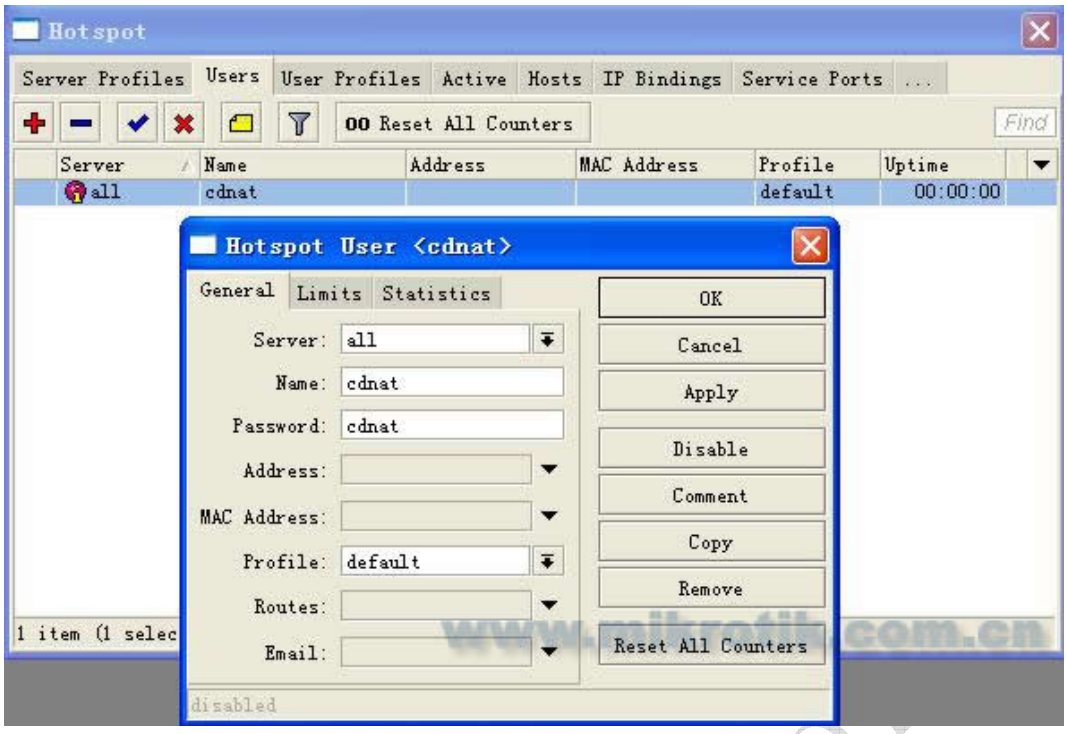


在 user profile 里面一般配置如下几个参数：

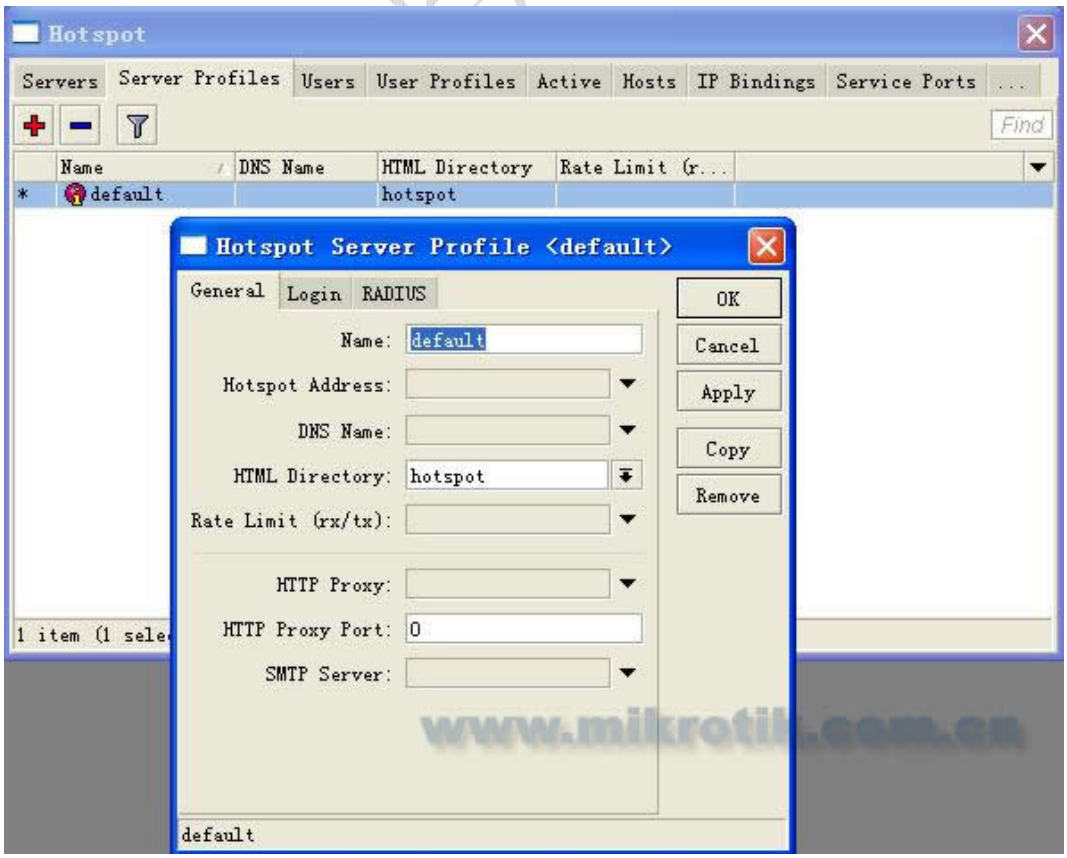
Idle-Timeout：用户在一定时间内没有任何流量发出后自动注销连接
Keepalive-Timeout：路由器主动通过 ICMP 探测主机是否在线，如果在一定时间为探测到自动注销连接（如果用户机开启防火墙，路由器无法探测到）
Shared-users：帐号的分享用户多少，默认为 1，即仅一个用户使用该帐号。
Rate-Limit：分配每个帐号带宽，格式为“上行 / 下行”
Transparent-proxy：透明代理功能是否开启，一般使用 Hotspot 认证建议不用打开此参数。
其他参数请参考具体 Hotspot 手册。

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：

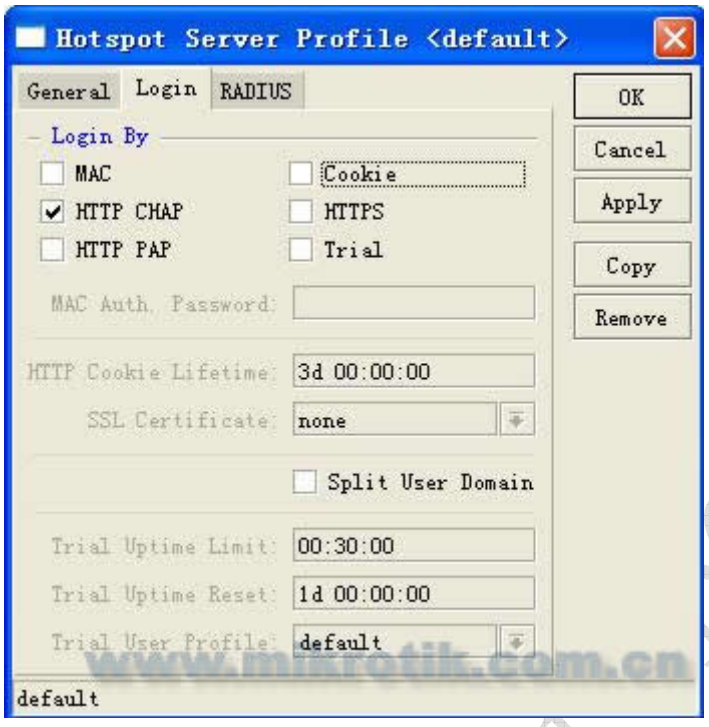


这里默认 server 服务器为 all，
Name 用户名：cdnat
Password：cdnat
Profile：用户组规则，这里选择我们之前设置的 default 规则
配置完用户规则后，进入 ip hotspot server profile，配置服务器器规则：



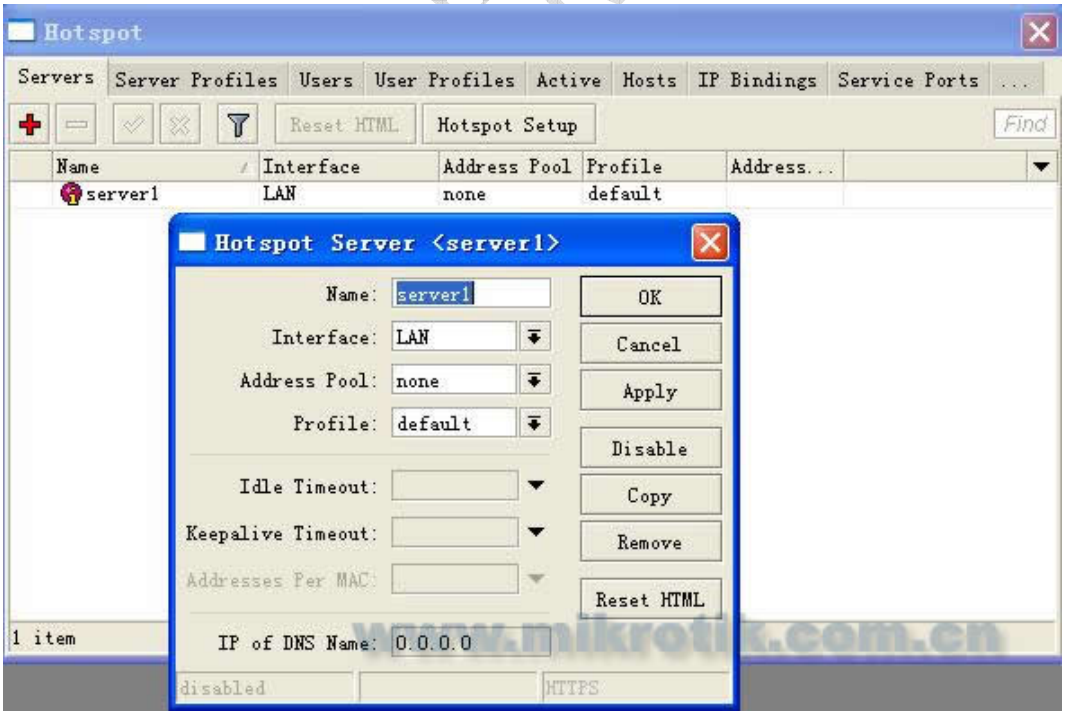
在 General 选项中选择 HTML Directory 为默认的 hotspot 文件路径，同时也可以选择自己定义的文件名路径。

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。



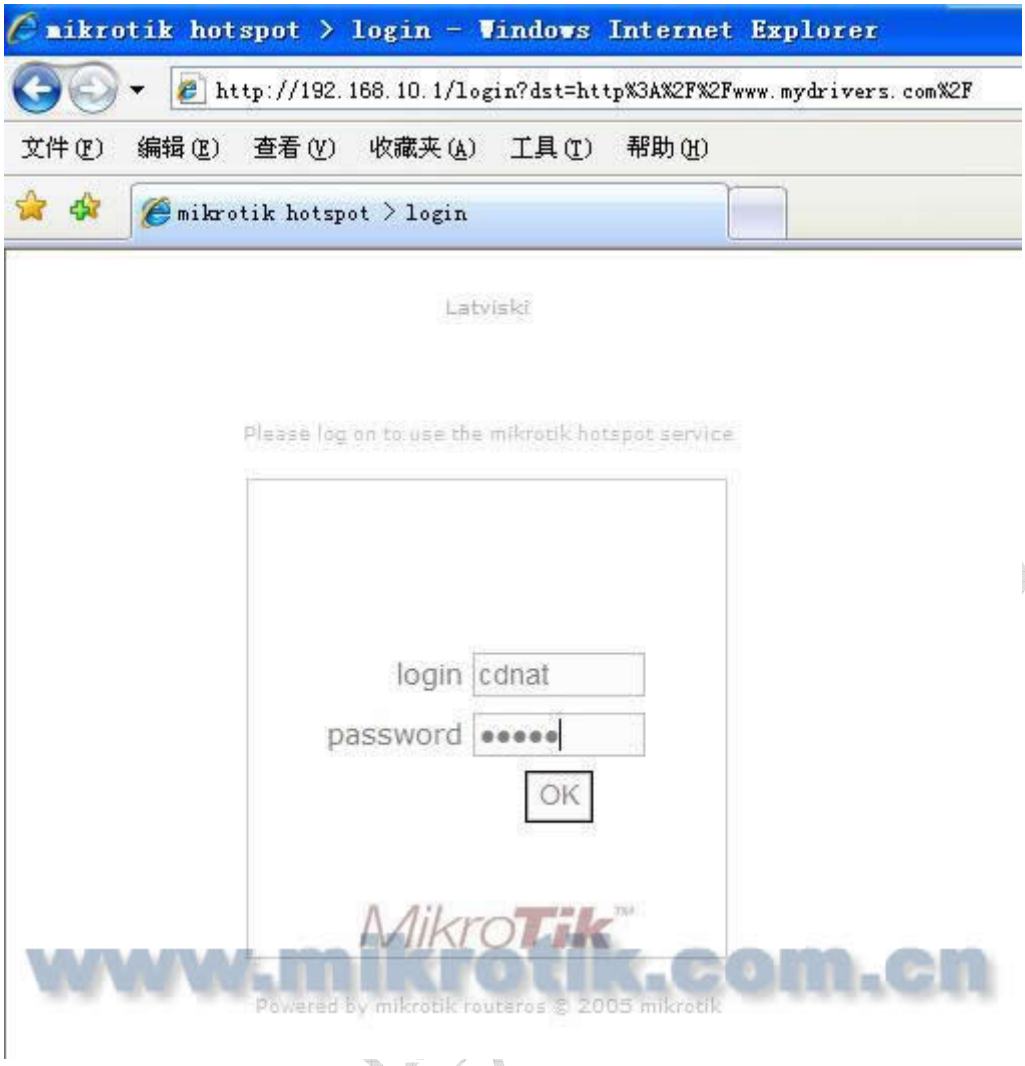
至于 Radius 根据需要开启。

配置完成以上参数后，最后我们启用 Hotspot 服务器：



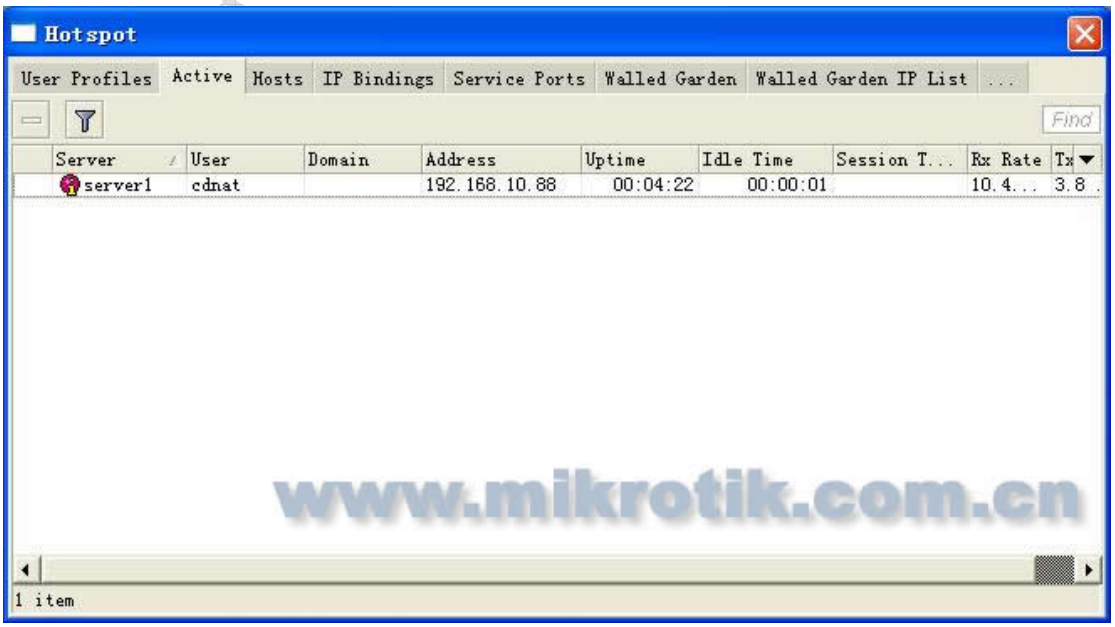
当启用完成后，所有对路由器或者外网访问都需要通过 web 认证，在用户没有认证的情况下，当用户随便输入一个网站都会跳转到认证页面。

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：



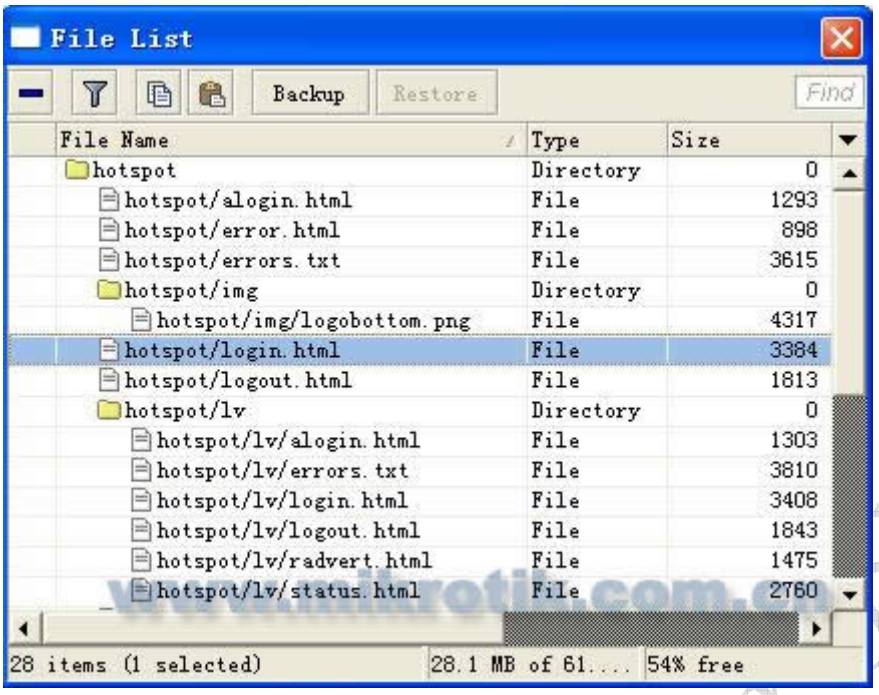
获取现时用户列表：

```
[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”，

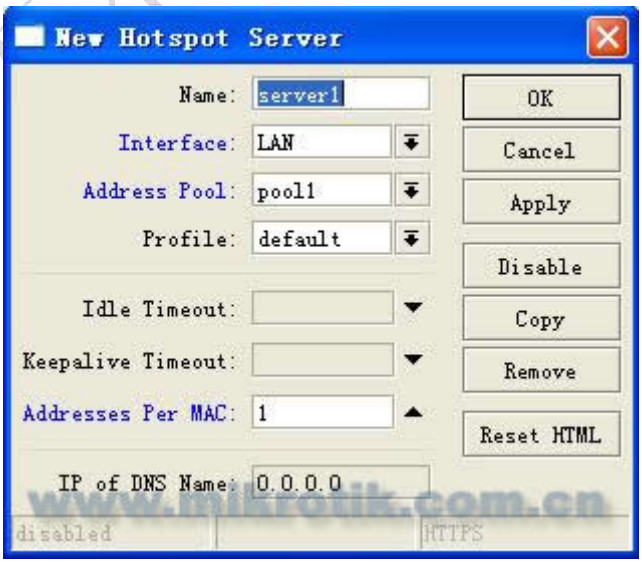


认证页面我们可以通过修改 login.html、logout.html 和 status.html 的 web 界面得到你想要的网页画面或者 log。

启用 Hotspot 的即插即用功能

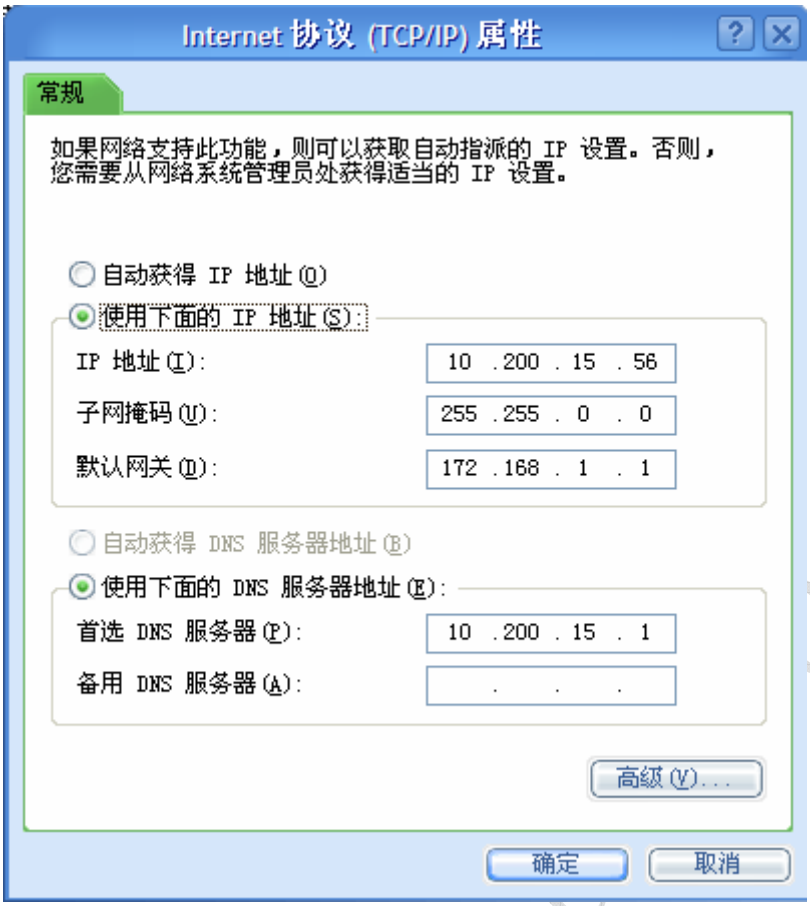
从 2.7 的版本就开始支持 upnp 的即插即用功能，即当用户和 Hotspot 认证服务器在同一局域网内，不管局域网用户设置任何的 IP 地址（前提是用户必须设置任意的 IP 地址、网关和 DNS）都可以被 Hotspot 认证服务器获取，并在 Hotspot 的 Host 中分配一个新的虚拟 IP 地址，并对用户作一对一的 NAT 转换。Hotspot 的即插即用方式分成适用于：流动性较强的公共场所，如机场、车站、公园，也可以应用到酒店和小区中。

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：

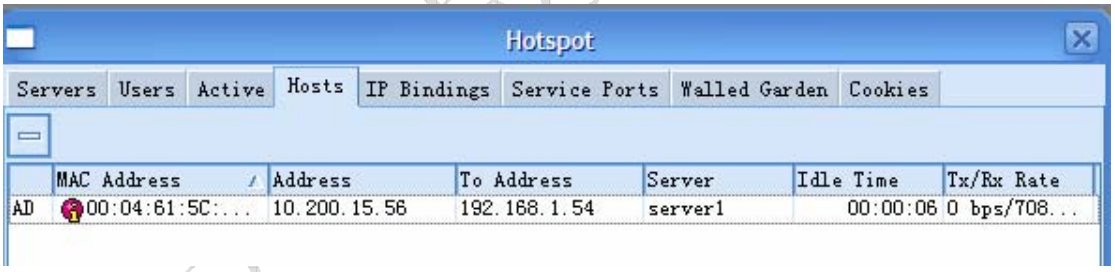


Addresses Per MAC 这个是每个 IP 对应的 MAC 地址，这里我们设置为 1，即一个 IP 对应一个 MAC 地址。

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**

Hotspot 基本设置

命令名：***/ip hotspot setup***

问题

address pool of network (名称) - HotSpot 网络的 IP 地址池

dns name (*文本*) - 网关的 DNS 域名（将会静态地在本地 DNS 代理上配置）

dns servers (*IP 地址*,[*IP address*]) - HotSpot 客户的 DNS 服务器

hotspot interface (名称) - 运行 HotSpot 的接口

ip address of smtp server (*IP 地址*; 默认: **0.0.0.0**) - 重定向 SMTP 请求的 SMTP 服务器的 IP 地址（TCP 端口 25）

- **0.0.0.0** – 无重定向

local address of network (*IP 地址*; 默认: **10.5.50.1/24**) – 接口的 HotSpot 网关地址

masquerade network (yes | no; 默认: **yes**) – 是否伪装 HotSpot 网络

name of local hotspot user (文本; 默认: **admin**) - 自动创建的用户的用户名

passphrase (文本) -输入的认证书的 **passphrase**

password for the user (文本) - 自动创建的用户的密码

select certificate (名称 | none import-other-certificate) - 从已输入的认证列表中选择 SSL 认证

- **none** – 不使用 SSL
- **import-other-certificate** - 设置没有输入的认证，并再次询问这个问题

实例

为了在 ether1 接口上配置 HotSpot（已经配置了地址 192.0.2.1/25），并添加用户 admin 及密码 rubbish：

