MikroTik RouterOS

脚本编写和 WLAN 无线教程

Mikrotik 公司 1996 年成立于拉脱维亚，RouterOS 是 1997 年研发的专业路由系统，经历了多年的发展，已经发展到 6.0 版本。最初 MikroTik 开发 RouterOS 目的是解决无线局域网传输问题（WLAN），后来通过不断扩展功能实现了多种功能集于一身路由操作系统。在国内早期最大的使用人群是网吧、小区宽带和企业网络管理，这个和国外的情况有点差别，在国外 RouterOS 不仅用于解决路由管理，大多应用在 WLAN 的覆盖和传输，RouterOS 在基于 802.11abgn/ac 协议上的高带宽传输有自己的明显优势，特别是他独有的 Nstrem 和 Nv2 协议，近段时间国内的 WLAN 市场发展非常迅猛，对基于 RouterOS 开发的 RouterBOARD 产品需求也在不断增长。其实当前的 RouterOS 已经具备 1Gbps 网络环境的完整解决方案。通过脚本可以让 RouterOS 完成二次功能开发和应用，在设计上具备一定的开放性。

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

从 2003 年开始系统接触和使用 RouterOS，多年的学习和工作中积累了大量的实际经验，在这期间也得到广大 RouterOS 爱好者的支持和帮助，也是大家相互学习的过程。本套课程通过各方面的数据整理后编辑而成，从最基础的安装、登陆配置、多线路策略和流量控制、到诸如 WLAN 配置、VPN 实现、脚本编写等高级应用进行深入的讲解，适合于所有使用或学习 RouterOS 的朋友，在此基础上还有另外一套教程：《RouterOS 脚本编写和 WLAN 无线教程》，以及 MikroTik 配套的监控软件《The Dude 中文实用教程》。

从 RouterOS 6.0 版本开始，MikroTik 对 RouterOS 做大的改动，从内核优化、Queue 大改动、新的 Tilte 硬件构架等，核心改动较之前变化很大，性能提升明显。但对于 x86 平台的用户感觉没有 RouterBOARD 那么明显，因为后期的 RouterBOARD 加入了 FastPath 功能，能通过硬件快速转发数据报，这个是 x86 平台无法做到的，但从总体上内核的优化和 Queue 性能的改进也能带来不小的提升。

《RouterOS 脚本编写和 WLAN 无线教程》对 RouterOS 的脚本编写和 WLAN 的基础知识和常见问题做通俗的讲解，通过 RouterOS 来介绍 WLAN 网络的常见的应用，如覆盖、点对点、点对多点、中继、WDS 和 Mesh 网络，能让读者对 MikroTik 产品 WLAN 应用和配置等得到充分了解，当然这些基本的 WLAN 无线知识也可以应用到其他产品中。

该手册是提供对 MikroTik RouterOS 嵌入式脚本介绍，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。对于 RouterOS 的脚本操作，需要读者有一定的编程知识，而且对 RouterOS 各个功能熟悉和掌握。

学习 RouterOS 还是希望大家具备一定的路由交换的基础，这样看更容易理解，网络的路由交换是任何网络工程师入门的基础，因此我在教程里开始加入了一些基础知识，希望初步涉及网络的朋友能了解下，但这些基础知识还不够，需要大家自己去学习相关的路由交换内容，初级的网络工程师如 CCNA 或 HCNA 等，这些资料可以去看看和学习。

版本: V6.6 e
适用于: RouterOS v5.x v6.x
编写: 余 松

网站 www.irouteros.com
E-mail: athlon_sds@163.com
内容如有更新，恕不通知!

目 录

第一章 Scheduler（计划任务）

Scheduler 计划任务，通过设置定时或周期安排执行相应的脚本操作，计划任务能有效完成预期的任务，说的高级点就是基于 Scheduler 和 Script 两个功能，实现自己编写自己的脚本，实现智能路由器。

规格

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

1.1 计划任务介绍

计划任务列表通过调用脚本，并触发脚本执行，在指定的时间或者是在指定的时间周期执行任务，在计划任务调用脚本，可以设置脚本名称或执行写入脚本。

属性描述

interval (*时间*; 默认: **0s**) - 脚本执行的间隔周期时间，脚本将在指定的时间周期内反复执行。
name (*名称*) – 任务名称
on-event (*名称*) – 脚本执行名。通过调用**/system script** 里的脚本规则名称，也可以直接写入脚本。
run-count (*只读: 整型*) – 监视脚本使用数，这个计数器记录当每个脚本执行一次，计数器便增加 1
start-date (*日期*) – 开始脚本执行的日期
start-time (*时间*) – 开始脚本执行的时间
startup – 默认在系统启动 3 秒后执行脚本. 计划任务选项里对 start-time 设置了 startup，则在系统启动完成后 3 秒运行。

Run-count 记录了计划任务的执行次数，当路由器重启后，计数器会重置，如果有复杂的脚本执行模式，通常可能会涉及到计划多个脚本，在多个计划任务中切换，执行一个时禁用另外一个。

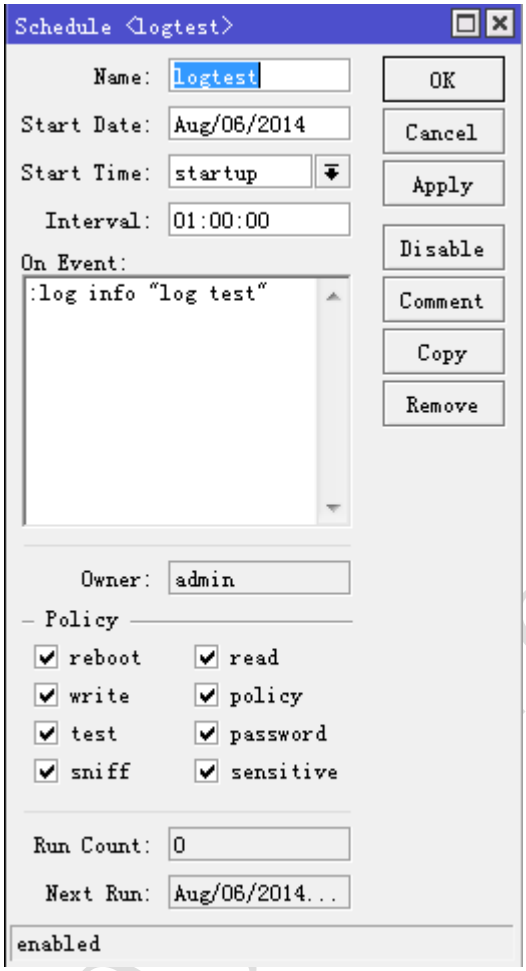
1.2 计划任务事例

通过下面的简单事例，介绍下计划任务在 RouterOS 是如何操作执行的。

事例 1: 我们添加一个任务执行系统日志记录测试，并间隔 1 小时执行一次，在 log 日志中显示“log test”，on-event 我可以直接写入 RouterOS 脚本

```
[admin@MikroTik] system script> add name=logtest on-event=:log info “test”
start-time=startup interval=1h
[admin@MikroTik] /system scheduler> print detail
Flags: X - disabled
 0    name="logtest" start-time=startup interval=1h on-event=:log info "log test"
owner="admin" policy=ftp,reboot,read,write,policy,test,winbox,password,sniff,sensitive,
        api run-count=5 next-run=09:05:19
```

在 winbox 中，将脚本直接配置到 Schedule 的 On-event 下：



事例 2：添加 2 个脚本改变带宽设置队列规则"ABC"，每天上午 9 点限制为 10Mb/s，下午 5 点限制为 20Mb/s。这个队列的规则。

先在 queue simple 添加一条对内网 ip 段 192.168.0.0/24 的流控规则(6.0 版本前配置流控目标 IP 为 target-addresses，6.0 开始为 target)：

```
[admin@MikroTik] /queue simple> add name=ABC target=192.168.0.0/24
max-limit=10M/10M
[admin@MikroTik] /queue simple> print value-list
              name: queue1
              target: 192.168.0.0/24
              parent: none
      packet-marks:
              priority: 8/8
               queue: default-small/default-small
              limit-at: 0/0
           max-limit: 10M/10M
          burst-limit: 0/0
    burst-threshold: 0/0
         burst-time: 0s/0s
        bucket-size: 0.1/0.1
```

然后配置脚本，并添加计划任务(**注**：在 2.9 之前定义脚本查找字符串是不需要加双引号的，但在 3.0 后中需要注明字符串，需加上双引号，如查找流控规则名”ABC”)，进入 script 脚本编辑器，添加两台脚本规则 start_limit 和 stop_limit

```
[admin@MikroTik] queue simple> /system script
[admin@MikroTik] system script> add name=start_limit source={/queue simple set [find
name=”ABC”] max-limit=10M/10M }
[admin@MikroTik] system script> add name=stop_limit source={/queue simple set [find
name=”ABC”] max-limit=20M/20M }
[admin@MikroTik] system script> print
  0 name="start_limit" source="/queue simple set [find name=”ABC”] max-limit=10M/10M "
    owner=admin run-count=0

  1 name="stop_limit" source="/queue simple set [find name=”ABC”] max-limit=20M/20M "
    owner=admin run-count=0
```

进入计划任务下使用 on-event 调用脚本编辑器中的两条脚本，可以直接在 on-event 中填写脚本名称

```
[admin@MikroTik] system script> .. scheduler
[admin@MikroTik] system scheduler> add interval=24h name="set-10M" \
\... start-time=9:00:00 on-event=start_limit
[admin@MikroTik] system scheduler> add interval=24h name="set-20M" \
\... 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-10M    start... oct/30/2008 09:00:00    1d                  0
  1    set-20M   stop_... oct/30/2008 17:00:00    1d                  0
[admin@MikroTik] system scheduler>
```

事例 3：下面是通过电子邮件发送每周备份路由器配置的脚本：

```
[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>
```

用 RouterOS 电子邮件发送，需要对你接收邮箱设置，即开启接收邮箱的 SMTP 服务，操作路径**/tool e-mail**
例如（**注：**建议是自己的 SMTP 服务器，一些正规网站的邮件服务器可能会将发送信息屏蔽）：

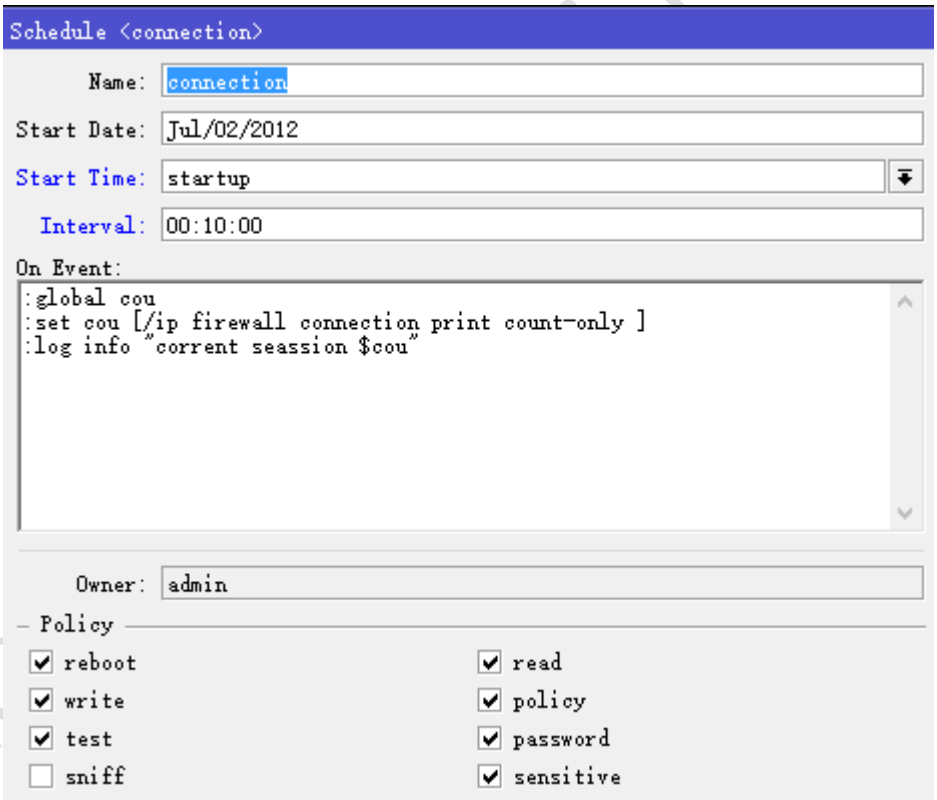
```
[admin@MikroTik] tool e-mail> set server=159.148.147.198 from=SysAdmin@host.com
[admin@MikroTik] tool e-mail> print
    server: 159.148.147.198
      from: SysAdmin@host.com
[admin@MikroTik] tool e-mail>
```

事例 4：下面的例子每 10 分钟统计一次 firewall connection 的会话数，并记录在 log info 里:

首先我们编写脚本，获取会话数量，并通过:log info 命令记录当前会话数

```
:global cou
:set cou [/ip firewall connection print count-only ]
:log info "corrent seassion $cou"
```

进入 schedule 新建一个计划任务取名 connection，设置 interval 为 10 分钟，On-event 加入脚本



以上的计划任务都涉及了 RouterOS 的 Script 脚本编辑，所以 RouterOS 要完成指定的计划任务，会编写脚本是必须的。

第二章 RouterOS Script（脚本）

在 RouterOS 中一个脚本配置构成由控制命令和表达式**(ICE - internal console expression 内部控制台表达式)**。

RouterOS 操作命令，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，即通过**“/”**来达到命令执行的目的。在脚本 **ICE** 表达式前缀需要用 **“:”** 并能在任何目录路径下操作。

一个事件（event）用来触发脚本执行，在 RouterOS 下包括：System Scheduler, Traffic Monitoring Tool, Netwatch Tool 。

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

注： RouterOS2.9 本版与 RouterOS 3.0 的脚本有一定的区别：

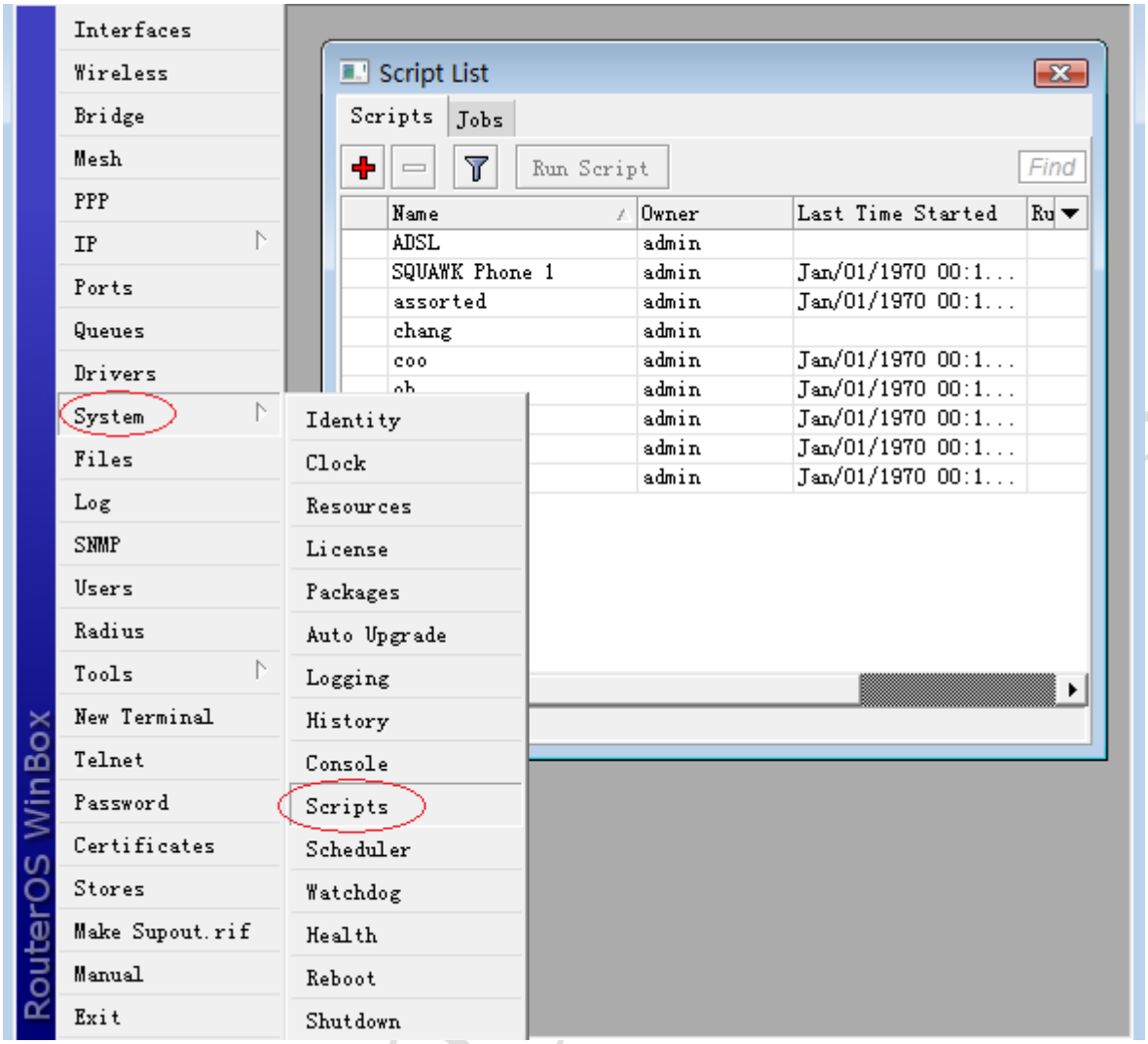
1、RouterOS3.0 字符参数后需要使用引号注明
如 **comment=”test”；name=”pppoe-out1”；:set i $”tx-current”**

2、RouterOS3.0 的变量定义不支持“中横杠”的定义
如 :**global test-address** 这样定义在 3.0 和 4.0 中是非法的

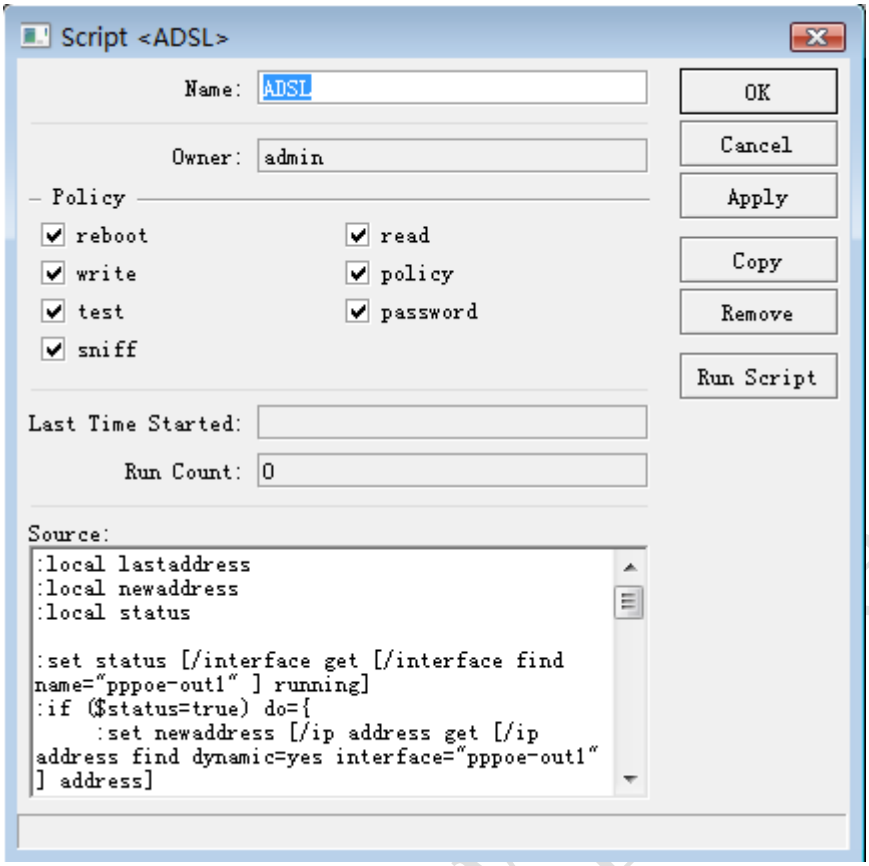
2.1 RouterOS 脚本基本操作

操作路径：`/system script`

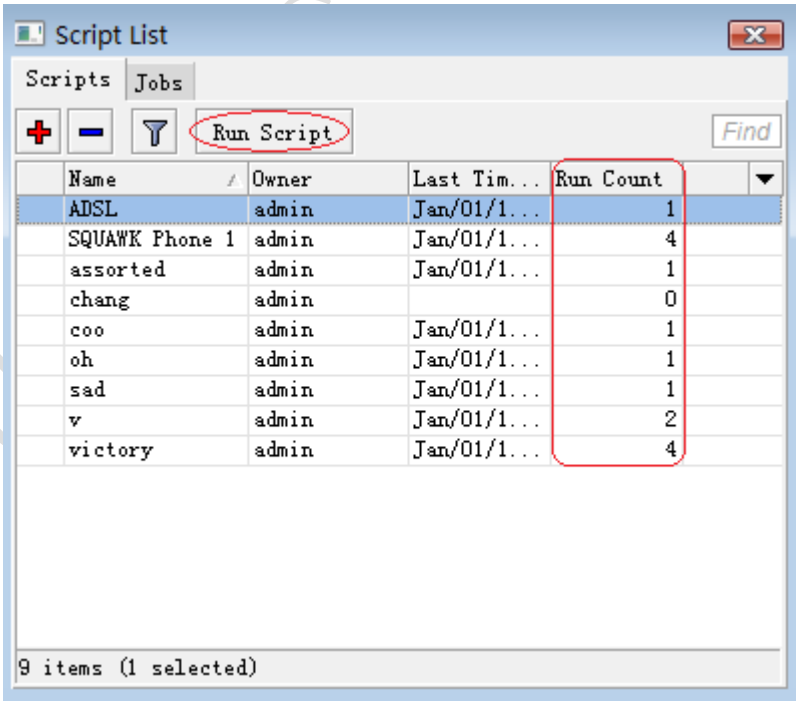
脚本编写进入 script 目录下，在 script 中我们可以定义多条脚本规则,如下图：



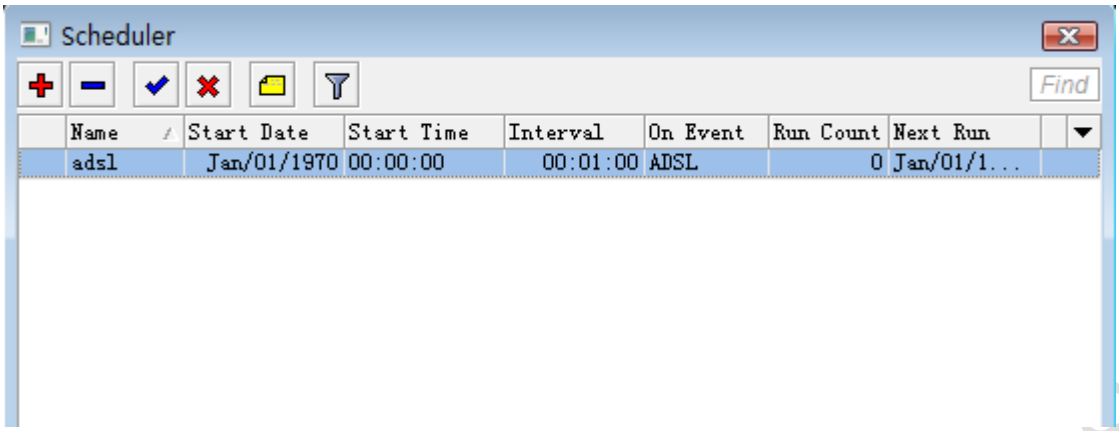
我通过 script 编辑器编辑脚本：



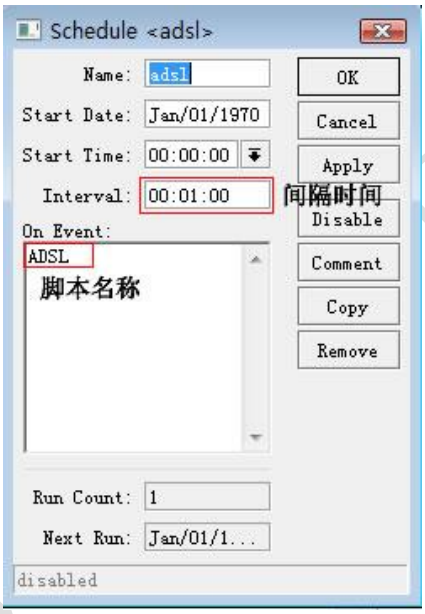
我们可以通过 Run Script 命令运行当前的脚本，在 Run Count 中会纪录脚本运行的次数：



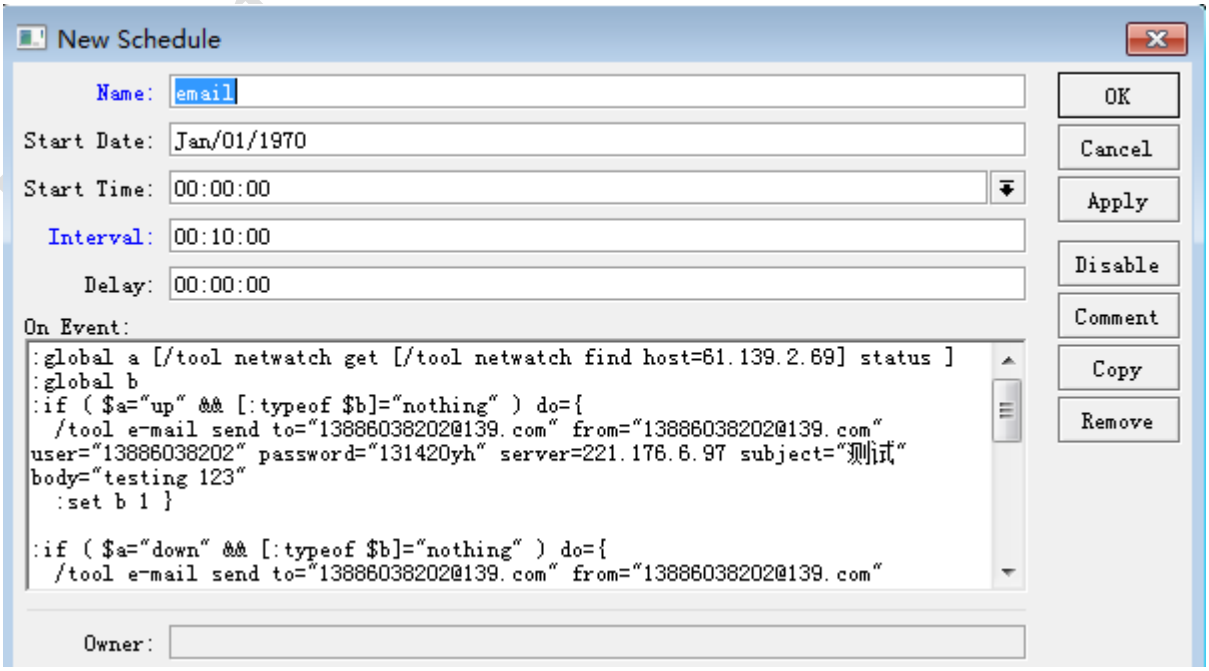
在 Script 中我们编辑好脚本后，我们可以通过 RouterOS 相应的功能调用脚本，并执行这些脚本，如/system scheduler（计划任务）、/tool 中的 netwatch、traffic-monitor 等。通常执行脚本在 scheduler 计划任务中最常用，如这里我们使用了 ADSL 脚本，需要每间隔 1 分钟执行一次，我们可以通过 scheduler 来完成，通过/system scheduler 进入计划任务目录：



添加一个名称为 adsl 的计划任务规则，设置 Interval 时间 1 分钟，On Event 中添加脚本名称：



注：我们也可以在 on event 中直接输入脚本的命令：



2.2 RouterOS 脚本语法

RouterOS 脚本被划分为多个命令行，命令行是一个接一个单元执行直到脚本结束或者直到 错误发生。

命令行结构

RouterOS 控制台是使用下面的命令语法：

[前缀] [路径] 命令 [未命名参数] [参数=[值]] .. [参数=[值]]

- **[前缀]** – 如果命令是 ICE 或者路径通过":" 或者 "/" 字符表示
- **[路径]** – 得到操作菜单的路径
- **命令** – 一个命令获取在指定的菜单路径下
- **[未命名参数]** – 即事先定义参数，如果命令需要必须指定该参数
- **[参数]** – 按先后顺序各自定义值

命令行结束以**“;”**标识为代表或者换行，在结束命令行有时不需要**“;”**或者换行

独立的命令包含 **(), []** 或者 **{}**不需要任何的结尾命令字符，命令结尾取决于脚本的内容

```
:if ( true ) do={ :put "lala" }
```

每条命令行包含其他命令行，起始通过方括号定义"[ ]"

```
:put [/ip route get [find gateway=1.1.1.1]];
```

注意，上面这条代码包含 3 条命令行：

- :put
- /ip route get
- find gateway=1.1.1.1

命令行能通过多余一个行的方式建立，可以查看后面的行连接规则．几种常见的命令实例

- **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 命令操作

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

命令分解

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

For 循环

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

命令分解：

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

多网卡流量查看

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

命令分解：

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

变量

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。

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] >
```

注释

一个注释从“#”号字符开始执行，并结束在一行的结尾，空格或者任何其他标示不允许在#标示之前。如果“#”字符出现在一个字符串中将不会考虑为一个注释内容。

```
# this is a comment （正确注释）
 # bad comment   （错误注释）
:global a; # bad comment  （错误注释）

:global myStr "lala # this is not a comment"   （不是注释）
```

行连接

通过“\”字符可以将两行或者多行连接到一个逻辑行。一行以反斜杠结束不能进行注释，即一个反斜杠不能连接一个注释。一个字符串不会继续一个指令。

```
:if ($a = true \
     and $b=false) do={ :put “$a $b”; }
```

```
:if ($a = true \      # bad comment  （错误注释）
     and $b=false) do={ :put "$a $b"; }

# comment \
    continued - invalid  (语法错误)
```

指令之间的空格

空格可以用于分隔指令。空格必须在两个指令之间仅在他们一系列互相关联的事物解释为一个不通过的指令，例如：

```
{
   :local a true; :local b false;
# 空格不需要
   :put (a&&b);
# 空格需要
   :put (a and b);
}
```

空格不允许

- 在 '<参数>=' 之间不允许
- 在'from=' 'to=' 'step=' 'in=' 'do=' 'else=' 之间不允许

```
#错误:
:for i from = 1 to = 2 do = { :put $i }

#正确:
:for i from=1 to=2 do={ :put $i }
:for i from= 1 to= 2 do={ :put $i }

#错误
/ip route add gateway = 3.3.3.3

#正确
/ip route add gateway=3.3.3.3
```

关键字

下面的字符是关键字，不能用作变量和功能名称：

```
and      or       not      in
```

分隔符

下面记号作为分隔符的语法：

```
()  []  {}  :  ;  $  /
```

数据类型

RouterOS 脚本语言有以下数据类型：

类型	属性
number	- 64bit 整型，支持十六进制输入；
boolean	- 布尔型，真与假；
string	- 字符型；
IP	- IP 地址；
内部 ID	- 十六进制，前缀通过‘*’标记。每个菜单目录下都被分配唯一的数字——内部 ID；
time	- 日期与时间值；
array	- 一个数组有序的值；
nil	- 如果没有值被分配，变量的默认值

命令替换和返回值

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

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

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。
-> - 通过关键字获取数组成员

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。

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

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] >
```

操作数组

RouterOS v6 支持对数组成员定义关键字 key，通过”->”方式获取，操作如下：

```
[admin@x86] >:global aaa {a=1;b=2}
[admin@x86] > :put ($aaa->"a")
1
[admin@x86] > :put ($aaa->"b")
```

注意：数组元素定义关键 key，与小写字母的字符有区别，数组关键字 key 必须放在引号中，例如：

```
[admin@ce0] > {:local a { "aX"=1 ; ay=2 }; :put ($a->"aX")}

1
```

数组元素值输出

Foreach 命令能通过依次查询数组元素和关键 key，并输出：

```
[admin@ce0] > :foreach k,v in={2; "aX"=1 ; y=2; 5} do={:put ("$k=$v")}

0=2
1=5
aX=1
y=2
```

当 foreach 命令使用一个值，元素会逐一返回 ：

```
[admin@ce0] > :foreach k in={2; y=2; "aX"=1 ; 5} do={:put ("$k")}

2
5
1
2
```

上面的输出顺序大家都看到了，感觉有的奇怪，因为数组会首先查询普通元素，然后再是关键 key，关键 key 输出时是按照字母顺序执行，因此我们看到普通元素首先被输出，然后是关键 key，关键 key 又按照字母顺序输出结果，

修改单个数组元素值

```
[admin@MikroTik] > :global a {x=1; y=2}
[admin@MikroTik] > :set ($a->"x") 5
[admin@MikroTik] > :environment print
a={x=5; y=2}
```

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] > :
environment  do       for       led     nothing  resolve  tobool   tonum   while
terminal     error    foreach   len     parse    set      toid     tostr
beep         execute  global    local   pick     time     toip     totime
delay        find     if        log     put      toarray  toip6    typeof
```

常用命令属性

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 – 根据条件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) – 执行的操作

```
[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) – 定义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) – 定义循环计数器变量名称。
- in (数组列表) – 数组列表范围或路径
- 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] >
```

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) – 配合 while 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] >
```

Typeof – 判断变量类型，返回值 num、str、nothing、time、ip、bool

输入参数:

unnamed(name) – 变量名称

判断一个变量的类型

```
[admin@MikroTik] >:global a 192.168.1.1
[admin@MikroTik] > :put [:typeof $a]
ip
[admin@MikroTik] >:global b
[admin@MikroTik] >:put [:typeof $b]
nothing
```

2.3 Scripte 事例

启动延迟

如果你的脚本依赖接口相关的配置，可能有 RouterOS 刚启动时无法执行的情况，建议延迟执行或者检查所有需要的接口可以获取的时候执行，下面的脚本定义有 10 个接口，设置等待时间 30 秒，在 30 秒内没有 10 个接口，就在日志中显示接口没有加载完

```
#初始化变量 i
:local i 0
#接口总数量
:local x 10
#最大等待时间 30
:local t 30
#while 循环如果 i 小于 30，且接口数量小于 10 执行
```

```
while ($i < $t && [:len [/interface find]] < $x) do={
:put $i
:set $i ($i + 1)
:delay 1
}
#如果 i 等于 t，就说明接口没有加载完，反之加载完成可以执行后续脚本
if ($i = $t) do={
:log warning message="Could not load all physical interfaces"
} else={
#执行你的后续脚本
}
```

自动创建多条策略

在 firewall **input** 规则中通过脚本添加接受从 1.1.1.1 开始到 1.1.1.100 地址的数据包，即 100 条规则 ：

```
:for e from 1 to 100 do={
  /ip firewall filte add \
    chain=input src-address=(“1.1.1.” . $e)
   }
```

我们使用类似的脚本编写，添加 200 个 IP 地址的流量规则：

```
:for e from 1 to 200 do={
  /queue simple add target-addresses=("192.168.1." . $e) max-limit=256000/512000
}
```

获取 bandwidth 测试参数

这个事例描述的是如果获取 **bandwidth-test** 命令的结果。 在事例中使用 global（全局变量）另外一个脚本在同一时间运行，并获取当前的 TX 参数。

```
  :global i
  /tool bandwidth-test 1.1.1.1 direction=transmit duration=14s
    do={
      :if ($status="running") do={
        :set i $”tx-current”
        }
      }
  }
```

创建一个文件

在 v3.x 不能直接创建文件，然而有一种变通方法，我们创建一个 myFile 文件，默认是 txt 格式，通过 set 命名可写入相应的内容

```
/file print file=myFile
/file set myFile.txt contents="123"
```

检查 IP 地址在一个接口上是否改变

有时运营商提供动态的 IP 地址，这个脚本会比较动态 IP 地址是否改变。

```
:global currentIP;

:local newIP [/ip address get [find interface="ether1"] address];

:if ($newIP != $currentIP) do={
    :put "ip address $currentIP changed to $newIP";
    :set currentIP $newIP;
}
```

分离子网掩码

下面的脚本可用于分离子网掩码的操作：

```
方法 1
:global ipaddress 10.1.101.1/24

:for i from=( [:len $ipaddress] - 1) to=0 do={
      :if ( [:pick $ipaddress $i] = "/") do={
              :put [:pick $ipaddress 0 $i]
      }
}

方法 2
:set ipaddress [:pick $ddns-ip 0 ([:len $ddns-ip] - 3)]

方法 3
:set ipaddress [:pick $ipaddress 0 [:find $ddns-ip "/"]]
```

域名解析

许多用户通过解析 DNS 域名来替换 Radius 服务器、VPN 服务器和防火墙等 IP 地址，例如下面是如何解析一个 Radius 服务器域名的 IP 地址。下面是配置一个 Radius 服务器连接，配置内容是如下：

```
/radius
add address=3.4.5.6 comment=myRad
```

这里的脚本将 server.example.com 的域名解析为 IP 地址，比较之前解析的 IP 地址，如果不相同则进行替换：

```
/system script add name="resolver" source= {

:local resolvedIP [:resolve "server.example.com"];
:local radiusID [/radius find comment="myRad"];
:local currentIP [/radius get $radiusID address];

:if ($resolvedIP != $currentIP) do={
   /radius set $radiusID address=$resolvedIP;
   /log info "radius ip updated";
}

}
```

这个脚本将在 scheduler 每间隔 5 分钟运行一次：

```
/system scheduler add name=resolveRadiusIP on-event="resolver" interval=5m
```

产生备份文件并通过 e-mail 发送

这个脚本产生备份文件，并发送到规定的 e-mail 地址，Mail 题目包含路由器的名字，当前日期和时间
注意：在脚本执行前，SMTP 服务器必须配置。查看在/tool e-mail 的配置选项

```
/system backup save name=email_backup
/tool e-mail send file=email_backup.backup to="me@test.com" body="See attached
file" \
   subject="$[/system identity get name] $[/system clock get time] $[/system clock
get date] Backup")
```

解析域名 IP 地址，并添加入 address-list

这个实例对一些域名有多 IP 地址的网站进行解析，我们可以通过设置计划任务每间隔一个周期执行脚本，比如 netbar.qq.com 进行解析，每次解析对比 ip firewall address-list 是否存在相同 IP，如果没有相同 IP 地址，则添加入 address-list。

```
:global a [:resolve netbar.qq.com]
:global b
:foreach i in=[/ip firewall address-list find list=qqgame] do={
   :if ($a = [/ip firewall address-list get $i address ]) do={
    :set b 1
  }
   else={
    :set b 0
  }
```

```
}
:if ($b = 0) do={ /ip firewall address-list add list=qqgame address=$a }
```

使用注释

如果有时候在一个规则中与其他规则许多属性相同，指定的参数就无法从规则中获取（例如：firewall 或者 routing 规则等）。这里我们可以通过编辑注释来解决。I

假设，我们需要从 ip firewall nat 中修改端口映射的目标地址，目标 IP 地址是 218.16.18.12，而我们做了多条端口映射规则，为方便查找和修改，我们可以通过注释标记。

```
[admin@MikroTik] ip firewall nat> set 0 comment=dst1
```

现在我们看看一个具体的应用实例，ADSL 的 IP 地址变动后，我们修改端口映射的 IP 地址：

```
:global adsl "pppoe-out1"   # 定义 ADSL 拨号接口名称变量: pppoe-out1
:global adsllast       # 之前的 ADSL IP 地址变量

:global adslip [ /ip address get [/ip address find interface=$adsl] address ] //
获取当前 ADSL ip 地址

:if ([ :typeof $adsllast ] = nil ) do={ :set adsllast 0.0.0.0/0 } # 判断之前的 ADSL
IP 地址是否为空,如果为空设置一个 0.0.0.0/0 的 IP 地址

:if ([ :typeof $adsl ] = nil ) do={

  :log info ("No ip address present on " . $adsl . ", please check.") # 判断当前
ADSL IP 是否为空,否则执行 else 的操作

} else={

  :if ($adslip != $adsllast) do={
    :log info [/ip firewall nat set [/ip firewall nat find comment = "dst1"]
dst-address= $adslip ]    #  判断当前 IP 和之前 IP 是否相同，如果不同便修改 dst-address
的 IP 地址
    :log info "ADSL IP: UPDATE!"

    :set adsllast $adslip   # 交换 IP 地址

  } else={

    :log info "ADSL IP: No change"
 }

}
```

在这个事例中，我们可以看到通过添加注释，为编写脚本判断正确的规则。

DDNS 动态域名配置

这里我们来看看，当我们使用 ADSL 时由于重新拨号或者租约到期，IP 地址是动态变化的，而又需要让外网的客户通过动态域名来访问我们的内部服务器,这里我们需要使用到 DDNS 的动态域名脚本,注意 RouterOS 支持的动态域名服务器只有 www.changeip.com

```
:log info "DDNS: Begin"    #在 log 日志中显示 DDNS 开始运行

:global ddnsuser "ddns.test.com"   # 定义 ddnsuser 用户名变量
:global ddnspass "cdnat"              # 定义 ddnspass 密码变量
:global ddnshost "ddns.test.com"   # 定义 ddnshost 主机
:global ddnsinterface "pppoe-out1"  # 定义 ADSL 拨号接口，用户获取 IP
:global ddnsipt  #定义一个零时存储的 IP 地址变量
:global ddnsip [ /ip address get [/ip address find interface=$ddnsinterface]
address ]  # 根据 ADSL 拨号的接口获取 IP 地址，并分配给 ddnsip

:if ([ :typeof $ddnslastip ] = nil ) do={ :global ddnslastip 0.0.0.0/0 }  # 查
看 ddnslastip 变量是否为空，如果为空则分配 0.0.0.0/0 地址
:if ([ :typeof $ddnsip ] = nil ) do={

  :log info ("DDNS: No ip address present on " . $ddnsinterface . ", please check.")
# 查看 ddns-ip 变量是否为空，如果为空则在 log 中提示未获取 IP

} else={

  :if ($ddnsip != $ddnslastip) do={                                    # 否则执行
新 IP 与以前的 IP 地址对比
    :set ddnsipt [:pick $a 0 ([:len $a] - 3)]                          # 去掉 IP
地址后面的子网掩码，并将修改后的 IP 分配给 ddns-ipt 变量
    :log info [/ip fi nat set [/ip fi nat find comment = s1] to-address $ddnsipt ]
# 根据注释名称修改相应的 nat 规则
    :log info [/ip fi nat set [/ip fi nat find comment = a1] dst-address $ddnsip ]
    :log info [/ip fi nat set [/ip fi nat find comment = a2] dst-address $ddnsip ]
    :log info [/ip fi nat set [/ip fi nat find comment = a3] dst-address $ddnsip ]
    :log info [/ip fi nat set [/ip fi nat find comment = a4] dst-address $ddnsip ]
    :log info [/ip fi nat set [/ip fi nat find comment = a5] dst-address $ddnsip ]
    :log info [/ip fi nat set [/ip fi nat find comment = a6] dst-address $ddnsip ]

    :log info "DDNS: Sending UPDATE!"     # log 中提示 DDNS 更新
    :log info [ /tool dns-update name=$ddnshost address=[:pick $ddnsip 0 [:find
$ddnsip "/"] ] key-name=$ddnsuser key=$ddnspass ]
    # 发送最新的 IP 地址，到 DDNS 服务器
    :global ddnslastip $ddnsip   # 将新老 IP 地址交换

  } else={
    :log info "DDNS: No change"
```

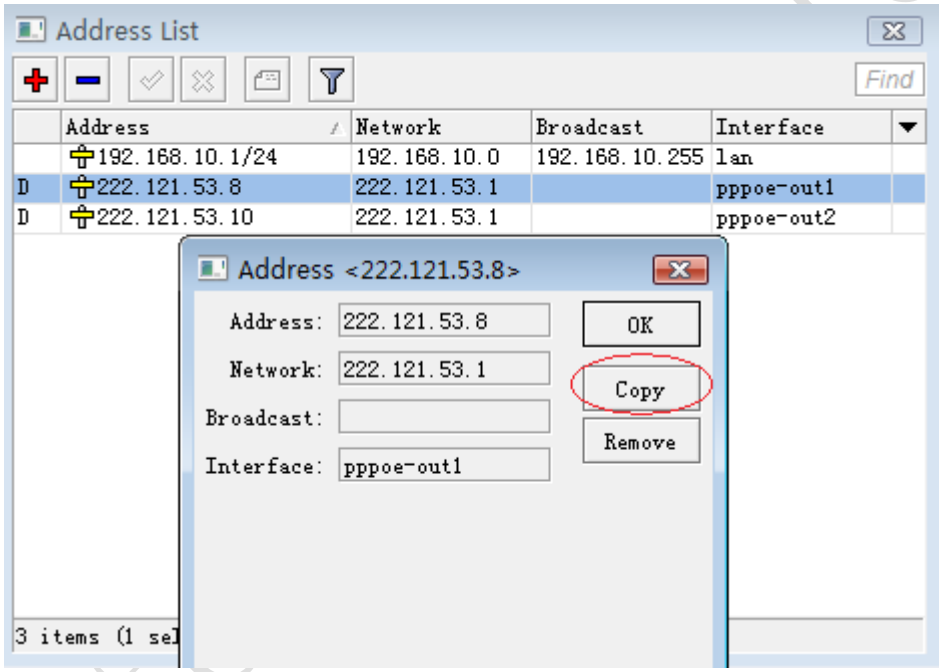
```
 }
}
:log info "DDNS: End"
```

相同 ADSL 网关脚本修改

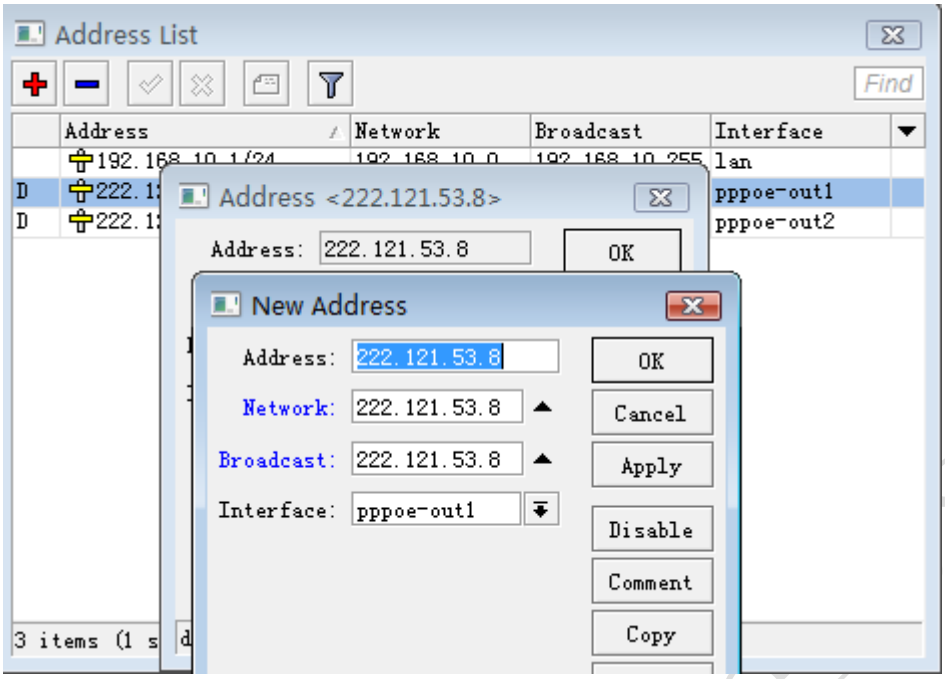
注： 该脚本适用于 v3.0 版本，v4.0 在相同 ADSL 网关下可以通接口设置，已经不需要脚本。

当我们使用到多线路的 ADSL 时，可能会遇到这些 ADSL 的网关是相同的，因为 RouterOS 只能识别不同网关的策略路由，因此在多 ADSL 相同网关的情况下，我们可以通过隧道协议的特点，用本地 IP 地址做为路由器的网关。

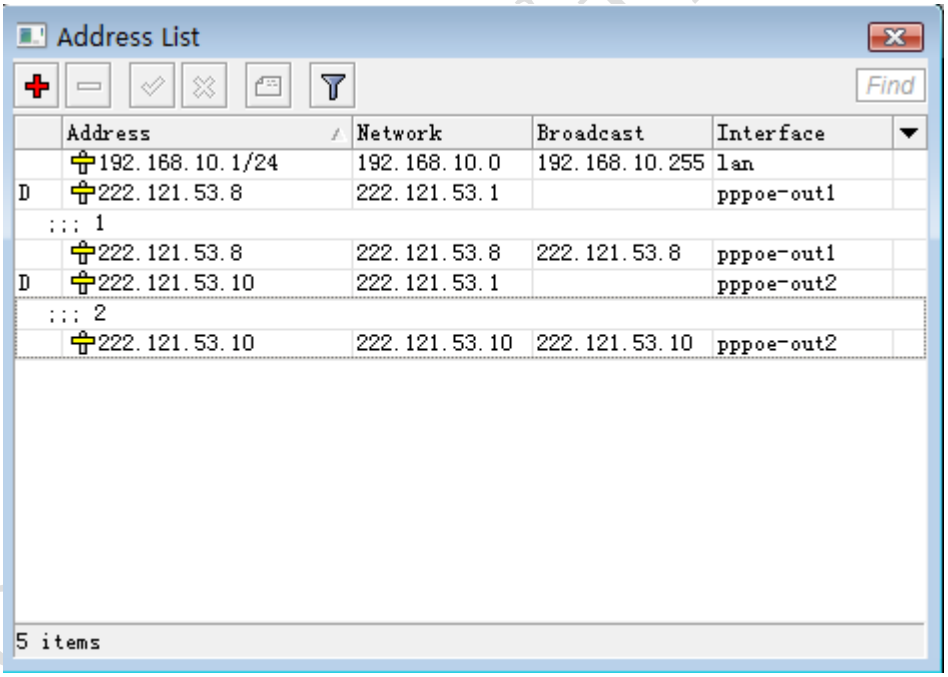
假设我们有 2 条 ADSL，分别为 pppoe-out1 和 pppoe-out2，做 ADSL 操作的时候需要将动态获取的 ADSL 地址修改为静态，通过在/ip address 中使用 copy 命令操作，如下图



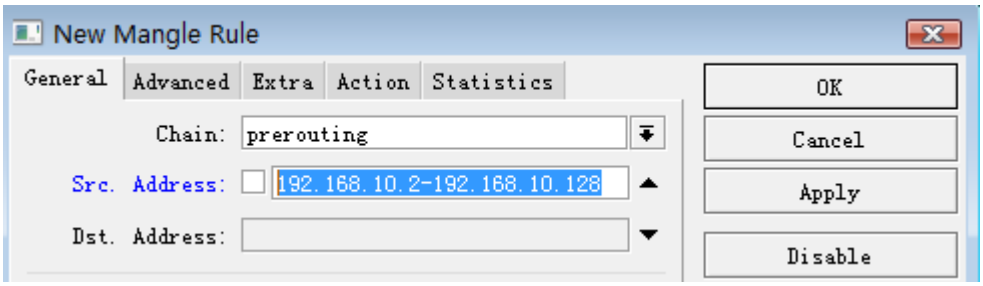
上图获取到的 IP 地址是 222.121.53.8，通过 copy 命令修改 Network 和 Boradcast 地址也为 222.121.53.8，如下：

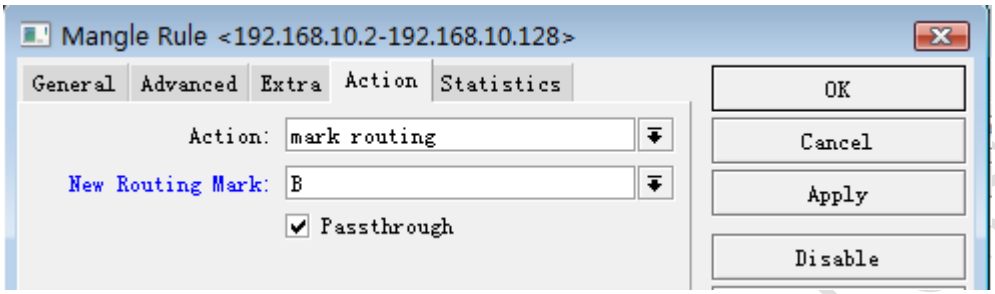


将下面 2 条 ADSL 的 IP 地址都修改为静态的，并给静态的 IP 地址标记上注释 1 和 2：



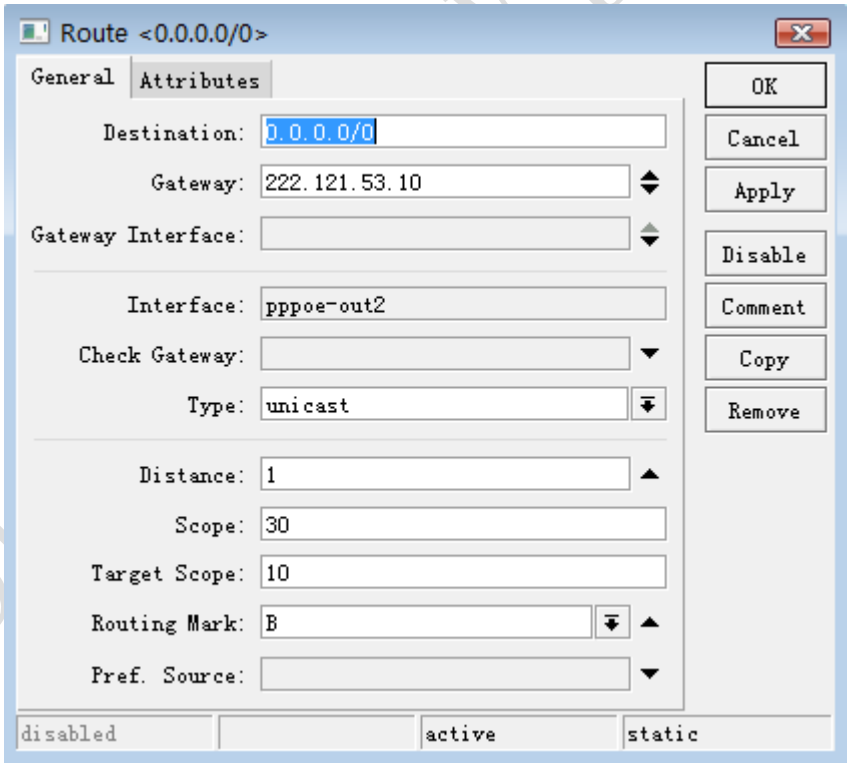
我们将网络的内的用户平均分组为 A 和 B，A 组走 1 号 ADSL 线路，而 B 组走 2 号 ADSL 线路。1 号 ADSL 设置为默认路由，即 RouterOS 的默认网关出口，这里我们将 2 号线作为用户 B 组的策略路由，我们在 ip firewall mangle 中配置，定义 chain=prerouting src-address=192.168.10.2-192.168.10.128 action=mark-routing new-routing-mark=B



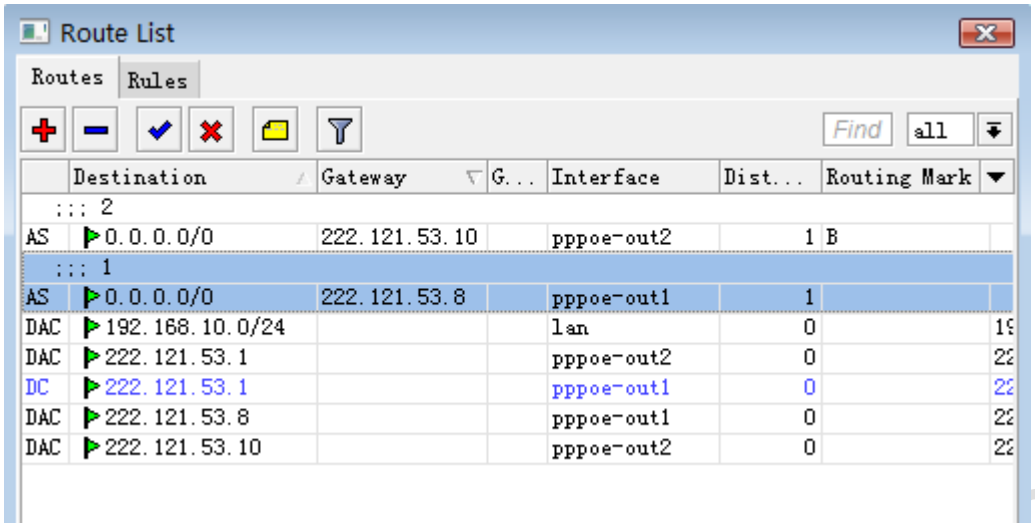


这里我们只需要定义 B 组用户，A 组用户只需要走 1 号 ADSL 的默认网关，定义 B 组用户走 2 号线路的 ADSL

配置路由，这里我们设置 1 号 ADSL 的 IP222.121.53.8 为默认网关，即 ADSL 的默认网关。2 号线路的 ADSL 我们用 222.121.53.10，做为 B 组线路的网关，并在 ip route 中添加 routing-mark 的标记



为 pppoe-out2 做注释标记 2，配置完成后的路由，B 组用户通过 22.121.53.10 的网关出去，剩下的用户通过默认网关如下图



接下来我们需要通过脚本，来判断 1 和 2 号线 ADSL 的 IP 地址是否变动，如果 IP 变动后用脚本会自动修改变动的 1 和 2 号线 ADSL 参数

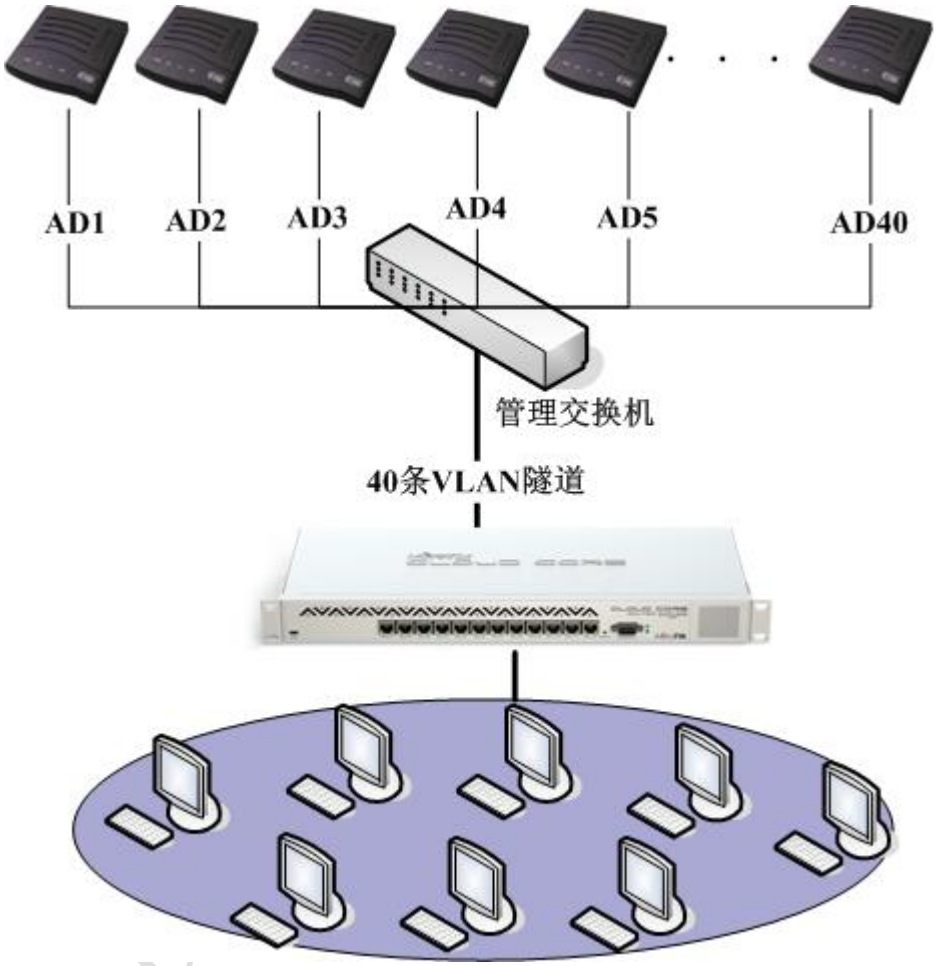
```
:local lastaddress
:local newaddress
:local status

:set status [/interface get [/interface find  name="pppoe-out1" ] running]
:if ($status=true) do={
     :set newaddress [/ip address get [/ip address find dynamic=yes
interface="pppoe-out1" ] address]
     :set newaddress [:pick $newaddress 0 [:find $newaddress "/"]]
     :set lastaddress [/ip address get [/ip address find dynamic=no
interface="pppoe-out1"] address]
     :set lastaddress [:pick $lastaddress 0 [:find $lastaddress "/"]]
     :if ($lastaddress != $newaddress) do={
          :log info [/ip address set [/ip address find comment="1"]
address=$newaddress network=$newaddress broadcast=$newaddress]
         :log info [/ip route set [/ip route find comment="1"] gateway=$newaddress]
         }
 }
:set status [/interface get [/interface find  name="pppoe-out2" ] running]
:if ($status=true) do={
     :set newaddress [/ip address get [/ip address find dynamic=yes
interface="pppoe-out2" ] address]
     :set newaddress [:pick $newaddress 0 [:find $newaddress "/"]]
     :set lastaddress [/ip address get [/ip address find dynamic=no
interface="pppoe-out2"] address]
     :set lastaddress [:pick $lastaddress 0 [:find $lastaddress "/"]]
     :if ($lastaddress != $newaddress) do={
          :log info [/ip address set [/ip address find comment="2"]
address=$newaddress network=$newaddress broadcast=$newaddress]
         :log info [/ip route set [/ip route find comment="2"] gateway=$newaddress]
```

```
        }
}
```

40 条线路负载均衡配置与脚本

RouterOS 支持多线路的负载均衡，某小区为了节约费用，采用 40 条 2M 带宽的 AD 通过做汇聚实现高带宽的小区带宽，为解决接口问题采用一台 Cisco 的 48 口的交换机做 VLAN 接入 40 条 AD，让后通过 VLAN 连接到 RouterOS 进行拨号，再做 PCC 负载均衡，网络拓扑图如下：



外网接入的方法是在交换机和 RouterOS 路由器上划分 VLAN，然后在 ROS 对应的 VLAN 上做 PPPoE-CLIENT。

1、首先划分 VLAN（我们这里是从 2 开始排序的），脚本如下：

```
[admin@MikroTik] > :for i from=2 to=41 do= {interface vlan add name=("vlan".$i)
vlan-id=$i interface=ether2-wan }
```

2、然后添加 PPPOE 拨号（先添加拨号再手动输入 每个 AD 的帐号和密码，40 条 AD 设置还是要花点时间了），脚本如下：

```
[admin@MikroTik] > :for i from=2 to=41 do= {interface pppoe-client add name=("pp
poe-out".$i) user=$i password=$i interface=("vlan".$i)}
```

3、我们这里采用 PCC 的负载均衡，在 ip firewall mangle 里添加相应的 PCC 规则，通过一些脚本添加 PCC 的规则，注意：如果 PPPoE 客户端拨号没有成功，那么添加的规则则为红色的，拨号成功后自动正常

```
[admin@MikroTik] > :for i from=2 to=41 do={/ip firewall mangle add chain=input
a
ction=mark-connection new-connection-mark=conn1 in-interface=("pppoe-out".$i)}
```

4.然后标记路由让从哪个接口进来的数据就从哪个接口出去：

```
[admin@MikroTik] >  :for i from=2 to=41 do= {ip firewall mangle add chain=output
 connection-mark=("conn".$i) action=mark-routing new-routing-mark=("rout".$i)}
[admin@MikroTik] >
```

5．然后将所有内网出来的数据通过 pcc 的 both-addresses 分成 40 分并标记连接和路由：

```
[admin@MikroTik] > :for i from=2 to=41 do= {/ip firewall mangle add chain=prerou
ting src-address-list=lan-add action=mark-connection
new-connection-mark=("conn"
.$i) per-connection-classifier=("both-addresses:40/".$i) comment=$i
{... /ip firewall mangle add chain=prerouting src-address-list=lan-add action=ma
rk-routing new-routing-mark=("rout".($i-2)) connection-mark=("conn".$i)}
```

6.为每个路由标记添加路由并添加 pppoe-out41 为默认路由：

```
[admin@MikroTik] > :for i from=2 to=41 do= {ip route add dst-address=0.0.0.0/0
g
ateway=("pppoe-out".$i) routing-mark=("rout".$i)}
[admin@MikroTik] >ip routed add dst-address=0.0.0.0/0 gateway=pppoe-out41
```

7．最后做 NAT 伪装，一般最好是对每个出口进行伪装：

```
[admin@MikroTik] > ip firewall nat add chain=srcnat action=masquerade
```

PCC 负载均衡脚本

为了方便快速设置，我们可以通过脚本进行规则的循环添加：

```
:global interface "ether1-lan"
:global pppoe "pppoe-out"
:global address "192.168.0.0/24"
:global n 6
:for i from=1 to=$n do={
        :log info "a1"
        /ip firewall mangle add chain=input in-interface=($pppoe . $i)
action=mark-connection new-connection-mark=("pcc" . $i)
        :log info "a2"
```

```
        /ip firewall mangle add chain=prerouting src-address=$address
per-connection-classifier=("both-addresses:" . $n . "/" . $i-1)
dst-address-type=!local action=mark-connection new-connection-mark=("pcc" . $i)
        /ip firewall mangle add chain=prerouting src-address=$address
connection-mark=("pcc" . $i) action=mark-routing new-routing-mark=("route" . $i)
        /ip firewall mangle add chain=output connection-mark=("pcc" . $i)
action=mark-routing new-routing-mark=("route" . $i)

}
```

array 数组

RouterOS 的数组使用和其他语言大同小异，简单介绍下数组的使用

一个数组的定义如下，使用全局变量定义数组 array，包括元素 1,2,3,4

```
:global array {1;2;3;4}
```

输出数组的值

```
:put $array
```

输出第 2 个元素值，记住数组序列是从 0 开始计数

```
:put [:pick $array 1]
```

我们也可以将一个元素赋值给一个变量

```
:global tmp [:pick $array 2]
```

我们可以通过 len 得到数组的长度，并输出

```
:put [:len $array]
```

通过:len 得到该数组有多少个元素，这样有利于我们后面一些代码的处理，比如 for

```
:for i from=0 to=[:len $array]
```

我们也可以使用 foreach 循环取值

```
:foreach i in=[/ip arp find] do={
:set arraymac ($arraymac ,[/ip arp get $i mac-address ])
:set arraynum ($arraynum , $i)
}
```

下面是一个通过比较在 arp 列表里是否存在相同 mac 地址并删除的实例（脚本写的很丑，请谅解），注意定义数组变量时最好不要用全局变量，因为脚本执行完后，全局变量是不会被清空的，下次执行时会重复追加元素，一般使用局部变量 local 定义：

```
# 定义 mac 数组

:local arraymac

# 定义标号数组

:local arraynum

# 相同 mac 的标号数组

:local arrays

# 第一个相同 mac 标号变量

:local macfirst
:local n
:local n1

# 取 arp 下的 mac 地址和他们在 arp 列表中的标号

:foreach i in=[/ip arp find] do={
:set arraymac ($arraymac ,[/ip arp get $i mac-address ])
:set arraynum ($arraynum , $i)
}

#取得数组长度

:set n [:len $arraymac]

#双循环比较

:for m from=0 to=($n-1) do={
:for k from=1 to=$n do={
:if ([:pick $arraymac $m] = [:pick $arraymac $k]) do={
:set macfirst [:pick $arraynum $m]
:set arraySN ($arraynum , [:pick $arraynum $k])
}
}
}

#取得数组长度

:set n1 [:len $arraySN]

#删除 arp 列表中相同的 mac 地址

:for j from=0 to=$n1 do={
```

```
/ip arp remove [:pick $arraySN $j]
}
/ip arp remove $macfirst
```

数组的使用还有很多方面，而且非常有用，这个就需要各位在实践中摸索！

通过声控判断 WLAN 信号强度

我们可以使用 RouterOS 的脚本 beep 语句，配合循环操作，来判断 WLAN 的信号强度，该脚本主要应用在点对点的 WLAN 搜索信号使用。

下面是 ap-bridge 使用的声控信号强度脚本，注意 ap-bridge 使用的时候需要填写对方无线模块的 MAC 地址，才能获取信号强度

```
:local beep "10ms";
:local s85 "1350ms";
:local s80 "850ms";
:local s75 "650ms";
:local s70 "450ms";
:local s65 "350ms";
:local s60 "250ms";
:local s55 "200ms";
:local s50 "150ms";
:local s45 "100ms";
:local s40 "60ms";
:local s20 "20ms";
:global fr
:for i from=1 to=50 do={
:set fr [/interface wireless registration-table get [/interface wireless
registration-table find radio-name="000C4223D23E"] signal-strength ]
:set fr [:pick $fr 0 [:find $fr "d" ]]
 :if ($fr <= -85 && $fr > -88) do={
  :for i from=1 to=2 do={ :beep length=$beep; :delay $s85; }
  }
 :if ($fr <= -80 && $fr > -85) do={
  :for i from=1 to=3 do={ :beep length=$beep; :delay $s80; }
  }
 :if ($fr <= -75 && $fr > -80) do={
  :for i from=1 to=3 do={ :beep length=$beep; :delay $s75; }
  }
 :if ($fr <= -70 && $fr > -75) do={
  :for i from=1 to=6 do={ :beep length=$beep; :delay $s70; }
  }
 :if ($fr <= -65 && $fr > -70) do={
  :for i from=1 to=8 do={ :beep length=$beep; :delay $s65; }
  }
 :if ($fr <= -60 && $fr > -65) do={
```

```
 :for i from=1 to=11 do={ :beep length=$beep; :delay $s60; }
 }
:if ($fr <= -55 && $fr > -60) do={
 :for i from=1 to=13 do={ :beep length=$beep; :delay $s55; }
 }
:if ($fr <= -50 && $fr > -55) do={
 :for i from=1 to=18 do={ :beep length=$beep; :delay $s50; }
 }
:if ($fr <= -45 && $fr > -50) do={
 :for i from=1 to=25 do={ :beep length=$beep; :delay $s45; }
 }
:if ($fr <= -40 && $fr > -45) do={
 :for i from=1 to=31 do={ :beep length=$beep; :delay $s40; }
 }
:if ($fr <= -20 && $fr > -40) do={
 :for i from=1 to=40 do={ :beep length=$beep; :delay $s20; }
 }

 }
}
```

该脚本在信号强度越强的情况下，发声频率越高

下面是 Station-wds 使用的声控信号强度脚本：

```
:local beep "10ms";
:local s85 "1350ms";
:local s80 "850ms";
:local s75 "650ms";
:local s70 "450ms";
:local s65 "350ms";
:local s60 "250ms";
:local s55 "200ms";
:local s50 "150ms";
:local s45 "100ms";
:local s40 "60ms";
:local s20 "20ms";
:for i from=1 to=100 do={
/interface wireless monitor wlan1 interval=1 do={
 :if ($"signal-strength" <= -85 && $"signal-strength" > -88) do={
  :for i from=1 to=2 do={ :beep length=$beep; :delay $s85; }
  }
 :if ($"signal-strength" <= -80 && $"signal-strength" > -85) do={
  :for i from=1 to=3 do={ :beep length=$beep; :delay $s80; }
  }
 :if ($"signal-strength" <= -75 && $"signal-strength" > -80) do={
  :for i from=1 to=4 do={ :beep length=$beep; :delay $s75; }
  }
```

```
 :if ($"signal-strength" <= -70 && $"signal-strength" > -75) do={
  :for i from=1 to=6 do={ :beep length=$beep; :delay $s70; }
  }
 :if ($"signal-strength" <= -65 && $"signal-strength" > -70) do={
  :for i from=1 to=8 do={ :beep length=$beep; :delay $s65; }
  }
 :if ($"signal-strength" <= -60 && $"signal-strength" > -65) do={
  :for i from=1 to=10 do={ :beep length=$beep; :delay $s60; }
  }
 :if ($"signal-strength" <= -55 && $"signal-strength" > -60) do={
  :for i from=1 to=12 do={ :beep length=$beep; :delay $s55; }
  }
 :if ($"signal-strength" <= -50 && $"signal-strength" > -55) do={
  :for i from=1 to=16 do={ :beep length=$beep; :delay $s50; }
  }
 :if ($"signal-strength" <= -45 && $"signal-strength" > -50) do={
  :for i from=1 to=24 do={ :beep length=$beep; :delay $s45; }
  }
 :if ($"signal-strength" <= -40 && $"signal-strength" > -45) do={
  :for i from=1 to=34 do={ :beep length=$beep; :delay $s40; }
  }
 :if ($"signal-strength" <= -20 && $"signal-strength" > -40) do={
  :for i from=1 to=48 do={ :beep length=$beep; :delay $s20; }
  }
 }
}
```

声音控制脚本

警报声

```
:for i from=1 to=3 step=1 do={
   :beep frequency=550 length=494ms;
   :delay 494ms;
   :beep frequency=400 length=494ms;
   :delay 494ms;
 }
```

电话铃声

```
:for i from=1 to=10 step=1 do={
   :beep frequency=1195 length=22ms;
   :delay 22ms;
   :beep frequency=2571 length=22ms;
   :delay 22ms;
 }
```

Coo 发声

```
:for i from=0 to=150 step=10 do={
  :beep frequency=(1295 - i) length=22ms;
  :delay 22ms;
  :beep frequency=(1095 + i) length=22ms;
  :delay 22ms;
 }
```

操作成功发声

```
:beep frequency=523 length=200ms;
 :delay 1000ms;

 :beep frequency=523 length=200ms;
 :delay 1000ms;

 :beep frequency=523 length=200ms;
 :delay 1000ms;

 :beep frequency=659 length=700ms;
 :delay 700ms;

 :beep frequency=784 length=500ms;
 :delay 500ms;

 :beep frequency=523 length=200ms;
 :delay 1000ms;

 :beep frequency=523 length=200ms;
 :delay 1000ms;

 :beep frequency=523 length=200ms;
 :delay 1000ms;

 :beep frequency=659 length=700ms;
 :delay 700ms;

 :beep frequency=784 length=500ms;
 :delay 800ms;

 :beep frequency=784 length=400ms;
 :delay 400ms;

 :beep frequency=884 length=200ms;
 :delay 200ms;

 :beep frequency=784 length=200ms;
```

```
:delay 200ms;

:beep frequency=687 length=200ms;
:delay 200ms;

:beep frequency=659 length=200ms;
:delay 200ms;

:beep frequency=579 length=200ms;
:delay 200ms;

:beep frequency=519 length=400ms;
:delay 400ms;
```

第三章 WLAN 无线基础知识

3.1 802.11 传输协议

学习 WLAN 无线知识前，我们需要先来了解下关于 802.11 无线协议的基本应用和技术，这样才能真正理解和操作包括 RouterOS 在内的所有 802.11 协议的无线设备，同时也能理解为什么构建 WiFi 或 WLAN 网络前需要做一些勘查和分析，为什么无线网络比有线网络方便，但会遇到比有线网络更复杂的问题。

不管是 RouterOS 还是其他无线设备配置基本是相差不大，关键是需要对无线协议、硬件设施、网络环境以及周边地理与环境情况进行分析，确定你选择的是正确的解决方案，虽然 RouterOS 的 802.11 无线技术不是最先进的，但 RouterOS 在整体的网络解决方案中有较大的优势，因为他的各项网络功能是其他无线网络设备不能比拟的。

802.11 传输协议我们知道分为 abgn，他们的频率使用又分为 2.4GHz 和 5GHz，802.11bg 使用的是 2.4G 频段，802.11a 使用的是 5G 频段，2.4GHz 频率更低，传输距离、穿透和绕射能力更强，而 5GHz 频率较高，传输距离、穿透和绕射能力都不如 2.4GHz，但其数据承载能力较 2.4G 更强，且该频段较为干净，干扰小。 所以，通常 2.4GHz 会被用于 WiFi 的覆盖上网，5GHz 会用于 WLAN 的数据传输。

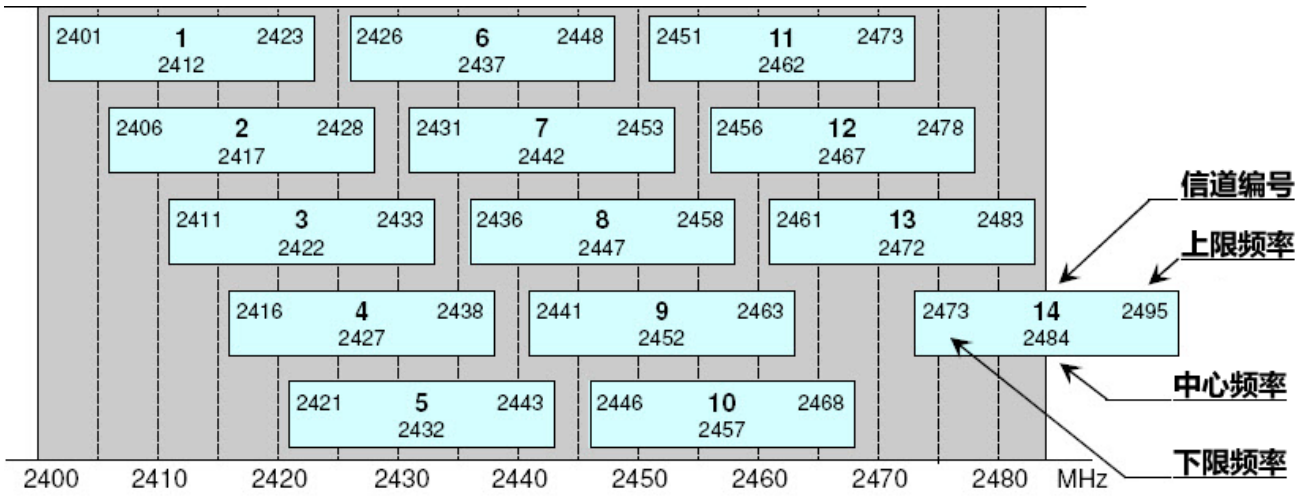
我们在市面上可以看到 11n 的产品说明，例如 802.11bgn 代表的是基于 2.4GHz 的 11n 协议，802.11an 代表的是支持 5GHz 的 11n 协议，如果是 802.11abgn，即同时支持 2.4GHz 和 5GHz 频率。

下面是各个协议的具体参数：

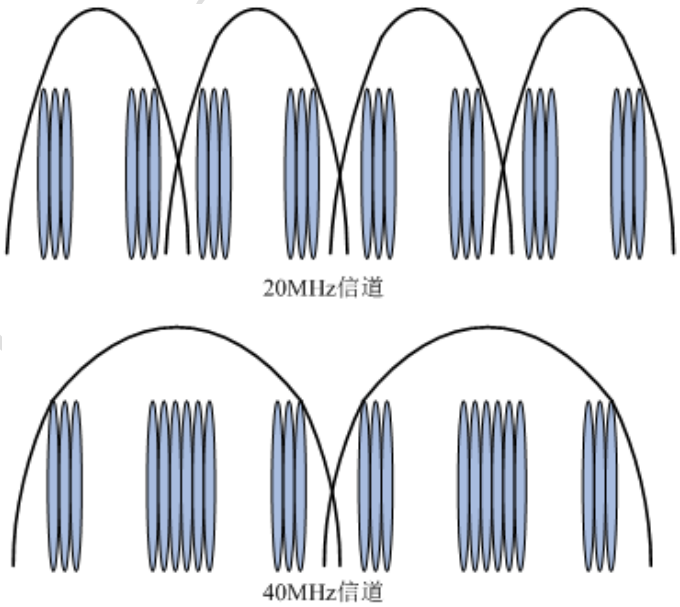
标准	工作频段	频道间隔	最大速率
802.11b	2.4GHz： 2312-2599MHz	 5MHz	 11Mbps
802.11g **G-Turbo**	2.4GHz 2312-2599MHz	 5MHz **44MHz**	 54Mbps **108Mbps**
802.11a **A-Turbo**	5GHz： 4920-6100MHz	 5MHz 10MHz 20MHz **40MHz**	 13.5Mbps 27Mbps 54Mbps **108Mbps**
802.11n	2.4/5Ghz： 2312-2599Mhz 4920-6100Mhz	 5MHz 10MHz 20MHz 40MHz(2×20MHz) 60MHz(3×20MHz)	 37.5Mbps 75Mbps 150Mbps 300Mbps 450Mbps
802.11ac	2.4/5Ghz： 2312-2599Mhz 4920-6100Mhz	20MHz 40MHz 80MHz 160MHz	1.52~2.26Gbps