```
[admin@MikroTik] > ip hotspot setup
hotspot interface: ether1
local address of network: 192.0.2.1/24
masquerade network: yes
address pool of network: 192.0.2.2-192.0.2.126
select certificate: none
ip address of smtp server: 0.0.0.0
dns servers: 192.0.2.254
dns name: hs.example.net
name of local hotspot user: admin
password for the user: rubbish
[admin@MikroTik] >
```

HotSpot 接口设置

操作路径: ***/ip hotspot***

HotSpot 系统应用于一个独立的接口，你可以在不同的接口上配置不同的 HotSpot 服务器。

属性描述

addresses-per-mac (*integer* | unlimited; default: **2**) - 允许与特定 MAC 地址绑定的 IP 地址数量（有很小的机会降低基于接管了所有自由 IP 地址的拒绝服务攻击）

unlimited - 每个 MAC 地址的 IP 地址数量无限制

address-pool (*name* | none; default: **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 (*read-only: 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 (*time* | none; default: **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，没有来自哪个客户的流量也没有流出路由器的流量。达到超时时间后，用户将被主机列表清除，被用户使用的地址也将被释放

none – 不切断空闲用户

interface (*name*) - 运行 HotSpot 的接口

ip-of-dns-name (*read-only: IP address*) - HotSpot 接口概要中设置的 HotSpot 网关 DNS 名称的 IP 地址

keepalive-timeout (*time* | none; default: **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 (*name*; default: **default**) - 接口的默认 HotSpot 概要

reset-html (*name*) - 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。注意当你认证通过后，所有的 IP 地址都会看起来好象来只有一个 MAC 地址。

实例

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>
```

HotSpot Server

操作路径: ***/ip hotspot profile***

属性描述

dns-name (*text*) - 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 (*text*; 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 (*time*; default: **3d**) - HTTP cookies 的有效时间

http-proxy (*IP address*; default: **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 (*multiple choice:* 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 - 对散列的密码使用 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; default: **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; default: **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 (*text*; default: **""**) - 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 address*; default: **0.0.0.0**) - 默认 SMTP 服务器无条件地用于重定向

split-user-domain (yes | no; default: **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 (*name* | none; default: **none**) - 对 HTTPS 认证使用的 SSL 认证名。不用语其他认证方法

trial-uptime (*time*/*time*; default: **30m/1d**) - 仅当认证方式为询问时使用。

trial-user-profile (*name*; default: **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; default: **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。为了使用 RADIUS 认证，**/radius** 目录就必须相符地设置。询问认证方式必须允许与其他认证方法一同使用。

属性描述

domain (*read-only: text*) - 域名(如果从用户名中分离出来的话)

expires-in (*read-only: time*) - cookie 合法存在的时间

mac-address (*read-only: MAC address*) – 用户的 MAC 地址

user (*read-only: name*) – 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 服务概要它是可以改变的，例如：

```
/ip hotspot profile set default http-cookie-lifetime=1d
```

实例

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>
```

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; default: **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 address*) -目的 web 服务器的 IP 地址
dst-host (*wildcard*; default: **""**) - 目的 web 服务器的域名（这是一个通配符）
dst-port (*integer*; default: **""**) -客户发送请求的目的 TCP 端口
method (*text*) – 请求的 HTTP 方法
path (*text*; default: **""**) -请求的路径 (这是一个通配符)
server (*name*) - 应用该规则的 HotSpot 服务器名
src-address (*IP address*) - 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）。

实例

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 - 无需优先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 address*) -目的 web 服务器的 IP 地址
dst-host (*wildcard*; default: **""**) - 目的 web 服务器的域名（这是一个通配符）
dst-port (*integer*; default: **""**) -客户发送请求的目的 TCP 端口
protocol (*integer* | 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 (*name*) - 应用该规则的 HotSpot 服务器名
src-address (*IP address*) - 发送请求的用户 IP 地址

一对一 NAT 静态地址绑定

操作路径: ***/ip hotspot ip-binding***

你可以静态地设置基于源 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：它们可以不必登陆网络就能工作）并完全阻止一些地址。

属性描述

address (*IP address* / [*netmask*]; default: **""**) - 源 IP 地址或客户网络
mac-address (*MAC address*; default: **""**) - 客户的源 MAC 地址
server (*name*|all; default: **all**) - 客户将连接到的服务器名
to-address (*IP address*; default: **""**) - 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 - 执行翻译，但不包含必须登陆 HotSpot 系统的客户
blocked - 不会执行翻译，并且所有来自主机的包都将被丢弃

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。

活动主机列表

操作路径: ***/ip hotspot host***

这个目录显示了所有连接到 HotSpot 网关的活动主机。这个列表包含所有一对一 NAT 翻译。

属性描述

address (*read-only: IP address*) - 客户的原始 IP 地址
authorized (*read-only: flag*) - 客户是否成功地被 HotSpot 系统认证
blocked (*read-only: 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 (*read-only: name*) - 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 (*read-only: flag*) - 是否客户不需要 HotSpot 系统的认证
bytes-in (*read-only: integer*) - 路由器从客户接收的字节数
bytes-out (*read-only: integer*) - 路由器发送到客户的字节数
host-dead-time (*read-only: time*) - 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 (*read-only: time*) – 闲置的时间
idle-timeout (*read-only: time*) - 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 (*read-only: time*) - 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 (*read-only: MAC address*) - 实际的用户 MAC 地址
packets-in (*read-only: integer*) - 路由器接收客户的包数
packets-out (*read-only: integer*) - 路由器发送到客户的数据包数
server (*read-only: name*) - 主机连接到的服务器名
static (*read-only: flag*) - 翻译是否是来自静态 IP 绑定列表
to-address (*read-only: IP address*) - 主机翻译成的原始 IP 地址
uptime (*read-only: time*) - 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 (*name*) – 项目编号
comment (*text*) - 产生客户对静态条目的评论
type (regular | bypassed | blocked) - 静态项目的类型

HotSpot 热点用户管理

基本信息

本文档提供了认证，权限和用户参数以及热点网关系统配置的信息。

规格

功能包要求: ***system***
等级要求: *Level1*
操作路径: ***/ip hotspot user***
标准及技术: RADIUS
硬件使用: 本地的传输记录会添加到内存中

热点用户概要

操作路径: ***/ip hotspot user profile***

热点用户概要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 (*name* | none; 默认值: **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，不过它不使用 DHCP 但嵌入的一对一 NAT。
none - 不再向这个概要中的用户再分配 IP 地址
advertise (yes | no; 默认值: **no**) - 是否对此概要启用强制广告弹出
advertise-interval (*multiple choice: time*; 默认值: **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; 默认值: **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 (*multiple choice: text*; 默认值: **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; 默认值: **none**) - 授权用户空闲超时时间（非活跃状态的最长时间）。它用于探测用户没有使用外部网络（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 (*name*) - 应用于来自此概要用户向内数据包的防火墙链的名称
incoming-packet-mark (*name*) - 自动置于来自此概要每个用户所有数据包的包标记
keepalive-timeout (*time* | none; 默认值: **00:02:00**) - 授权客户的持活超时时间。用于探测改客户的电脑是活跃的并可以到达。如果在这个期间检测失败，那么用户会被登出，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 - 不切断不可达用户
name (*name*) - 概要参考名
on-login (*text*; 默认值: **""**) - 用户登入后运行的脚本名
on-logout (*text*; 默认值: **""**) -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 (*name*) - 应用于此概要用户的向外流出的包的防火墙链名称
outgoing-packet-mark (*name*) - 自动设置在此概要每个用户的所有数据包的包标记
rate-limit (*text*; 默认值: **""**) – 从路由器角度来看的 **rx-rate[/tx-rate] [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*; 默认值: **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 (*integer*; 默认值: **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; 默认值: **none**) – 热点 servlet 状态页面自动刷新间隔

transparent-proxy (yes | no; 默认值: **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。

热点用户

操作路径: ***/ip hotspot user***

属性描述

address (*IP address*; 默认值: **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 (*read-only: integer*) - 接收用户的总字节数

bytes-out (*read-only: integer*) - 发送给用户的总字节数

limit-bytes-in (*integer*; 默认值: **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 (*integer*; 默认值: **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 (*time*; 默认值: **0s**) - 用户的总正常运行时间限制

0s – 无限制

mac-address (*MAC address*; 默认值: **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 (*name*) - 用户名

packets-in (*read-only: integer*) - 接收到用户的最大包数量

packets-out (*read-only: integer*) - 发送给用户的最大包数

password (*text*) - 用户口令

profile (*name*; 默认值: **默认值**) – 用户资料

routes (*text*) - 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 (*name* | all; 默认值: **all**) - 该用户允许登陆的服务器

uptime (*read-only: 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。

实例

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>
```

热点认证在线用户

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 (*read-only: IP address*) - 用户的 IP 地址
blocked (*read-only: flag*) - 是否以广告将用户阻挡（例如：通常适当的广告未决）。
bytes-in (*read-only: integer*) - 路由器从客户收到的字节数
bytes-out (*read-only: integer*) - 由器发送到客户的字节数
domain (*read-only: text*) - 用户范围（如果从用户名中分离出来）
idle-time (*read-only: time*) - 用户被闲置的时间
idle-timeout (*read-only: time*) -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 (*read-only: time*) - 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 (*read-only: integer*) - 用户被允许发送给路由器的最大字节数
limit-bytes-out (*read-only: integer*) - 路由器被允许发送到客户的最大字节数

login-by (*multiple choice, read-only:* cookie | http-chap | http-pap | https | mac | trial) - 用户使用的认证方法

mac-address (*read-only: MAC address*) – 用户的实际 MAC 地址

packets-in (*read-only: integer*) - 路由器接受来自客户的包数量

packets-out (*read-only: integer*) - 路由器发送给客户的包数量

radius (*read-only:* yes | no) - 用户是否通过 RADIUS 认证

server (*read-only: name*) - 用户登陆所指定的服务器

session-time-left (*read-only: 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登出前保持的登入状态时间

uptime (*read-only: time*) -当前用户的会话时间 (例如：用户登入的时间)

user (*read-only: name*) - 用户名

HotSpot 防火墙部分

描述

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流量都将会与流量通过路由器一样的方式被处理。

```
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
```

不像仅有 TCP 协议相关的 Walled Garden 条目被添加的 NAT 表格，在包过滤器中 **hs-unauth** 链会添加任何你在**/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。

EoIP 隧道

基本信息

EoIP（Ethernet over IP）隧道是一个建立了在 IP 连接上两个路由器间的以太网隧道的 MikroTik RouterOS 协议。EoIP 接口表现的如同以太网接口。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）之间有物理以太网接口和光纤一样。这个协议使得多重网络方案成为可能。

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: *Level1 (limited to 1 tunnel)* , *Level3*
操作路径: ***/interface eoip***

标准与技术: GRE (RFC1701)
硬件使用: *Not significant*

描述

EoIP 接口应该在有 IP 等级连接可能的两个路由器上配置。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。

EoIP 配置

操作路径: ***/interface eoip***

属性描述

arp (disabled | enabled | proxy-arp | reply-only; default: **enabled**) -地址解析协议
mac-address (*MAC address*) - 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 (*integer*; default: **1500**) -最大传输单元。默认值提供了最大的兼容性
name (*name*; default: **eoip-tunnelN**) - 作为参考的接口名
remote-address - EoIP 隧道 IP 地址的另一端——必须是 MikroTik 路由器
tunnel-id (*integer*) - a unique tunnel identifier

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再分段存储（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。

实例

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>
```

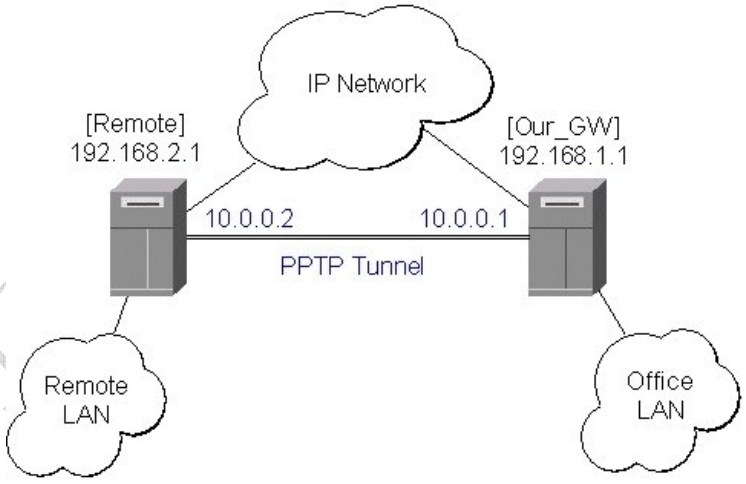
EoIP 应用实例

描述

我们假设要桥接两个网络：'Office LAN'和'Remote LAN'。网络通过路由器[Our_GW]以及[Remote]连接到一个 IP 网络。IP 网络可以是私有企业网或者因特网。这两个路由器通过这个 IP 网络通信。

实例

我们的目标是创建在路由器和桥之间且两个网络都通过它的一个安全频道。



为了在两个路由器之间创建一个安全的以太网桥，你应该

1. 在他们之间创建一个 PPTP 隧道。Our_GW 将成为 PPTP 服务器：

```
[admin@Our_GW] interface pptp-server> /ppp secret add name=joe service=pptp \
\... password=top_s3 local-address=10.0.0.1 remote-address=10.0.0.2
[admin@Our_GW] interface pptp-server> add name=from_remote user=joe
[admin@Our_GW] interface pptp-server> server set enable=yes
[admin@Our_GW] interface pptp-server> print
Flags: X - disabled, D - dynamic, R - running
  #    NAME             USER       MTU   CLIENT-ADDRESS  UPTIME   ENC...
```

```
 0    from_remote         joe
[admin@Our_GW] interface pptp-server>

The Remote router will be the pptp client:

[admin@Remote] interface pptp-client> add name=pptp user=joe \
\... connect-to=192.168.1.1 password=top_s3 mtu=1500 mru=1500
[admin@Remote] interface pptp-client> enable pptp
[admin@Remote] interface pptp-client> print
Flags: X - disabled, R - running
 0  R name="pptp" mtu=1500 mru=1500 connect-to=192.168.1.1 user="joe"
      password="top_s2" profile=default add-default-route=no

[admin@Remote] interface pptp-client> monitor pptp
     status: "connected"
     uptime: 39m46s
     encoding: "none"

[admin@Remote] interface pptp-client>
```

查阅 PPTP 接口手册获得更多关于设置加密频道的细节。

2. 通过在两个路由器添加 EoIP 隧道接口配置 EoIP 隧道。当对 EoIP 隧道指定变量值时，使用 PPTP 隧道接口的 IP 地址：

```
[admin@Our_GW] interface eoip> add name="eoip-remote" tunnel-id=0 \
\... remote-address=10.0.0.2
[admin@Our_GW] interface eoip> enable eoip-remote
[admin@Our_GW] interface eoip> print
Flags: X - disabled, R - running
 0    name=eoip-remote mtu=1500 arp=enabled remote-address=10.0.0.2 tunnel-id=0
[admin@Our_GW] interface eoip>

[admin@Remote] interface eoip> add name="eoip" tunnel-id=0 \
\... remote-address=10.0.0.1
[admin@Remote] interface eoip> enable eoip-main
[admin@Remote] interface eoip> print
Flags: X - disabled, R - running
name=eoip mtu=1500 arp=enabled remote-address=10.0.0.1 tunnel-id=0
[Remote] interface eoip>
```

3. 在两个路由器上的 EoIP 和以太网接口之间启用桥接：

在 Our_GW 上：

```
[admin@Our_GW] interface bridge> add
```

```
[admin@Our_GW] interface bridge> print
Flags: X - disabled, R - running
 0  R name="bridge1" mtu=1500 arp=enabled mac-address=00:00:00:00:00:00 stp=no
    priority=32768 ageing-time=5m forward-delay=15s
    garbage-collection-interval=4s hello-time=2s max-message-age=20s

[admin@Our_GW] interface bridge> add bridge=bridge1 interface=eoip-remote
[admin@Our_GW] interface bridge> add bridge=bridge1 interface=office-eth
[admin@Our_GW] interface bridge> port print
Flags: X - disabled, I - inactive, D - dynamic
 #    INTERFACE     BRIDGE  PRIORITY PATH-COST
 0    eoip-remote    bridge1 128     10
 1    office-eth     bridge1 128     10
[admin@Our_GW] interface bridge>
```

同理，对 Remote：

```
[admin@Remote] interface bridge> add
[admin@Remote] interface bridge> print
Flags: X - disabled, R - running
 0  R name="bridge1" mtu=1500 arp=enabled mac-address=00:00:00:00:00:00 stp=no
    priority=32768 ageing-time=5m forward-delay=15s
    garbage-collection-interval=4s hello-time=2s max-message-age=20s

[admin@Remote] interface bridge> add bridge=bridge1 interface=ether
[admin@Remote] interface bridge> add bridge=bridge1 interface=eoip-main
[admin@Remote] interface bridge> port print
Flags: X - disabled, I - inactive, D - dynamic
 #    INTERFACE     BRIDGE  PRIORITY PATH-COST
 0    ether         bridge1 128     10
 1    eoip-main     bridge1 128     10
[admin@Remote] interface bridge> port print
```

4. 来自同一网络的地址既可以用于 Office LAN 又可以用于 Remote LAN。

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——它们不应该一样！

PPTP

基本信息

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 (limited to 1 tunnel)* , *Level3 (limited to 200 tunnels)* , *Level5*
操作路径: ***/interface pptp-server***, ***/interface pptp-client***
标准与技术: PPTP (RFC 2637)
硬件使用: *Not significant*

描述

PPTP 是使用 PPP 传输 IP 流的安全隧道。PPTP 把 PPP 封装在 IP 运行上的虚拟线路。PPTP 包含 PPP 以及 MPPE（Microsoft 点对点加密）来加密连接。这个协议的目的是做在路由器之间的也可以是路由器和 PPTP 客户间的被妥善管理的安全连接（客户可达并且/或者包含在几乎所有操作系统中，包括 Windows）。

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 流按照因特网号码分配管理机构(IANA)的指定使用 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 用的流量和协议 47 流量以路由通过防火墙或路由器以便被大多数防火墙和路由器使用。

PPTP 连接可能被限制或不可能设置通过一个伪装了的/NAT 的 IP 连接。请参考 Microsoft 和这部分结尾的 RFC 连接以获得更多信息。

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 (*multiple choice:* 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 (*integer*; 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 (*integer*; 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 (*name*; default: **pptp-outN**) - 参考接口名
password (*text*; default: **""**) -当登陆远程服务器时用户的密码
profile (*name*; default: **default**) -当连接到远程服务器时使用的概要简介
user (*text*) - 当登陆到远程服务器时使用的用户名

实例

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
```

监视 PPTP 客户

命令名: ***/interface pptp-client monitor***

属性描述

encoding (*text*) -加密及编码（如果非对称，使用‘/’分隔）在该连接中使用
status (*text*) - 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

实例

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>
```

PPTP 服务器设置

操作路径: ***/interface pptp-server server***

描述

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 (*multiple choice:* 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 (*integer*; 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 (*integer*; 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）

实例

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>
```

PPTP 服务器用户

操作路径：***/interface pptp-server***

在 PPTP 服务器配置里有两种类型的条目——静态用户和动态连接。如果用户数据库或 **default-profile** 把它的 **local-address** 和 **remote-address** 设置的正确，那么一个动态连接就可以被建立。当添加了静态用户，默认概要应该剩下其默认值并且只有 PPP 用户（在 **/ppp secret** 中）需要配置。**注意**在两种情况中 PPP 用户都必须做相应的配置。

属性描述

client-address (*IP address*) -显示已连接客户的 IP 地址（不能在这里设置）
encoding (*text*) -在该连接中使用的加密及编码（如果不对称，使用‘/’分隔）
mtu (*integer*) - 客户的 MTU（不能在这里设置）
name (*name*) – 接口名
uptime (*time*) - 显示用户已连接的时间
user (*name*) - 静态配置或动态添加用户的名称

实例

为 **ex1** 用户添加一个动态条目：

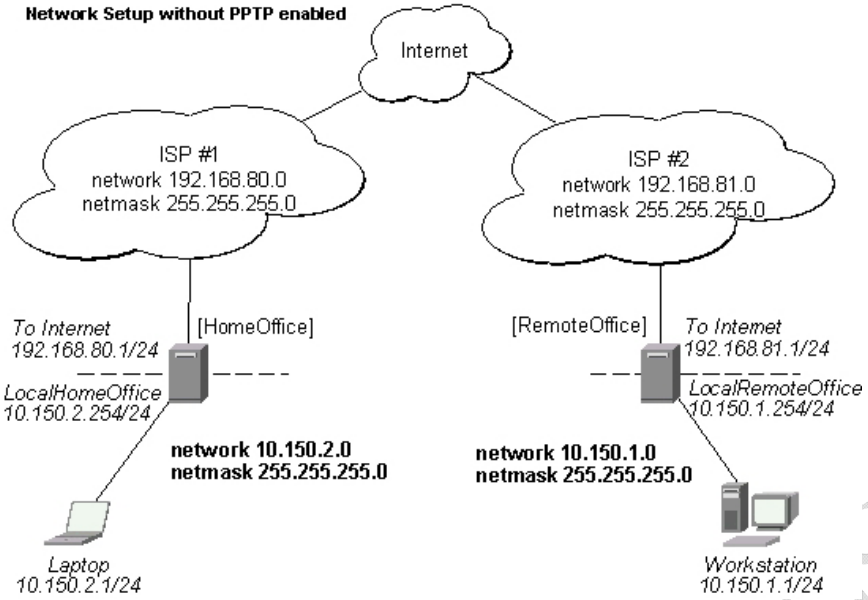
```
[admin@MikroTik] interface pptp-server> add user=ex1
[admin@MikroTik] interface pptp-server> print
Flags: X - disabled, D - dynamic, R - running
  #    NAME              USER        MTU   CLIENT-ADDRESS  UPTIME   ENC...
  0  DR <pptp-ex>          ex          1460  10.0.0.202     6m32s    none
  1    pptp-in1           ex1
[admin@MikroTik] interface pptp-server>
```

在这个例子中除了我们刚添加的一个用户还要显示一个已连接的用户 **ex**。

PPTP 应用实例

Router-to-Router 安全隧道实例

以下是一个使用因特网上的加密 PPTP 隧道连接两个企业网的例子：



在这个例子中有两个路由器：

- [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

每个路由器连接到一个不同的 ISP。任何一个路由器可以通过因特网访问其他的路由器。

在 Preforma PPTP 服务器，用户必须对客户设置：

```
[admin@HomeOffice] ppp secret> add name=ex service=pptp password=lkjrht
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="lkjrht" profile=default
     local-address=10.0.103.1 remote-address=10.0.103.2 routes==""

[admin@HomeOffice] ppp secret>
```

然后应该在 PPTP 服务器列表中添加用户：

```
[admin@HomeOffice] interface pptp-server> add user=ex
[admin@HomeOffice] interface pptp-server> print
Flags: X - disabled, D - dynamic, R - running
 #    NAME              USER       MTU   CLIENT-ADDRESS  UPTIME   ENC...
 0    pptp-in1           ex
```

```
[admin@HomeOffice] interface pptp-server>
```

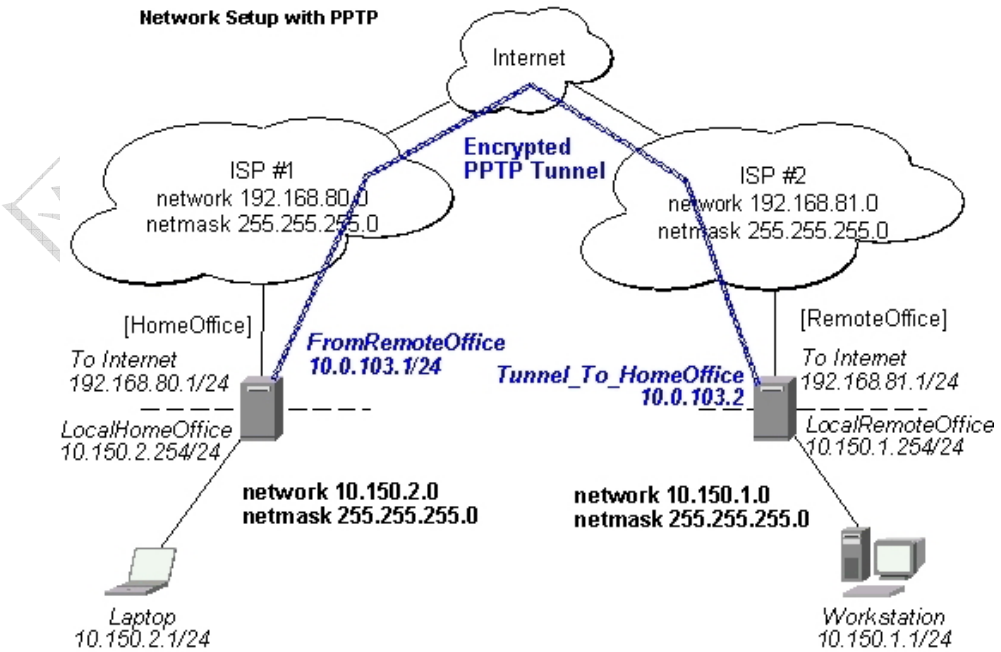
最后，启用服务器：

```
[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>
```

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=lkjrht 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="lkjrht" profile=default add-default-route=no

[admin@RemoteOffice] interface 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 的路由器之间的以太网点对点连接。它使得在第三网络部分上的路由器之间能够直接通信。



为了在 PPTP 隧道上路由本地企业网你需要添加以下路由：

```
[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
```

在 PPTP 服务器上它或者可以通过使用用户配置的 **routes** 参数完成：

```
[admin@HomeOffice] ppp secret> print detail
Flags: X - disabled
 0   name="ex" service=pptp caller-id="" password="lkjrht" 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="lkjrht" 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>
```

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：

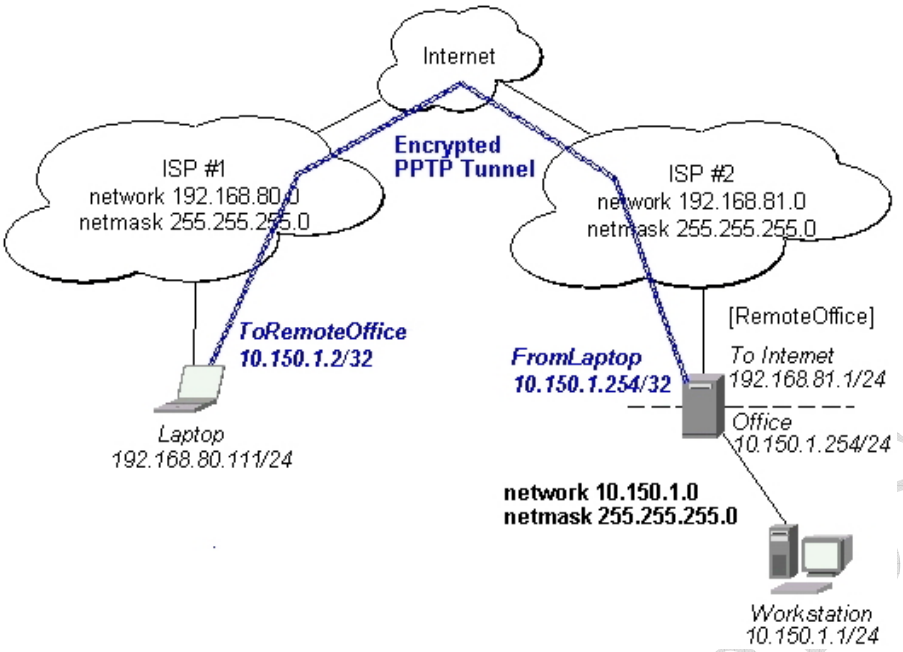
```
[admin@RemoteOffice]> /ping 10.150.2.254
10.150.2.254 pong: ttl=255 time=3 ms
10.150.2.254 pong: ttl=255 time=3 ms
10.150.2.254 pong: ttl=255 time=3 ms
ping interrupted
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 3/3.0/3 ms
```

要在这个安全隧道上桥接一个 LAN，请参考 EoIP 部分手册的例子。要想对该隧道上的流量设置最大速度，请查询 Queues 部分。

通过 PPTP 隧道连接一个远程客户

下面的例子显示了如果通过给定电脑和远程办公网络同一网络 IP 地址的 PPTP 加密隧道把一个电脑连接到一个远程办公网络（不需要在 EoIP 隧道上桥接）

请查询如何设置一个你使用的软件的 PPTP 客户的手册。



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

客户电脑可以通过因特网访问路由器：

在 PPTP 服务器上设置用户帐号：

```
[admin@RemoteOffice] ppp secret> add name=ex service=pptp password=lkjrht
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="lkjrht" profile=default
     local-address=10.150.1.254 remote-address=10.150.1.2 routes==""

[admin@RemoteOffice] ppp secret>
```

然后应该在 PPTP 服务器列表中添加用户：

```
[admin@RemoteOffice] interface pptp-server> add name=FromLaptop user=ex
[admin@RemoteOffice] interface pptp-server> print
Flags: X - disabled, D - dynamic, R - running
 #    NAME              USER       MTU   CLIENT-ADDRESS  UPTIME   ENC...
 0    FromLaptop         ex
[admin@RemoteOffice] interface pptp-server>
```

并且启用服务：

```
[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>
```

最后，代理 ARP 必须在'Office'接口上启用：

```
[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>
```

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，IP 协议 47 应该通过。

IPsec

基本信息

规格

功能包要求: ***security***
等级要求: *Level1*
子目录要求: ***/ip ipsec***
标准与技术: IPsec
硬件使用: *consumes a lot of CPU time (Intel Pentium MMX or AMD K6 suggested as a minimal configuration)*]

描述

IPsec (IP 安全)支持 IP 网络上的安全（加密的）通信。

加密

在数据包 src-natted 之后，但在把它放在接口队列之前，询问 IPsec 策略数据库以搞清楚是否数据包应该加密。安全策略数据库（SPD）是包含两部分的规则列表：

- **Packet matching** -把包源/目的，协议及端口（TCP 及 UDP）进行比较在策略规则一个接一个的估值。
- **Action** -如果规则匹配杂在规则定义的动作将执行：
  - **accept** - 当作没有 IPsec 的村贼继续接受包
  - **drop** – 丢弃包
  - **encrypt** – 加密包

每个 SPD 规则都可以与判定包加密的参数（密匙，算法，SPI）的一些安全联合（SA）相关。

注意只有当可用的策略规则的 SA 存在时包才加密。当没有合法的策略规则 SA 时通过设置 SPD 规则安全“等级”用户可以控制所发生的一切：

- **use** - 如果没有合法的 SA，发送未加密的包（像接受规则）
- **acquire** -发送未加密包，但要求 IKE 端口监督程序建立新的 SA
- **require** -丢弃包，要求 IKE 端口监督程序建立新的 SA

解密

当本地主机接收到加密的数据包（在 **dst-nat** 和 **input** 过滤器之后）时，合适的 SA 会被查询以解密（使用包源，目的，安全协议以及 SPI 值）。如果没有找到 SA，数据包将被丢弃。如果找到了 SA，数据包将进行解密。然后解密的数据包域会与 SA 连接到的策略规则进行比较。如果包不匹配策略规则则丢弃。如果包解密地很好（或者认证地很好）那么它将会“再被接受一次”——它通过 **dst-nat** 然后再一次路由（路由会发现做什么——或者转发或者本地交付）。

注意在 **forward** 和 **input** 防火墙链之前，在本地注意没有解密的包将通过反转它的匹配规则与 SPD 进行比较。如果 SPD 需要加密（有与匹配 SPD 规则相关的合法 SA），那么丢弃包。这就称为进入策略检查。

因特网密匙交换

因特网密匙交换（IKE）是一个为因特网安全协会和密匙管理协议（ISAKMP）框架提供认证密匙材料的协议。也有其他与 ISAKMP 工作的网密匙交换方案，但是 IKE 是用的最广泛的一种。它们提供主机认证方法和安全协会自动管理（SA）。

大多数时间 IKE 端口监督程序什么也不做。当它激活时有两种可能情况：

- 有一些被策略规则捕获的需要加密或认证的流量，但策略并没有任何 SA。策略通知 IKE 端口监督程序，然后 IKE 端口监督程序初始化到远程主机的连接。
- IKE 端口监督程序响应远程连接。

在两种情况下，同等体建立并执行两个阶段：

- **阶段 1** - 同等体同意它们将在随后的 IKE 消息和认证中使用的算法。用于所有 SA 获取密匙和保护之后的主机间 ISAKMP 交换的键入资料也将生成。
- **阶段 2** -同等体建立一个或多个被 IPsec 使用加密数据的 SA。所有由 IKE 端口监督程序建立的 SA 将有寿命值（限制时间，在这个时间过了之后 SA 将变为非法，或者可以被该 SA 加密的数据量，或者以上二者都有）。

有两个寿命值——软寿命值的和硬寿命值的。当 SA 达到他的软寿命值门限时，IKE 端口监督程序会接受到一个通知并开始另一个阶段 2 交换以取代该 SA 为一个新的。如果 SA 达到了它的硬寿命值，它会被丢弃。

IKE 可以随意地提供一个完美转发掩饰（PFS），它是密匙交换的属性，反过来也意味着对 IKE 它折中了长时期阶段 1 密匙将不会被轻易地允许访问被 SA 在阶段 1 建立保护的所有 IPsec 数据。它意思就是一个附加的键入资料将对每个阶段 2 都被生成。

键入资料的生成是计算的非常昂贵的。例如 modp8192 群的使用可以花去数秒，即使在很快的电脑上。它一般发生在每个阶段 1 交换，且仅发生在任何主机对之间，然后保留很久。PFS 也把这个昂贵的操作添加到每个阶段 2 交换。

Diffie-Hellman MODP 组

Diffie-Hellman (DH)密匙交换协议允许两个没有任何初始公享保密的对象安全地创建一个。支持后面的组合指数（MODP）Diffie-Hellman（也叫“Oakley”）组群：

Diffie-Hellman 组	标准	参考
Group 1	768 bits	RFC2409
Group 2	1024 bits	RFC2409
Group 5	1536 bits	RFC3526

IKE 流量

为了避免一些 IKE 包的问题，找到一些 SPD 规则并用未建立的 SA（这个包或许正在常识建立）进行加密，本地产生的有 UDP 源端口 500 的数据包不会被 SPD 处理。相同地将被本地交付的有 UDP 目标端口 500 的数据包也不会在输入策略检测中处理。

设置过程

为了使 IPsec 与使用 IKE-ISAKMP 的自动键入一起同做，你需要配置 **policy**，**peer** 及 **proposal**（可选的）条目。

For manual keying you will have to configure **policy** and **manual-sa** entries.对于手工键入你要配置 **policy** 和 **manual-sa** 条目。

策略设置

操作路径: ***/ip ipsec policy***

策略列表需要判定是否应该把加密应用到一个数据包上。

属性描述

action (accept | drop | encrypt; default: **accept**) - 指定对匹配规则的数据包采取何种动作
  accept – 通过数据包
  drop – 丢弃数据包
  encrypt - 应用在该策略及它的 SA 中指定的变换
decrypted (*integer*) - 由策略解密的输入数据报的数量
dont-fragment (clear | inherit | set; default: **clear**) - 不分段的 IP 标题域状态
  clear - 清除（复位）域，以便先前标记的**不分段的**包进行分段
  inherit - 不改变域
  set - 设置域，以便每个匹配该规则的包都不被分段
dst-address (*IP address*/*netmask*:*port*; default: **0.0.0.0/32:any**) – 目的 IP 地址
encrypted (*integer*) - 由策略加密的输出的包数量
in-accepted (*integer*) - 由策略通过的没有进行尝试解密的输入包数量
in-dropped (*integer*) - 由策略丢弃的没有进行尝试解密的输入包数量
ipsec-protocols (*multiple choice:* ah | esp; default: **esp**) - 指定你所想要应用到匹配的流上的认证标题和压缩安全有效载荷协议的联合体。AH 在 ESP 之后应用，并且以防隧道模式 ESP 将被应用于隧道模式且 AH 用于传输模式。
level (acquire | require | use; default: **require**) -如果一些对该策略的 SA 不能找到则指定需要做什么：
  use - 跳过变换，不丢弃百也不从 IKE 端口监督程序获取 SA
  acquire - 跳过该变换但从 IKE 端口程序获取 SA
  require -丢弃包但获取 SA
manual-sa (*name*; default: **none**) - 将被用于对该策略创建 SA 的 manual-sa 模板名
none - 不设置手动键入
not-decrypted (*integer*) - 策略尝试解密的输入包数量，但是要丢弃
not-encrypted (*integer*) - 策略尝试加密的输出包数量，但要丢弃
out-accepted (*integer*) - 由策略不做任何尝试加密就通过的输出包的数量
out-dropped (*integer*) - 由策略丢弃的不做任何加密尝试的输出包数量
ph2-state (*read-only:* expired | no-phase2 | established) - 密匙建立过程的指示
  expired -有一些来自先前阶段 2 的剩余物。一般地它与 **no-phase2** 类似
  no-phase2 - 目前没有密匙建立
  estabilished - 合适的 SA 在适当位置且一切都工作良好
proposal (*name*; default: **default**) - 将被 IKE 端口监督程序发送以建立该策略 SA 的提议信息名
protocol (*name* | *integer*; default: **all**) - 协议名或号码
sa-dst-address (*IP address*; default: **0.0.0.0**) – SA 目的 IP 地址
sa-src-address (*IP address*; default: **0.0.0.0**) – SA 源 IP 地址
src-address (*IP address*/*netmask*:*port*; default: **0.0.0.0/32:any**) – 源 IP 地址
tunnel (yes | no; default: **no**) - 指定是否使用隧道模式

注： 所有数据包都是在隧道模式封装为 IPIP，并且他们的新 IP 标题 **src-address** 和 **dst-address** 都设置为这个策略的 **sa-src-address** 和 **sa-dst-address** 值。如果你不使用隧道模式（即你使用传输模式），然后只有那些源和目的地址都和 **sa-src-address** 与 **sa-dst-address** 一样的包才可以被策略处理。传输模式只能与产生于且用于 IPsec 同等

体（建立了安全关联的主机）的包一起工作。要加密网络（或者一个网络和一个主机）之间的流量，你必须使用隧道模式。最好让 **dont-fragment** 为清除的，因为加密的包总比初始的包大因此他们需要分割。

如果你使用 IKE 自动建立 SA，那么在两边的路由器上的策略都必须精确地相互匹配，例如在一个路由器上的 **src-address=1.2.3.0/27** 在另一个上的为 **dst-address=1.2.3.0/28** 则不能运行。在一个路由器上的源地址值必须与另一个路由器的目的地址值相对等，反之亦然。

例子

添加一个加密两个主机(10.0.0.147 及 10.0.0.148)间所有流量的策略，我们需要做如下：

```
[admin@WiFi] ip ipsec policy> add sa-src-address=10.0.0.147 \
\... sa-dst-address=10.0.0.148 action=encrypt
[admin@WiFi] ip ipsec policy> print
Flags: X - disabled, D - dynamic, I - invalid
 0   src-address=10.0.0.147/32:any dst-address=10.0.0.148/32:any protocol=all
     action=encrypt level=require ipsec-protocols=esp tunnel=no
     sa-src-address=10.0.0.147 sa-dst-address=10.0.0.148 proposal=default
     manual-sa=none dont-fragment=clear

[admin@WiFi] ip ipsec policy>
```

查看策略统计，做如下：

```
[admin@WiFi] ip ipsec policy> print stats
Flags: X - disabled, D - dynamic, I - invalid
  0   src-address=10.0.0.147/32:any dst-address=10.0.0.148/32:any
      protocol=all ph2-state=no-phase2 in-accepted=0 in-dropped=0
      out-accepted=0 out-dropped=0 encrypted=0 not-encrypted=0 decrypted=0
      not-decrypted=0

[admin@WiFi] ip ipsec policy>
```