802.11bg 2.4G 频率占用

2.4GHz 频段中，同一区域覆盖范围内最多容纳 3 个互不重迭的通道（如下图），每通道最大可占用 22 MHz 的频带；11b 采用 DSSS 扩频和 CCK 的调制方式最高提供 11Mbps 的速率，11g 采用 OFDM 的扩频方式，可提供 54Mbps 的速率，如 2412MHz 频率,在使用时候会干扰到上至 2423MHz 和下至 2401MHz

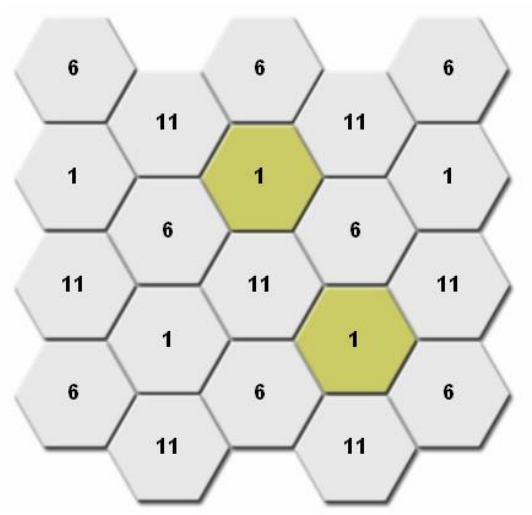


和 11g 类似，在 2.4GHz 频段可以有三个不重迭的通道，但是对于 11n 来说，使用 20MHz 就意味着只能达到 150Mbps 的速率。11n 与 11g 的 Turbo 模式类似，一旦使用 40MHz 带宽，在 2.4GHz 频段只有一个通道可用，基本上无法规模部署，解决方法是使用 5GHz 频段，带宽资源丰富（11n 选择 5G 主要应用于数据传输，而非对终端使用者的覆盖）。



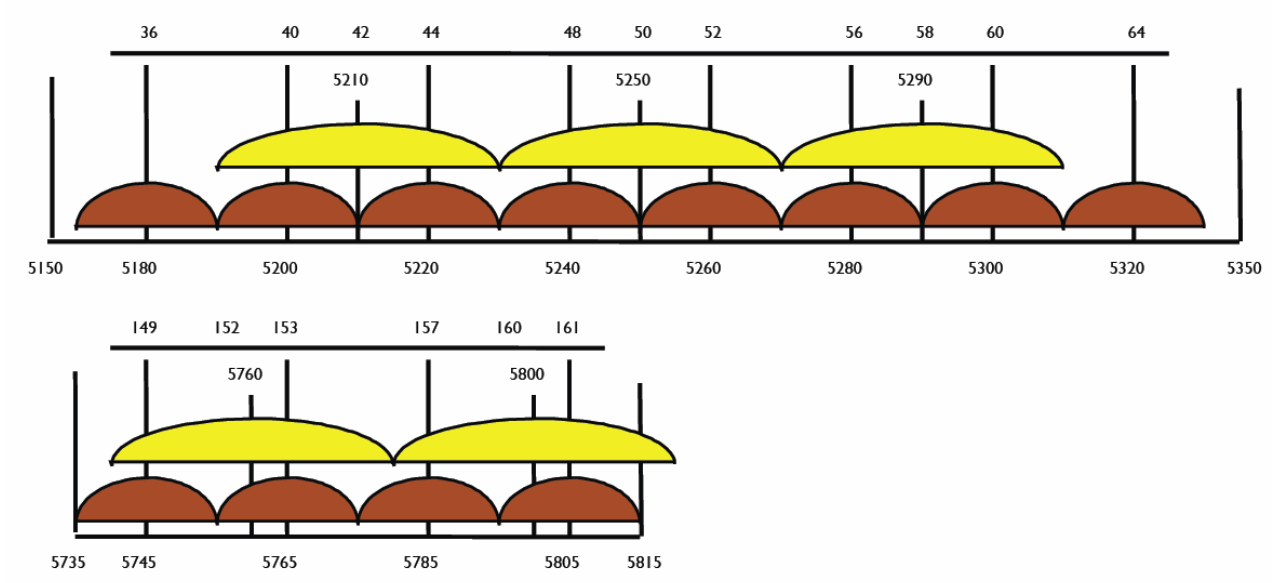
WLAN 2.4G 频率覆盖位置规划

多个邻居设备间频率不能相同，频率应使用不同频道，如图多个基站情况相邻设备间频道在 1、6、11，可以避免频道交叉而带来的干扰问题：



5G 频道使用情况

802.11a 的 5G 分为 5.2G 和 5.8G 两个段频道，5G 有 8+4 个非重迭信道，每个信道使用 16.6MHz 的带宽，比 11g 的带宽更小，这就是为什么我们用 5G 做为骨干传输的原因。



上图，我们可以看到从 5180-5320 每隔 20MHz 一个频道，共 8 个频道，从 5745-5805 的 4 个频道，而当我们采用 5G-turbo 模式（即多信道传输），只能获得 5 个独立的频道，5210、5250、5290、5760 和 5800。

3.2 天线（antenna）

天线是 WLAN 网络的重要组成部分，通过天线将 WLAN 设备发射出的信号放大，并覆盖或传向指定的方向。那天线在是如何工作的？下面我们简单介绍下天线的几个特性

天线方向性

发射天线的基本功能之一是把从馈线取得的能量向周围空间辐射出去，基本功能之二是把大部分能量朝所需的方向辐射。垂直放置的半波对称振子具有平放的 “面包圈” 形的立体方向图（图 1）。图 2

与图 3 给出了它的两个主平面方向图，平面方向图描述天线在某指定平面上的方向性。从图 2 侧视可以看出，在振子的轴线方向上辐射为零，最大辐射方向在水平面上；而从图 3 俯视可以看出，在水平面上各个方向上的辐射一样大。

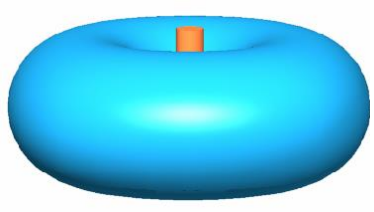


图 1 面包圈 立体图

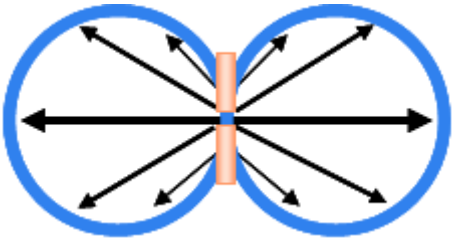


图 2 侧视图

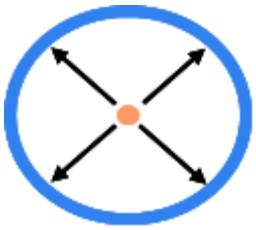
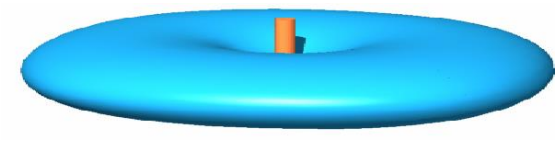
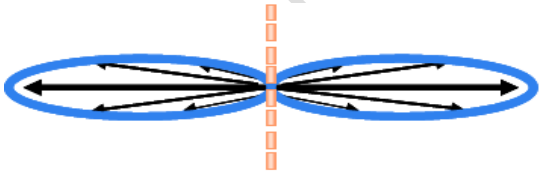


图 3 俯视

天线方向性增强

若干个对称振子组阵，能够控制辐射，产生“扁平的面包圈”，把信号进一步集中到在水平面方向上。下图是 4 个半波对称振子沿垂在线下排列成一个垂直四元阵时的立体方向图和垂直面方向图。

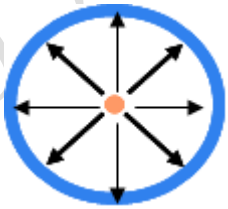




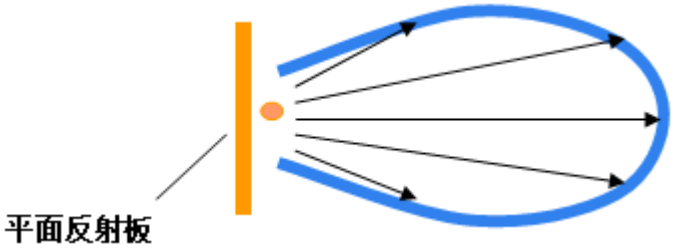
扁平面包圈

利用反射板把辐射能控制到单侧方向

平面反射板放在数组的一边构成扇形区覆盖天线。下面的水平面方向图说明了反射面的作用-----反射面把功率反射到单侧方向，提高了增益。



全向阵（不带平面反射板）



扇形区覆盖 （带平面反射板）

抛物反射面的使用，更能使天线的辐射，像光学中的探照灯那样，把能量集中到一个小立体角内，从而获得很高的增益。不言而喻，抛物面天线的构成包括两个基本要素：抛物反射面 和 放置在抛物面焦点上的辐射源。

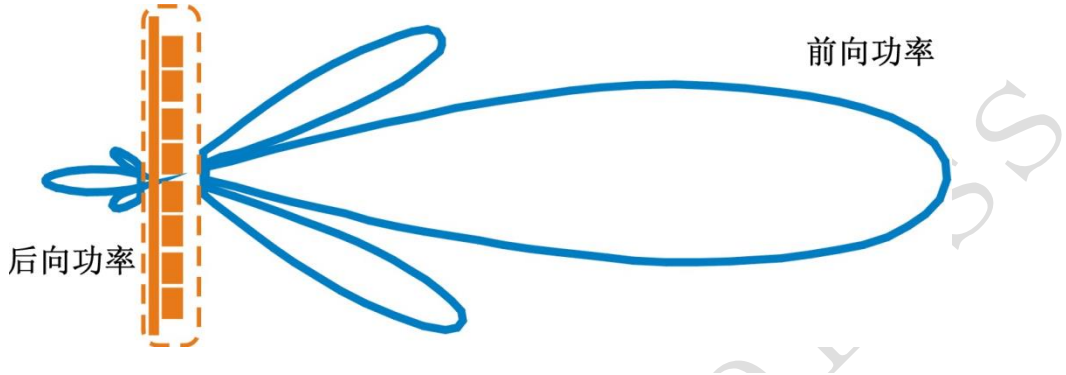
增益

增益是指：在输入功率相等的条件下，实际天线与理想的辐射单元在空间同一点处所产生的信号的功率密度之比。它定量地描述一个天线把输入功率集中辐射的程度。增益显然与天线方向图有密切的关系，方向图主瓣越窄，副瓣越小，增益越高。 可以这样来理解增益的物理含义------为在一定的距离上

的某点处产生一定大小的信号，如果用理想的无方向性点源作为发射天线，需要 100W 的输入功率，而用增益为 G = 13 dB = 20 的某定向天线作为发射天线时，输入功率只需 100 / 20 = 5W；换言之，某天线的增益，就其最大辐射方向上的辐射效果来说，与无方向性的理想点源相比，把输入功率放大的倍数。

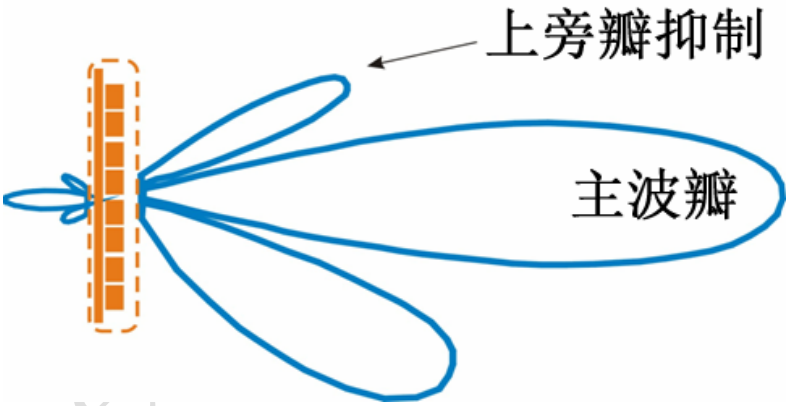
前后比

下图中，前后瓣最大值之比称为前后比，记为 F / B 。前后比越大，天线的后向辐射（或接收）越小。其典型值为 （18 ~ 30）dB，特殊情况下则要求达（35 ~ 40）dB .

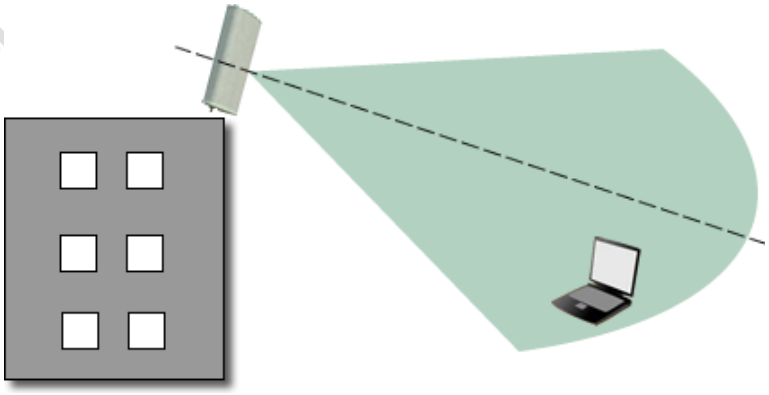


上旁瓣抑制

对于基站天线，人们常常要求它的垂直面（即俯仰面）方向图中，主瓣上方第一旁瓣尽可能弱一些。这就是所谓的上旁瓣抑制 。基站的服务对象是地面上的移动电话用户，指向天空的辐射是毫无意义的。

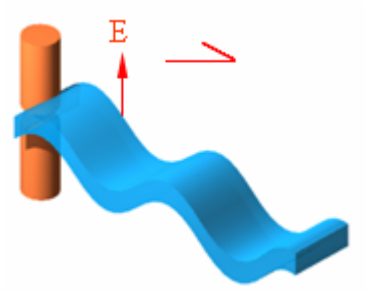


注：天线的下倾——为使主波瓣指向地面，安置时需要将天线适度下倾，这样的情况用在当 WLAN 设备假设在高处，并对区域覆盖时。

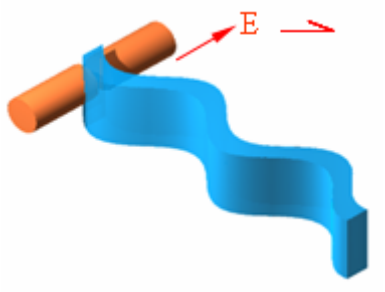


天线的极化

天线向周围空间辐射电磁波。电磁波由电场和磁场构成。人们规定：电场的方向就是天线极化方向。一般使用的天线为单极化的。下图示出了两种基本的单极化的情况：垂直极化---是最常用的一种极化方式；水平极化---在一些场合也会被用到。



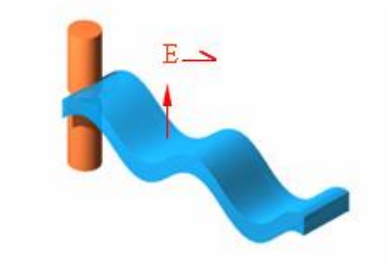
垂直极化



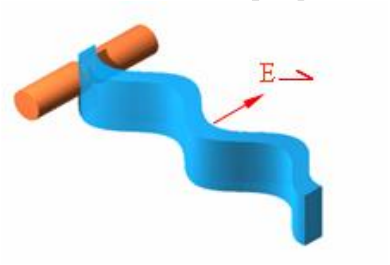
水平极化

双极化天线

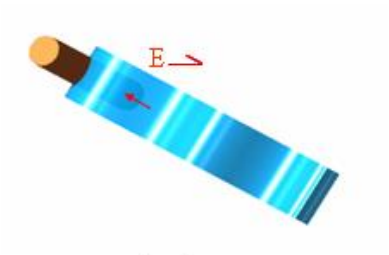
下图示出了另两种单极化的情况：±45° 极化，这样将极化方式增加到四种，也就是说我们在安装天线时多了两种方式，但±45° 极化通常使用的较少。见下图：



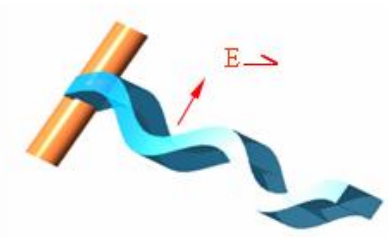
垂直极化



水平极化

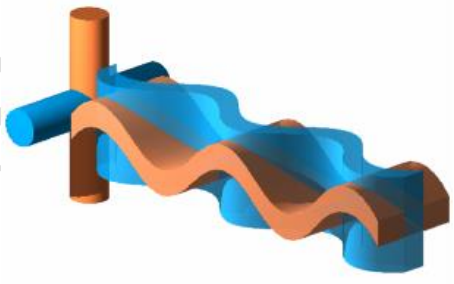


+45° 极化

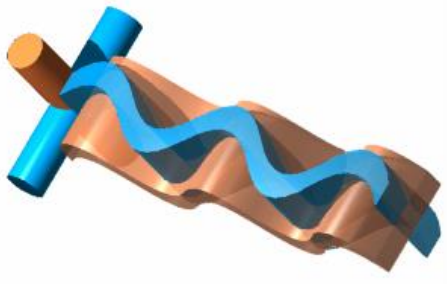


-45° 极化

把垂直极化和水平极化两种极化的天线组合在一起，或者把 +45° 极化和 -45° 极化两种极化的天线组合在一起，就构成了一种新的天线---双极化天线。



±90° 双极化



±45° 双极化

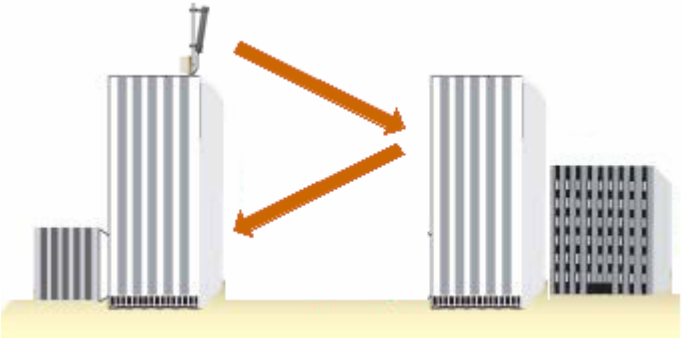
双极化天线在 RouterOS 上主要应用在 bonding、Nstreme Dual 和 802.11n 的高带宽传输应用上，例如 RB-SXT 集成的就是双极化 5G 天线。

建议：密集城区和普通城区覆盖优先选择 ±45° 双极化天线；一般郊区，农村和郊区可选择±90° 双极化天线。

电波的多径传播

在超短波、微波波段，电波在传播过程中还会遇到障碍物(例如楼房、高大建筑物或山丘等) 对电波

产生反射。因此，到达接收天线的含有多种反射波（广意地说，地面反射波也应包括在内），这种现象叫为多径传播。



由于多径传输，使得信号场强的空间分布变得相当复杂，波动很大，有的地方信号场强增强，有的地方信号场强减弱；也由于多径传输的影响，还会使电波的极化方向发生变化。另外，不同的障碍物对电波的反射能力也不同。例如：钢筋水泥建筑物对超短波、微波的反射能力比砖墙强。我们应尽量克服多径传输效应的负面影响，这也正是在通信质量要求较高的通信网中，人们常常采用空间分集技术或极化分集技术的缘由。

电波的绕射传播

在传播途径中遇到大障碍物时，电波会绕过障碍物向前传播，这种现象叫做电波的绕射。超短波、微波的频率较高，波长短，绕射能力弱，在高大建筑物后面信号强度小，形成所谓的“阴影区”。信号质量受到影响的程度，不仅和建筑物的高度有关，和接收天线与建筑物之间的距离有关，还和频率有关。

例如有一个建筑物，其高度为 10 米，在建筑物后面距离 200 米处，接收的信号质量几乎不受影响，但在 100 米处，接收信号场强比无建筑物时明显减弱。注意，诚如上面所说过的那样，减弱程度还与信号频率有关，对于 216 ～ 223 兆赫的电视射频信号，接收信号场强比无建筑物时低 16 dB，对于 900 兆赫的手机射频信号，接收信号场强比无建筑物时低 22dB .

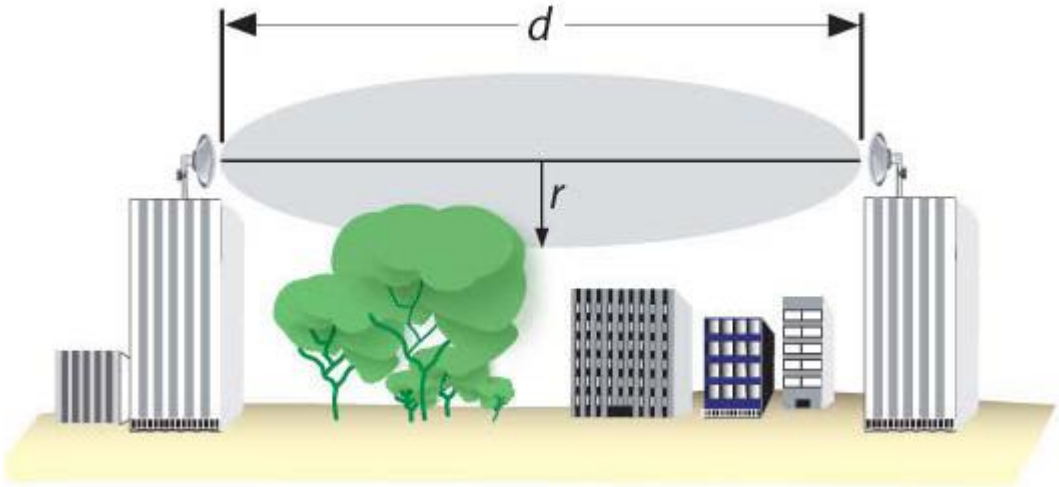
如果建筑物高度增加到 50 米，则在距建筑物 1000 米以内，接收信号的场强都将受到影响而减弱。也就是说，频率越高、建筑物越高、接收天线与建筑物越近，信号强度与通信质量受影响程度越大；相反，频率越低，建筑物越矮、接收天线与建筑物越远，影响越小。

因此，选择基站场地以及架设天线时，一定要考虑到绕射传播可能产生的各种不利影响，注意到对绕射传播起影响的各种因素。

菲涅尔区(Fresnel Zone)

Fresnel 区是一个视线区域的无线电波分布范围，这个区域必须无障碍，否则信号强度会被削减。例如 在一个 16 公里使用 5.8G 连接的无线，60%的 fresnel 区是一个 8.7 米的圆球区，在 2.4GHz 同样的距离是 13.6 米。点对点和点对多点的环境尽量选择周围的制高点，在无线覆盖如小区、城市等，将来寻找障碍较少的网站，这是为增加视距传播，增加覆盖距离和范围，以便减少网站数量。

在收发天线之间连一条线，以这条线为轴心，以 R 为半径的一个类似于管道的区域内，没有障碍物的阻挡。如图所示，这个管道称为菲涅尔区(Fresnel Zone)，菲涅尔区是一个椭球体，收发天线位于椭球的两个焦点上，



例如：在一个 16 公里使用 5.8G 连接的无线，60%的 fresnel 区是一个 8.7 米的圆球区，在 2.4GHz 同样的距离是 13.6 米。

板状天线高增益的形成

为提高板状天线的增益，还可以进一步采用八个半波振子排阵 前面已指出，四个半波振子排成一个垂直放置的直线阵的增益约为 8 dB；一侧加有一个反射板的四元式直线阵，即常规板状天线，其增益约为 14 ~ 17 dB 。 一侧加有一个反射板的八元式直线阵，即加长型板状天线，其增益约为 16 ~ 19 dB . 不言而喻，加长型板状天线的长度，为常规板状天线的一倍，达 2.4 m 左右。

高增益栅状抛物面天线

从性能价格比出发，人们常常选用栅状抛物面天线作为主天线。由于抛物面具有良好的聚焦作用，所以抛物面天线集射能力强，直径为 1.5 m 的栅状抛物面天线，在 900 兆频段，其增益即可达 G = 20 dB . 它特别适用于点对点的通信，例如它常常被选用为直放站的施主天线。

抛物面采用栅状结构，一是为了减轻天线的重量，二是为了减少风的阻力。

抛物面天线一般都能给出 不低于 30 dB 的前后比 ，这也正是直放站系统防自激而对接收天线所提出的必须满足的技术指标，当然使用碟形抛物面天线效果会比栅状抛物面天线好，但成本会高出许多。

3.3 天线类型

看看 WLAN 常用的几种天线类型，我们可以分为全向、扇区和定向几种天线：

2.4/5G 全向天	2.4/5G 平板扇	2.4G 抛物面栅	5G 抛物面栅	5G 抛物面碟

线	区天线	格天线	格天线	型天线
				
用于全向覆盖	可用于扇形区域的覆盖或者点对多点传输	用于定向的 2.4G 点对点传输	用于定向的 5G 点对点传输	用于远距离定向的 5G 点对点传输
2.4/5G 双极化天线	**2.4/5G 集成外壳天线**	**2.4G 室内覆盖吸顶天线**		
				
11n 的高带宽传输	将设备集成在天线内	用于室内 WiFi 覆盖		

点对点安装

在安装点目测周围环境，判断远程目标的大概方向，天线对准目标，然后设备启动后，如果设备启动后未发现信号，可通过逐步转动天线寻找信号，当发现信号后，双方根据信号强度做微调，直到达到最好的信号强度为止。

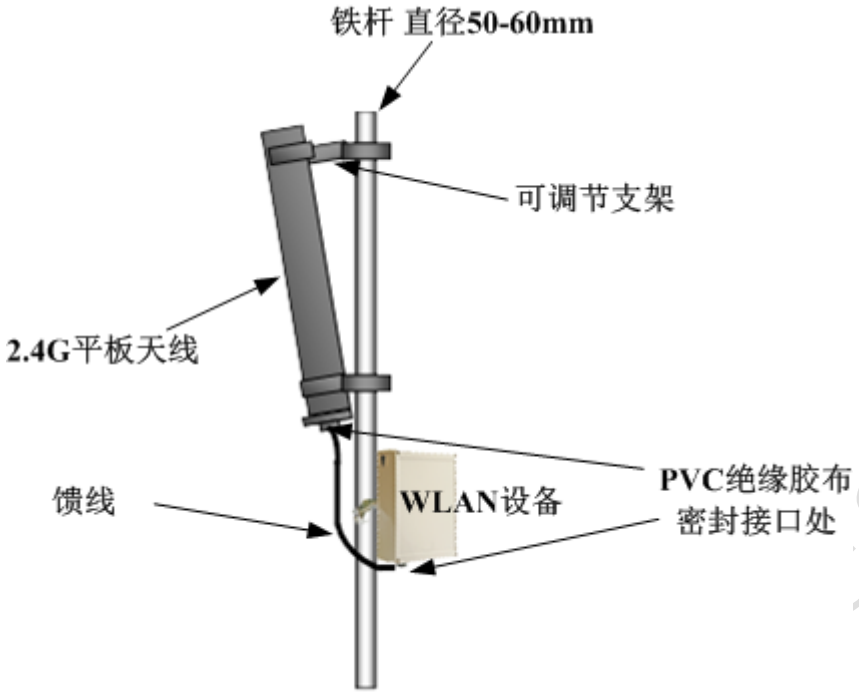
覆盖安装

- 天线高度：天线过高会降低天线附近的覆盖电平（特别是垂直向下的）俗称“塔下黑”，特别是全向天线最为明显
- 天线高度过高容易造成越区覆盖问题，影响网络质量。
- 天线方向角：天线主播瓣方向指向高用户人群区域，可以加强该区域的信号强度。
- 相邻天线的交叉覆盖不宜超过 10%
- 需要考虑天线的俯仰角度，特别在高处调整俯仰调度，有利于附近平地的信号接收。

天线安装

室外天线安装需要相应的配套设施，确保天线在室外不受各种环境影响，如下图的 2.4G 平板天线安装

- 需要直接 50-60mm 的铁杆，通过 U 形卡固定天线
- 使用相应高质量的馈线连接天线和 WLAN 设备
- WLAN 设备固定在天线的下方，尽量将设备安装在较低位置，通过馈线连接
- 天线接头和 WLAN 设备接头通过 PVC 绝缘材料或者胶布包裹密封



室外设备实际安装示意图：

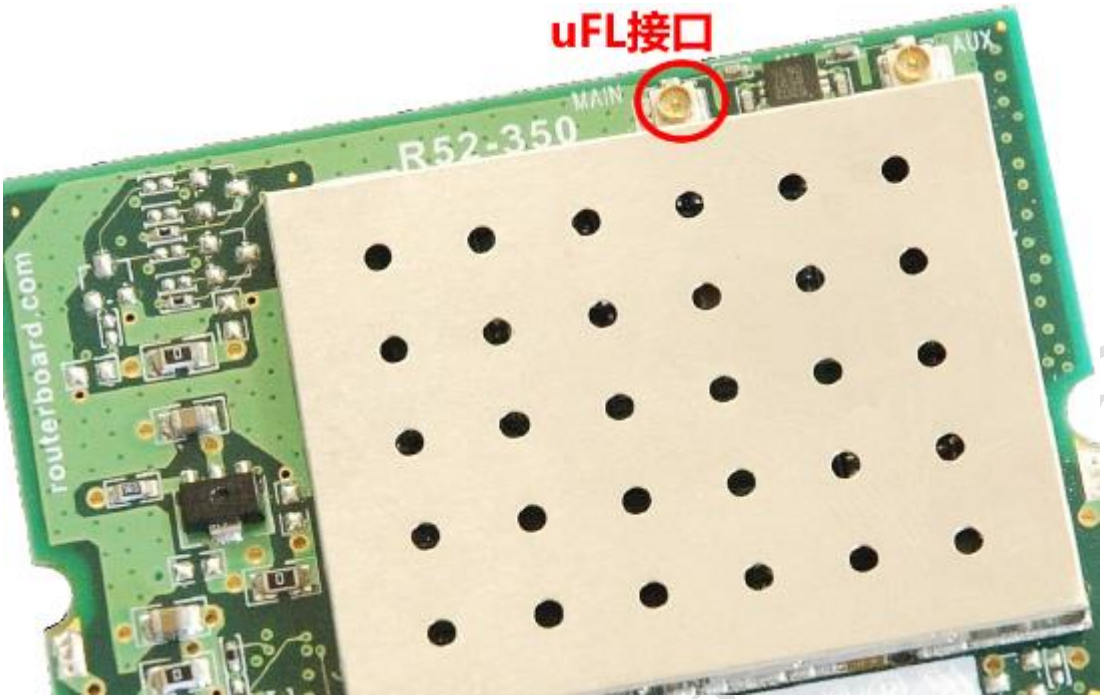
3.4 接头与线材类型

这里介绍下设备常用的接头与线材类型，有助于你面对各种设备或网卡与天线连接时，能正确的选择线材和接口类型。

1、无线网卡接口，以及可能会涉及到的线材接头类型：

无线网卡接口类型	线材类型 1	线材类型 2
uFL 接头	uFL to SMA 跳线	uFL to N 跳线
MMCX 接头	MMCX to SMA 跳线	MMCX to N 跳线

下图是无线网卡的 uFL 接头：

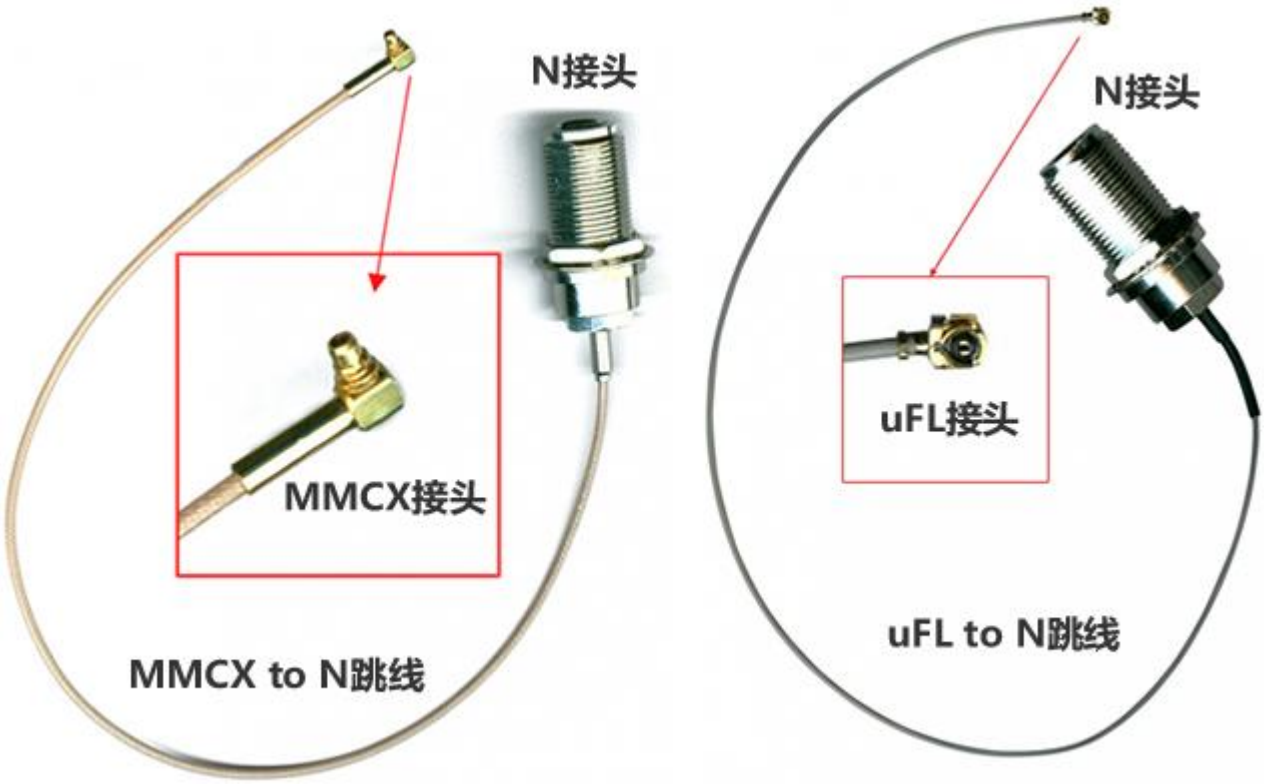


下图是无线网卡的 MMCX 接头

2、认识无线网卡的接头，现在认识下从无线网卡连接到天线的接口和线材

天线接口或线材连接头	线材类型 1
SMA 型（公头，母头）	SMA to N 馈线
N 型（公头，母头）	N to N 馈线

下面是 MMCX to N 跳线和 uFL to N 跳线，主要用于无线网卡与 N to N 馈线连接



下面是 SMA 接头（公头与母头）和 SMA 的馈线：

下面是 N to N 馈线，主要用于设备和天线连接：

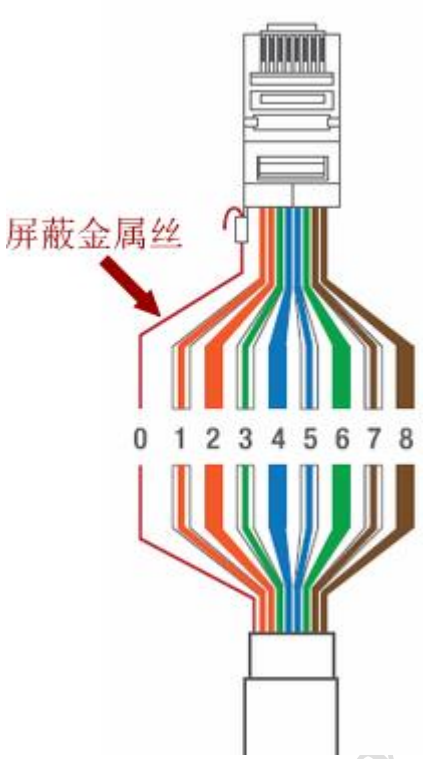
3.5 RouterBOARD 设备接地

屏蔽双绞线

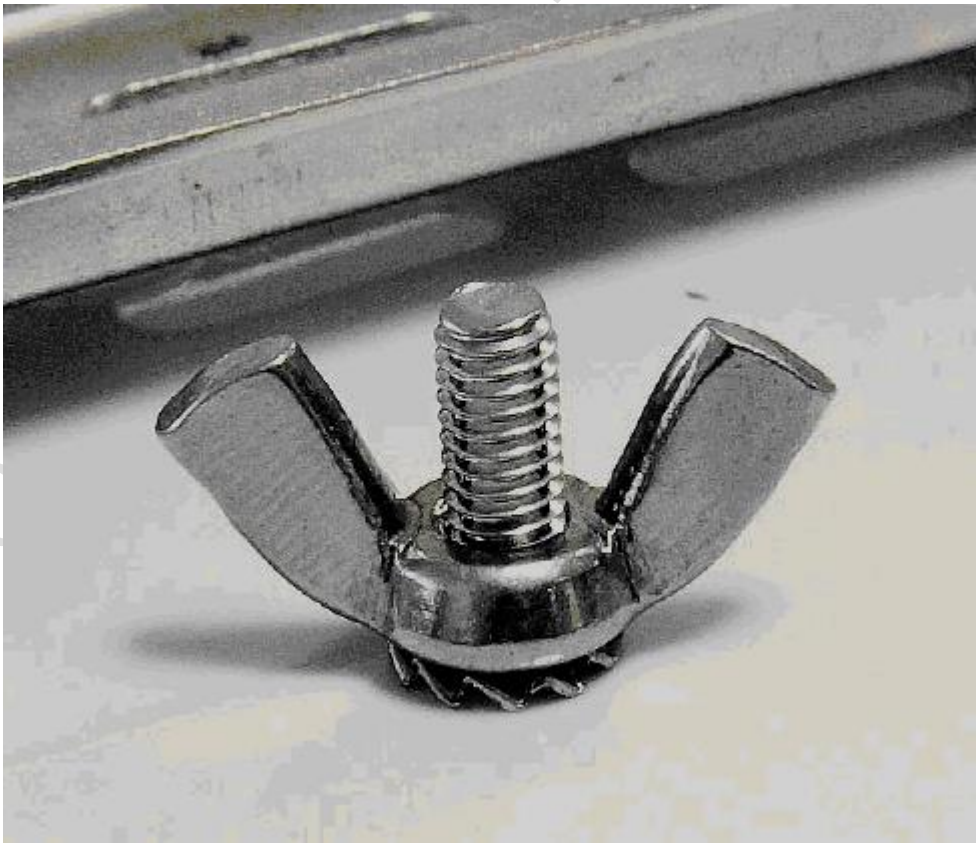
室外无线安装在建筑物顶端或铁杆上，天线和无线设备都应当适当的接地，天线避雷器必须安装在外接的馈在线（靠近天线或者天线接头位置）防止设备损坏。注意天线避雷器如果没有接地将没有任何作用。

使用 7mm 直径（1 AWG）的金属线包裹在耐腐作材料接地。需要确定你接地是有效的，即需要连接到建筑物或者附近的接地设施，例如：楼房的避雷针（楼房主水管也有一定的接地作用），如果是小设备可以选用较细的金属线，RB 设备的屏蔽双绞线接地有以下要求

1、室外安装要求使用室外的屏蔽双绞线，也要选择屏蔽的 RJ45 水晶头或者带有接地线接口的 RJ45 水晶头



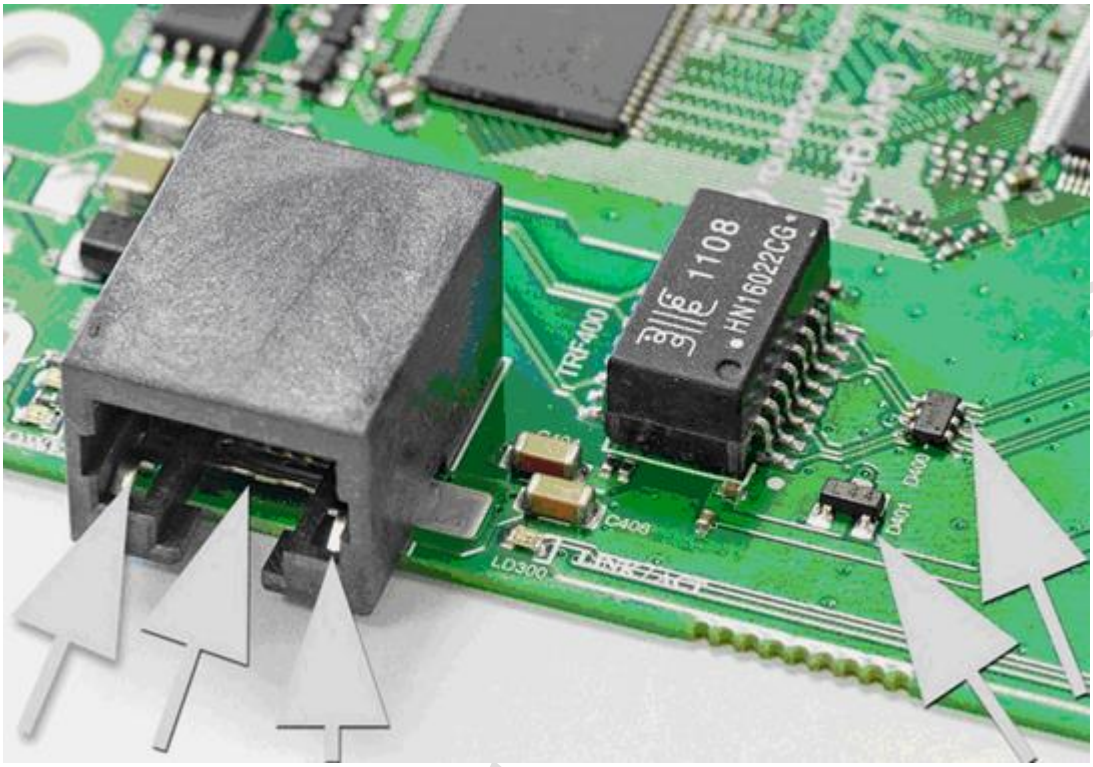
2、接地线还要连接到 RouterBOARD 上，通过 RouterBOARD 的螺丝孔位与机壳的孔位连接，一般金属机壳螺丝孔位是直接可以接地的,塑料室外机壳金属孔位都会与外部的接地金属片相连。



接地线应连接到室外壳的接地金属螺丝上

3、以太网避雷器不建议使用，需要考虑是否影响 POE 连接，以太网避雷器可能影响 POE 使用，造成 POE 电压衰减。如果使用以太网避雷器请放置在室外，并且和 RouterBOARD 接地线一起接地主要考虑。

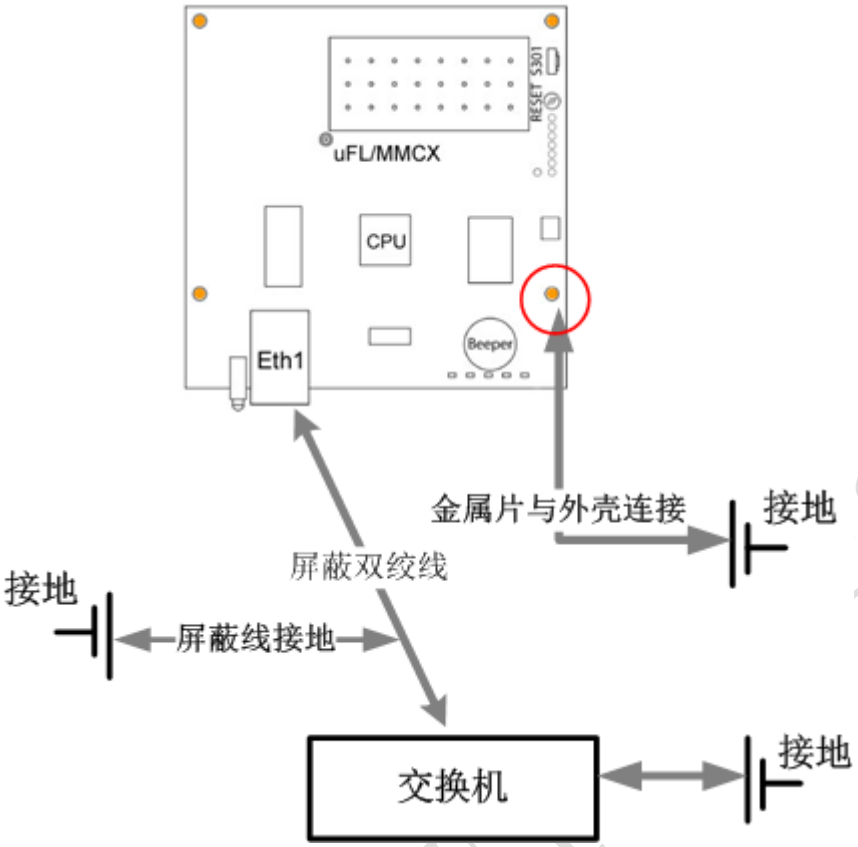
RouterBOARD 的 ESD（防静电）保护



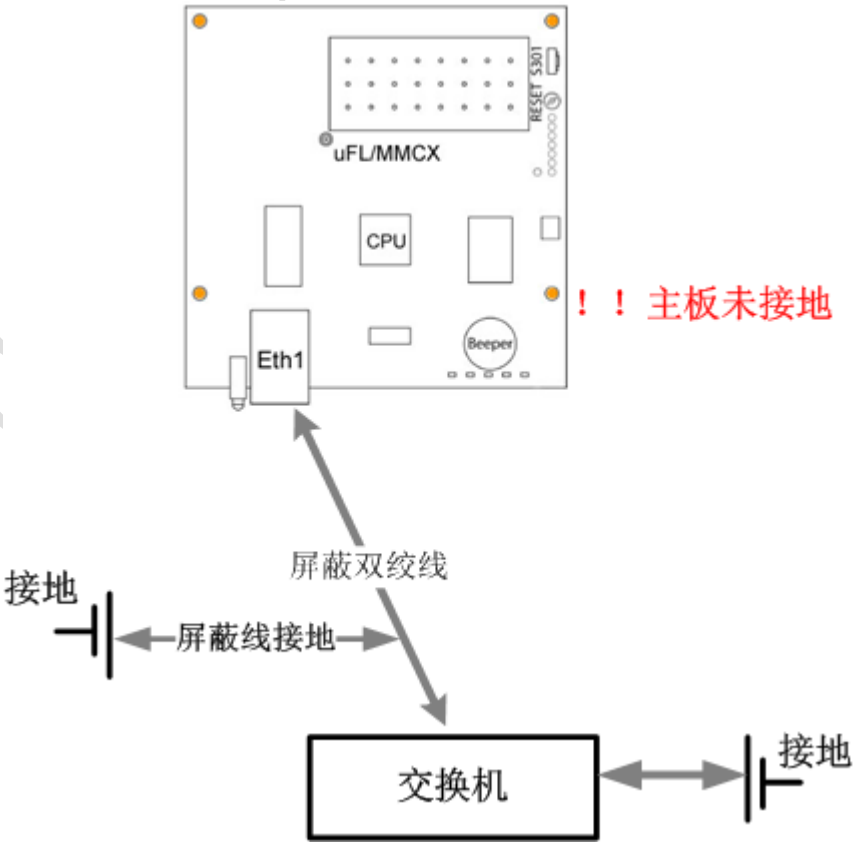
1. 三个箭头标记了以太网口的接地金属片，RouterBOARD 以太网口左右两边的 2 个金属片，用于屏蔽双绞线接地
2. 中间箭头指向的金属片是主板的接地片，主板通过螺丝安装孔金属与外壳的接地相连接（如果你外壳安装孔位没有接地设计，你可以用一条金属线连接相应的接地设施）
3. 在图的右侧两个箭头为 ESD 防静电保护芯片，如果在没有使用屏蔽双绞线时，ESD 芯片可以保护 CPU 和其他部分主板部件

如果仅使用屏蔽双绞线保护有效性很低的，这样主板自身没有接地。你需要两种方式都采用，才能有效保护设备，下面有两种方式，推荐使用第一种方式：

方式一（屏蔽双绞线 + 设备接地）：



方式二：仅有屏蔽双绞线：



注意！即使你没有将无线设备接地，仅使用了屏蔽双绞线，但你应将双绞线连接对端设备交换机、PC 或 RouterBOARD 等室内设备接地。

这里说明下接地能有效降低室外雷击或感应电损坏的几率，但也不是 100%保证不被损坏，但有效地保护是必须的。

3.5 WLAN 网卡和馈线损坏检测

控制 WLAN 信号接收发送、处理和译码是由 WLAN 芯片完成，如 AR5414，AR9220 等，功率放大器（PA）是将 WLAN 信号放大组件，是将 WLAN 信号功率放大，通常我们说的无线网络功率 100mW、300mW 和 500mW 就是由功率放大器完成，功率放大器就像扩音喇叭的作用。

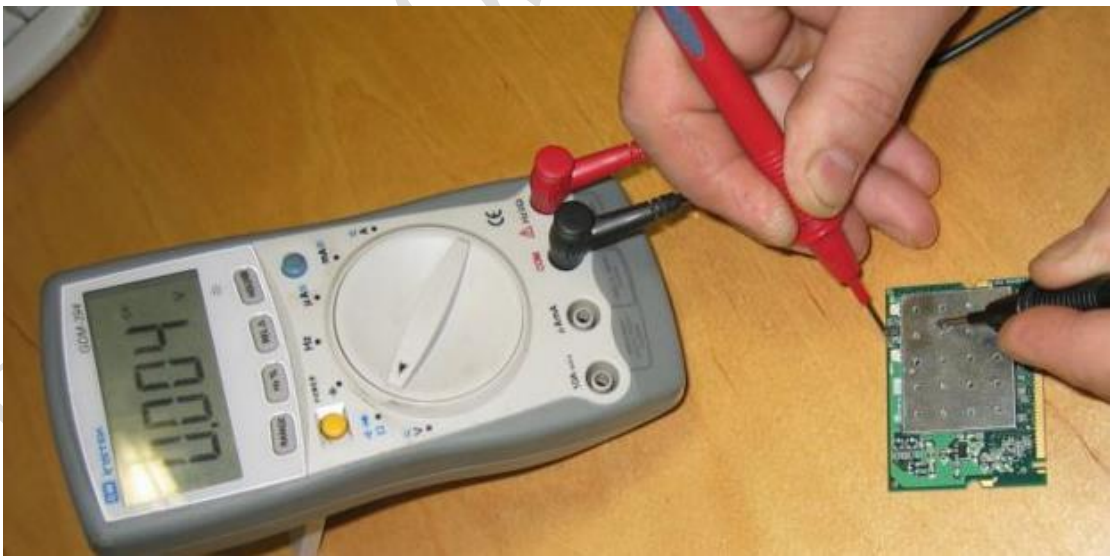
如果功率放大部分损害，直接导致 WLAN 网卡的信号发射和接收变弱，无法连接到其他 WLAN 设备。通过当把一张无线网卡插入 RouterBOARD，如果立马发现无线网卡手感变得发烫时，可能 PA 已经损坏。这样的情况可能是使用者造成，也可能是出厂问题。

R52, R52Hn 和 R52H ESD 损坏测试

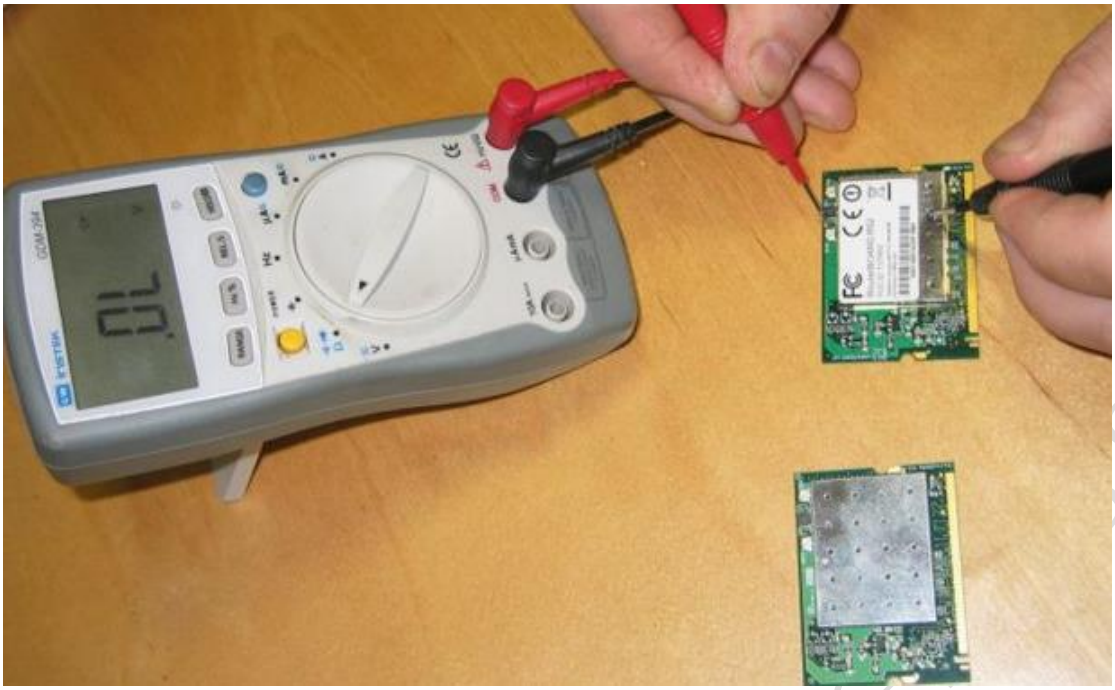
在无线网卡有一个静电保护器，用于静电保护，通常称为 ESD，ESD（Electro-Static discharge）的意思是"静电释放"。ESD 是 20 世纪中期以来形成的以研究静电的产生、危害及静电防护等的学科。因此，国际上习惯将用于静电防护的器材统称为 ESD，中文名称为静电阻抗器。

通常不正确的接地会导致无线网卡在雷暴或其他静电环境损坏。如果 R52 或 R52H 在雷暴天气后网卡不能正常工作，可以使用万用表检测。下面的实例可以检测是否损坏：

首先将万用表调整到电阻档，测试电阻，通过测试静电保护器，如果发现有电阻值，说明静电保护器已经被击穿，即损坏，这样情况并不能保证能修复网卡，如下图：。



正常的网卡测试如下：这张图的万用表显示的 OL（电阻无穷大），万用表还有显示为 1，也是同样的表示未短路



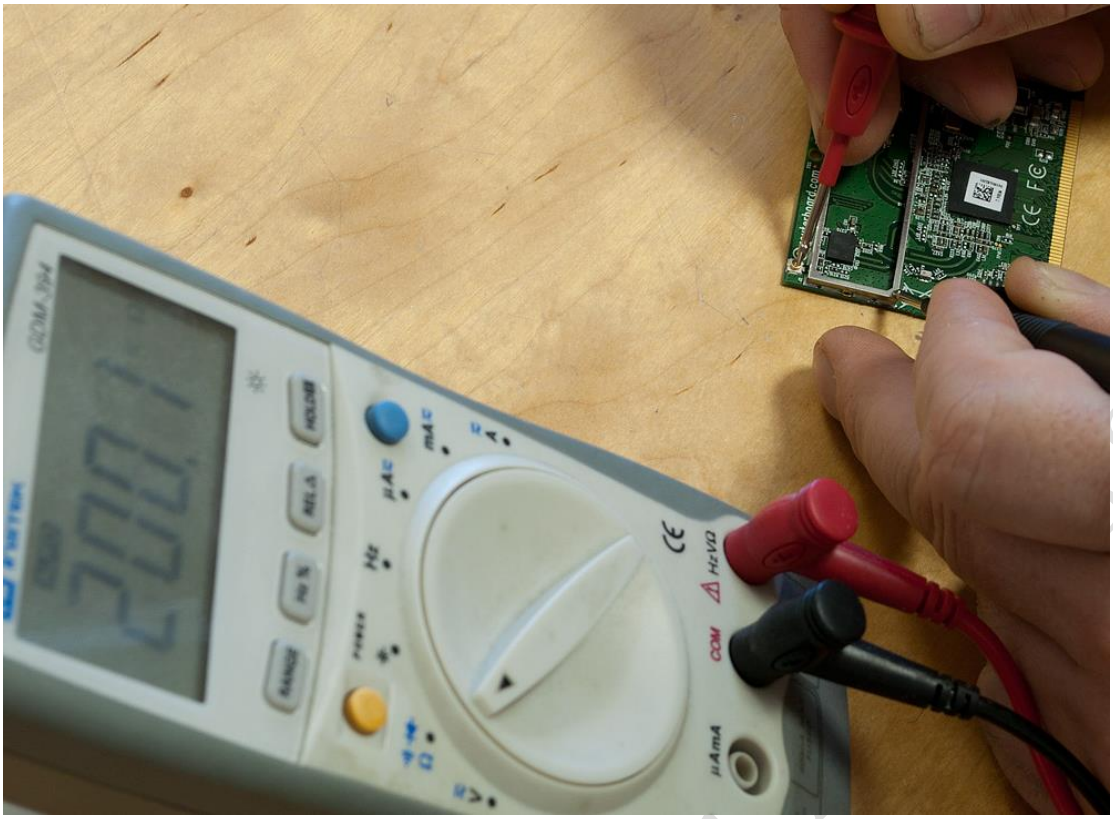
R52Hn 网卡测试 Chain 0 的位置：

R52Hn 网卡测试 chain 1 的位置：

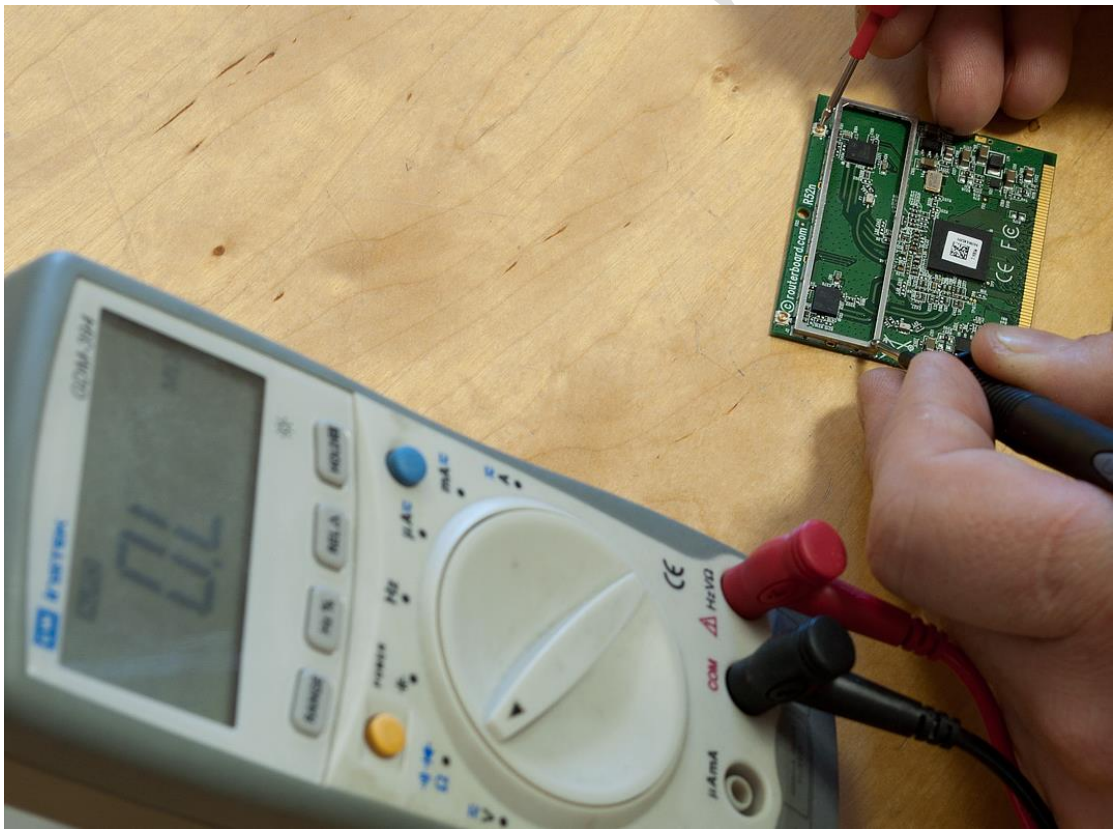
R52n 天线电路损坏测试

下面的图片展示了如何测试天线电路是否损坏，如果电阻低于无穷大说明网卡被击穿，这样情况是几乎无法修复。按照官方的意思就是没有必要返厂维修了。

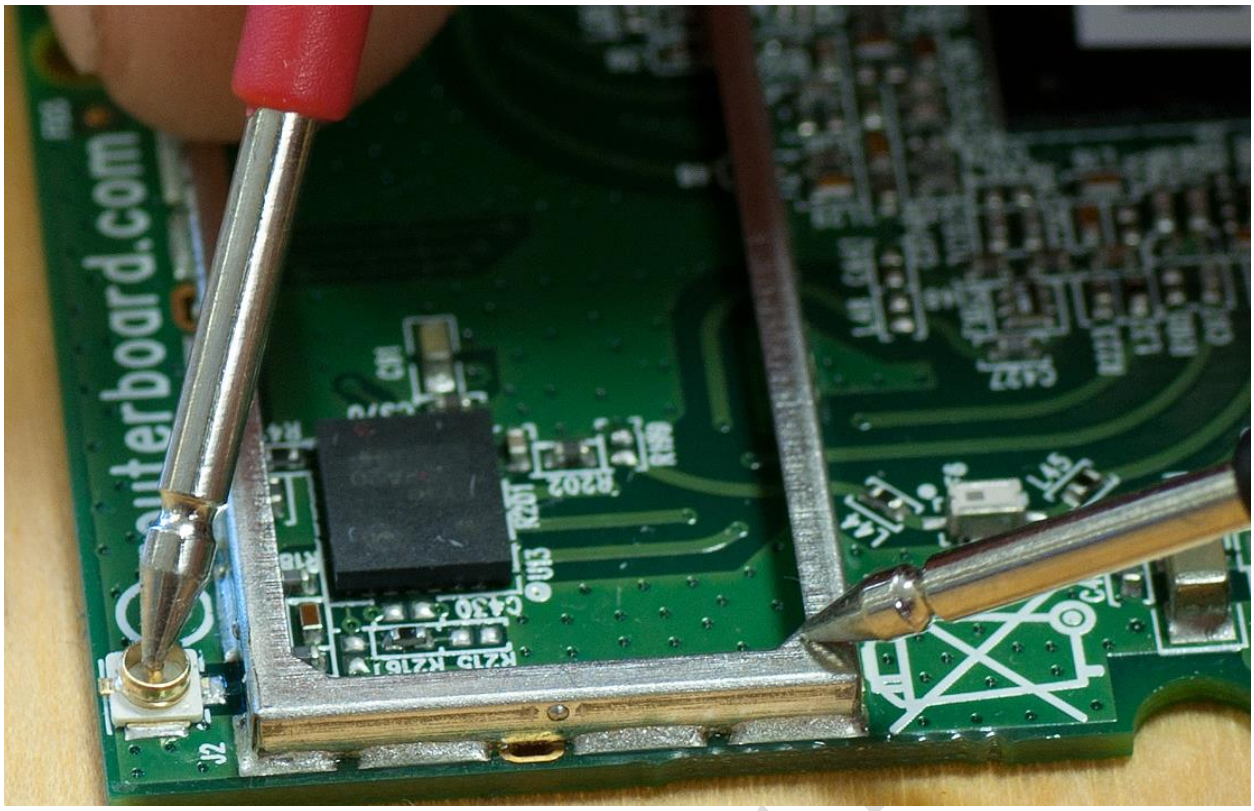
测试天线 chain 图片，下面是损坏的无线网卡：



下面这个 chain 是正常：

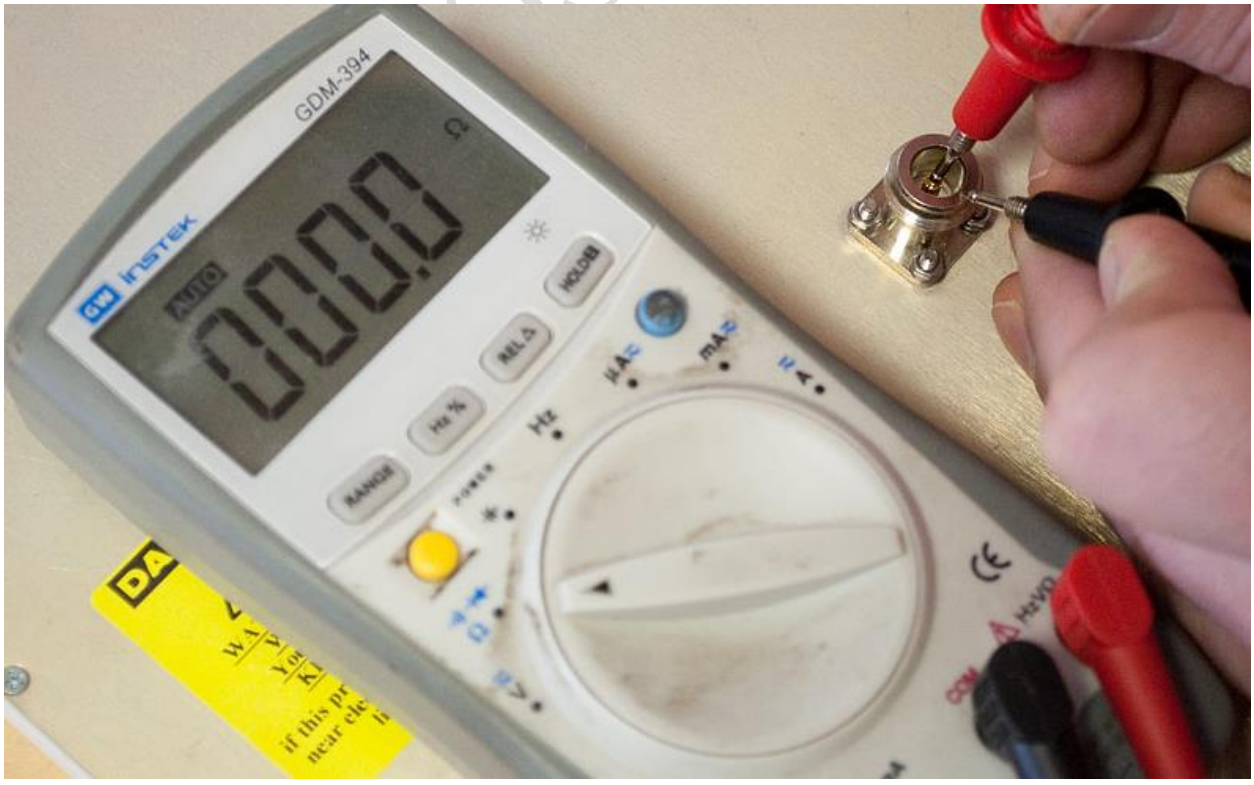


下面这张图展示了测试区域，一支笔测试 chain 接口的芯，一支测试金属盖表皮

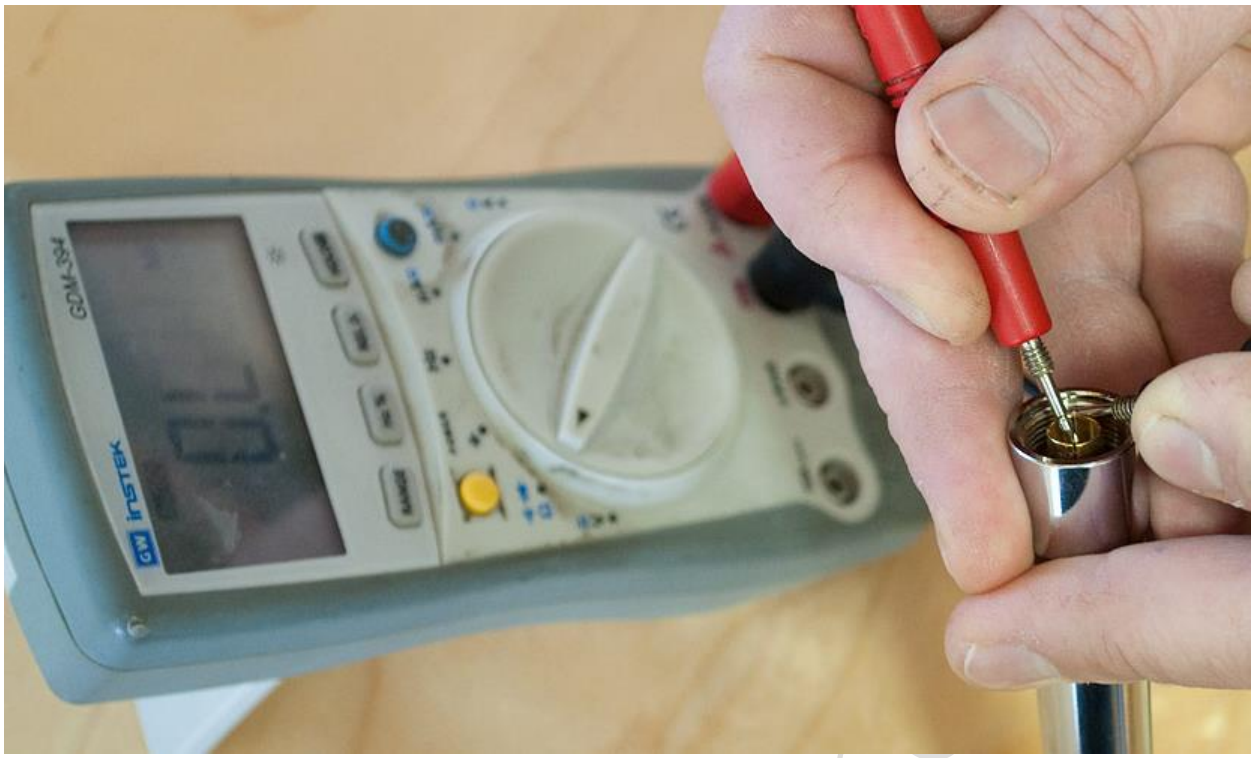


馈线短路测试

馈线用于连接天线和网卡，制作馈线或安装时可能会造成损坏，通常测试馈线是否短路也是使用万用表的电阻档，测试方法类似，我们通过测试馈线接头，如果出现如下图，表示馈线或接头已经短路

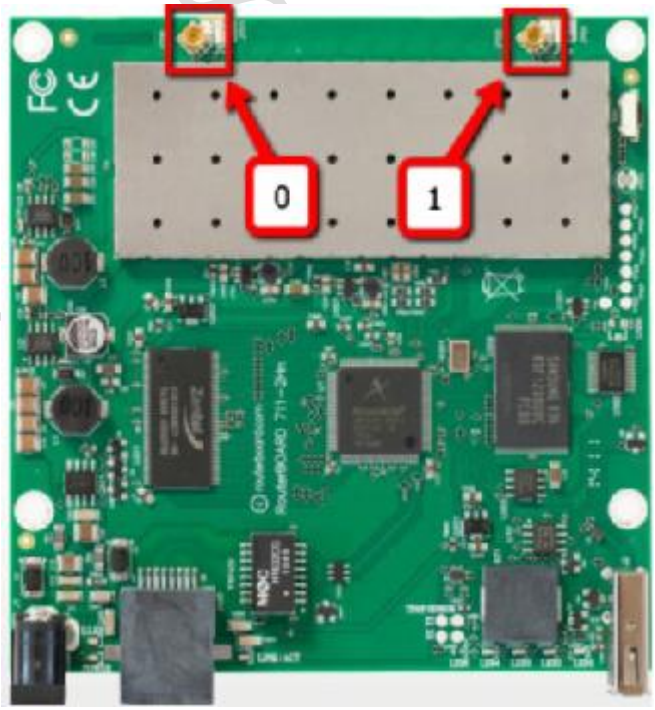


下面情况则表示馈线和接头正常，没有短路：

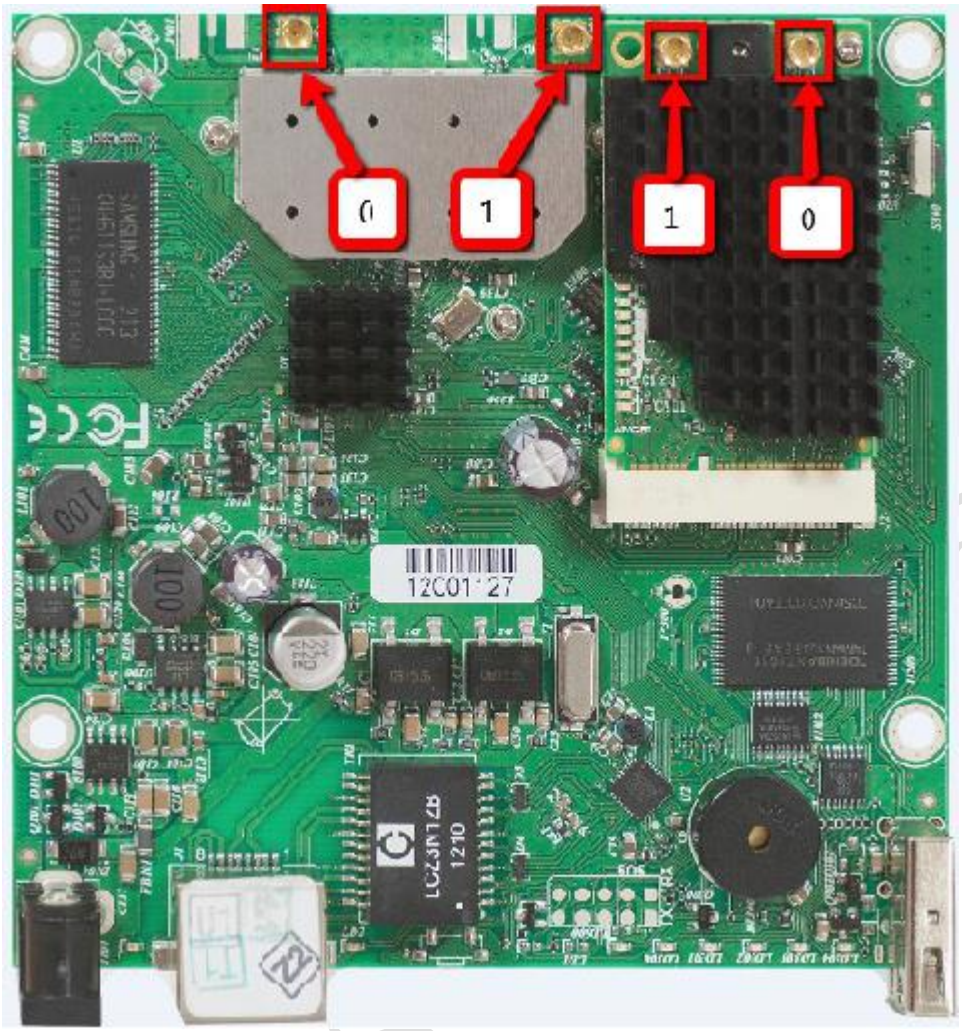


注意事项

1、RouterBOARD 第一次启动时无线网卡默认配置脚本会启用，11n 网卡的两个 chains 会同时开启，因此确保两个天线连接头与天线、至少要和跳线连接，避免造成功率放大器输出损坏。注意这张 RB911 的 11n 无线网卡 chain0 和 chain1 的位置



2、下面是 RB912 上安装了 RB11e 无线网卡，与板载的网卡 chain0 和 chain1 是相反的



3.6 RouterOS 支持的无线网卡

RouterOS 主要以支持 Atheros 厂商芯片为主，也支持部分其他厂商的芯片，所以你在挑选网卡是，一定要注意无线网卡的芯片。下面是 RouterOS 支持的大部分网卡芯片驱动（网卡数据仅供参考），如果需要使用 Superchannel 功能都必须采用 Atheros 厂商的芯片。

这里主要介绍 Atheros 芯片的无线网卡，当前市面上主要流行的 abg 网卡主要是 AR5212 和 AR5414 两种芯片，AR5414 在功耗和性能方面要优于 AR5212。abgn 方面主要以 AR9220 系列为主。Atheros 芯片提供无线信号的处理，而发射功率是由各个 OEM 工厂生产时，所使用的功率放大控制器决定，所以无线网卡的发射功率不是由芯片决定，我们需要看每个 OEM 厂商的数据确定具体参数。

下面是一张 MikroTik 的 R52Hn 的无线网卡，采用的是 AR9220 的芯片，支持 802.11abgn 属于双频无线网卡，当然只能同时工作在一个频率上，要么 bgn 模式，要么 an 模式。

下面是 R52Hn 参数的简单介绍

R52Hn	
802.11a	支持
802.11b	支持
802.11g	支持
802.11n	支持
无线接口	MMCX
接口类型	miniPCI
芯片	AR9220
输出功率	25dBm @ abgn
5GHz	4920 - 6100GHz（5MHz step）
2GHz	2.192 - 2.539GHz (5 MHz step)
重量	20g
工作温度	-50℃ to +60℃

这些参数是一张网卡的基本信息，如果要看他的具体性能，我们还要了解他的发射功率和接收灵敏度的数据，如下

下面是在不同协议下的接收灵敏度和发射功率的带宽情况

在 802.11b 协议下 1Mbit 带宽要求的信号接收灵敏度为-93dBm，而此时发射功率可以达到 24dBm；11Mbit 情况下同样是收灵敏度为-93dBm，而此时发射功率可以达到 24dBm。

802.11b	RX Sensitivity 接收灵敏度	TX Power 发射功率
1Mbit	-93	24
11Mbit	-93	24

在 802.11g 协议下 6Mbit 带宽要求的信号接收灵敏度为-94dBm，而此时发射功率可以达到 25dBm；54Mbit 情况下同样是收灵敏度为-81dBm，而此时发射功率可以达到 22dBm。

802.11g	RX Sensitivity 接收灵敏度	TX Power 发射功率
6Mbit	-94	25
54Mbit	-81	22

在 802.11bgn 下 MCS0 20MHz 为-94dBm，发射功率可以达到 25dBm，以下依次类推

802.11bgn 2.4GHz	RX Sensitivity 接收灵敏度	TX Power 发射功率
MCS0 20MHz	-94	25
MCS0 40MHz	-92	24
MCS7 20MHz	-78	21
MCS7 40MHz	-75	20

802.11a 协议情况下

802.11a	RX Sensitivity 接收灵敏度	TX Power 发射功率
6Mbit	-97	25
54Mbit	-80	21

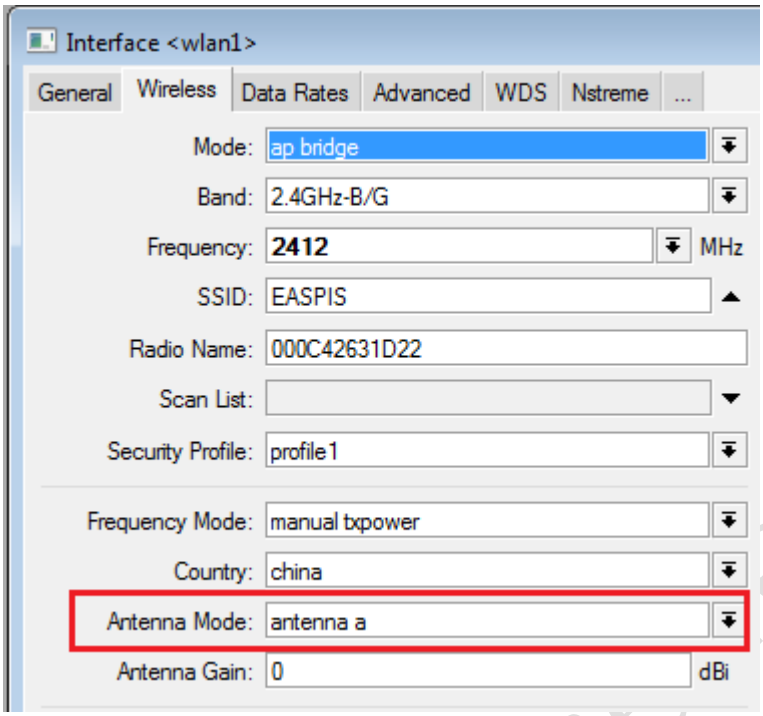
802.11an 协议情况下

802.11an 5GHz	RX Sensitivity 接收灵敏度	TX Power 发射功率
MCS0 20MHz	-97	24
MCS0 40MHz	-92	22
MCS7 20MHz	-77	18
MCS7 40MHz	-74	17

无线网卡都是时分工作方式，即接收和发送不是在同一时间完成的，是把时间划分开，一段时间用于发射信号，一段时间用于接收信号，即半双工模式。802.11abg 网卡和 802.11n 的网卡在接口上有所不同，一般情况下不管是 11abg 网卡，还是 11n 网卡界面都会是 2 个，但他们有所不同。

11abg 代表网卡是 AR5414，接口都有标明 MAIN 和 AUX，即一个主接口，一个辅助接口，默认情况是网卡采用 MAIN 发生和接收信号，AUX 是不工作的，如下图

11abg 网卡是采用 MAIN 发送还是 AUX 发送，我们可以在 RouterOS 中设置，默认是采用 MAIN 接口。如下图在 RouteroS 无线网卡中的 wireless 参数中的 antenna Mode 中设置，默认是选择 antenna-a，即 MAIN 接口，如果选择 antenna-b 则是 AUX 接口



11n 网卡的区别

而 11n 的网卡不同，因为采用了 MIMO 技术，所以分为 1×1 和 2×2，甚至 3×3 模式，即 150Mbit、300Mbit 和 450Mbit 的区别。如果 11n 有 1 个接口代表 1×1，有 2 个接口就是 2×2，每个接口都是一个独立的信号源，不会像 11abg 网卡区分 MAIN 和 AUX 的界面，只能由一个接口发射和接收信号。

无线网卡功率换算

通常我们可以看到无线网卡发射功率有 dBm 和 mW（毫瓦），即增益和功率两个参数，不过这两个参数是可以换算的，如下面是 dBm 和 mW 对照表

dBm	mW		dBm	mW
0	1.0 mW		26	400mW
1	1.3 mW		27	500mW
2	1.6 mW		28	640mW
3	2.0 mW		29	800mW
4	2.5 mW		30	1.0W
5	3.2 mW		31	1.3W
6	4.0 mW		32	1.6W
7	5.0 mW		33	2.0W
8	6.0 mW		34	2.5W
9	8.0 mW		35	3.0W
10	10 mW		36	4.0W
11	13 mW		37	5.0W
12	16 mW		38	6.0W
13	20 mW		39	8.0W

14	25 mW		40	10W
15	32 mW		41	13W
16	40 mW		42	16W
17	50 mW		43	20W
18	64 mW		44	25W
19	80 mW		45	32W
20	100 mW		46	40W
21	128 mW		47	50W
22	160 mW		48	64W
23	200 mW		49	80W
24	250 mW		50	100W
25	320 mW		60	1000W

3.7 RouterOS WLAN 构建常见问题

1、我应该把中心 AP 放在那里？

中心 AP 应该被放在一个地区的制高点，使得周围的用户都在视距范围内，例如高层建筑的屋顶，铁塔等

2、构建一个中心基站需要什么？

中心基站设备组成包括 MikroTik 无线路由器，全向天线或者扇区天线与设备连接的馈线、电源等，MikroTik 路由器连接有线网络， 路由器配置为桥接模式，用于连接无线和有线网络，通过全向天线将信号发送到周边的客户。

3、一个中心基站能连接多少个客户端？

支持 2007 个客户端，然而实际情况并不是如此，需要根据系统的性能和承载能力。实际环境中终端 PC 的数量，带宽情况和信号连接状态都会影响，802.11a 下支持 20-30 个左右客户比较合适，802.11bg 下接入端最好在 10-20 个客户端，如果你通过流量控制和数据过滤就能更好的对他们进行管理。

4、我需要连接一个客户端的网络，应该怎么做？

你需要一个客户端设备（CPE），例如一个 MikroTik 无线路由设备、以太网接口、定向天线、低损耗馈线。MikroTik 无线路由器可以为本地客户端的网络提供需多功能，如防火墙、NAT、流量控制、DHCP 服务等。定向天线应该安装在可以看见中心基站的位置。

5、每个系统的传输速度如何？

RouterOS 支持 802.11abgn 无线传输协议，2.4GHz 在 802.11b 模式下，数据传输是 11Mbps。然而实际吞吐量在 5-6Mbps。5GHz 在 802.11a 模式和 2.4GHz 的 802.11g 模式下，数据传输为 54Mbps。

5GHz 的 802.11a 模式下，为得到理想的带宽，在中心基站和客户端最好使用 800MHz 的 CPU。同样 RouterBOARD 系列建议使用 400 系列和 600 系列。所有用户都可以分配到相同的带宽。

6、能否限制每个用户的带宽？

是的，可以限制每一个用户的带宽速度，通过 RouterOS 的 queue 选项，如果你是 bridge 桥接无线网卡和有线网卡，请将 bridge setting 里的 use-ip-firewall 选项开启。

7、中心基站与客户端之间无线传输最大距离能达到多少？

最大距离和天线、馈线、传输功率和信号接收的灵敏度、周围环境和天线安放的位置等有关系。
在 2.4GHz, 中心基站和客户端通常不会超过 10-12km.
在 5GHz,我们已测试中心基站使用 17dBi 平板天线，客户端使用 30dBi 圆盘抛物面天线连接距离在 25 公里，实际传输速率 10Mbps

8、我能否从设备使用更长的馈线连接到天线？

可以，但无线传输距离和信号会受到影响。

9、我是否使用功率放大器增加距离？

可以，功率放大器有增强功率的作用，增加传输距离。同样他也可以连接馈线，增加馈线的传输距离。

10、无线连接是否要求在视线范围内？

是的，视距范围内总是被需要的，直接能看到对方，即两个连接点中间不能存在障碍物。

11、什么是 Fresnel 区?

Fresnel 区是一个视线区域的无线电波分布范围，这个区域必须无障碍，否则信号强度会被削减。例如 在一个 16 公里使用 5.8G 连接的无线，60%的 fresnel 区是一个 8.7 米的圆球区，在 2.4GHz 同样的距离是 13.6 米。

12、我是否可以将两个无线网络桥接？

是的，能使用 MikroTik 无线路由器建立透明桥接在两个设备间，具体可查看 RouterOS 技术文档。

13、安装一个 Wlan 无线系统需要多长时间？

一个基本的无线系统，如包含 3-5 个客户端的系统，在人员足够的情况下需大概 1-2 天时间

14、Wlan 运行在 Station 模式下是否能做桥接？

不能，station 模式不支持桥接功能，station 应用于三层的 IP 通信连接。

15、Wlan 桥接模式一般使用哪种？

一般 RouterOS 的桥接模式选择 ap-bridge 对 station-wds，需要开启 WDS 选项，并设置默认的桥接参数。

16、802.11n 能使用 wds 模式吗？

在 5.0 前 RouterOS 对 802.11n 仅支持 EoIP 隧道的传输模式，在 5.0 后启用 Nv2 协议后支持高带宽传输的 WDS 模式。

17、RouterOS 最大的 5G 传输带宽能达到多少

我们所测试到的单网卡，最大单向带宽在 5G-Turbo 模式下，可以达到 75Mbps，双向带宽在 40Mbps 左右，当然使用 802.11n 的 5G 传输，可以获得更高的带宽，合适的环境和设备下可以得到近 200Mbps 的带宽。

18、mode=bridge 模式支持那种连接方式

采用 bridge 模式只能支持与 ap-bridge、station-wds 和 bridge 连接的通信，即只支持点对点无线连接，如果你采用 RB411 无线设备 RouterOS 是 L3 级，那么 2 个 RB411 点对点通信只能使用使用 bridge 模式。在 5.0 版本后出现的 station-bridge 模式也是可以和 bridge 通信

19、什么是 Nstreme

Nstreme 是 MikroTik 独立开发的一套无线传输协议，是将多个帧进行重组，即将数据量较小的帧重新组合成大的帧进行转发，提高数据传输的效率，有助于 Wlan 无线传输带宽的提升，5.0 后 Nstreme 改进版本 Nv2（Nstreme version2）采用 TDMA 技术有效的支持了 11n 的高带宽传输。

20、什么是 Nstreme Dual

MikroTik 开发的双向传输协议，即每个设备采用两个无线模块，一个无线模块做 tx（发送），一个无线模块做 rx（接收）把数据接收发送分离成两个无线传输的方式，有助于提高无线传输的带宽和效率。

21、WLAN 与 WiFi 区别

WLAN 是 Wireless Local Area Network 的缩写，指应用无线通信技术将计算机设备互联起来，构成可以互相通信和实现资源共享的网络体系。无线局域网本质的特点是不再使用通信电缆将计算机与网络连接起来，而是通过无线的方式连接，从而使网络的构建和终端的移动更加灵活。Wi-Fi（WirelessFidelity），无线保真技术与蓝牙技术一样，同属于在办公室和家庭中使用的短距离无线技术。WI-FI 是 WLAN 的一个标准 WLAN 是无线局域网，无线局域网是由无线设备构成的，包括无线路由器或其他发射装置以及各种例如笔记本、平板计算机、手机等网络终端，设备之间是通过 WiFi 无线技术连接的。

22、RB751U

RB751U 集成一个 2GHz 802.11bgn 无线网卡，并内置了 PIF 2.5dBi 天线。同时提供一个外接天线的 MMCX 接头。由于该设备提供内置和外置天线，所以需要特别说明天线在 wireless 选项中的配置如下：

- Chain0

one antenna for TX

one antenna for RX

- Chain1

one antenna for TX/RX

MMCX 外接天线接口

如果启用 MMCX 接口，需设置天线模式为 antenna-b，在 wireless HT 菜单下禁用内置的 Chain1 天线

3.8 WiFi 覆盖

WiFi 覆盖上网，是现在主流，三大运营商、其他小运营商、公共场所、咖啡厅、酒店、高校等等，都在用 WiFi，还有一些餐饮行业演变的无线点餐系统，都在使用 WiFi 覆盖，大多采用的是 802.11bg 协议，而 802.11bgn（基于 2.4G）也在逐步增长。

对于 WiFi 覆盖我们采用的是 802.11bgn 协议，即 2.4G 频率（2412MHz~2462MHz，共 11 频道，再加上另外 3 个非中国标准的频道 2467，2472，2484），就算频率有 14 个频道，也只有 3 个不相互干扰的频道，如此少的干净频道，根本不能满足现在的需求，再加上环境的频率干扰，对 WiFi 质量造成极大的影响。这个问题在 2003 年后的成都已经比较严重，2.4G 的数据传输在城区传输很困难，虽然覆盖相对好些，但用户多了也凸显问题。

当然我们解决这个问题，可以选择的最简单的方法就是加大设备的发射功率，盖过其他的设备和干扰，但这个方法也被大多厂商采用，自然设备之间的抗干扰就此抵消。其他厂商就开始寻找其他方法，例如智能天线和修改 802.11 协议的访问（CSMA/CD）为 TDMA 等，我分析下 3 种特点

发射功率

我们传统意义上，认为设备发射功率高，可以提高传输距离和覆盖范围，但我们紧紧理解到的是发射功率，忽略了接收灵敏度，即设备接收到用户端的信号强度，这个参数很关键，我们虽然加大了发射功率，客户端收到了，但回传给设备，设备却无法收到，就像一个人说话加了扩音器声音特别大，但下面人说话他却什么都听不到，一样的没用，我们不仅要提高功率，还要注意接收灵敏度，RouterOS 上大多是扩展网卡，我们需要选择好的无线网卡，既要看发射功率，又要看接收灵敏度，至于网卡参数就看厂商是否凭良心说话了，还需要自己对比测试就知道

下面是R52H的参数

协议	输出功率	接收灵敏度
IEEE 802.11a:	24dBm	-90dBm @ 6Mbps
	19dBm	-70dBm @ 54Mbps
IEEE 802.11b:	25dBm	-92dBm @ 1Mbps
	25dBm	-87dBm @ 11Mbps
IEEE 802.11g:	25dBm	-90dBm @ 6Mbps
	20dBm	-70dBm @ 54Mbps

某厂家网卡

协议	输出功率	接收灵敏度
IEEE 802.11b:		-96dBm @ 1Mbps
		-90dBm @ 11Mbps
IEEE 802.11g:		-93dBm @ 6Mbps
		-74dBm @ 54Mbps

从上面的表我们可以对比下，不同网卡的接收灵敏度，在 802.11b 协议下-92dBm，R52H 速率为 1Mbps，而另外一个厂商的网卡的接收灵敏度在 1Mbps 是-96dBm，对比下 11g 协议也可以看出差别

在发射功率上，我们不仅要看发射功率，还要看看接收灵敏度，不同的 WiFi 都会写明这些参数，RouterOS 只要不是集成无线网卡的 RB 设备，都可以更换其他厂商的 miniPCI 无线网卡，给予灵活的发射功率选择，这点是 RouterOS 的优势，但选择无线网卡就是一个费心思的工作！

802.11 协议优化改进

我们知道标准的 802.11 协议访问采用的是（CSMA/CA）方式，即利用它检测和避免当两个或两个以上的网络设备需要进行数据传送时网络上的冲突。这个方式并非像以太网一样，有线以太网我们是可以看到各有 2 条线（1、2、3、6）进行接受和发送数据的，而 WiFi 则并不是，他采用的同一频率向用户发射数据，通

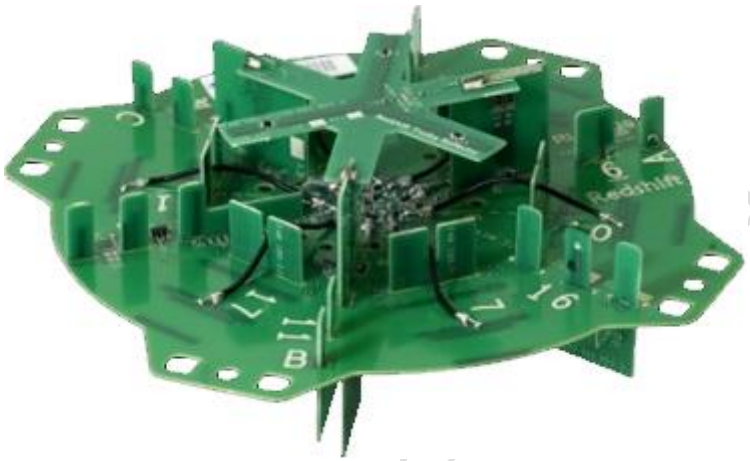
过时分间隔利用同一频率来接收用户回传数据，所以为什么有 CSMA/CA 利用 ACK 信号来避免冲突的发生，也就是说，只有当客户端收到网络上返回的 ACK 信号后才确认送出的数据已经正确到达目的地址。

当用户多后，在同一频率下为避免频率占用时间和采用载波监听冲突检查造成的效率下降，采用了时分多址技术，我们可以在 RouterOS 的 Nstreme 选项里找到 disable CSMA 和 Enable Polling 的选项，即禁用 CSMA，启用 Polling，MikroTik 早就采用轮询令牌的方式来解决冲突的问题，用来提高多点访问的问题，但这个技术仅适用于设备与设备之间，后来的 Nv2 协议，引入 TDMA 技术也是如此。

智能天线

一部分有实力的厂商开始引入智能天线，这种方式是采用多个全向天线，组成天线数组，如果你是一个军事发烧友，应该知道相控阵雷达，原理就和相控阵雷达一样，我采用多个天线发射和接收用户信号，相控阵雷达就像昆虫的复眼一样，多个天线模块组成，在同一平面扫描，实现同时跟踪多个目标

当多用户接入无线网络后，我们对用户的信号和访问进行处理，得到他在各个天线上的信号情况，当其中一个天线信号质量相对于其他天线要好，设备就将该用户的数据连接转移到指定信号好的天线，达到无线信号的优化，其实这样说也就是把文章做在了天线上，但其实也是通过软件和硬件结合实现的



不过智能天线大多采用的是全向天线，主要是接收来自各个方向的 WiFi 信号，包括各种反射回来的多路径信号，将这些信号收集处理，给 AP 设备优化用户信号，这种反射多路径信号在室内和建筑结构复杂的环境有很大的优势，能提高 AP 的覆盖范围，但在室外空旷地带，他就无法和定向天线的 AP 比较，因为全向天线方向性差，制约了覆盖距离，室外空旷地带没有那么多反射点，智能天线也不能发挥他的优势，所以要看情况而定，智能天线的优势更多集中在室内

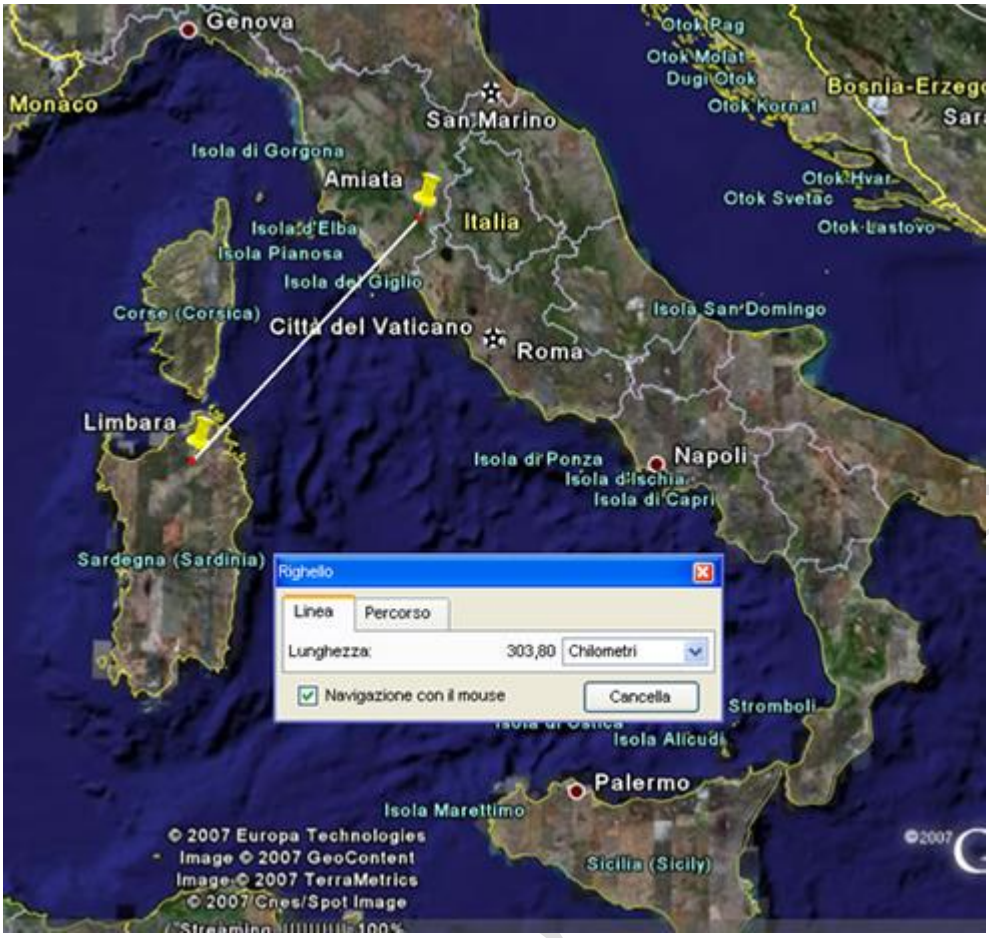
RouterOS 当然无法实现智能天线的技术，因为他一套模块组合的系统，非一套完整的 AP 设备解决方案，就算有也没有天线技术的支持。

3.9 WLAN 无线数据传输

WLAN 无线传输，和覆盖不一样，最简单是点对点的无线传输，也许对 RouterOS 无线了解的人应该知道下面的信息：

在 2007 年 6 月 16 日，在意大利已经实现一个 304 公里的无线连接（100 公里陆地和 200 公里海平面），在 Amiata mount (1734asl)与 Limbara mount (1300 ca asl) 之间。

采用一对基于 RouterOS 配置的无线主板，一对 XR5 802.11a 600mW 无线网卡，一对自制 120cm 圆盘式高增益天线，使用水平极化，两端的信号强度从-58 dBm 到-62dBm，速率在 12 到 48Mbps



300 公里的数据传输，的确很牛！当然 RouterOS 运用了自己的私有协议 Nstreme，优化 ACK 时间、压缩帧数据、多点访问时采用轮询技术等，新的 Nstreme Version 2（Nv2），之前提到引入了 TDMA 协议，实现更多的设备接入，优化 11n 的传输，而且降低了系统资源消耗。如果用 Nv2 协议 11n 的带宽可以达到近 200Mbps 的 TCP 数据传输，这也是 RouterOS 最大的优势。

为什么 RouterOS 在 11n 的产品上前期都集中在 5G 的 802.11an 上，也是他主攻传输市场的原因，如果做 WiFi 覆盖的确没有太多技术优势,但也并非否定 RouterOS 的 WiFi 覆盖，如果加上大功率，高接收灵敏度的无线网卡，同样也能得到较好的覆盖效果，只是与智能天线设备在室内覆盖和复杂结构环境下，就要差点了。至于周围环境干扰严重的情况下，是任何无线设备都不能回避的问题。

第四章 RouterOS 无线功能介绍

要求功能包： wireless

RouterOS 无线协议遵循 IEEE 802.11 标准，并完全支持 802.11a、802.11b、802.11g 和 802.11n，并增加了如 WPA、WEP 和 AES 加密, Wireless Distribution System (WDS), Dynamic Frequency selection（DFS 动态频率选择）, Virtual Access Point（虚拟 AP）, 以及 MikroTik 的 Nstreme 和 NV2 私有协议等等。

无线能工作在多个模式下：station（client）, access point（AP）和 wireless bridge 等，station 模式也可以分为多个模式，关于完整的介绍，可以参考 station 模式介绍

4.1 RouterOS 无线介绍

MikroTik RouterOS 当前所支持的协议：

- **2.4ghz-b** - IEEE 802.11b
- **2.4ghz-b/g** - IEEE 802.11b 与 IEEE 802.11g
- **2.4ghz-g-turbo** - IEEE 802.11g 支持 108 Mbit
- **2.4ghz-only-g** - IEEE 802.11g 支持 54 Mbit
- **5ghz** - IEEE 802.11a 支持 54 Mbit
- **5ghz-turbo** - IEEE 802.11a 支持 108 Mbit
- **2ghz-b/g/n** - IEEE802.11bgn (基于 4.0beta3 以上版本)分别兼容 2.4GHz 频段的 11Mbit、54Mbit 和 150Mbit~450Mbit
- **2ghz-onlyn** - IEEE802.11n (基于 4.0beta3 以上版本)仅支持 2.4GHz 频段的 150Mbit~450Mbit
- **5ghz-a/n** - IEEE802.11an (基于 4.0beta3 以上版本)分别兼容 5GHz 频段的 54Mbit 和 150Mbit~450Mbit
- **5ghz-onlyn** - IEEE802.11n (基于 4.0beta3 以上版本)仅支持 5GHz 频段的 150Mbit~450Mbit

MikroTik 提供了强大的无线设置功能，同样产品包括针对 Wlan 开发的 RouterBOARD 硬件，型号从 RB100、RB500、RB400、RB600、RB700、RB800、RB900 和一体无线设备等（具体型号和参数可以登录官网 www.routerboard.com）能应用在 802.11abgn/ac 的无线点对点、点对多点、漫游和覆盖等方案中。

4.2 RouterOS 支持的 WLAN 连接方式

MikroTik RouterOS 提供了多无线连接方式：**点对点连接、点对多点连、无线接力、无线漫游等、独有的 bonding、Nstreme、Nstreme2 协议、Nv2 协议和 Mesh 网状网络**。

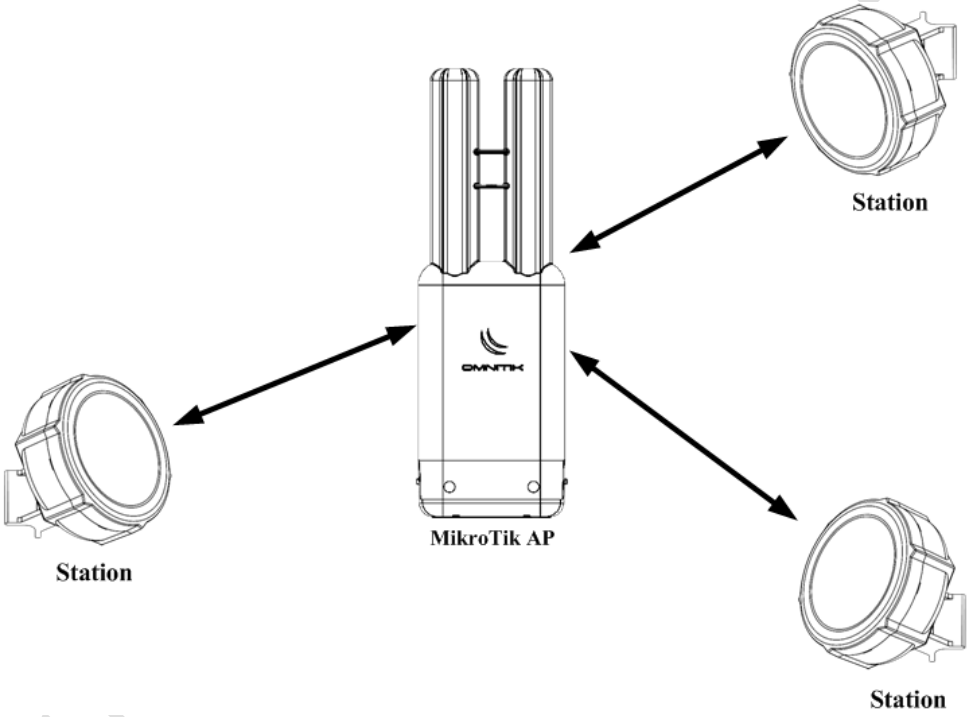
点对点连接

当两个网络之间可采用点对点的无线连接方式。只需在每个网段中都安装一个 AP，通过点对点传输实现网络信号的传输和互联。在点对点连接方式中，天线最好全部采用定向天线，已得到更好的信号和带宽。

点对点方式有两种一中是我们常见的的桥接模式，我们可以采用 ap-bridge 或者 birdge 方式，在这里我们推荐使用 ap-bridge 与 station-wds 的桥接方式。

点对多点连接

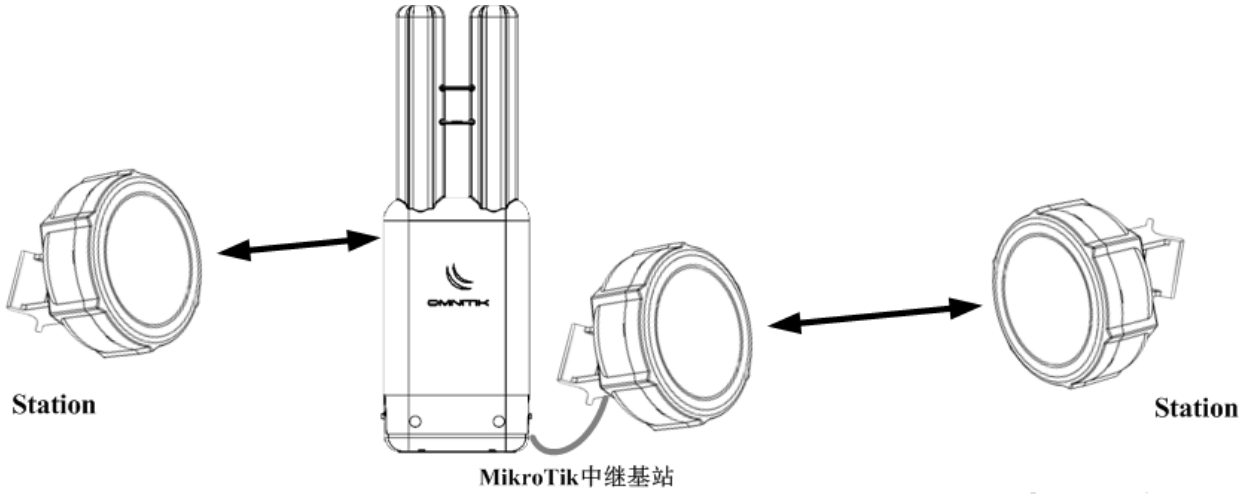
当三个或三个以上的网络之间采用光纤或双绞线等有线方式难以连接时，可采用点对多点的无线连接方式。只需在每个网段中都安装一个 AP，即可实现网段之间点到点连接，也可以实现有线主干的扩展。如下图



在点对多点连接方式中，一个 AP 设置为中心的 ap-bridge，其他接收机站则全部设置为 station 或 station-wds。在点对多点连接方式中，中心点一般采用全向天线或者扇形面的天线，客户端则最好采用定向天线。

无线中继

当两个网络间的距离已经超过无线网络产品所允许的最大传输距离时，或者虽然两个网络间的距离并不遥远，在两个网络之间有建筑或其它物体阻挡，可以寻找一个中继点实现传输信号的接力，如下图



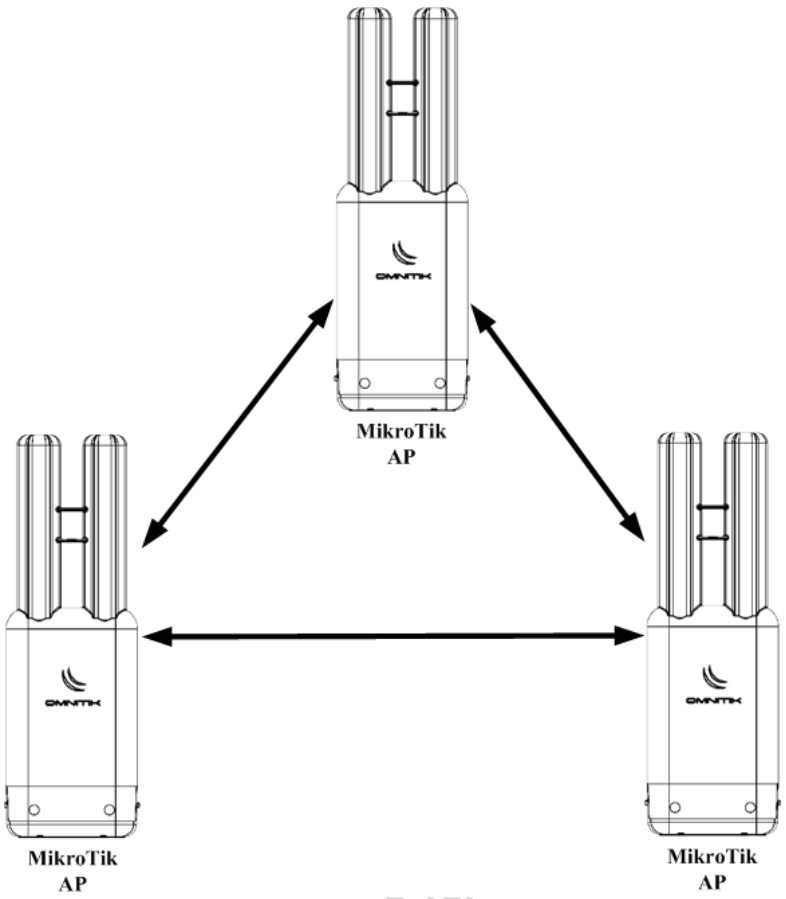
无线接力工作方式与点对多点非常类是，但无线接力是指在一个设备上添加两张或两张以上的网卡，做中继传输，他和点对点或点对多点不同的地方在于，一个 AP 上有多个 wlan 网卡接口，需要将它们做成桥接或者路由方式，多无线网络有助于提高数据的转发量。

无线漫游（WDS）

多个中心基站设备可以为在网络范围内各个位置之间漫游的移动式无线客户机工作站设备服务。 多基站配置中的漫游无线工作站具有以下功能：

- 在需要时自动在基站设备之间切换，从而保持与网络的无线连接。
- 只要在网络中的基站设备的无线范围内，就可以与基础架构进行通信。

在城市某区域或者在网络跨度很大的大型企业中，人们可能需要完全的移动上网需求，此时，可以在网络中设置多个 AP，使装备有无线网卡的移动终端实现如手机般的漫游功能（如图 4）。使用无线漫游方案，随时随地访问他们所需要的网络资源。

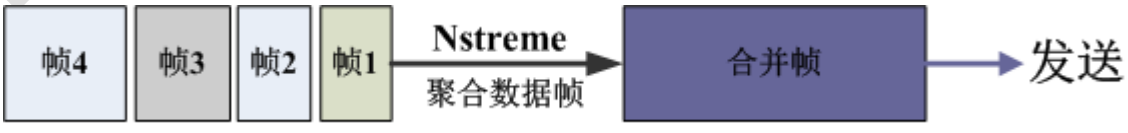


这就是所谓的无缝漫游，在移动的同时保持连接。原因很简单，AP 除具有网桥功能外，还具有传递功能。这种传递功能可以将移动的工作站从一个 AP“传递”给下一个 AP，以保证在移动工作站和有线主干之间总能保持稳定的连接，从而实现漫游功能。

Nstreme 与 Nstreme v2

这功能属于 MikroTik 专有的无线协议，在长距离和带宽上有非常突出的表现， 随着无线协议的发展从早期支持 802.11abg 的 Nstreme 协议开始，到现在的 Nstreme v2 全面支持 802.11n 协议，同时 Nstreme v2 引入了 TDMA 技术，能更好的支持点对多点的连结也兼容 802.11abg 协议。

Nstreme 协议主要特点是通过将多个帧重组后，将多个帧重新组合成一个巨帧一次发送，目的是减少无线传输时过多的 ACK 请求，一些小帧过多的消耗无线连结的资源，通过 Nstreme 协议合并后，减少了 ACK 重复发送的请求，特别是在信号较差、距离较远的情况下。



早期的 Nstreme 协议支持三种模式：

- **Point-to-Point mode** – 点对点模式通过在每一个点架设一个无线设备实现
- **Dual radio Point-to-Point mode (nstreme2)** – 这个协议是通过在每一个点架设两套无线设备，同时分别一对做接收和一对做发送，实现双向的通信。能做到高速连接。

- **Point-to-Multipoint** – 点对多点模式的客户论旬（类似 AP 控制的令牌环）

Nstreme 协议是专用于长距离无线传输，正常的无线连接在长距离传输时，会产生高传输延迟。使用 Nstreme 协议后这个问题被消除。 当我们要求高速数据传输的时候，需要保证足够的上下行带宽时，我们可以选择 Nstreme Dual 的模式，一对网卡做上行，一对做下行数据传输。

Nstreme 协议更多应用在 802.11abg 上，随着 802.11n 的出现新的 Nstreme v2 协议提高了传输带宽和多点的性能。

Nstreme Version 2（Nv2）

Nv2 是 MikroTik 为 802.11n 优化的私有无线协议，基于 TDMA 技术（Time Division Mulitple Access 时分多址），Nv2 基于 TDMA 好处在具有更大的吞吐量、低延迟、适用于点对多点网络连接。

TDMA 是一个频段访问共享网络，允许多个用户在同频率下通过在不同时间段信号间隔访问方式，在属于他自己的时间间隔内，每一次用户传输一连串的数据。这个允许多个网站共享相同的传输介质，在一段时间内使用一部分的频率信道

工作在 Atheros 芯片上，AR5212 和更新的 AR5414，802.11n 系列芯片包括 R52n、R52Hn、AR9220 和 AR9300 系列芯片等，从 RouterOS v5.0beta5,你可以在 wireless 菜单下配置 Nv2，Nv2 协议限制了 511 个客户端。

Nv2 最重要的好处：

- 增加传输速率，特别是在 802.11n 模式下
- 更多的客户连接到点对多点网络（PTM）
- 更低的延迟
- 没有传输距离限制

从 RouterOS v5.0beta5,你可以在 wireless 菜单下配置 Nv2，Nv2 协议限制了 511 个客户端

下面是 Nstreme 协议支持在不同协议下与其他标准 WLAN 无线产品的速率比较情况：

标准	工作频段	频率占用空间	理论最大速率	标准 WLAN	Nstreme
802.11b	2.4GHz： 2312-2599MHz	5MHz	11Mbps	5.5Mbps	**7Mbps**
802.11g **G-Turbo**	2.4GHz 2312-2599MHz	5MHz 44MHz	54Mbps 108Mbps	27Mbps 54Mbps	**37Mbps** **68Mbps**
802.11a **A-Turbo**	5GHz： 4920-6100MHz	5MHz 10MHz 20MHz 40MHz	13.5Mbps 27Mbps 54Mbps 108Mbps	6.75Mbps 13.5Mbps 27Mbps 54Mbps	**9Mbps** **18Mbps** **37Mbps** **74Mbps**
802.11n	2.4/5Ghz： 2312-2599Mhz	5MHz	37.5Mbps	18.5Mbps	**28Mbps ‘**

	4920-6100Mhz	10MHz 20MHz 40MHz(2×20MHz) 60MHz(3×20MHz)	75Mbps 150Mbps 300Mbps 450Mbps	37.5Mbps 75Mbps 150Mbps ？？	**50Mbps** **100Mbps** **200Mbps** **？？**

Mesh 无线网状网络

MikroTik 支持的 STP 和 HWMP+两种方式的 Mesh：

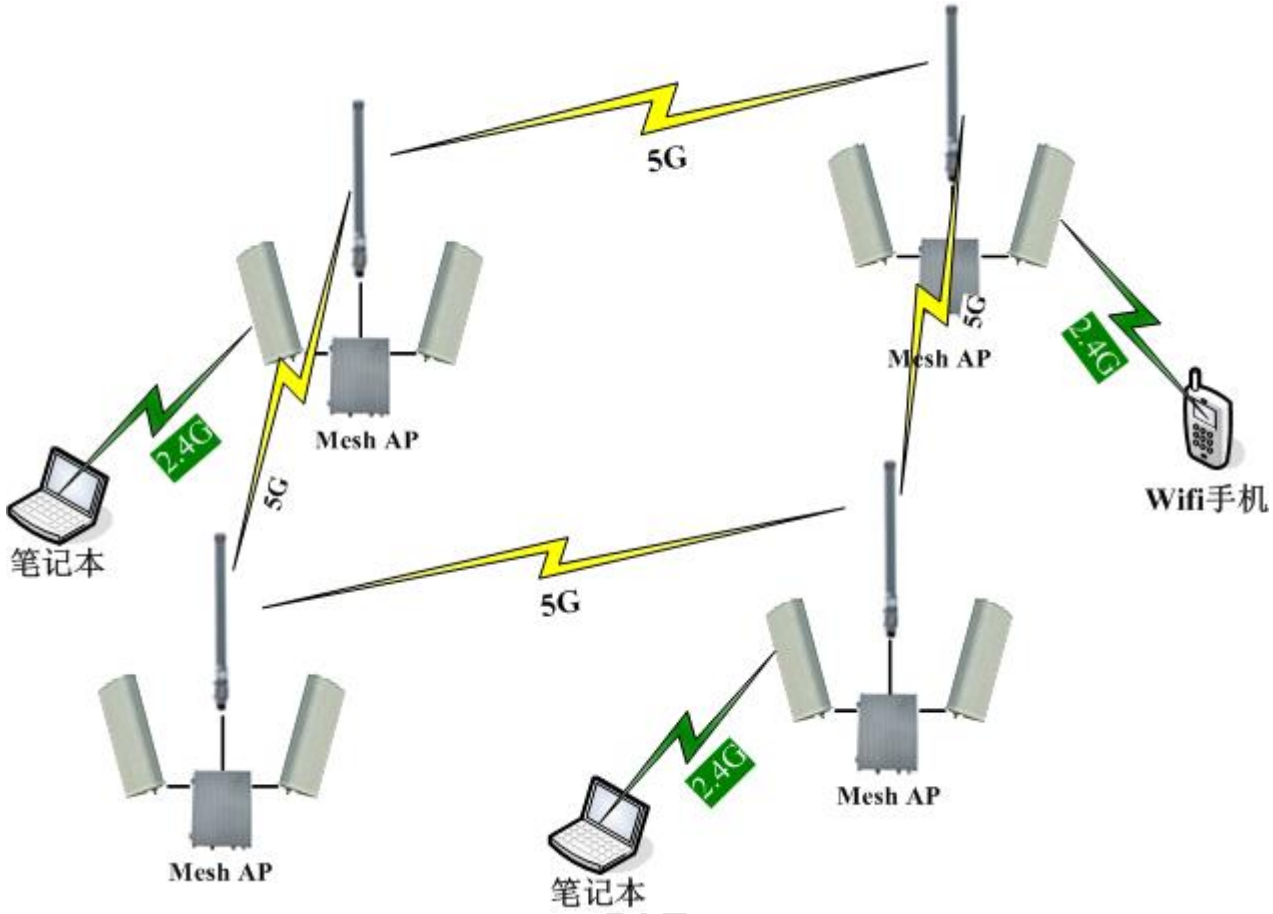
STP 生成树协议的英文缩写。该协议可应用于环路网络，通过一定的算法实现路径冗余，从而避免报文在环路网络中的增生和无限循环。在 v3.0 后支持快速生成树协议 RSTP，在收敛速度上更快，通过优先级划分线路的优先路径。

HWMP+是 MikroTik 为无线网状网络 Mesh 定义的 2 层路由协议。基于 IEEE802.11s 草案 Hybrid Wireless Mesh Protocol (HWMP)，能用于替代 STP 生成树协议确保环路的最优路径。这种分布式系统不仅能应用到无线分布系统（WDS）。HWMP+网状网络同样也支持以太网接口的网状网络，因此你可以用于简单的以太网分布系统，或者同时连接 WDS 和以太网。构建一个大型的无线网络，如城市 Wlan 网络。

STP 和 HWMP+协定的 Mesh 在 MikroTik 特性：

每台 AP 的无线模块发射模式都配置为 ap-bridge，根据需要配置频率和 SSID。骨干与覆盖 SSID 不同，但骨干与骨干之间 SSID 相同，覆盖也一样。

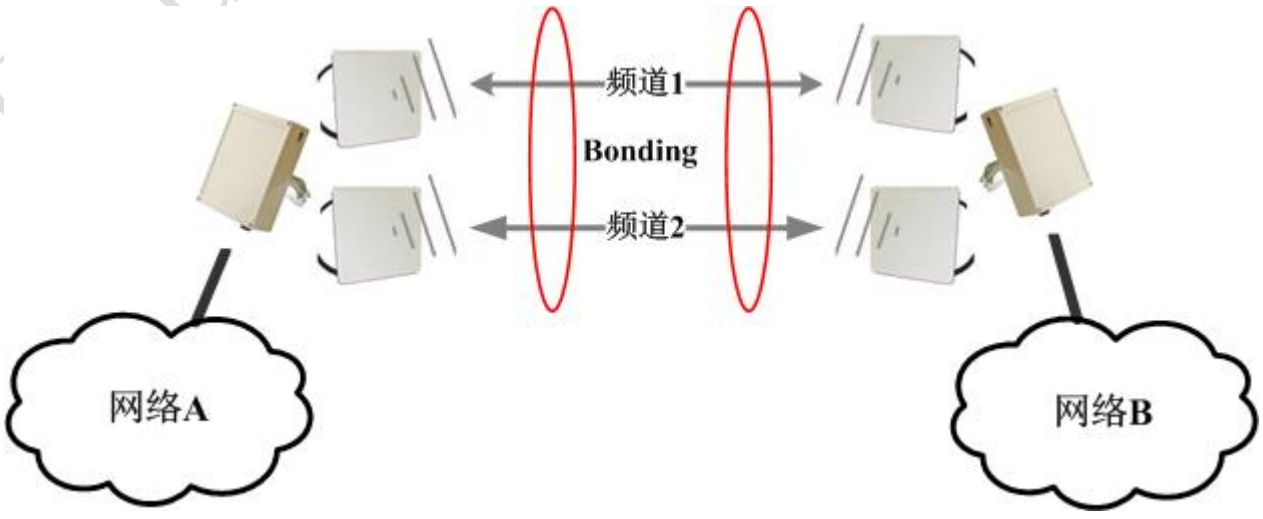
WDS 模式：v2.9 选择 dynamic，v3.0 选择 dynamic-mesh（v3.0 的 dynamic-mesh 效率要比 v2.9 的高）配置 Bridge 或者 Mesh 参数，并将指定的接口定义入 Bridge 或 Mesh 中，但 Bridge 和 Mesh 不能被同时使用。



MikroTik bonding 功能

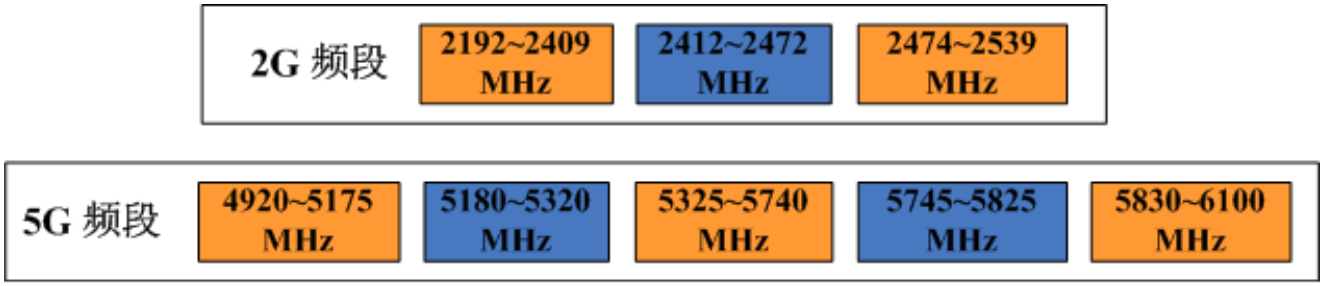
Bonding 是通过汇聚多个接口到一个虚拟的连结上，这种方式可以获得更高的带宽或提供失效转移接管。 Bongding 操作必须用于二层链路层，不支持三层 IP 层的应用。（关于 RouterOS bonding 介绍请参考**《RouterOS 中文网络教程》**）

通过 Bonding 功能，可以将两条或者两条以上的无线链路绑定在一起，起到将无线网络合并带宽的作用，如下图，是为了提高网络 A 和网络 B 直接的带宽，将 2 条无线绑定在一起，也可用起到线路备份的作用：

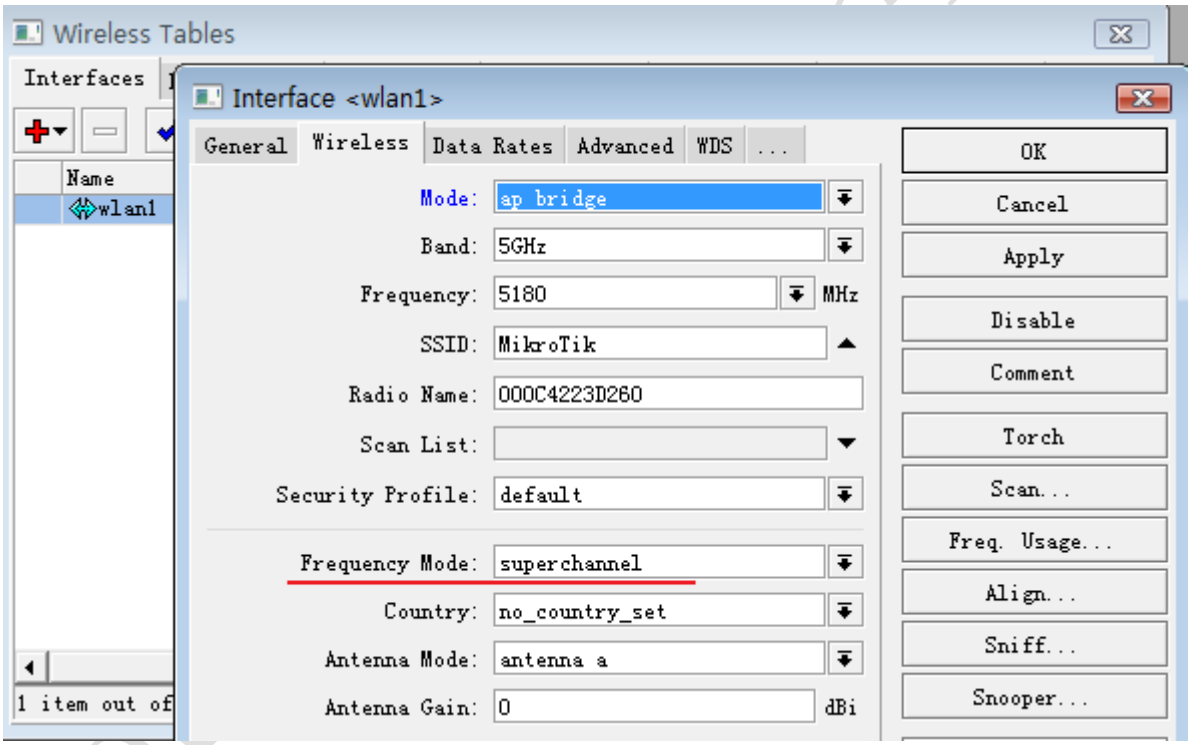


MikroTik Superchannel

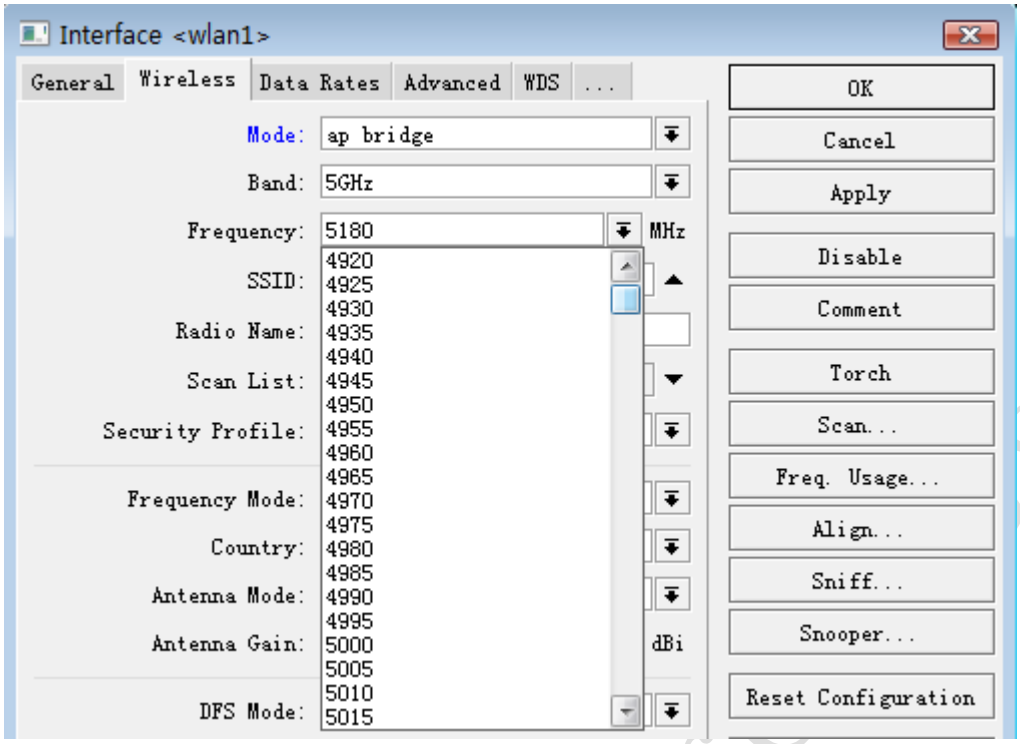
MikoTik 的 Superchannel 模式是，增加了无线的发射频率，能够获取更多的无线通道，避免 WLAN 因为较少的通道而带来的干扰问题，RouterOS 支持的 2G 和 5G 频率范围：



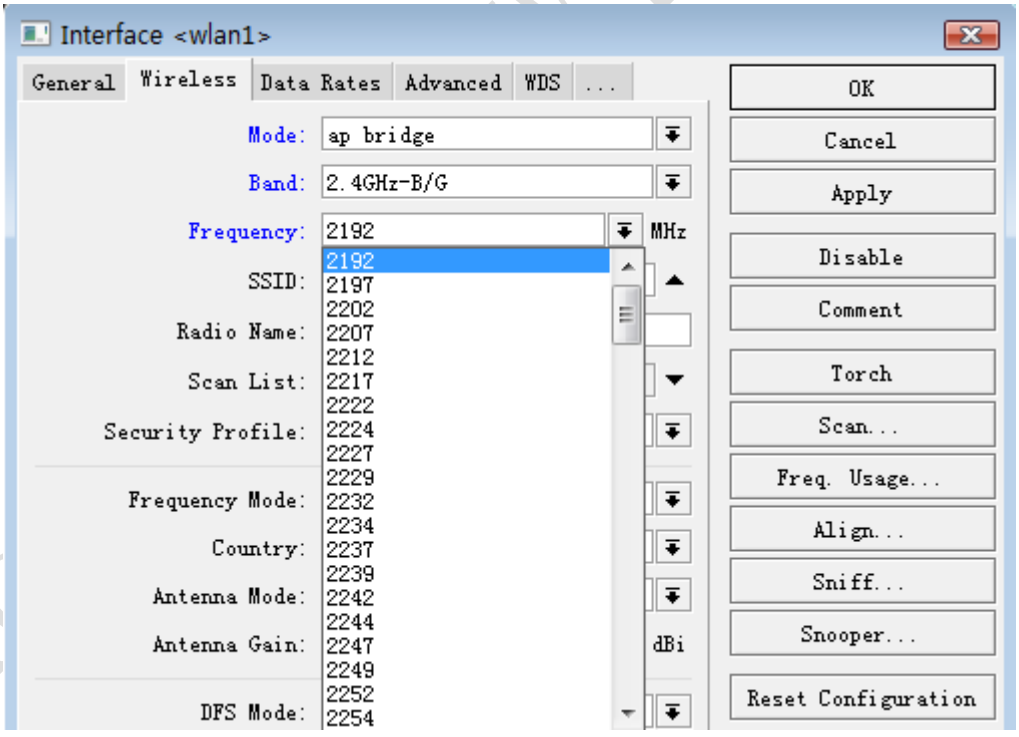
我们可以进入 wireless 目录下选择 wlan1 后，点击 Advance Mode，看到 Frequency-Mode 选项，并选择为 superchannel，这样我们便可以将超级频道启动（升级超级频道需要另外注册 key）如下图：



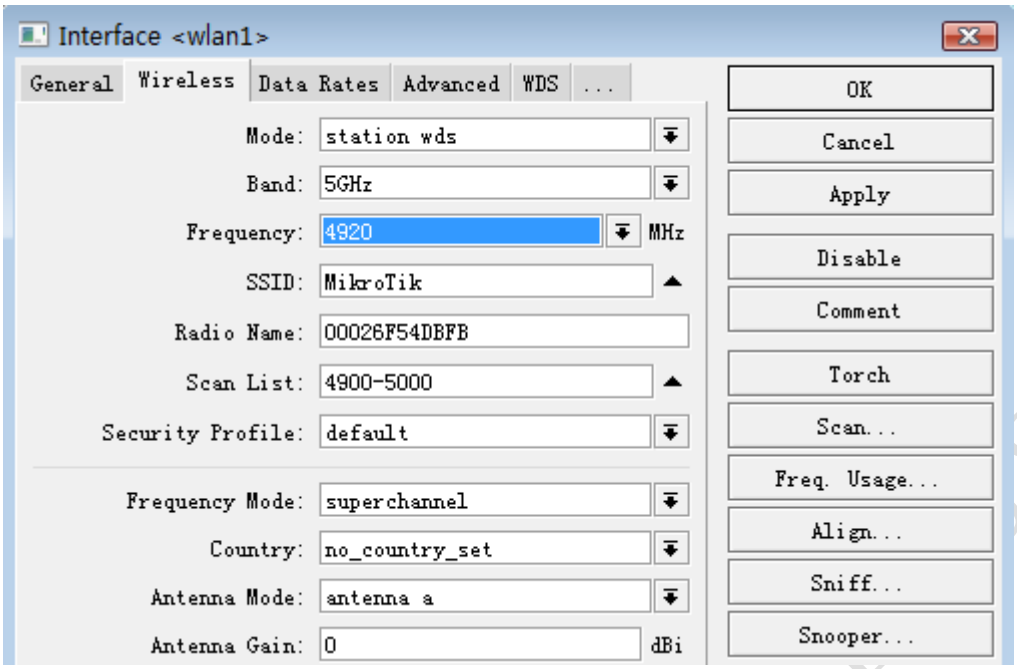
启动超级频道后，我们可以看到 5GHz 频道下可以从 4920MHz~6100MHz，每 5MHz 一跳，如下图



2.4G 频段，是从 2192MHz~2539MHz，也是每 5MHz 一跳，如下图：



注意：RouterOS 设置 superchannel 时，当 AP 端设备使用 ap-bridge 设置为超级频道后，在 station 或者 station-wds 搜索频道时，只会按照默认频率搜索，搜索时间会很长，可能会花费 1～2 分钟时间，为了提高连接效率，我们需要在 scan-list 设置搜索频率，如下图，设置 4900-5000MHz 的频率搜索范围



4.3 RouterOS 802.11 协议

802.11 二层桥接限制

802.11 数据帧格式

首先要说明的是 802.11 的帧格式很特别，它的长度是可变的。不同功能的数据帧长度会不一样。这一特性说明 802.11 数据帧显得更加灵活，然而，也会更加复杂。802.11 的数据帧长度不定主要是由于以下几点决定的

mac 地址数目不定，根据帧类型不同，mac 802.11 的 mac 地址数会不一样。比如说 ACK 帧仅有一个 mac 地址，而数据帧有 3 个 mac 地址，在 WDS 模式（下面要提到）下，帧头竟然有 4 个 mac 地址。我们看看 802.3 的帧结构

802.3 MAC 帧格式。

Preamble	SFD	Dst	Src	Length	Data	FCS

帧内容：

- Preamble（前导序列）：由 62bit 交替出现的 0,1 序列组成。设置目的,接收端物理层同步位时钟。
- SFD（起始域）：“11”表示有用资料开始。
- Dst(目的地址域):由 6 字节组成,表目的节点地址。
- Src(源地址域):由 6 字节组成,表源节点地址。
- Length(长度域):由 2 字节组成。,数据域长度。
- Data(数据域):46 字节~1500 字节之间。
- FCS(校验域):4 字节组成。

802.11 帧结构

Frame Control	Duration	RA	TA	DA	Seq	SA	Data	FCS

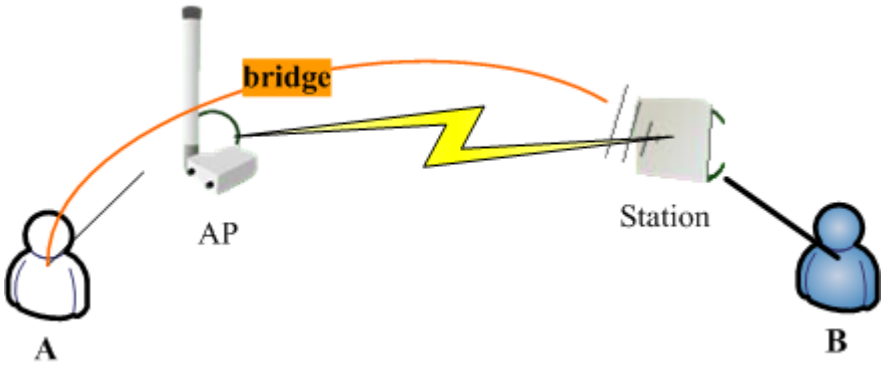
我们可能会碰到以下类型的 mac 地址

- RA(receiver address)：无线网络中，该数据帧的接收者
- TA(transmitter address)：无线网络中，该数据帧的发送者
- DA(destine address)：该帧的目的 mac 地址
- SA(source address)：该帧的源 mac 地址

这里的 DA 和 SA 含义和普通以太网中的 Dst 和 Src 地址含义一样，在无线网络中可能我们需要通过 AP 把数据发送到其它网络内的某台主机中。但是有的人会奇怪，直接在 RA 中填这台主机的 mac 地址不就久好了么。但是请注意 RA 的含义，说的是无线网络中的接收者，不是网络中的接收者，也就是说这台目的主机不再无线网络范围内。在这种情况下我们的 RA 只是一个中转，所以需要多出一个 DA 字段来指明该帧的最终目的地，当然，如果有了 DA 那必须有 SA，因为若目的主机要响应的话，SA 字段是必不可少的。(假设没有 SA 字段，那么目的主机响应的数据报就只能发送到源主机所属的 AP 上了)

我们要知道 Access Point 是一个访问节点，即 AP 是一个用于终端设备网络连接的节点，终端设备包含了各种 PC、笔记本和各种设备，我们可以称他们为 station，但后来我们需要通过 802.11 协议实现网桥传输，但是最初 802.11 协议的 AP 被期望能通过无线桥接二层的帧，但 station 设备并没有考虑二层的桥接

我们考虑一下网络：



这里 A 到 AP 与 station 到 B 都是通过以太网连接，但是 AP 到 Station 是通过无线。根据 802.11，AP 能透传二层桥接在 A 和 Station 之间，但不能实现桥接传输在 AP 和 B 之间，或者 A 和 B 之间。

802.11 协议标准指定帧在 station 和 AP 设备之间 只能传输帧的头部包含 3 个 MAC 地址

帧传输从 AP 到 station 包含以下 MAC 地址

- TA(无线发送者地址) - AP 的地址
- DA(目标地址)- station 设备的地址，也是无线接收者的地址
- SA(源地址)– 发送者的源 MAC 地址

帧传输从 station 到 AP 包含以下 MAC 地址：

- RA(无线接收者地址)- AP 的 MAC 地址
- DA(目标地址) – 达到的目标地址

- SA(源地址) - station 设备的地址，即无线传输者 MAC 地址

这样每个帧都包含无线发送者和接收者的 MAC 地址，包含 3 个 mac 地址的帧格式是不能适应透明的二层桥接，因为 station 不能发送与自己不同的源 MAC 地址，例如从 B 来的帧，并且同时 AP 也不能在帧里包含 B 的地址

在 WDS 模式下，一个 AP 互相连接的系统数据帧会有 4 个地址，RA，TA 表示接收端和发送端，这两个地址用于无线传输的时候。还有 2 个地址是 DA 和 SA，分别跟以太网中一样表示源地址和目的地址

- RA(receiver address)：无线网络中，该数据帧的接收者
- TA(transmitter address)：无线网络中，该数据帧的发送者
- DA(destine address)：该帧的目的 mac 地址
- SA(source address)：该帧的源 mac 地址

这样的帧在无线连接里包含了必要的二层网桥信息,但不幸的是 802.11 协议没有指明 WDS 如何建立和维护无线网络的连接，因此任何使用 4 个 MAC 地址的帧需要进行明确的执行。

4.4 基本无线速率和 MCS 速率

基本速率是无线 AP 设备要求终端必须满足的速率条件，在满足这个条件下才能连接到 AP，下面是关于 802.11n 的 MCS 对应速率表，MCS 0～7 使用单条空间流，即 MIMO 为 1×1，当 MCS＝7 时，速率值最大。MCS 8～15 使用两条空间流，即 MIMO 为 2×2，当 MCS＝15 时，速率值最大。

MCS 对应速率表（带宽为 20MHz）

MCS 索引	空间流数量	调制方式	速率(Mb/s)	
			800ns GI	400ns GI
0	1	BPSK	6.5	7.2
1	1	QPSK	13.0	14.4
2	1	QPSK	19.5	21.7
3	1	16-QAM	26.0	28.9
4	1	16-QAM	39.0	43.3
5	1	64-QAM	52.0	57.8
6	1	64-QAM	58.5	65.0
7	1	64-QAM	65.0	72.2
8	2	BPSK	13.0	14.4
9	2	QPSK	26.0	28.9
10	2	QPSK	39.0	43.3
11	2	16-QAM	52.0	57.8
12	2	16-QAM	78.0	86.7
13	2	64-QAM	104.0	115.6

MCS 索引	空间流数量	调制方式	速率(Mb/s)	
			800ns GI	400ns GI
14	2	64-QAM	117.0	130.0
15	2	64-QAM	130.0	144.4

MCS 对应速率表（带宽为 40MHz）

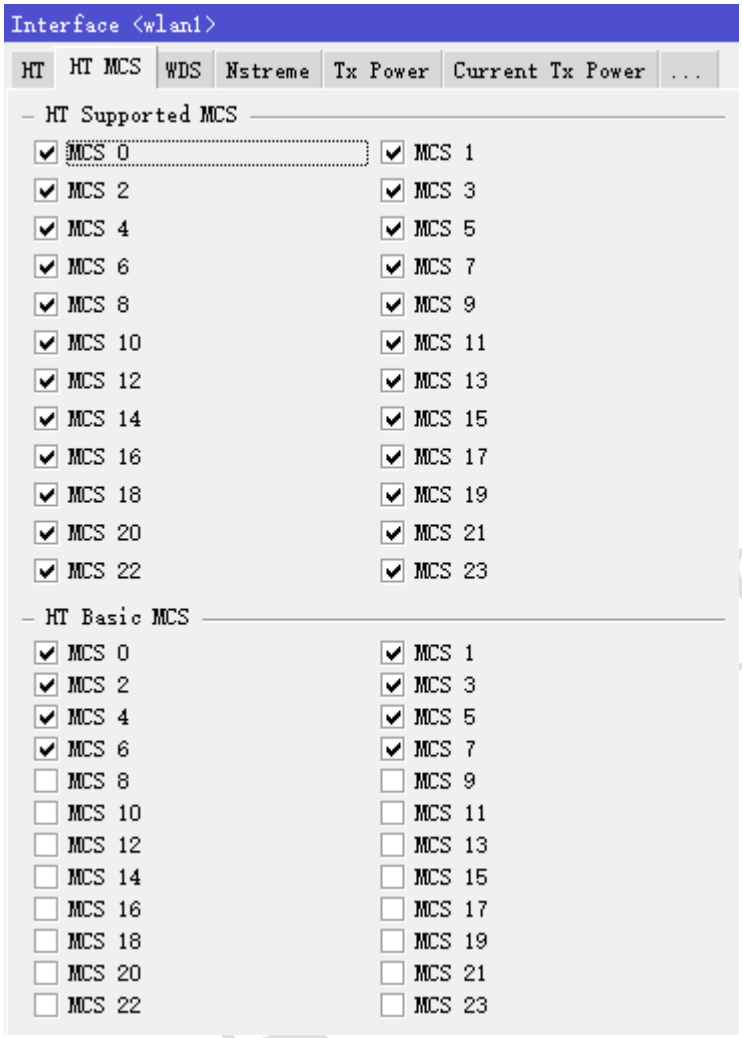
MCS 索引	空间流数量	调制方式	速率(Mb/s)	
			800ns GI	400ns GI
0	1	BPSK	13.5	15.0
1	1	QPSK	27.0	30.0
2	1	QPSK	40.5	45.0
3	1	16-QAM	54.0	60.0
4	1	16-QAM	81.0	90.0
5	1	64-QAM	108.0	120.0
6	1	64-QAM	121.5	135.0
7	1	64-QAM	135.0	150.0
8	2	BPSK	27.0	30.0
9	2	QPSK	54.0	60.0
10	2	QPSK	81.0	90.0
11	2	16-QAM	108.0	120.0
12	2	16-QAM	162.0	180.0
13	2	64-QAM	216.0	240.0
14	2	64-QAM	243.0	270.0
15	2	64-QAM	270.0	300.0

在 RouterOS 里 MCS 速率配置分为两类：**Basic-MCS** 和 **Support-MCS**：

Basic -MCS：基本 MCS 是指 AP 正常工作所必须支持的 MCS 速率集，终端设备必须满足 AP 所配置的基本 MCS 速率才能够与 AP 进行关联。

Support-MCS：支持 MCS 速率集是在满足 AP 的 basic-MCS 集的条件下，AP 所能够支持的更高的速率，你可以配置 support-MCS 速率选择更高的速率与 AP 进行关联。

在 RouterOS 可以看到如下配置选项：



当 **rate-set=configured** 时，即无线的速率手动调节，下面是每种协议可以选择的速率对应关系

802.11 协议	可选设置
2.4ghz-b	basic-b, supported-b
2.4ghz-b/g, 2.4ghz-onlyg	basic-b, supported-b, basic-a/g, supported-a/g
2.4ghz-onlyn 2.4ghz-b/g/n	basic-b, supported-b, basic-a/g, supported-a/g, ht-basic-mcs, ht-supported-mcs
2.4ghz-g/n	basic-a/g,supported-a/g,ht-basic-mcs,ht-supported-mcs
5ghz-a	basic-a/g,supported-a/g
5ghz-a/n, 5ghz-onlyn	basic-a/g,supported-a/g,ht-basic-mcs,ht-supported-mcs
5ghz-a/n/ac, 5ghz-onlyac	basic-a/g,supported-a/g,ht-basic-mcs,ht-supported-mcs,vht-basic-mcs,vht-supported-mcs

当你设置 **rate-set=configured** 时，无线速率不再由 RouterOS 自定义，而是手动调整，该操作需要谨慎，特别是在WiFi覆盖场景，特别是basic-rate参数非常关键，即AP对终端设备基本速率的要求，如果把basic-rate调整过高，会导致部分设备无法连接。

4.5 RouterOS 各种 station 模式

该模式在 wds-mode 关闭下使用

这个是标准模式，在 station 模式下不支持二层桥接，如果试图将无线网卡放入 bridge 将不能获得预期的结果，但从另一方面考虑如果二层桥接不是必须的解决方式，可以选择路由和 MPLS 交换，这个模式支持所有的 ROS 无线协议

station-wds

该模式在 wds-mode 开启下使用

这个模式仅能与 RouterOS AP 工作（一些 WRT 软件支持），为此需要通过协商连接，所以 AP 端需要为对应的 station 建立一个独立的 WDS 接口，这个接口能通过点对点连接 AP 和对应的，无论是从 AP 到 station、还是 station 到 AP 或者终端之间的转发，都会保留二层 MAC 地址

这个模式支持所有的 RouterOS 无线传输协议（非 RouterOS 设备不支持），当使用标准 802.11 协议这个模式使用 4 个地址帧格式，如果是 MikroTik 私有协议（Nstreme 或 Nv2）将采用内部方法

这个模式能安全的使用二层桥接，即 AP 创建独立的 WDS 接口能使用桥接防火墙和 RSTP 环路探测和回避等

station-pseudobridge

这个模式从无线连接观点看与标准 station 模式连接相同，不过有限的支持二层桥接

基于 MAC 地址转换的 IPv4 数据传输，当帧发向 AP 通过 MAC 地址列表替换 IPv4 数据的源 MAC 地址（为了能使用 3 个 MAC 地址的帧格式），反过来替换发现 station 的目标 MAC 地址，也同样能建立 VLAN 帧的封装（但不能支持 PPPoE 透传）

station-pseudobridge-clone

这个模式和 station-pseudobridge 模式相同，This mode is the same as station-pseudobridge mode，除了连接到 AP 使用“clone”MAC 地址外，

station-bridge

这个模式仅能工作在基于 RouterOS 的 AP，提供对 station 设备透明二层桥接的支持，RouterOS AP 接受当启用 bridge 模式时客户端采用 station-bridge 模式的连接，即这个模式支持 AP 使用 bridge 模式，客户端使用 station-bridge，这样两个 L3 的设备可以实现点对点的网桥传输（主要应用于 Nv2 传输），这个模式同样能实现安全的二层桥接，即 AP 创建独立的 WDS 接口能使用桥接防火墙和 RSTP 环路探测和回避

Station Roaming

Station Roaming 模式是在 RouterOS v6.35 开始支持，winbox 管理操作接口是在 v6.38.3 版本添加。该模式仅支持 802.11 无线协议，且配置为 station 模式下使用，当 RouterOS 无线客户端使用 802.11 无线协议连接到 AP ，并在指定的时钟周期执行背景扫描。当背景扫描找到一个较好的 AP 信号，会尝试漫游到该 AP。扫描时间间隔在无线信号变得很差时，会缩短扫描时间，但当无线客户端信号很好时，时间间隔会增长。

下面是配置实例

```
[admin@MikroTik] /interface wireless> set 0 mode=station-wds
station-roaming=enabled  wireless-protocol=802.11
```

测试操作丢包 1 个，切换标准无法知道，以下是根据 RouterOS 的 wireless debug 日志看到的情况，切换过程

```
16:05:07 wireless,debug wlan1: start background scan
16:05:10 wireless,debug wlan1: background scan complete, must select network
16:05:10 wireless,debug wlan1: no network that satisfies connect-list,  by
default
 choose with strongest signal
16:05:10 wireless,debug wlan1: found better AP E4:8D:8C:BD:14:D1
16:05:10 wireless,info E4:8D:8C:60:B6:CD@wlan1: lost connection, roaming
16:05:10 wireless,debug wlan1: connect to better AP E4:8D:8C:BD:14:D1
16:05:10 wireless,info E4:8D:8C:BD:14:D1@wlan1 established connection on
2462000, SSID mik1
```

4.6 Repeater 中继器

从 RouterOS 6.35 加入了 Repeater 无线中继功能，Wireless repeater 允许无线网卡从 AP 端接收信号，并能使用相同的无线网卡复制信号给其他客户端连接。Wireless repeater 通过配置无线网卡连接 AP 使用 station-bridge 或 station-pseudobridge 选项，并创建一个虚拟 AP 接口，然后创建一个 bridge 界面，并将两个界面（主网卡和虚拟网卡）添加到 bridge 中。

如果你的 AP 支持 **button-enabled WPS** 模式，你可以使用自动设置命令

```
/interface wireless setup-repeater wlan1
```

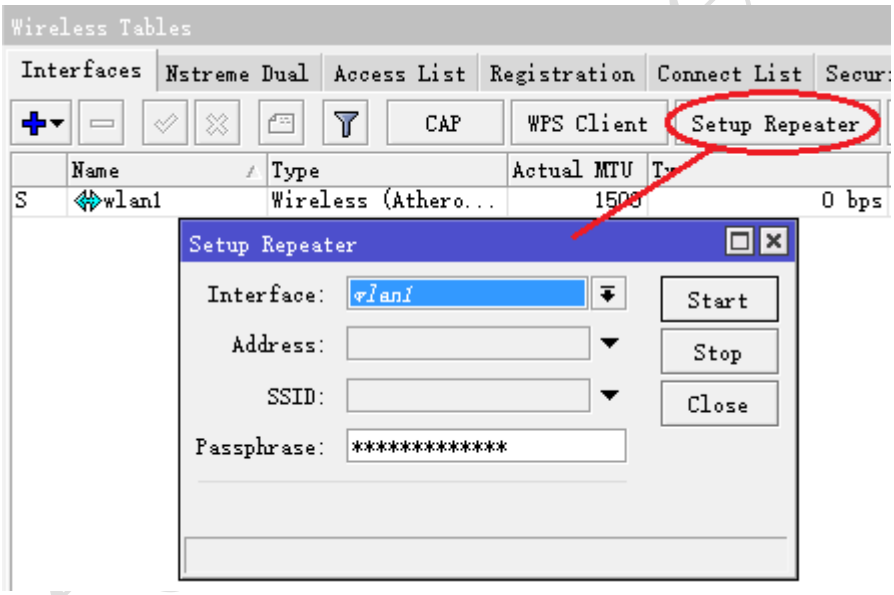
setup-repeater 命令有以下步骤

- 当按钮被按下后，搜索 WPS AP
- 从 AP 获取 SSID，key，频道
- 复位主无线网卡配置
- 创建新的 bridge 接口，并删除主无线网卡和虚拟网卡在其他 bridge 端口下的配置
- 删除所有添加到该主无线网卡的虚拟网卡
- 创建 security profile，取规则是 "<interfacename>-<ssid>-repeater"，如果有相同 security profile 存在，将无法创建新规则，只做配置更新设置

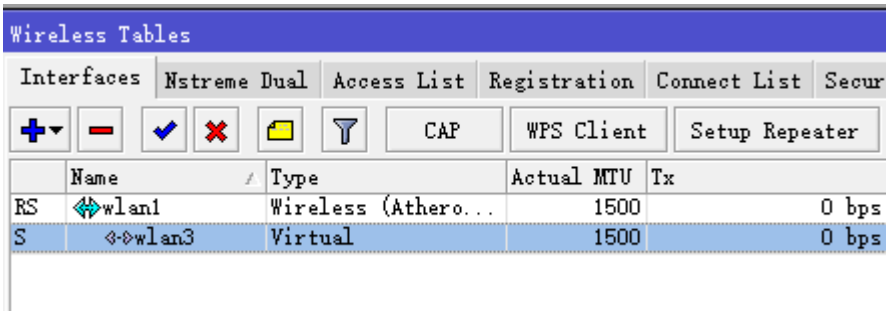
- 配置主无线网卡，接口模式选择方式：如果对端 AP 支持桥接模式，选择使用 station-bridge，如果 AP 支持 WDS 模式，选择使用 station-wds，如果 AP 之前两种模式都不是则选择使用 station-pseudobridge
- 创建虚拟 AP 接口，并使用相同的 SSID 和 security profile
- 如果主无线网卡没有在其他 bridge 里，会创建一个新的 bridge 接口，并将主无线网卡添加进去
- 将创建的虚拟 AP 接口，添加到与主无线网卡同一个 bridge 下
- 如果你的 AP 不支持 WPS，只能通过手动设置，使用以下参数
- address – AP 的 MAC 地址
- ssid – 中继器连接 AP 的 SSID
- passphrase – AP 的验证密码-如果对应 AP 的密码指定，将搜索 AP，并在/interface wireless security-profile 创建指定密码的安全策略。如果密码未指定，将通过 WPS 找到密码

setup-repeater 手动配置过程，可以通过搜索查找当前周围的 AP，至少需要在除 interface 选项外，设置一个参数，即对端 AP 的连接密码。

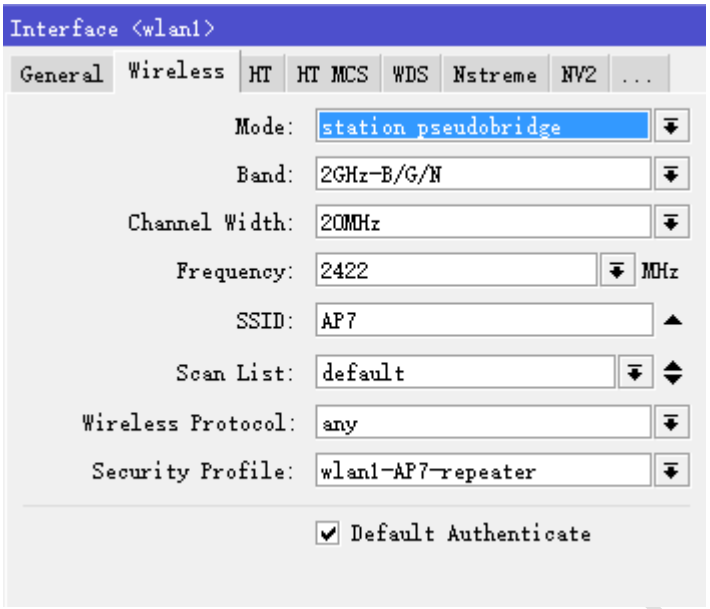
根据中继器配置要求填写入连接 AP 的密码信息填



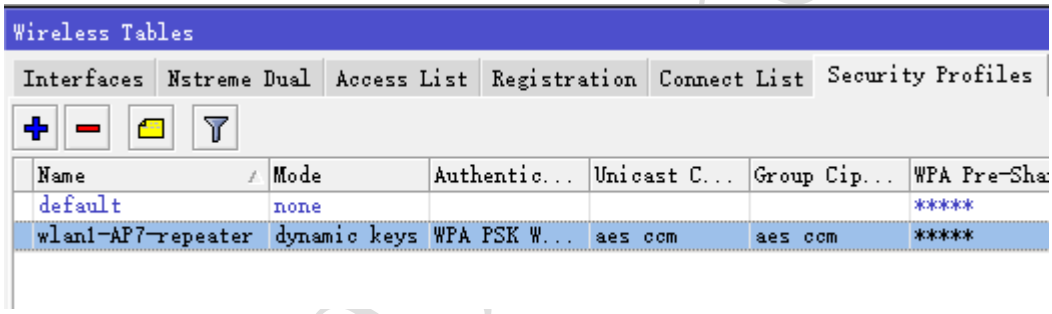
配置完成后，点击 start，开始搜索 AP，同时 RouterOS 会复位当前无线网卡的配置，并尝试连接对应 AP，连接成功后，会自动创建一个虚拟 AP，如下图创建了 wlan3 的虚拟 AP，并从属于 wlan1 物理无线网卡