同等体

操作路径: ***/ip ipsec peer***

同等体配置设定用于建立 IKE 端口监督程序（阶段 1 配置）的连接。这个连接会被用于协商密匙与 SA 算法。

属性描述

address (*IP address*/*netmask*:*port*; default: **0.0.0.0/32:500**) -地址前缀。如果远程同等体的地址与这个前缀匹配，那么这个对等体配置会在认证和建立阶段 1 时被使用。如果几个对等体的地址与几个配置条目匹配那么有最具体的那个（例如，有最大掩码的那个）将被使用。

dh-group (*multiple choice:* modp768 | modp1024 | modp1536; default: **esp**) - Diffie-Hellman MODP 群(密码强度)

enc-algorithm (*multiple choice:* des | 3des | aes-128 | aes-192 | aes-256; default: **3des**) - 加密算法。算法以强度的递增顺序命名。

exchange-mode (*multiple choice:* main | aggressive | base; default: **main**) - 根据 RFC 2408，不同的 ISAKMP 阶段 1 交换模式不使用其他模式只使用 main 除非你知道你自己所做的。

generate-policy (yes | no; default: **no**) -允许对不存在的策略建立 SA。这样的策略会在 SA 的寿命时间自动创建。这样它就可能创建 IPsec 安全 L2TP 隧道，或者任何其他远程对等体的 IP 在配置时间未知的设置。

hash-algorithm (*multiple choice:* md5 | sha; default: **md5**) - 散列算法。SHA（安全无用信息运算法则）更强大，但更慢。

lifebytes (*integer*; default: **0**) - 阶段 1 寿命：指定在 SA 丢弃之前可以传送的字节数

0 - SA 过期不应该归咎为字节计数过量

lifetime (*time*; default: **1d**) - 阶段 1 寿命：指定 SA 合法的时间。SA 将在这个时间后丢弃

proposal-check (*multiple choice:* claim | exact | obey | strict; default: **strict**) - 阶段 2 寿命检测逻辑：

claim - 使用最短的建议和配置的寿命时间并通知启动程序

exact - 要求寿命时间相同

obey - 接受启动程序发出的一切

strict - 如果建议的寿命比默认的长，那么拒绝建议值否则接受建议寿命值

secret (*text*; default: **""**) -保密字符串。如果它以'0x'开始，则它被分析为十六进制值

send-initial-contact (yes | no; default: **yes**) - 指定是否发送初始 IKE 信息或等待远程端

注： AES（高级加密标准）加密算法比 DES 快得多，所以推荐如果有可能就使用这个算法类。但是，AES 的速度也是它的缺点，因为它可能潜在地破裂的快一点，所以当你需要安全时使用 AES-256 或者当速度很重要时使用 AES-128。两个同等体必须有相同的加密和认证算法，DH 群和交换模式。一些老硬件可能只支持 DES 和 MD5。

你应该对信任的同等体设置 **generate-policy** 标志为 **yes**，因为对已建立的策略没有验证。为了保护自己不受到危险有害的事件伤害，对所有你不想被加密的网络在策略列表的顶部添加策略 **action=accept**。由于动态策略在列表的底部已经添加了，所以它们不会覆盖你的配置。

实例

以 **secret=gwejimezyfopmekun** 对 **10.0.0.147** 定义新的对等体配置：

```
[admin@WiFi] ip ipsec peer>add address=10.0.0.147/32 \
\... secret=gwejimezyfopmekun
[admin@WiFi] ip ipsec peer> print
Flags: X - disabled
  0   address=10.0.0.147/32:500 secret="gwejimezyfopmekun" generate-policy=no
      exchange-mode=main send-initial-contact=yes proposal-check=obey
      hash-algorithm=md5 enc-algorithm=3des dh-group=modp1024 lifetime=1d
      lifebytes=0

[admin@WiFi] ip ipsec peer>
```

远程对等体统计

操作路径：***/ip ipsec remote-peers***

这个子目录为你提供当前已经与该路由器建立阶段 1 连接的远程对等体的各种统计。注意如果对等体没有在这里出现，那就以为着没有 IPsec 流量与他交换。例如，手动配置的 SA 将不会出现在这里。

属性描述

estabilished (*read-only: text*) -当阶段 1 与对等体建立连接后显示数据和时间
local-address (*read-only: IP address*) – 本地 ISAKMP SA 地址
ph2-active (*read-only: integer*) -与该对等体当前正在进行的阶段 2 协商的数量
ph2-total (*read-only: integer*) - 与该对等体总的阶段 2 协商的数量
remote-address (*read-only: IP address*) – 对等体的 IP 地址
side (*multiple choice, read-only:* initiator | responder) - 显示哪边初始化连接
  initiator - 由该路由器开始的阶段 1 协商
  responder - 由对等体开始的阶段 1 协商
state (*read-only: text*) - 与对等体进行的阶段 1 协商的状态
  estabilished - 普通工作状态

实例

查看当前已建立的 SA：

```
[admin@WiFi] ip ipsec> remote-peers print
  0 local-address=10.0.0.148 remote-address=10.0.0.147 state=established
    side=initiator established=jan/25/2003 03:34:45 ph2-active=0 ph2-total=1
[admin@WiFi] ip ipsec>
```

已安装的 SA

操作路径: ***/ip ipsec installed-sa***

这个设备提供关于已建立的包括密匙的安全关联的信息。

属性描述

add-lifetime (*read-only: time*) - SA 安装计算的软/硬过期时间
auth-algorithm (*multiple choice, read-only:* none | md5 | sha1) - SA 中使用的认证算法
auth-key (*read-only: text*) -十六进制字符串格式的认证密匙
current-addtime (*read-only: text*) - SA 建立时的时间
current-bytes (*read-only: integer*) -由 SA 密码机算法处理的数据量
current-usetime (*read-only: text*) -该 SA 第一次被使用的时间
direction (*multiple choice, read-only:* in | out) – SA 方向
dst-address (*read-only: IP address*) - 来自各自策略的 SA 的目的地址
enc-algorithm (*multiple choice, read-only:* none | des | 3des | aes) - SA 中使用的加密算法
enc-key (*read-only: text*) - 十六进制字符串格式的认证密匙（不适用于 AH SA）
lifebytes (*read-only: integer*) -已处理数据的软/硬过期时间门限
replay (*read-only: integer*) - 以字节呈现的重播窗口大小。这个窗口保护了接受器不受到因拒绝老的或复制的包而引起的重播攻击。
spi (*read-only: integer*) – SA 的 SPI 值，以十六进制格式呈现
src-address (*read-only: IP address*) - 从各自策略获取的 SA 源地址

state (*multiple choice, read-only:* larval | mature | dying | dead) – SA 存活阶段

use-lifetime (*read-only: time*) -从第一次使用 SA 开始计数的软/硬过期时间

实例

样本打印输出如下：

```
[admin@WiFi] ip ipsec> installed-sa print
Flags: A - AH, E - ESP, P - pfs, M - manual
  0 E   spi=E727605 direction=in src-address=10.0.0.148
        dst-address=10.0.0.147 auth-algorithm=sha1 enc-algorithm=3des
        replay=4 state=mature
        auth-key="ecc5f4aee1b297739ec88e324d7cfb8594aa6c35"
        enc-key="d6943b8ea582582e449bde085c9471ab0b209783c9eb4bbd"
        add-lifetime=24m/30m use-lifetime=0s/0s lifebytes=0/0
        current-addtime=jan/28/2003 20:55:12
        current-usetime=jan/28/2003 20:55:23 current-bytes=128
  1 E   spi=E15CEE06 direction=out src-address=10.0.0.147
        dst-address=10.0.0.148 auth-algorithm=sha1 enc-algorithm=3des
        replay=4 state=mature
        auth-key="8ac9dc7ecebfed9cd1030ae3b07b32e8e5cb98af"
        enc-key="8a8073a7afd0f74518c10438a0023e64cc660ed69845ca3c"
        add-lifetime=24m/30m use-lifetime=0s/0s lifebytes=0/0
        current-addtime=jan/28/2003 20:55:12
        current-usetime=jan/28/2003 20:55:12 current-bytes=512
[admin@WiFi] ip ipsec>
```

刷新已安装 SA 列表

命令名：***/ip ipsec installed-sa flush***

有时候在不正确/不完整协商发生后，需要手动地刷新已安装 SA 的列表以便 SA 可以再协商。这个选项由 **flush** 命令提供。

属性描述

sa-type (*multiple choice:* ah | all | esp; default: **all**) - 指定 SA 类型以刷新

ah - 删除 AH 协议，仅限 SA

esp -删除 ESP 协议，仅限 SA

all - 删除 AH 和 ESP 协议，仅限 SA

实例

刷新所有已安装的 SA：

```
[admin@MikroTik] ip ipsec installed-sa> flush
[admin@MikroTik] ip ipsec installed-sa> print
[admin@MikroTik] ip ipsec installed-sa>
```

计数器

操作路径: ***/ip ipsec counters***

属性描述

in-accept (*read-only: integer*) -显示由 **accept** 策略匹配的输入包的数量
in-accept-isakmp (*read-only: integer*) - 显示经过端口 500 通过而没匹配任何规则的 UDP 包数量
in-decrypted (*read-only: integer*) - 显示被成功解密的输入包的数量
in-drop (*read-only: integer*) - 显示由 **drop** 策略匹配的输入包的数量（或者不需要所有必要 SA 的 **level=require** 的 **encrypt** 策略）
in-drop-encrypted-expected (*read-only: integer*) - 显示被 **encrypt** 策略匹配的且因为没有加密而被丢弃的输入包数量
out-accept (*read-only: integer*) -显示由 **accept** 策略匹配的输出包数量（包含默认的"accept all"情况）
out-accept-isakmp (*read-only: integer*) - 显示在源端口 500 本地产生的被通过而没有匹配策略的 UDP 包数量
out-drop (*read-only: integer*) - 显示由 **drop** 策略匹配的输出包的数量(或者没有所有必要 SA 的 **level=require** 的 **encrypt**）
out-encrypt (*read-only: integer*) - 显示成功加密的输出包的数量

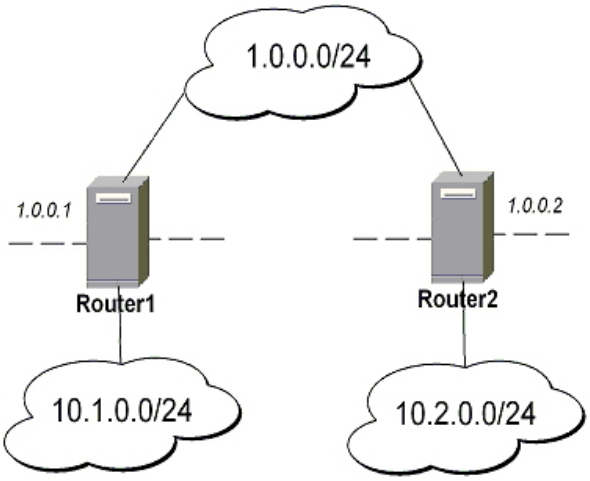
实例

查看当前统计：

```
[admin@WiFi] ip ipsec> counters print
                 out-accept: 6
           out-accept-isakmp: 0
                   out-drop: 0
                out-encrypt: 7
                  in-accept: 12
             in-accept-isakmp: 0
                    in-drop: 0
                in-decrypted: 7
   in-drop-encrypted-expected: 0
[admin@WiFi] ip ipsec>
```

应用实例

MikroTik 路由器到 MikroTik 路由器



- 使用自动键入 ESP 的传输模式例子
  - 对于 **Router1**

```
[admin@Router1] > ip ipsec policy add sa-src-address=1.0.0.1
sa-dst-address=1.0.0.2  action=encrypt
[admin@Router1] > ip ipsec peer add address=1.0.0.2  secret="gvejimezyfopmekun"
```

  - 对于 **Router2**

```
[admin@Router2] > ip ipsec policy add sa-src-address=1.0.0.2
sa-dst-address=1.0.0.1 action=encrypt
[admin@Router2] > ip ipsec peer add address=1.0.0.1  secret="gvejimezyfopmekun"
```

- 使用自动键入 ESP 和 Router 1 上自动生成策略以及 Router 2 上静态策略的传输模式例子
  - 对于 **Router1**

```
[admin@Router1] > ip ipsec peer add address=1.0.0.0/24 \\...
secret="gvejimezyfopmekun" generate-policy=yes
```

  - 对于 **Router2**

```
[admin@Router2] > ip ipsec policy add sa-src-address=1.0.0.2
sa-dst-address=1.0.0.1  action=encrypt
[admin@Router2] > ip ipsec peer add address=1.0.0.1  secret="gvejimezyfopmekun"
```

- 使用手动键入的 AH 的隧道模式例子
  - 对于 **Router1**

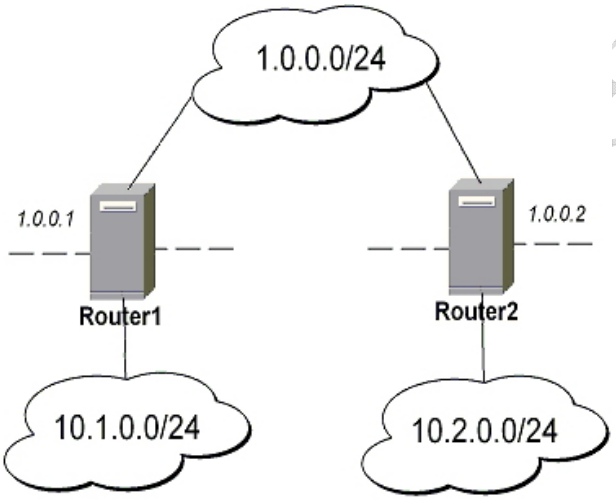
```
[admin@Router1] > ip ipsec manual-sa add name=ah-sa1  ah-spi=0x101/0x100
ah-key=abcfed
```

```
[admin@Router1] > ip ipsec policy add src-address=10.1.0.0/24
dst-address=10.2.0.0/24 action=encrypt ipsec-protocols=ah  tunnel=yes
sa-src=1.0.0.1 sa-dst=1.0.0.2 manual-sa=ah-sa1
```

- 对于 **Router2**

```
[admin@Router2] > ip ipsec manual-sa add name=ah-sa1  ah-spi=0x100/0x101
ah-key=abcfed
[admin@Router2] > ip ipsec policy add src-address=10.2.0.0/24
dst-address=10.1.0.0/24 action=encrypt ipsec-protocols=ah  tunnel=yes
sa-src=1.0.0.2 sa-dst=1.0.0.1 manual-sa=ah-sa1
```

两个伪装的 MikroTik 路由器之间的 IPsec



1. 在 SRC-NAT 中添加 accept 和 masquerading 规则
   - 对于 **Router1**

```
[admin@Router1] > ip firewall nat add src-address=10.1.0.0/24
dst-address=10.2.0.0/24
[admin@Router1] > ip firewall nat add out-interface=public  action=masquerade
```

   - 对于 **Router2**

```
[admin@Router2] > ip firewall nat  add src-address=10.2.0.0/24
dst-address=10.1.0.0/24
[admin@Router2] > ip firewall nat add out-interface=public  action=masquerade
```

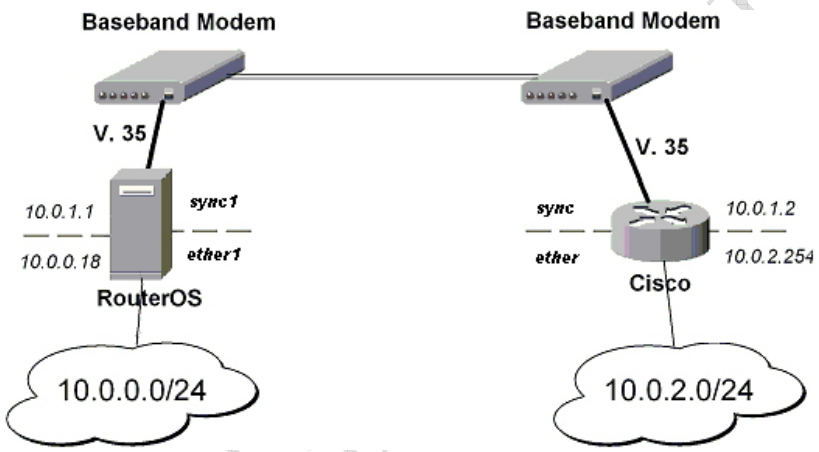
2. 配置 IPsec
   - 对于 **Router1**

```
[admin@Router1] > ip ipsec policy add src-address=10.1.0.0/24
dst-address=10.2.0.0/24 action=encrypt tunnel=yes  sa-src-address=1.0.0.1
sa-dst-address=1.0.0.2
[admin@Router1] > ip ipsec peer add address=1.0.0.2  exchange-mode=aggressive
secret="gvejimezyfopmekun"
```

- 对于 **Router2**

```
[admin@Router2] > ip ipsec policy add src-address=10.2.0.0/24
dst-address=10.1.0.0/24 action=encrypt tunnel=yes  sa-src-address=1.0.0.2
sa-dst-address=1.0.0.1
[admin@Router2] > ip ipsec peer add address=1.0.0.1  exchange-mode=aggressive
secret="gvejimezyfopmekun"
```

MikroTik 路由器到CISCO 路由器



我们将在隧道模式配置 IPsec 以保护相连子网间的流量。

1. 添加对等体（用阶段 1 配置参数），DES 和 SHA1 将被用于保护 IKE 流量
   - 对于 **MikroTik** 路由器

```
[admin@MikroTik] > ip ipsec peer add address=10.0.1.2 secret="gvejimezyfopmekun"
enc-algorithm=des
```

   - 对于 **CISCO** 路由器

```
! Configure ISAKMP policy (phase1 config, must match configuration
! of "/ip ipsec peer" on RouterOS). Note that DES is default
! encryption algorithm on Cisco. SHA1 is default authentication
! algorithm
crypto isakmp policy 9
  encryption des
  authentication pre-share
  group 2
  hash md5
```

```
  exit

! Add preshared key to be used when talking to RouterOS
crypto isakmp key gvejimezyfopmekun address 10.0.1.1 255.255.255.255
```

2. Set encryption proposal (phase2 proposal - 设置加密建议（阶段 2 建议——将用于加密实际数据的设定）以使用 DES 加密数据
   - 对于 **MikroTik** 路由器

```
[admin@MikroTik] > ip ipsec proposal set default enc-algorithms=des
```

   - 对于 **CISCO** 路由器

```
! Create IPsec transform set - transformations that should be applied to
! traffic - ESP encryption with DES and ESP authentication with SHA1
! This must match "/ip ipsec proposal"
crypto ipsec transform-set myset esp-des esp-sha-hmac
  mode tunnel
  exit
```

3. 添加匹配子网间流量并且用在隧道模式用 ESP 加密的策略规则
   - 对于 **MikroTik** 路由器

```
[admin@MikroTik] > ip ipsec policy add  src-address=10.0.0.0/24
dst-address=10.0.2.0/24 action=encrypt  tunnel=yes sa-src=10.0.1.1
sa-dst=10.0.1.2
```

   - 对于 **CISCO** 路由器

```
! Create access list that matches traffic that should be encrypted
access-list 101 permit ip 10.0.2.0 0.0.0.255 10.0.0.0 0.0.0.255
! Create crypto map that will use transform set "myset", use peer 10.0.1.1
! to establish SAs and encapsulate traffic and use access-list 101 to
! match traffic that should be encrypted
crypto map mymap 10 ipsec-isakmp
  set peer 10.0.1.1
  set transform-set myset
  set pfs group2
  match address 101
  exit
! And finally apply crypto map to serial interface:
interface Serial 0
  crypto map mymap
  exit
```

4. 检测 IPsec 隧道

- 在 **MikroTik** 路由器我们可以看到已安装的 SA

```
[admin@MikroTik] ip ipsec installed-sa> print
Flags: A - AH, E - ESP, P - pfs, M - manual
  0 E   spi=9437482 direction=out src-address=10.0.1.1
        dst-address=10.0.1.2 auth-algorithm=sha1 enc-algorithm=des
        replay=4 state=mature
        auth-key="9cf2123b8b5add950e3e67b9eac79421d406aa09"
        enc-key="ffe7ec65b7a385c3" add-lifetime=24m/30m use-lifetime=0s/0s
        lifebytes=0/0 current-addtime=jul/12/2002 16:13:21
        current-usetime=jul/12/2002 16:13:21current-bytes=71896
  1 E   spi=319317260 direction=in src-address=10.0.1.2
        dst-address=10.0.1.1 auth-algorithm=sha1 enc-algorithm=des
        replay=4 state=mature
        auth-key="7575f5624914dd312839694db2622a318030bc3b"
    enc-key="633593f809c9d6af" add-lifetime=24m/30m use-lifetime=0s/0s
    lifebytes=0/0 current-addtime=jul/12/2002 16:13:21
    current-usetime=jul/12/2002 16:13:21 current-bytes=0

    [admin@MikroTik] ip ipsec installed-sa>
```

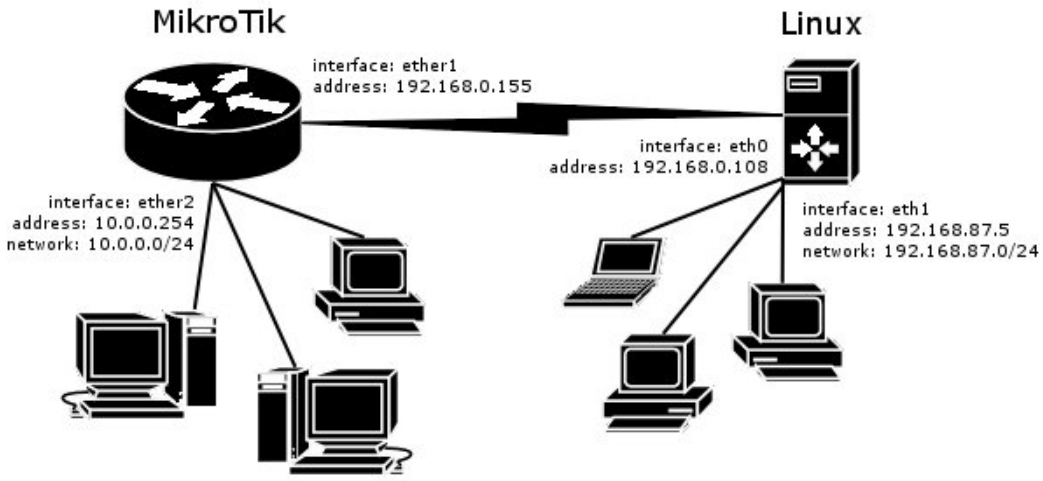
- 在 **CISCO** 路由器

```
cisco# show interface Serial 0
interface: Serial1
   Crypto map tag: mymap, local addr. 10.0.1.2
   local  ident (addr/mask/prot/port): (10.0.2.0/255.255.255.0/0/0)
   remote ident (addr/mask/prot/port): (10.0.0.0/255.255.255.0/0/0)
   current_peer: 10.0.1.1
   PERMIT, flags={origin_is_acl,}
   #pkts encaps: 1810, #pkts encrypt: 1810, #pkts digest 1810
 #pkts decaps: 1861, #pkts decrypt: 1861, #pkts verify 1861
    #pkts compressed: 0, #pkts decompressed: 0
    #pkts not compressed: 0, #pkts compr. failed: 0, #pkts decompress failed: 0
   #send errors 0, #recv errors 0
    local crypto endpt.: 10.0.1.2, remote crypto endpt.: 10.0.1.1
    path mtu 1500, media mtu 1500
    current outbound spi: 1308650C
    inbound esp sas:
    spi: 0x90012A(9437482)
    transform: esp-des esp-sha-hmac ,
    in use settings ={Tunnel, }
    slot: 0, conn id: 2000, flow_id: 1, crypto map: mymap
    sa timing: remaining key lifetime (k/sec): (4607891/1034)
    IV size: 8 bytes
    replay detection support: Y
    inbound ah sas:
```

```
inbound pcp sas:
outbound esp sas:
spi: 0x1308650C(319317260)
transform: esp-des esp-sha-hmac ,
in use settings ={Tunnel, }
slot: 0, conn id: 2001, flow_id: 2, crypto map: mymap
sa timing: remaining key lifetime (k/sec): (4607893/1034)
IV size: 8 bytes
replay detection support: Y
outbound ah sas:
outbound pcp sas:
```