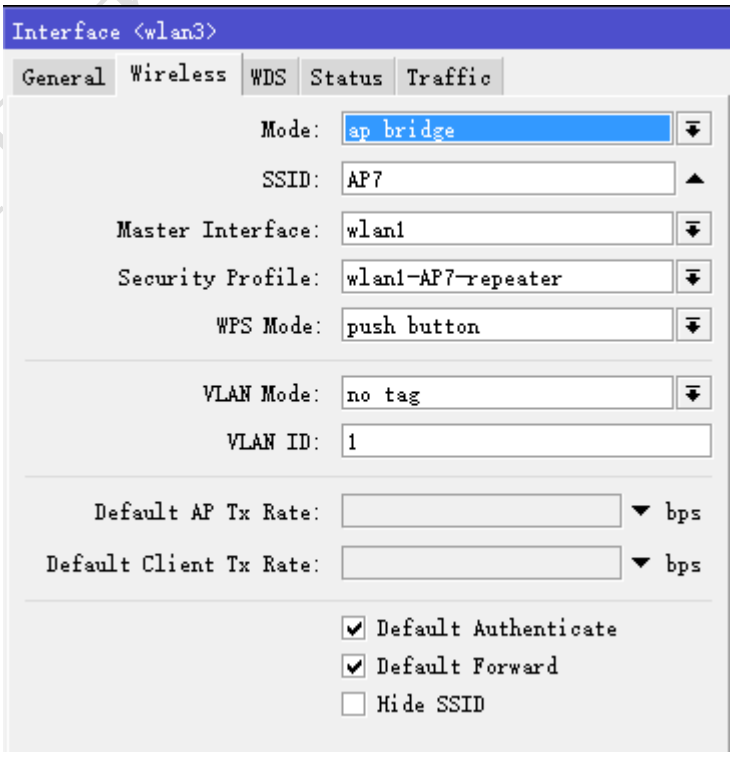
从下面物理网卡的配置可以看到，RouterOS 在复位网卡后，重新配置 mode=pseudobridge，并添加了安全策略为 wlan1-AP7-repeater



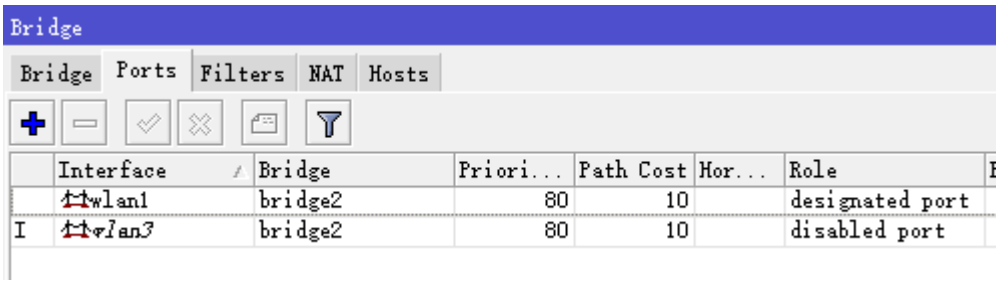
我们可以在 security-profiles 查看到新增的安全策略



虚拟无线网卡 wlan3 配置，模式为 ap-bridge，SSID 为相同的 AP7，安全策略为：wlan1-AP7-repeater



然后我们在 bridge 可以查看到 RouterOS 会自动为中继无线添加两个接口的 bridge 桥接配置，wlan1 和 wlan3 添加到 bridge2 接口下，完成桥接



4.7 RouterOS Wireless 基本参数介绍

属性	描述
adaptive-noise-immunity (*ap-and-client-mode* | *client-mode* | *none*; 默认: none)	自适应噪音免疫，该属性仅基于 Atheros 厂商芯片，且是 AR5212 芯片或更高网卡芯片能生效
antenna-gain (整型 *[0..4294967295]*; 默认: 0)	天线增益，单位 dBi, 被用于计算最大传输功率，主要根据各个国家对发生功率要求限制决定
antenna-mode (*ant-a* | *ant-b* | *rxa-txb* | *txa-rxb*; 默认: *ant-a*)	设置天线使用船速和接收方式，仅 802.11abg 可以选 *ant-a* - 仅用'a'天线接口（网卡 main） *ant-b* -仅用'b'天线接口（网卡 aux） *txa-rxb* - 使用'a'天线发射，'b'天线接收 *rxa-txb* -使用'a'天线接收，'b'天线发射
area (*字符*; 默认:)	识别无线网络工作组，该值通过 AP 宣布，并匹配其他 AP 的 connect-list 下的 area-prefix。这个是一个专属区域扩展
arp (*disabled* | *enabled* | *proxy-arp* | *reply-only*; 默认: enabled)	请参考 ARP 地址解析协议
band (*2ghz-b* | *2ghz-b/g* | *2ghz-b/g/n* | *2ghz-onlyg* | *2ghz-onlyn* | *5ghz-a* | *5ghz-a/n* | *5ghz-onlyn*; 默认:)	定义无线频段和带宽速率
basic-rates-a/g (*12Mbps* | *18Mbps* | *24Mbps* | *36Mbps* | *48Mbps* | *54Mbps* | *6Mbps* | *9Mbps*; 默认: 6Mbps)	定义 a/g 带宽速率
basic-rates-b (*11Mbps* | *1Mbps* | *2Mbps* | *5.5Mbps*; 默认: 1Mbps)	定义 b 带宽速率
bridge-mode (*disabled* | *enabled*; 默认: enabled)	仅 AP-bridge 和 bridge 模式可选
burst-time (*整型*| *disabled*; 默认:)	
channel-width (*10mhz* | *20/40mhz-ht-above* | *20/40mhz-ht-below* | *20mhz* | *40mhz-turbo* | *5mhz*; 默认: 20mhz)	允许 ht 使用高于和低于 20MHz 扩展频率。扩展频率允许 11n 设备使用 40MHz 的频谱从而增加最大吞吐量

comment (*字符*; 默认:)	界面注释描述
compression (*yes* | *no*; 默认 no)	设置属性为 yes，即允许硬件压缩。无线网卡必须支持硬件压缩功能。连接设备没有使用压缩仍然能工作。
country (*name of the country* | *no_country_set*; 默认: no_country_set)	各个国家可以获得的频段范围、每个频率最大发射功率。也可以自定义 scan-list（用于其他频段或超级频段搜索）. 值选择为 *no_country_set* 是遵循 FCC 频道设置。
default-ap-tx-limit (整型 *[0..4294967295]*; 默认: 0)	ap-tx-limit 值对 client 的带宽速率起作用（仅支持 MikroTik 设备），将不会匹配 access-list 条目
default-authentication (*yes* | *no*; 默认: yes)	AP 模式可以选，该值是否验证客户端，如果设置为 yes，将不会匹配任何 access-list 列表的条目，设置为 no 将匹配 access-list 条目。对于 station mode 连接到 AP，该值设置为 yes 将不会匹配 connect-list 的任何条目。
default-client-tx-limit (*整型 [0..4294967295]*; 默认: 0)	Client-tx-limit 值对 client 的带宽速率起作用（仅支持 MikroTik 设备），将不会匹配 access-list 条目
default-forwarding (*yes* | *no*; 默认: yes)	该值设置为 yes 默认转发客户端数据，将不会匹配 access-list 下任何条目。如果设置为 no 将匹配 access-list 条目
dfs-mode (*no-radar-detect* | *none* | *radar-detec*; 默认: none)	DFS (Dynamic Frequency Selection)，动态频率选择技术，用于探测网络内的频率使用情况，动态选择频率 *none* - 禁用 DFS。 *no-radar-detect* - 从 scan-list 选择指定范围探测频率。'wds-slave'模式该设置不能生效。 *radar-detect* - 选择最低可用的网络频率，并选择在 60 秒内没有探测到的频率。另外，选择不同频率，这个设置要求国家的注册通过 该属性仅支持 AP 模式
disable-running-check (*yes* | *no*; 默认: no)	当设置为 yes，网卡将总是运行标记
disabled (*yes* | *no*; 默认: yes)	是否禁用无线网卡
disconnect-timeout (*time [0s..15s]*; 默认: 3s)	在帧在最低的数据率下发送已失败，被用于探测发送 3 次失败信号的间隔周期，在此时“3 * (hw-retries + 1)”。在整个 on-fail-retry-time 周期内 disconnect-timeout 数据报发送将被重新发。如果没有帧在整个 diconnect-timeout 周期内没有成功，无线连接关闭，并在 log 事件中记录 "extensive data loss"大量数据丢失。当帧成功发送该定时器会重设。
distance (*integer* | *dynamic* | *indoors*; 默认: dynamic)	连接距离，通常计算 ack 值，即多长时间等待确认单播帧。dynamic 值控制 AP 探测，并使用最小 ack 时间连接客户端,ack 不能被用于 Nsteme 协议。
frame-lifetime (*integer [0..4294967295]*; 默认: 0)	当帧发送时间长度超过了 frame-lifetime 丢弃该帧，默认值为 0，帧被丢弃只会在连接关闭后。
frequency (*整型 [0..4294967295]*; 默认:)	频率以 MHz 单位工作

	频率选择依赖你选择的 band，且与你选择的 country（国家）和无线网卡功能参数有关 这个设置不能工作在任何 station 模式、wds-slave 模式和 DFS 启用状态下。
frequency-mode (*manual-txpower* | *regulatory-domain* | *superchannel*; 默认: manual-txpower)	这里可以获得三种频率模式： *regulatory-domain* - 限制可获得的频率和功率，都根据你选择各个 country 国家的要求 *manual-txpower* - 如同上，但没有限制最大功率 *superchannel* - 性能测试模式（Conformance Testing Mode）. 允许支持无线网卡支持的所有频率。 列出所有频段可以获得的频率通过命令/wireless info print。superchannel 这个模式应该被用在有限范围，或者你有特殊的许可在特定的地区。在 v4.3 前被称为 Custom Frequency Upgrade 或 Superchannel，需要申请注册，v4.3 后不需要申请 key 升级该功能
frequency-offset (*整型 [-2147483648..2147483647]*; 默认: 0)	如果无线网卡工作在不同的频率，RouterOS 会显示，并允许指定偏移值。频率变频器被集成在无线网卡内部。例如你的网卡工作在 4000MHz，但 RouterOS 显示为 5000MHz，设置偏移量为 1000MHz，这样 RouterOS 会显示正确的值。这个值以 MHz 正负为单位。
hide-ssid (*yes* | *no*; 默认: no)	*yes* - AP 不会包含 SSID 在信号标示帧内，并不会响应广播 SSID 的请求。 *no* - AP 将包含 SSID 在信号标示帧内，并回应广播 SSID 的请求。 这个属性仅适用于 AP 模式，设置该参数为 yes，客户端软件将不会显示该网络名称。修改这个参数别并不会提升无线网络性能，因为 SSID 会被包含在其他帧里。
ht-ampdu-priorities (整型列表 *[0..7]*; 默认: 0)	AMPDU 优先发送的帧会得到协商和使用(汇总帧，并通告块发送确认)，使用 AMPDU 会增加吞吐量，但可能会增加延迟，因此可能会在实时传输上不那么令人满意（语音和视频）。因此选择默认的 AMPDU 可以获得较好速率
ht-amsdu-limit (*整型 [0..8192]*; 默认: 8192)	当协商时设备被允许默认值最大 AMSDU。AMSDU 聚合能明显增加小帧的吞吐量，但可能在丢失聚合帧的情况下增加传输延迟。发送和接收 AMSDU 都会增加 CPU 负载。
ht-amsdu-threshold (*整型 [0..8192]*; 默认: 8192)	最大帧长度允许包含进 AMSDU。
ht-basic-mcs (*列表 (mcs-0 | mcs-1 | mcs-2 | mcs-3 | mcs-4 | mcs-5 | mcs-6 | mcs-7 | mcs-8 | mcs-9 | mcs-10 | mcs-11 | mcs-12 | mcs-13 | mcs-14 | mcs-15 | mcs-16 | mcs-17 | mcs-18 | mcs-19 | mcs-20 | mcs-21 | mcs-22 | mcs-23)*; 默认: mcs-0; mcs-1; mcs-2; mcs-3; mcs-4; mcs-5; mcs-6; mcs-7)	调制与编码，这个是每个连接客户端都必须支持（请参考 802.11n MCS 规范）

ht-guard-interval (*any* | *long*; 默认: any)	是否允许使用监视间隔 (参考 802.11n MCS 规范可以看到如何影响吞吐量)。"any" 将会使用短距离或长距离之间，这个依赖于数据传输率"long" 将使用长距离。
ht-rxchains (整型列表*[0..2]*; 默认: 0)	那一个天线用于接收数据
ht-supported-mcs (*列表(mcs-0* | *mcs-1* | *mcs-2* | *mcs-3* | *mcs-4* | *mcs-5* | *mcs-6* | *mcs-7* | *mcs-8* | *mcs-9* | *mcs-10* | *mcs-11* | *mcs-12* | *mcs-13* | *mcs-14* | *mcs-15* | *mcs-16* | *mcs-17* | *mcs-18* | *mcs-19* | *mcs-20* | *mcs-21* | *mcs-22* | *mcs-23)*; Default: mcs-0; mcs-1; mcs-2; mcs-3; mcs-4; mcs-5; mcs-6; mcs-7; mcs-8; mcs-9; mcs-10; mcs-11; mcs-12; mcs-13; mcs-14; mcs-15; mcs-16; mcs-17; mcs-18; mcs-19; mcs-20; mcs-21; mcs-22; mcs-23)	设备广播所支持的调制和编码方式。
ht-txchains (整型列表*[0..2]*; 默认: 0)	那一个天线用于传输数据
hw-fragmentation-threshold (*整型[256..3000]* | *disabled*; 默认: 0)	当传输基于无线介质时采用 byte 为单位，指定最大分片数据报长度。802.11 标准数据报分片允许在无线传输前将数据报分片传输从而增强传输成功率(只有在分片数据报没有被正确重新发送传输). 注意分片数据报的传输比传输非分片数据报更有效，因为协议开销并在两端增加资源耗用（传输和接收）
hw-protection-mode (*cts-to-self* | *none* | *rts-cts*; 默认: none)	帧保护属性，参考帧保护介绍
hw-protection-threshold (*integer [0..65535]*; 默认: 0)	帧保护属性，参考帧保护介绍
hw-retries (*整型 [0..15]*; 默认: 7)	在不考虑一个传输失败情况下，重新发送帧的次数。当一个传输失败帧被重新发送的次数。数据率降低接近掉线，帧会再次发送。在 on-fail-retry-time 的时间内，支持最低速率传输到目的地，出现连续三次失败，在失败后，帧会继续重发，直到重新传输成功，或者直到客户端在 disconnect-timeout 值后断开连接。如果 frame-lifetime 值到期，帧也会被丢弃掉
l2mtu (*整型 [0..65536]*; 默认: 2290)	
mac-address (*MAC*; 默认:)	无线网卡 MAC 地址
master-interface (*字符*; 默认:)	当启用 virtual-ap（虚拟 AP）时，选择的无线网卡，virtual-ap 只能工作在 master 网卡设置为 ap-bridge、bridge 和 wds-salve 模式下。该参数仅支持虚拟 AP。
max-station-count (*整型 [1..2007]*; 默认: 2007)	最大关联的客户端数量，AP 之间的 WDS 连接也包括在内，该属性可以控制连接客户端数量，限制连接客户端或者避免 AP 连接过于饱和。
mode (*station* | *station-wds* | *ap-bridge* | *bridge* | *alignment-only* | *nstreme-dual-slave* | *wds-slave*	选择设备的工作模式，station 或者 AP 等 Station 模式:

| *station-pseudobridge* | *station-pseudobridge-clone* | *station-bridge*; 默认: station)	*station* - 基本的 station 模式，寻找并连接到可用的 AP。（不支持桥接功能） *station-wds* - 类似 station 模式，但但能与 AP 创建 WDS 连接，AP 端必须配置启用 WDS 模式才能与 station-wds 连接。 *station-pseudobridge* - 类似 station 模式，但会加上 MAC 地址翻译进行传输，允许网卡做桥接（但桥接非完整意义的透传，因为使用了 MAC 地址翻译） *station-pseudobridge-clone* - 类似 *station-pseudobridge*，但使用了 station-bridge-clone-mac 地址连接到 AP。 AP 模式: *ap-bridge* - 基本的访问节点模式，支持各种类型的连接。 *bridge* - 类似 *ap-bridge*,但被限制连接一个客户端 。 *wds-slave* - 类似 *ap-bridge*，但要与 AP 相同的 SSID，且要启用 WDS 连接。如果连接丢失或者无法建立，这是将会继续搜索相同 SSID 的 AP。wds-slave 如同 AP，但不主动宣告自己 SSID。如果 dfs-mode 选择 *radar-detect*，这时 AP 将启用 hide-ssid ,将在雷达探测周期里无法搜索 特殊模式: *alignment-only* - 将无线网卡放入一个连续的传输模式，被用于校准远程天线 *nstreme-dual-slave* - 允许这个无线网卡选择 nstreme-dual 设置。 MAC 地址翻译在 pseudobridge 模式下，通过检查和建立相关 IP 和 MAC 地址的对应关系表，所有数据报发送到 pseudobridge,并将收到的 MAC 地址存储到地址翻译列表中，通过 MAC 地址翻译的形式转发数据，因此超过一个主机的桥接网络不能建立二层的协议，如透传 PPPOE 拨号；注意：当前 pseudobridge 不支持 IPv6。 Virtual AP 不具备这个模式属性，他们遵从 master 网卡的模式。
mtu (*整型 [0..65536]*; 默认: 1500)	最大传输单元
multicast-helper (*default* | *disabled* | *full*; 默认: **default**)	当设置为 full 参数时，将发送单播目标 MAC 地址，解决在无线传输的组播问题。该选项仅支持在 AP 端设置，而客户端需要配置 **station-bridge** 模式。该功能从 v5.15 开始启用 • disabled – 禁用 helper 功能，通过组播 MAC 地址发送组播包 • full – 所有组播包 MAC 地址都转换为单播 MAC 地址优先发送 • default – 默认选择当前的 disabled 设置。
name (*字符*; 默认:)	无线网卡名称
noise-floor-threshold (*default* | *integer*	这个属性只能在 AR5211 芯片上生效

[-128..127]; 默认: default)	
nv2-cell-radius (*整型 [10..200]*; 默认: 30)	设置冲突时间槽的影响长度，AP 分配给客户端的初始连接，同样也是评估到客户端的距离，当设置过小，远程的客户端将会出现连接问题或者掉线（ranging timeout 错误）。这个设置虽然整个运行周期内影响可以被忽略，但为了获得最大性能，在没有必要的情况下请不要增加该设置，因此 AP 不会预留时间，实际上不会被使用，而是分配它实际数据传输。 对于 AP: 远处客户端距离，单位 km 对于 station：无效
nv2-noise-floor-offset (*default* | *整型 [0..20]*; 默认: default)	
nv2-preshared-key (*字符*; Default:)	
nv2-qos (*default* | *frame-priority*; 默认: default)	设置数据报优先级机制，首先数据从优先级高的队列发送，这时低队列优先级数据要等到 0 队列的优先级到达。当高优先级的连接满时，低优先级数据不会被发送。在 AP 上使用该参数请谨慎。 frame-priority - 手的设置能被 mangle 规则调用。 default - 默认设置为小包提供最低延迟的优先级
nv2-queue-count (*integer [2..8]*; 默认: 2)	
nv2-security (*disabled* | *enabled*; 默认: disabled)	Nv2A 安全设置
on-fail-retry-time (*time [100ms..1s]*; 默认: 100ms)	极低的速率下，在第三次发生失败后，等待指定间隔重试时间。
periodic-calibration (*default* | *disabled* | *enabled*; 默认: default)	如果 default-periodic-calibration 属性被启用，则可以设置默认启用该周期校正，这个属性类型依赖无线网卡的类型，这个属性仅能用于 Atheros 芯片。
periodic-calibration-interval (*integer [1..10000]*; 默认: 60)	这个属性仅能用于 Atheros 芯片的网卡
preamble-mode (*both* | *long* | *short*; 默认: both)	短报文模式是 802.11b 标准的选项，能减低每个帧的开销 对于 AP: *long* - 不使用短报文 *short* - 宣布短报文功能，不接受来至没有这个功能客户端的链接 *both* - 对双方宣布短报文功能 对于 station: *long* - 不使用短报文功能 *short* - 如果 AP 不支持短报文，将不连接 *both* - 如果 AP 支持启用短报文
prism-cardtype (*100mW* | *200mW* | *30mW*; 默认:)	指定安装的 prism 网卡的类型（现在 prism 网卡几乎很少见到）
proprietary-extension (*post-2.9.25* | *pre-2.9.25*;	管理帧中的一个信息单元包含了 RouterOS 的私有信息。

Default: post-2.9.25)	这个参数控制如下信息： *pre-2.9.25* - 这个代表旧的版本 2.9.25 前，能与高版本的 RouterOS 交互信息，该模式与一些客户端不兼容，例如 intel 迅驰（Centrino）客户端。 *post-2.9.25* - 使用标准方式，将兼容更新的无线客户端
radio-name (*字符*; 默认: MAC 地址)	设备的名称描述，将在无线登记表显示（registration table）远程设备的 MAC 地址。
rate-selection (*advanced* | *legacy*; 默认: legacy)	
rate-set (*configured* | *default*; 默认: default)	可以获得两个选项： *default* - 使用默认所支持的设置，basic-rates 和 supported-rates 下的参数会被自动锁定 *configured* - 使用 use values from basic-rates 和 supported-rates 下的参数，手动调制，注意 g 模式下，同时使用"rates-b"和"rates-a/g"属性.
scan-list (可以通过逗号分隔频率或者"-"定义频率范围| *default*; 默认: default)	*default* 值指根据无线网卡支持，且频率模式（frequency-mode）、当前国家（country）频率规范等设置可以获得的频率范围(可以通过 info 查看"/interface wireless info> pirnt"). 默认搜索列表 5ghz 频段下，每间隔 20MHz 步进搜索，在 *5ghz-turbo* 频段下每间隔 40MHz，2.4G 则间隔 5MHz。如果 scan-list 采用手动设定，所有指定的频率都会被搜索（例如：scan-list=*default,5200-5245,2412-2427* 即会使用默认频段搜索，并添加从 5200-5245 或 2412-2427 频率范围.）
security-profile (*字符*; 默认: default)	从 security-profiles 获取加密策略
ssid (*字符 (0~32 字符)*; 默认: 根据 RouterOS 的/system identity 取值)	SSID (Service Set Identifier)服务集标识符的缩写,它是用来区分一个无线网络接入点与另一个接入点的标识符
station-bridge-clone-mac (*MAC*; 默认:)	这个属性仅在 *station-pseudobridge-clone* 模式下生效 通过使用指定的 MAC 地址连接到 AP，如果这个值为 *00:00:00:00:00:00*, station 将会首先使用无线网卡的 MAC 地址 当在 station 内部的设备需要传输数据到 AP 端，这些设备的 MAC 地址将会被替换成该指定的 MAC 地址发送
supported-rates-a/g (速率列表 *[12Mbps | 18Mbps | 24Mbps | 36Mbps | 48Mbps | 54Mbps | 6Mbps | 9Mbps]*; Default: 6Mbps; 9Mbps; 12Mbps; 18Mbps; 24Mbps; 36Mbps; 48Mbps; 54Mbps)	被支持的速率列表，被用于除 *2ghz-b* 的频段
supported-rates-b (*list of rates [11Mbps | 1Mbps | 2Mbps | 5.5Mbps]*; Default: 1Mbps; 2Mbps; 5.5Mbps; 11Mbps)	支持的速率列表，被用于 *2ghz-b*, *2ghz-b/g* 和 *2ghz-b/g/n* 频段 bands. 两个设备连接使用的速率要求同时被设备所支持。这个属性仅在 rate-set 被选择下生效
tdma-debug (*整型 [0..4294967295]*; 默认: 0)	TDMA 调试

tdma-hw-test-mode (*整型 [0..4294967295]*; 默认:)	
tdma-override-rate (*12mbps* | *18mbps* | *24mbps* | *36mbps* | *48mbps* | *54mbps* | *6mbps* | *9mbps* | *disabled* | *ht20-mcs...* | *ht40-mcs...*; 默认: disabled)	
tdma-override-size (*整型 [0..4294967295]*; 默认:)	
tdma-period-size (*整型 [1..10]*; 默认: 2)	指定 TDMA 以毫秒为周期，能帮助长距离的传输，有助于增加带宽，但同时也会增加延迟
tdma-test-mode (*整型 [0..4294967295]*; 默认: 0)	
tx-power (; Default:)	无线网卡发射功率设置
update-stats-interval (; 默认:)	多长时间要求客户端更新信号强度和 CCQ 值 打开 registration-table 同样可以更新，这个属于 RouterOS 扩展属性
wds-cost-range (*disabled* | *time [10s..5h]*; 默认: disabled)	桥接（Bridge）的 WDS 连接成本开销（Port cost）自动调整，该值通过测量链路的吞吐量。如果超过 10%的变动，或者超过 20 秒没有做调整，Port cost 每 5 秒被重新计算和调整 如果设置该参数为 0，即禁用自动成本调整 自动调整不能工作在手动配置的一个 bridge 端口下，即需要指定 wds-default-bridge
wds-default-bridge (*字符* | *none*; 默认: none)	当 WDS 连接已建立，并且 WDS 接口进入运行状态 (*running*),即可以将你预先设定的 bridge 接口添加到这个属性中。当 WDS 连接断开，会自动从 bridge 中将 WDS 接口删除。如果这个 WDS 接口被手动添加到 bridge 中，将不会再次添加
wds-default-cost (*整型 [0..4294967295]*; 默认: 100)	初始化 bridge 接口的 WDS 连接成本（cost），注：有线以太网预设为 10，无线默认为 100
wds-ignore-ssid (*yes* | *no*; 默认: no)	默认为 no，两个 AP 能创建在相同的频率下，且都拥有相同的 SSID。如果这个属性被设置为 yes，这时 SSID 将不会监测远程的 SSID，这个属性不会在客户端为 station-wds 模式下生效，同样不能工作在 wds-mode 为 *static-mesh* 或者 *dynamic-mesh*.
wds-mode (*disabled* | *dynamic* | *dynamic-mesh* | *static* | *static-mesh*; 默认: disabled)	控制 WDS 如何连接其他设备 (AP 和客户端的 station-wds 模式)。 *disabled* - 不启用 WDS 连接 *static* - 仅允许 WDS 连接通过手动配置，即需要添加 WDS 接口绑定对方 MAC 地址 *dynamic* - 允许 WDS 连接，不用手动增加 WDS 接口，自动建立连接，并且当连接断开后会自动从无线网络列表

	中删除 *-mesh* 模式，使用更好的方式在 AP 间建立连接，不兼容非 mesh 的 AP 模式，这个方式是为避免单边的 WDS 连接创建在两个 AP 中的一个。 当 AP 或者 station 与其他 AP 建立 WDS 连接，通过使用 connect-list 检查这个连接是否被允许；如果 station 通过 station-wds 模式连接到 AP，AP 可以通过 access-list 这个 station 是否被允许加入
wmm-support (*disabled* | *enabled* | *required*; 默认: disabled)	指定是否启用 WMM

第五章 RouterOS WiFi 覆盖配置

5.1 WiFi 覆盖介绍

我们通常说的 WiFi 覆盖指的是通过 AP 为终端使用者，如笔记本电脑、安装无线网卡的台式 PC、支持 WiFi 的手机、Pad 等终端设备。即这些终端设备接收到 AP 信号后，不再进行数据转发，数据在此处理和终结。

WiFi 的覆盖我们一般区分为室内和室外覆盖，我们在之前的 RouterOS WiFi 分析提到 RouterOS 802.11 协议中优势主要是 WLAN 设备间的传输，而对于终端使用者的覆盖相对较弱，并不代表 RouterOS 覆盖方面完败与其他任何设备，只是相对于纯 AP 设备覆盖抗干扰性和优化上来说没有特别明显的优势，但在各种功能上有无法比拟的优势。

我们这里讲解的是 RouterOS 的基本 WLAN 覆盖，这里我称为 WiFi 覆盖，比较贴近流行的终端覆盖应用，我们把 WiFi 覆盖分为室内和室外，室内覆盖和室外覆盖区别在于环境结构上的区别，室内覆盖和室外覆盖都需要进行具体的分析!在我们之前的 RouterOS 分析中我已经提到关于 RouterOS 在覆盖方面具体问题。

室内覆盖

在室内覆盖上，结构比较覆盖，我们选择室内的中间区域进行覆盖，在室内安放多个 AP 时，考虑 AP 覆盖的均匀性。室内覆盖对 RouterOS 选择无线网卡特别重要，即无线网卡的发射功率和接收灵敏度。我建议选择 200-350mw 无线网卡，接收灵敏度的要求就特别考验无线网卡了，很多无线网卡虽然发射功率很强，但接收灵敏就不是那么如人意！发射功率代表你说话，接收灵敏度就代表你听力！即使你声音很大所有人都能听到，但别人听到后回复你，你却听不到，那这个通信交流也是失败的。

室外覆盖

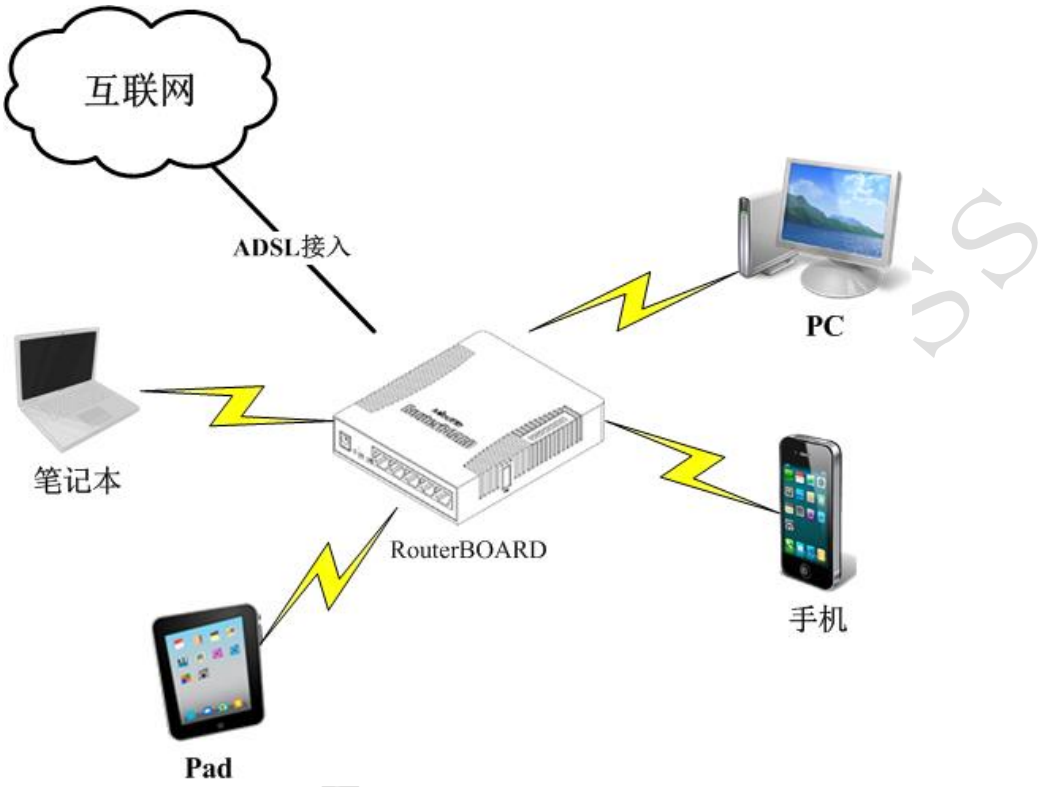
室内覆盖选择天线多为全向天线，增益从 6-12dBi，我们选择天线肯定希望发射增益越大越好，因为我们知道增益可以增加信号在某一方向的强度，如果是 1-2 个 AP 到可以选择较强的信号，但多了就要考虑 AP 之间的相互干扰性，毕竟 AP 众多就向之前提到的噪音相互干扰。

WiFi 的覆盖一般有路由和桥接模式，大多厂商的 AP 设备预设采用桥接，但随着市场和客户需求的多样化，AP 的功能也在增加。对于 RouterOS 来说你不用担心他的功能，因为在某种程度上讲 RouterOS 是“全能”的，这个可能初学者是最有体会的，你拿家用的 TP-LINK 无线路由器配置接口和 RouterOS 的 winbox 接口一对比，吓你一跳！因为 RouterOS 集成了各种网络常见功能。如果你对 OSI 七层模型或者 TCP/IP 协议不太了解的话，可能理解路由和桥接模式比较困难，那建议你去百度或者 google 下，因为这对于无线网络应用非常重要，也是网络入门的基础，我建议你仔细掌握数据链路层、网络层和传输层，什么是交换机、网桥；什么是路由器等，这里我就不再多讲，你可以去参考下我的 RouterOS 网络教程，简单讲诉了基本的网络知识，毕竟这里我们的重点是在讲 RouterOS 的 WiFi 配置。

5.2 RouterOS WiFi 覆盖事例

普通 WiFi 上网

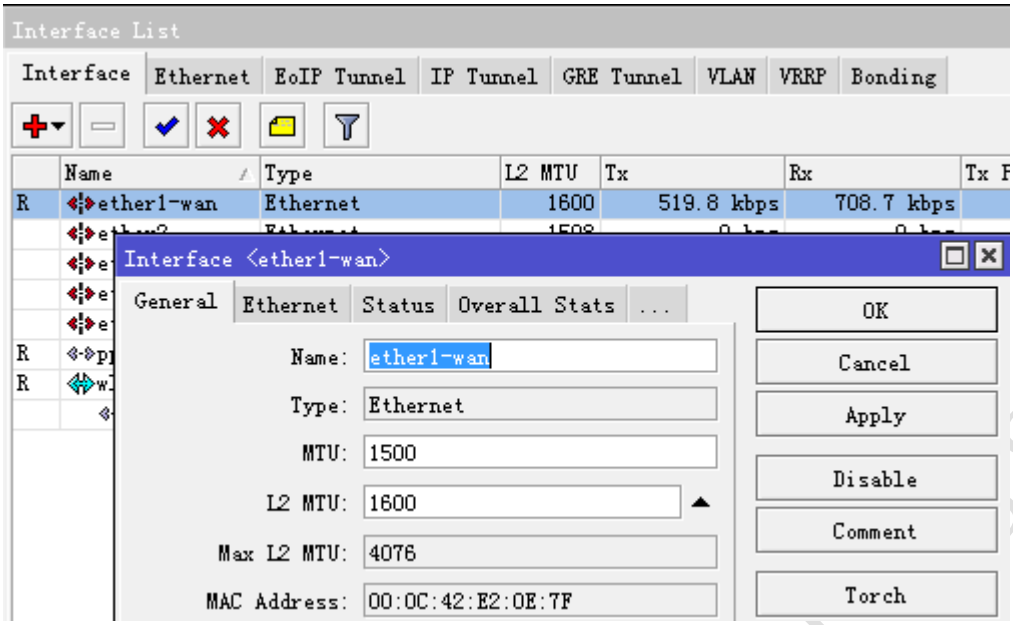
RouterOS 无线覆盖我们先从普通无线路由器最常见的路由方式说起，这个方式配置步骤和家庭的无线路由器一样，但不同的是 RouterOS 配置过程和接口相对复杂，例如：我们办公室内通过运营商光猫连接，通过 PPPoE 拨号上网，RouterOS 需要至少一个以太网口，然后通过一个 802.11bgn 的无线网卡覆盖办公室，如下图：



普通的步骤是

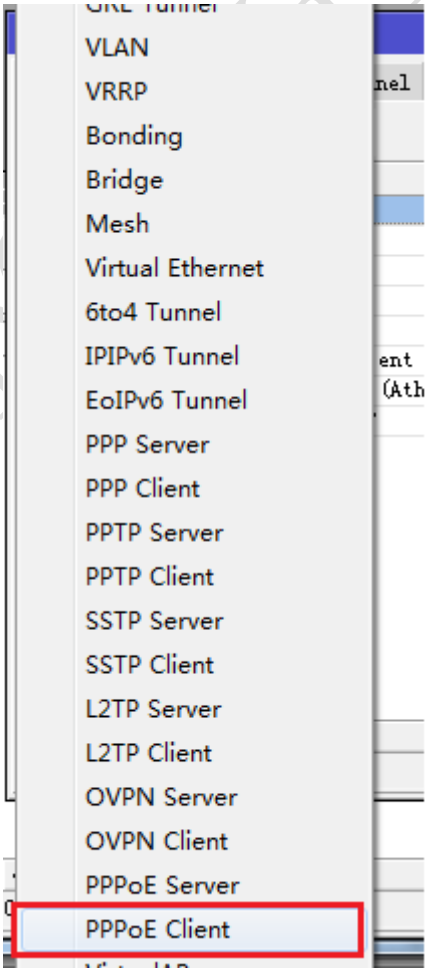
1、 配置 ADSL 拨号的 PPPoE
2、 配置我们 RouterOS 上的 wlan1 无线网卡采用 802.11bgn 参数
3、 配置 wlan1 无线网卡的 IP 地址为 192.168.100.1/24
4、 配置 DHCP 服务器和启用 DNS 缓存
5、 配置 ip firewall nat 的地址隐藏

那么我们首先来配置 RouterOS 的 PPPoE 拨号，我们登录 winbox 进入 interface 界面

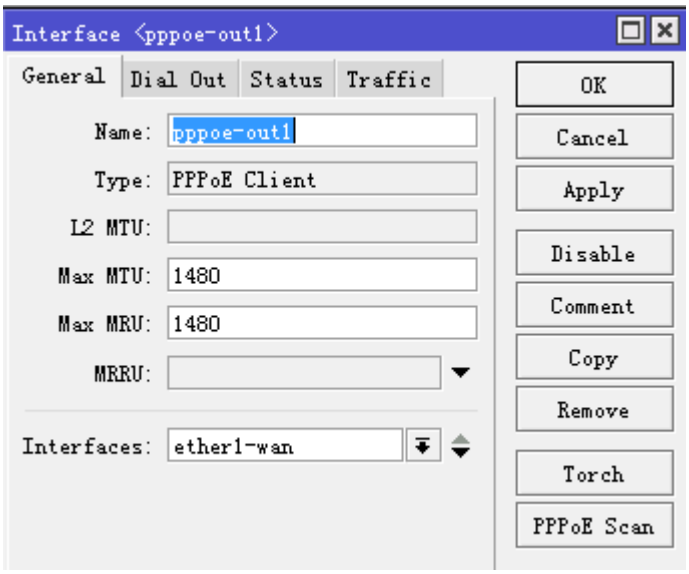


我习惯选择 ether1 配置为外网接入网口，并取名为 ether1-wan

然后添加点击加号添加 pppoe-client



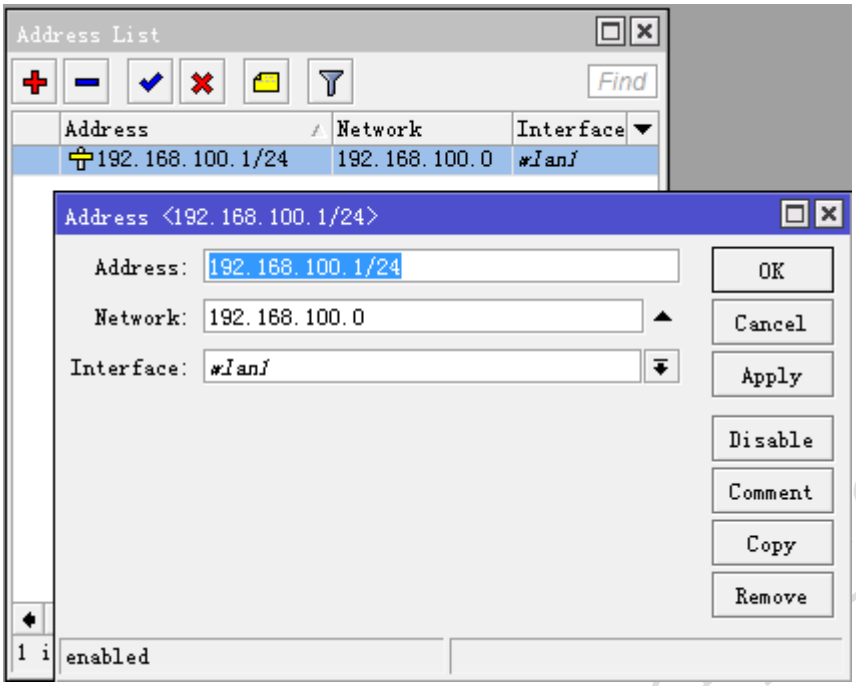
进入 pppoe-out1 的配置接口，选择 interface=ether1-wan，其他参数默认



选择 Dial-out 接口，设置账号 88888888 和密码 12345678，即 user 和 password，选择 Add default router 添加默认路由，记住 user peer DNS 要选择上，该参数是使用对端服务器分配的 DNS

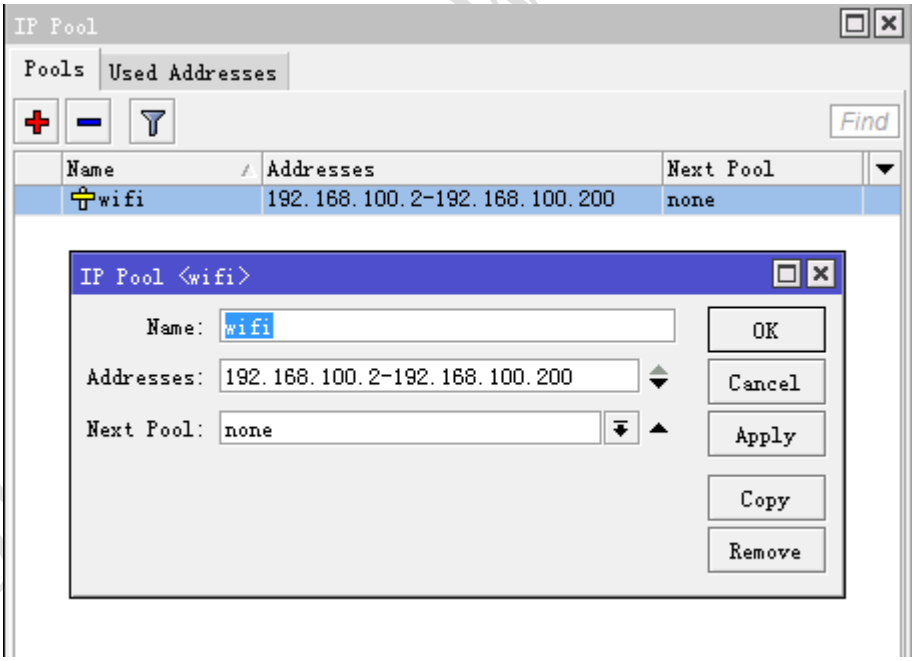
这样 pppoe 拨号设置完成，当连接上 ADSL Moden 后，RouterOS 会自动拨号，并获取 IP 地址

我们给 wlan1 网卡配置 IP 地址 192.168.100.1/24

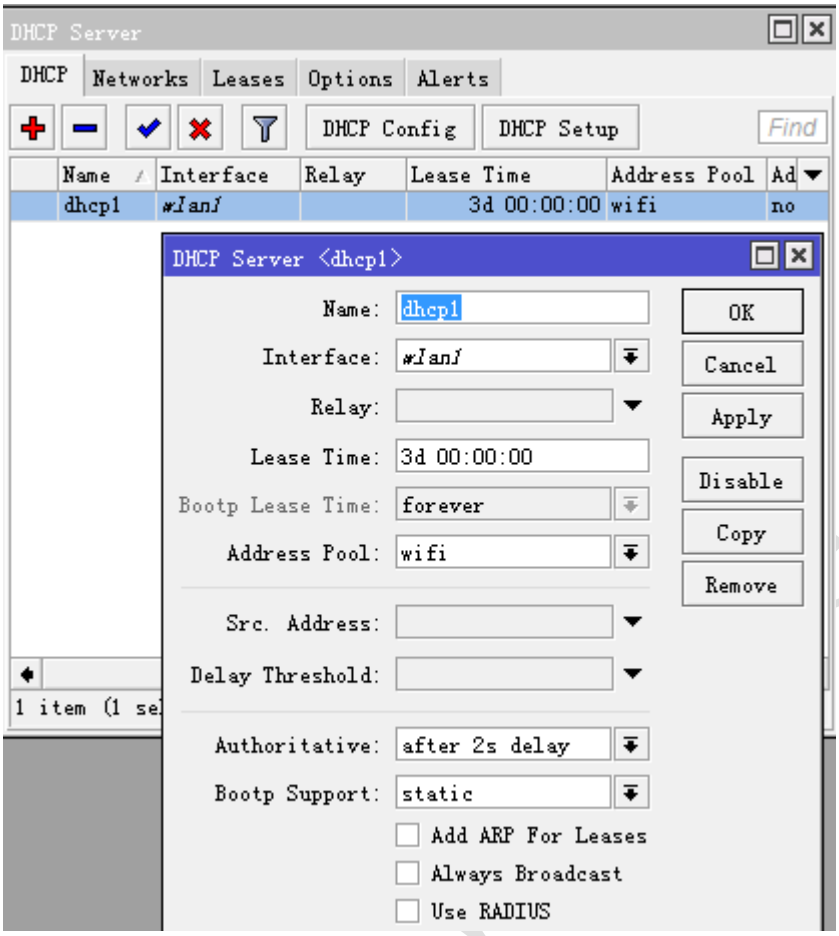


既然是 SOHO 的无线上网，用户肯定不用手动分配 IP 地址，需通过配置 DHCP 服务，向用户分配地址

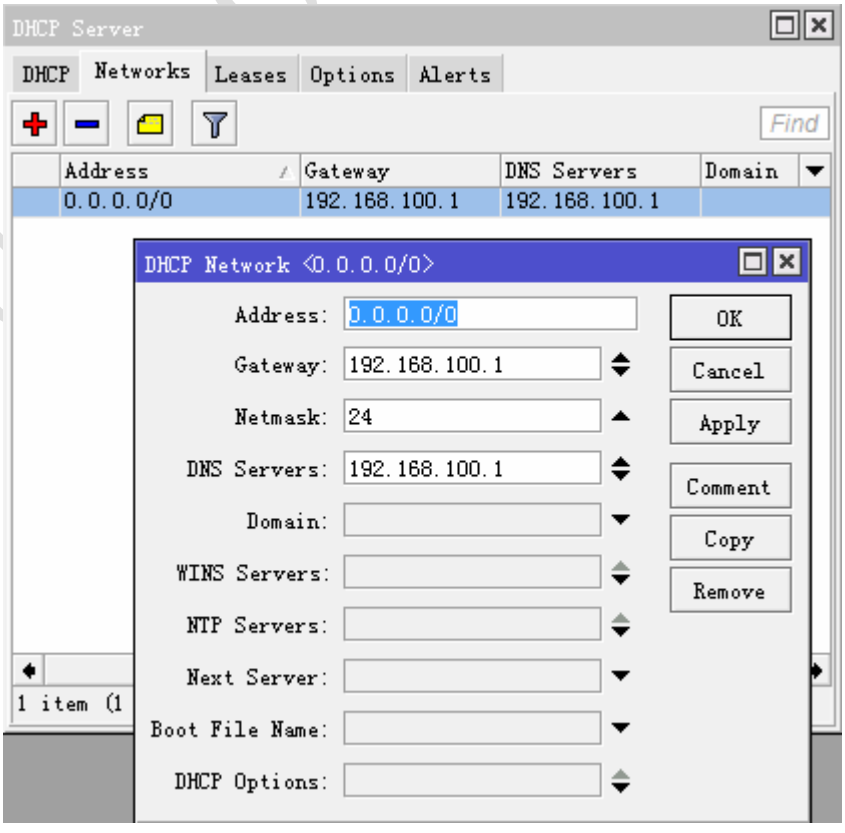
首先定义地址池 pool，进入 ip pool 中添加分配给用户的地址段 192.168.100.2-192.168.100.200



进入 ip dhcp-server 建立 DHCP 服务，选择我们刚才配置的 wifi 地址池

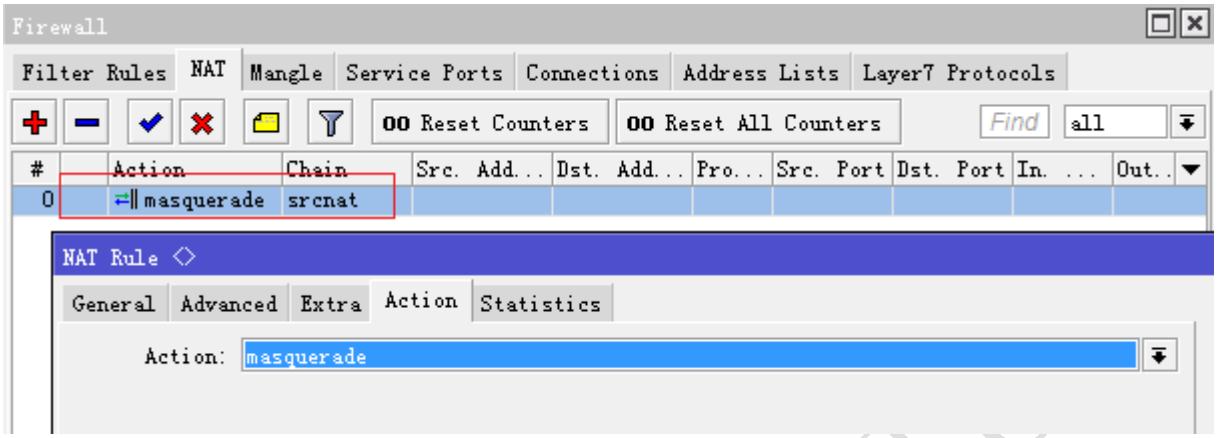


进入 network 选项，配置分配给用户的的网关、子网掩码和 DNS，这里我们启用了 DNS 缓存，直接使用网关作为用户 DNS

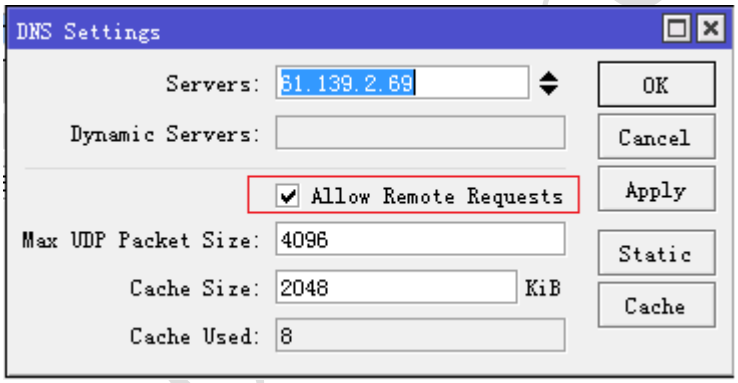


配置 nat 规则

进入 ip firewall nat，添加一条 srcnat 规则为 masquerade，转换内网 IP 到外网



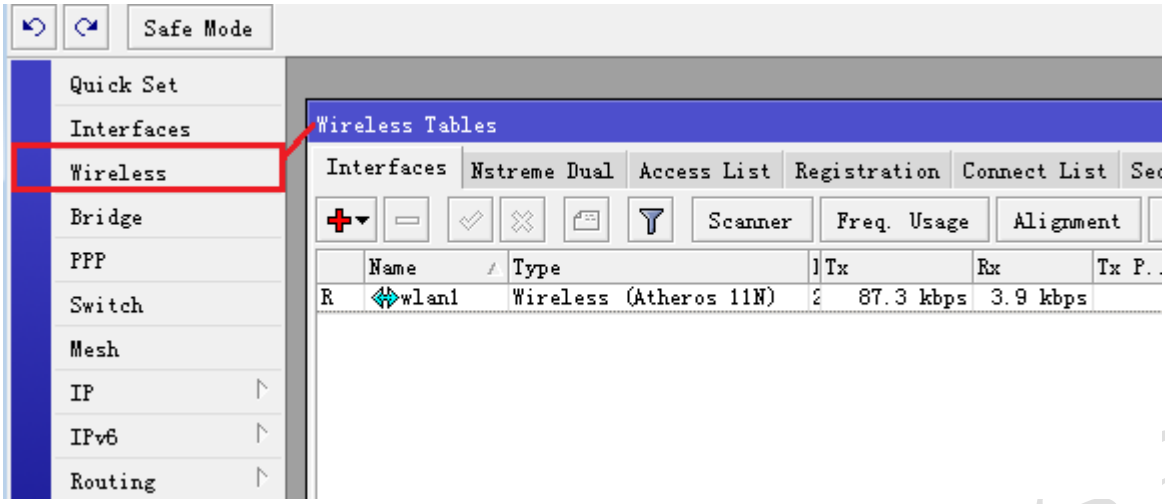
配置 DNS 缓存，即内网用户可以使用 192.168.100.1 作为 DNS



无线网卡配置

下一步，我们需要配置无线网卡，在 RouterOS 中所有 802.11 协议的无线网卡都识别为 wlan，根据数量的多少顺序编号，这里我们只有一张无线网卡，预设名称为 wlan1

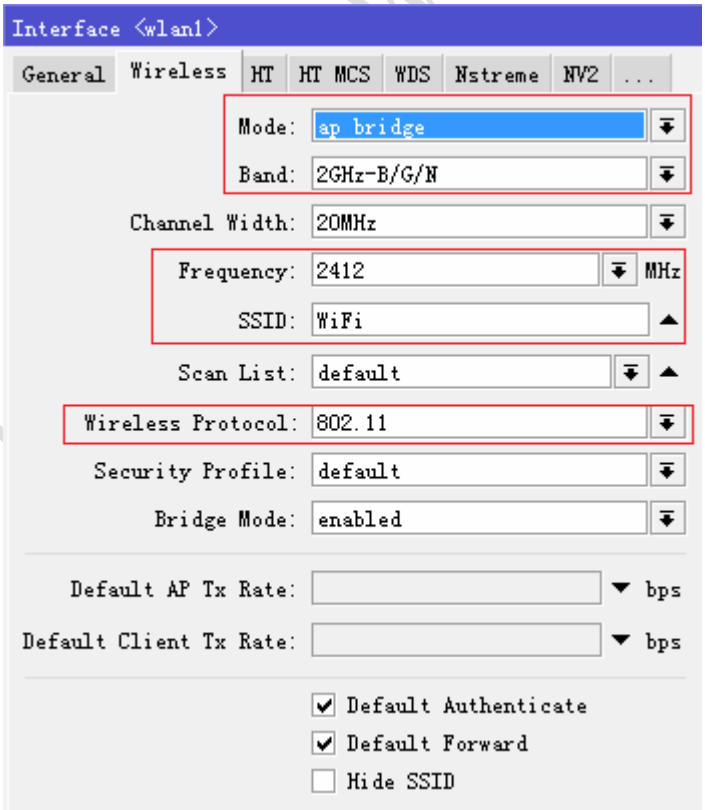
我们进入 Wireless 菜单，可以看到无线配置列表，里面有一张 wlan1 网卡，从 Type 属性可以看到无线网卡为 Atheros 11N 网卡



所有 WiFi 设备通用的配置基本上需要设置一下参数：

1、Mode：无线模式，覆盖都采用 ap-bridge 模式；
2、Band：使用的频段可以选择 2GHz 或者 5GHz，普通的 WiFi 覆盖通常采用 2GHz；
3、Frequency：无线发射频率，选择 1-11 频段，特定的国家可以增加 3 个频段；
4、SSID：即 Service Set Identifier 的缩写，即无线网络名称，建议不要设置中文，以免出现识别问题。

下面是进入 wlan1 的 Wireless 项目下的配置参数

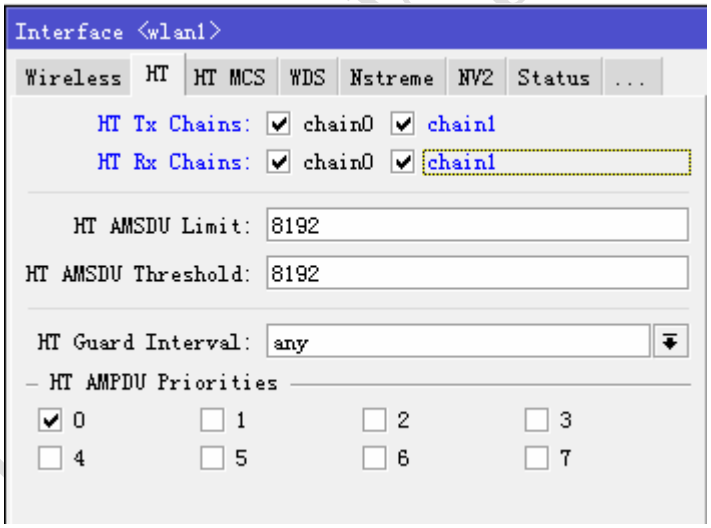


Wireless-protocol 参数

从 5.0rc1 开始加入了新的 wireless 设置参数 wireless-protocol，注意设置将根据所需要的无线网络模式，例如你需要兼容那几种模式，如下表：

值	AP	client
unspecified	基于老版本的 nstreme 或者 802.11	连接到老版本的 nstreme 或者 802.11
any	如同 unspecified	搜索所有匹配的网络，不论协议。
802.11	建立 802.11	只能连接到标准的 802.11 网络
nstreme	建立 Nstreme	只能连接到 Nstreme
nv2	建立 NV2	只能连接到 NV2
nv2-nstreme-802.11	建立 NV2	搜索 Nv2 网络，如果找到有适当的网络，并连接。 否则搜索 Nstreme 网络，如果找到有适当的网络，并连接。 否则搜索 802.11 网络，如果找到有适当的网络，并连接。
nv2-nstreme	建立 NV2	搜索 Nv2 网络，如果找到有适当的网络，并连接。 否则搜索 Nstreme 网络，如果找到有适当的网络，并连接。

采用 802.11n 的协议，需要配置 HT 参数，即采用 MIMO 是 1x1，还是 2x2 模式，如果你是 2x2 的 MIMO 设备，选择 chain0 和 chain1



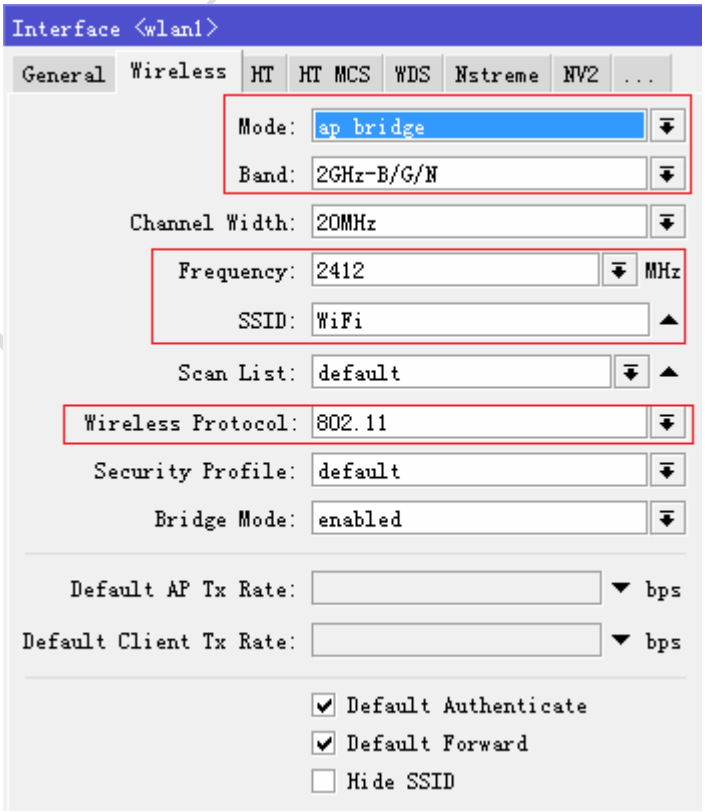
这样一个 SOHO 的 WiFi 覆盖就配置完成。

基于网桥的覆盖

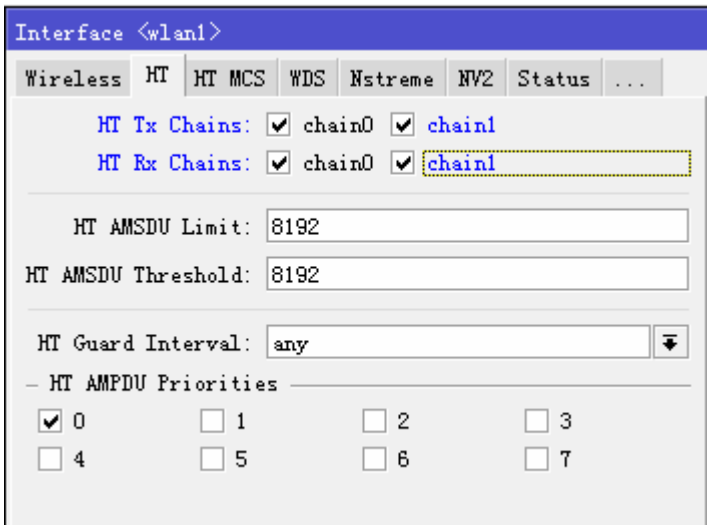
网桥覆盖，是在室内或周围空间较大，需要多个 AP 覆盖时采用的方案，且接入使用者较多，每个设备仅需要做普通网桥使用，配置比 SOHO 方式简单。

网桥覆盖与 SOHO 覆盖区别是将路由和 WiFi 分离开，路由负责外网连接，AP 负责 WiFi 覆盖

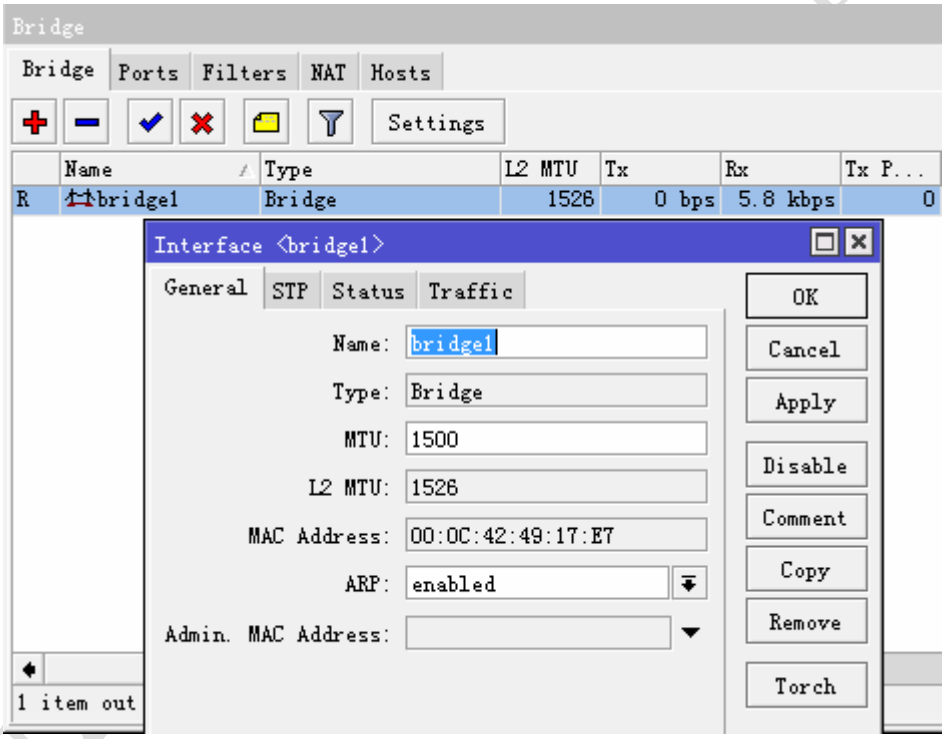
配置 wlan1 参数，基本与 SOHO 方式配置一样：



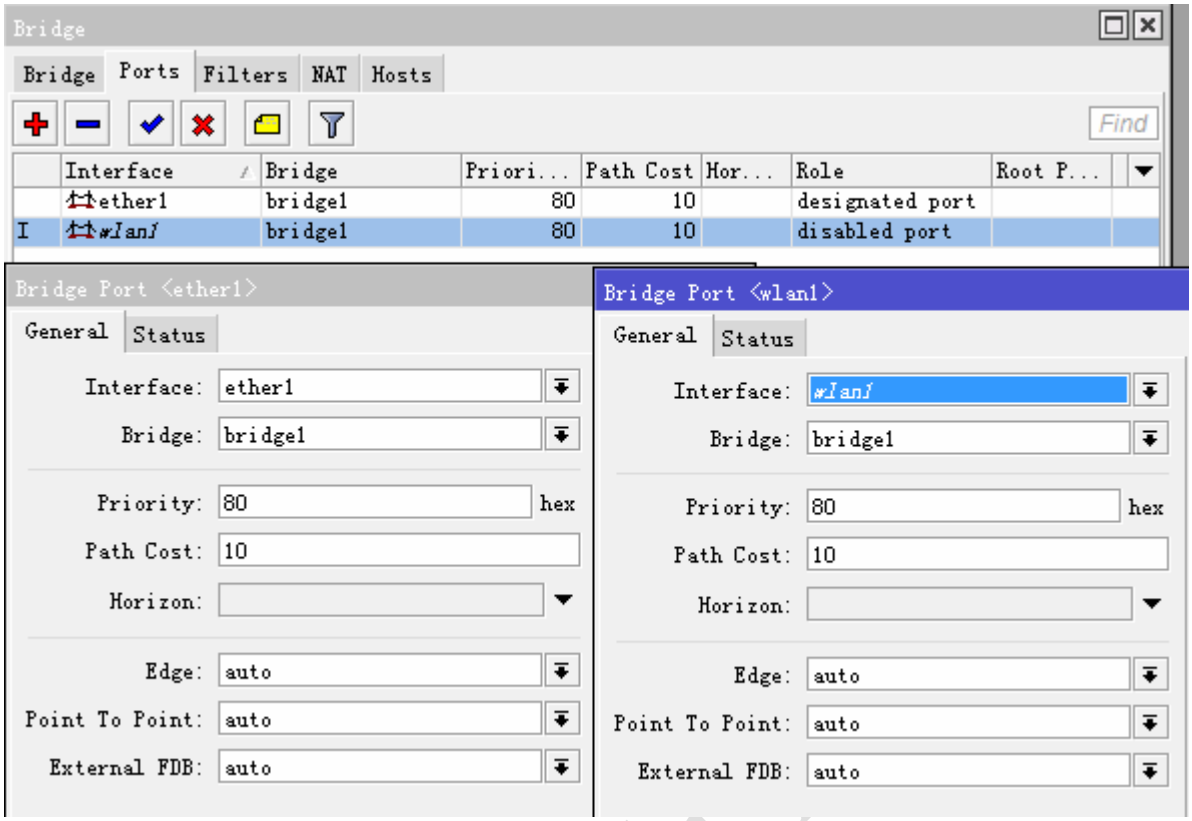
配置 HT 参数：



与 SOHO 的区别就在与配置网桥，我们需要进入 bridge 菜单下，添加 bridge1 规则

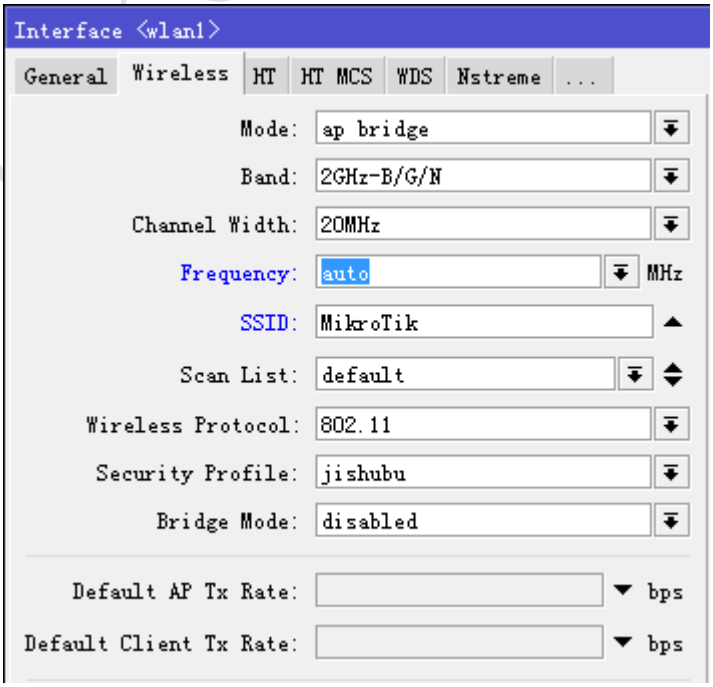


将 ether1 和 wlan1 添加到网桥 bridge1 中



自动频率选项

从 RouterOS v6.12 开始，你可以设置 AP 发射频率为“auto”（自动），这个特性可以避开干扰，并提升你的无线网络性能。无线频率 auto 设置简化了操作，RouterOS 会检查该无线网络区域，并选择一个频率，远离区域中的其他无线网络频率。设置如下



PS:这个功能只是方便大家设置网络，不过在复杂的无线网络环境中，也很难起到作用。

5.3 Access List 访问控制列表

操作路径： /interface wireless access-list

Access list 用于 AP 限制其他设备的连接，通过访问列表控制终端设备的各种参数。Access list 执行过程：

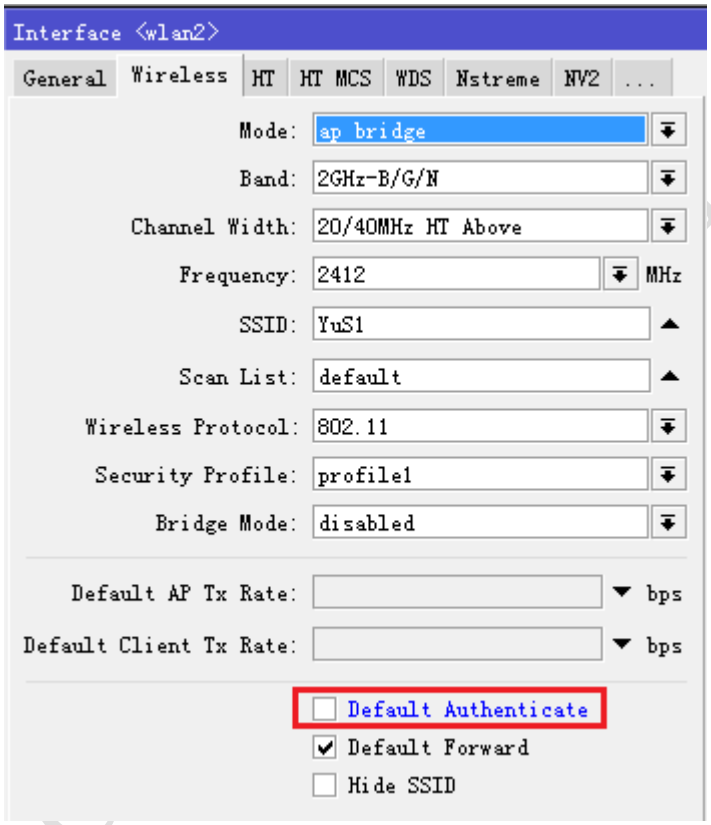
- Access list 规则通过循环检测是否存在匹配的规则
- 被禁用的规则会被忽略执行
- 当存在多条相同规则时，仅匹配从上往下的第一条规则
- 如果没有针对远程连接的匹配的规则，会使用默认的无线配置
- 当在 access list 的规则中选择了 **authentication**=*no*，那么该匹配规则的无线客户端将被拒绝连接

属性	描述
ap-tx-limit (*整型 [0..4294967295]*; 默认: 0)	速率控制，限制发送到客户端的速率，限制客户端下行。当设置为 0 时，表示不限制速率，单位为 bits。
authentication (*yes* | *no*; 默认: yes)	客户端验证程序 *no* - 客户端总是被拒绝连接 *yes* - 启动验证程序，指定该接口相应的 security-profile 安全策略，如果没有加密，使用默认配置
client-tx-limit (*整型 [0..4294967295]*; 默认: 0)	控制客户端发出的速率，限制客户端上行。当设置为 0 时，表示不限制速率，单位为 bits。 这个属性仅支持 RouterOS 的客户端
comment (*字符*; 默认:)	注释说明
disabled (*yes* | *no*; 默认: no)	禁用规则
forwarding (*yes* | *no*; 默认: yes)	数据转发，类似与客户端之间数据隔离 *no* - 客户端不能与其他客户端交换数据，仅能与相连的 AP 通信 *yes* - 客户端能通过相连的 AP 与其他客户端交换数据
interface (*字符* | *all*; 默认: all)	当规则设置为 interface=*all* ，即匹配所有的无线网卡，如果要匹配指定的网卡，可以通过该属性选择
mac-address (*MAC*; 默认: 00:00:00:00:00:00)	规则会匹配指定的客户端 MAC 地址，即对客户端 MAC 地址绑定。
management-protection-key (*string*; Default: "")	
private-algo (*104bit-wep* | *40bit-wep* | *aes-ccm* | *none* | *tkip*; 默认: none)	仅 WEP 加密模式支持
private-key (*字符*; 默认: "")	仅 WEP 加密模式支持
private-pre-shared-key (*字符*; 默认: "")	被用于 WPA 的 PSK 模式
signal-range (*数值范围 - 指定数值范围在-120..120*;	规则匹配客户端信号强度是否在指定的范围内

默认: -120..120)	如果客户端的信号超出了这个指定范围，AP 将与客户端断开连接
time (*时间范围,sun,mon,tue,wed,thu,fri,sat - 一天的时间选择*; 默认:)	规则匹配指定的时间周期 AP 将在指定的时间过后断开与客户端的连接。

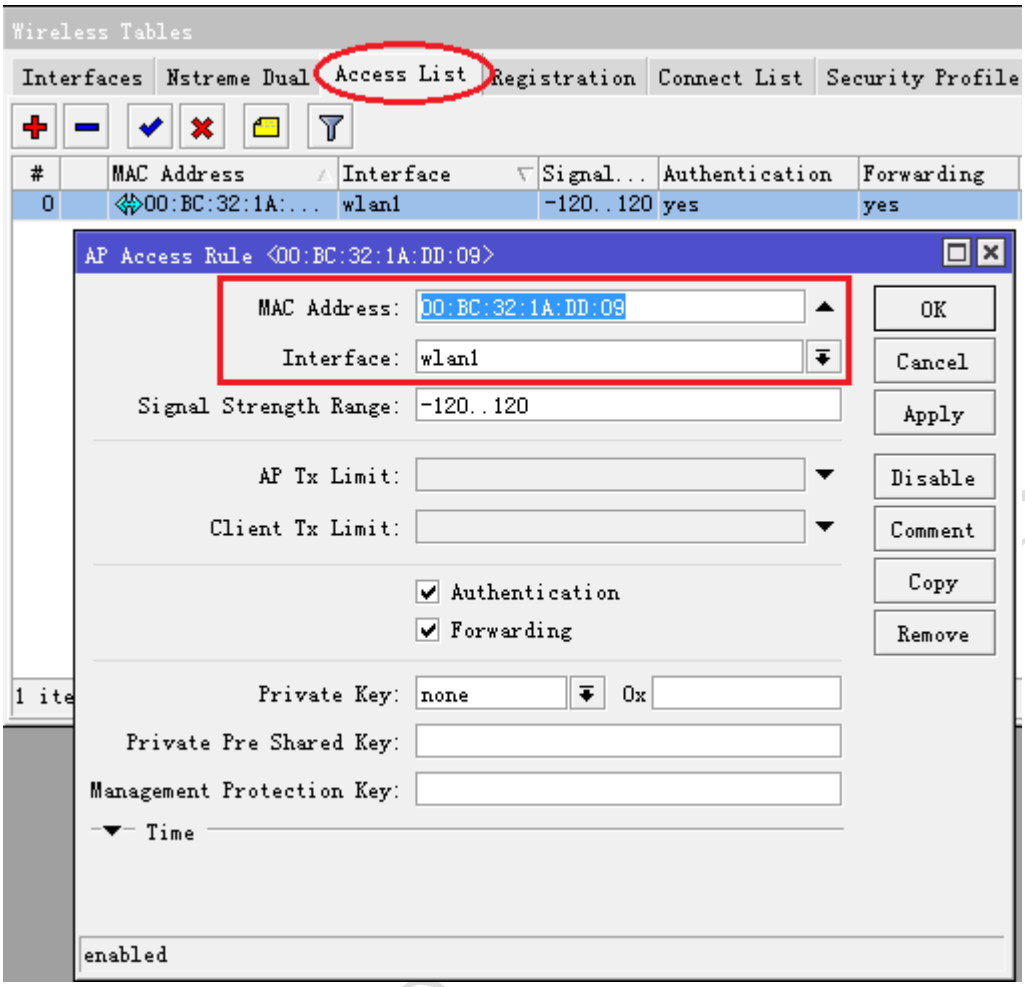
如何使用 Access-list 控制客户端

要让 access-list 中的规则生效，我们需要将 wireless 菜单下的 default-authenticate 参数选择为 no，即预设连接情况下客户端不允许自动验证通过

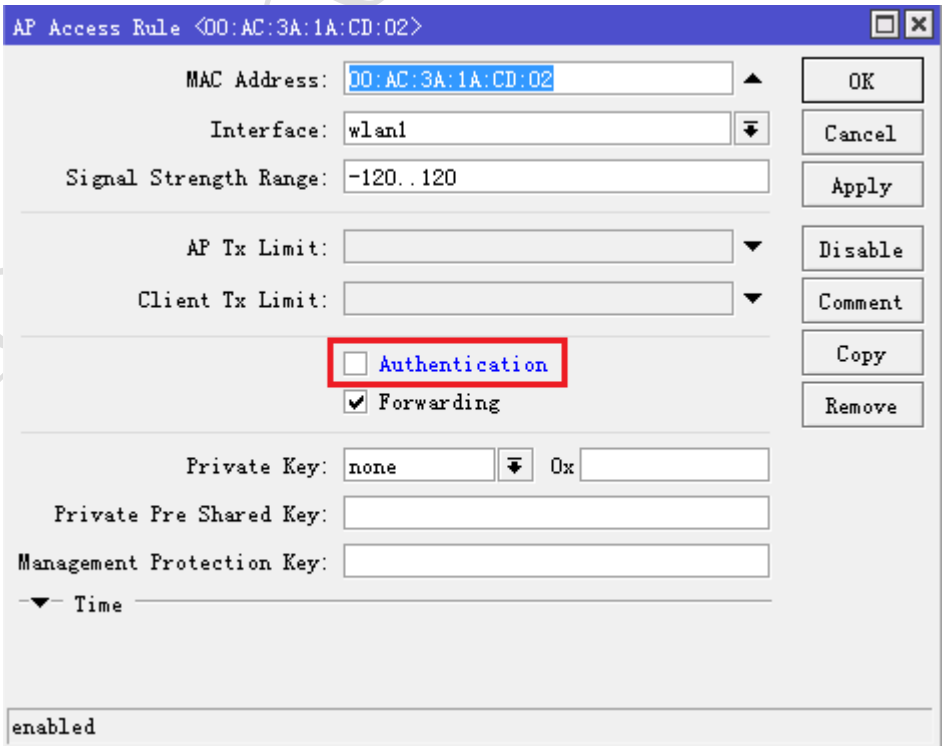


当我们关闭掉 default-authenticate，所有连接 AP 的客户端或者 station 要会进入 access-list 进行匹配，如果没有匹配的设备将无法连接到 AP。这样的操作类似于我们有线网络中通过 MAC 地址绑定计算机一样