MikroTik 路由器与 Linux FreeS/WAN

在测试设想中我们有两个私有网络：连接到 MT 的 10.0.0.0/24 和连接到 Linux 的 192.168.0.0/24。MT 与 Linux 在公网 192.168.0.0/24 上连接在一起：



- FreeS/WAN 配置：

```
config setup
interfaces="ipsec0=eth0"
klipsdebug=none
plutodebug=all
plutoload=%search
plutostart=%search
uniqueids=yes

 conn %default
 keyingtries=0
 disablearrivalcheck=no
 authby=rsasig

 conn mt
 left=192.168.0.108
 leftsubnet=192.168.87.0/24
```

```
right=192.168.0.155
rightsubnet=10.0.0.0/24
authby=secret
pfs=no
auto=add
```

- **ipsec.secrets** 配置文件：

```
192.168.0.108 192.168.0.155 : PSK "gvejimezyfopmekun"
```

- MikroTik 路由器配置：

```
[admin@MikroTik] > /ip ipsec peer add address=192.168.0.108 \
\... secret="gvejimezyfopmekun" hash-algorithm=md5  enc-algorithm=3des \
\... dh-group=modp1024 lifetime=28800s

[admin@MikroTik] > /ip ipsec proposal auth-algorithms=md5 \
\... enc-algorithms=3des pfs-group=none

[admin@MikroTik] > /ip ipsec policy add sa-src-address=192.168.0.155 \
\... sa-dst-address=192.168.0.108 src-address=10.0.0.0/24 \
\... dst-address=192.168.87.0/24 tunnel=yes
```

PPPoE

基本说明

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。

MikroTik 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。

快速设置向导

配置 MikroTik RouterOS PPPoE 客户端

1.仅添加一个 pppoe-client:

```
/interface pppoe-client add name=pppoe-user-mike user=mike password=123
interface=wlan1  service-name=internet disabled=no
```

配置 MikroTik RouterOS 去访问集线服务器(PPPoE Server)

2. 为客户端添加一个地址池从 **10.1.1.62** to **10.1.1.72** 并命名为“pppoe-pool”：

```
/ip pool add name="pppoe-pool" ranges=10.1.1.62-10.1.1.72
```

3. 添加 PPP 策略，命名为 **pppoe-profile，local-address** 将是访问本地路由器的地址，客户端的地址来至 **pppoe-pool**:

```
/ppp profile add name="pppoe-profile" local-address=10.1.1.1
remote-address=pppoe-pool
```

4. 添加用户名 **mike** 和密码 Add **123**：

```
/ppp secret add name=mike password=123 service=pppoe profile=pppoe-profile
```

5.现在添加 pppoe 服务器设置：

```
/interface pppoe-server server add service-name=internet interface=wlan1
default-profile=pppoe-profile
```

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），最大支持 65535 个连接。*

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 (*multiple choice:* 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

事例

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。

事例

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>
```

PPPoE Server 设置 (Access Concentrator)

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 (*name*; 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

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。

事例

将 PPPoE 服务器添加到 **ether1** interface 上，并提供 **ex** 服务名，每次仅允许一个主机连接:

```
[admin@MikroTik] interface pppoe-server server> add interface=ether1 \
\... service-name=ex one-session-per-host=yes
[admin@MikroTik] interface pppoe-server server> print
Flags: X - disabled
  0 X service-name="ex" interface=ether1 mtu=1480 mru=1480
      authentication=mschap2,mschap,chap,pap keepalive-timeout=10
      one-session-per-host=yes default-profile=default

[admin@MikroTik] interface pppoe-server server>
```

PPPoE 服务器用户

操作路径: **/interface pppoe-server**

属性描述

name (*名称*) – 接口名称
service-name (*名称*) – 用户连接的服务名
remote-address (*MAC 地址*) – 已连接客户端的 MAC 地址
user (*名称*) – 已连接的用户名
encoding (*文本*) – 该连接使用加密和编码
uptime – 显示客户端已连接了多长时间

事例

查看当前连接的用户：

```
[admin@MikroTik] interface pppoe-server> print
Flags: R - running
  #   NAME      SERVICE REMOTE-ADDRESS    USER    ENCO... UPTIME
  0 R <pppoe-ex> ex     00:C0:CA:16:16:A5 ex            12s

[admin@MikroTik] interface pppoe-server>
```

断掉用户名为 **ex** 的连接:

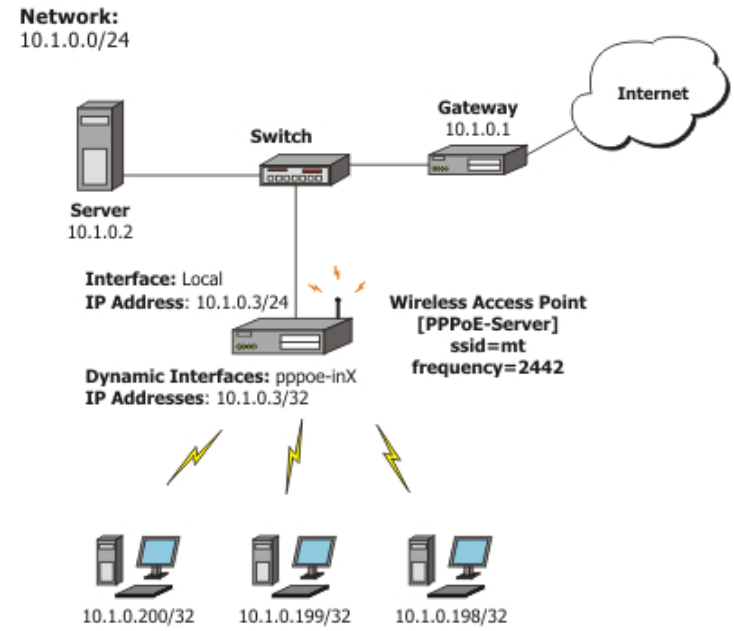
```
[admin@MikroTik] interface pppoe-server> remove [find user=ex]
[admin@MikroTik] interface pppoe-server> print

[admin@MikroTik] interface pppoe-server>
```

应用事例

PPPoE 在一个多点连接的 802.11g 无线网络

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。

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
```

最后不要忘了设置 IP 伪装。

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

VLAN

基本信息

VLAN 是 MikroTik RouterOS 的 802.1Q VLAN 协议的执行。它允许你在单个以太网或无线接口上拥有多个虚拟 LAN，给予了高效分离 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 和基于 Linux 的，以及很多两层交换机也都支持。

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)
硬件使用: *Not significant*

描述

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)

VLAN 设置

操作路径：***/interface vlan***

属性描述

arp (disabled | enabled | proxy-arp | reply-only; default: **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 (*name*) - VLAN 网络的物理接口
mtu (*integer*; default: **1500**) – 最大传输单元
name (*name*) - 参考接口名
vlan-id (*integer*; default: **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>
```

应用实例

MikroTik Routers 上的 VLAN 例子

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>
```

Web 代理

基本信息

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

快速设置向导

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)*

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。

设置

操作路径：***/ip proxy***

属性描述

cache-administrator (*text*; 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 | *integer*: 0..4294967295; default: **none**) -指定最大磁盘缓存大小，以 kb 计算
max-fresh-time (*time*; 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 (*integer*; 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 (*integer*; 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 (*integer*; 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 | *integer*: 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。

实例

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>
```

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 (*read-only: integer*) - 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 (*text*) - 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。

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; default: **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 (*read-only: integer*) -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。

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; default: **allow**) - 指定对已匹配包的动作
  allow - 从匹配的请求缓存对象
  deny – 不从匹配的请求缓存对象
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 (*read-only: integer*) - 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

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 (*read-only: integer*) - 用于缓存的磁盘空间
hits (*read-only: integer*) – 在缓存中找到并开始被服务的请求数
hits-sent-to-clients (*read-only: integer*) – 由缓存服务的数据量
ram-cache-used (*read-only: integer*) – 用于存贮缓存的 RAM 空间
received-from-servers (*read-only: integer*) – 从其他服务器接受的数据量
requests (*read-only: integer*) – 已处理的请求量
sent-to-clients (*read-only: integer*) – 发送到该代理服务器客户的数据量
status (*read-only: text*; 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 (integer) – 最大缓存大小，可以访问 web 代理
total-ram-used (*read-only: integer*) – 用于代理的总 RAM 大小

uptime (*read-only: time*) - 代理最近一次启动后的时间

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 (*read-only: IP address*) – 连接的 IP 地址
protocol (*read-only: text*) – 协议名
rx-bytes (*read-only: integer*) - 客户接收的字节量
src-address (*read-only: IP address*) – 连接源发站的 IP 地址
state (*read-only:* 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 (*read-only: integer*) - 由客户发送的字节数

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 (*read-only: integer*) - 被缓存列表拒绝的插页数
errors (*read-only: integer*) – 磁盘或其他系统相关的错误数量
no-memory (*read-only: integer*) – 由于没有足够内存而没有存贮的对象数量
successes (*read-only: integer*) - 成功缓存插页的数量
too-large (*read-only: integer*) – 过大而不能存储的对象数量

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 (*read-only: integer*) - 被访问列表拒绝的请求数
expired (*read-only: integer*) -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 (*read-only: integer*) - 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 (*read-only: integer*) - 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 (*read-only: integer*) -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 (*read-only: integer*) - 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。

命令描述

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

HTTP 方法

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

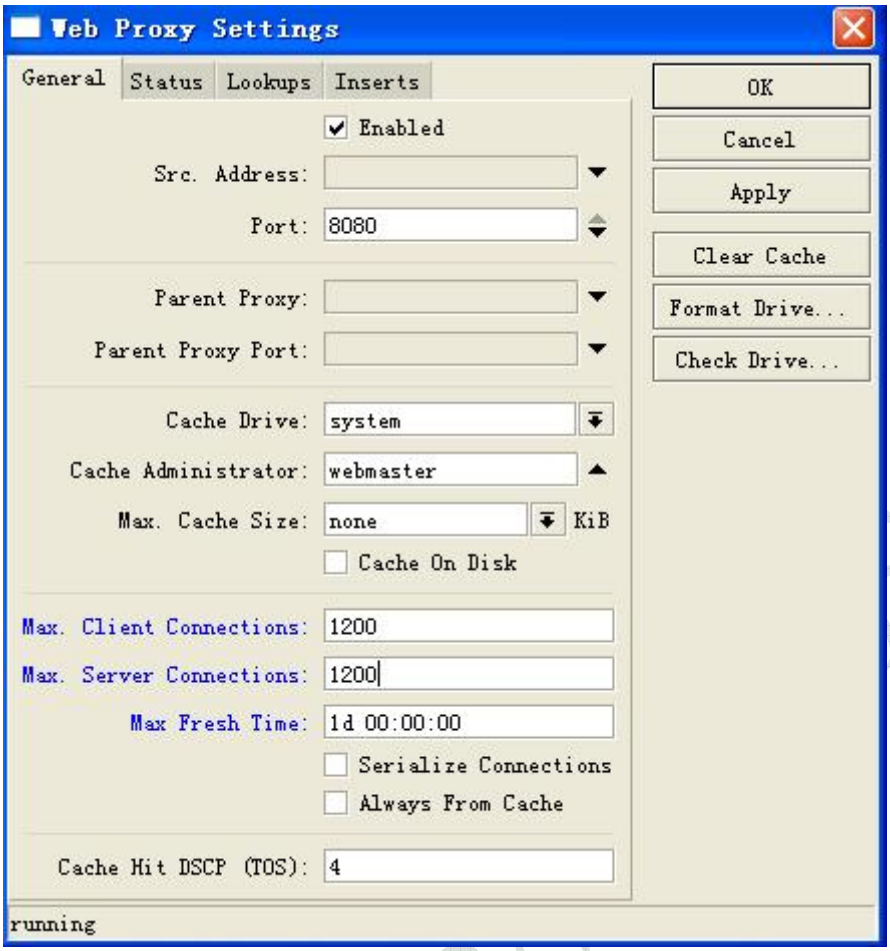
这个方法调用一个远程的，请求信息的应用层循环。最终的请求接受者应该把接受到的信息作为一个 200（OK）回应的实体正文反射给客户。最终接受者是起源服务器或者第一个代理或者在请求中接受 0 值的 **Max-Forwards** 的网关。一个 **TRACE** 请求不一定要包含一个实体。

如何使用 Proxy 禁止网站和相关文件下载

通过使用 web-proxy 禁止网站和禁止下载

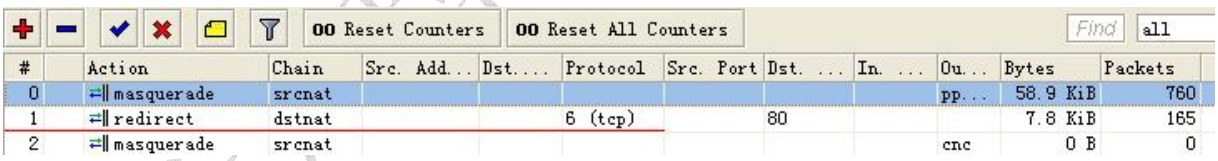
首先配置 web-proxy，配置参数如下：

```
[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，禁止外网通过路由器代理上网：

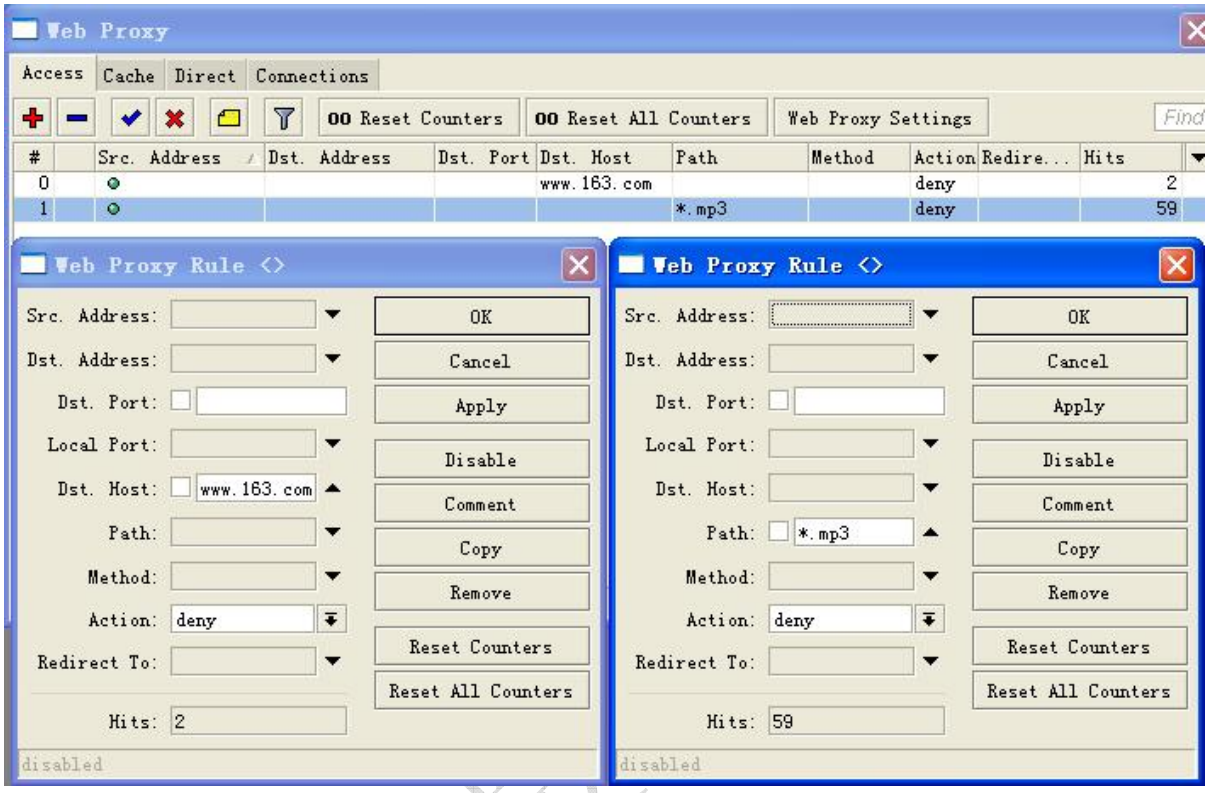
```
/ip firewall filter
chain=input in-interface=<Your WAN Port> src-address=0.0.0.0/0 protocol=tcp dst-port=8080
action=drop
```

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
```

Log 日志管理

不同的系统事件和状态信息都能被纪录下，日志能被存储到本地路由器的文件中；显示在控制台中；发送一个 email 或者远程的运行 syslog 下载程序的服务器上。

属性

日志有不同的组或者项目，日志来至于每个项目能被配置味丢弃，记录局部或者远程。局部日志文件能存储到内存中（内存记录为默认，在重启或者断电后日志会丢失）或者在硬盘上（对 Flash 硬盘有一定损害）。

操作路径：***/system logging***

属性描述

action (*name*; 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

事例

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>
```

Actions（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 (*name*) – 发送到指定的 email 地址 (仅在 action 设置为 **email**)
memory-lines (*integer*; 默认: **100**) – 在本地内存缓存记录的数量(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 (*name*) – 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

事例

添加一个新的 action 并取名为 short，将日志记录到本地内存，在内存中的记录数量小于 50 条：

```
[admin@MikroTik] system logging action> add name=short \
\... 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   short                          memory
[admin@MikroTik] system logging action>
```

日志信息 Log Messages

操作路径: ***/log***

显示局部存储日志信息

属性描述

message (*read-only: 文本*) – 信息文本
time (*read-only: 文本*) – 事件的日期和时间
topics (*read-only: 文本*) – 项目信息的从属

事例

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.
```

IP 日志访问记录

基本信息

认证授权及访问记录管理特征提供了本地和/或远程（在 RADIUS 服务器上）点对点和 HotSpot 用户管理以及流量记录（所有的 IP 流量通过路由器都会被计算；本地流量管理是一个可选项）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。

规格

功能包要求: ***system***
认证等级: *Level1*
操作路径: ***/user***, ***/ppp***, ***/ip accounting***, ***/radius***
标准与技术: RADIUS
硬件使用: *Traffic accounting requires additional memory*

本地 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; default: **no**) - 是否启用了本地 IP 访问记录日志
account-local-traffic (yes | no; default: **no**) - 是否计算来自/到达路由器的流量访问
threshold (*integer*; default: **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。

实例

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>
```

实例

查看未计算的包：

```
[admin@MikroTik] ip accounting uncounted> print
   packets: 0
     bytes: 0
[admin@MikroTik] ip accounting uncounted>
```

本地 IP 访问管理列表

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 (*read-only: integer*) - 字节总数，以条目匹配
dst-address (*read-only: IP address*) - 目的 IP 地址
dst-user (*read-only: text*) – 接受者的名称(如果可应用)
packets (*read-only: integer*) - 包的总数，以这个条目匹配
src-address (*read-only: IP address*) - 源 IP 地址
src-user (*read-only: text*) – 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。

实例

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>
```

Web 获取本地 IP 访问管理列表

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 应该连接到 URL：**http://routerIP/accounting/ip.cgi**

属性描述

accessible-via-web (yes | no; default: **no**) – 是否 snapshot 通过 web 可用
address (*IP address/netmask*; default: **0.0.0.0**) -允许存取 snapshot 的 IP 地址范围

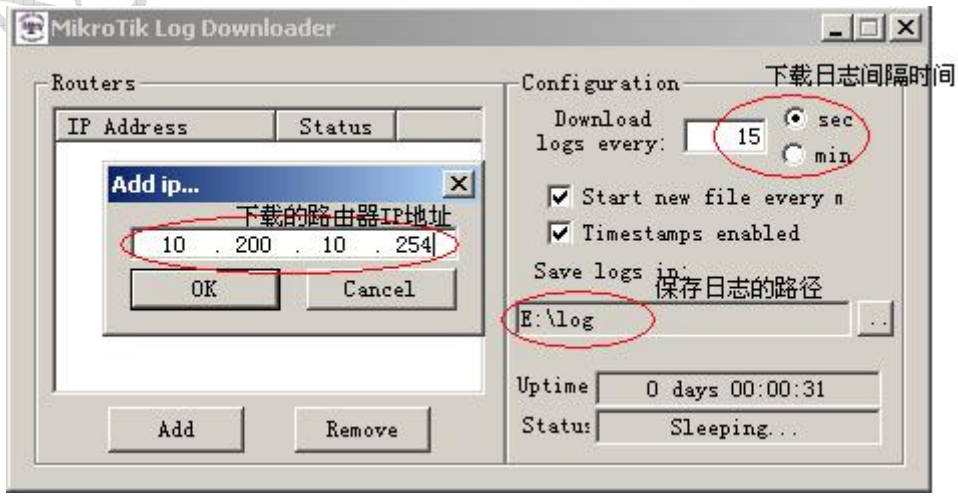
实例

仅启用来自 **192.168.10.10** 主机对流量日志的 web 访问：

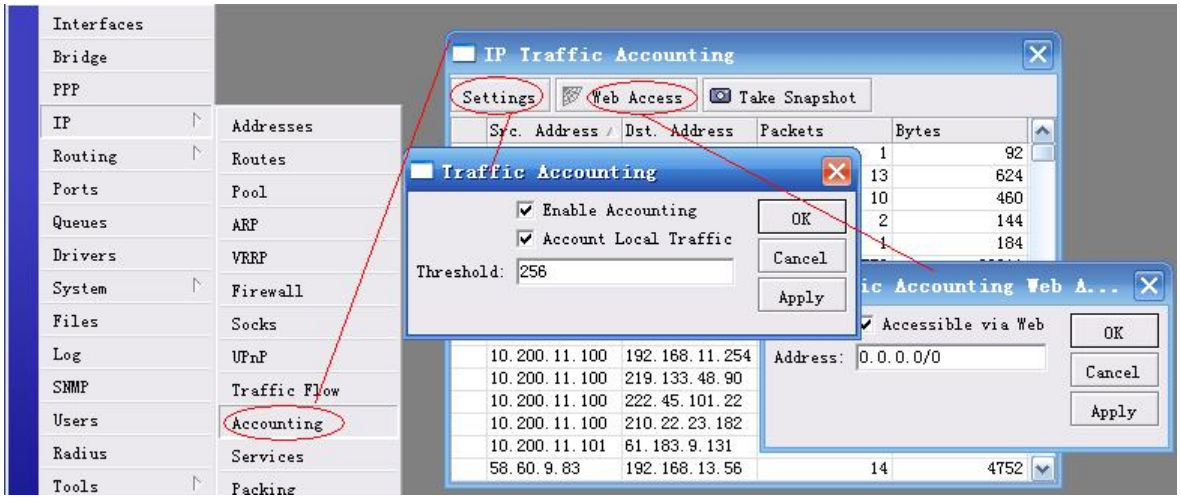
```
[admin@MikroTik] ip accounting web-access> set accessible-via-web=yes \
\... address=192.168.10.10/32
[admin@MikroTik] ip accounting web-access> print
   accessible-via-web: yes
              address: 192.168.10.10/32
[admin@MikroTik] ip accounting web-access>
```

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。

RouterOS Script 操作手册

这个手册是提供对 RouterOS 嵌入式脚本命令即操作介绍

主机脚本提供了自动维护路由器任务的功能，通过借助用户自定义发生事件脚本。一个脚本配置构成由命令和表达式(ICE - internal console expression 内部控制台表达式). 配置命令为标准的 RouterOS 命令，例如：/ip firewall filter add chain=forward protocol=gre action=drop 这个是描述在防火墙中过滤 gre 协议的操作。在脚本表达式前缀需要用 “ **:** ”并能在任何目录下操作。

事件用来触发脚本执行包括：System Scheduler, Traffic Monitoring Tool, Netwatch Tool

功能包需求: ***system***
操作路径: ***/system script***

相关功能

- Software Package Management
- System Scheduler
- Network Monitor
- Traffic Monitor
- Serial Port Monitor

控制台命令语法

属性

控制台命令被用于下面部分，输入在列表中的命令到控制台：

- **Prefix（前缀）** - 指示那一个命令到一个 ICE，如：脚本”**:** ” **:**put 或者命令部分从根目录下执行，如 “ ***/*** ”

```
[admin@MikroTik] ip firewall mangle> /ping 10.0.0.1
```

- **Path（路径）** – 希望到达目录的一个相关路径，如：**.. filter**

```
[admin@MikroTik] ip firewall mangle> .. filter print
```

- **Action（执行）** – 在指定的目录下一个可操作的执行命令，如：**add**

```
[admin@MikroTik] ip firewall mangle> /ip firewall filter add chain=forward action=drop
```

- **unnamed parameter（无名参数）**- 需要通过一些执行和输入固定格式在命令后的执行名称，如 **10.0.0.1**

```
[admin@MikroTik] ip firewall mangle> /ping 10.0.0.1
```

- **name[=value]（参数值）**- 一个跟在参数名后的各自的值，如：**ssid=myssid**

```
/interface wireless set wlan1 ssid=myssid
```

事例

在下面的例子中解释了控制台内的部分命令:

```
/ping 10.0.0.1 count=5
```

前缀	/
执行	ping
未命名参数	10.0.0.1
name[=值]	count=5

```
.. ip firewall rule input
```

路径	.. ip firewall rule
路径项目	input

```
:for i from=1 to=10 do={:put $i}
```

前缀	:
执行	for
未命名参数	i
name[=值]	from=1 to=10 do={:put $i}

```
/interface monitor-traffic ether1,ether2,ipip1
```

前缀	/
路径	interface
执行	monitor-traffic
未命名参数	ether1,ether2,ipip1

变量

属性

RouterOS 脚本语言支持两种类型的变量，**global**（系统变量）和 **local**（仅当前脚本运行的变量）取变量值使用'$'标记符号，但除了 **set** 和 **unset** 后面不需要' $'标记外，其他的都需要使用该标记。一个变量必须在脚本中首先被声明，下面有四种类型的变量：

- **全局变量** – 使用 **global** 关键字定义，全局变量可用被所有脚本和通过控制台登陆到的同一台路由器调用。注意，重启后全局变量无法保存。
- **本地变量** – 使用 **local** 关键字定义，本地变量不能和其他任何脚本或其他控制台登陆的共享。本地变量值会随脚本执行完成而丢失。
- **循环变量** – 在 **for** 和 **foreach** 内部定义，这里的变量仅能使用在 do 命令块中，在命令执行完成后将被删除掉
- **监听变量** – 一些 **monitor** 命令在 **do** 中能插入变量或控制命令额。

你分配一个新的变量值使用**:set** 命令，并定义两个为命名的参数：变量名称和新的变量值。如果一个变量不需要长时间被调用，可以通过**:unset** 命令释放变量。如果释放一个本地变量，该值会清空。如果你释放一个全局变量，该值仍然会保存在路由器中，但是在当前脚本无法调用。

注：

循环变量会影响到前面已经声明过的同样名称的变量。

事例

```
[admin@MikroTik] ip route> /
[admin@MikroTik] > :global g1 "this is global variable"
[admin@MikroTik] > :put $g1
this is global variable
[admin@MikroTik] >
```

命令替换和返回值

属性

一些终端命令是非常有用的，如他们可以输出一个变量值给其他命令。在 RouterOS 终端控制中通过命令得到返回值。返回值不会被显示出来。从一个命令中得到返回值，应包含在“[ ]”括号中。之前执行返回值的命令所得到的值包含在括号中，这个称为命令替换。

命令产生的返回值，但不限制 **find**，返回一个参考特殊项目 **ping**，返回 ping 成功的数目， **time**，返回测量时间长度值，**incr** 和 **decr**，返回新的变量值，**add**，返回内部最新建立项目编号

Example

find 命令的使用方法：

```
[admin@MikroTik] > /interface
[admin@MikroTik] interface> find type=ether
[admin@MikroTik] interface>
[admin@MikroTik] interface> :put [find type=ether]
*1,*2
[admin@MikroTik] interface>
```

这个方式你能看到内部控制台的项目编号。自然的，你能使用他们到其他的命令的操作中：

```
[admin@MikroTik] interface> enable [find type=ether]
[admin@MikroTik] interface>
```

运算符

属性

RouterOS 控制台能对数值、时间值、IP 地址、字符串和表等做简单的运算。从一个表达式中得到结果。

命令描述

- - 一元减法。对一个数值做反运算。
- - 二进制减，扣除两个数值、两个时间值、两个 IP 地址或 IP 地址和其数值。
! – 逻辑非（**NOT**）。
/ - 除法运算符。
. – 连接符，连接两个字符串或拼接一个表到其他表上或拼接一个元素给一个表。
^ - 位运算移（**XOR**）。
~ - 按位反， which inverts bits in IP address
***** - 乘法运算符。
& - 位运算与（**AND**）
&& - 逻辑与（**AND**）
+ - 加法运算符。对两个数值、两个时间值或 IP 地址做加法运算。
< - 小于符。返回值为布尔型。
<< - 左移运算符。
<= - 小于等于符，返回值为布尔型。
> - 大于符。返回值为布尔型。
>= - 大于等于符，返回值为布尔型。
>> - 右移运算符。
| - 位运算或（**OR**）
|| - 逻辑或（**OR**），返回值为布尔型。

注：

当比较两个数组的时注意：如果他们各自元素是相等的，那么两个数组即相等。

事例：

运算符的优先级和求值命令

```
[admin@MikroTik] ip firewall rule forward> :put (10+1-6*2=11-12=2+(-3)=-1)
false
[admin@MikroTik] ip firewall rule forward> :put (10+1-6*2=11-12=(2+(-3)=-1))
true
[admin@MikroTik] ip firewall rule forward
```

逻辑非（NOT）

```
[admin@MikroTik] interface> :put (!true)
false
[admin@MikroTik] interface> :put (!(2>3))
true
[admin@MikroTik] interface>
```

逻辑运算

```
[admin@MikroTik] interface> :put (-1<0)
true
[admin@MikroTik] >
1
```

按位反

```
[admin@MikroTik] interface> :put (~255.255.0.0)
0.0.255.255
[admin@MikroTik] interface>
```

加法运算

```
[admin@MikroTik] interface> :put (3ms + 5s)
00:00:05.003
[admin@MikroTik] interface> :put (10.0.0.15 + 0.0.10.0)
cannot add ip address to ip address
[admin@MikroTik] interface> :put (10.0.0.15 + 10)
10.0.0.25
[admin@MikroTik] interface>
```

减法运算

```
[admin@MikroTik] interface> :put (15 - 10)
5
[admin@MikroTik] interface> :put (10.0.0.15 - 10.0.0.3)
12
[admin@MikroTik] interface> :put (10.0.0.15 - 12)
10.0.0.3
[admin@MikroTik] interface> :put (15h - 2s)
14:59:58
[admin@MikroTik] interface>
```

乘法运算

```
[admin@MikroTik] interface> :put (12s * 4)
00:00:48
[admin@MikroTik] interface> :put (-5 * -2)
10
[admin@MikroTik] interface>
```

除法运算

```
[admin@MikroTik] interface> :put (10s / 3)
00:00:03.333
[admin@MikroTik] interface> :put (5 / 2)
2
[admin@MikroTik] interface>
[admin@MikroTik] > :put (0:0.10 / 3)
00:00:02
[admin@MikroTik] >
```

各种逻辑比较运算

```
[admin@MikroTik] interface> :put (10.0.2.3<=2.0.3.10)
false
[admin@MikroTik] interface> :put (100000s>27h)
true
[admin@MikroTik] interface> :put (60s,1d!=1m,3600s)
true
[admin@MikroTik] interface> :put (bridge=routing)
false
[admin@MikroTik] interface> :put (yes=false)
false
[admin@MikroTik] interface> :put (true=aye)
false
[admin@MikroTik] interface>
```

逻辑与和或运算

```
[admin@MikroTik] interface> :put ((yes && yes) || (yes && no))
true
[admin@MikroTik] interface> :put ((no || no) && (no || yes))
false
[admin@MikroTik] interface>
```

按位与、或、异或运算

```
[admin@MikroTik] interface> :put (10.16.0.134 & ~255.255.255.0)
0.0.0.134
[admin@MikroTik] interface>
```

移位运算

```
[admin@MikroTik] interface> :put (~((0.0.0.1 << 7) - 1))
255.255.255.128
[admin@MikroTik] interface>
```

连接运算符

```
[admin@MikroTik] interface> :put (1 . 3)
13
[admin@MikroTik] interface> :put (1,2 . 3)
1,2,3
[admin@MikroTik] interface> :put (1 . 3,4)
13,4
[admin@MikroTik] interface> :put (1,2 . 3,4)
1,2,3,4
[admin@MikroTik] interface> :put ((1 . 3) + 1)
14
[admin@MikroTik] interface>
```

数据类型

属性

RouterOS 区分几种数据类型，字符型、布尔型、数值型、时间型、IP 地址、内码和列表。 RouterOS 首先会试着将任何值转换为制定的类型。

内部脚本语言弥补了特殊函数之间类型转换的不足。通过内部的 **toarray**, **tobool**, **toid**, **toip**, **tonum**, **tostr** 和 **totime** 函数每个值转换到相应的列表（**list**）中，对应为：**boolean**, **internal number**, **IP address**, **number**, **string** 或 **time**.

数字类型在内部表示为 64 位带符号的整型， 因此一个数字类型值变量可用长度从-9223372036854775808 到 9223372036854775807. 同样可用输入十六进制的数值，在前面加入 **0x** ， 例如：

```
[admin@MikroTik] > :global MyVar 0x10
[admin@MikroTik] > :put $MyVar
16
[admin@MikroTik] >
```

列表通过逗号来区分值的次序，在空白出使用逗号间隔方式部推荐使用，因为这会让控制终端无法识别字符的边境。

Boolean 型的值为 **true** 或 **false**. 控制终端判断 **true** 为 **“yes”**， **false** 为 **“no”** 。

时间间隔可以输入 HH:MM:SS 例如：

```
[admin@MikroTik] > :put 01:12:1.01
01:12:01.010
[admin@MikroTik] >
```

或者通过累计数字，具体指明单位时间的标记(**d** 对应 days, **h** 对应 hours, **m** 对应 minutes, **s** 对应 seconds，以及 **ms** 对应 milliseconds)例如：

```
[admin@MikroTik] > :put 2d11h12
2d11:00:12
[admin@MikroTik] >
```

时间单位：

- **d, day, days** – 一天，或者 24 小时
- **h, hour, hours** – 一小时
- **m, min** – 一分钟
- **s** – 一秒
- **ms** – 一毫秒，等同 0.001 秒

控制终端同样接受时间为小数点的形式：

```
[admin@MikroTik] > :put 0.1day1.2s
02:24:01.200
[admin@MikroTik] >
```

命令参考文档

RouterOS 有多个嵌入式的控制终端命令和表达式 ICE 不依赖于当前操作目录。这些命令不能直接改变配置，但他们可以做日常的维护工作。所有 ICE 可以通过在操作符输入“ ： ”后敲击“ ？ ” 显示出。例如：

```
[admin@MikroTik] > :
beep        execute  global  list    pick    time    toip    typeof
delay       find     if      local   put     toarray  tonum   while
do          for      led     log     resolve  tobool   tostr
environment  foreach  len     nothing  set     toid    totime
[admin@MikroTik] >
```

命令属性

beep – 通过 PC 内置的蜂鸣器或者扬声器发出一个指定 **length** （时间长度）的 **frequency** Hz（频率 Hz）.

输入参数

frequency (*整型*; 默认: **1000**) – 信号频率大小用单位 Hz

length (时间; 默认: **100ms**) – 信号长度

```
[admin@MikroTik] > :beep length=2s frequency=10000

[admin@MikroTik] >
```

delay – 在一个给定的时间长度不做任何操作

输入参数

delay-time (*时间*) – 等待的时间长度

- **omitted** – 无限制延迟

do – 反复执行命令直到获取一个适当的值。如果没有参数获取，**do** 只执行有效操作一次，其中不会有什么作用。 如果一逻辑条件被指定到 **while** 参数种,将会在命令执行后作判断，在该条件判断中为 ***true***， **do** 语句会被一次一次的执行直到满足 ***false*** 条件， **if** 参数，在做后面语句的任何操作时判断一次，如果 ***false*** 不会执行任何的操作

输入参数

unnamed (*文本*) – 反复执行的操作

while (yes | no) – 条件语句，这是判断每一次执行后的执行结果

if (yes | no) – 条件语句，这是判断一次执行之前的声明

```
[admin@MikroTik] > {:global i 10; :do {:put $i; :set i ($i - 1);} \
\... while (($i < 11) && ($i > 0)); :unset i;}
10
9
8
7
6
5
4
3
2
1
[admin@MikroTik] >
```

environment print – 显示关于当前变量的初始化情况。所有在系统中的全局变量 global variables 被以标题为 **Global Variables** 下列出。所有变量插入当前脚本（通过**:local**、通过**:for** 或者**:foreach**）被以标题为 **Local Variables** 下列出。

创建变量并显示出他们的列表：

```
 [admin@MikroTik] > :local A "This is a local variable"
[admin@MikroTik] > :global B "This is a global one"
[admin@MikroTik] > :environment print
Global Variables
B=This is a global one
Local Variables
A=This is a local variable
[admin@MikroTik] >
```

find – 查找字符串内的一个字符或者一个元素在项目内的值，根据变量类型并返回一个变量的位置。

输入参数

unnamed(*文本* | 列表) – 搜索字符或者字符列表值，并执行相关的操作

unnamed(*文本*) – 字符的搜索

unnamed（整型) – 搜索的起始位置，或搜索到目标返回的位置

```
[admin@MikroTik] interface pppoe-server> :put [:find "13sdf1sdfss1sfsdf324333"  ]
0
[admin@MikroTik] interface pppoe-server> :put [:find "13sdf1sdfss1sfsdf324333" 3  ]
1
[admin@MikroTik] interface pppoe-server> :put [:find "13sdf1sdfss1sfsdf324333" 3  3]
17
[admin@MikroTik] interface pppoe-server> :put [:find "1,1,1,2,3,3,4,5,6,7,8,9,0,1,2,3" 3 ]
4
[admin@MikroTik] interface pppoe-server> :put [:find "1,1,1,2,3,3,4,5,6,7,8,9,0,1,2,3" 3 3]
4
[admin@MikroTik] interface pppoe-server> :put [:find "1,1,1,2,3,3,4,5,6,7,8,9,0,1,2,3" 3 4]
5
[admin@MikroTik] interface pppoe-server> :put [:find "1,1,1,2,3,3,4,5,6,7,8,9,0,1,2,3" 3 5]
15
[admin@MikroTik]
```

for – 执行所给定的数次反复循环的命令，通过 **from** 和 **to** 设置起始和结算参数。

输入参数

unnamed (*名称*) – 循环计数器变量名称。

from (*整型*) – 循环起始的变量值

to (*整型*) – 循环结束的变量值

step (整型; 默认: **1**) – 递增变量. 在循环起始到结束中间每循环一次的间距变量

do (*文本*) – 执行包含在内的命令

```
[admin@MikroTik] > :for i from=1 to=100 step=37  do={:put ($i . " - " . 1000/$i)}
1 - 1000
38 - 26
75 - 13
[admin@MikroTik] >
```

foreach – 执行所提供在列表中的每一个元素

输入参数

unnamed (*名称*) -循环计数器变量名称。

in (*列表*) – 变量列表范围或路径

do (*text*) - 执行包含在内的命令

显示出一个 interface 中获得列表各自的 IP 地址

```
:foreach i in=[/interface find type=ether ] \
\... do={:put ("+--" . [/interface get $i name]); \
\... :foreach j in=[/ip address find interface=$i]
\... do={:put ("|  `--" . [/ip address get $j address])}}
+--ether1
|  `--1.1.1.3/24
|  `--192.168.50.1/24
|  `--10.0.0.2/24
+--ether2
|  `--10.10.0.2/24
[admin@MikroTik] >
```

global – 声明全局变量

输入参数

unnamed(*名称*) – 变量名称

unnamed(*文本*) – 值，分配给变量的内容

```
[admin@MikroTik] > :global MyString "This is a string"
[admin@MikroTik] > :global IPAddr 10.0.0.1
[admin@MikroTik] > :global time  0:10
[admin@MikroTik] > :environment print
Global Variables
IPAddr=10.0.0.1
time=00:10:00
MyString=This is a string
Local Variables
[admin@MikroTik] >
```

if – 条件语句. 如果一个逻辑判断为真，这时执行 **do** 分程序块中的命令，否则选择 **else** 分程序块中的执行。

输入参数

unnamed(yes | no) – 逻辑条件语句，在执行之后声明内容前判断一次

do(*文本*) – 如果 if 语句判断为真，在这个分程序块的命令会被执行。

else(*文本*) -如果 if 语句判断为假，在这个分程序块的命令会被执行。

通过 if 语句检查 firewall 中是否有任何规则被添加

```
[admin@MikroTik] > :if ([:len [/ip firewall filter find]] > 0) do={:put true} else={:put false}
true
[admin@MikroTik] >
```

检查网关是否能到达。在这个事例中网关地址为 **10.0.0.254**

```
[admin@MikroTik] > :if ([/ping 10.0.0.254 count=1] = 0) do {:put "gateway unreachable"}
10.0.0.254 ping timeout
1 packets transmitted, 0 packets received, 100% packet loss
gateway unreachable
[admin@MikroTik] >
```

led – 允许控制系统内嵌的 LED（发光二极管）。这个命令仅能在 RouterBOARD 平台与安装 **routerboard** 或 **rb500** 功能包。LED 数量根据 RouterBOARD 型号不同而定。

输入参数

led1(yes | no) – 控制第一个 LED

led2(yes | no) -控制第二个 LED

led3(yes | no) -控制第三个 LED

led4(yes | no) -控制第四个 LED

length(*time*) – 具体指定操作的长度

- **omitted** – LED 长亮

打开 LED2 和 LED3 时间为 5 秒

```
[admin@MikroTik] > :led led2=yes led3=yes length=5s
```

len – 返回在字符串的字符数或列表中的元素数目

输入参数

unnamed(*name*) – 返回字符串或列表的长度

```
[admin@MikroTik] > :put [:len gvejimezyfopmekun]
17
[admin@MikroTik] > :put [:len gve,jim,ezy,fop,mek,un]
6
[admin@MikroTik] >
```

list – 显示一个能获得给定匹配关键字的列表

输入参数

unnamed(*文本*) – 第一个搜索关键字

unnamed(*文本*) -第二个搜索关键字

unnamed(*文本*) -第三个搜索关键字

显示控制台命令下含有 **hotspot**, **add** 和 **user** 部分的命令与路径

```
[admin@MikroTik] > :list user hotspot "add "
List of console commands under "/" matching "user" and "hotspot" and "add ":

ip hotspot profile add name= hotspot-address= dns-name= \
\... html-directory= rate-limit= http-proxy= smtp-server= \
\... login-by= http-cookie-lifetime= ssl-certificate= split-user-domain= \
\... use-radius= radius-accounting= radius-interim-update= copy-from=
ip hotspot user add server= name= password= address= mac-address= \
\... profile= routes= limit-uptime= limit-bytes-in= limit-bytes-out= \
\... copy-from= comment= disabled=
ip hotspot user profile add name= address-pool= session-timeout= \
\... idle-timeout= keepalive-timeout= status-autorefresh= \
\... shared-users= rate-limit= incoming-filter= outgoing-filter= \
\... incoming-mark= outgoing-mark= open-status-page= on-login= on-logout= copy-from=
[admin@MikroTik] >
```

local – 声明本地变量

输入参数

unnamed(*名称*) – 变量名称

unnamed(*文本*) – 值，分配给变量的内容