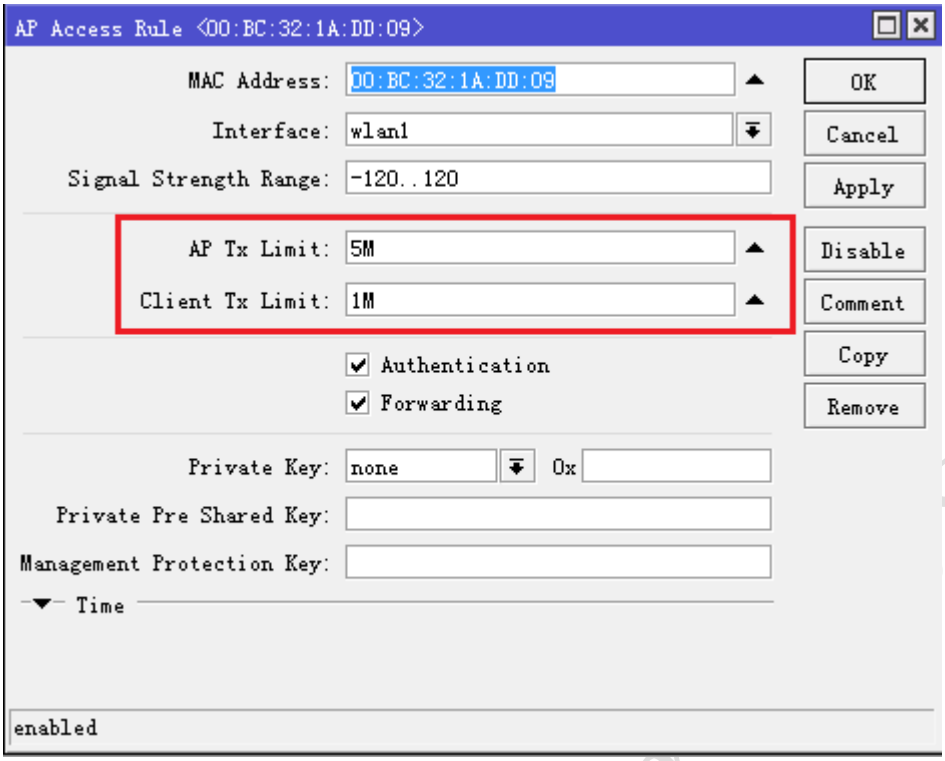
假如我们有这样一个客户端要对其进行连接控制，MAC 地址为：00:bc:32:1a:dd:09，连接无线网卡 wlan1



在 access-list 规则中的 authenticate 参数，当我们在规则中关闭后，表示对该用户拒绝连接到 AP，例如我们要禁止 MAC：00:AC:3A:1A:CD:02 的连接



当远程设备是 RouterOS 的 station，我们可以通过 ap-tx-limit 和 client-tx-limit 限制 station 的连接速率



如上图，我们限制了 00:BC:32:1A:DD:09 的 station 的速率，AP 发向 station 的带宽为 5Mbps，station 发向 AP 的带宽为 1M，即对于 station 而言下载为 5Mbps，上传为 1Mbps。

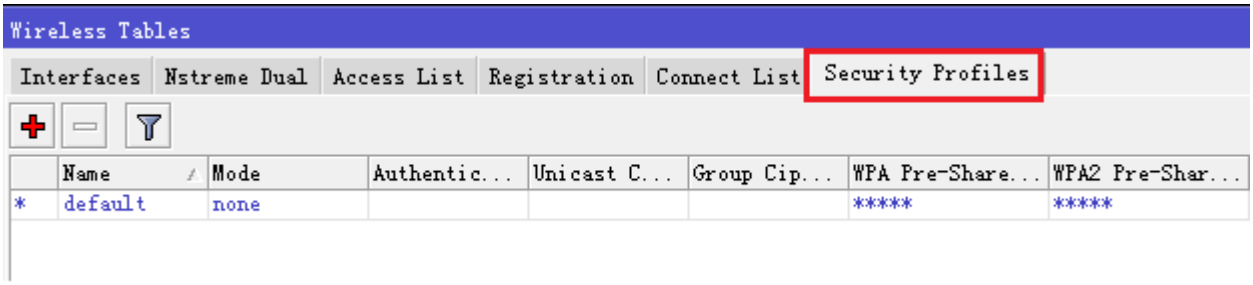
5.4 安全策略

操作路径: /interface wireless security-profiles

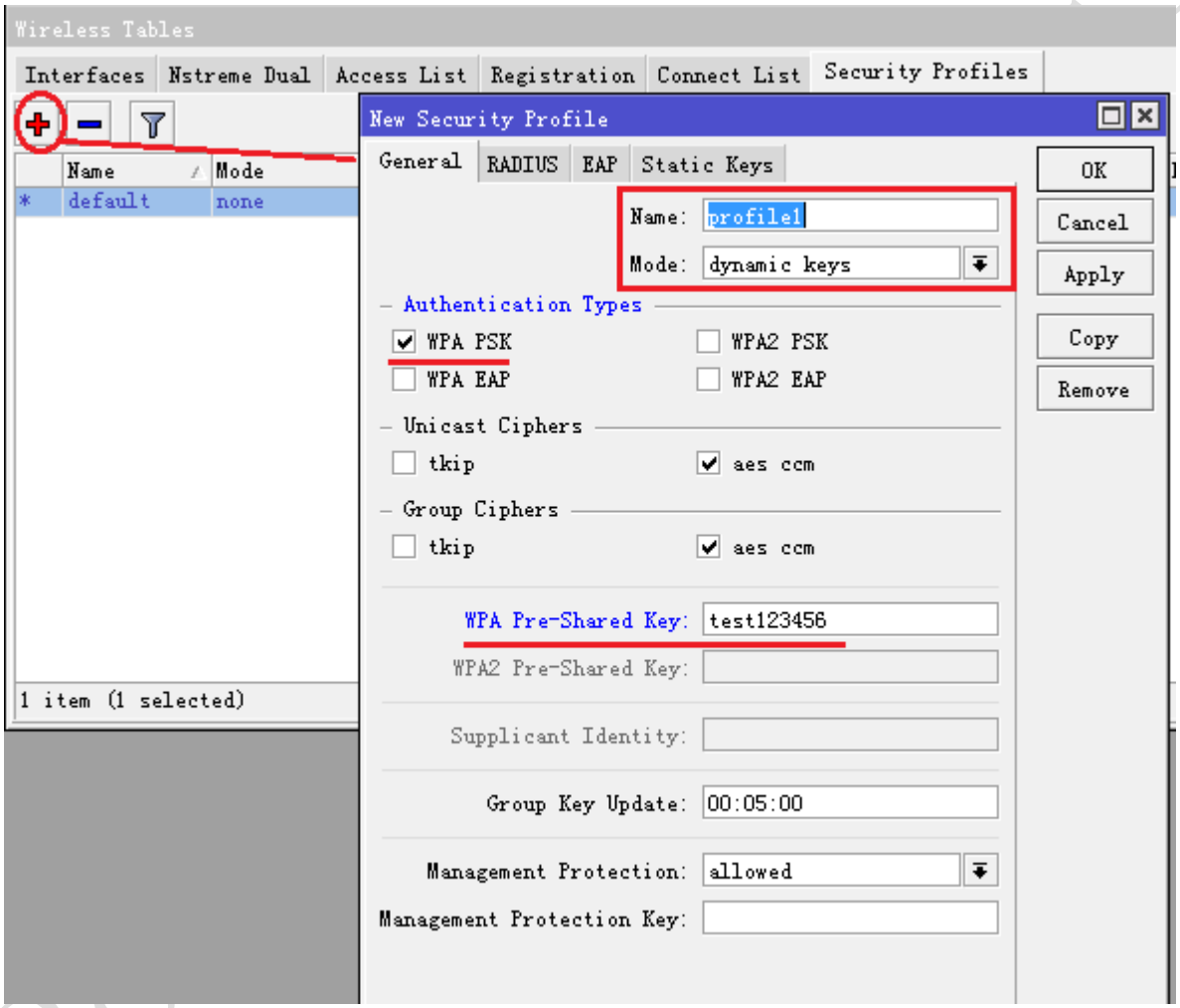
安全策略是在**/interface wireless security-profiles** 路径下配置，这里我们可以配置 802.11 传输协议的加密方式。配置安全策略规则定义不同的加密方式，规则被定义后，可以被应用到 Wireless 配置窗口里的 security-profile 参数

- **mode** (none, static-keys-optional, static-keys-required 或 dynamic-keys; 默认为: none):
- **none -** 不采用任何加密。
- **static-keys-required -** 采用静态加密的 WEP 模式，不接收也不发送为加密的帧。如果设备采用 station 模式下选择 static-keys-required，将无法连接到一个采用 **static-keys-optional** 的 AP 设备
- **static-keys-optional** -采用静态加密的 WEP 模式，支持加密和解密
- **dynamic-keys** - 动态加密的 WPA 模式

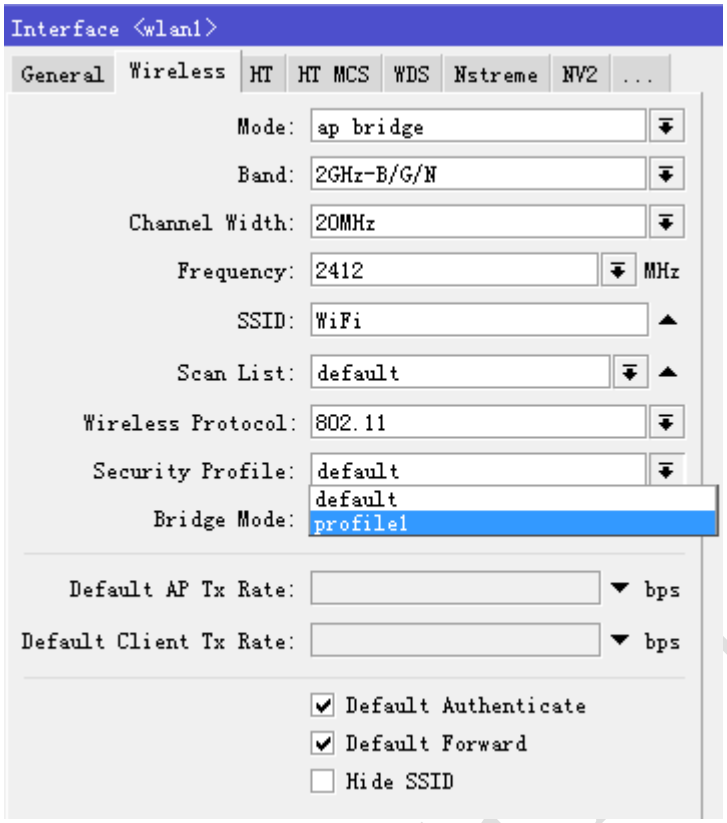
这里我们以 dynamic-keys 为事例，演示下 WPA 模式的配置，我们首先进入 Security Profiles 菜单下



新增加一条规则 profile1，选择 mode=dynamic keys，Authentication-types=WPA PSk，Unicast-Ciphers=aes ccm，并设置 WPA Pre-shared Key=test123456



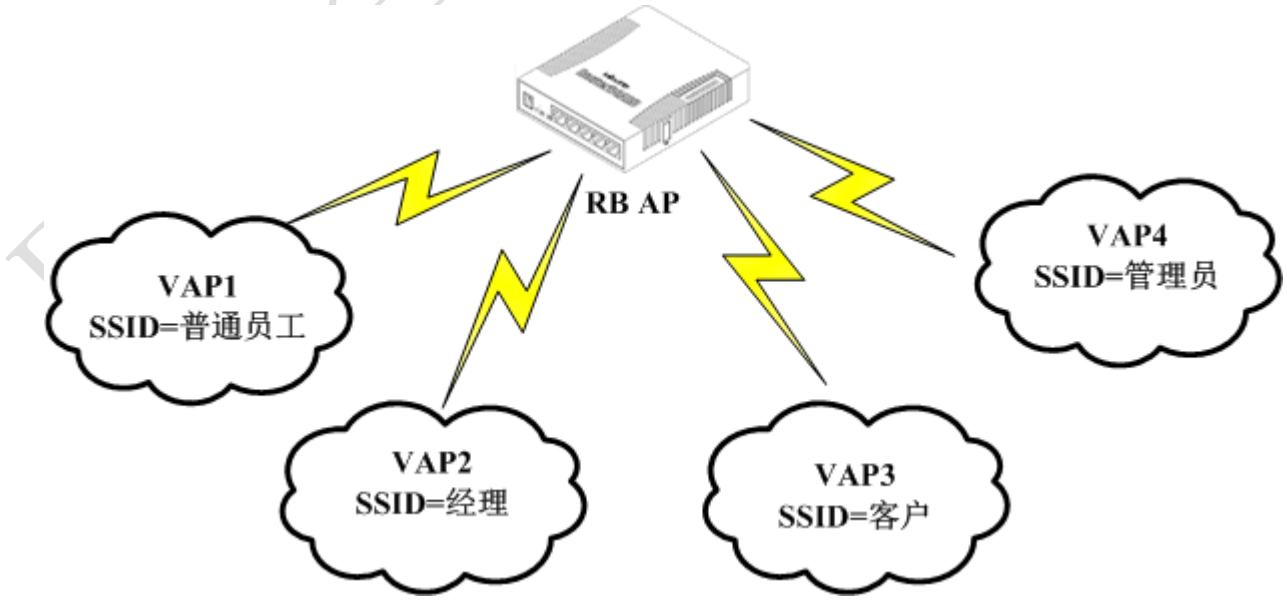
进入 wireless 菜单下，将 security-profile 设置为 profile1



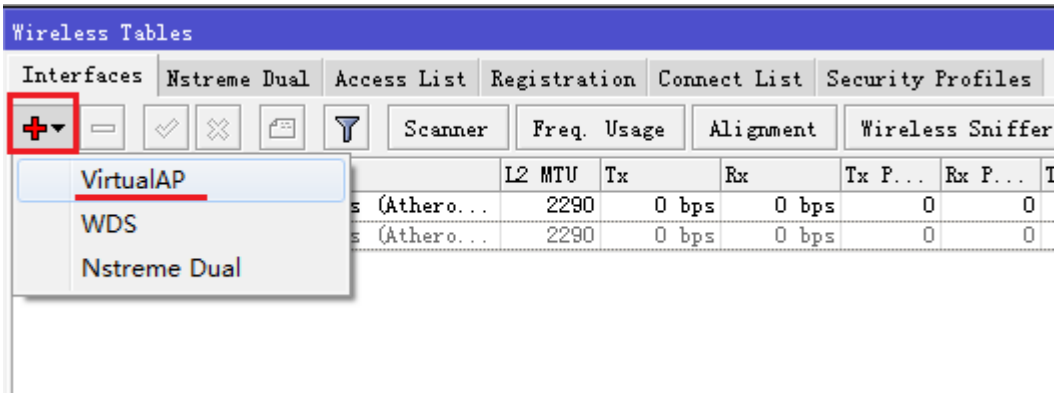
5.5 虚拟 AP(VAP)

当我们想通过一台 AP 在一个区域内建立多个 SSID，并对不同的 SSID 下的用户进行管理，例如：办公区域里，可以区分为员工、经理和老板的 WiFi 上网，在家庭中可以用来区分父母和子女的 WiFi 上网

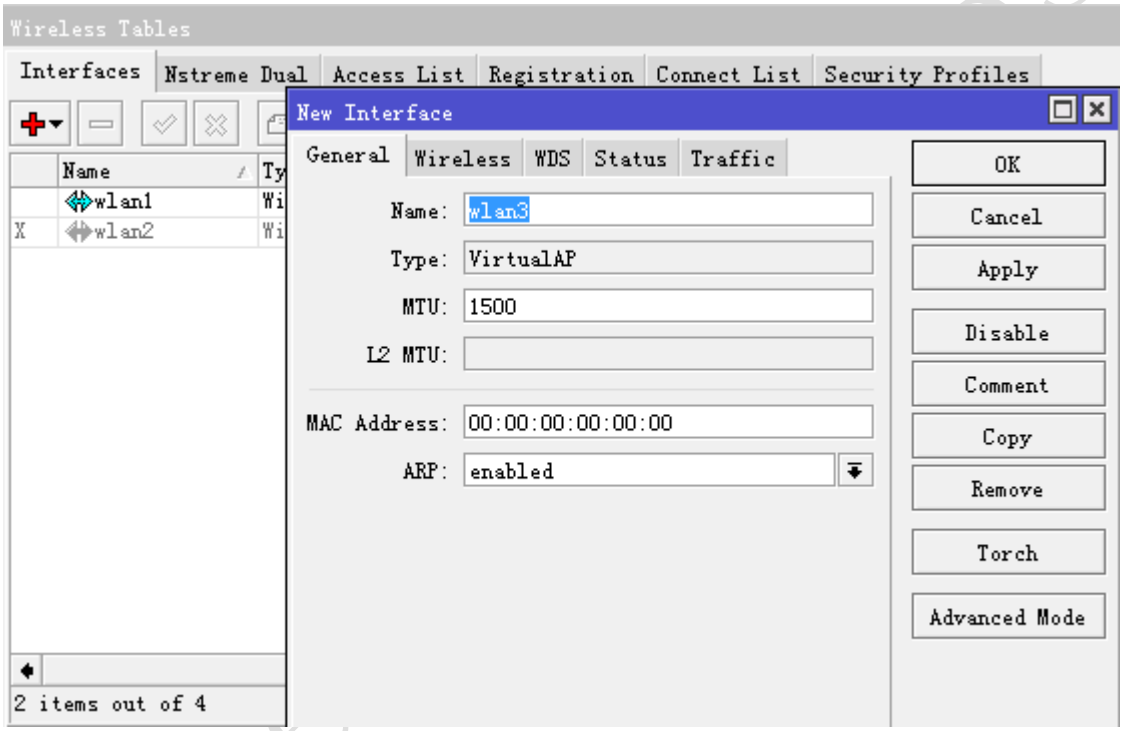
虚拟 AP 的好处在与通过一个物理网卡，模拟出多个 AP 信号，在一个区域内广播多个 SSID，让不同的用户选择对应的 SSID。当然虚拟 AP 也支持加密的安全策略，有助于你对 WiFi 网络的区域划分和管理。



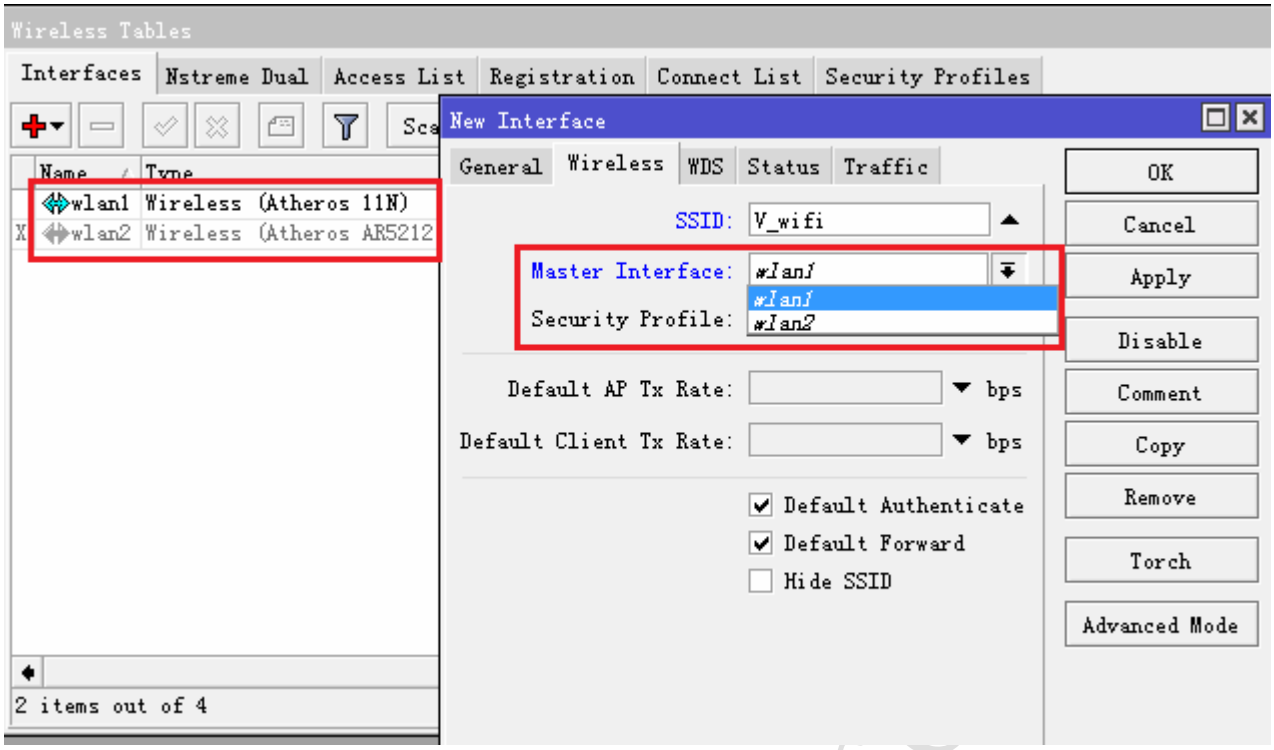
我们建立一个虚拟 AP，可以进入 wireless 目录下，点加号可以找到 VirtualAP



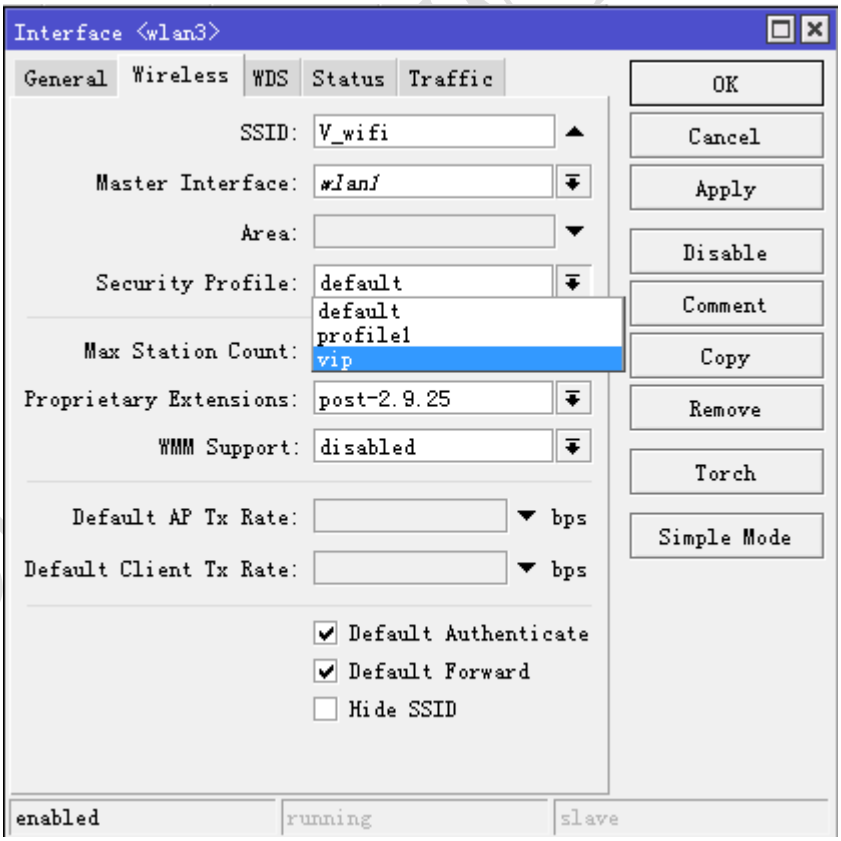
打开后，我们新建一个名为 wlan3 的虚拟 AP，可以看到 type 为 VirtualAP



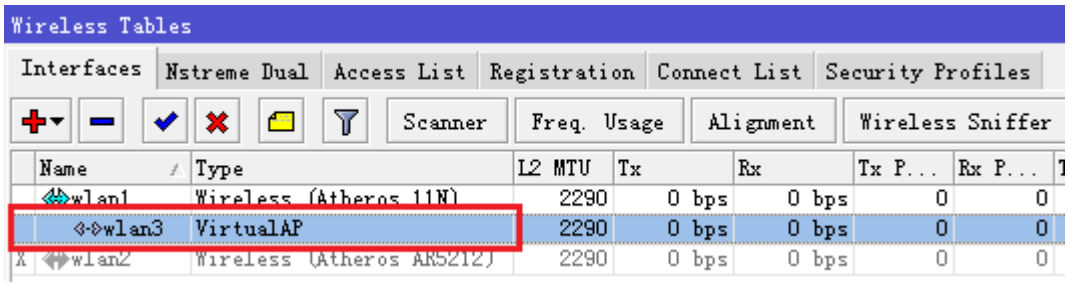
打开虚拟 AP 的 wireless 菜单，可以到基本和物理网卡相同的配置参数，主机 Master-interface 是选择虚拟 AP 从属于那个物理网卡



我们定义了 SSID 为 V_wifi，从属于 wlan1 物理网卡，且我们选择加密的 security-profile=vip



添加完成后，我们可以看到 wlan3 从属于 wlan1 网卡下：



这时你可以在终端设备上搜索到 V_wifi 的信号。

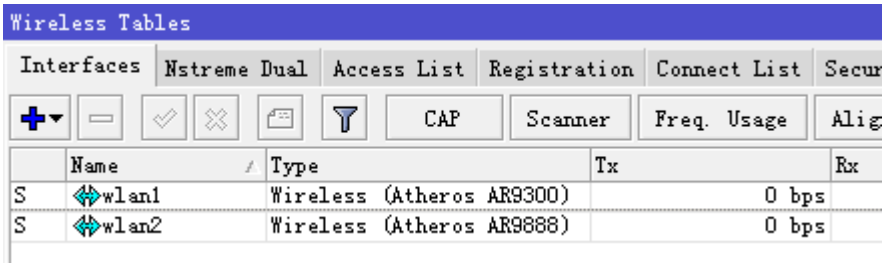
5.6 hAP ac 双频合一配置

hAP ac lite、hAP ac 和 hAP ac2（包括 wAP ac 和 cAP ac）三款基于 802.11ac 的家用办公无线路由器，支持 2.4G 和 5G 双频，也就是 802.11bgn 和 802.11ac 两个协议同时工作。由于 802.11ac 采用 5G 频率覆盖范围有限，一般仅能在视距内传输，但能提供更高的带宽，如果 3×3 80MHz 的 MIMO 可以获得 1.3G 的带宽，如果非视距内只能通过 802.11bgn 来弥补。下面是关于 802.11n 和 802.11ac 的技术参数，以及 hAP ac 和 hAP ac lite 在无线 MIMO 的区别

技术规格	802.11n	802.11ac
频率	2.4G，5G	5G
调制方案	OFDM	OFDM
信道带宽	20，40MHz	20，40，80MHz
单流额定传输率	150Mbps（1×1 40MHz）	433Mbps（1×1 80MHz）
多流额度传输率	450Mbps（3×3 40MHz）	1.3Gbps（3×3 80MHz）
hAP ac MIMO	3×3 支持 450Mbps	3×3 支持 1.3Gbps
hAP ac lite MIMO	2×2 支持 300Mbps	1×1 支持 433Mbps

不过作为客户端同时只能连接一个频率，要么 2.4G 的 802.11bgn 或者 5G 的 802.11ac，一般 802.11ac 的路由器会提供 2.4G 和 5G 两个配置，即两个无线网络一个 2.4G 或一个 5G，配置不同两个不同的 SSID。也可以配置所谓的双频合一，即使采用双频合一相同的 SSID，客户端也只会连接到一个频率上，只是会在两个频率上根据信号强弱做切换。很多厂商的无线路由器提供了双频合一的设置，这样的设置客户端也只是在 5G 信号好的时候连接 802.11ac，当 5G 信号变弱后，切换到 2.4G 的 802.11bgn。

在 hAP ac 路由器里可以看到两个无线网卡，一个 wlan1 是 802.11bgn，一个 wlan2 是 802.11ac



两张无线网卡负责不同的协议，那该如何设置双频合一，AR9300 负责 802.11bgn，AR9888 负责 802.11ac，我的思路是把两个无线网卡做 WDS 漫游方式，即两个网卡 SSID 相同，无线网卡 mode=ap-bridge，通过桥接创建 rstp 协议的 WDS 无线漫游。配置如下

1、桥接配置

进入 bridge 创建 bridge1

```
/interface bridge
add name=bridge1 protocol-mode=rstp
```

进入 bridge port 将 wlan1 和 wlan2 加入 bridge1

```
/interface bridge port
add bridge=bridge1 interface=wlan1
add bridge=bridge1 interface=wlan2
```

2、网络配置

配置路由器 bridge1 的 IP 地址，即分给用户的 IP 地址段

```
/ip address
add address=192.168.88.1/24 interface=bridge1
```

创建 DHCP 服务，配置地址池：

```
/ip pool
add name=pool1 ranges=192.168.88.2-192.168.88.100
```

配置 DHCP 服务的接口和地址池

```
/ip dhcp-server
add address-pool=pool1 disabled=no interface=bridge1 name=server1
```

配置 DHCP 服务分配给用户的网关和 DNS 服务器

```
/ip dhcp-server network
add dns-server=192.168.88.1 gateway=192.168.88.1 netmask=24
```

配置 DNS 服务器 IP 地址和开启 DNS 本地解析

```
/ip dns
set servers=61.139.2.69 allow-remote-requests=yes
```

启用 nat 转换

```
/ip firewall nat
add action=masquerade chain=srcnat
```

3、无线网络配置

配置无线安全密码，创建 wpa/wpa2 的无线密码，设置为 1234567890

```
/interface wireless security-profiles
add mode=dynamic-keys authentication-types=wpa-psk,wpa2-psk
group-ciphers=tkip,aes-ccm name=yus unicast-ciphers=tkip,aes-ccm
wpa-pre-shared-key=1234567890 wpa2-pre-shared-key=1234567890
```

配置无线网卡，设置两个无线网卡 SSID 相同取名 yus，设置 wds 模式为 dynamic-mesh，wed-default-bridge 为 bridge1

```
/interface wireless
set [find default-name=wlan1] ssid=yus band=2ghz-b/g/n disabled=no
frequency=2422 mode=ap-bridge security-profile=yus
wds-default-bridge=bridge1 wds-mode=dynamic-mesh

set [find default-name=wlan2] ssid=yus band=5ghz-a/n/ac
channel-width=20/40/80mhz-Ceee disabled=no frequency=5745 mode=ap-bridge
security-profile=yus wds-default-bridge=bridge1 wds-mode=dynamic-mesh
```

以上配置完成后，终端设备会自动连接信号最强的频率，例如首先当连接上 5G 的 802.11ac，当你移动 5G 信号变弱后，终端设备会自动连接到 2.4G 的 802.11bgn，但如果你要从 802.11bgn 切换到 802.11ac，需要你终端设备去完成，而非路由器决定。无线漫游的切换都是由终端设备决定的。

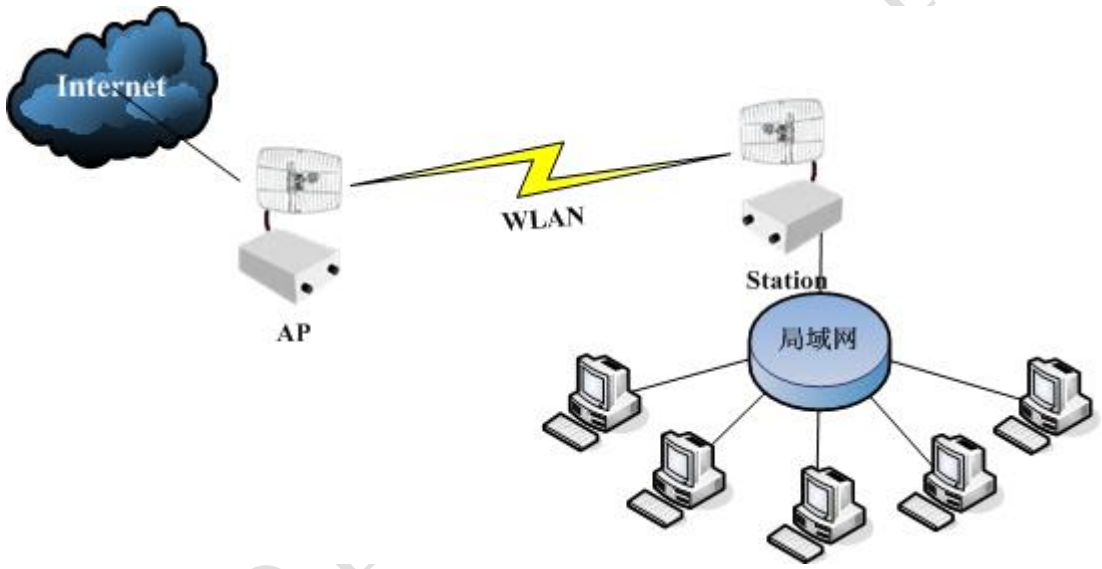
第六章 WLAN 点对点

现在随着无线 Wlan 技术的发展，带宽和距离已经得到成功解决，点对点的无线 wlan 传输已经非常成熟。通过 wlan 传输可以达到上百公里，最高带宽能到 100M。点对点的无线传输能为偏远的使用者上网或者需要低成本解决网络问题的用户提供方便快捷和低成本的接入。

6.1 点对点传输介绍

在 WLAN 点对点传输配置和应用是整个 WLAN 无线网络构建的基础，即后面的点对多点、中继传输、WDS 和 Mesh 网络都是基于点对点传输而变化的，所以在点对点应用中我们需要特别注意。

通过无线点对点的传输，将 Internet 数据连接到远程的局域网内，如图：



无线点对点传输优点：

1、长距离传输成本低，如果一个 5 公里的网络接入，通过光纤布线施工加材料费用成本在 5 万以上，如果是采用 wlan 的无线接入，如一套“Groove+抛物面天线”的点对点设备和配件不会超过 3 千元。

2、安装时间短，如果是光线接入 5 公里，施工安装时间也要好几天，而 wlan 接入，只需要两点间无阻挡，安装点确定好后，双方架好设备，直接安装，并调整信号也只需要 1 小时左右。

3、可持续性使用，由于无线安装方便，当这个点无需无线连接后，我们将设备换到其他需要无线网络的地方继续使用，而线缆在预埋好后，则很难再取出，即使取出也需要投入施工费用，几乎没有可持续使用的条件。

更宽的发射频率范围：

在发射频率方面，通过升级 superchannel 还可以得到更宽的频率，范围为：802.11b/g：2.312 – 2.497 (5MHz step)，802.11a： 4.920 – 6.100 (5MHz step)，更宽的频率能获得更稳定的传输频率。

无线设备传输带宽情况：

型号	性能	点对点，最大 TCP 传输
RB411/A/R	Atheros 300Mhz CPU, 64M 内存，1 个百兆 LAN 口，1 个 MiniPCI	40Mbps-50Mbps
RB433	Atheros 300Mhz CPU, 64M 内存，3 个百兆 LAN 口，3 个 MiniPCI	40Mbps-50Mbps
RB411AH	Atheros 680Mhz CPU, 64M 内存，1 个百兆 LAN 口，1 个 MiniPCI	60Mbps-80Mbps
RB433AH/U	Atheros 680Mhz CPU, 128M 内存，3 个百兆 LAN 口，3 个 MiniPCI	60Mbps-80Mbps
RB711	Atheros 400Mhz CPU, 32M/64M 内存，1 个百兆 LAN 口，集成 1 个 5GHz 802.11a/n Atheros AR9280	80Mbps-93Mbps
RBSXT-5HnD	Atheros 400Mhz CPU, 64M 内存，1 个百兆 LAN 口，集成 1 个 5GHz 802.11a/n Atheros AR9280，1 个 USB	80Mbps-93Mbps
RB800	PowerPC 800MHz 处理器，256M 内存，3 个千兆 LAN 口，4 个 MiniPCI，1 个 MiniPCI-e	150Mbps-180Mbps

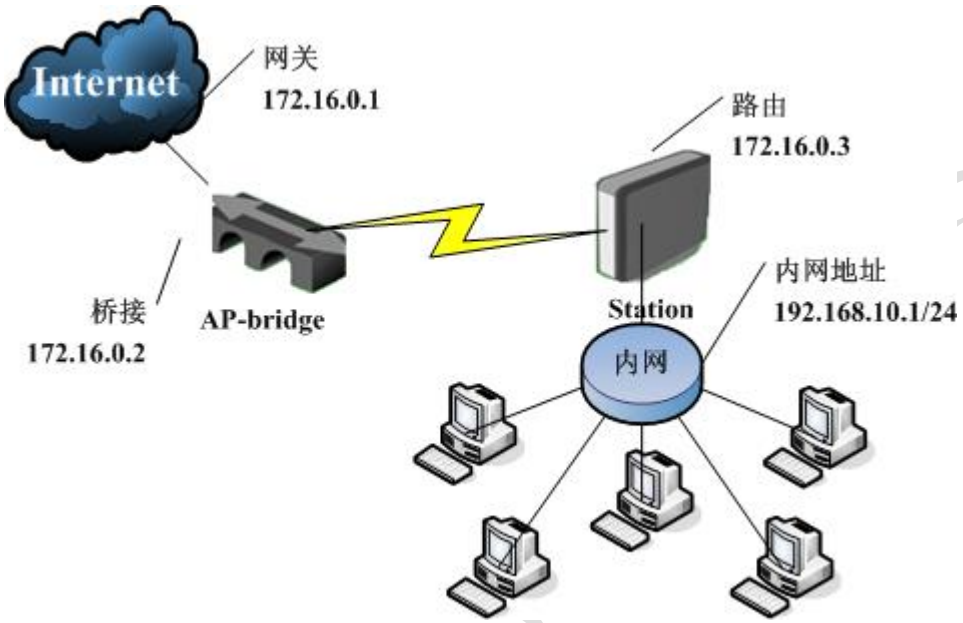
MikroTik 点对点模式的多种方式：

模式	应用	Nstreme 协议	Nv2	性能与提升
AP-Bridge to Station	路由模式，启用桥接需要配置 EoIP	支持	支持	取决于网卡速率和选择模式，支持 108M 模式和 11n 的 300M
AP-Bridge to Station-WDS	常用于桥接模式，自动添加到桥接设置中，也可用于路由	支持	支持	取决于网卡速率和选择模式，支持 108M 模式和 11n 的 300M
AP-Bridge to AP-Bridge	支持桥接和路由模式，桥接模式自动添加到桥接设置中，同样支持 MESH 和 WDS 模式	不支持	不支持	取决于网卡速率和选择模式，支持 108M 模式
Bridge to Bridge	桥接模式	不支持	不支持	取决于网卡速率和选择模式，支持 108M 模式
Bridge to station-bridge	为 Nv2 协议开发，替代 bridge to bridge	不支持	支持	取决于网卡速率和选择模式，支持 108M 模式和 11n 的 300M
bonding	桥接模式	支持	不支持	取决于网卡速率和选择模式，在 bonding 模式下的效果 1+1〉2，非常消耗 CPU 资源
Nstreme-dual	桥接和路由模式，双向一个发送，一个接收	支持	不支持	双向传输，取决于网卡速率和选择模式，支持 108M 模式

以上列表显示了多种无线模式的连接应用方式，Nstreme 协议是 MikroTik 长距离传输和获取高带宽下采用的，这样的模式可以让你的无线网络带宽得到有效的提升，5.0 后支持 Nv2 协议，能有效提升 11n 的带宽，最高可以达到 180Mbps。

6.2 AP-Bridge to Station 路由模式

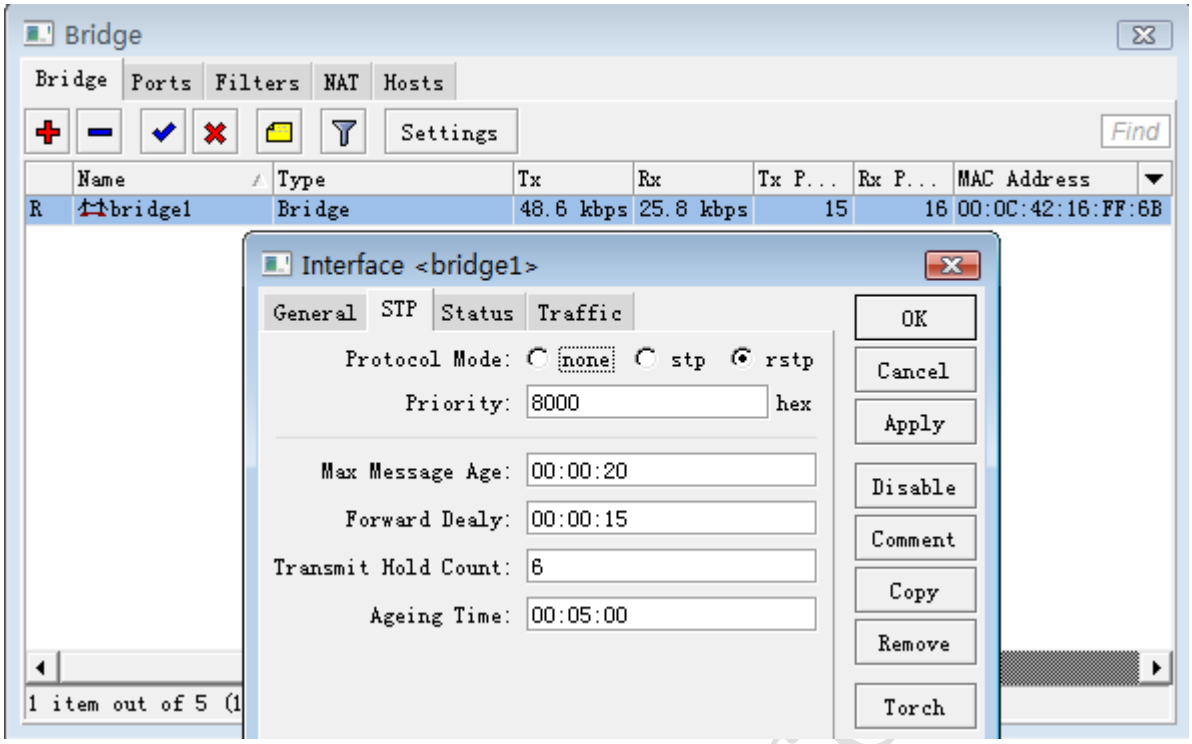
AP-bridge 和 Station 模式传输常用于路由模式，AP-bridge 设备仍然设置为桥模式，不启用 WDS。Station 设备则配置为路由模式。 如下图，外网的 Internet 通过 AP-bridge 设置的桥（IP 地址为 172.16.0.1）连接到 Station 端的路由（无线接口 IP 地址为 172.168.0.2，内网的以太网接口 IP 为 192.168.10.1，连接内网的 192.168.10.0/24 网络）

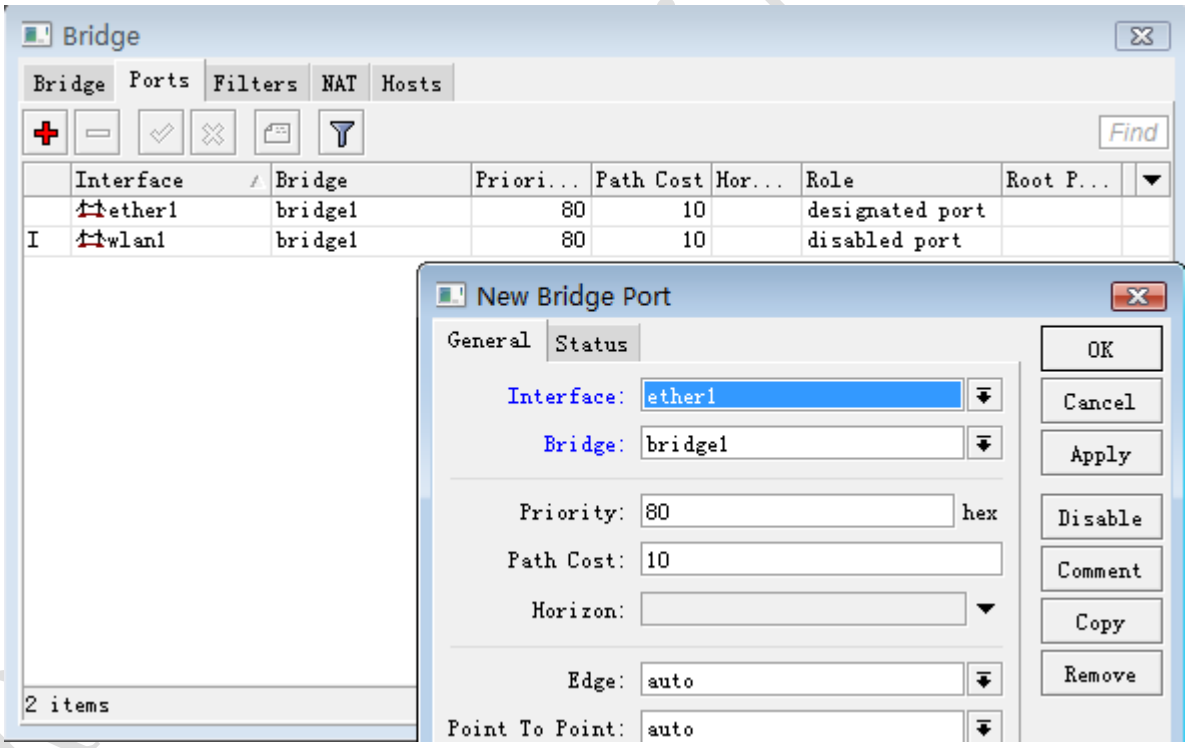


AP-bridge 与 Station 模式配置步骤：

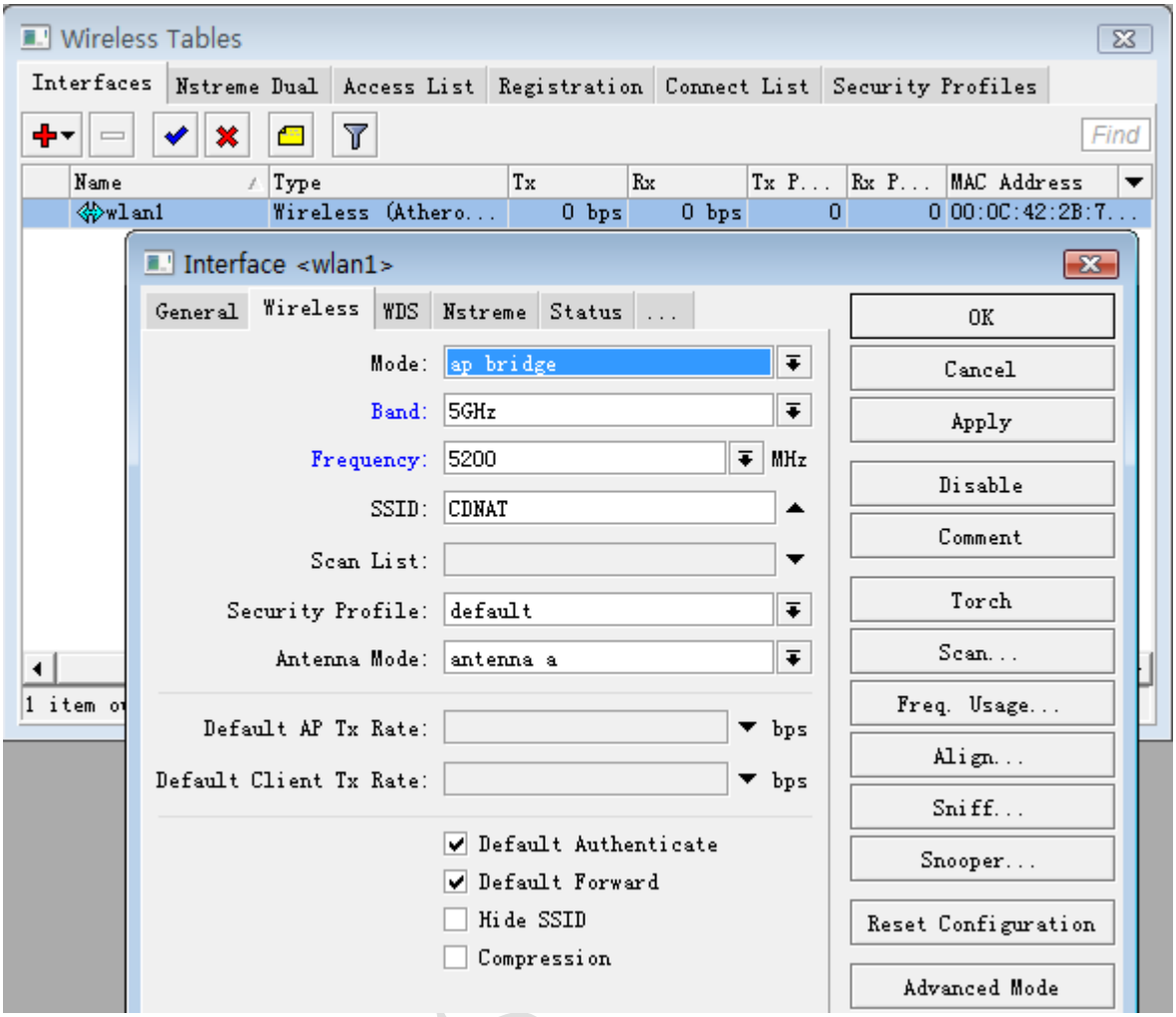
1、 在 AP-bridge 端配置桥接，将 Wlan1 和 ether1 添加如 bridge 中；
2、 配置 AP-bridge 无线参数，并在 ip address 中配置 bridge 的 IP 地址；
3、 在 Station 端配置 Wlan1 的无线 IP 地址，配置 ether1 的内网 IP 地址；
4、 配置 Station 端的无线参数，并测试无线网络与内部网络是否连接。

步骤 1：设置 AP-bridge 端的桥接，进入 birdge 添加一个桥，并设置 Port 的界面：

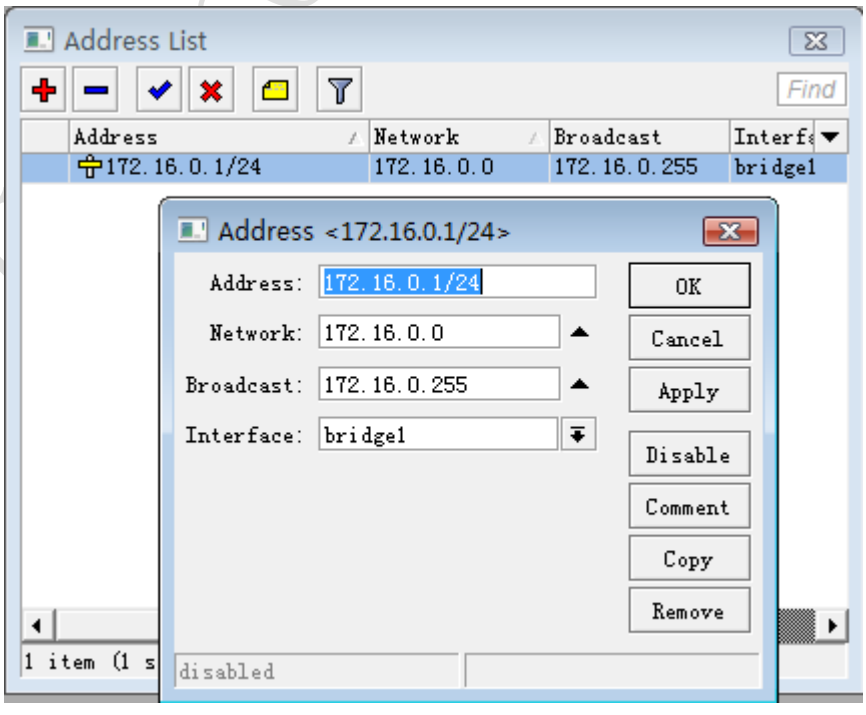




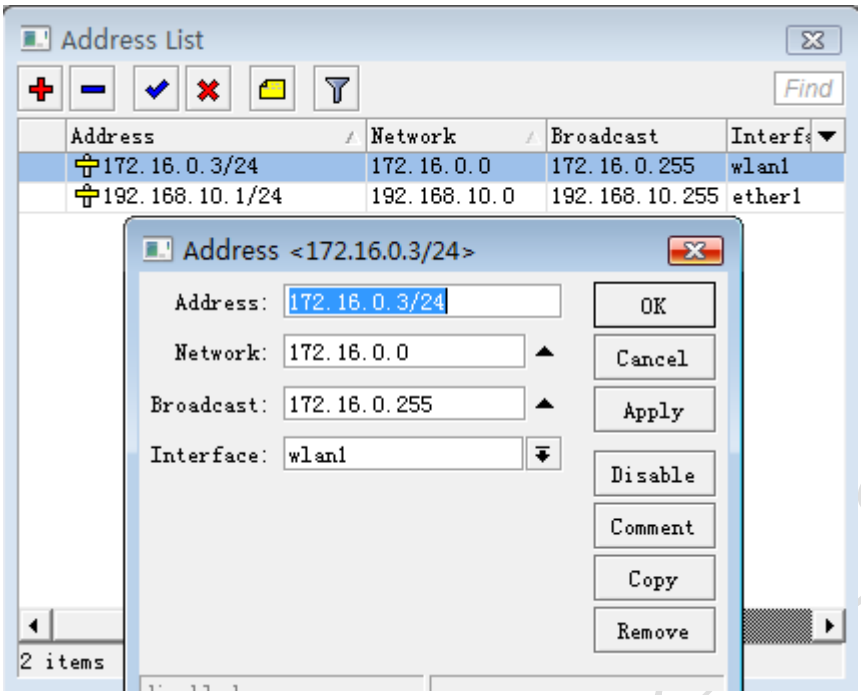
步骤 2：进入 wireless 配置 wlan1 的参数，配置 mode=ap-bridge，Band=5GHz，Frequency=5200，SSID=CDNAT，其他参数默认：



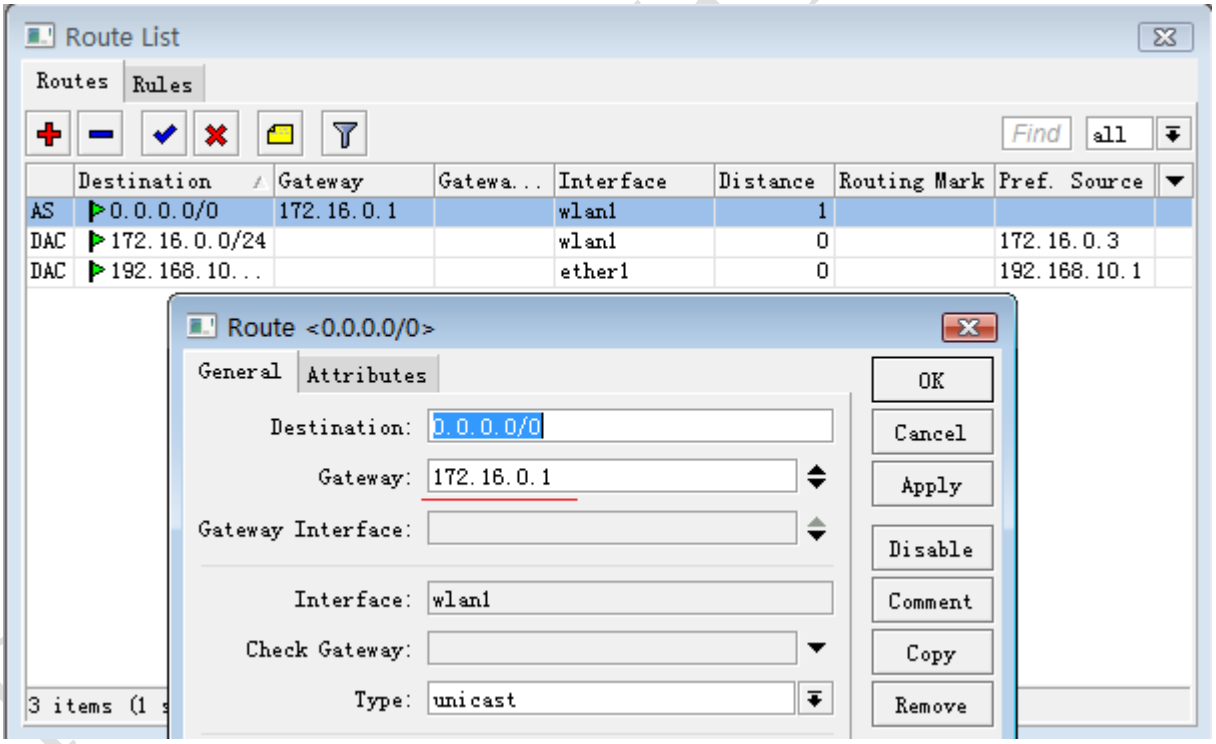
之后进入在 ip address 中添加 IP 地址，配置 IP 地址为 172.16.0.2



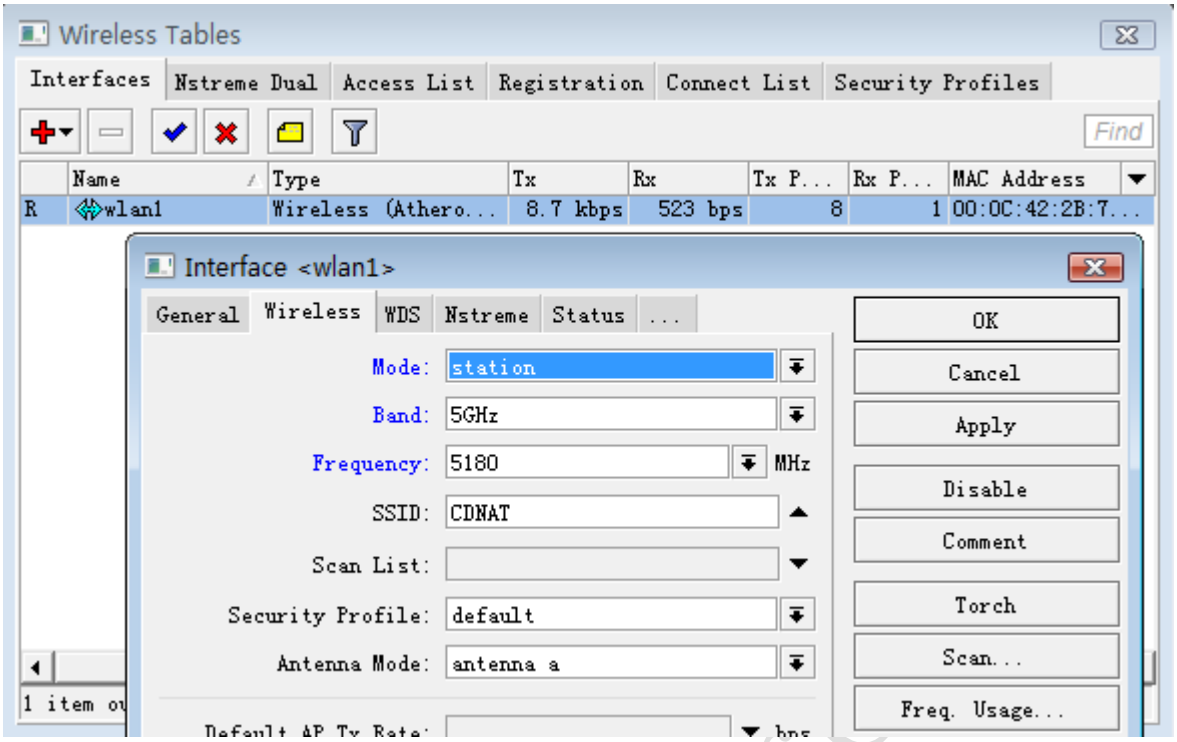
步骤 3：配置 Station 端的 wlan1 和 ether1 的 IP 地址，分别为 172.16.0.3 和 192.168.10.1，并配置网关地址 172.16.0.1。



在 ip route 中配置网关地址：

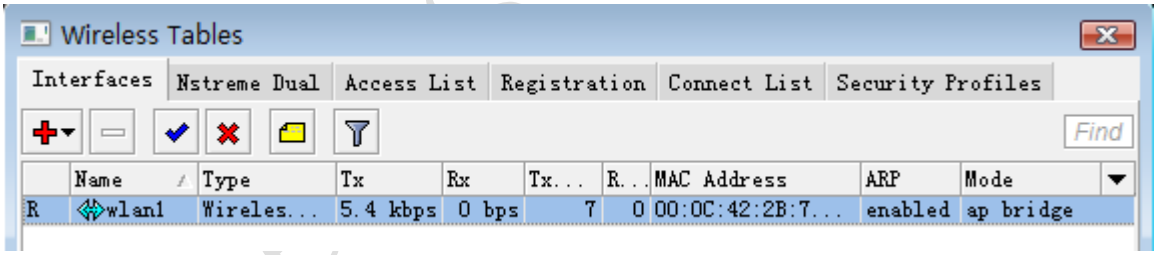


步骤 4：配置 Station 端的无线参数，设置 mode=station，Band=5G，SSID=CDNAT ：

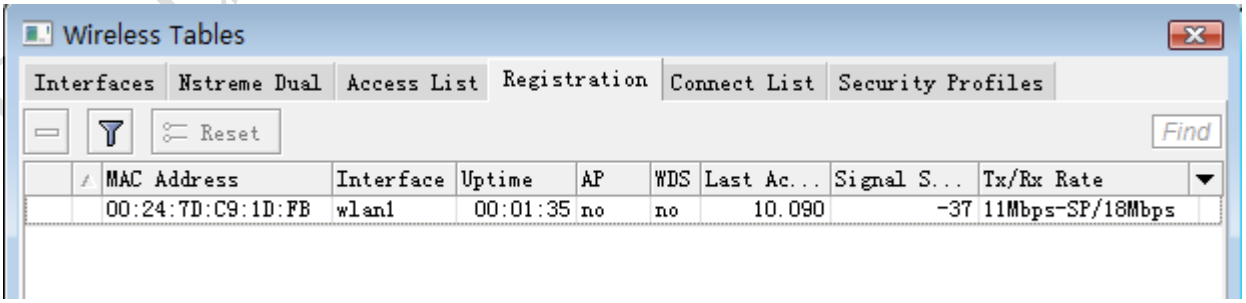


*注：*当 Mode 设置为 station 或者 station-wds 情况下，Band 和 SSID 与 AP 配置相同，Frequency 会自动适应 AP 的频率参数。

这样 AP-Bridge to Station 配置完成，，当连接后可以查看 wlan1 的无线状态会在项目最前面显示“R”运行：



无线注册信号与参数：



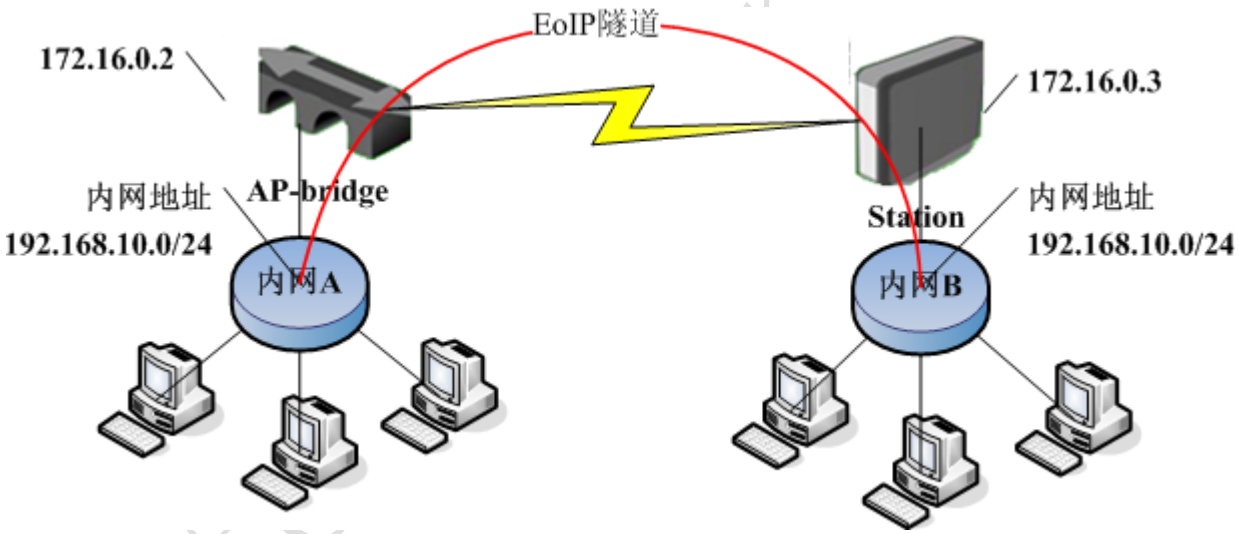
最后通过 Terminal 在终端使用 ping 命令检测 172.16.0.1，是否连接正常：

```
Terminal
  MikroTik RouterOS 3.16 (c) 1999-2008       http://www.mikrotik.com/

[admin@CDNAT] > ping 172.16.0.1
172.16.0.1 64 byte ping: ttl=64 time=1 ms
172.16.0.1 64 byte ping: ttl=64 time=1 ms
172.16.0.1 64 byte ping: ttl=64 time=1 ms
172.16.0.1 64 byte ping: ttl=64 time=1 ms
172.16.0.1 64 byte ping: ttl=64 time=1 ms
172.16.0.1 64 byte ping: ttl=64 time=1 ms
172.16.0.1 64 byte ping: ttl=64 time=1 ms
172.16.0.1 64 byte ping: ttl=64 time=1 ms
172.16.0.1 64 byte ping: ttl=64 time=4 ms
```

6.3 AP-Bridge to Station 的 EoIP 桥接模式

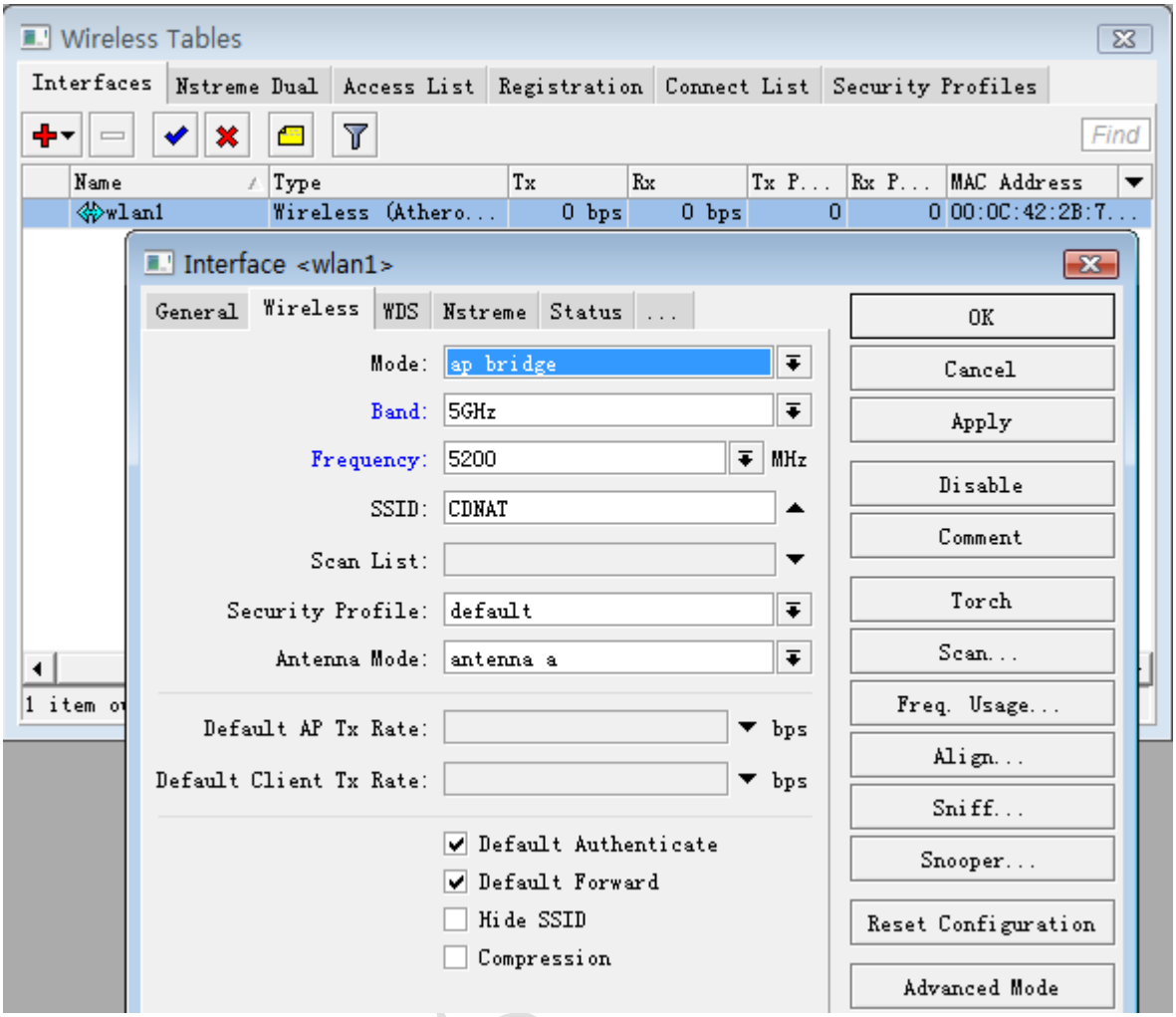
基于 AP-Bridge to Station 的 EoIP（基于 IP 传输的以太网协议）桥接，主要是早期为解决 RouterOS 不能实现桥接问题而设置的，通过 EoIP 我们可以将采用路由模式的 AP 的设备，在 IP 地址建立的 EoIP 隧道中透传二层数据。如下图：

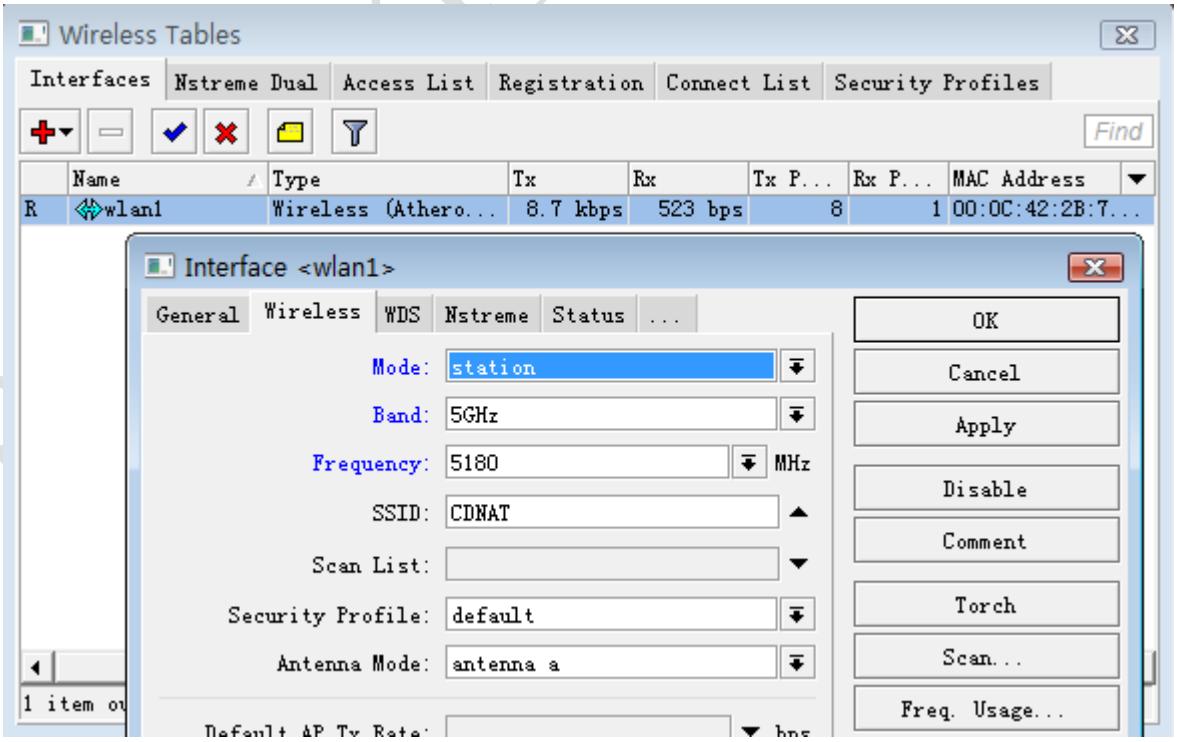


这里我们以之前的 AP-Bridge to Station 事例为基础，建立 EoIP 隧道：

1、 完成 AP-Bridge to Station 的无线连接和 IP 地址配置
2、 配置 AP-bridge 端的 EoIP 隧道参数，并配置 bridge 参数
3、 配置 station 端的 EoIP 隧道参数，并配置 bridge 参数

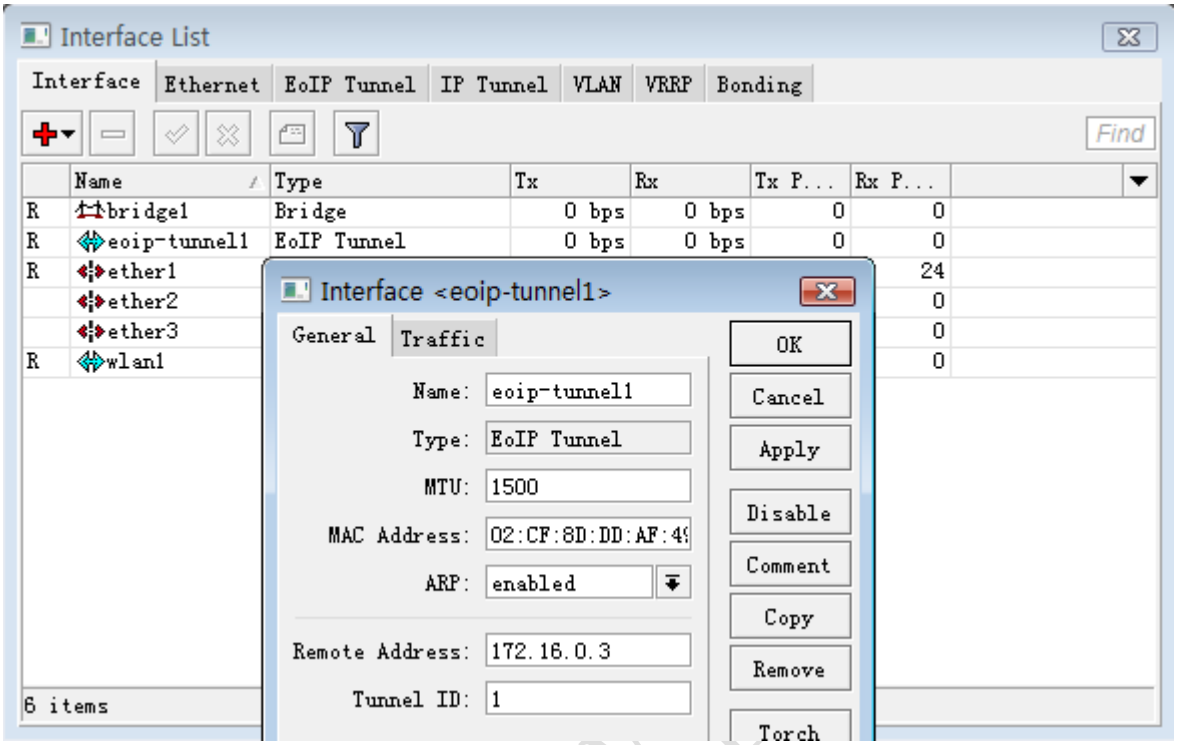
步骤 1：配置 AP-bridge 和 Station 的无线连接，这里的 AP-bridge 和 Station 配置和之前的相同



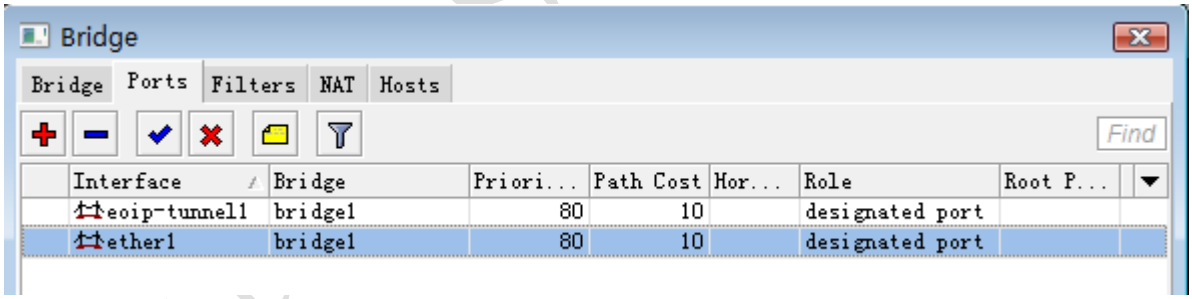


同样在 ip address 给 AP-bridge 的 wlan1 设置 172.16.0.2 的 IP 地址，Station 的 wlan1 设置 172.16.0.3 的 IP 地址。

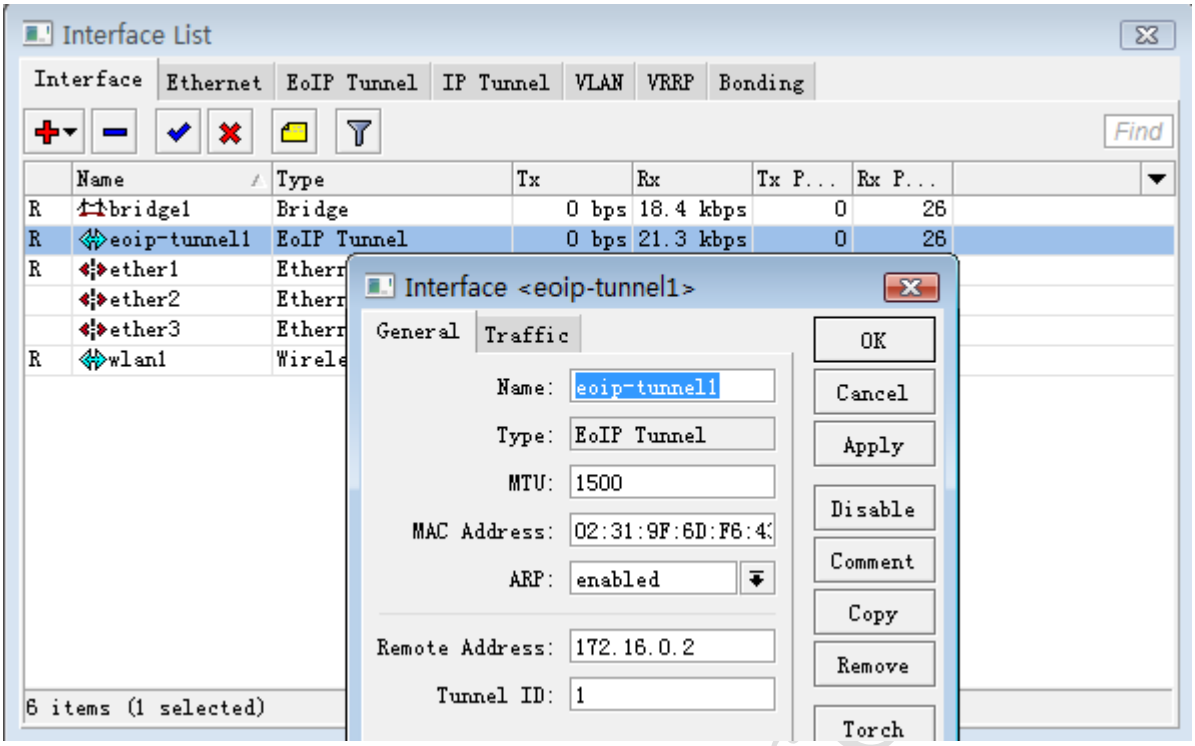
步骤 2：配置 AP-bridge 的 EoIP 隧道，填写对方的 Station 设备的 IP 地址 172.16.0.3，并设置相同的 Tunnel ID，这里我们设置为“1”