```
[admin@MikroTik] > :local MyString "This is a string"
[admin@MikroTik] > :local IPAddr 10.0.0.1
[admin@MikroTik] > :local time   0:10
[admin@MikroTik] > :environment print
Global Variables
Local Variables
```

```
IPAddr=10.0.0.1
time=00:10:00
MyString=This is a string
[admin@MikroTik] >
```

log – 通过参数添加一个指定的信息到系统 logs 中。

输入参数

unnamed(*名称*) – 记录日志的功能名称

unnamed(*文本*) – 被记录的文本信息

发送信息到 **info** 日志中

```
[admin@MikroTik] > :log info "Very Good thing happened. We have received our first packet!"
[admin@MikroTik] > /log print follow
...
19:57:46 script,info Very Good thing happened. We have received our first packet!
...
```

nothing – 没有任何操作，并返回值类型为“nothing”。 在条件语句中 nothing 等同“false”

从一个字符串中挑选一个不存在的符号

```
[admin@MikroTik] > :local string qwerty
[admin@MikroTik] > :if ([:pick $string 10]=[:nothing]) do={
{... :put "pick and nothing commands return the same value"}
pick and nothing commands return the same value
[admin@MikroTik] >
```

pick – 根据输入的值返回一个元素长度或一个子串值

输入参数

unnamed(*文本* | 列表) – 字符串或值列表的来源

unnamed(*整型*) – 字符串中子串的起始位置

unnamed(*整型*) -字符串中子串的结束位置

```
[admin@MikroTik] > :set a 1,2,3,4,5,6,7,8
[admin@MikroTik] > :put [:len $a]
8
[admin@MikroTik] > :put [:pick $a]
1
[admin@MikroTik] > :put [:pick $a 0 4]
1,2,3,4
[admin@MikroTik] > :put [:pick $a 2 4]
3,4
[admin@MikroTik] > :put [:pick $a 2]
3
[admin@MikroTik] > :put [:pick $a 5 1000000]
6,7,8
[admin@MikroTik] > :set a abcdefghij
[admin@MikroTik] > :put [:len $a]
10
[admin@MikroTik] > :put [:pick $a]
a
[admin@MikroTik] > :put [:pick $a 0 4]
```

```
abcd
[admin@MikroTik] > :put [:pick $a 2 4]
cd
[admin@MikroTik] > :put [:pick $a 2]
c
[admin@MikroTik] > :put [:pick $a 5 1000000]
fghij
```

put – 回复所提供的变量值到控制台

输入参数

unnamed(*文本*) – 需要回复的文本信息

显示 **ether1** 接口的 MTU 值

```
[admin@MikroTik] > :put [/interface get ether1 mtu]
1500
[admin@MikroTik] >
```

resolve – 解析 DNS 域名并返回主机的 IP 地址，首先需要配置好路由器的 DNS 参数(**/ip dns** 目录下)

输入参数

unnamed(*文本*) – 需要解析 IP 的主机域名

DNS 配置和 **resolve** 命令事例

```
[admin@MikroTik] ip route> /ip dns set primary-dns=159.148.60.2
[admin@MikroTik] ip route> :put [:resolve www.example.com]
192.0.34.166
```

set – 分配一个新值给变量

输入参数

unnamed(*name*) – 变量名称

unnamed(*text*) – 新的变量值

通过 **/ip route find dst 0.0.0.0** 的命令，查找路由表中 **dst-address** 返回值为 **0.0.0.0** 的值，这个值通常是路由器的默认网关，当查到后通过**/ip route set** 命令修改网关地址为 10.0.0.217

```
[admin@MikroTik] > /ip route set [/ip route find dst 0.0.0.0] gateway 10.0.0.1
[admin@MikroTik] >
```

time – 计算出所给命令的执行时间总长度

输入参数

unnamed(*text*) – 控制台命令测量执行时间

计算出解析 www.example.com 需要的时间

```
[admin@MikroTik] > :put [:time [:resolve www.example.com ]]
00:00:00.006
[admin@MikroTik] >
```

while – 反复执行给定的控制命令，直到逻辑条件为 *true*

输入参数

unnamed(yes | no) – 条件，在每一次执行前判断声明范围

do(*文本*) – 反复执行的控制命令

```
[admin@MikroTik] > :set i 0; :while ($i < 10) do={:put $i; :set i ($i + 1)};
```

```
0
1
2
3
4
5
6
7
8
9
[admin@MikroTik] >
```

计划任务 scheduler

基本信息

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***
技术与标准: None
硬件使用: *Not significant*

计划表配置

计划表能触发脚本执行，在指定的时间段或者是在指定的时间间隔.

属性描述

interval (*time*; 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 (*name*) – 任务名
on-event (*name*) – 脚本执行名。通过调用 **/system script** 里的脚本规则名称
run-count (*read-only: integer*) – 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 (*date*) – 开始脚本执行的日期
start-time (*time*) – 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。

实例

我们添加一个任务执行系统日志记录测试，这个操作为 **log-test**：

```
[admin@MikroTik] system script> add name=log-test source=:log message=test
[admin@MikroTik] system script> print
   0 name="log-test" source=":log messgae=test" owner=admin run-count=0
[admin@MikroTik] system script> .. scheduler
[admin@MikroTik] system scheduler> add name=run-1h interval=1h
on-event=log-test
[admin@MikroTik] system scheduler> print

Flags: X - disabled
 #   NAME     ON-EVENT  START-DATE  START-TIME INTERVAL    RUN-COUNT
 0   run-1h    log-test  mar/30/2004 06:11:35   1h         0
[admin@MikroTik] system scheduler>
```

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：

```
[admin@MikroTik] queue simple> add name=Cust0 interface=ether1 \
\... dst-address=192.168.0.0/24 limit-at=64000
[admin@MikroTik] queue simple> print
Flags: X - disabled, I - invalid
  0   name="Cust0" target-address=0.0.0.0/0 dst-address=192.168.0.0/24
      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>
```

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，即 smtp 服务的配置，操作路径**/tool e-mail.**例如：

```
[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>
```

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 message=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>
```

User Manager 操作手册

User Manager 是一套类 Radius 管理系统，他主要应用于：

- Hotspot 用户管理；
- PPP（PPTP/PPPoE）用户管理；
- DHCP 用户管理；
- 无线用户管理；
- RouterOS 登录帐号管理

User Manager 操作主要通过 Web 界面进行管理，方便的添加、删除和查询用户信息，现在的 User Manager 仍然在开发阶段，许多功能仍然在补偿。使用 User Manager 最少需要 32M 内存和 2M 硬盘空间。

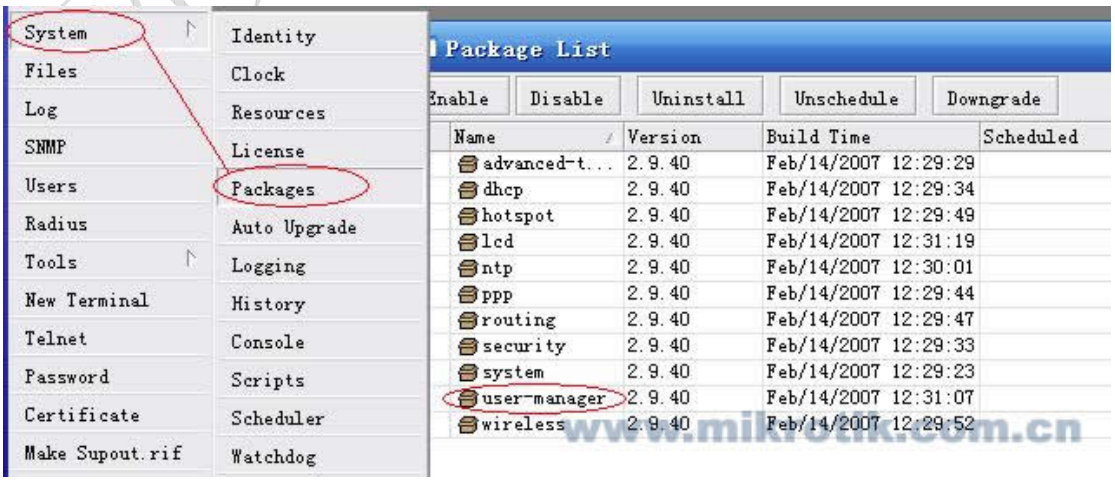
在 RouterOS v3.0 修改为在线用户许可方式：

- Level3 - 10 active users
- Level4 - 20 active users
- Level5 - 50 active users
- Level6 - Unlimited active users

如何在 Hotspot 中通过设置 User Manager 认证上网

初始化 User Manager

首先确定你是否安装了 User Manager 的功能包，我们可以在 RouterOS 中的/system packages 中查询到（User Manager 是在 RouterOS 2.9.25 后被添加的，在以后的几个版本中在不断的完善）：

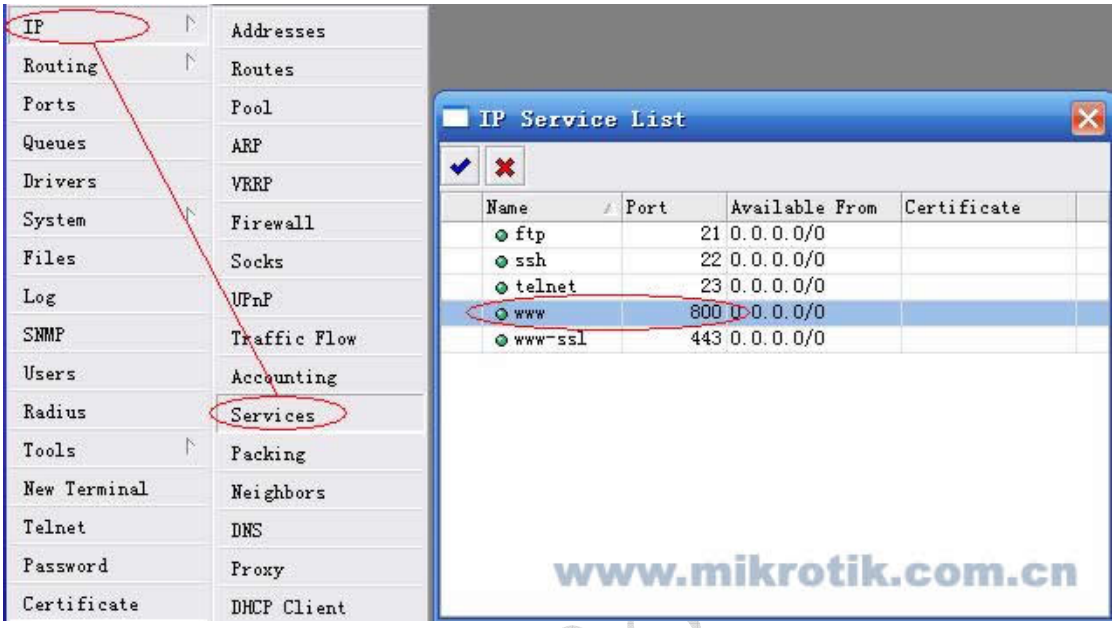


正确安装 User Manager 后，我们进入/tool usermanager 的目录配置相应的参数和启用管理帐号，我们需要进入命令行操作，才能初始化 User Manager 的管理账号，如下面我们进入 customer 目录添加 User Manager 的客户账号：

```
[admin@MikroTik] > tool
[admin@MikroTik] tool> user-manager
[admin@MikroTik] tool user-manager> customer
[admin@MikroTik] tool user-manager customer> add login=yus password=yus
```

登录名为 yus，登录密码为 yus。

当我们设置好后，我们可以同 web 页面登录到 User Manager 的管理页面，我们将 RouterOS 的 www 端口设置为 800：



设置 www 端口为 800，是为了在设置 Hotspot 后，能通过 Web 页面正常访问 User Manager。

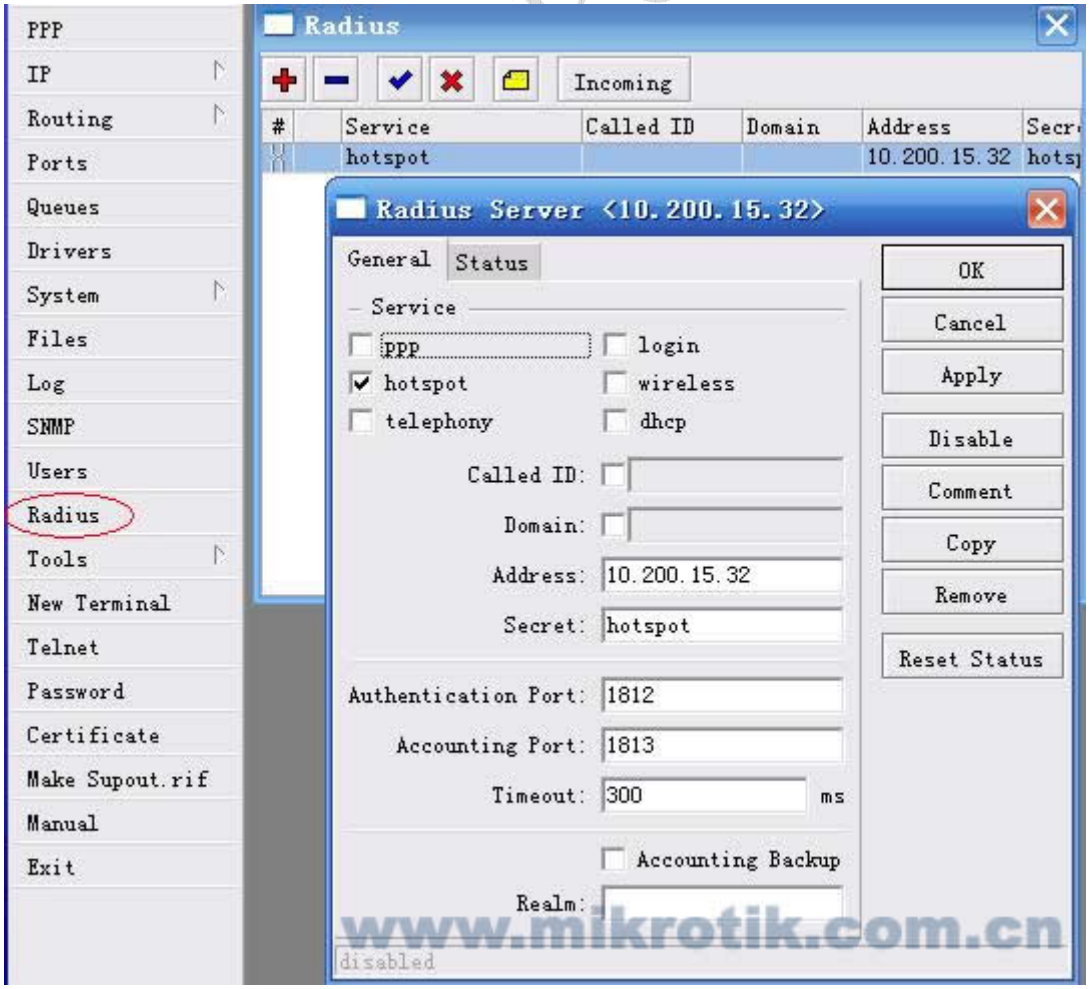
在设置完成后我们进入IE浏览器，打开 http://routerIP:800/userman便可以访问User Manager的登录页面：

在这里我们的 RouterOS IP 地址为 10.200.15.32，通过 Web 方式即能登录到 User Manager。

在完成 User Manager 的初始化配置后，我们需要和 RouterOS 建立 Radius 服务的连接，并且配置 Hotspot 的帐号和类型通过 User Manager 来认证。

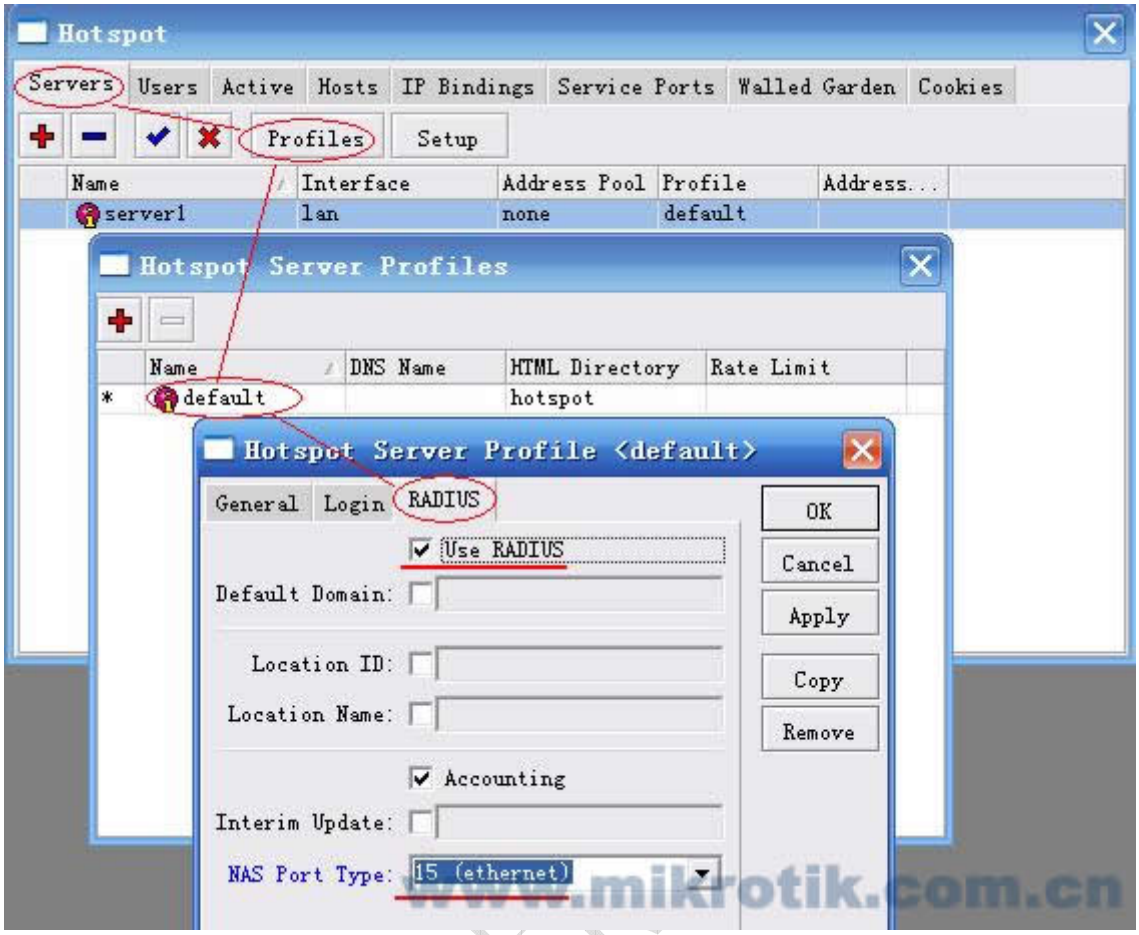
如何设置 RouterOS 中的 Radius

首先设置 RouterOS 上的 Radius 参数和 Hotspot 的配置，进入路由器的 Radius 目录设置 Radius 服务器的 IP 地址和访问密码，并配置 Hostpot 需要通过 Radius 认证：



这里是通过本来 Radius 做认证，所以 address 输入的是本地的 IP 地址，并设置 Secret 为 hotspot。

然后进入/ip hotspot 目录，在 servers 的 profile 中配置 Radius 服务：



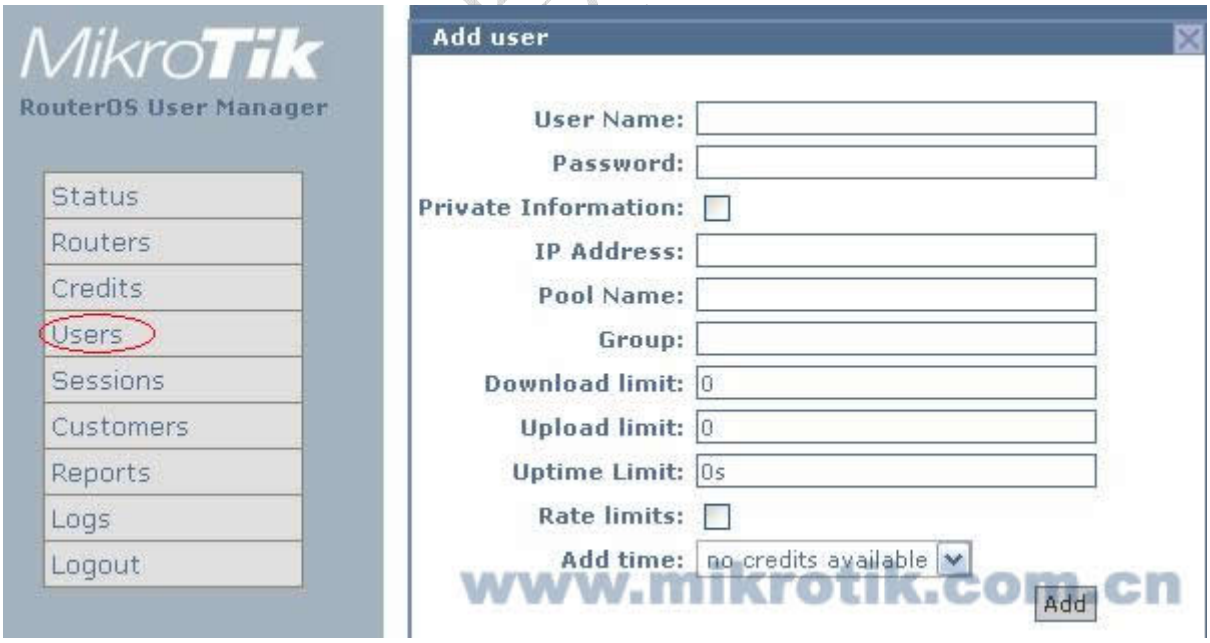
设置完 hotspot profile 的 Radius 参数后，这样 RouterOS 的 Radius 参数就配置完成，下面需要配置 User Manager 的参数：

进入 User Manager 中的 Router 项配置与本地的 RouterOS 连接参数，我们添加一个项目，将名称取名为“demo”，在 User Manager 中同样的我们将 IP 地址设置为 RouterOS 的本地 IP，Secret 为 hotspot。在 Routers 项目中可以添加多个 Radius 客户端，并能同时为多个 Radius 客户端提供认证。