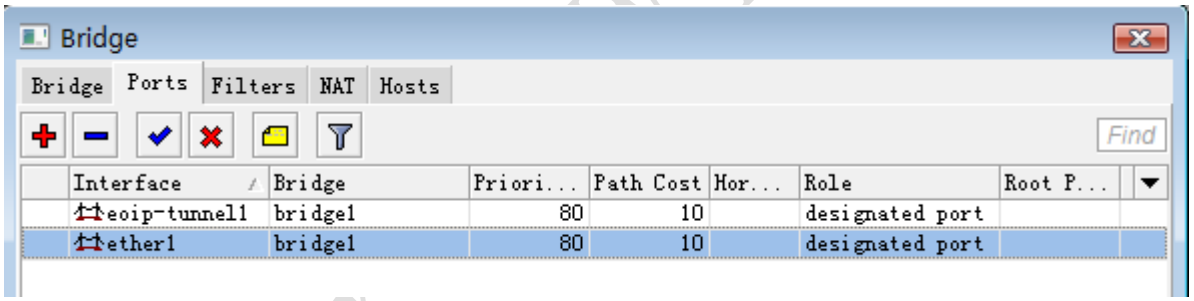
这里 bridge 这是与之前的事例有点不同，我们只需要将 AP-bridge 端的 ether1 和设置好的 eoip-tunnel1 添加进入桥中



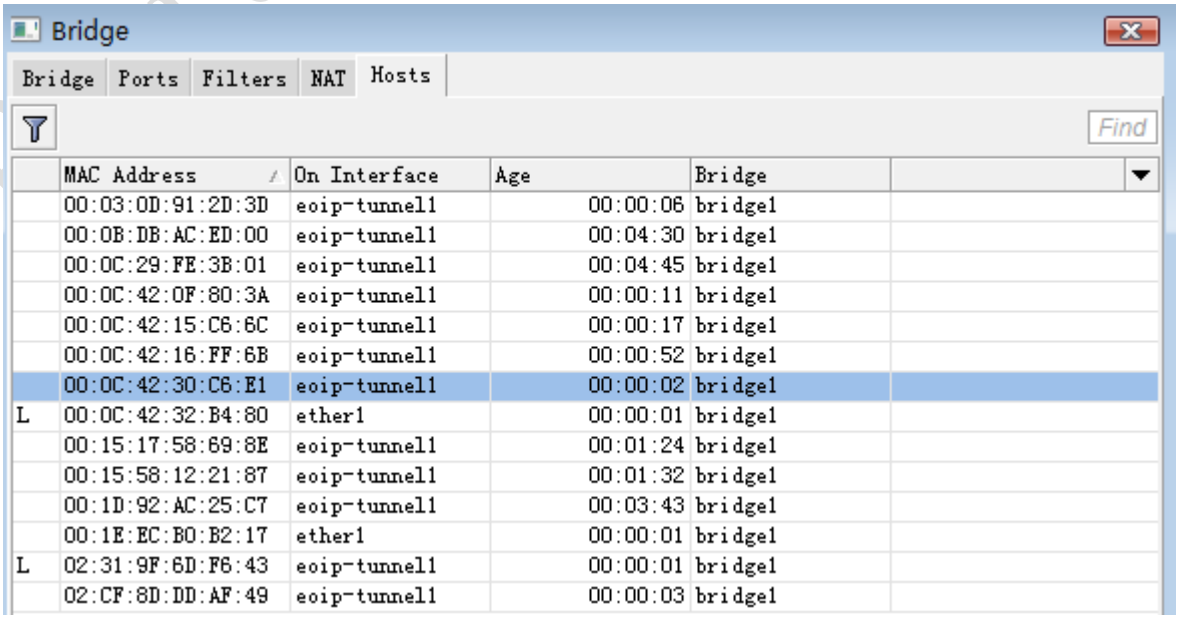
步骤 3：配置 Station 的 EoIP 隧道，填写对方的 AP-bridge 设备的 IP 地址 172.16.0.2，并设置相同的 Tunnel ID，这里我们设置为“1”



同样在 bridge 中添加 ether1 和 eoip-tunnel1 到 bridge1 里

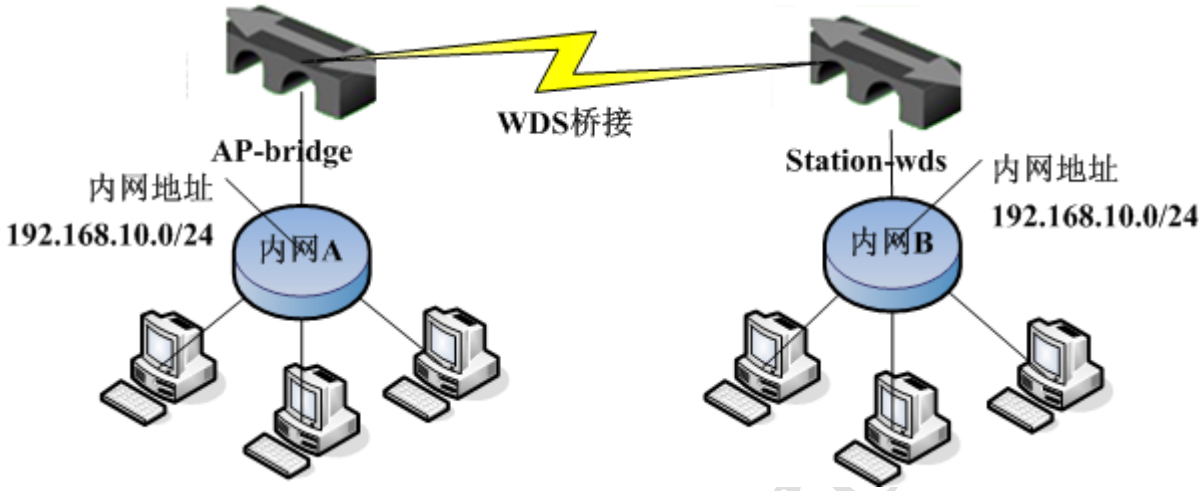


通过查看 bridge 的 Hosts 列表，可以看到 brdige 学习到在 on-interface 项目中有多个 eoip-tunnel1 的 MAC 地址，最前方 L 标示的是 Local 本地的接口。



6.4 AP-Bridge to Station-WDS 桥接模式

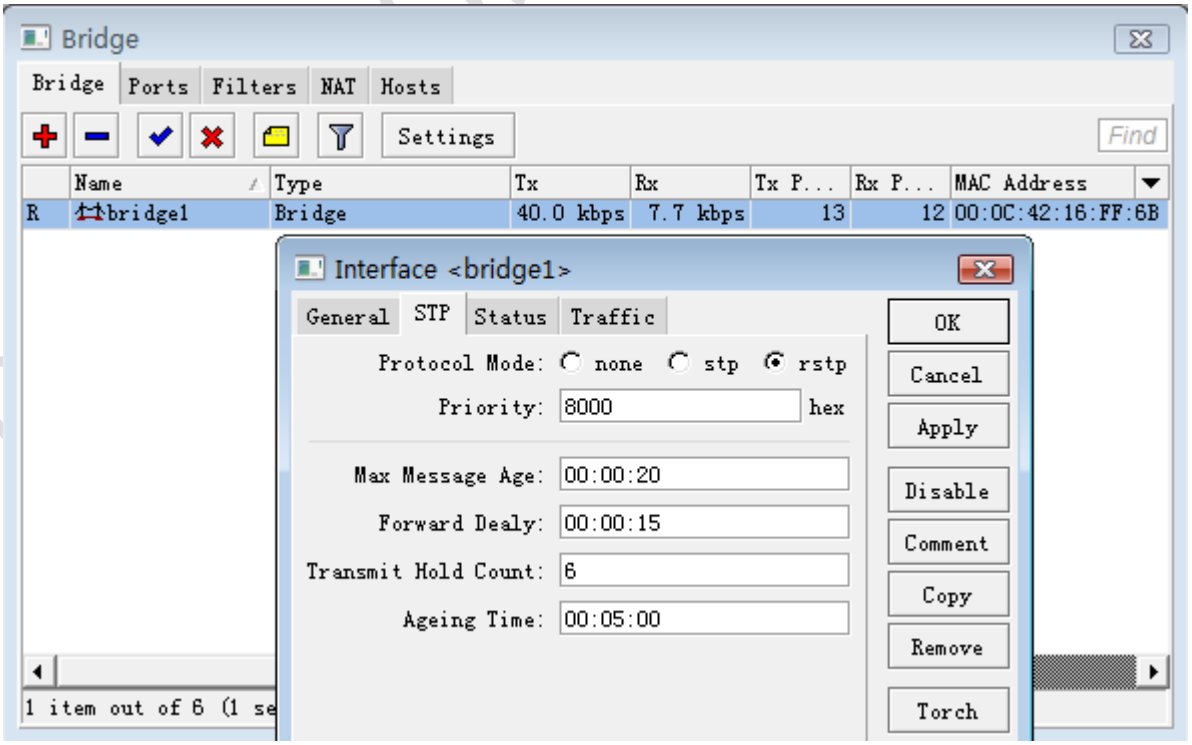
点对点方最常见的的就是 WDS 桥接模式，我们可以采用 ap-bridge 或者 birdge 方式，在这里我们推荐使用 ap-bridge 与 station-wds 的桥接方式，如下图：



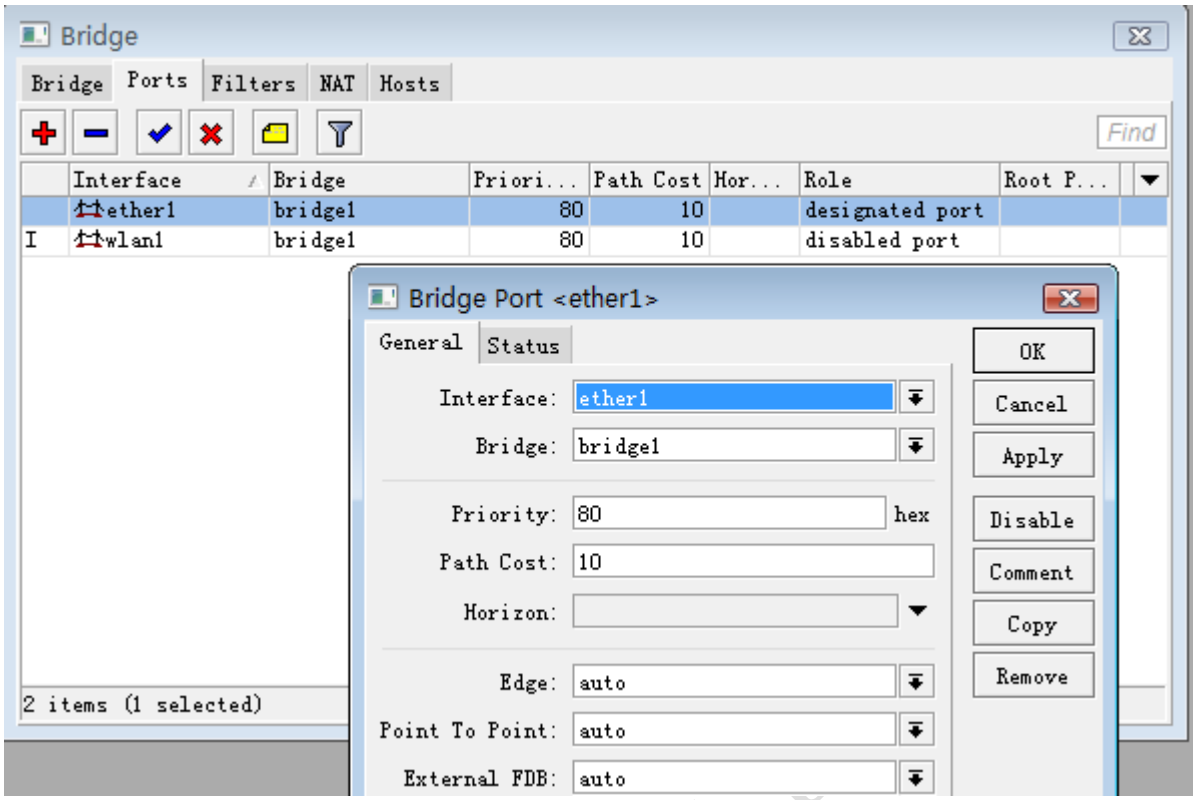
设置在 ap-bridge 和 station-wds 模式的我们分以下步骤：

1、 在 ap-bridge 和 station-wds 中添加 bridge，定义 bridge 的接口，并分配管理的 IP 地址
2、 配置 ap-bridge 和 station-wds 的无线参数
3、 检查桥接连接情况

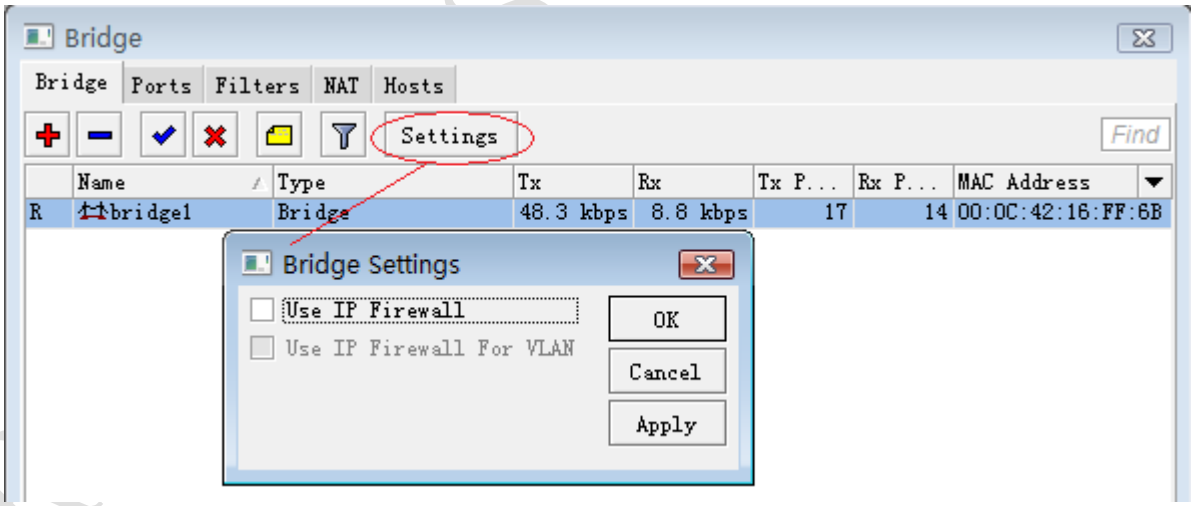
步骤 1：进入 bridge 添加 bridge1 的桥接，通常情况下我会开启 rstp 协议，启用生成树协定：



添加 ether1 和 wlan1 到 brdige1 桥接中，这里在 interface 中分别添加 ether1 和 wlan1 进入 bridge1 中。这样 ether1 和 wlan1 就实现了桥接功能，能实现数据二层的透明传输：



*注：*在 RouterOS3.0 的 bridge 中增加了一个设置选项，是否选择 ip firewall 过滤，如果不使用 ip firewall 过滤路由器的桥接转发速度将提升性能，但如果你要求对无线传输过程中的 IP 数据进行过滤处理，那就需要开启 use-ip-firewall 功能：



注：以上配置操作适用于 ap-bridge 和 station-wds 设备

设置完桥接后我们进入 ip address 给 **ap-bridge 和 station-wds** 的 bridge 配置一个 IP 地址 192.168.10.1/24 和 192.168.10.2/24，用于管理设备和监测用。这样 wlan1 口和 ether1 都能分配到这个地址。命令如下：

ap-bridge 设备

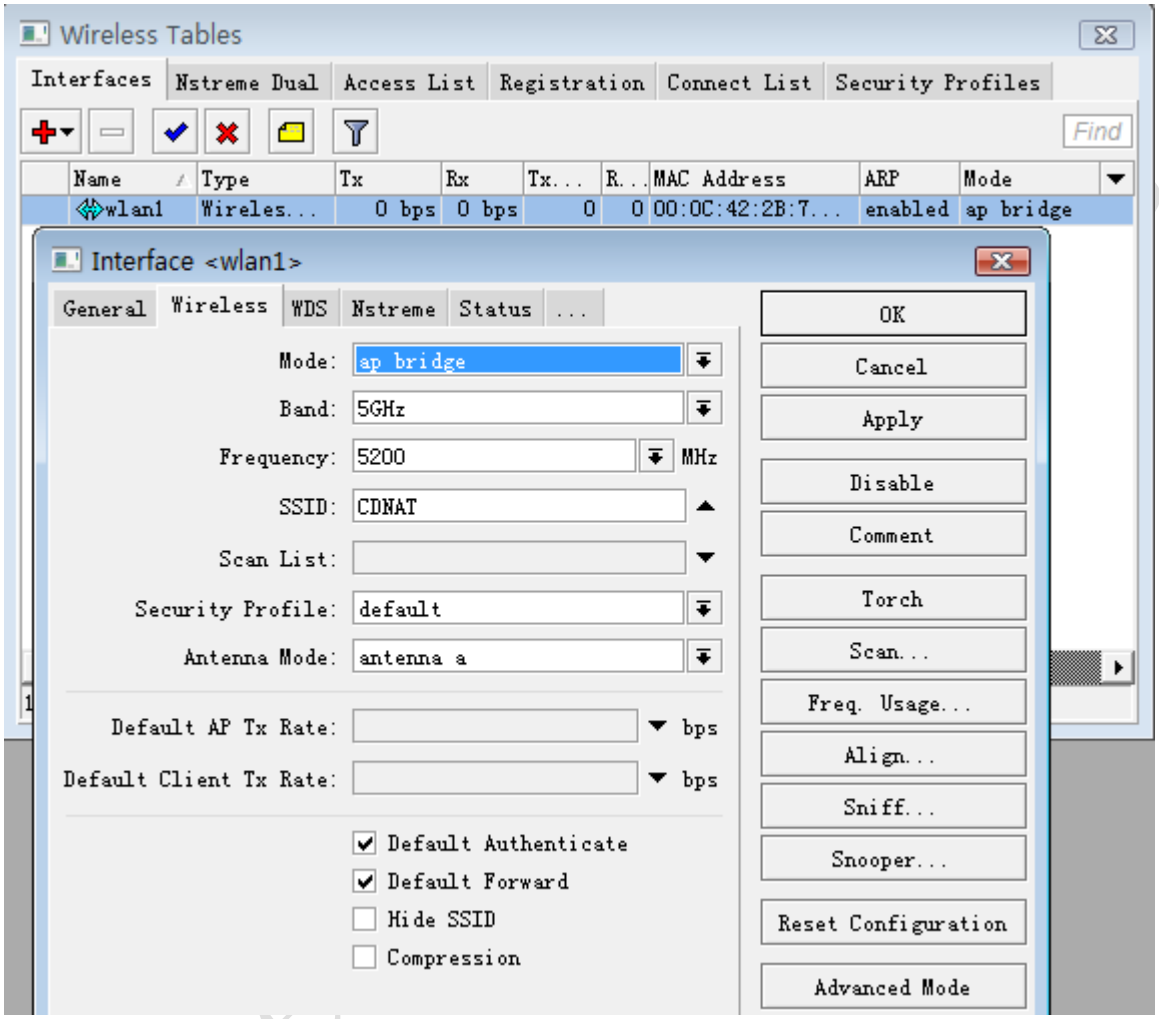
```
/ip address add address=192.168.10.1/24 interface=bridge1
```

station-wds 设备

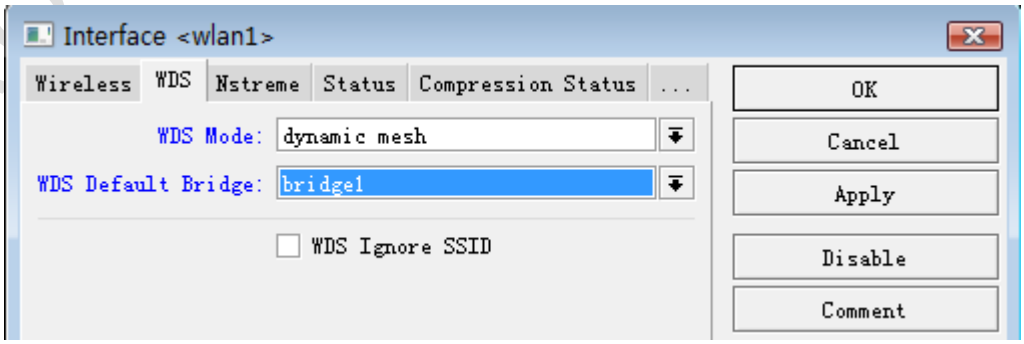
```
/ip address add address=192.168.10.2/24 interface=bridge1
```

步骤 2：桥接和 IP 地址设置好后，现在配置 ap-bridge 和 station-wds 的无线参数。

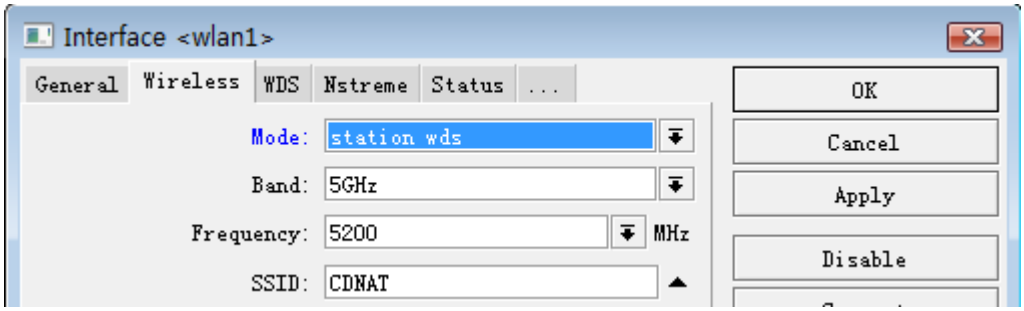
设置 ap-bridge 的无线，这里 mode=ap-bridge，band=5G，frequency=5200，SSID=CDNAT



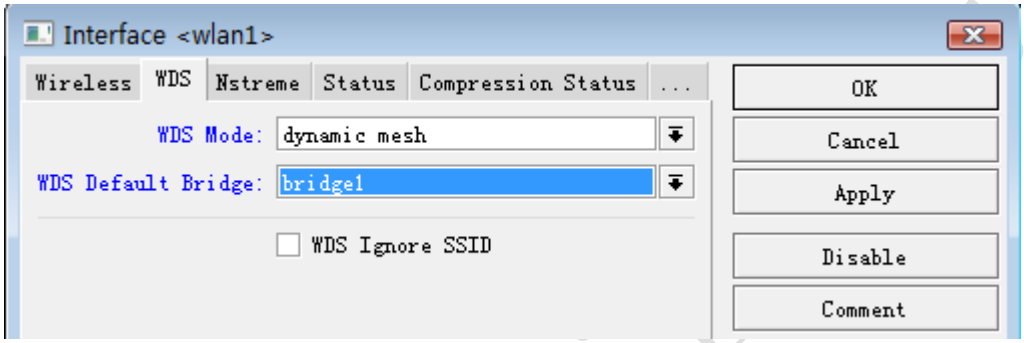
配置 ap-bridge 的 WDS 模式，配置参数 wds-mode=dynamic-mesh（动态方式），wds-default Bridge=bridge1（将连接无线添加到 bridge 中）



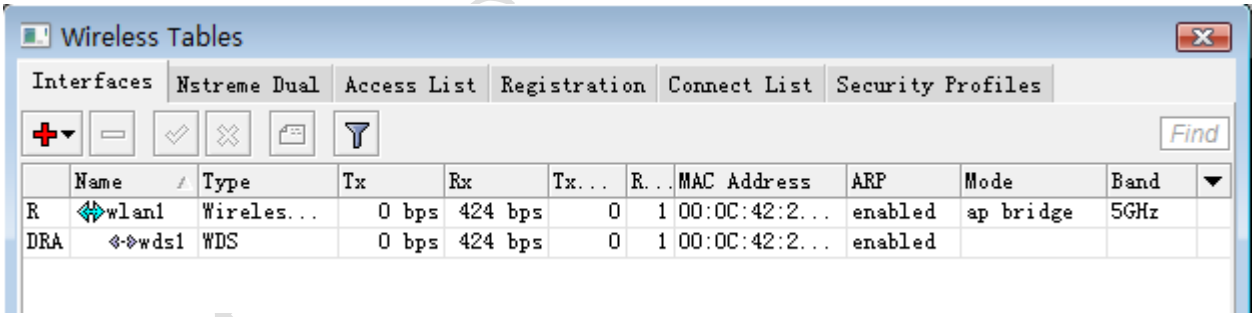
配置 station-wds 端的设置只需要将 Mode=station-wds band=5G，SSID=CDNAT，不需要设置 Frequency 参数，station-wds 在匹配 Band 和 SSID 后会自动搜索：



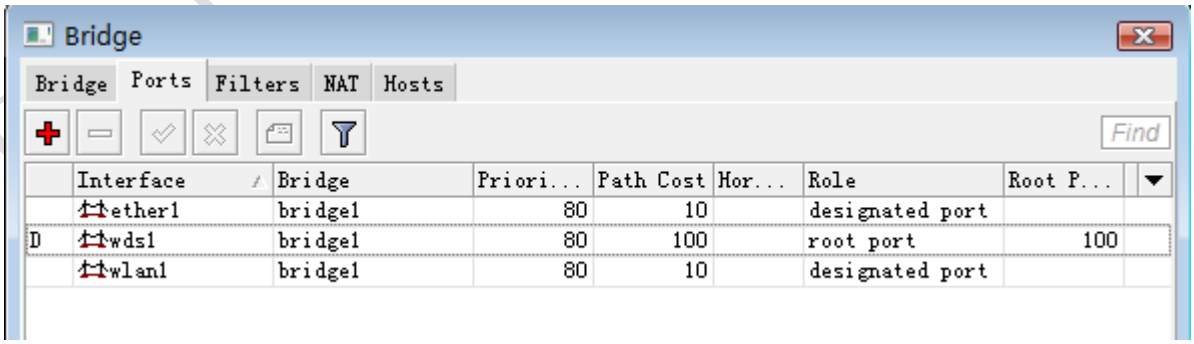
在 station-wds 模式下与 ap-bridge 的 WDS 参数配置相同



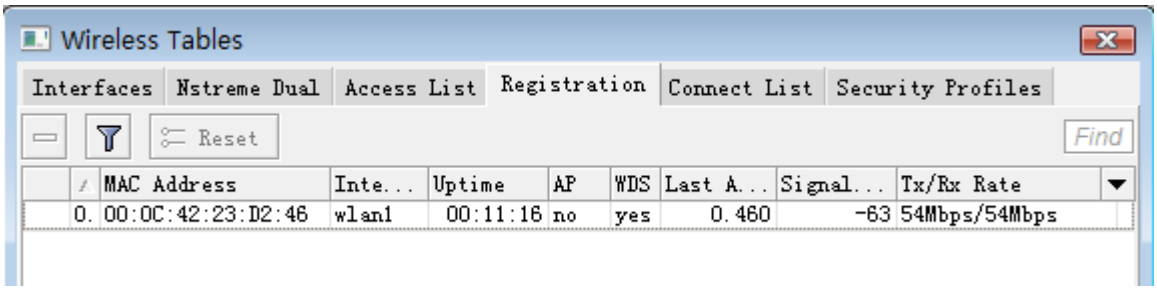
步骤 3：当配置完成后，我们可以通过在 ap-brdige 端的设备查看是否连接，如果正常连接后 ap-bridge 端的 Wireless Tables 下会在 wlan1 前现时“R”，并增加一个 wds1 的无线接口。



在 ap-bridge 下的 Bridge 中可以看到，WDS 模式自动将 wds1 接口添加到 Port 中：



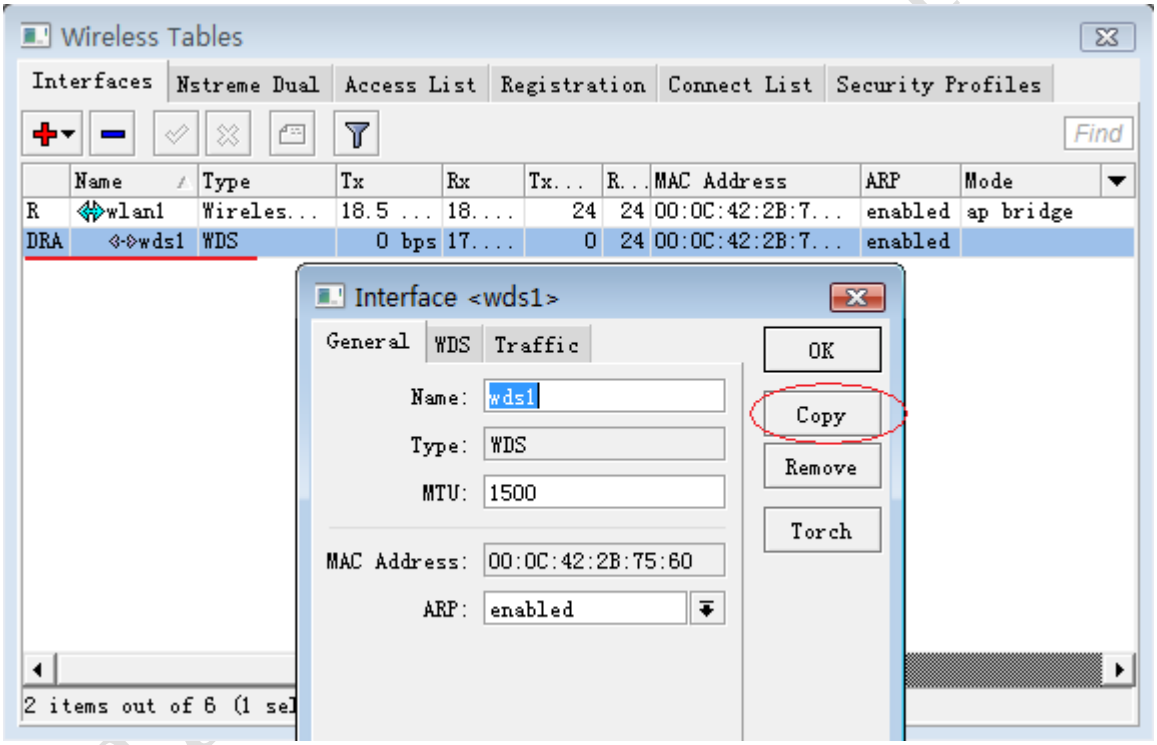
我们可以通过在无线注册指令清单中查看信号强度：



这里显示的是-63，信号能连接使用的最低值在“-88 到-90”，数字越接近正数“1”信号越强。Station-wds 端在连接后会自动适用 ap-bridge 的参数，并正常通信。

6.5 静态的 WDS 模式连接

设置静态的 WDS 连接可以避免其他为允许的无线连接进入我们的网络，保证网络不受到入侵，配置静态



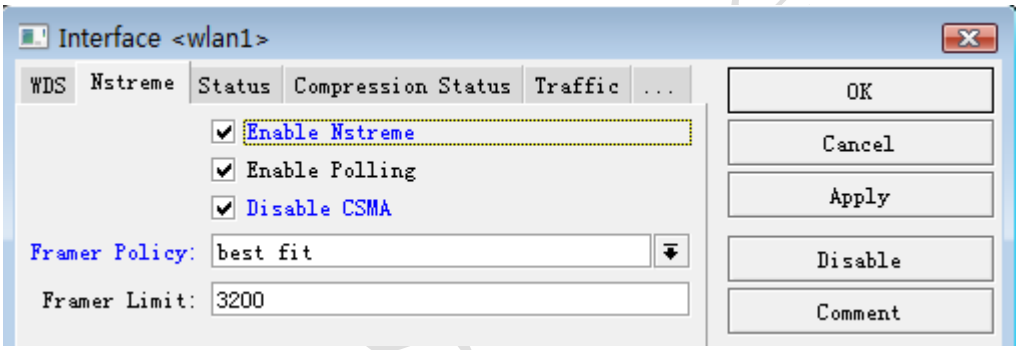
第七章 MikroTik 特有协议与应用

7.1 Nstreme 协议

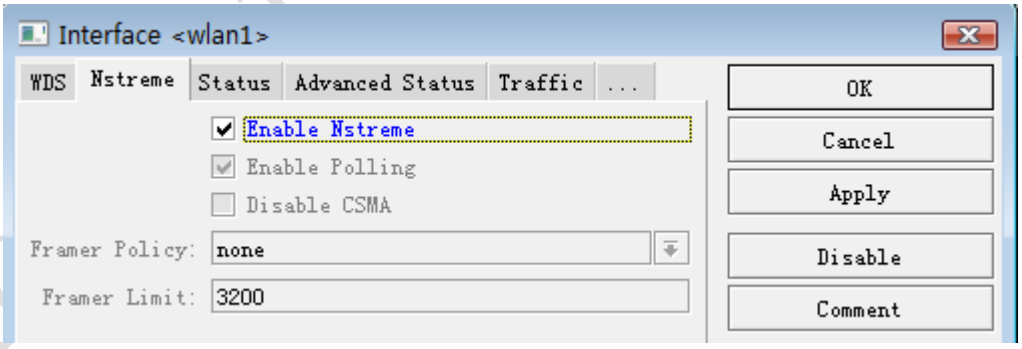
MikroTik 独有的 Nstreme 协议用于长距离的无线传输、增加带宽和提高网络质量的作用。 Nstreme 支持 **ap-bridge/bridge** to **station/station-wds**，不支持 ap-bridge to ap-bridge，所以通常 Nstreme 被用于点对点传输和点对多点传输，不能用于漫游组网。启用 Nstreme 的操作只需要进入无线网卡配置的 Nstreme 选项。

在我们做点对点传输的时候，可以启用 Nsteme 协议，提高网络安全和传输质量。这里我们基于点对点的 ap-bridge 和 station/station-wds 模式是配置为例，启用 Nsteme 协议。

ap-bridge 的配置，启用 Nsteme 协议，禁止 CSMA 协议，启用 Polling 轮询方式，并选择帧策略为 best fit 的帧传输，将帧封装为 3200，最大可以到 4000：



Station 和 station-wds 的配置，只需要启用即可，其他参数会自适应 AP 端：



Nstreme 协议必须双方同时启用，才能正常连接，所以配置 Nstreme 时一定要注意。

7.2 Nstreme Version 2 协议（NV2）

Nv2 协议由 MikroTik 独立基于 Atheros 802.11 无线芯片开发无线通讯技术，Nv2 是基于 TDMA (Time Division Multiple Access)介质访问技术替换 CSMA (Carrier Sense Multiple Access) 介质访问技术，用于普通的 802.11 设备。

TDMA 介质访问技术解决了隐藏节点问题，提高了媒体利用率，从而提高吞吐量和降低延迟，特别是在点对多点网络中。Nv2 支持 Atheros 802.11n 芯片，而老的 802.11a/b/g 芯片，从 AR5212 开始支持，不支持 AR5211 和 AR5210 芯片。

介质访问在 Nv2 网络中，是由 Nv2 AP 控制，Nv2 AP 将时间划分为固定大小的"周期"，这些"周期"根据 AP 和客户端的队列情况，动态划分为下行(从 AP 发送到客户端的数据)和上行(从客户端发送到 AP 的数据)部分。上行时间根据连接的客户机对带宽的需求进一步划分。在每个周期开始时，AP 广播计划告诉客户端他们应该在什么时候传输以及他们可以使用的时间。

为了允许新客户端连接，Nv2 AP 定期为"未指定的"客户端分配上行时间——然后新客户端使用这个时间间隔启动对 AP 的注册。然后 AP 估计与客户端之间的传输延迟，并开始定期为该客户机调度上行时间，以完成注册并从客户端接收数据。

Nv2 实现了基于每个客户端的动态速率选择和数据传输的 ARQ。这支持跨 Nv2 链路的可靠通信。

对于 QoS, Nv2 使用内置的缺省 QoS 调度程序实现了定义变量的优先级队列，该调度程序可以与基于防火墙 mangle 或使用 VLAN 优先级，以及 MPLS EXP 参数在网络上传输的优先级信息

Nv2 vs 802.11 区别

- **介质访问由 AP 预先分配** – 这样消除了隐藏的节点问题，并允许实现集中的媒体访问策略，AP 规划每个客户端使用的时间，并可以根据某些策略为客户端分配时间，而不是每个设备相互竞争媒体访问
- **减少传输延迟开销**– 在 Nv2 中没有每帧 ACK 请求，这样能有效的提示吞吐量，特别是在长距离链路上，数据帧和跟随 ACK 帧传输延迟会显著降低在介质中的使用率。
- 减少每帧开销– Nv2 实现帧聚合和分段发送，以最大限度地分配介质使用率，并减少每帧的开销。

Nv2 vs Nstreme 区别

- **减少轮询开销** – Nv2 协议不再是轮询每个客户的方式， Nv2 AP 广播上行调度，将时间分配给多个客户端，这样被称为"组轮询"– 不会浪费轮询每个客户端的时间，为实际的数据传输留下更多的时间。这样提高了吞吐量，特别是在点对多点配置下。
- **减少传输延迟开销** – Nv2 不会轮询每个客户端，这允许根据到客户端的估计距离(传输延迟)创建上行调度，从而使介质的使用率最大。这样提高了吞吐量，特别是在点对多点配置下。
- **更好地控制延迟** – 减少开销、调接周期长度和 QoS 策略等多种方式控制无线网络的延迟。

从 RouterOS v5.0beta5 开始，可以在 wireless 菜单下配置 Nv2，Nv2 协议限制了 511 个客户端

Nv2 参数

- **nv2-qos** 设置数据报的优先级机制，首先数据将从优先级高的开始发送，这时低队列数据，要等到 0 队列优先到达目的地为止才能发送。当高优先级队列数据连接满载，低优先级数据不能被发送，使用这个功能在 AP 上非常有效。
- **frame-priority –** 能在 mangle 里手动设置
- **default –** 默认将小包接收设置为最低延迟优先级
- **nv2-cell-radius** (默认 值: *30*); 这个设置会影响连接时间间隔大小，AP 分配给开始连接客户端估

算客户端距离的时间周期值。当时间太小，远程的客户的可能会出现连接问题，并且或由于“ranging timeout”断开连结错误，为了保持最高性能，没有必要的情况下，不要增加这个值。
  - on AP: 最远的客户端距离，单位 km
  - on station: 无作用
- **tdma-period-size** (默认 值: 2) 指定 TDMA 周期为毫秒。有助于较长距离的连结，能略微增加带宽，当然延迟也同样会增加

Nv2 可能问题

在长距离通过 tdma-period-size 可以增加吞吐量，每个“period”，离开 AP 后不会使用到的传输时间(等于一个往返时间– 时间在帧被发送和从客户端接收到), 它是用于确认客户端能接收到从 AP 发出的一个帧，即较长的距离，较长的时间周期不会被使用。

例如，AP 与客户端距离 30km，帧直接发送到需要 100us，接收一个往返需要大约 200us，tdma-period-size 默认是 2ms，即 10%的时间没有被使用（1000us=1ms，即 0.2ms/2ms=10%），当 tdma-period-size 增加到 4ms，仅有 5%的时间没有被使用，如果增加到 60km，往返时间为 400us，未使用时间 20%，这时 tdma-period-size 为 2ms，4ms 时为 10%，更大的 tdma-period-size 值增加连接延迟。

Nv2 兼容性

仅 RouterOS 设备支持 Nv2 技术，在搜索时，仅有 RouterOS 设备能发现支持 Nv2 技术的 AP。Nv2 网络将干扰其他网络相同频道的 AP，也同样包括附近的 Nv2，当 RouterOS 启用 Nv2 后将不能连接其他任何基于 TDMA 网络。

Wireless-protocol 参数

从 5.0rc1 开始加入了新的 wireless 设置参数 wireless-protocol，根据无线网络环境配置不同协议，以及需要兼容的模式，如下表：

值	AP	client
unspecified	建立基于老版本的 nstreme 或者 802.11	连接到老版本的 nstreme 或者 802.11
any	如同 unspecified	搜索所有匹配的网络，不论协议。
802.11	建立 802.11	只能连接到标准的 802.11 网络
nstreme	建立 Nstreme	只能连接到 Nstreme
nv2	建立 NV2	只能连接到 NV2
nv2-nstreme-802.11	建立 NV2	搜索 Nv2 网络，如果找到有适当的网络，并连接。 否则搜索 Nstreme 网络，如果找到有适当的网络，并连接。 否则搜索 802.11 网络，如果找到有适当的网络，并连接。
nv2-nstreme	建立 NV2	搜索 Nv2 网络，如果找到有适当的网络，并连接。 否则搜索 Nstreme 网络，如果找到有适当的网络，并连接。

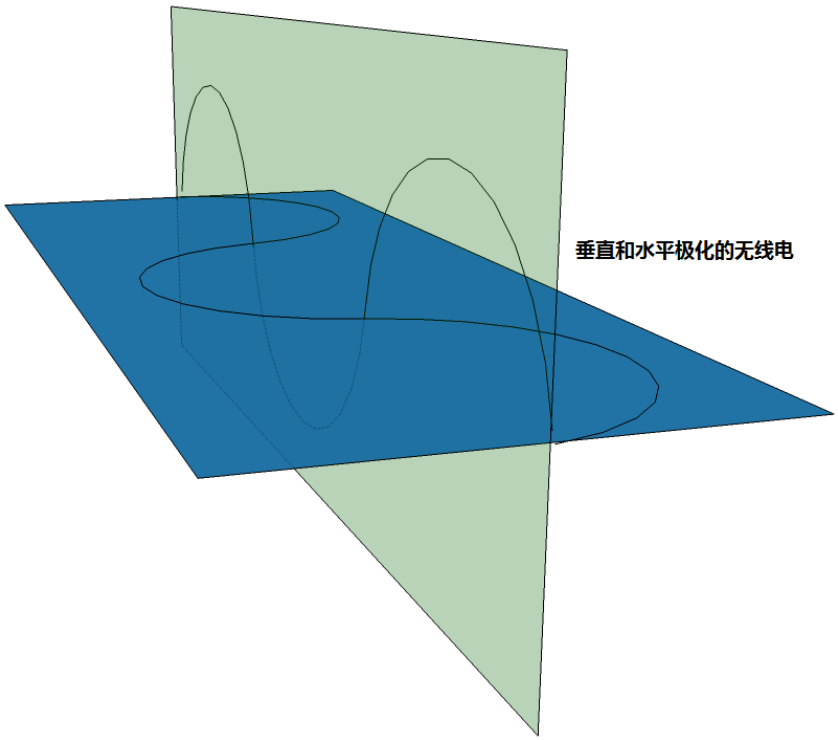
注意：**wireless-protocol** 值设置 **Nv2-nstreme-802.11** 指定某种混杂模式或某种类型协议，这样当客户机连接的网络协议发生更改时，这些值可以简化客户机配置。使用这些参数值可以帮助将网络迁移到 Nv2 协议连接。

7.3 配置 802.11n 的 Nv2 协议

要发挥 802.11n 首先要使用 MIMO 技术，即在前面介绍的 MIMO 技术，RouterOS 通过 Nv2 协议优化了 802.11n 协议的传输性能选择设备。

首先我们需要选择 v5.0 以上的版本，最好选择 RB400、RB700 或者 RB800 的设备，无线网卡使用 AtherosAR9000 系列，支且持 2x2 的无线网卡，并采用双极化天线，双极化天线。

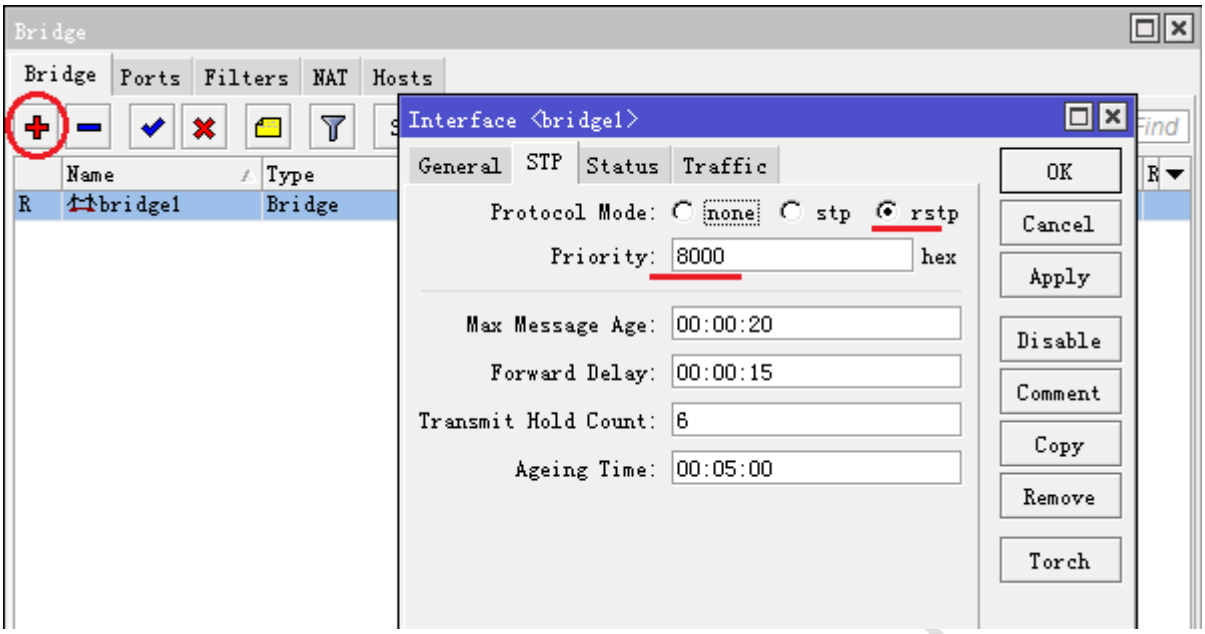
双极化天线采用垂直和水平极化方式传输无线信号，如下图：



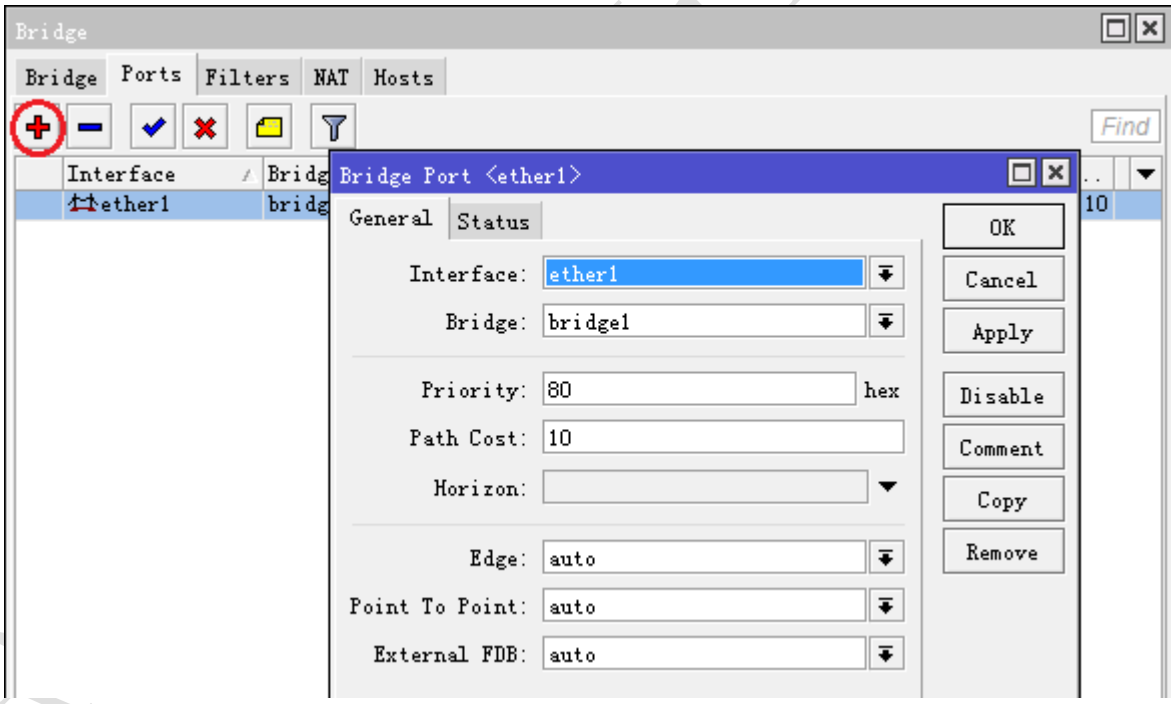
AP-bridge 配置：

首先设置两个 RouterBOARD 的 bridge 桥接，两个设备的 bridge 配置基本相同，只是 STP 里的 priority 参数不同，以下的 bridge 配置 2 台通用。

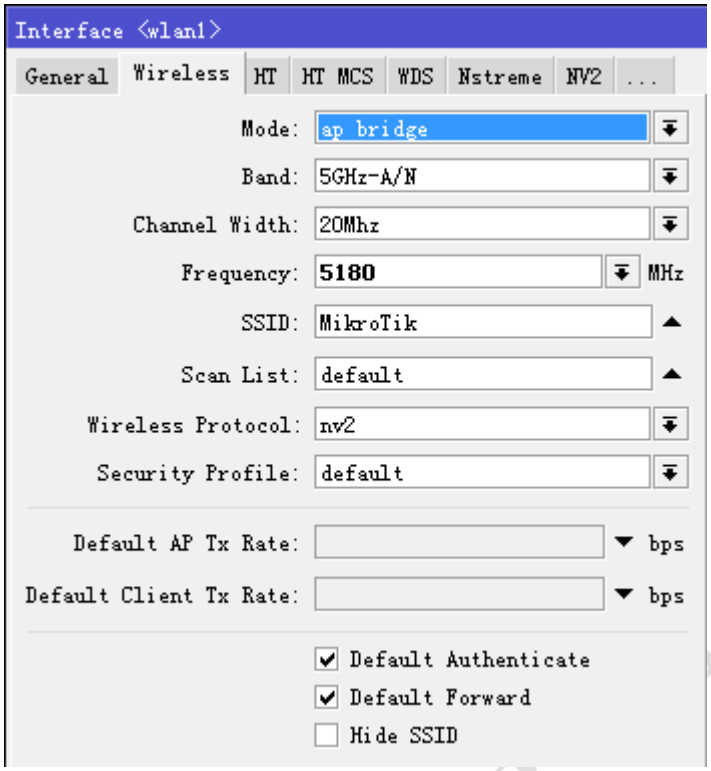
进入 bridge 菜单，添加一个 bridge1，并设置 STP 参数，选择模式为 rstp，两台设备的 priority 为 8000：



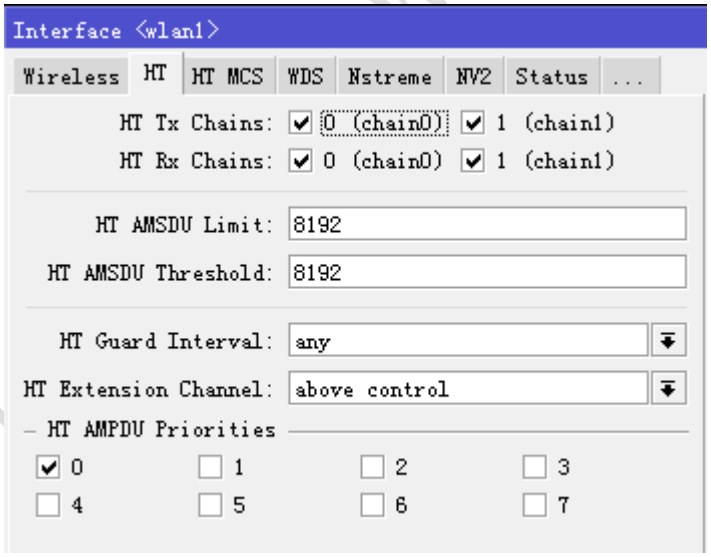
进入 port 标签，将 ether1 添加到 bridge1 里，这里我们只添加 ether1 的网卡，wlan 接口可以不用添加，在后面的无线 wds 配置，由 WDS 配置动态添加



无线网卡为 R52n，采用 5GHz-A/N，SSID 为 MikroTik，wireless-protocol 使用 nv2

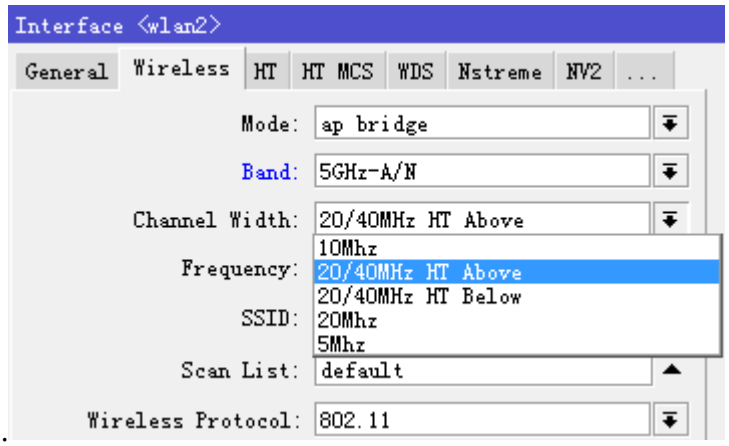


HT 设置，链路通道全部开启，选择 HT-Extension-Channel=above-control，在 v5.3 版本前的设置

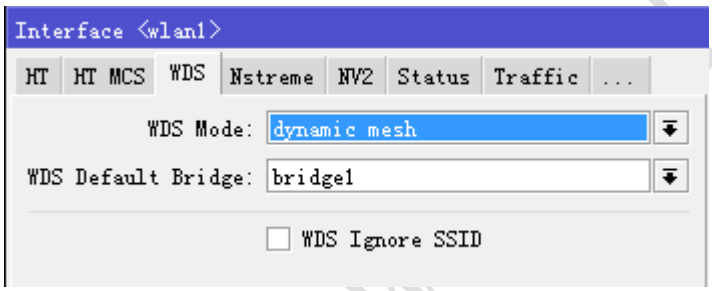


注：在 5.3 版本后 11n 的 HT Extension-Channel 选项变动

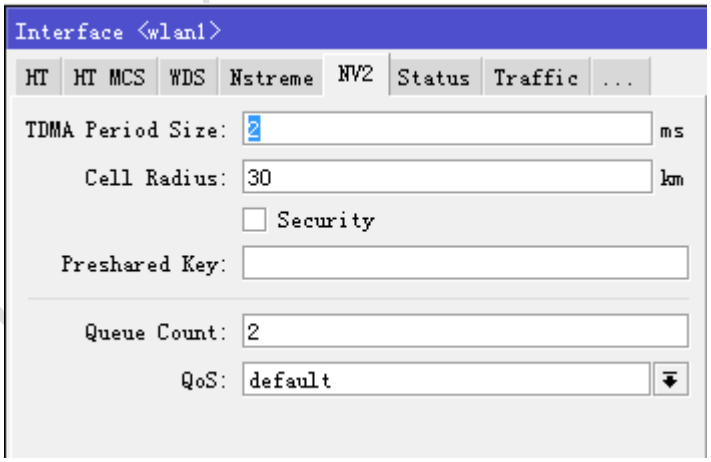
为了更简单的配置参数，避免选项在不同配置菜单下，从 v5.3 版本开始将 **ht-extension-channels** 融合到 **channel-width** 选项，以前被称为 40MHz，现在被称为 40MHz-turbo（非 11n 的卡），并在 802.11n 卡的 HT extension channel 被称为“20/40MHz Above”和“20/40MHz Below”，而且他们现在仅能在 channel-width 的选项中获取，如下图：



WDS 配置，选择 WDS 模式为动态 mesh，并设置默认桥接为 bridge1，这里会自动将 WDS 连接添加到刚才的 bridge1 port 中：

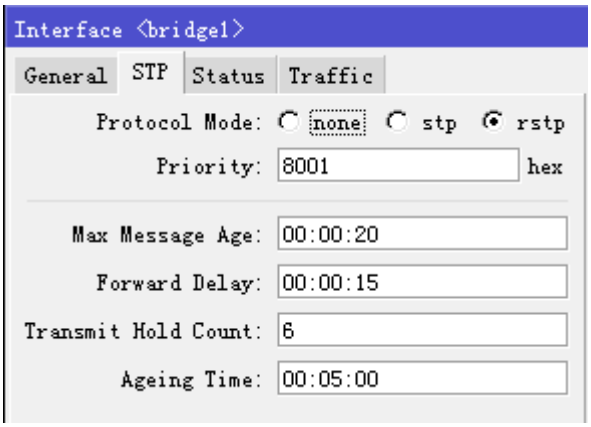


NV2 卷标下的配置默认即可！



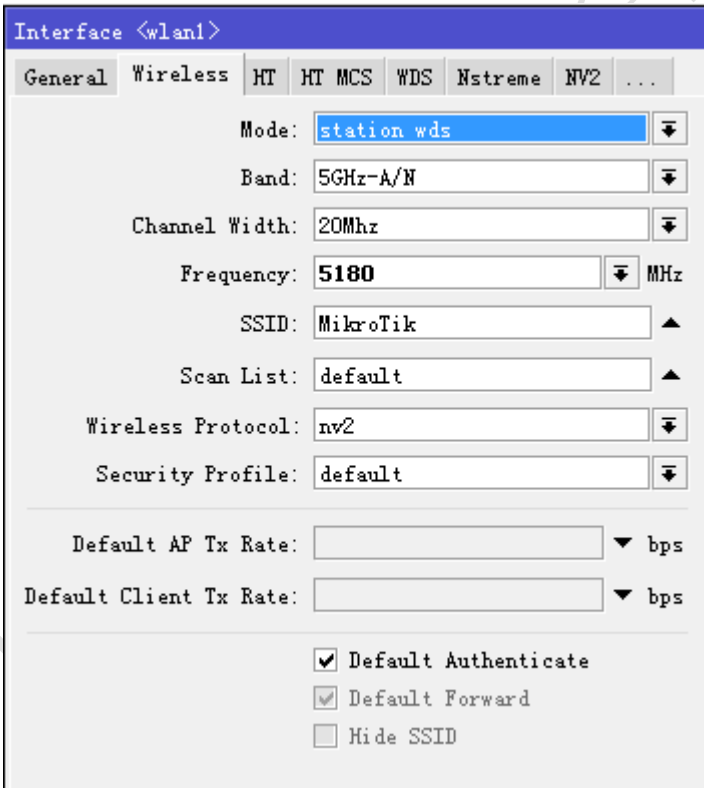
Station 配置

之前的 bridge 配置同上，不在多讲解，只是 rstp 的 priority=8001，以区别 ap-bridge 的 STP 优先级参数

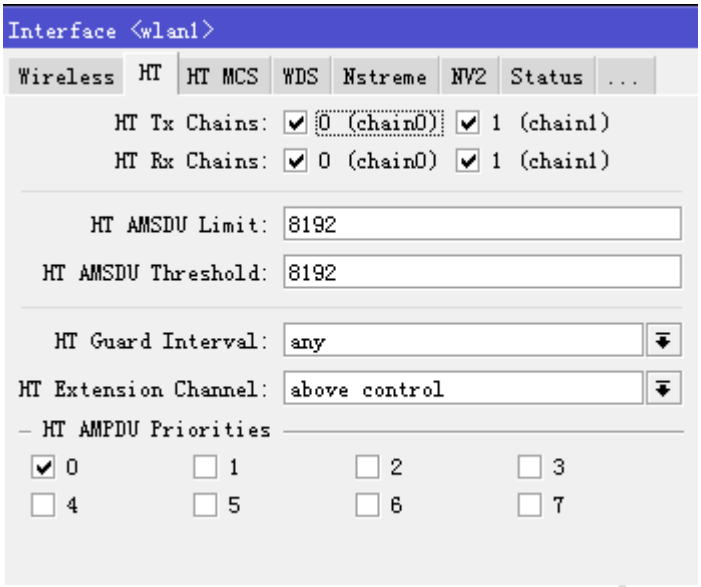


配置 station-wds

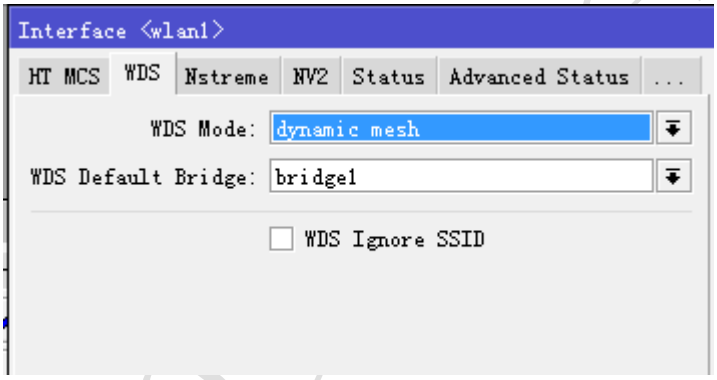
设置 mode 为 station-wds，同样选择 5GHz-A/N，SSID 为 MikroTik，并选择 wireless-protocol 为 nv2



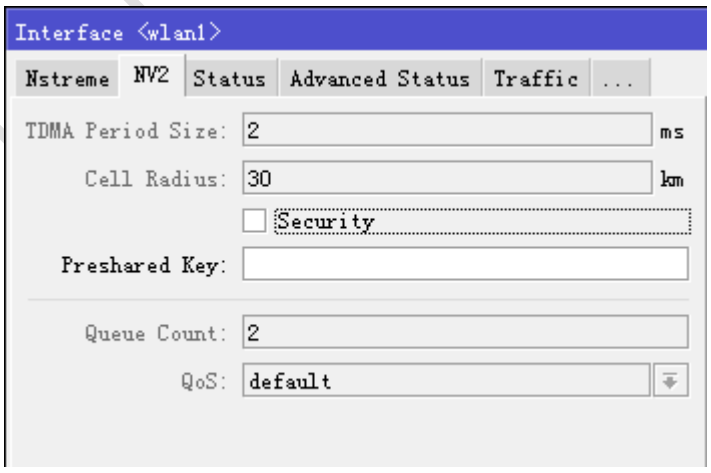
对应的 11n 的 HT 参数，将对应的 tx 和 rx 连结全部打开：



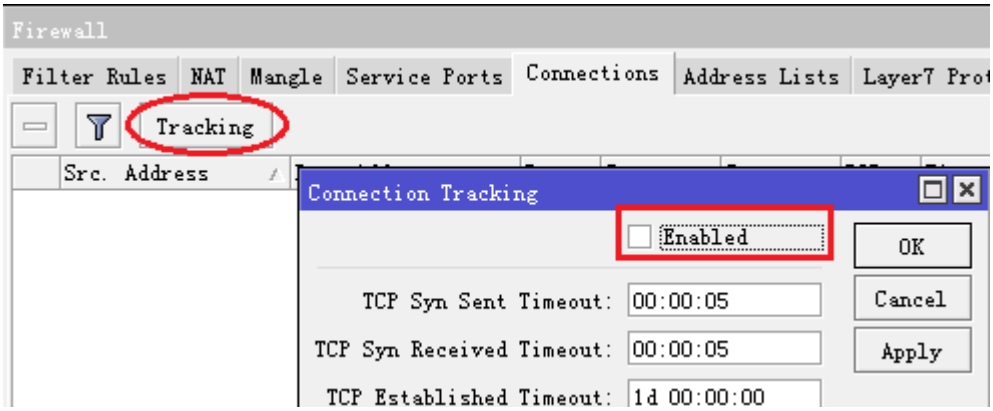
设置 WDS 模式：



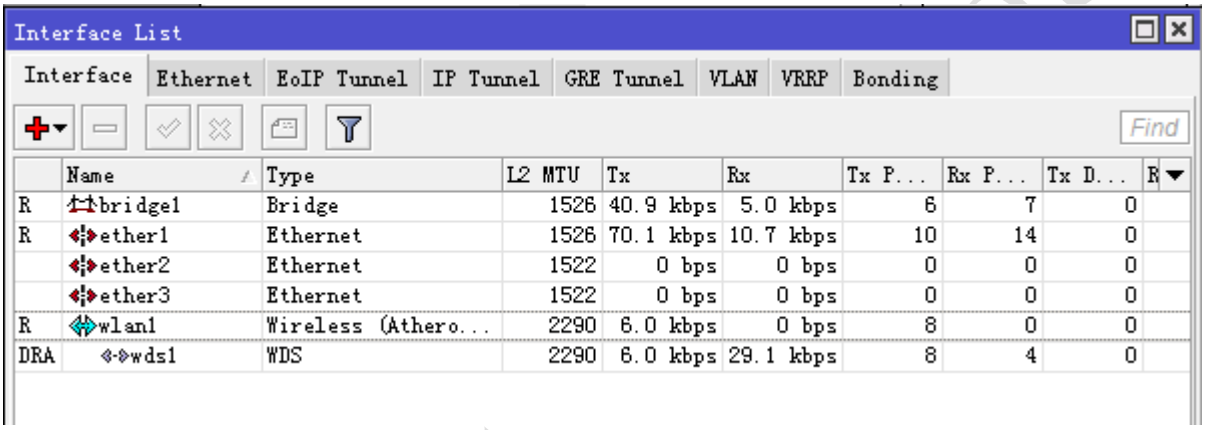
NV2 参数预设设置，也可以根据 AP 的 security 参数选择加密方式



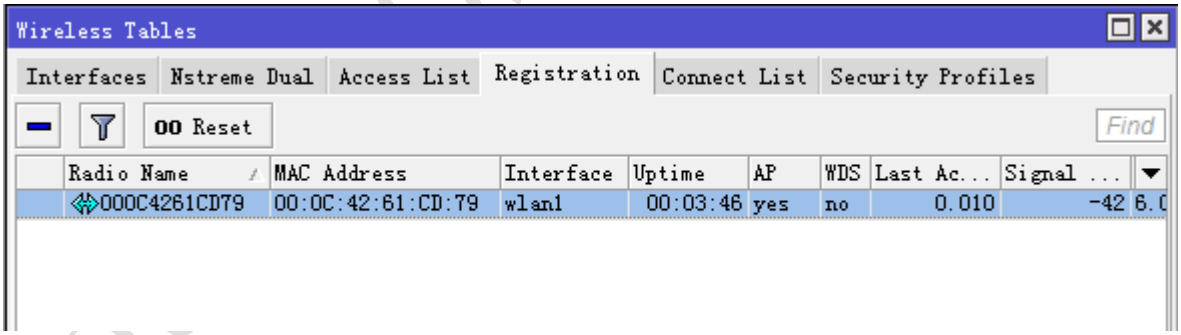
最后优化两台设备性能，我们将 nat 连接跟踪关闭，因为不需要使用到 nat 功能，减小 CPU 开销



连接完成：



信号强度



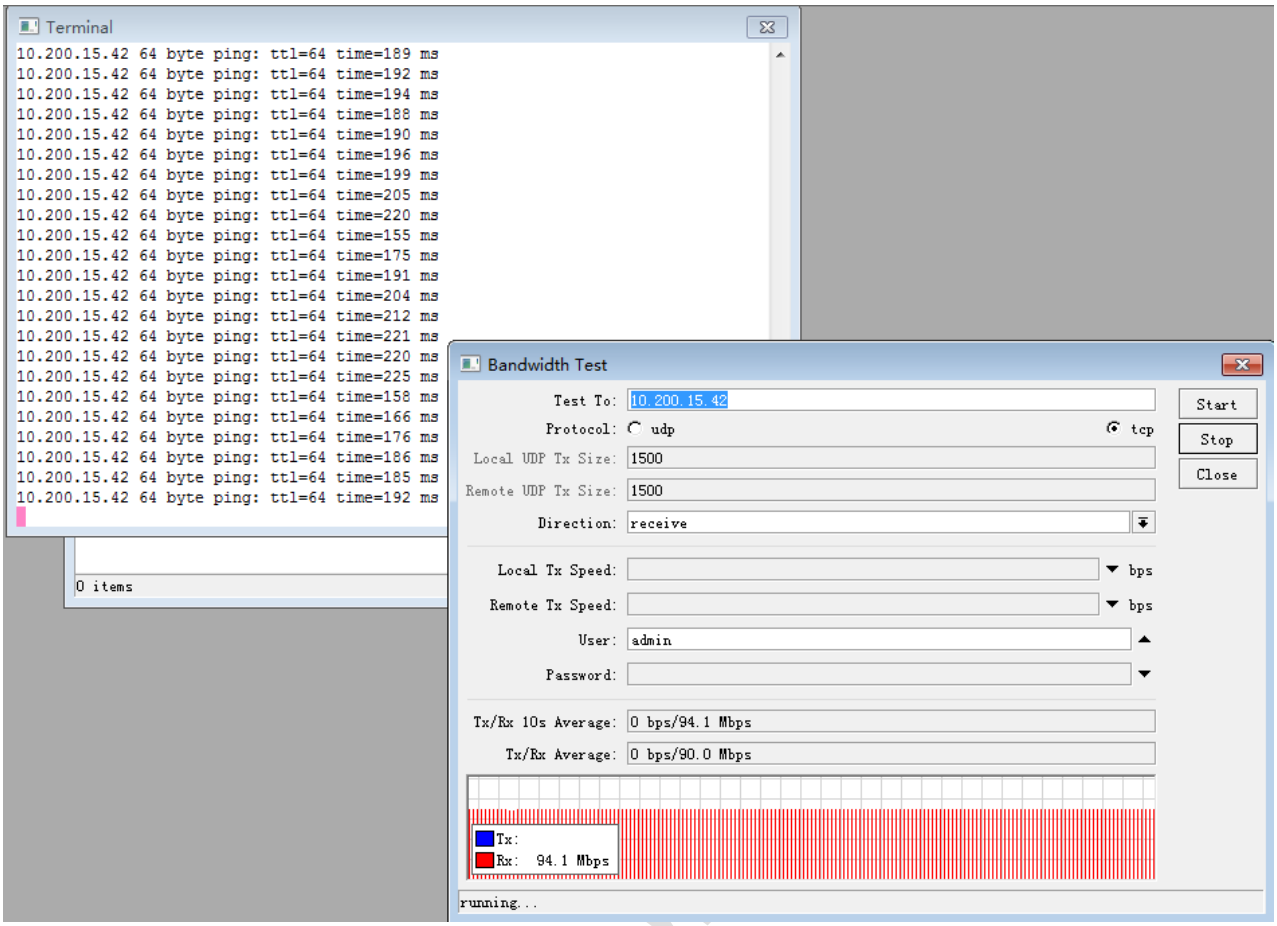
带宽测试环境：

服务端 RouterOS 3.30，硬件平台主板 5000VSA，CPU 至强 5405，2G 内存

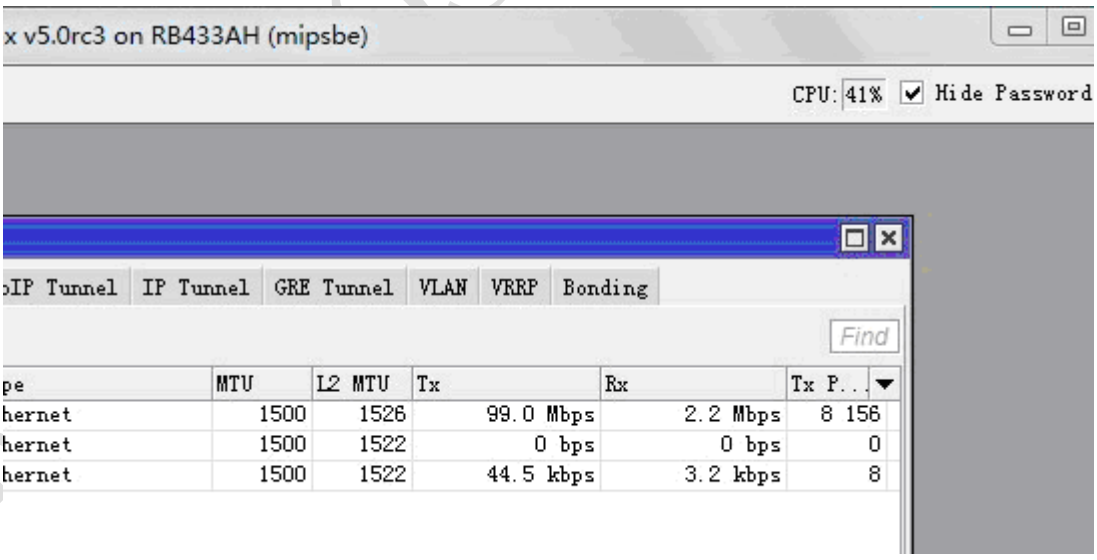
接收端：RouterOS 4.11，硬件平台：主板 5000VSA，CPU 至强 5420，2G 内存

网络结构：服务端 PC-----RB433AH+R52n ------ RB433AH+R52n --------接收端 PC

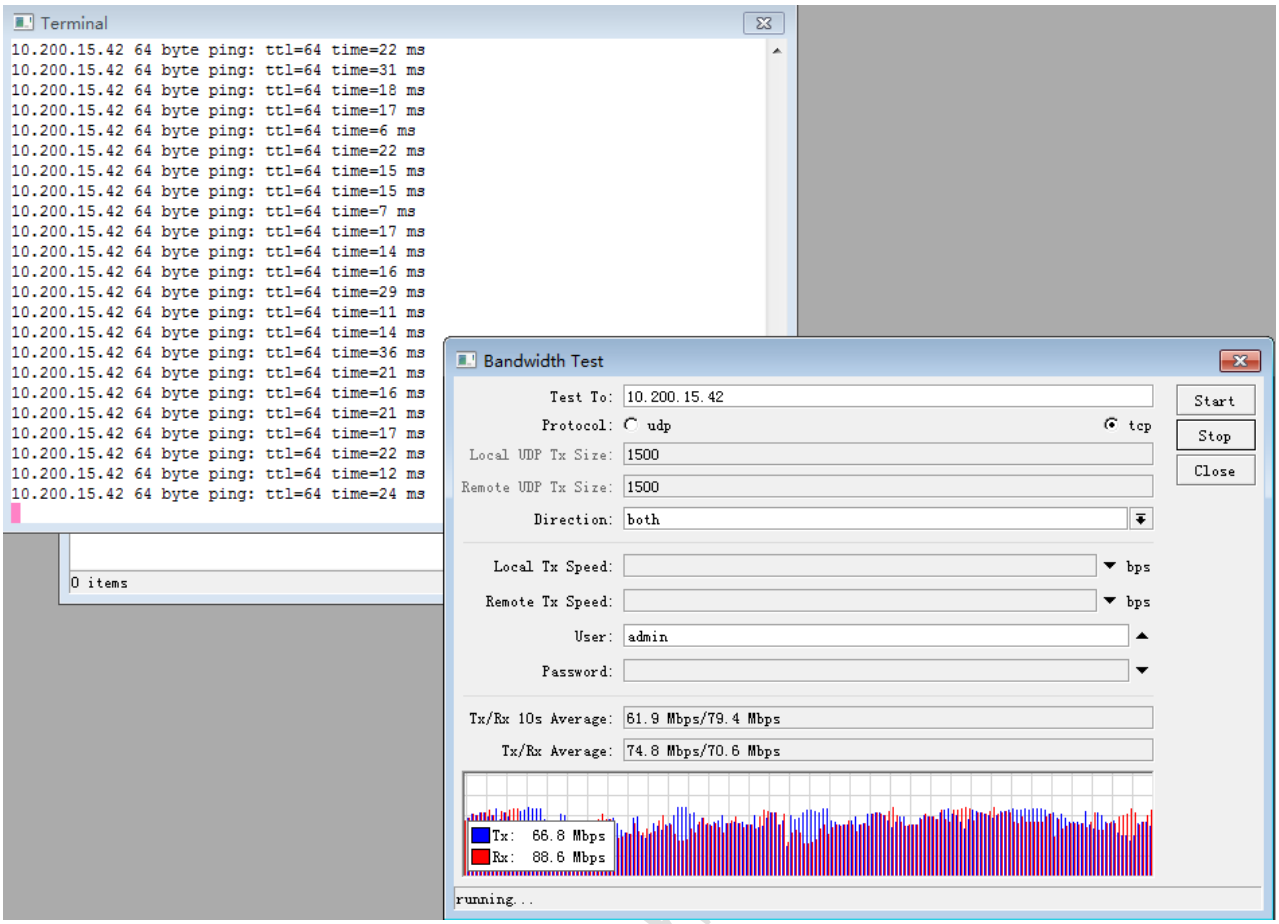
TCP 测试：单向 94.1Mbps



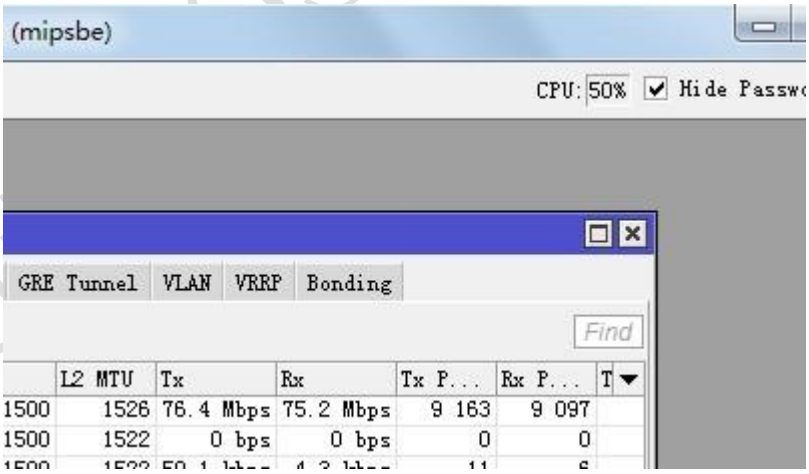
RB433AH 的 CPU 情况： 29-42%



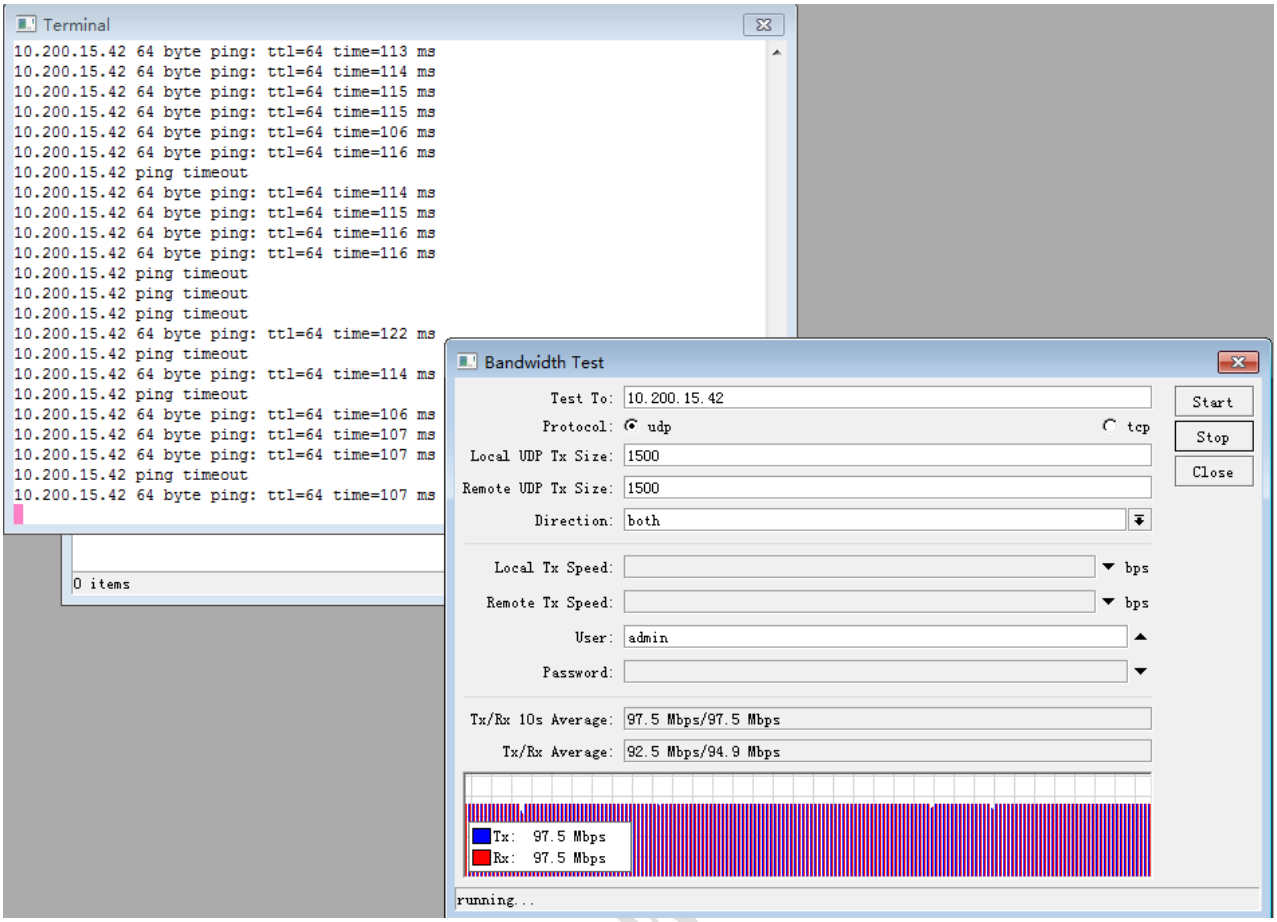
双向 74 / 70Mbps



RB433AH 的 CPU 情况： 45%-55%

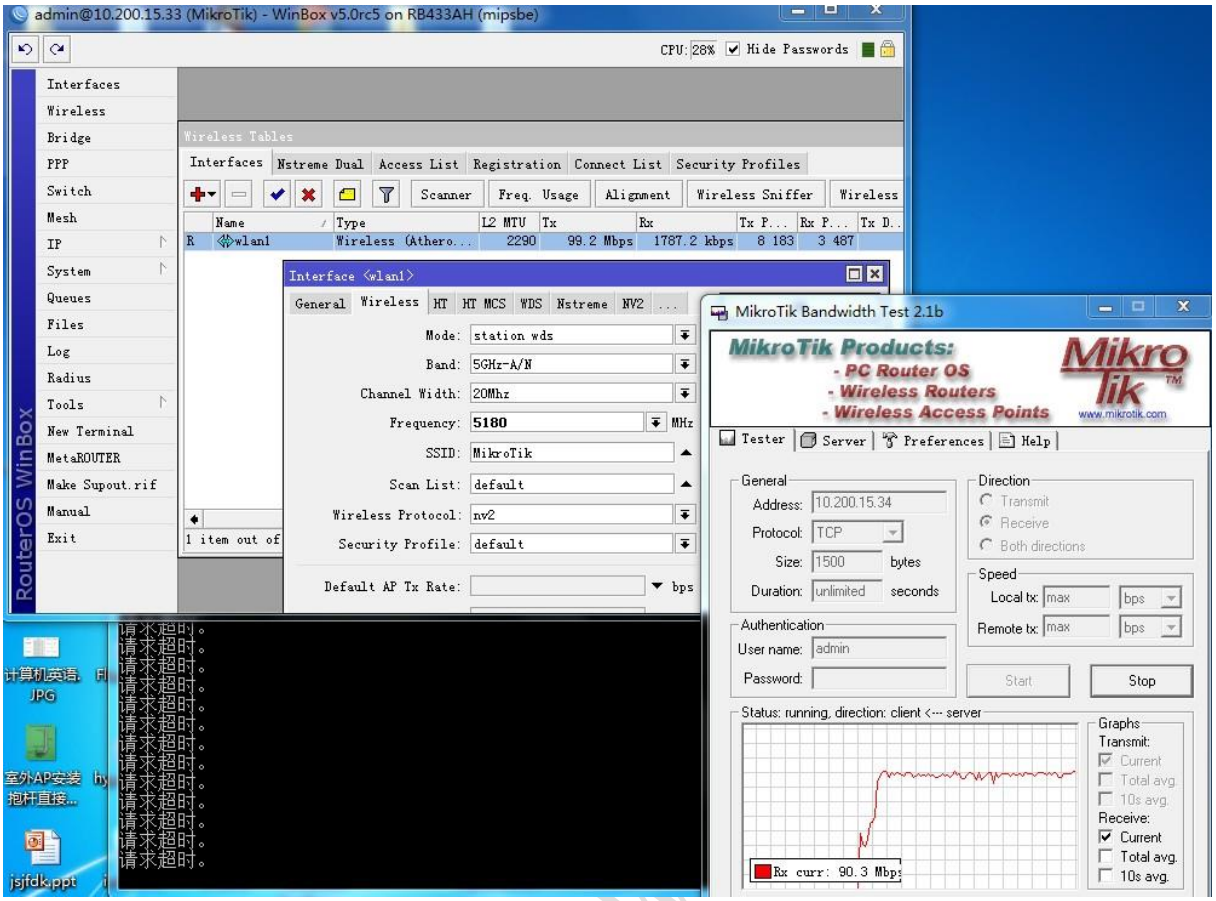


UDP：双向带宽都为 97.5Mbps



RB433AH 的 CPU 情况：47%-55%

以下是基于 Windows 的 bandwidth 软件测试带宽，可以看到在带宽满负荷的情况下，ping 请求已经超时，但连接并未断开，达到 90.3Mbps，



结论是：RB433AH 由于受到 100M 以太网卡的限制，无法发挥最大的性能，TCP 只能在 94.1Mbps 的带宽，从双线带宽测试看达到了 70Mbps 的全双工，如果是 RB435G 使用 1000M 以太网卡，单向 TCP 带宽可以达到 140-150Mbps，UDP 可以在 200-220Mbps，采用 RB800 会更好。

最近一朋友测试了一对 RB711GA-5HnD 的 11n 在 Nv2 下的传输，RB711GA-5HnD 是千兆以太网卡，TCP 带宽达到了 190Mbps。

7.4 Nv2 QoS

802.11 协议基于点对点双向通信其实是半双工模式，即通过时分间隔的双向传输。在很多情况下传输会出现流量过高而堵塞的情况，我们需要使用 QoS 来保证某些协议能优先通过。

Nv2（Nstreme Version 2）无线协议是 RouterOS v5.0 加入的，Nv2 基于 TDMA 时分多址协议，并只能工作在 RouterOS 设备上。Nv2 的 QoS 采用变量队列数，能自由配置 2，4 或 8 个队列，这个队列基于 IEEE802.1D。

Nv2 的 QoS 是通过数量可变的队列实现优先级，队列传输标准是基于 802.1D-2004，实际运行中 QoS 会对所有帧进行分析，队列优先级高的优先发送。Nv2 网络中的 QoS 策略是被 AP 控制的，客户端策略来至于 AP 端，仅需要 AP 的 QoS 策略配置即可

Nv2-qos=default

在这个模式发出的帧首先会被内建的 QoS 策略检查，队列的选择会针对数据报和长度决定。如果内建策略没有匹配会选择帧优先级字段，如 ICMP 默认是被 Nv2 QoS 优先处理。

Nv2-qos=frame-priority

这个模式 QoS 队列选择基于帧优先级字段，即我们开启了自定义的优先级策略配置

Frame-Priority 字段 RouterOS 必须知道，可以用的优先级。预先定义每一个标识为 2、4 或 8 的 Nv2 队列，分配到实际的数据报中。这个操作可以通过防火墙的 action=set-priority 预先设置。这个设置可以

从以下获得：

- 二层在 Bridge Filter 设置
- 三层在 ip firewall mangle 设置

注意：Frame-Priority 字段不会被保存在任何数据包头部，仅仅是 RouterOS 系统内部使用。

队列的数量选择基于 802.1D 的‘Frame-Priority’ 字段代码， 下面是优先级与传输类型的对比

优先级	传输类型
1	Background
2	Spare
0 (Default)	Best Effort
3	Excellent Effort
4	Controlled Load
5	Video
6	Voice
7	Network Control

映像 ‘Frame-Priority’ 字段到队列，依赖 Nv2-queue-count 参数（2、4 或 8 队列），映像代码如下：

nv2-queue=2	**nv2-queue=4**	**nv2-queue=8**
priority 0,1,2,3 -> queue 0	priority 0,1 -> queue 0	priority 0 -> queue 2
priority 4,5,6,7 -> queue 1	priority 2,3 -> queue 1	priority 1 -> queue 0
	priority 4,5 -> queue 2	priority 2 -> queue 1
	priority 6,7 -> queue 3	priority 3 -> queue 3
		priority 4 -> queue 4
		priority 5 -> queue 5
		priority 6 -> queue 6
		priority 7 -> queue 7

当选择 nv2-queue=2 时，字段 0-3 为队列 0，4-7 为队列 1，也就是队列数越多，对数据优先分类越细。

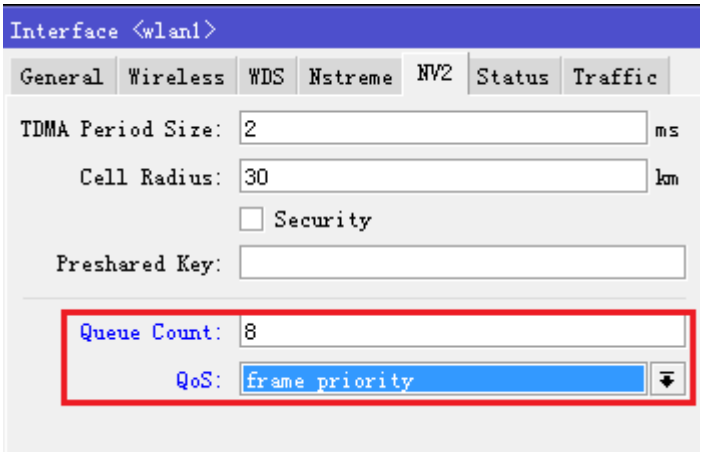
Nv2 QoS 实例

下面通过实例来讲解，我们通过 ICMP 来演示如何使用 Nv2 的 QoS，为了方便演示这里考虑 ICMP 协议优先通过，即不管 Nv2 无线链路传输带宽是否瓶颈都让 ICMP 协议优先通过，保持低延迟。

首先我们需要了解等级的排序，我们可以通过下面的表来匹配，但注意 0 为默认级优先级高于 1 和 2：

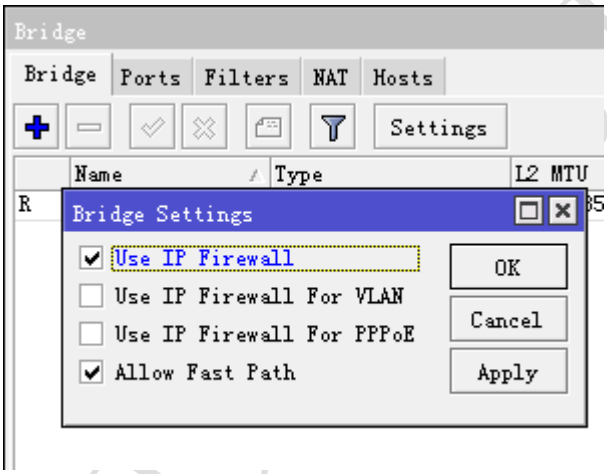
Set Priority	**优先级**
1	低
2	
0 (默认)	
3	
4	
5	
6	
7	高

我们将 Nv2 的配置修改为，开启 8 个队列，使用 Qos 为 frame priotity



三层 ip mangle 标记

我们需要预先定义 ICMP 协议，这里需要通过三层的 ip firewall mangle 定义 priority-field 字段，所以我们需要设置 bridge setting 中的 use-ip-firewall=yes



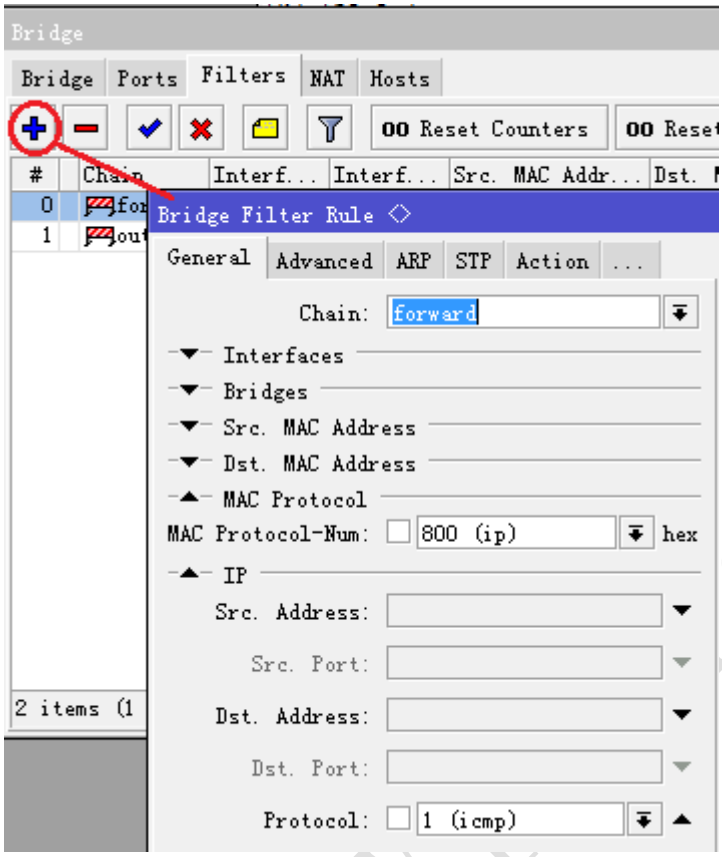

```
/ip firewall mangle
add chain=forward action=set-priority new-priority=7 protocol=icmp
add chain=output action=set-priority new-priority=7 protocol=icmp
```

这里设置两个链表，因为 forward 负责终端与终端的 icmp 资料，output 负责 AP 设备到对端 station 设备的 icmp 资料，设置 priority=7 是最高优先级。

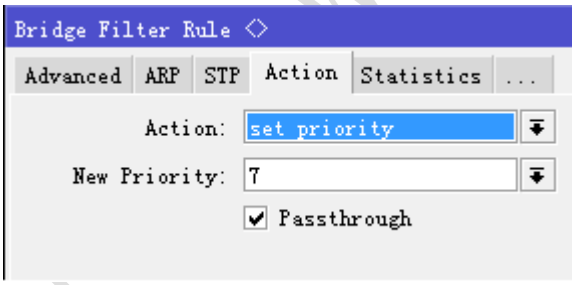
这里是使用了 mangle 来完成 nv2 qos 配置，我们也可以选择 bridge filter 完成，这样可以不需要开启 use ip firewall，减少复杂的策略调用，降低系统开销。当如果你有较为复杂的三层和四层协议需要控制，可以调度 mangle 策略

二层 bridge filter 标记

在 bridge filter 中添加策略，如下添加一条 forward 的 icmp 优先策略



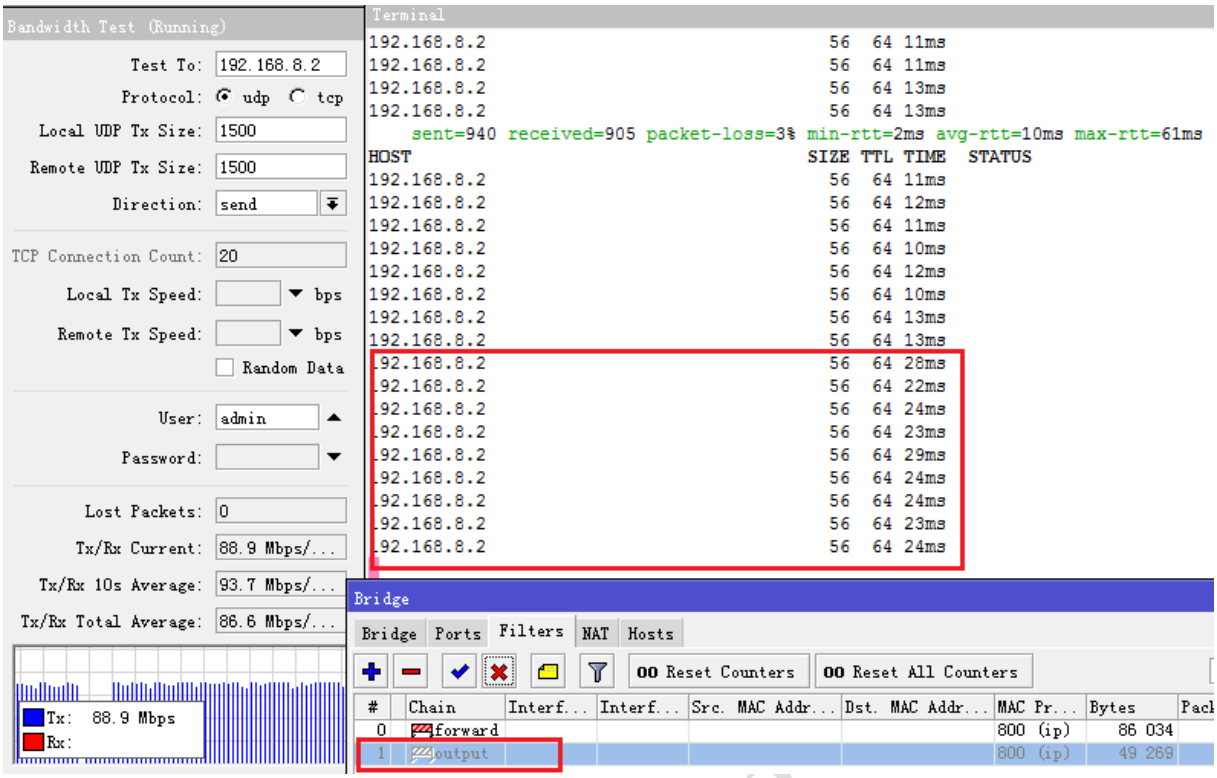
设置 priority 为 7



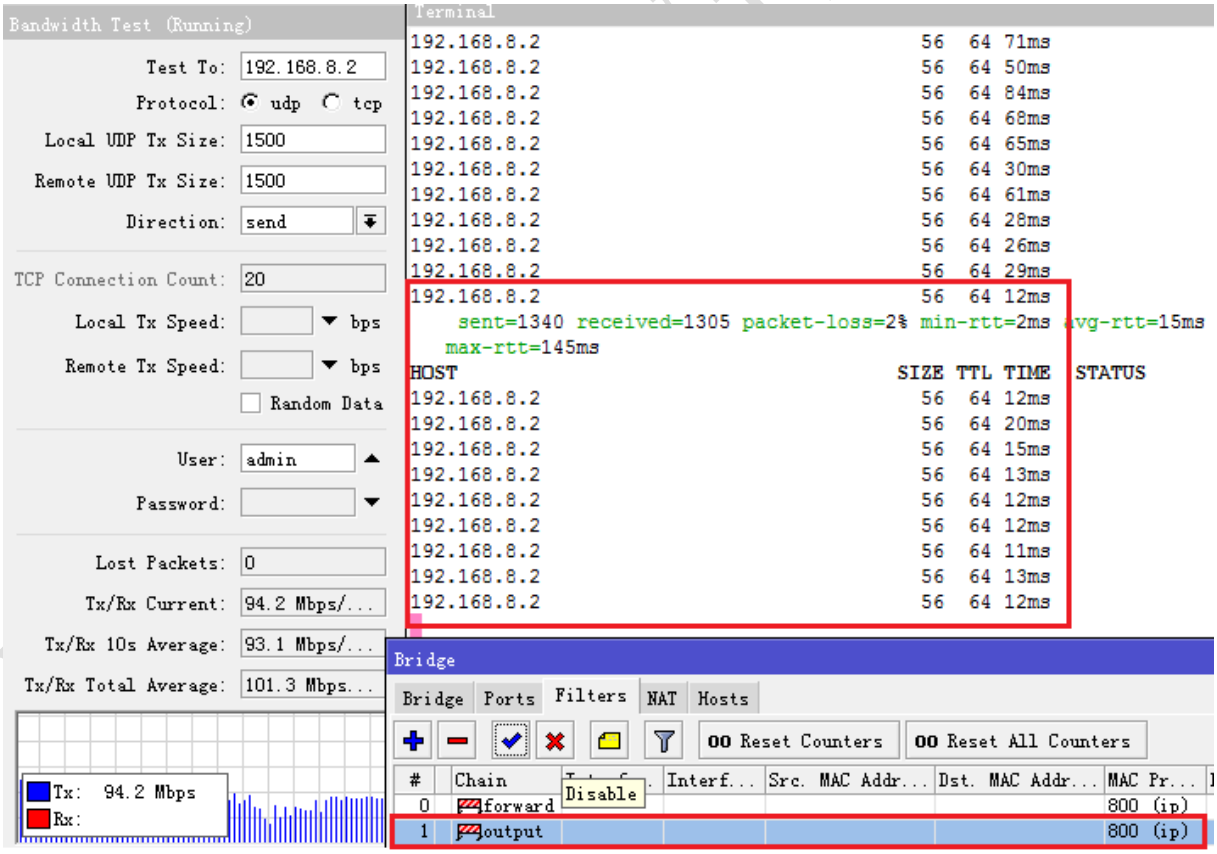
然后我们用相同的配置添加 output 规则：

```
/interface bridge filter
add action=set-priority chain=forward ip-protocol=icmp mac-protocol=ip
      new-priority=7
add action=set-priority chain=output ip-protocol=icmp mac-protocol=ip
      new-priority=7
```

当标记完成后，我们就可以测试 Nv2 的 QoS 应用是否生效，我们将带宽测试开启，延迟增加到 20～28m，这是 output 的 icmp 规则被禁用



我们开启 output 规则，明显延迟下降到 11～13ms



这里通过 ICMP 介绍了如何让指定的资料优先通过基于 Nv2 的 AP，根据实际生产环境，你可以制定自己需要的 QoS 规则，例如协议、端口或 L7 应用层等

7.5 Nv2 AP Synchronization

该功能是让多个 MikroTik Nv2 AP 在同一区域以更好的方式并存，减少彼此之间的干扰，该功能将同步多个 AP 之间在相同频段的传输/接收划分时间窗口。这允许同一区域多 AP 重复使用相同的无线频率，为多个 AP 提供更灵活的频率规划。

Nv2 同步过程：

- 首先要选择并设置 Nv2 AP 同步主机，即"nv2-mode=sync-master"，为 Nv2 AP 同步设置通讯密码"nv2-sync-secret"
- Nv2 从 AP 需要设置"nv2-mode=sync-slave"，并与主 Nv2 AP 设置相同无线频率，以及相同的"nv2-sync-secret"值
- 当主 AP 启用，从 AP 将尝试通过指定的"nv2-sync-secret"搜索主 AP
- 当主 AP 找到从 AP 后，从 AP 将计算到主 AP 的距离，因为可能与主 AP 不在相同的位置。
- 然后从 AP 开始工作在 AP 状态，并同步主 AP 匹配周期大小和下行比率
- 另外，如果有其他从 AP，可以通过该从 AP 进行同步。
- 从 AP 定期监听主 AP，并检查“nv2-sync-secret”是否仍然匹配并再次调整参数。如果主 AP 接口被禁用/启用，那么所有从接口也将被禁用，并将重新开始同步过程。
- 如果主 AP 停止工作，从 AP 同样将停止，他们之间将不在同步信息

配置事例

主 AP:

```
/interface wireless set wlan1 mode=ap-bridge ssid=Sector1 frequency=5220
nv2-mode=sync-master nv2-preshared-key=clients1 nv2-sync-secret=Tower1
```

从 AP:

```
/interface wireless set wlan1 mode=ap-bridge ssid=Sector2 frequency=5220
nv2-mode=sync-slave nv2-preshared-key=clients2 nv2-sync-secret=Tower1
```

在从 AP 上监控接口状态 Monitor interface on the Slave AP:

```
[admin@SlaveAP] /interface wireless> monitor wlan1
                 status: running-ap
                channel: 5220/20/an
      wireless-protocol: nv2
           noise-floor: -110dBm
     registered-clients: 1
   authenticated-clients: 1
         nv2-sync-state: synced
        nv2-sync-master: 4C:5E:0C:57:84:38
       nv2-sync-distance: 1
    nv2-sync-period-size: 2
  nv2-sync-downlink-ratio: 50
```

主 AP Debug 日志

```
09:22:08 wireless,debug wlan1: 4C:5E:0C:57:85:BE attempts to sync
```

从 AP Debug 日志:

```
09:22:08 wireless,debug wlan1: attempting to sync to 4C:5E:0C:57:84:38
09:22:09 wireless,debug wlan1: synced to 4C:5E:0C:57:84:38
```

7.6 Nstreme Dual

Nstreme Dual 是早期的 Nsteme 协议的改进，即进一步提高带宽，类似采用 2x2 传输方式，只不过是单向的 2x2。

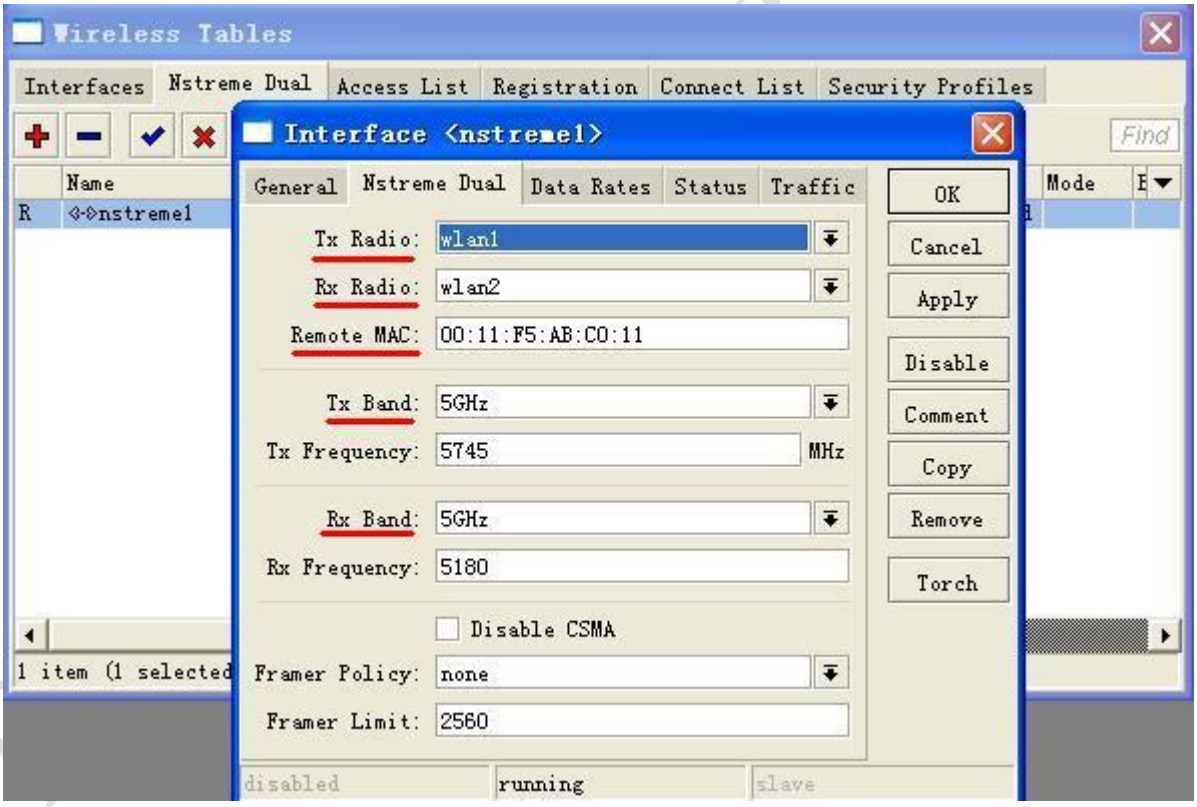
wireless 的 Nstreme Dual 目录项中，添加 nstreme 接口，需要注意以下参数：

- **Tx Radio：传输网卡**
- **Rx Radio：接收网卡**
- **Remote MAC：远程 nstreme 接口 MAC 地址（非物理网卡 MAC 地址）**
- **Tx band：传输频段**
- **Tx frequency：传输频率**
- **Rx band：接收频段**
- **Rx frequency：接收频率**

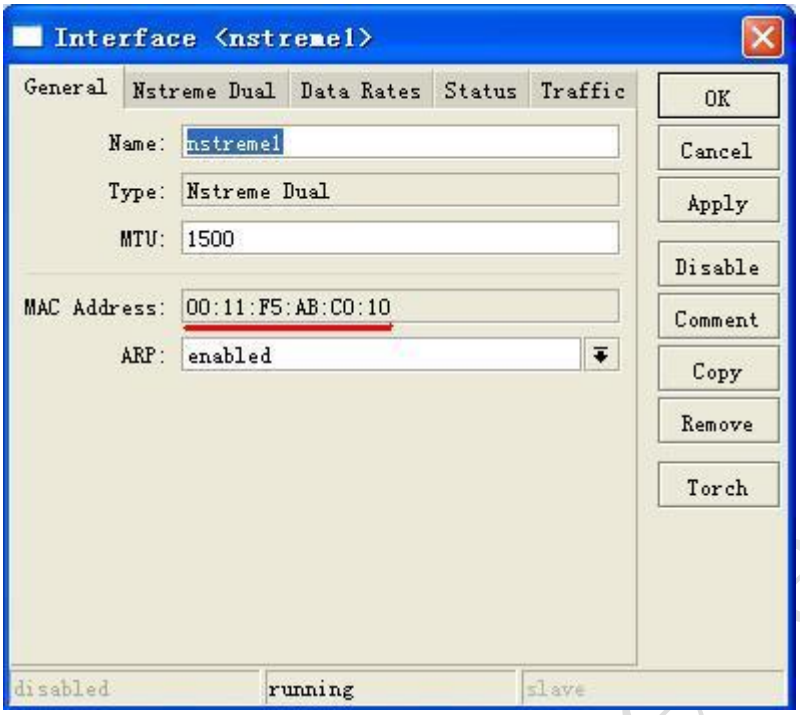
注意：两边设备的传输和接收频段和频率必须相同，双方的传输和接收频率应当对应。

假设我们有两个网站：网站 1 和网站 2，都采用 Dual radio Point-to-Point mode (nstreme2)的方式连接

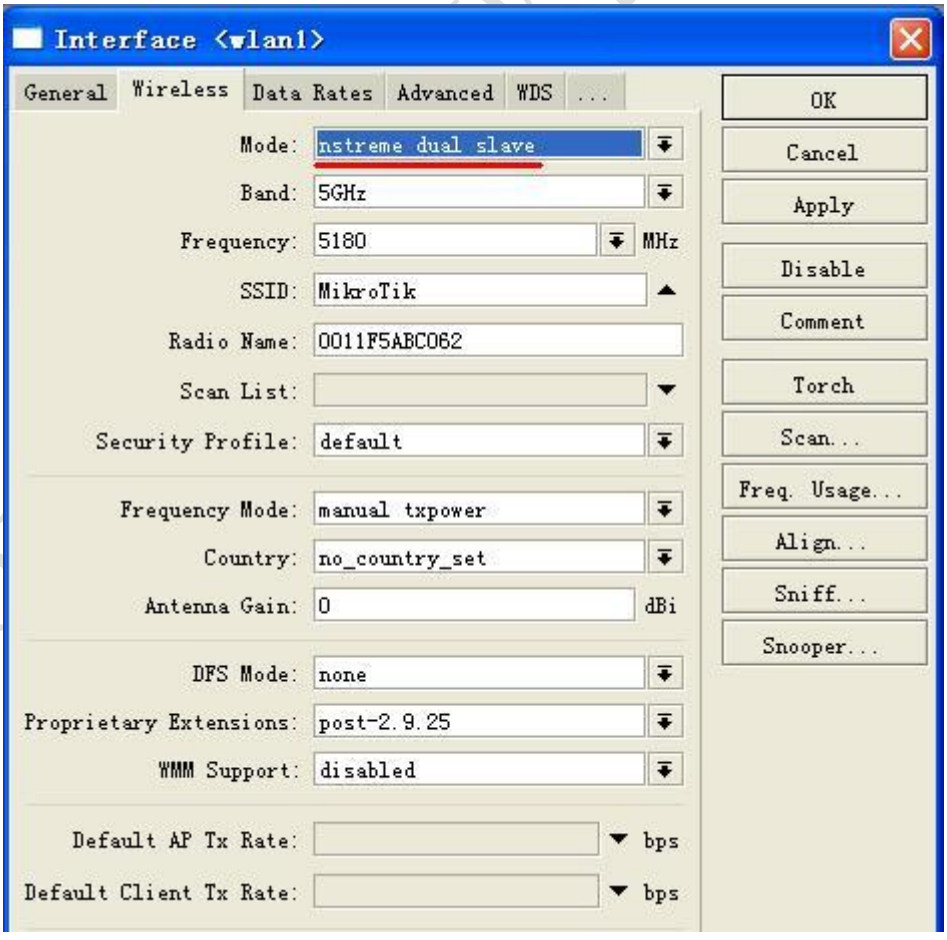
网站 1 的 Nstreme1 配置如下：



这里我们需要记录下网站 1 的 nstreme 的 MAC 地址，注意不是无线网卡的 MAC 地址：

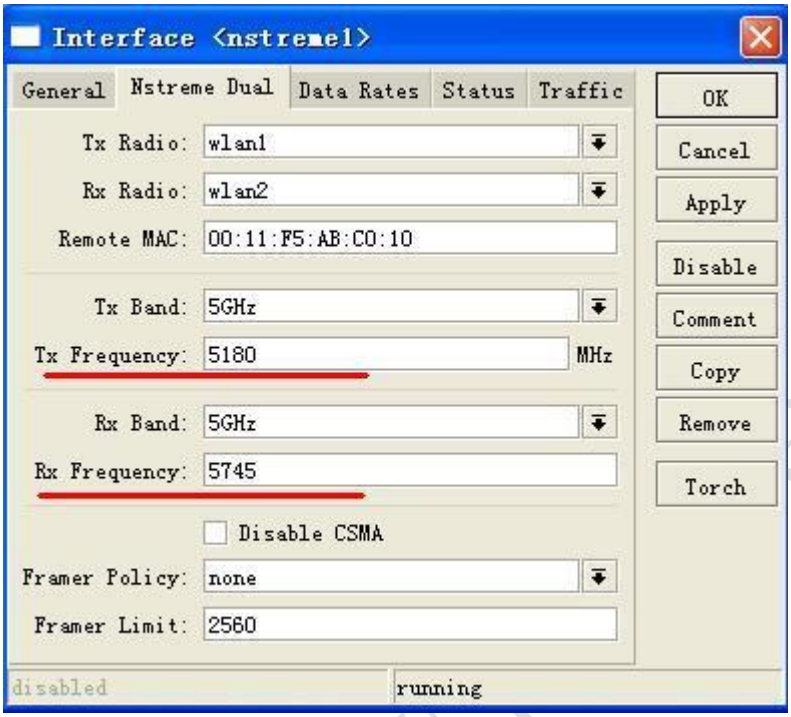


接下来配置网站 1 的两张无线网卡参数，两张无线网卡的 Mode 设置为 nstreme dual slave,其他参数默认即可。



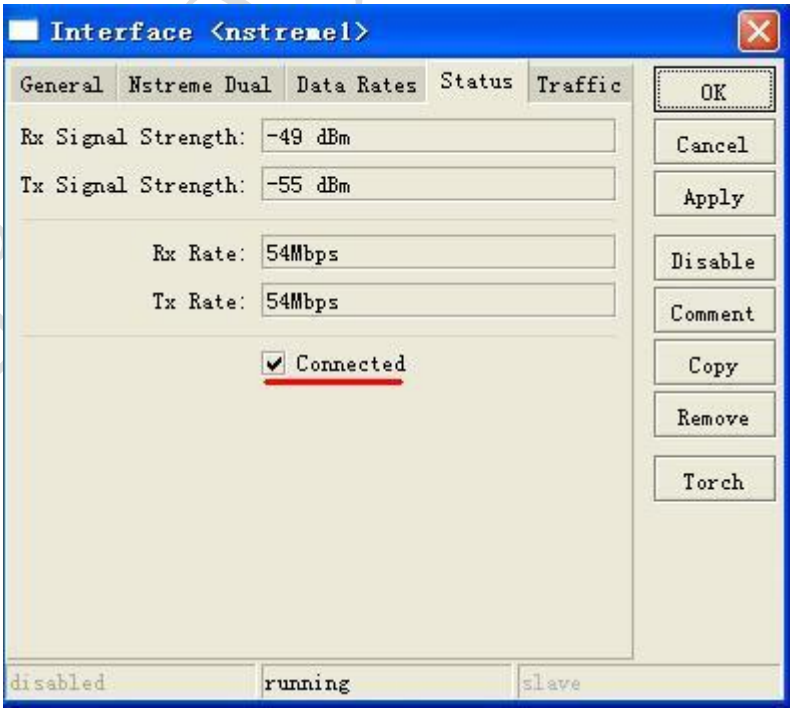
这里当无线网卡模式选择为 nstreme dual slave,网卡的频率和其他参数将从属于之前添加好的 nstreme1 网卡界面。

网站 2 的配置与网站 1 的配置相同，但有一个地方需要注意，Tx Radio 和 Rx Radio 需要设置对应。如下图：

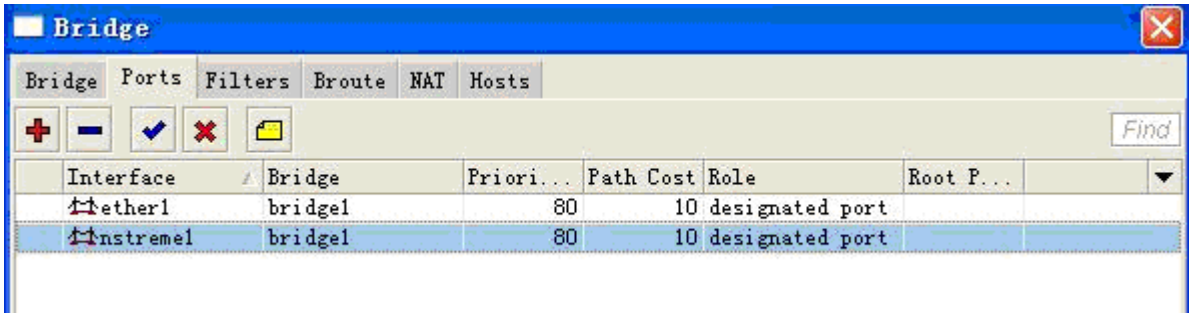


网站 2 的 Tx 频率要和对面网站 2 的 Rx 频率相同，同样网站 2 的 Rx 频率需要和网站 1 的 Tx 频率对应

当 Nstreme 连接上后，在 status 中显示 connected，并能看到其信号强度和速率:



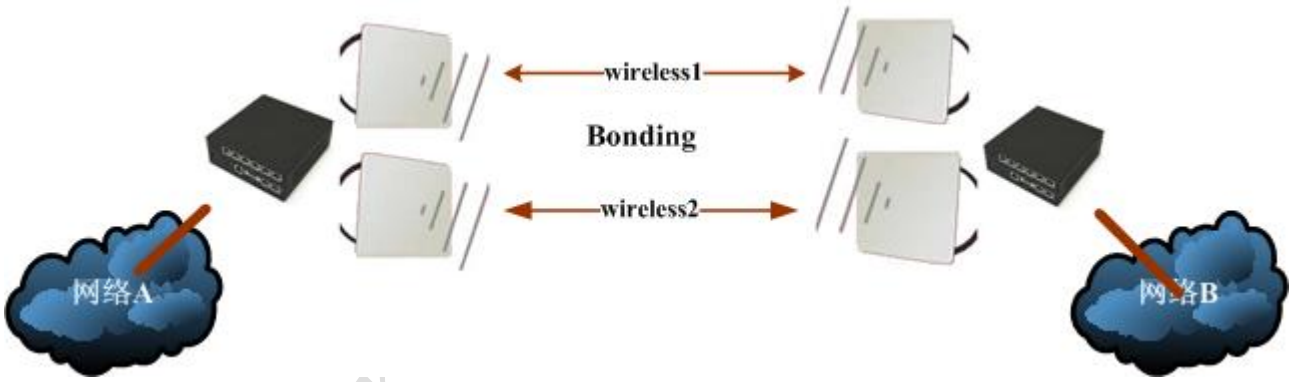
连接完成后，同样将无线建立桥接，进入 bridge 中，将 ether1 与 nstreme 归入 bridge1 中



7.7 无线网络 Bonding

Bonding 是基于二层的网卡绑定协议，可以将多个以太网卡加入到一个虚拟连接中，提供速率传输和容错功能。RouterOS 不仅支持普通的以太网卡的绑定，也可以支持基于 802.11 传输协议的无线网卡绑定

Bonding 操作是将 2 组或多组无线模块绑定在一起，起到带宽合并的作用，提高无线带宽的吞吐量。Bonding 功能是采用的是二层链路聚合带宽，所以对三层的 IP 网络没有直接的影响，和普通的三层负载均衡不同的是，不会受路由策略的影响。



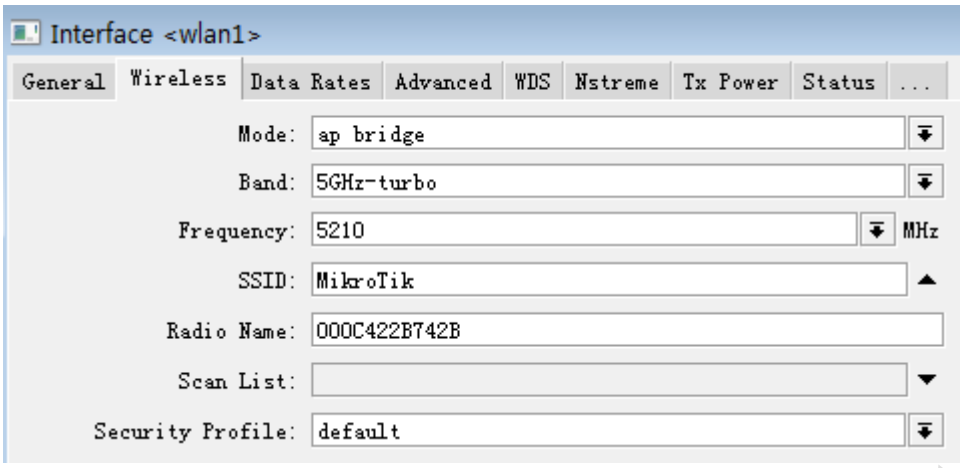
这里我们通过两台 RB600A 做测试，首先我们启用 2 个无线模块，配置 5G-Turbo 模式，提高无线带宽的吞吐量。

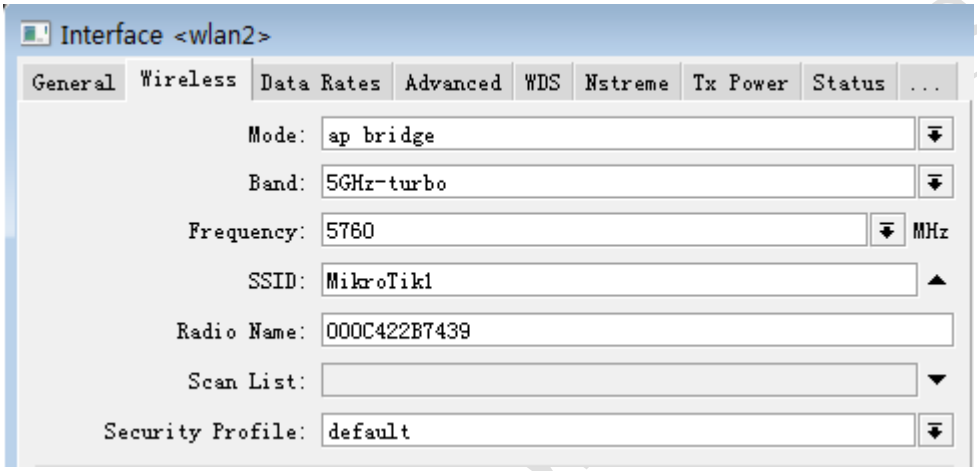
AP 配置

我们首先配置 AP 端的 RB600A，分别对 Wlan1 和 Wlan2 两个无线模块做配置：

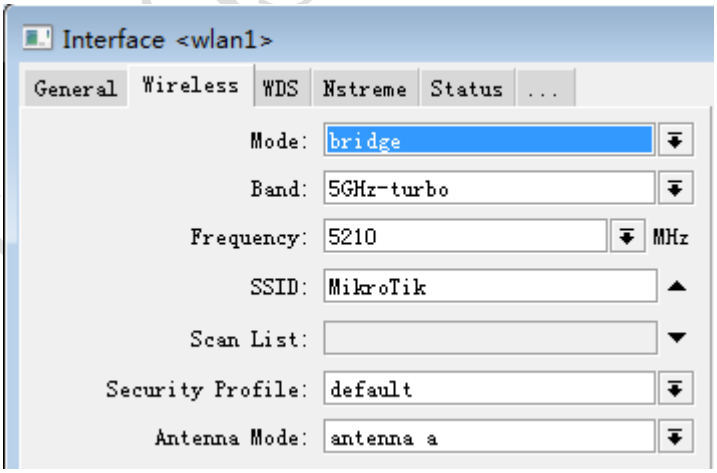
Wlan1	Wlan2
Mode：AP-bridge（bridge） Band：5GHz-turbo Frequency：5210MHz SSID：MikroTik	Mode：AP-bridge（bridge） Band：5GHz-turbo Frequency：5760MHz SSID：MikroTik1

下面分别是 wlan1 和 wlan2 的 AP 部分配置：

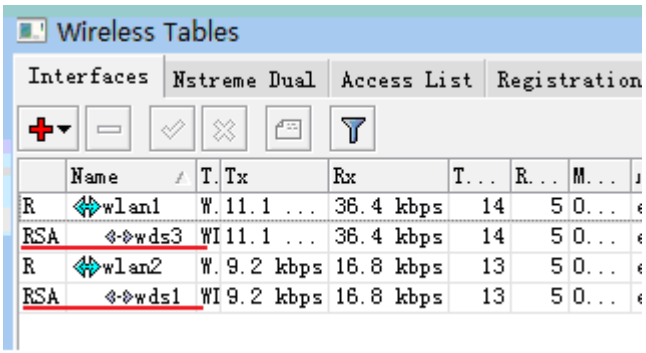




在 AP 下如果 ap-bridge 连接有问题，可以改用 bridge 模式，有时需要禁用启用网卡，才能正常连接：

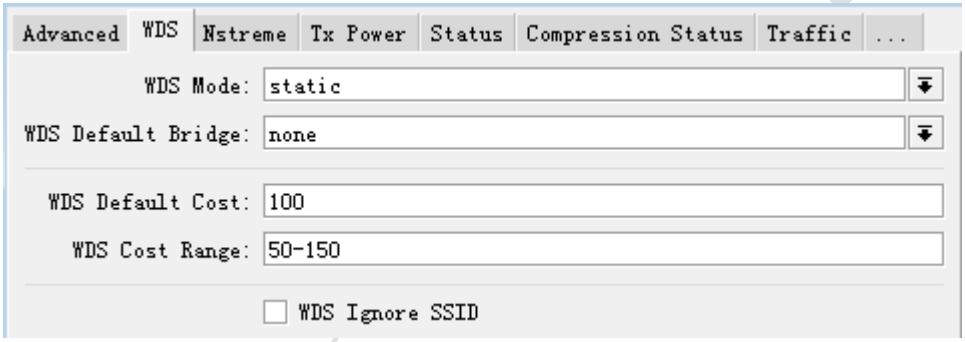


注：这里我们需要配置 WDS 的模式，因为在为连接前我们需要将 WDS-Mode 设置为 dynamic，当连接后在将 WDS 接口修改为 static 的静态模式。

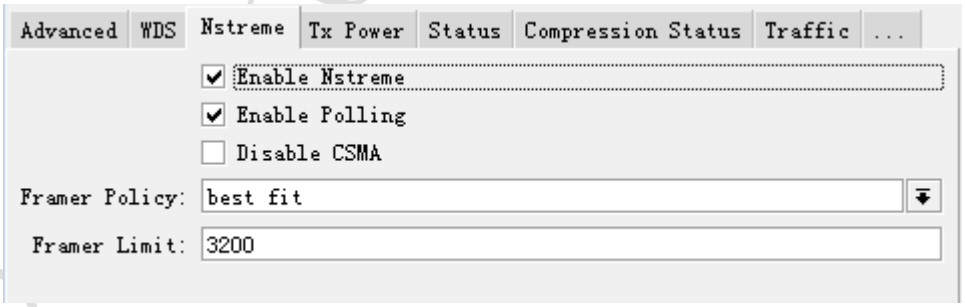


如上图，将 wds3 和 wds1 修改为静态的，前缀为 RSA 的标记，如果不是静态的模式，会显示 DRA。

连接后，我们将 wlan1 和 wlan2 的 WDS-Mode 修改为 Static，因为 bonding 需要指定网络接口，而如果采用动态 dynamic 模式，会在每次连接断开后，原来的 WDS 接口会失效，所以需要将 WDS 模式设置为静态的，在重复连接的情况下 bonding 功能才不会失效。



为了保证更好的带宽和传输质量，启用 Nstreme 协议，配置如下：

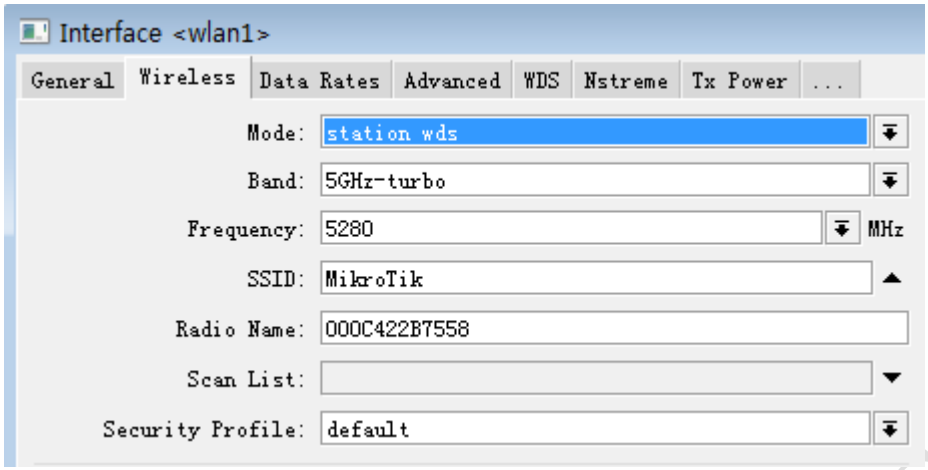


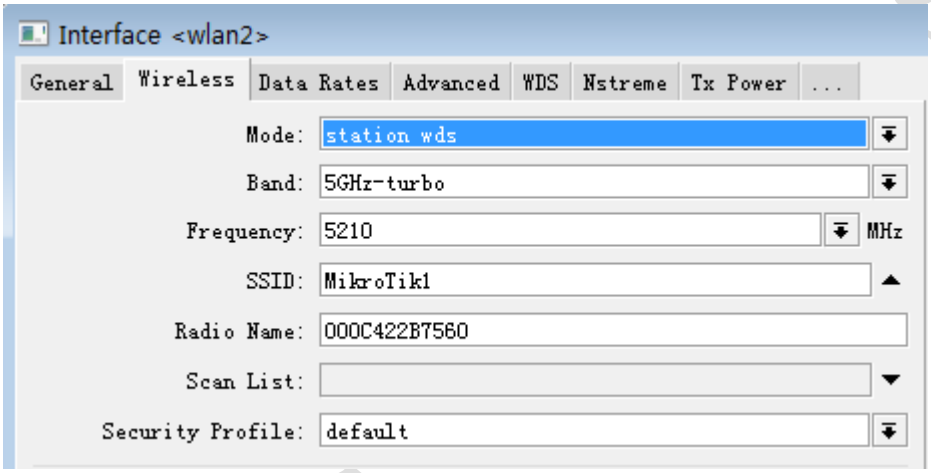
Station 配置

将 Station 设备的 wlan1 和 wlan2 配置为 station-wds 模式

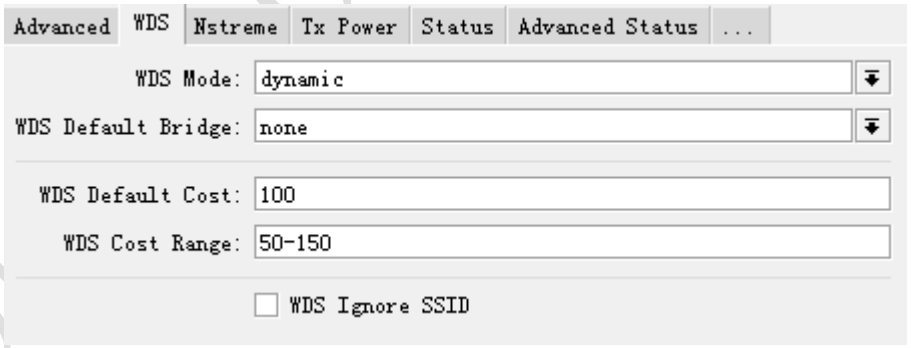
Wlan1	Wlan2
Mode：station-wds	Mode：station-wds
Band：5GHz-turbo	Band：5GHz-turbo
SSID：MikroTik	SSID：MikroTik1

下面分别是 wlan1 和 wlan2 的 station 部分配置

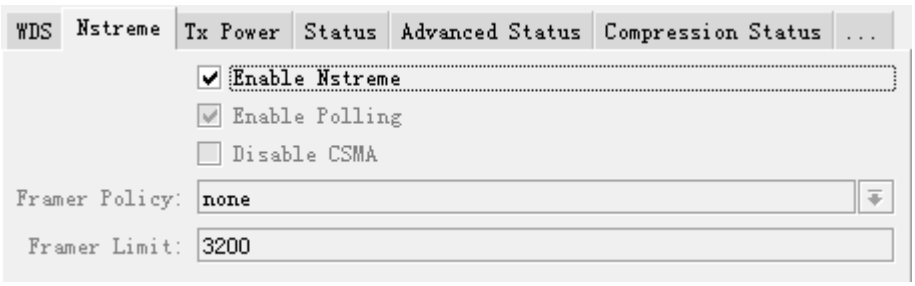




这里将 wds 参数配置为 dynamic

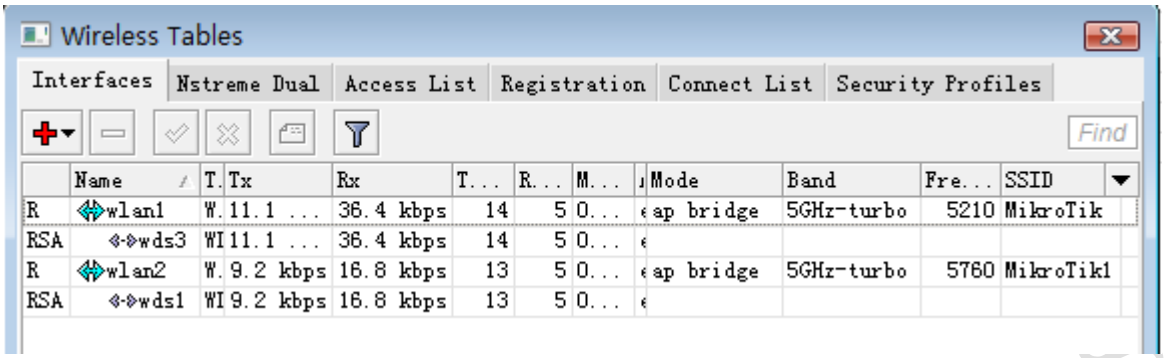


同样，启用 Nstreme 协议参数：

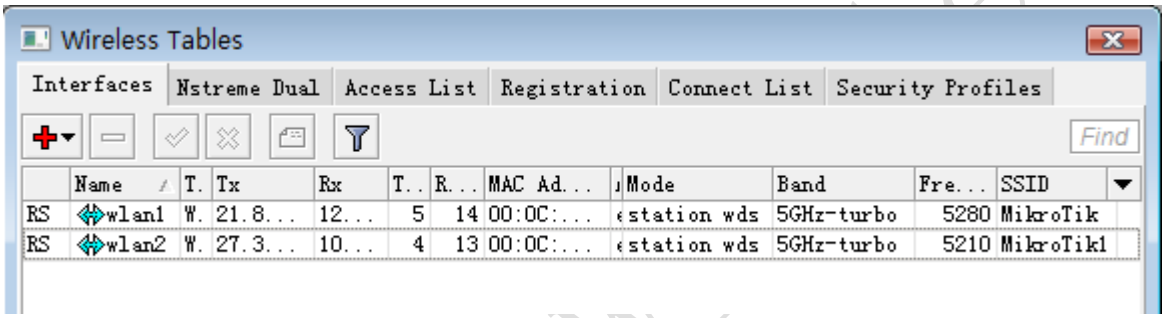


配置完成无线参数后，AP 和 Station 相互连接成功：

AP 连接状态：



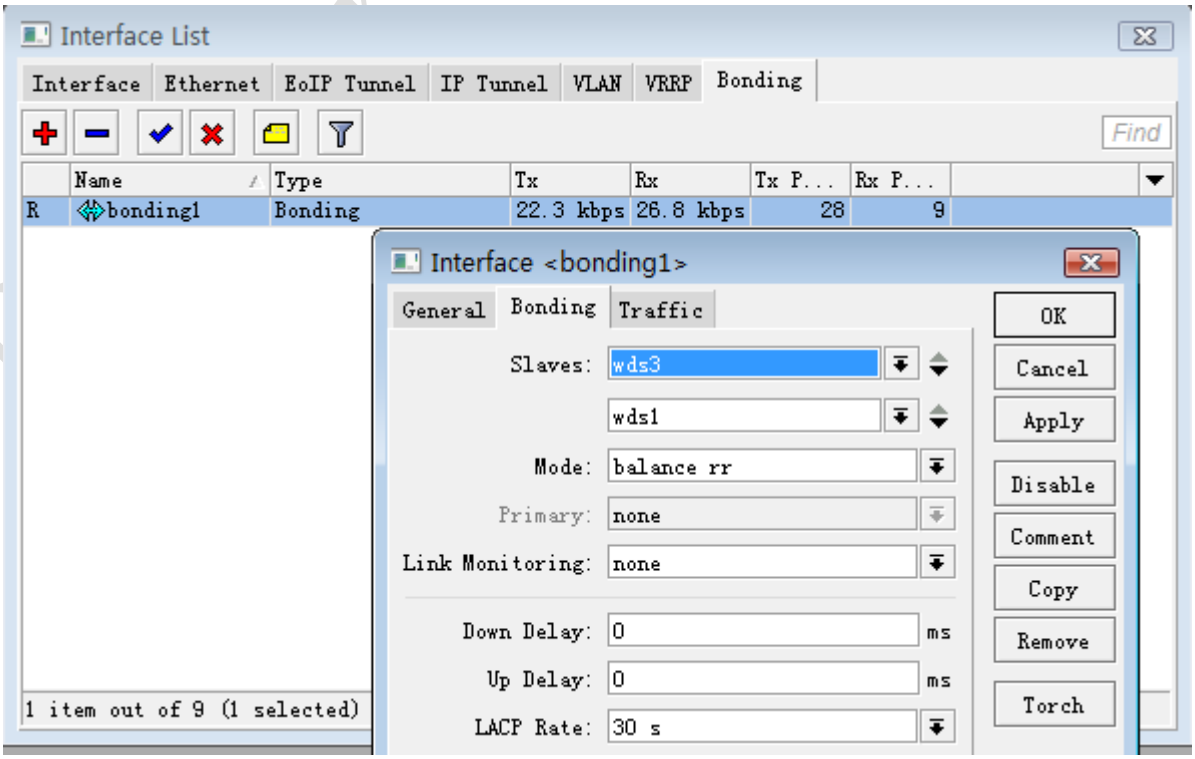
Station 连接状态：



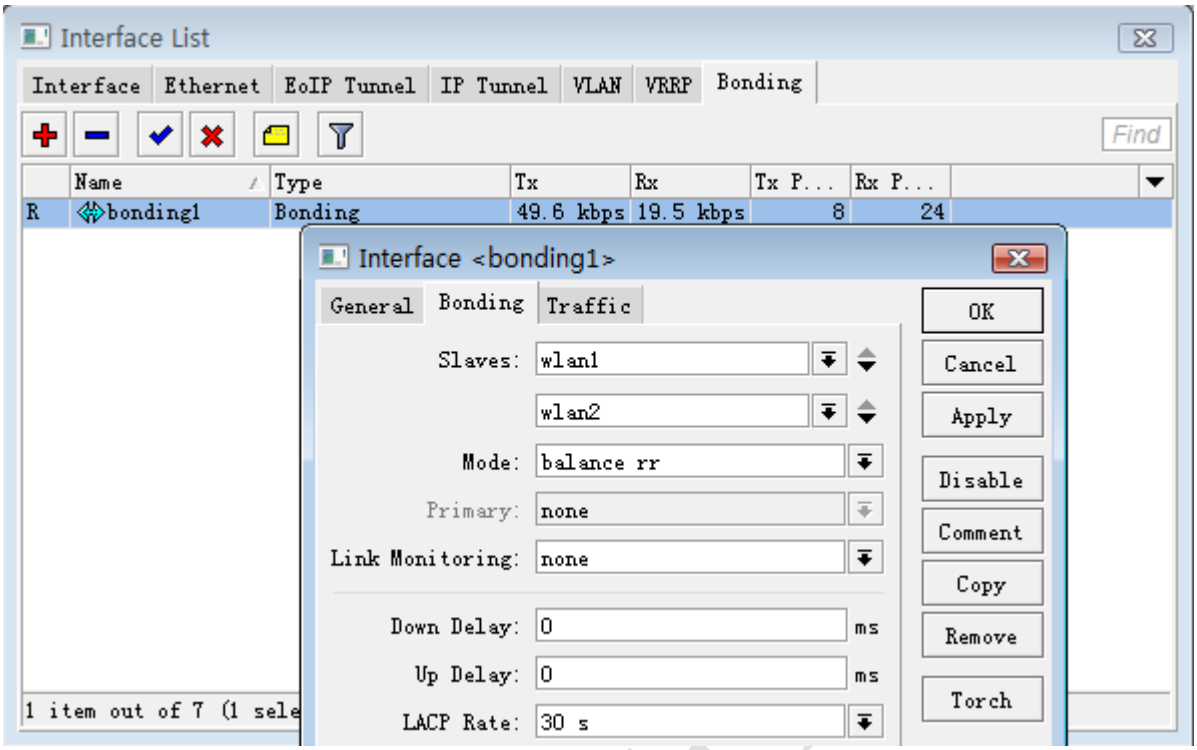
Bonding 配置

首先我们在 RouterOS 中添加 bonding，在 interface 中添加 bonding：

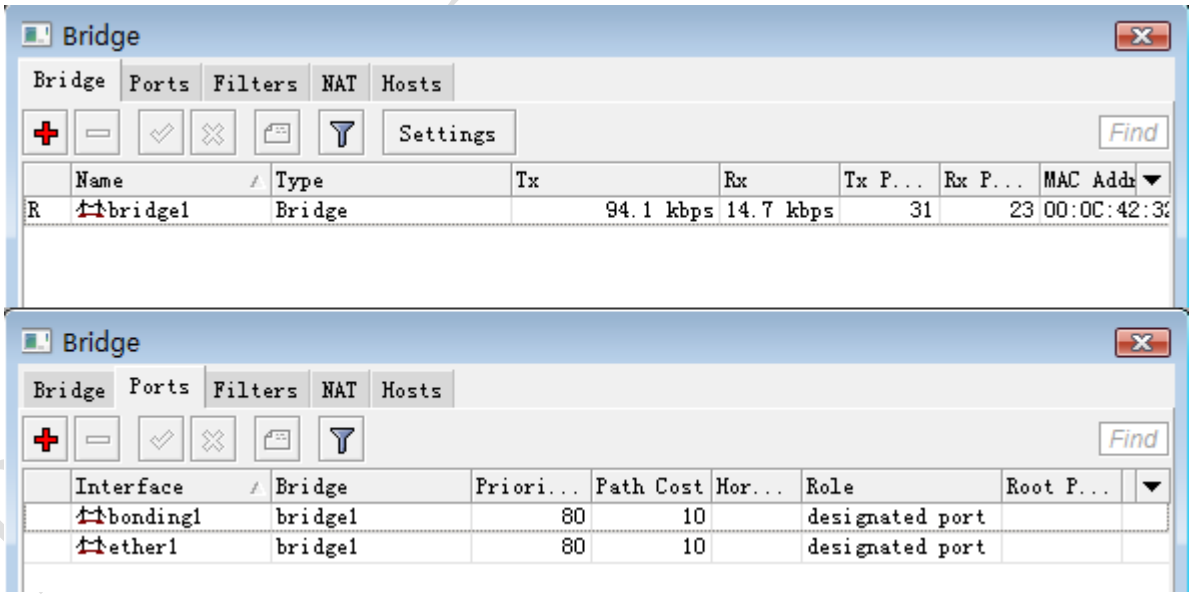
AP 端添加 bonding 时候，需要注意，绑定的不是 wlan1 和 wlan2，而是其自动增加的 wds 接口，这里分别是 wds3 和 wds1：



配置 station 端，直接添加 wlan1 和 wlan2：

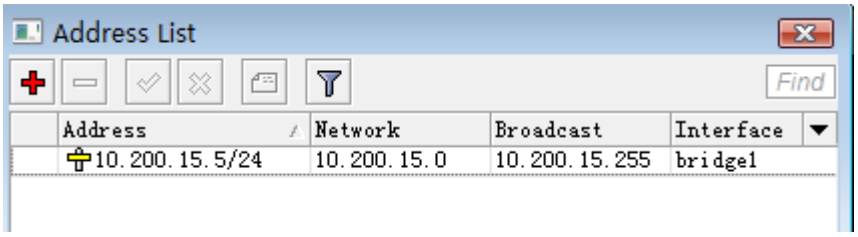


这样完成了 bonding 的参数，我们需要将绑定好的两组无线模块做成桥接模式，所以我们需要新建立 bridge 接口：

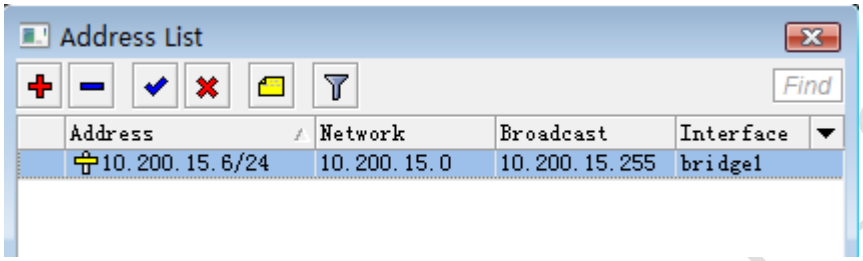


最后分配给 AP 和 Station 配置管理的 IP 地址到 bridge 上：

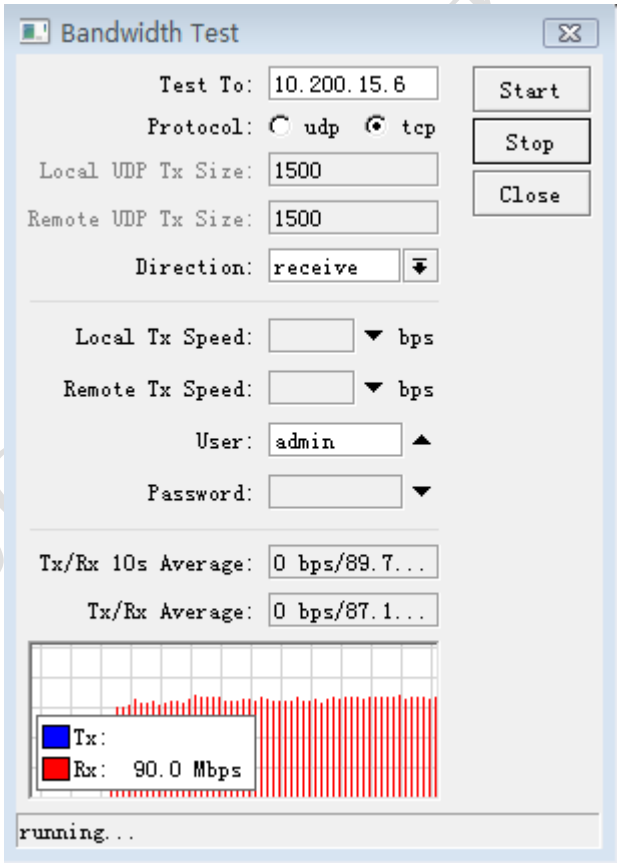
AP 的 IP 为 10.200.15.5



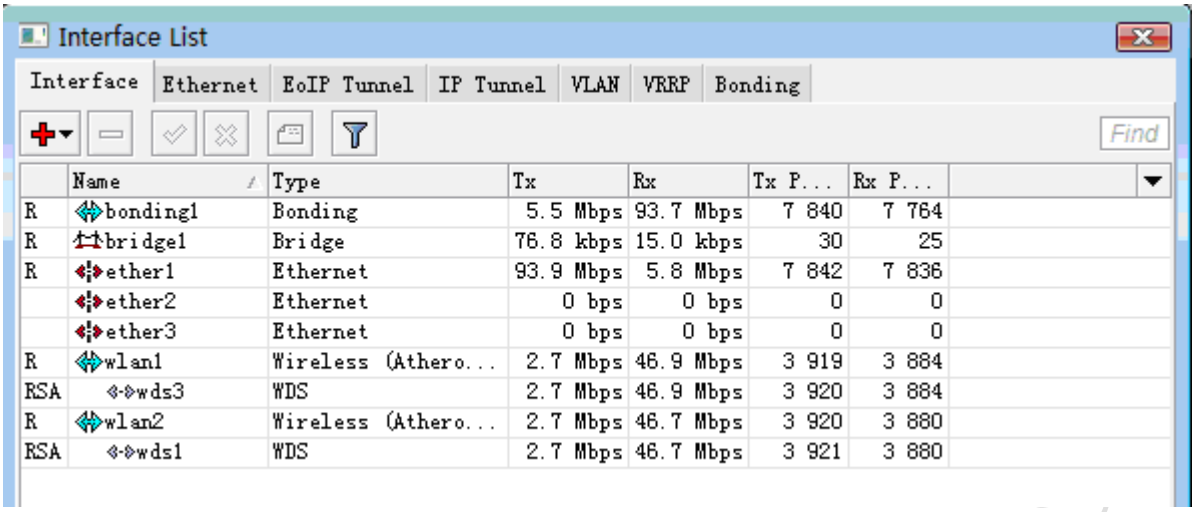
Station 的 IP 地址为 10.200.15.6



配置完 bridge 的桥接参数，bonding 操作就完成了，我们可以通过连接一台 PC 测试 bonding 后的带宽：



Bonding 模式会以最低带宽的无线模块为标准，即如果 wlan1 连接带宽为 60Mbps，而 wlan2 连接带宽为 40Mbps，则两个连接带宽合并应以 wlan2 为准，即 40Mbps*2=80Mbps。带宽的高低和信号强度有关，如果要获取高带宽，需要将信号强度调整到至少-60dBi 以上的信号。



在使用 bonding 模式时候，需要注意设备的 CPU 耗用，如果带宽没有达到理想的水平，可以通过提升 CPU 性能，如 RB600A 在运行 2 组无线模块的 bonding 时候，CPU 达到 100%，也就是说 Bonding 对 CPU 消耗非常大，需要考虑更好性能的设备。

7.8 Nstreme Dual 协议与 Bonding

Nstreme Dual 协议与 Bonding 协议都是为提高 WLAN 无线带宽而设计的，但他们有着不同的应用，通过下面的列表对比下：

功能/协议	Nstreme Dual	Bonding
支持无线模块	2 个模块，一个模块接收，一个模块发送	支持 2 个或 2 个以上模块，并且实现二层网络数据的汇聚
特点	上下行数据分开传输，提高网络双向传输效率，CPU 耗用较高	将多个无线模块汇聚，增加无线网络带宽，CPU 耗用特别高
设备需求	只能基于 1 对设备，完成协议	支持 1 对或多对设备组成，完成协议构建，在带宽需求〉100Mbps 情况下，需要多对设备完成，且需要高性能 PC 汇聚带宽
带宽	双向理想环境最大：65Mbps/65Mbps	单向理想环境最大（受设备组成数量、网卡速率、CPU 效率等）:〉100Mbps
应用范围	通过 1 对设备，提高双向传输的效率	对需要获取高带宽的环境

在更多的场合下 Bonding 模式更受欢迎，但对 CPU 耗用非常高。但随着 Nv2 协议的出现，Nv2 协议配合 802.11n 协议能得到更高的传输带宽，且处理器耗用更低。

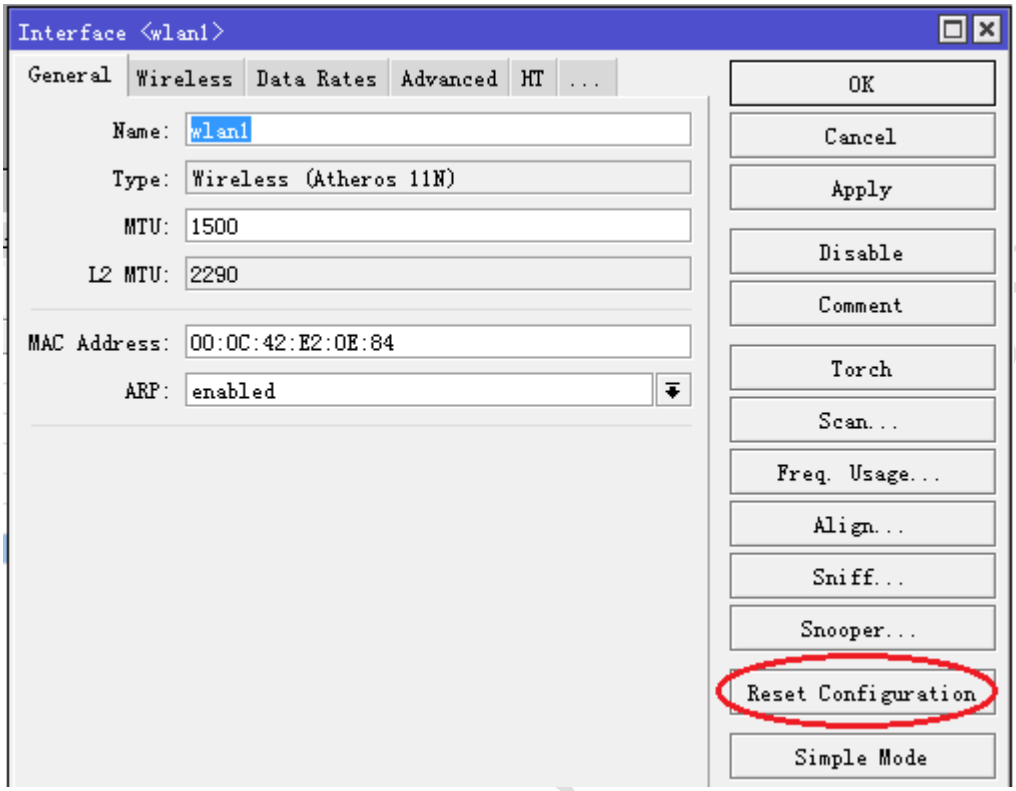
我们也可以使用基于 802.11n 协议的 Bonding 功能，即绑定 2 张 802.11n 的无线网卡，如果 1 张 802.11n 的无线网卡带宽是 120Mbps，2 张就是 240Mbps，不过这需要使用性能更高的设备如 RB800。

7.9 RouterOS 无线配置参数 FAQ

修改了一些无线网卡参数，造成连接变动的不稳定。

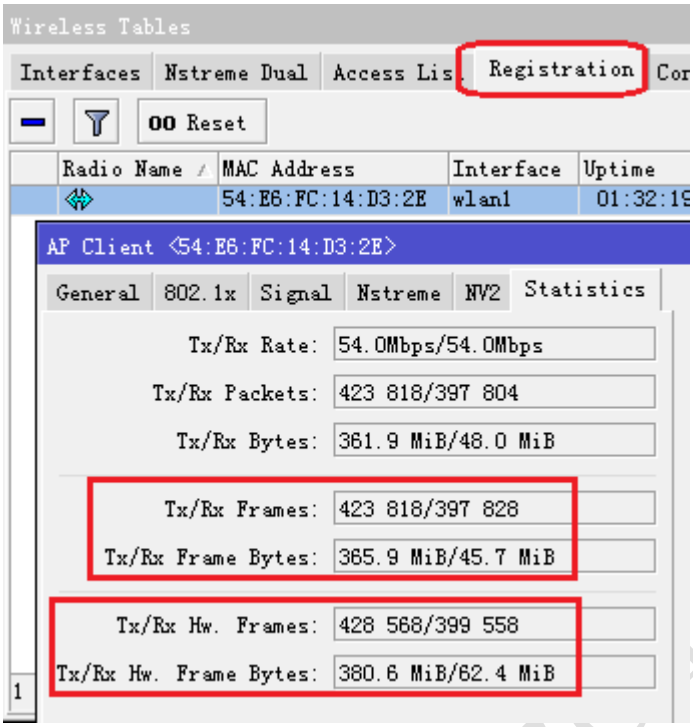
有时当你为了调整或优化一些连接修改了一些无线设置参数，但你又忘记了曾经设置过的参数，在

这样的情况下你可以在 wireless 配置菜单下使用 reset-configuration 命令，该命令会复位该无线网卡的设置参数到初始化状态。注意这个命令也会禁用该无线网卡，所以请你小心使用，特别是在通过无线连接到设备情况下调整



什么是无线重发（wireless retransmits），并如何查看?