当 User Manager 中的 Routers 参数配置完成后，RouterOS 就可以和 User Manager 相互通信传递用户认证信息了。

添加认证用户帐号

最后就是我们在 User Manager 的 Users 项目中配置用户的帐号：



User name：用户帐号名称

Password：用户密码

Private information：是否设置用户的个人信息资料

IP address：分配给用户的 IP 地址

Pool name：分配给用户的地址池（地址池从 RouterOS 中获取）

Group：设置 Hotspot 用户的 Profile 规则，仅限 Hotspot 使用。

Download limit：按照下行流量计费

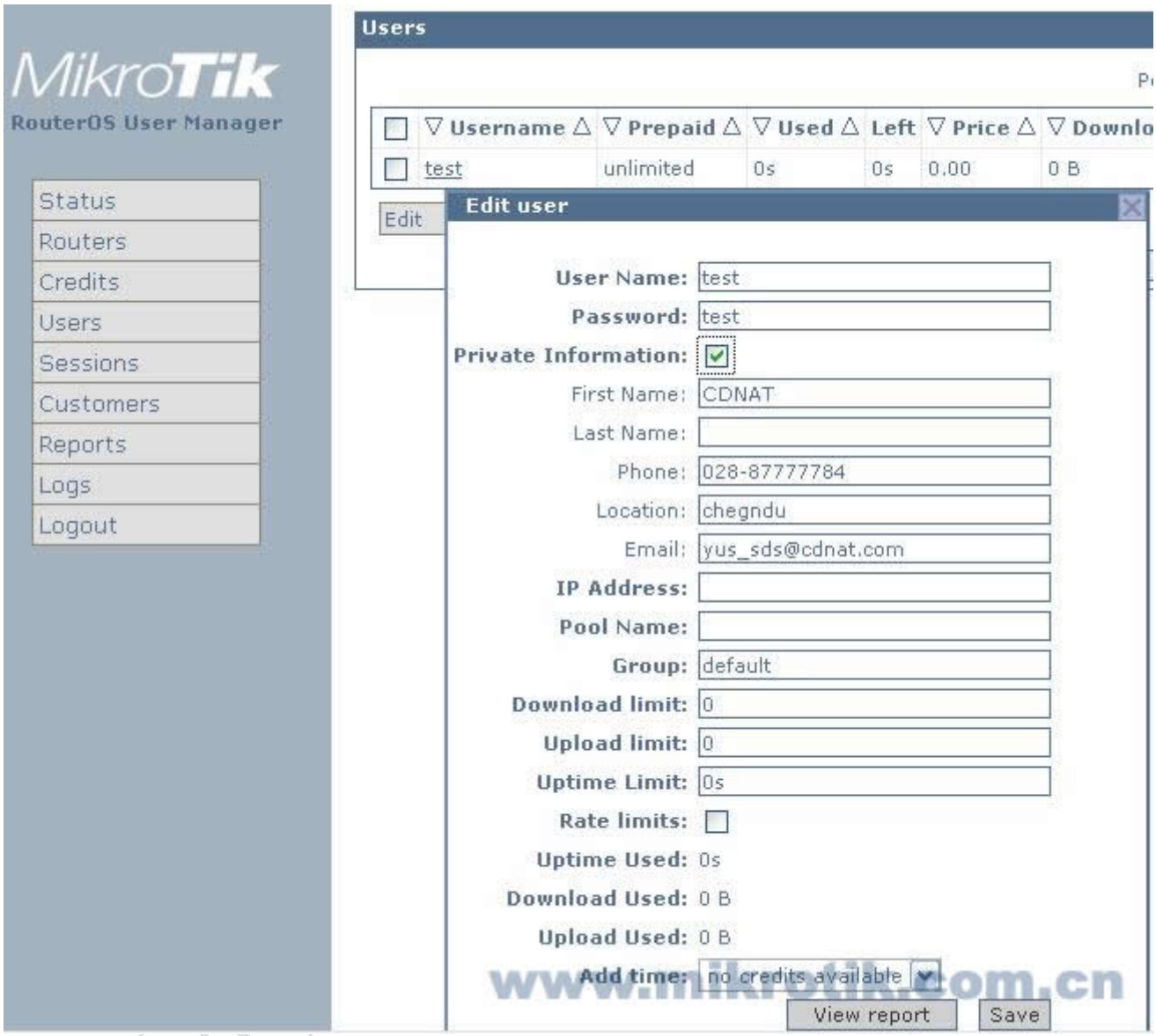
Upload limit：按照上行流量计费

Uptime limit：按照在线时间计费

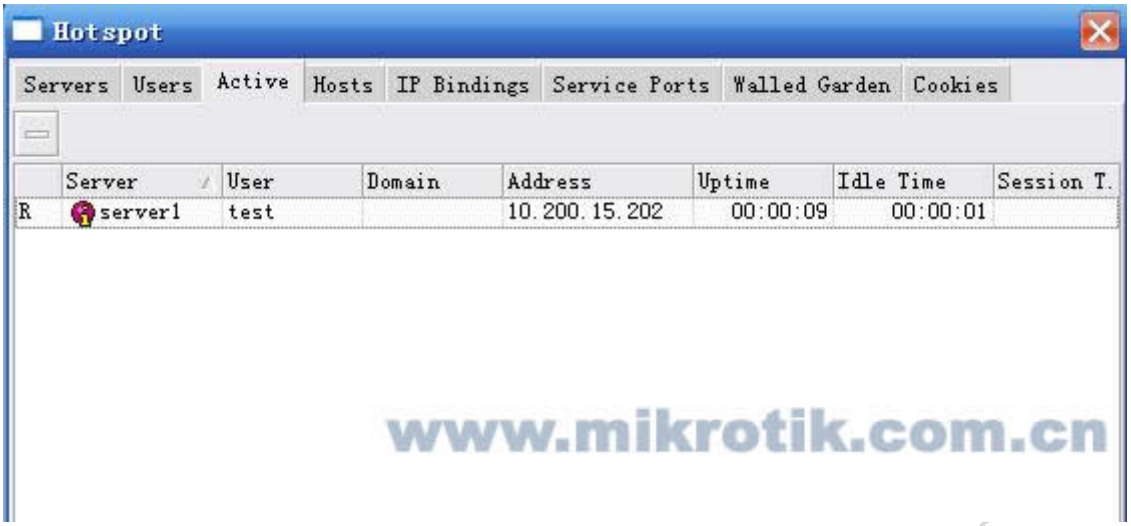
Rate limit：是否设置流量控制规则

Add time：添加时间控制规则

下面我们添加一个帐号，名称为 test，密码为：test，配置 Group 为 Hotspot 上的默认规则 default。



现在我们可以通过 User Manager 添加的帐号，已经可以在 Hotspot 认证上通过，如下图：



如果是通过 Radius 认证登陆到 Hotspot 上的，在该 test 登录行最前面会有一个“R”出现。

用户如何修改自己的密码

User Manager支持用户自主修改自己的密码，这样能让用户自己管理自己的帐号，在这里需要用户登录到指定的路径去：http://routerIP:800/user

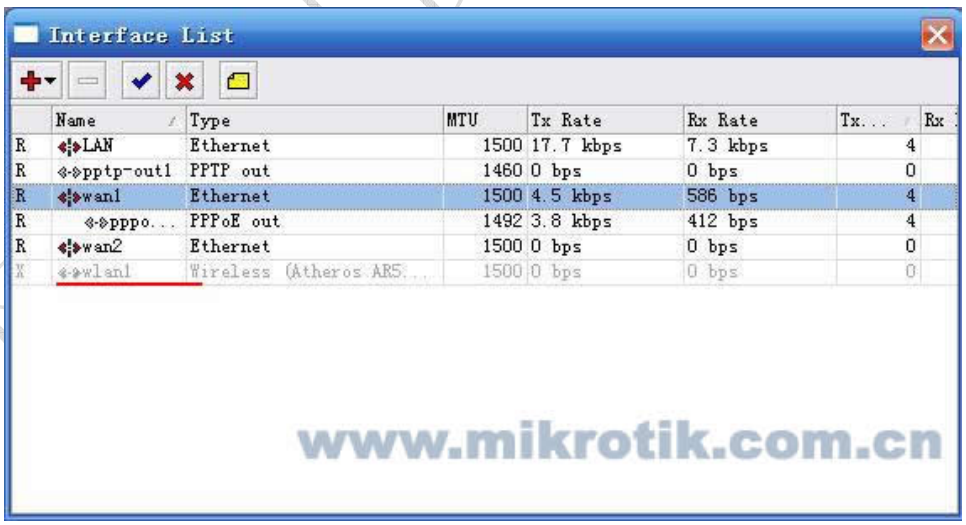
当用户登录到该页面后，只有输入自己的用户名和密码，就可以登录到设置页面，并修改自己的密码和个人信息资料：

通过 Webbox 配置 RouterOS 无线

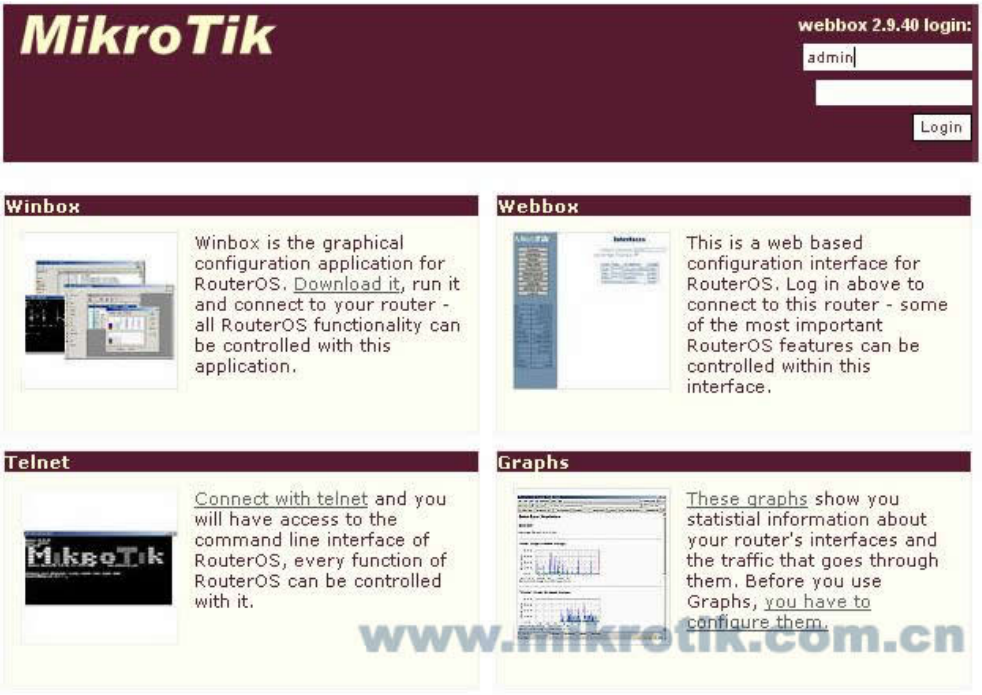
通过在一台路由器上安装了一张 TP-Link 的 WN510 PCMCIA 无线网卡实现小型的无线上网，该网卡支持 802.11b/g，最高 54Mb 的带宽。该无线网卡主要用于笔记本的 PCMCIA 接口做为客户端使用，价格自然就不贵。MikroTik RouterOS 支持该网卡芯片驱动，并能支持 AP 功能，这样通过寻找一张 PCMCIA 转 PCI 卡接到 PC 机上做无线 AP 使用或者也可以购买类似的 PCI 接口无线网卡。但由于 PCMCIA 本来设计为客户端使用，所以没有配置天线或者跳线接口，在无线覆盖范围比较窄，只能做到房间的覆盖。

先安装好无线网卡，然后进入 RouterOS 中查看是否寻找到，一般 PCMCIA 在 RouterOS 中是支持热拔插的。所以在 RouterOS 运行的时候插入网卡是能即时寻找到的，但有时也有特殊情况建议重启路由器。

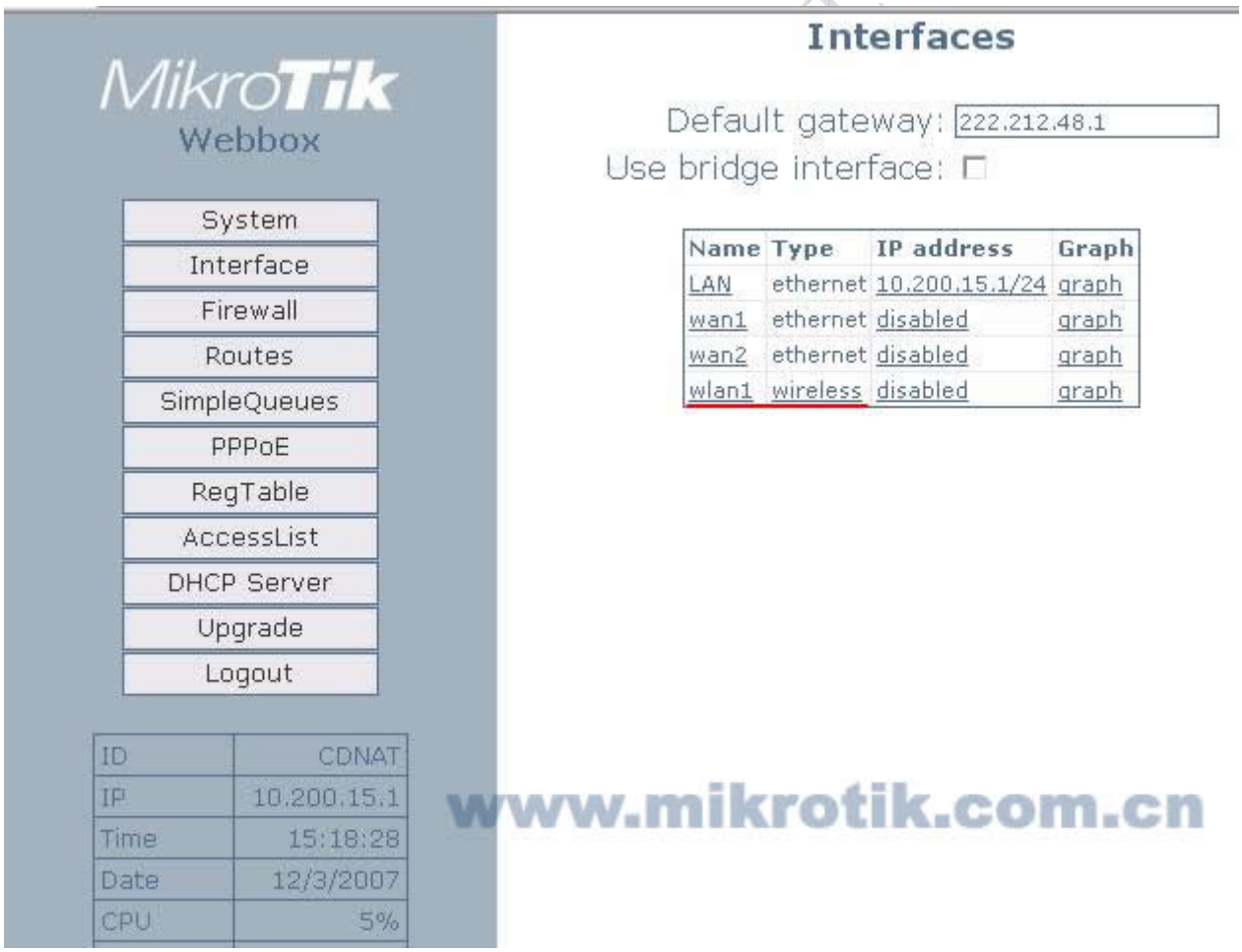
这里路由器使用的是 ADSL 接入，已配置好了 IP 地址、网关和 NAT 设置。当查找到无线网卡后，我们可以在 interface 中找到无线网卡，刚安装好的无线网卡，默认状态下是禁用的：



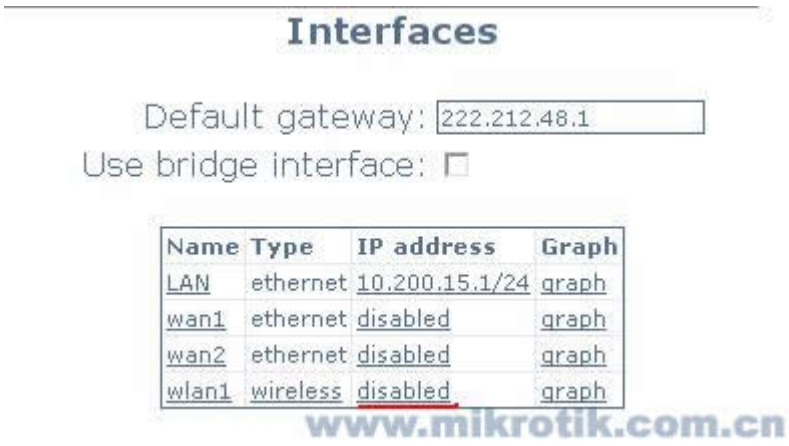
之后通过在 IE 浏览器中输入 RouterOS 的 IP 地址，进入 Webbox，在 Webbox 中配置无线要比在 Winbox 中简单而快捷，方便普通用户的使用：



通过在 login 处输入帐号和密码登陆到 Webbox，并进入 Interface 选项



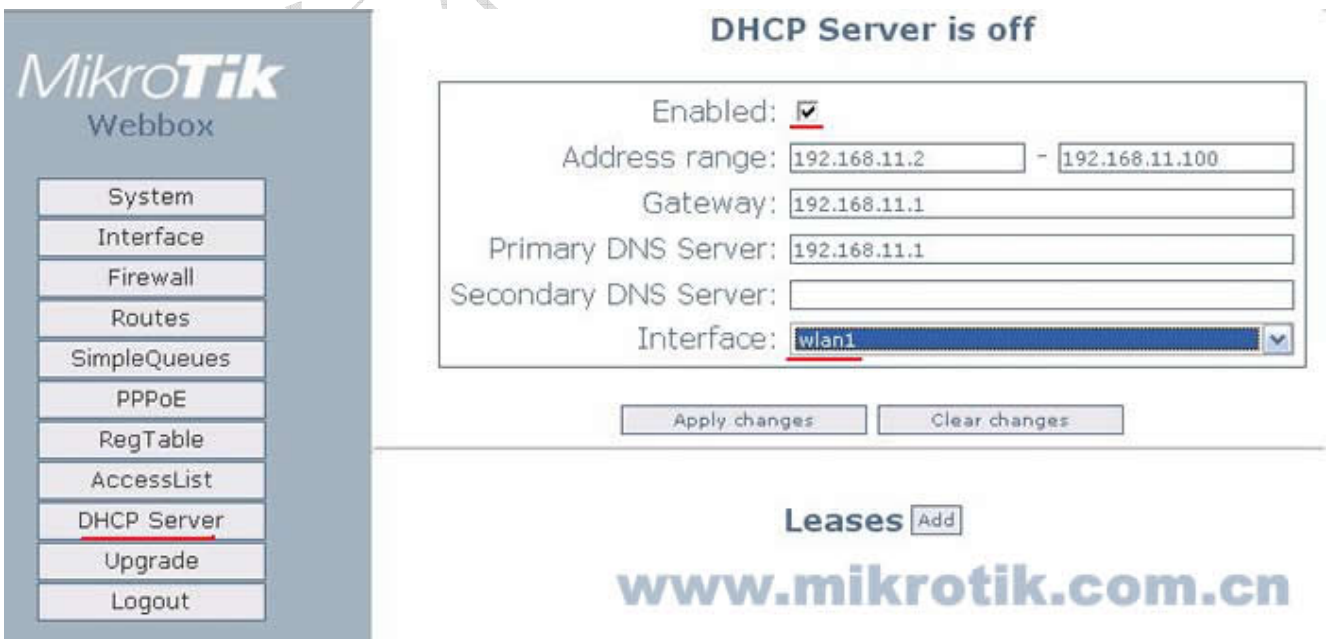
首先启用网卡和设置无线网卡 IP 地址，可以看到 wlan1 的 IP address 为 disabled，即被禁用的，我们需要在下面启用网卡



点击 disabled 进入 IP 地址配置：

这里我们选择 Configure an IP address Manually 即手动配置一个 IP 地址：192.168.11.1/24

然后进入 DHCP-Server 配置 IP 地址分配，将 192.168.11.2-192.168.11.100 的 IP 地址分配到 wlan1 上：



之后返回 Interface，可以看到最下面有一个 wlan1 的无线网卡，Type 为 wireless，我们点击 wireless 进入无线参数配置：

这里面有四个参数需要配置：

SSID：为身份验证 ID

Mode：无线发射的方式

Band：无线频段，即 802.11 协议类型选择

Frequency：无线发射频率

在这里配置 SSID 为“CDNAT”，这样无线局域网中所以需要连接的 ID 都将使用他。Mode 配置为：“ap-bridge”，即 AP 模式访问节点。Band 配置默认频段为 2.4GHz-b，这里我们可以选择 2.4GHz-b/g、2.4GHz-g-only 三个选项。2.4GHz-b 的带宽是 11M，2.4GHz-g-only 为 54M，如果你的无线网络内同时存在 b 和 g 的网卡，可以设置为 2.4GHz-b/g。

Frequency 是设置无线频道的，即发射的频率，支持 11 个频道。表示方法采用实际频率，2Ghz 频道范围：2412,2417,2422,2427,2432,2437,2442,2447,2452,2457,2462；从低到高为 1 到 11 频道。

在剩下的参数设置为默认即可，在最下面的 WiFi Portected Access(WPA)中配置无线的加密参数，这个可以根据你的需要来配置，这样可以避免其他无线网卡接收到你发射的信号。

最后根据无线局域网络需要配置如下：

现在我们可以通过笔记本或者台式机的无线网卡搜索无线信号，

然后通过配置 WindowsXP 的无线网卡，并点击连接 CDNAT 的无线网络。然后配置好 IP 就可以上网了。

我们在 Webbox 中的 RegTable 中可以看到无线网卡注册上的信息、MAC 地址和信号强度：

通过你可以使用 copy to access list 添加静态的访问列表，管理无线上网的用户。