无线重发是当网卡发出一个帧，并没有收到来至对端确认（ACK），你再次发出帧直到你收到对端的确认（ACK）。无线重发会增加延迟和降低吞吐量。如果无线连接存在无线重发情况，你需要比较两个字段，在 registration table: **frames** 和 **hw-frames** ，如果 *hw-frames* 值大于 *frames* 即无线连接正在重发数据。这两个值相差不大，可以被忽略。但如果 *hw-frames* 有 2、3 或者 4 倍大于 *frames* ，你需要对无线连接进行故障排查。



我是否能在 Nstreme 连接下比较 frames 与 hw-frames?

frames 只是统计那些包含实际资料的内容。在 Nstreme 方式中，如果没有其他数据，仅 ACK 会能被一个独立的帧传输，ACK 帧是不会被添加到 **frames** 统计中的，但他们会出现在 **hw-frames** 如果双向传输达到最大速度（即，不会有 ack 帧情况），这时你不能比较 **frames** 与 **hw-frame**

如何设置发射功率（TX-power）?

tx-power 预设的最大 tx-power 值是无线网卡能使用的，且是从 EEPROM（电可擦只读存储器（Electrically Erasable Programmable Read - Only Memory））获取的值，如果你想使用更大的 tx-power 值，你可以设置该参数，但这个操作**需要由你自己负责**，很可能造成你的无线网卡的损坏！通常，使用这个参数来减低 tx-power。

通常 tx-power 控制应选择默认设置，修改预设的设置需要参考无线网卡的情况，但如果没有测试的情况下，大多可能造成距离和吞吐量的减低，可能会发生一下的情况：

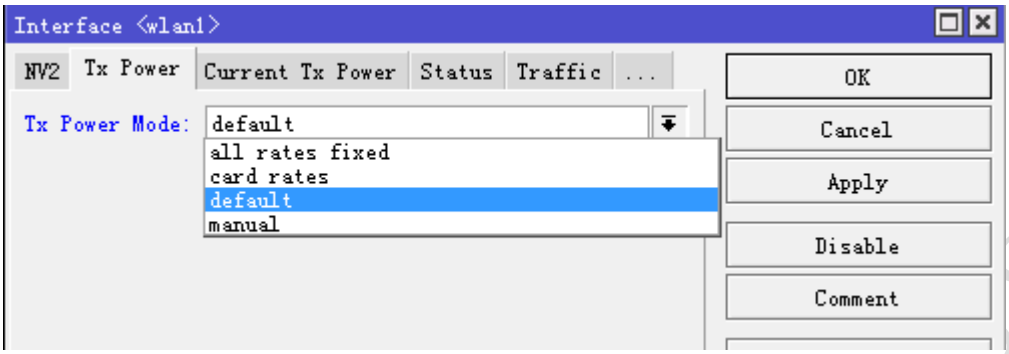
- 功率放大芯片过热，网卡将会出现低效率，并出现更多的数据误差；
- 放大芯片超负载，这样的情况出现更多的数据误差；
- 无线网卡过大的功耗，可能超出 3.3V 的供电，导致电压骤降，并出现设备重启或主板温度过高

什么样的 TX-power-mode 更好?

TX-power-mode 告诉无线网卡选择哪一种被使用，默认选择为 *default*.

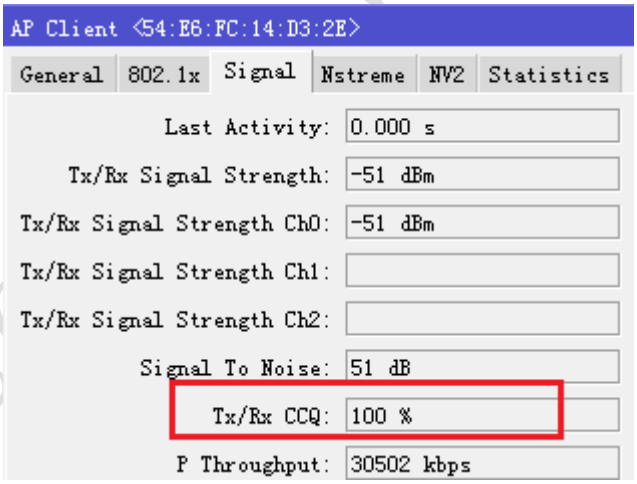
- **default** 即 tx-power 值从无线网卡的 EEPROM 中读取，并将忽略用户在 ***tx-power*** 里这种的参数
- **card-rates** 即通过用户指定 **tx-power** 值，并从无线网卡的 EEPROM 中获取的信息根据传输功率算法计算出不同速率下的 **tx-power** （速率越高功率越低）
- **all-rates-fixed** 即由使用者指定无线网卡将在任何速率下使用相同的 **tx-power** 值

注意：不建议使用'**all-rates-fixed**' 模式，因为无线网卡 tx-power 在高速率下会降低，如果强迫 tx-power 使用固定值，结果是在高速率下仍然会出现为修改之前的 tx-power 情况，甚至可能对网卡造成损害。在大多情况下，如果你想修改 tx-power 建议选择 ***tx-power-mode=card-rates*** ，并建议不要增加 tx-power。



什么是 CCQ，且如何判断这个值对无线网络的影响？

CCQ 客户端连接质量（Client Connection Quality）是一个百分比数值，显示有效的传输带宽，CCQ 值加权平均数 Tmin/Treal，他们通过每次传输的帧计算得出，Tmin 是多次在高速率下没有重新发送要传送的帧，Treal 是多次在实际情况下传输出的帧 ，即我们认为的**理论值**除以**实际值**，我们把 Tmin 看成理论值，即我们要多次传输的帧，把 Treal 看成实际发送帧的情况（当然实际情况会受到环境影响造成帧的多次重发），如果重发次数多了 Treal 就会大于 Tmin， 如果 CCQ 是 100%，说明理论发送和实际发送相等，传输链路正常，但当链路出现问题时，Treal 会大于 Tmin，那百分比就会降低。



什么是 adaptive-noise-immunity 设置?

Adaptive Noise Immunity (ANI) 自适应噪音免疫，即动态调整各种接收参数减低干扰，这个设置被添加到 Atheros AR5212 无线网卡和更高的网卡芯片中。

RouterOS 无线支持什么样的错误校正方式?

ARQ 方式能被 Nstreme 协议支持。普通的 802.11 标准协议不包含 ARQ - 损坏的重新发送的帧基于 ACK 协议。RouterOS 支持正向纠错与编码速率： 1/2, 2/3, 或 3/4.

增加一个放大器是否会提高我的无线连接速度？

这个需要考虑你的信号质量和噪音情况。记住你要获得一个好的无线链路，需要的是较低的发射功率和好质量的天线，有时候盲目使用放大器会增加噪音和造成无线链路的各种故障。

放大器会提高传输和接收的信号，因此在一个“安静”区域，仅你一个无线设备，且很少的噪音和其他无线设备，结果是你可以得到很好的效果。但另外一种情况，“拥挤”的区域，很多无线设备在工作，你将会增强信号，也同样会增多来至其他设备或者周边噪声源的信号，可能引起整体信号质量的降低。

你通常从 11b 无线信号上获得较好的信号，在 802.11g 协议下会产生更多的噪点，因此需要对有用的信号进行过滤。

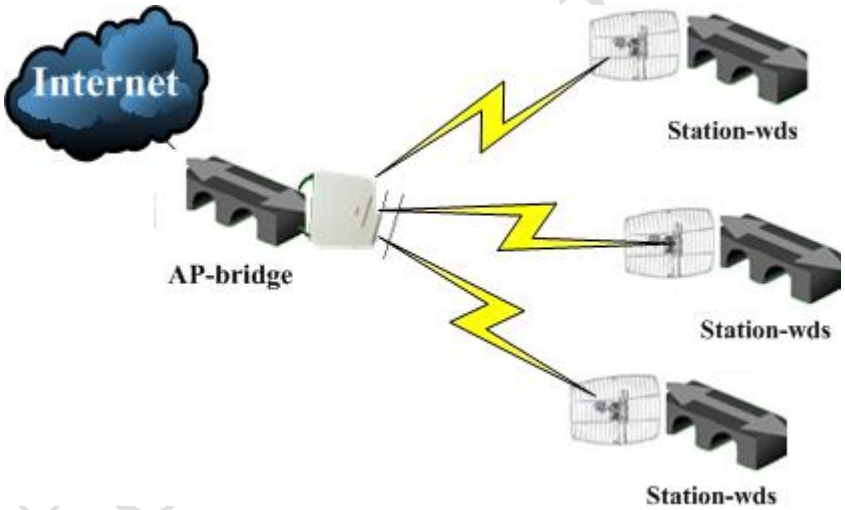
第八章 WLAN 点对多点与中继

在大型区域和城区的大型的无线网络构建中，会应用到更多中继和点对多点等方式的无线传输，在 Wlan 无线中继和点对多点的无线传输有以下两种事例，桥接和路由两种网络构建模式：

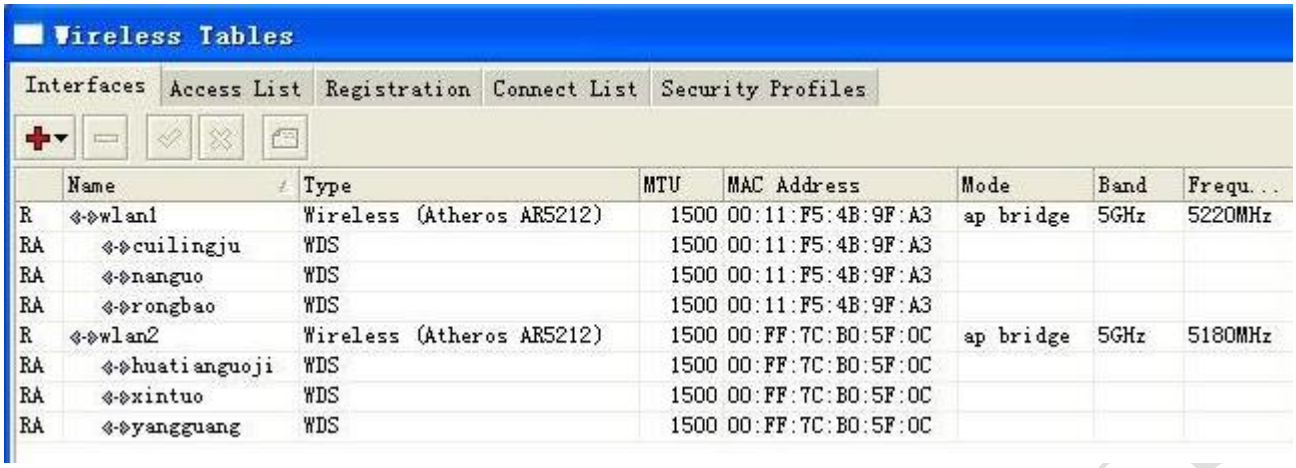
- 桥接模式：通过在设备上建立 bridge，在 Wlan 无线网络中实现二层数据链路的传输，这种方式适合于小型的无线网络建设。
- 路由模式：通过在设备上建立多级 IP 地址路由，通过传输 IP 数据报连接各个设备和终端。这样的模式更适合于大型的无线网络组建。

8.1 桥接模式的点对多点

桥接模式的点对点与中继，适用用于二层数据的透传，对于网络组建和网络传输协议更改较容易，桥模式透传适合于小型的无线网络。基于点对点 WDS 桥模式，即 1 个中心 ap-bridge 设备通过 1 个大扇角平板天线覆盖周围的区域，远程多个 station-wds 设备通过定向天线进行连接，如下图：



中心点配置为 ap-bridge 的桥接，其他 3 个终端点配置为 station-wds 模式的桥接，这样的配置和普通 AP-Bridge to Station-WDS 模式完全一样，只是增加了多个 station-wds 终端机站，即点对点传输的一种演变。下图是一个双网卡的点对多点的连接状态图，wlan1 和 wlan2 都分别连接了 3 个 station-wds 的终端设备：



	Name	Type	MTU	MAC Address	Mode	Band	Frequ...
R	wlan1	Wireless (Atheros AR5212)	1500	00:11:F5:4B:9F:A3	ap bridge	5GHz	5220MHz
RA	cuilingju	WDS	1500	00:11:F5:4B:9F:A3			
RA	nanguo	WDS	1500	00:11:F5:4B:9F:A3			
RA	rongbao	WDS	1500	00:11:F5:4B:9F:A3			
R	wlan2	Wireless (Atheros AR5212)	1500	00:FF:7C:B0:5F:0C	ap bridge	5GHz	5180MHz
RA	huatianguoji	WDS	1500	00:FF:7C:B0:5F:0C			
RA	xintuo	WDS	1500	00:FF:7C:B0:5F:0C			
RA	yangguang	WDS	1500	00:FF:7C:B0:5F:0C			

桥模式点对多点的特点：

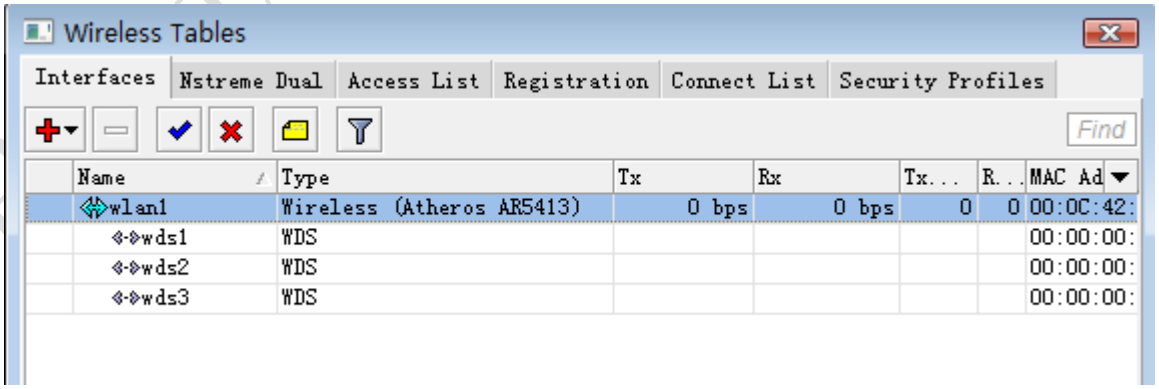
1、 在一个 90 度天线能够覆盖的扇形区域内有多个终端点，可以通过点对多点的方式连接。通常中心基站采用 5G 90 度的平板天线，终端设备采用抛物面的定向天线与中心连接，根据具体情况，能在 10-15 公里左右正常连接。

2、 桥模式的点对多点架设和配置简单，新增节点无需增加或者修改网络参数，适用于 PPPoE 拨号认证和各种三层网络的传输认证。

桥模式点对多点的缺点：这样的点对多点，是通过桥模式连接，所以二层广播数据会在各个基站之间相互传输，降低网络效率，特别是在网络不断扩大的情况下。通常我们可以通过 bridge filter 来限制各个 wds 接口的广播数据。

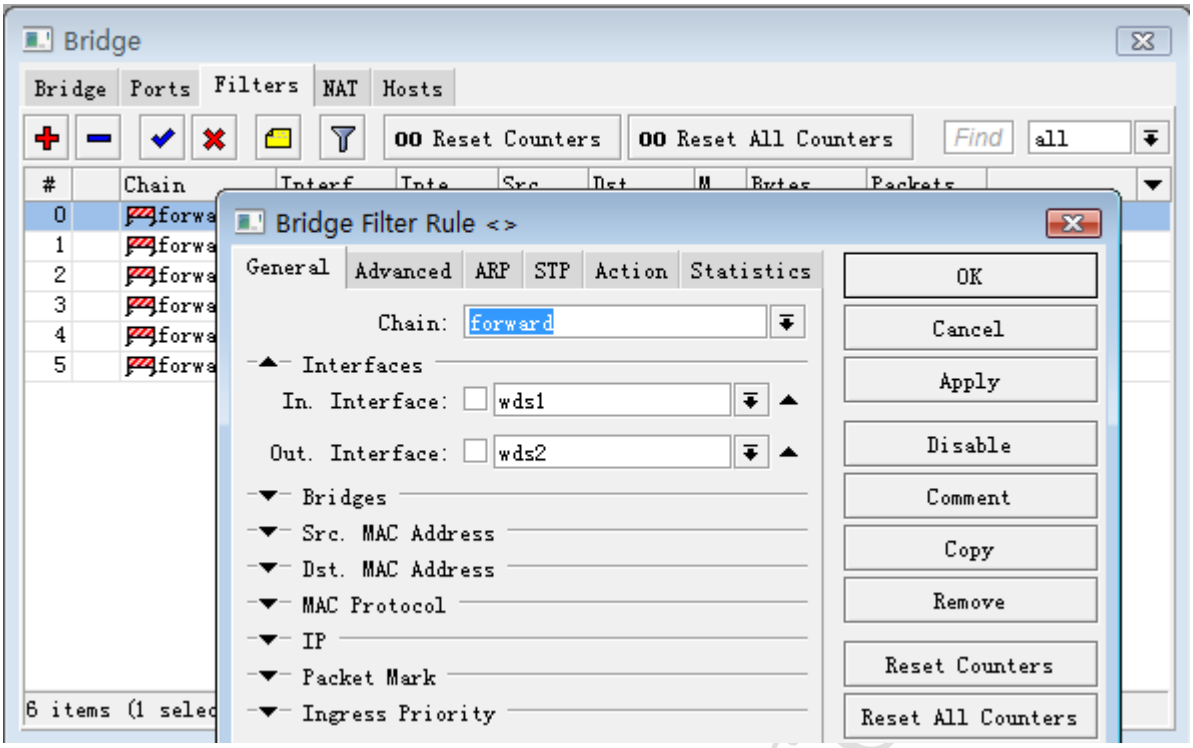
8.2 桥接 WDS 的端口隔离

这里我们假设有一个 wlan1 通过点对多点的桥接连接了 3 个 station-wds 的终端设备，分别是 wds1、wds2 和 wds3。由于 3 个 wds 桥接设备之间会互相广播二层数据，甚至某些客户中了 arp 病毒也会通过 wds 桥接攻击到其他网络的用户，我们需要通过/bridge filter 隔离 wds1、wds2 和 wds3 的数据。

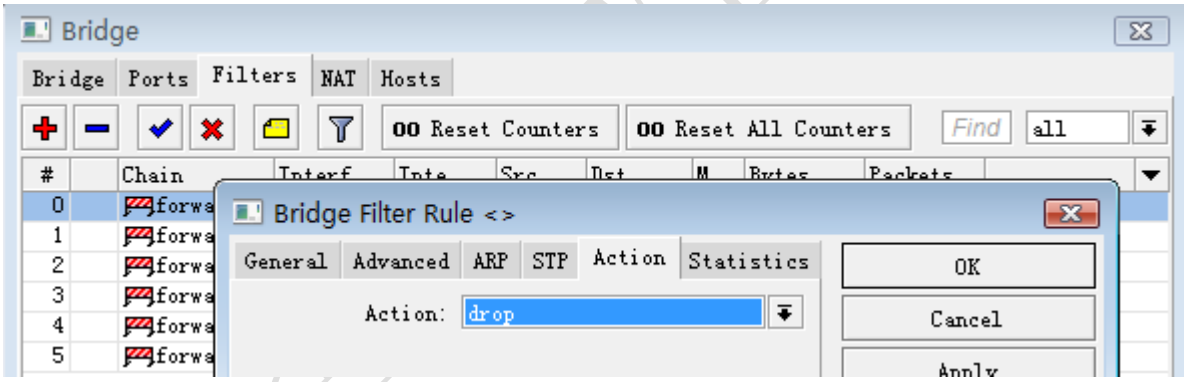


	Name	Type	Tx	Rx	Tx...	R...	MAC Ad
	wlan1	Wireless (Atheros AR5413)	0 bps	0 bps	0	0	00:0C:42:
	wds1	WDS					00:00:00:
	wds2	WDS					00:00:00:
	wds3	WDS					00:00:00:

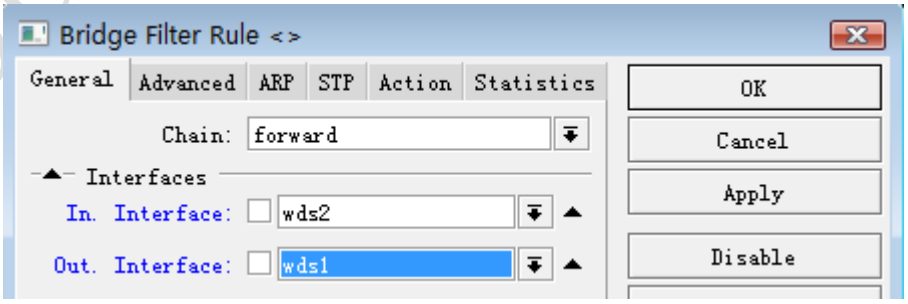
我们进入 bridge filter，通过添加规则，设置 in-interface（数据进入接口），out-interface（数据发出接口），如下图数据进入接口是 wds1，发出接口是 wds2，由于是通过桥转发的，所以我们选择 chain=forward：



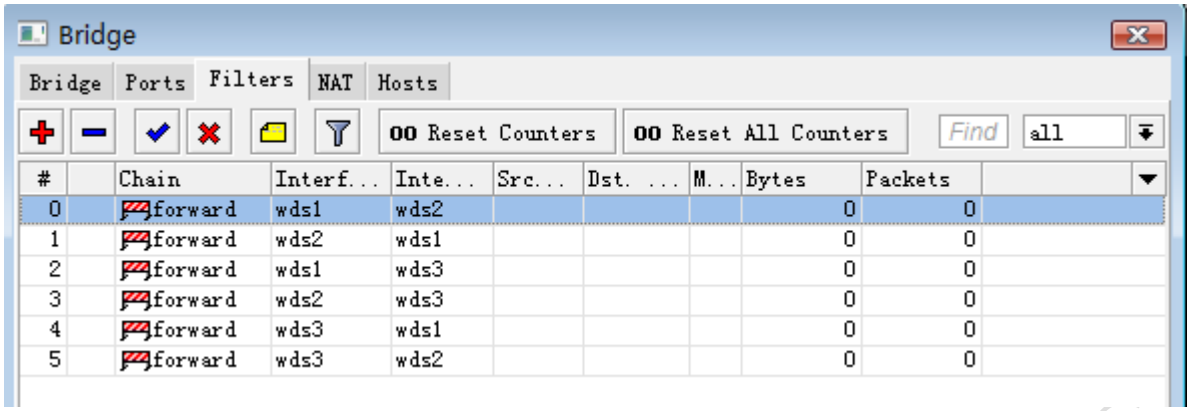
接下来我们配置 action=drop 丢弃他们之间转发的数据：



由于数据传输是双向的，所以我们需要反向填写 in-interface 和 out-interface，同样需要将 action 设备为 drop，如图：



一共有 3 个 wds 接口，每对接口需要配置 2 条规则，所以一共需要配置 6 条过滤规则限制数据广播：



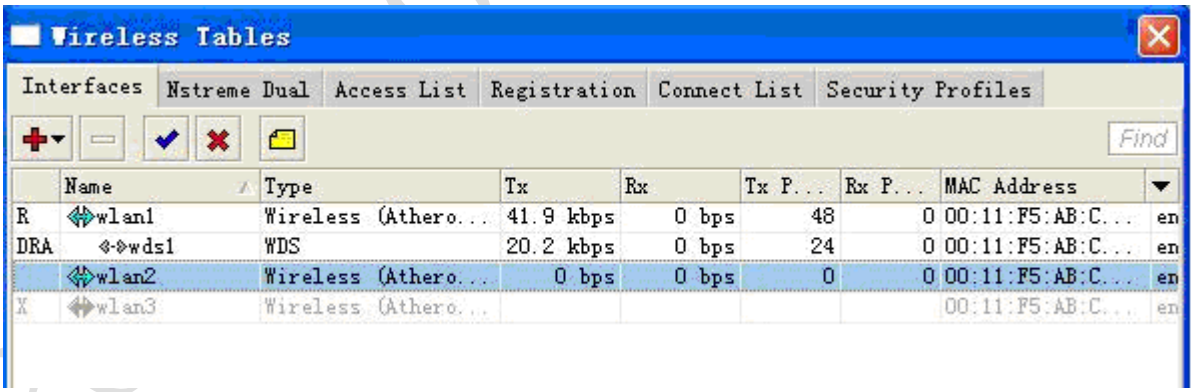
通过以上配置可以限制多个 wds 接口之间的数据广播，一定程度上提高 wlan 的无线网络传输效率，但不能根本性的解决。（关于更多的 bridge 过滤规则，请参考**《RouterOS 中文网络教程》**）

8.3 桥接模式的中继

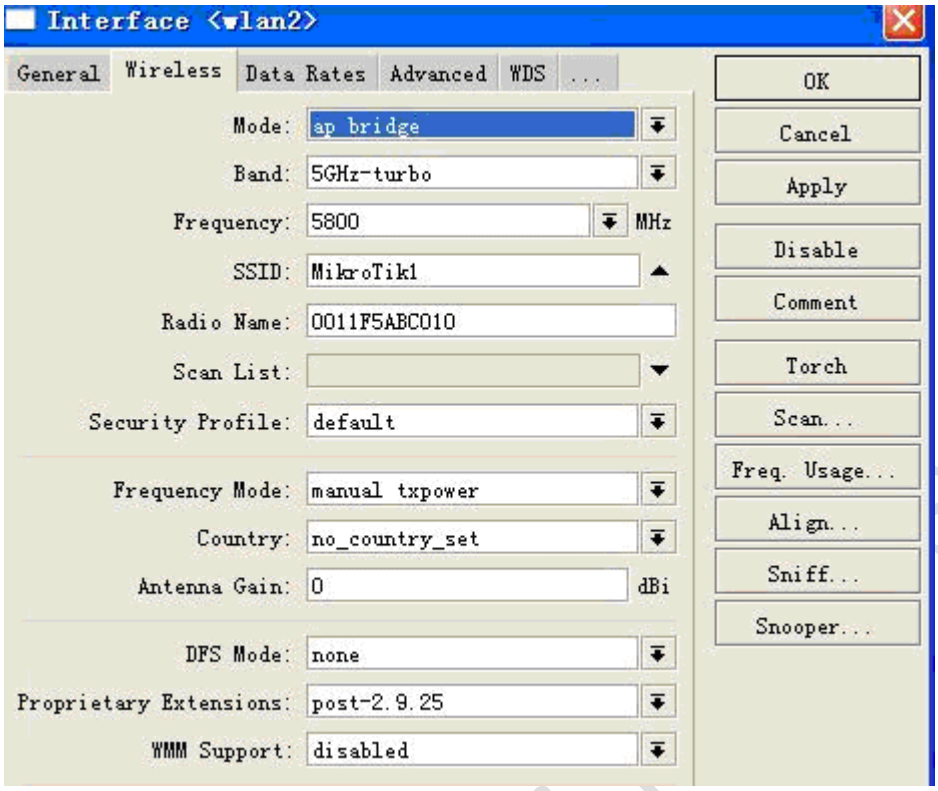
wlan 无线中继是指在一个设备上添加两张或两张以上的网卡，做中继传输，他和点对点或点对多点不同的地方在于，将 2 个周边的网络进行数据中转，这样的情况会有如下：

- 两个基站之间由于视距阻挡，不能直接连接需要采用中继的方式；
- 两个基站之间由于距离太远，信号衰减较大，需要中继放大；
- 两个基站之间由于其他特殊原因无法直接连接。

将两个无线连接做透明的桥接传输，这里我们用两张网卡做事例，我们在原有的设备上增加了一张 wlan2 的网卡，与 wlan1 做桥接实现中继：

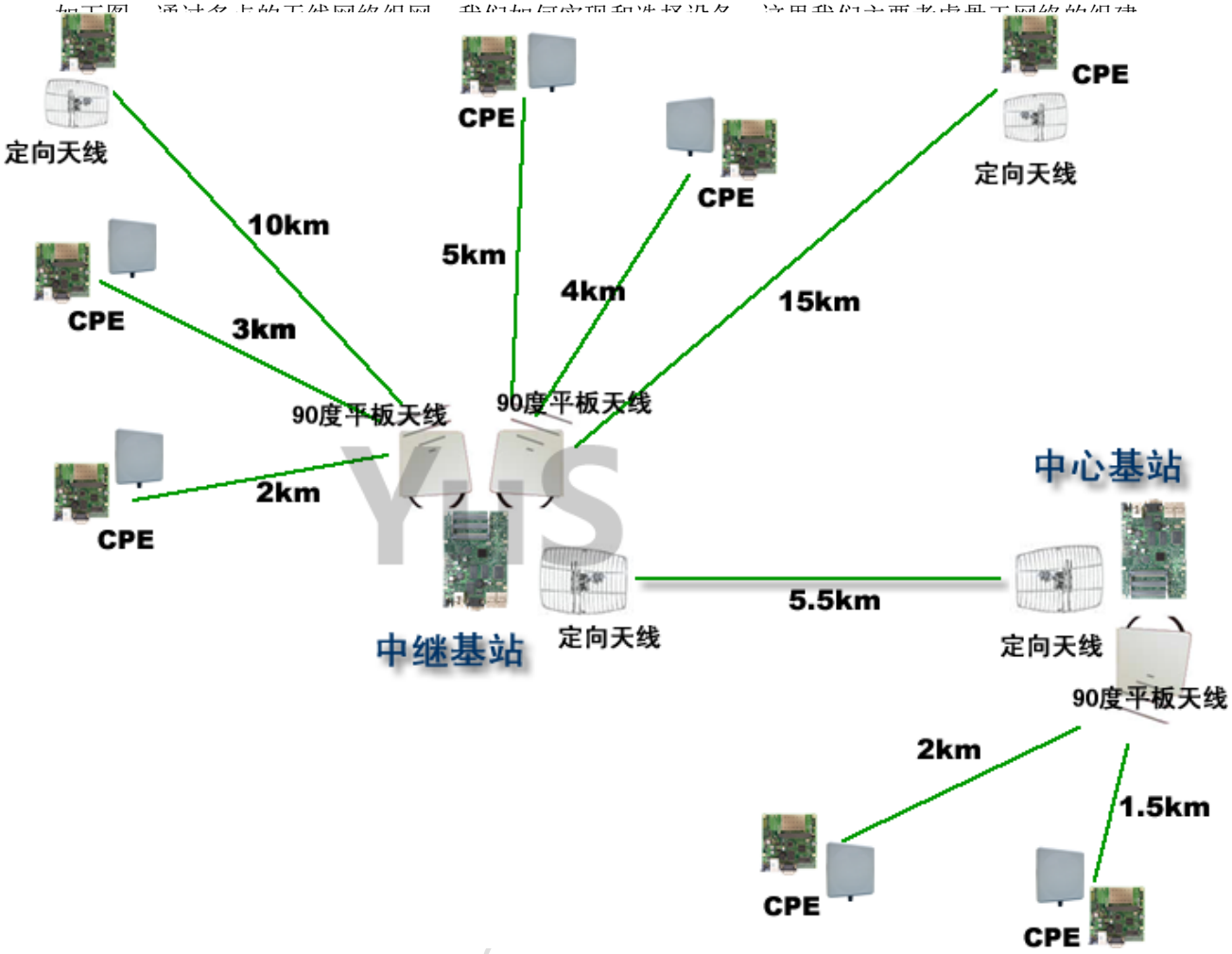


我们设置 wlan2 的参数为 Mode 为 ap-bridge，Band 为 5G-turbo，频率为 5800，SSID 为 MikroTik1



在 bridge1 中添加 wlan2 的界面，同样在 WDS 选项我们把 WDS Mode 为 dynamic（动态方式），WDS Default Bridge 为 bridge1，这样 wlan1 与 wlan2 被系统自动添加到 bridge1 中，中继的桥接设置就实现了，其实配置和普通的 wds 模式桥接完全一样，只是将 2 个无线模块都放到一个 bridge 中。

8.4 桥接模式点对多点和中继的综合应用



所有的 Wlan 无线设备采用 5G 频段做为骨干传输，5G 频段干扰小，传输效率稳定，设备的结构组成如下：

终端点 CPE：全部采用 RB411 或者 RB711 型号，配置 5G 定向抛物面天线，用于远距离的接收。最高能提供 45Mbps 的 TCP 带宽，如果说 RB711 采用 11n 的 Nv2 协议可以达到 93Mbps。

中心基站：为数据接入的中心节点，这个点为了得到更好的数据和传输，我们采用 RB433（300MHz 处理器）或 RB433AH（680MHz 处理器）型号的无线设备，如果考虑高处理能力可以考虑 RB800，全千兆设备。因为该点需要和中继点和两外两个终端接入点连接，所以采用双网卡发射信号，同时支持 2 个 5G 信号输出，分别通过一个定向天线与中继点连接，一个用 90 度的平板天连接两个终端点 RB411 或者 RB711。

中继基站：采用 RB433AH 型号设备，安装 3 张无线网卡，配置一个连接信号点的 RB433AH 的设备。另外两张连接 2 个 90 度平板天线，对下面的 6 个点分别做点对多点的连接。

注：一个无线网卡可以支持 2007 个无线客户端，但实际环境中可以连接的实际用户在 20-30 个，因为当客户端多后会形成相互的无线信号干扰。而这里我们采用一个平板连接 3 个客户端，原因是保证每个客户端的带宽，因为每个客户端都会平分带宽，在每个客户端的信号强度和环境适宜，中心点能提供最大 60Mbps 的带宽情况下，3 个客户端每个可以获取 20Mbps 的带宽。

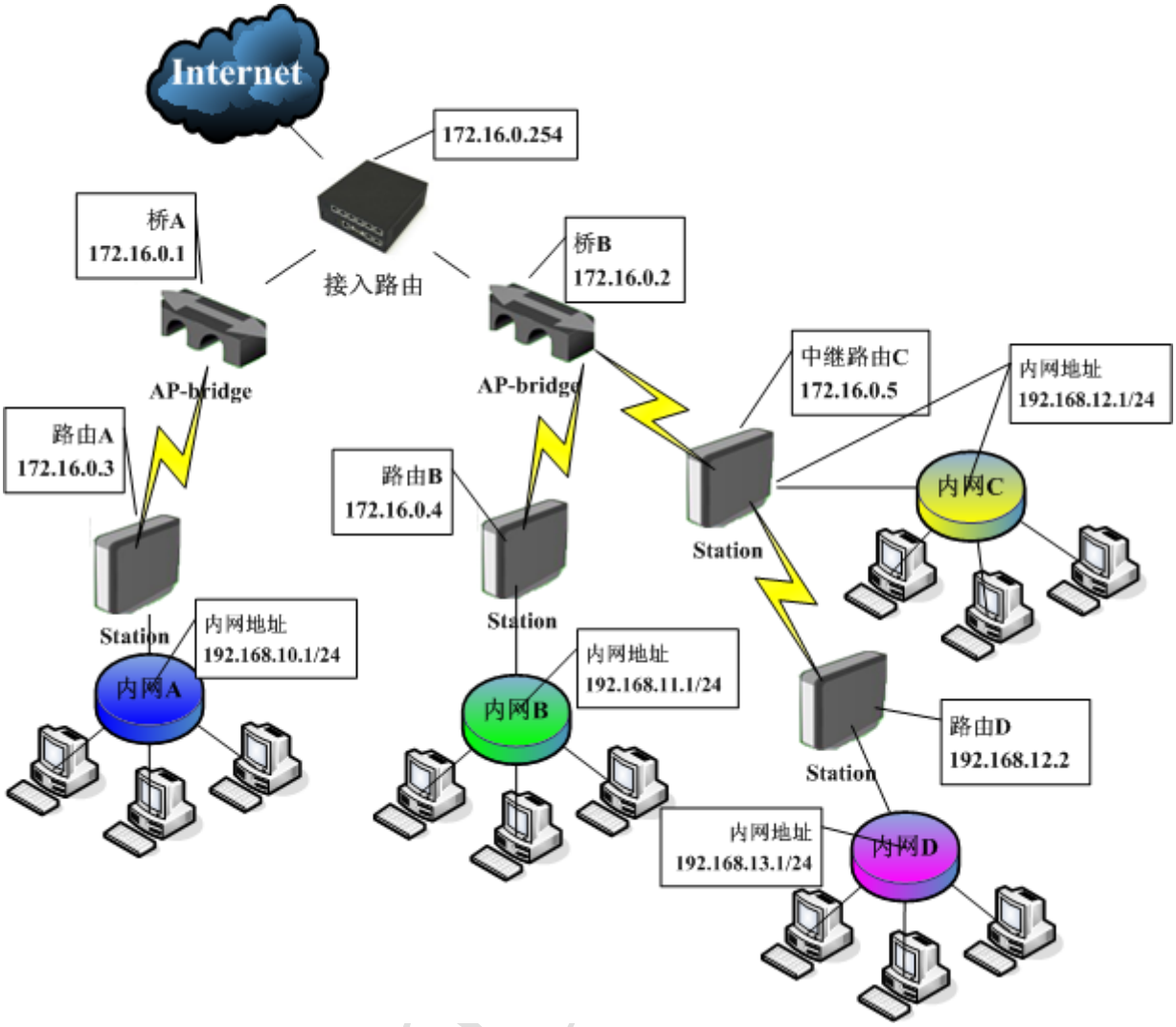
注：在点对多点的方式下同样采用桥接的方式，中心 AP 的设置和点对点的 AP1 设置是完全一样的，至于远程的多个 station 客户的配置也是和点对点的 AP2 一样设置为 station-wds。

点对多点与中继下可以选用的模式如下表：

模式	参数	Nstreme 协议	Nv2 协议
AP-Bridge to Station	路由模式，启用桥接需要配置 EoIP	支持，取决于网卡速率和选择模式，最高 108M	暂不支持
AP-Bridge to Station-WDS	桥接模式，自动添加到桥接设置中	支持，取决于网卡速率和选择模式，最高 108M	支持，11n 支持近 200Mbps 带宽
AP-Bridge to AP-Bridge	支持桥接和路由模式，桥接模式自动添加到桥接设置中，同样支持 MESH 和 WDS 模式	不支持，取决于网卡速率和选择模式，最高 108M	不支持
Bridge to Bridge	桥接模式，自动添加到桥接设置中	支持，取决于网卡速率和选择模式，最高 108M	不支持，
bridge to station-bridge	桥接模式，自动添加到桥接设置中	支持，取决于网卡速率和选择模式，最高 108M	支持，11n 支持近 200Mbps 带宽

8.5 路由模式的 wlan 网络

路由模式的 wlan 网络构建要比 wds 桥模式的无线网络较为复杂，但适合于大型 wlan 无线网络的构建，特别是 WISP 的运营，这样的模式通过多级路由实现，能将二层的广播限制在一个路由范围内，降低网络效率。如下图的一个结构：

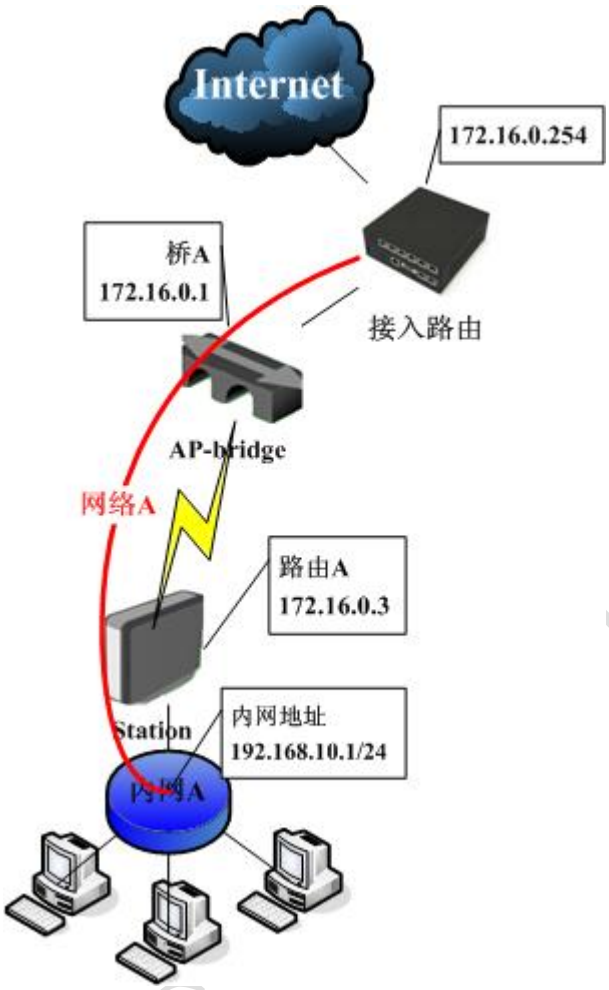


这样的无线网络要求在接入路由和各级节点设备时配置路由表，随着设备和网络的不断扩大路由表的维护也同样增加，这样的网络构建我们通常使用 ap-bridge to station 模式，ap-bridge 采用透明桥连接，而 station 则是配置路由模式。这样网络构建可以看作多个点对点的 ap-brdge to station 模式的组合模式。

在这样的网络环境需要在接入路由器上配置路由表，首先我们通过分步骤来了解网络拓扑，

步骤 1：

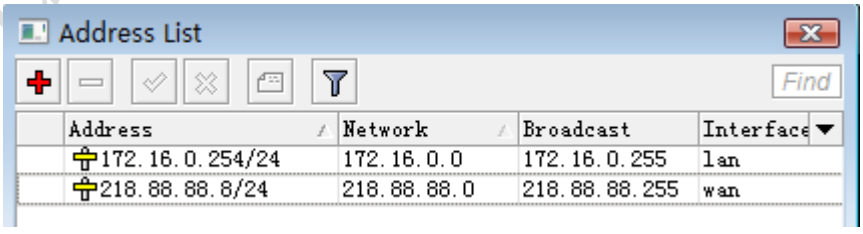
网络 A 的用户需要访问 Internet 就必须经过接入路由，这里从接入路由开始一共有 3 个设备，分别是接入路由、桥 A 和路由 A，如下图：



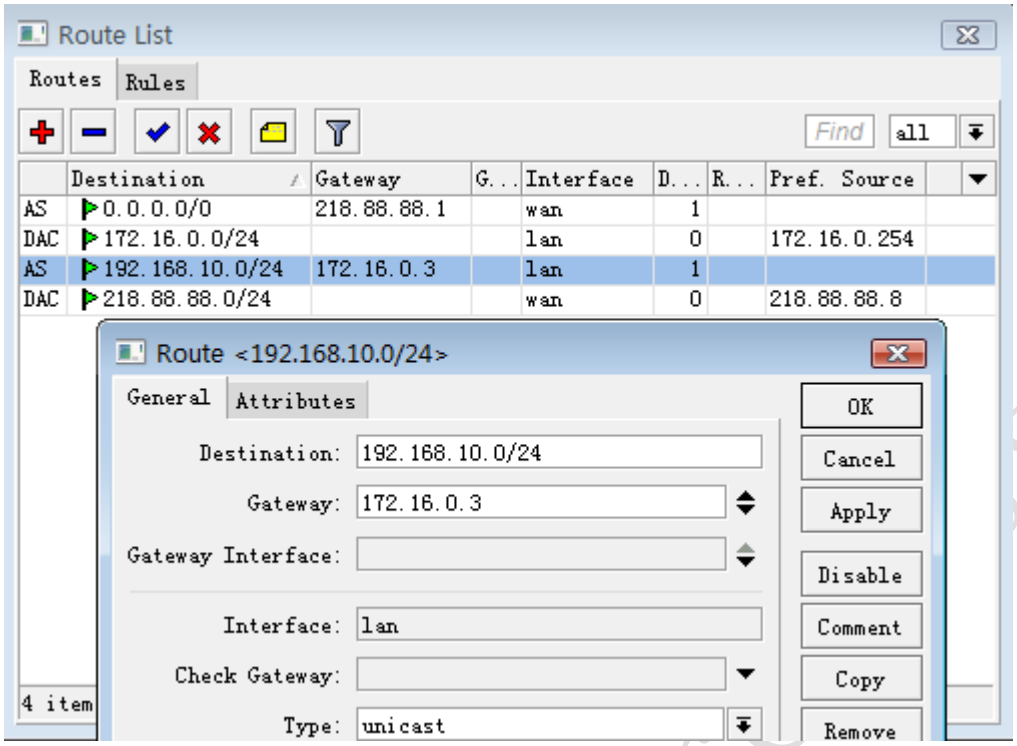
在这个网络中**桥 A** 是一个二层的网桥设备，对于三层来说他完全是透明的，可以不用考虑路由问题，那在这个网络结构中，我们只需要考虑 2 个设备即**接入路由**和**路由 A**。在这两个设备分别由 2 个网段组成，分别是 172.16.0.0/24 和 192.168.10.0/24，我们需要通过配置路由表让数据能正常连接：

接入路由配置：

首先配置接入路由的 IP 地址，wan 口的外网连接 IP 地址是 218.88.88.8/24，内网连接用的 IP 地址是 172.16.0.254/24，在/ip address 配置如下：

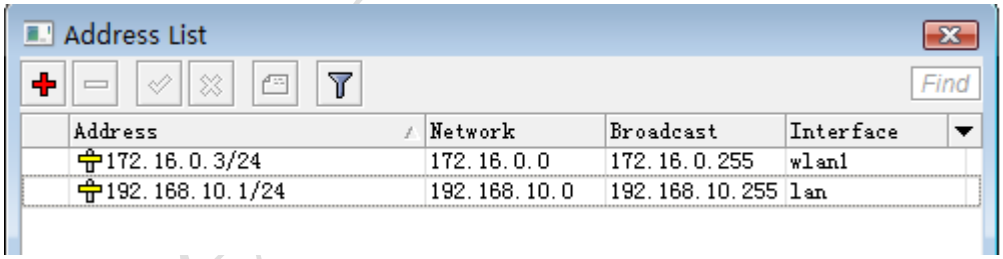


接下来我们需要配置，让接入路由知道 192.168.10.0/24 这个网段需要经过那条线路可以到达，由于 192.168.10.0/24 网络在**路由 A：172.16.0.3** 后，即配置一条目标地址路由 dst-address=192.168.10.0/24 gateway=172.16.0.3：

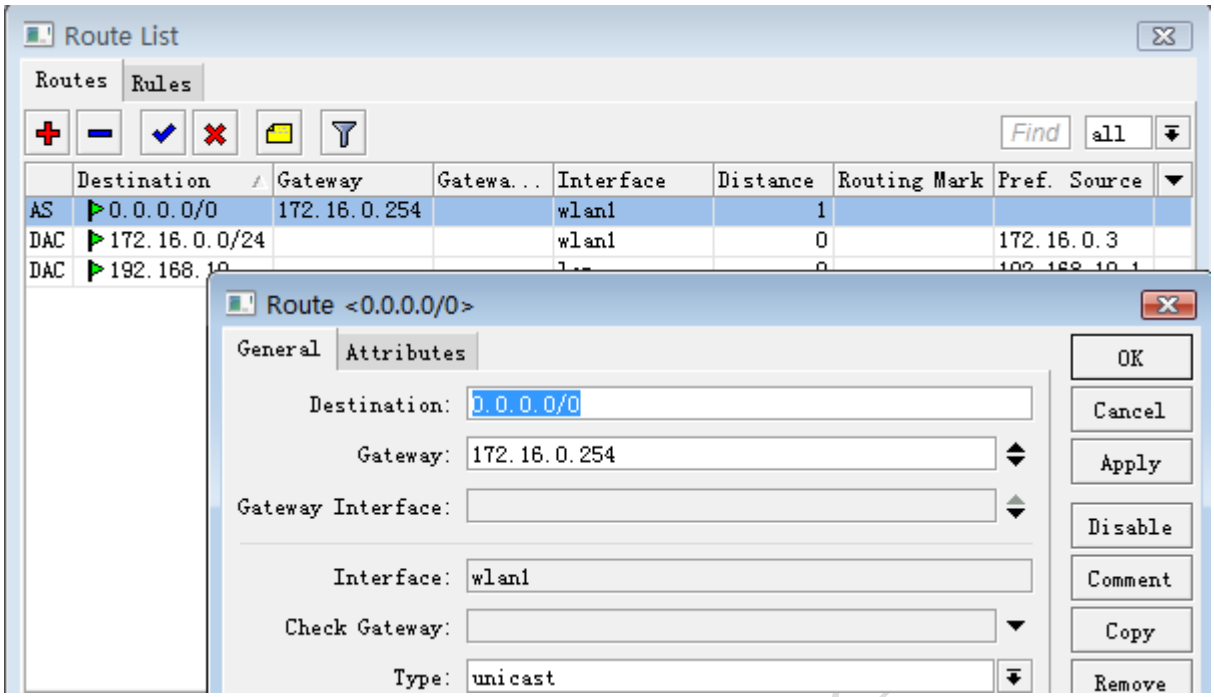


路由 A 配置：

路由 A 我们用 192.168.10.1 做为内部网关的 IP 地址，连接接入路由的 IP 是 172.16.0.3，IP 地址配置如下：

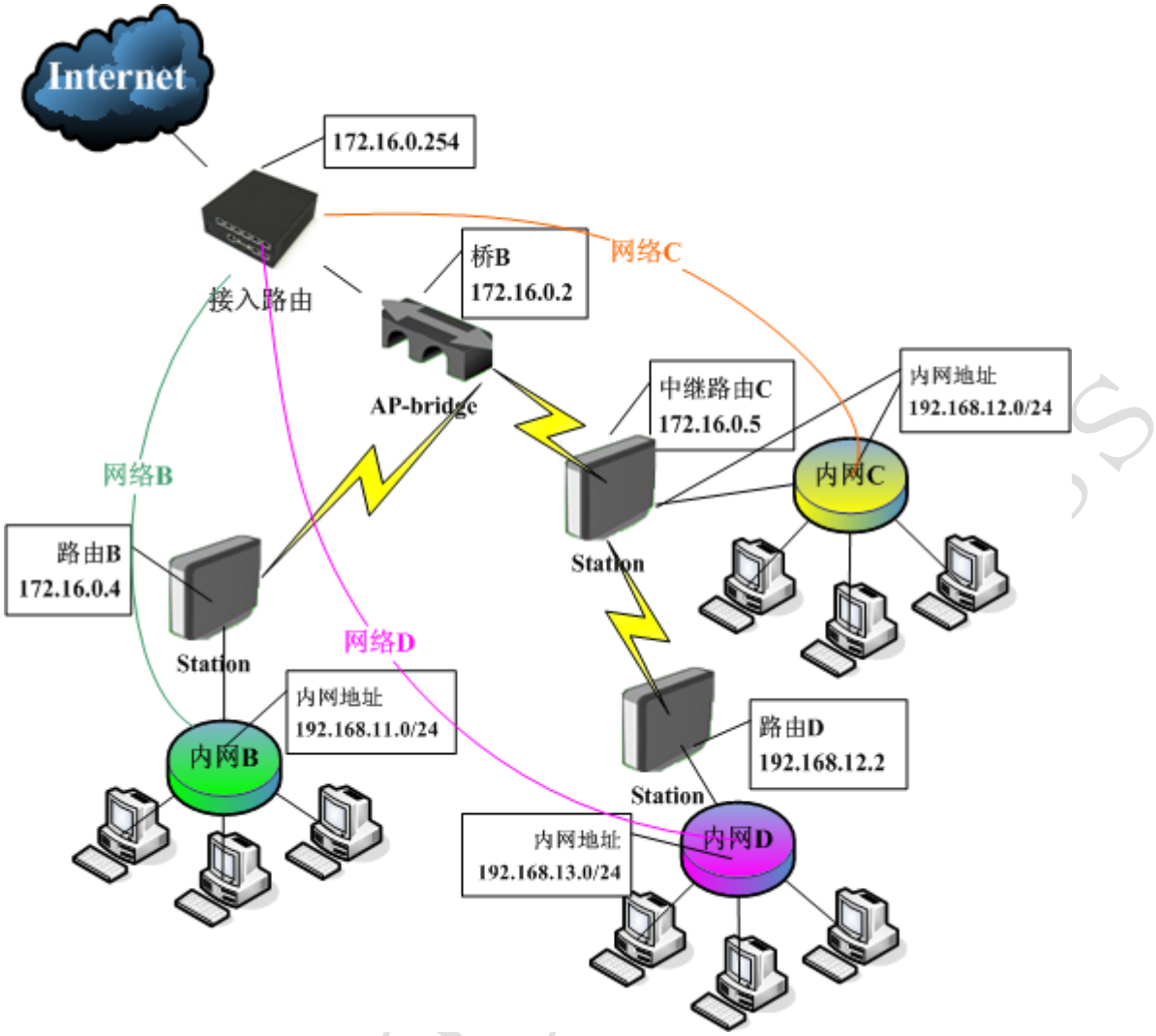


接下来配置路由 A 的网关，因为只需要让 192.168.10.0/24 内网络的用户直接上网，我们可以直接将默认网关指向接入路由 172.16.0.254



步骤 2：

现在我们来看看**网络 C** 和**网络 D**，这里同样是要将两个网络连接到 Internet，但这个网络比之前的步骤 1 要复杂一点，因为这里是串联了 2 级的网络，**桥 B** 通过点对多点分别连接了**路由 B** 和**中继路由 C**，而且在**中继路由 C** 下连接了**路由 D** 的设备，从接入路由开始一共连接了 6 台设备。如下图：



网络中**桥 B** 为网桥，看作透明设备，我们只需要配置**接入路由、路由 B、中继路由 C** 和**路由 D** 的路由表，我们通过下面的表看看：

接入路由关于这三个网络的路由表

目标地址	网关	界面	备注
192.168.11.0/24	172.16.0.4	Lan	指向路由 B
192.168.12.0/24	172.16.0.5	Lan	指向中继路由 C
192.168.13.0/24	172.16.0.5	Lan	指向中继路由 C（因为 192.168.13.0/24 段在中继路由后）

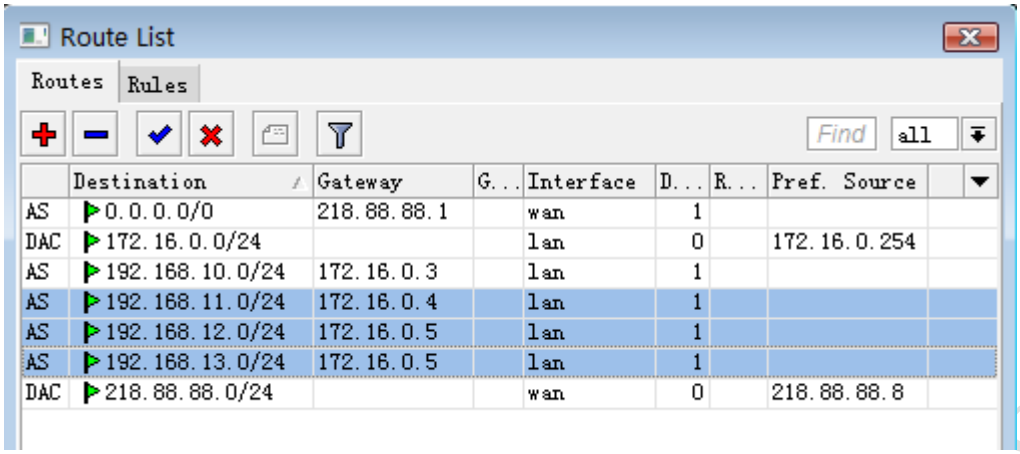
中继路由 C 的路由表

目标地址	网关	界面	备注
0.0.0.0/0	172.16.0.254	Wlan1	到接入路由的默认网关
192.168.13.0/24	192.168.12.2	Wlan2	指向路由 D

而对于**路由 B** 和**路由 D** 只需要将默认网关指向上一级路由的网关 IP 即可，如步骤 1，因为他们是最后一级设备，不需要配置其他的路由表。

接入路由

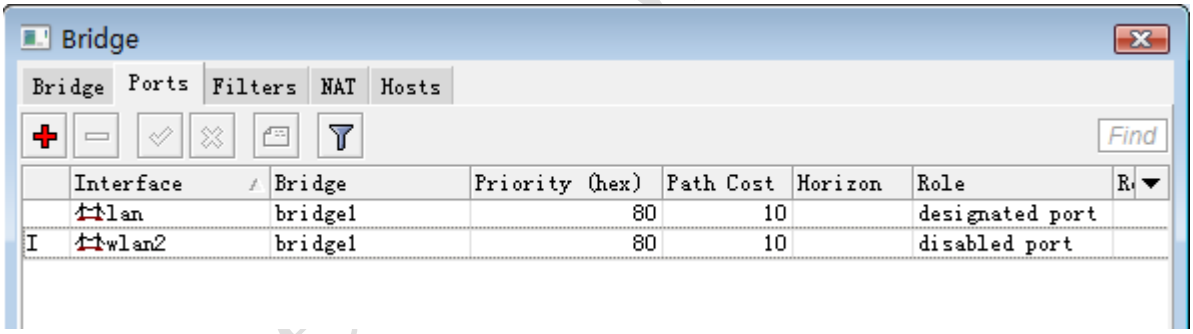
在接入路由上我们只需要配置三条目标地址路由，如下图：



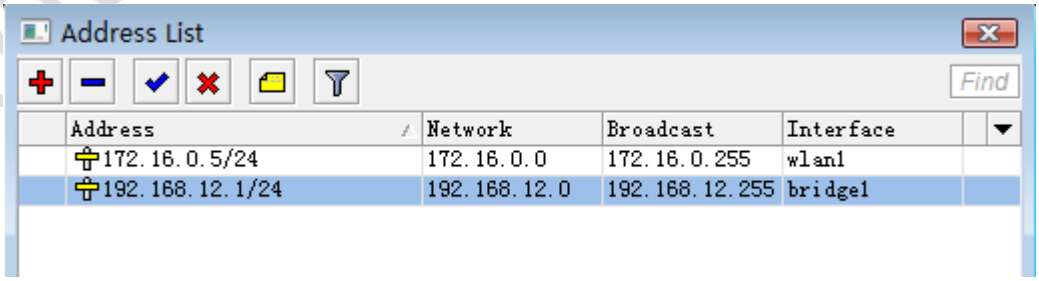
中继路由 C

这里我们增加了一个中继路由，这个路由和以往的路由配置有点不同，中继路由我们采用的 2 张无线模块，即 1 张用于连接**桥 B** 的网络配置为 station 模式，而另外一张即要和路由 D 连接配置为 ap-bridge 模式，又要和以太网卡做桥接。

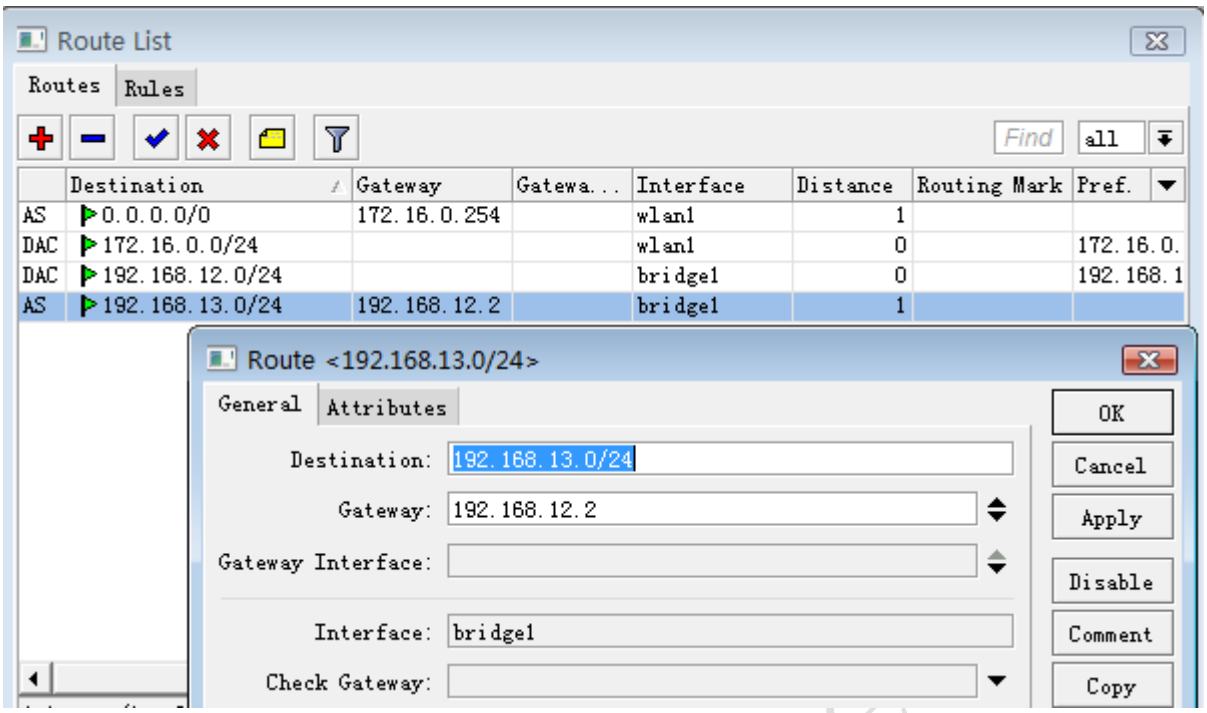
在中继路由 C 中我们用到 2 张无线模块，1 个以太网卡，wlan1 用于连接桥 B，wlan2 用于连接路由 D，而以太网卡则和中继路由 C 下的网络 C 相连接。由于 wlan2 和以太网卡属于一个局域网内，所以我们将他们放到一个桥中，由于 wlan2 使用的是 ap-bridge 模式，所以支持桥接模式，添加桥后配置如下（无线部分请参考之前的 ap-bridge to station）：



接下来我们配置 IP 地址，中继路由 C 的 wlan1 的 IP 地址为 172.16.0.5/24，内网连接路由 D 和网络 C 的地址是 192.168.12.1/24，我们将 wlan2 和 lan 口放到一个桥中，所以将内网 IP 设置道 bridge1 上：

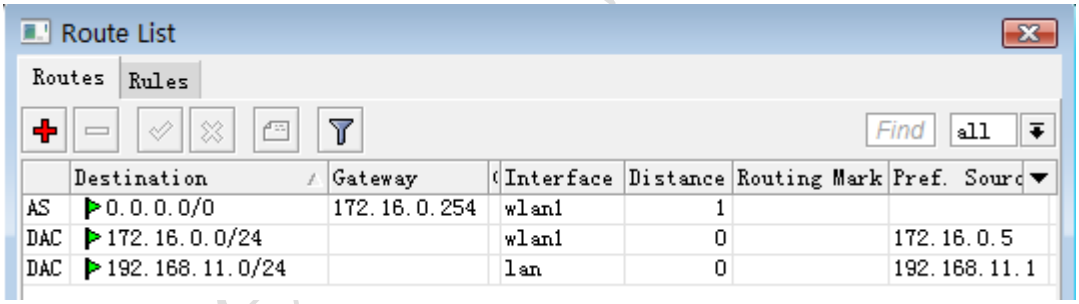


接下来我们配置路由部分，添加 dst-address=192.168.13.0/24 gateway=192.168.12.2：

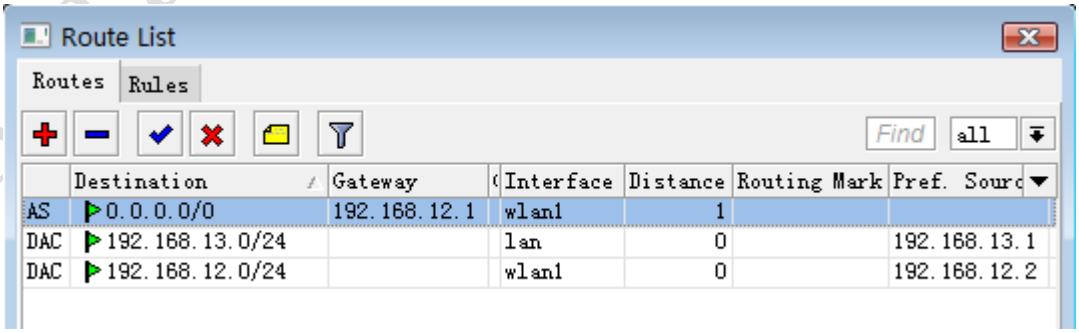


路由 B 和路由 D

添加完相应的 IP 地址后，路由 B 和路由 D 的配置和之前的路由 A 是完全相同的，只是路由 B 与路由 D 所指向的网关不同，路由器 B 指向的网关为 172.16.0.254，因为他的 wlan1 与接入路由在同一网段，如图：



路由 D 因为在路由 C 的 192.16.12.0/24 的网段下，所以需要将网关指向与他同一网段的 192.168.12.1



第九章 MikroTik Mesh 无线网状网络

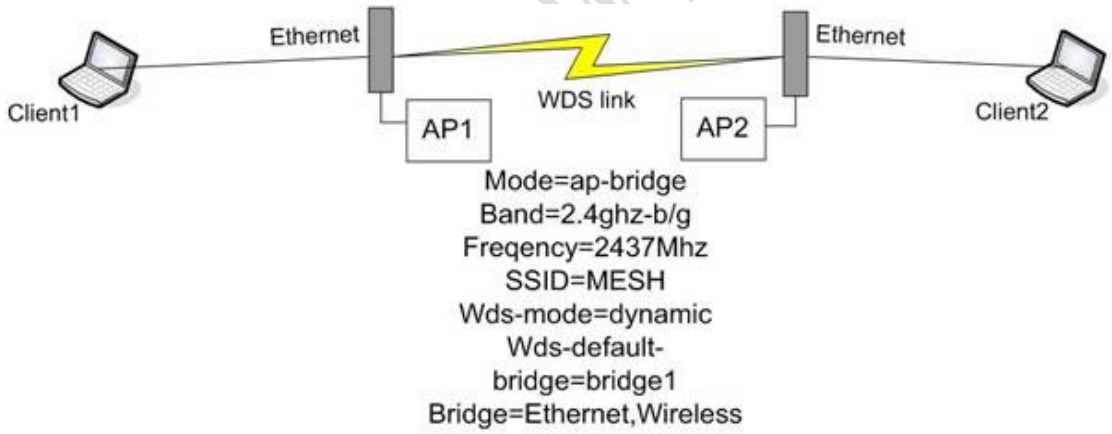
MikroTik 支持的 Mesh 无线网状网络构建可以分为两种：

一、基于 STP 的 Mesh：STP（Spanning Tree Protocol）是生成树协议的英文缩写。该协议可应用于环路网络，通过一定的算法实现路径冗余，同时将环路网络修剪成无环路的树型网络，从而避免报文在环路网络中的增生和无限循环，但是它还是有缺点的，STP 协议的缺陷主要表现在收敛速度上。了解决 STP 协议的这个缺陷，定义了快速生成树协议 RSTP（Rapid Spanning Tree Protocol）。RSTP 协议在 STP 协议基础上做了重要改进，使得收敛速度快得多（最快 1 秒以内）。通过 RSTP 构架 Mesh 网络，通过路径成本和优先级计算优化整个无线网络。

二、HWMP+协议的 Mesh：基于来至 IEEE802.11s 草案 Hybrid Wireless Mesh Protocol (HWMP)，能用于替代 STP 生成树协议确保环路的最优路径。 HWMP+ 协议并不能兼容 HWMP 的 IEEE 802.11s 草案。

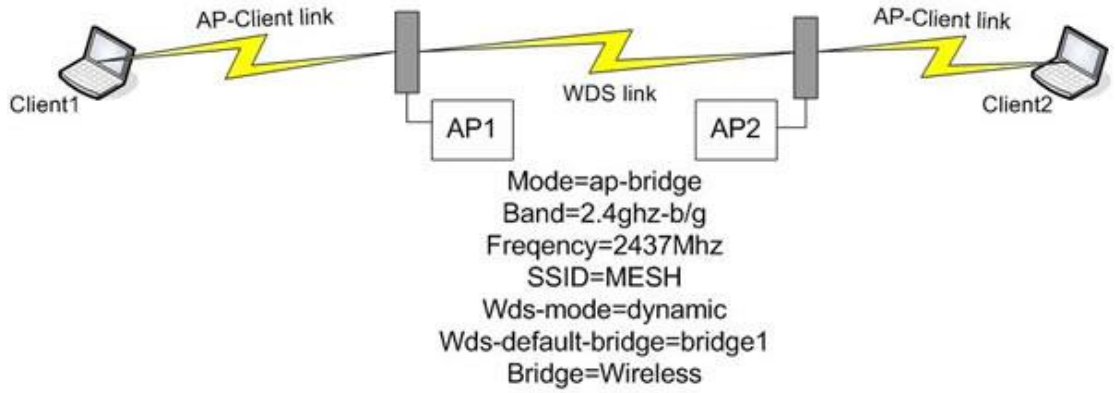
9.1 MikroTik 无线网状网络的构建

我们如何理解无线网状网络（Mesh）,首先我们通过下面的拓扑图来具体了解一下 Mesh 的概念。首先我们来看看建立一个点对点的无线传输：



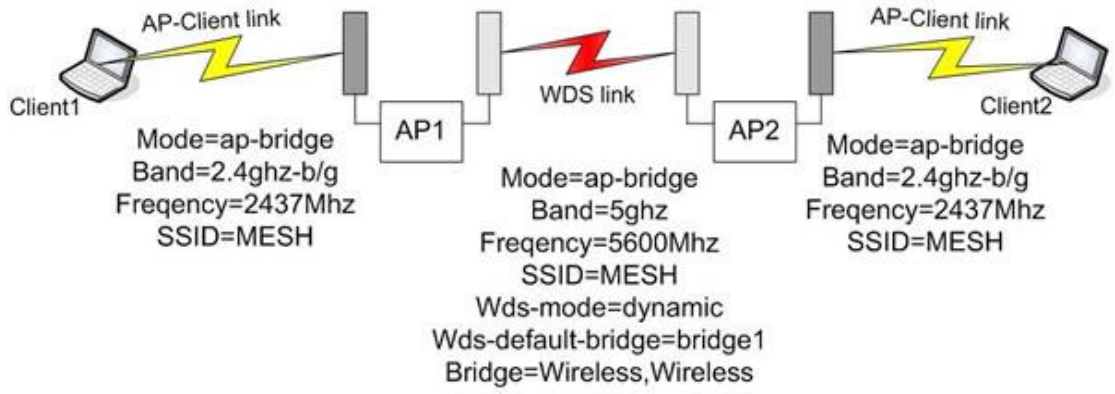
上图是通过 AP1 和 AP2 使用 WDS 模式将两个远程的以太网络桥接起来，这样 Client1 和 Client2 能直接通过二层互访。

可能因为两个以太网传输或者客户端需要，我们将以太网络该无无线网络：



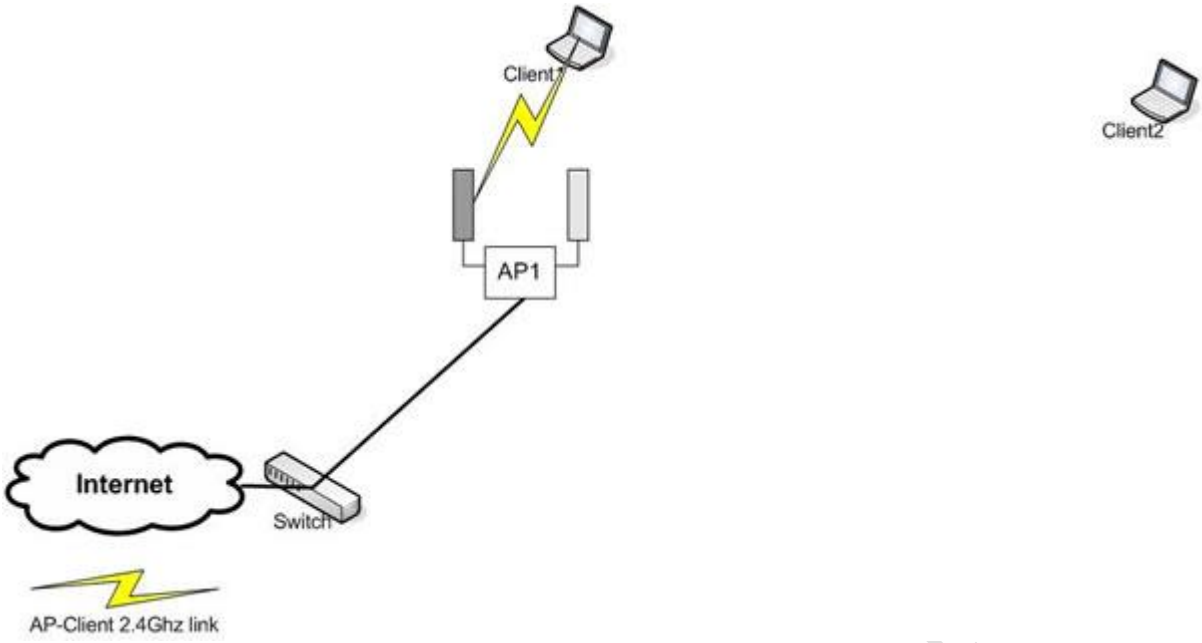
上图我们同样使用 WDS 模式，并通过无线中继的方式或者称为无线跳跃方式，通过 AP1 和 AP2 采用扇形或者全向天线中继信号，将 Client1 和 Client2 连接起来。这样我们通过无线方式替代了，有线的以太网络。如果相对于复杂的施工环境来说，在无线方面相对于有线更加方面和快捷，甚至在某些情况下无线更加的节约成本。

在上面的方案中，我们的两个 AP 都使用的是一张网卡，相对来说一张网卡做转发效率并不高，在大数据情况下延迟会明显增加，为了改善这样的情况，我们在每个 AP 上多增加一个网卡，一张做 AP 间的传输，一张做客户端的覆盖。如下图：

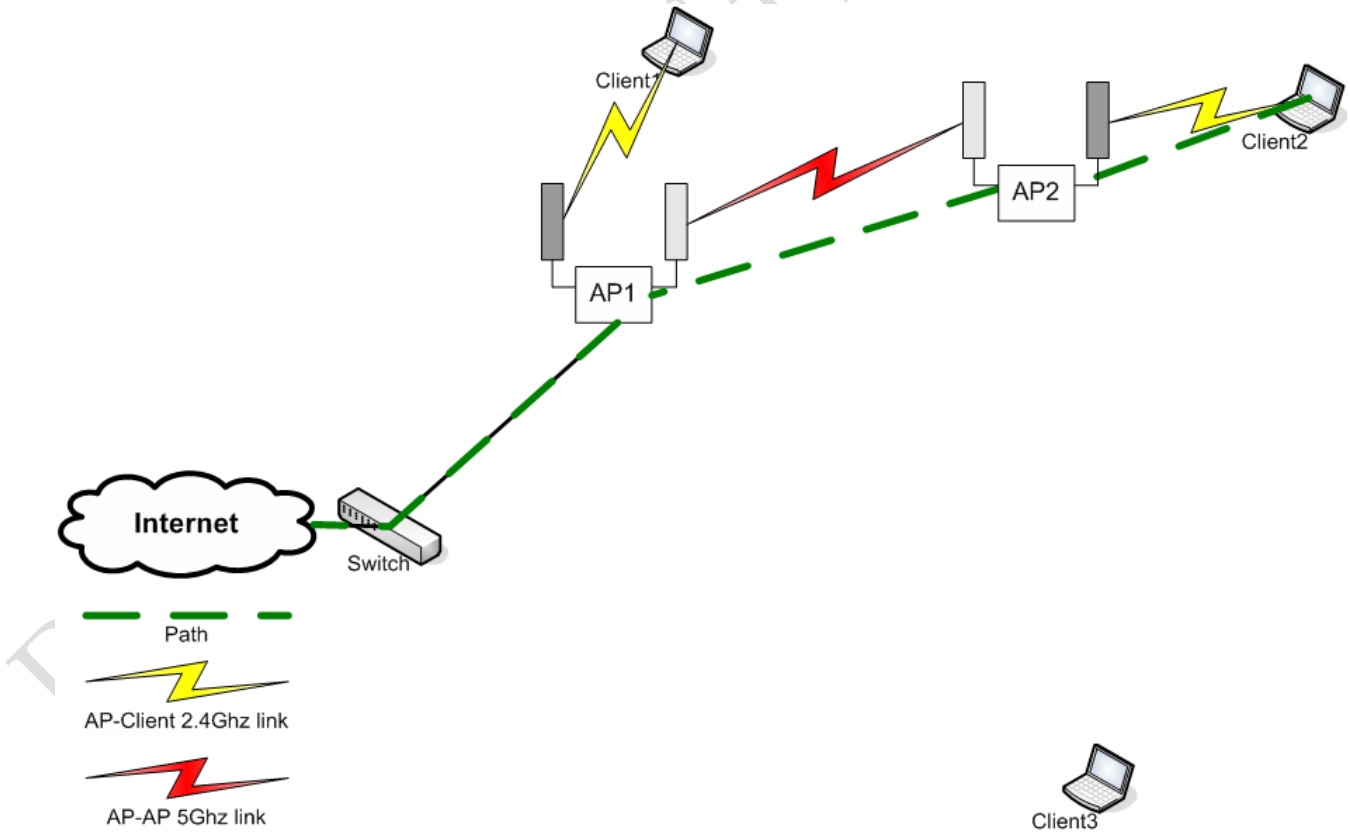


从这上面可以看到红色的线路代表 AP 之间的通信，采用的是 5G 频率，而 AP 到客户端则采用 2.4G 的频率，这样的方式增加了无线网络的转发效率。

以上是 MikroTik 如何使用点对点的无线网络拓扑。接下来我们逐步看看无线网状网络 Mesh 的构建过程：

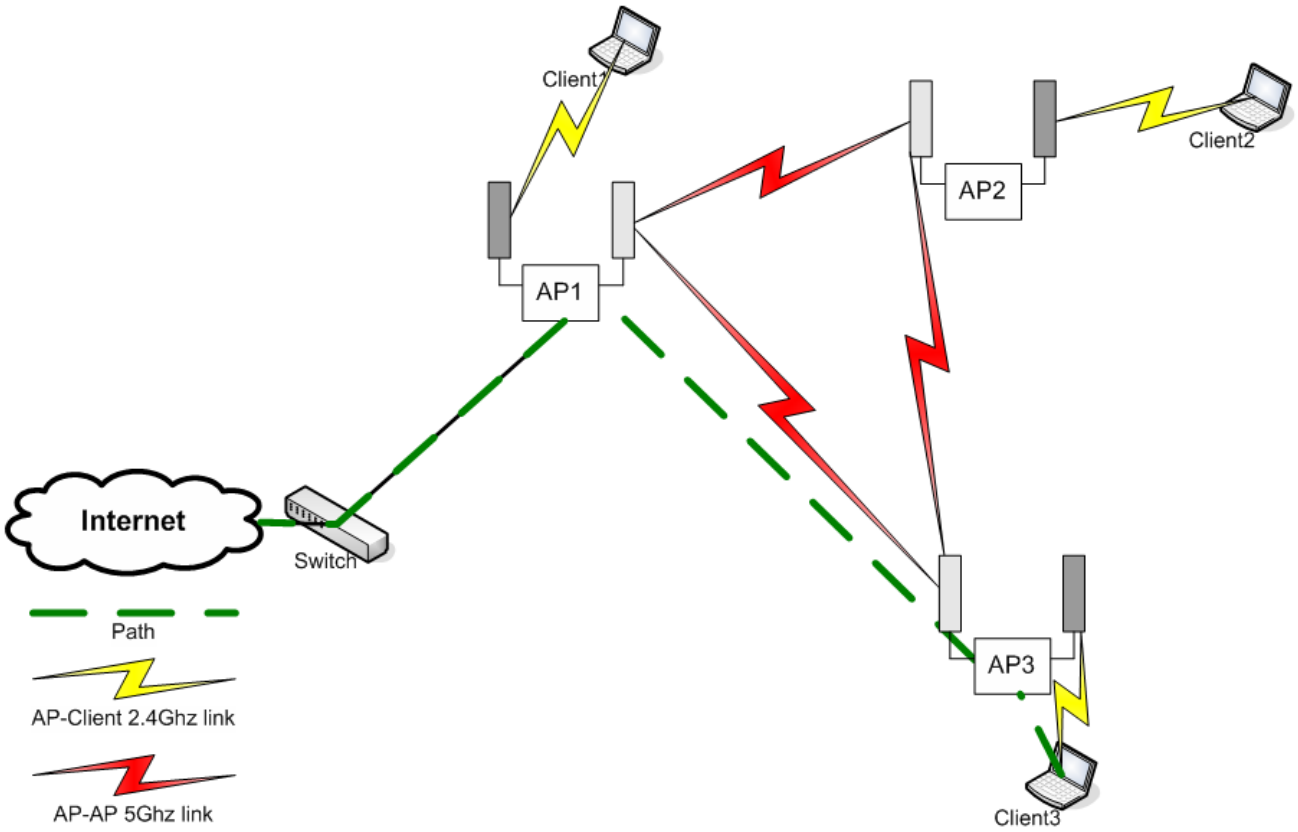


首先我们看看上面的图，Internet 通过交换机连接到 AP1，AP1 在做桥接后，将数据传输到 Client1，但在远程（在几公里外）的 Client2 需要接入 Internet，我们通过下图来看如何实现 Client2 连接如 Internet：



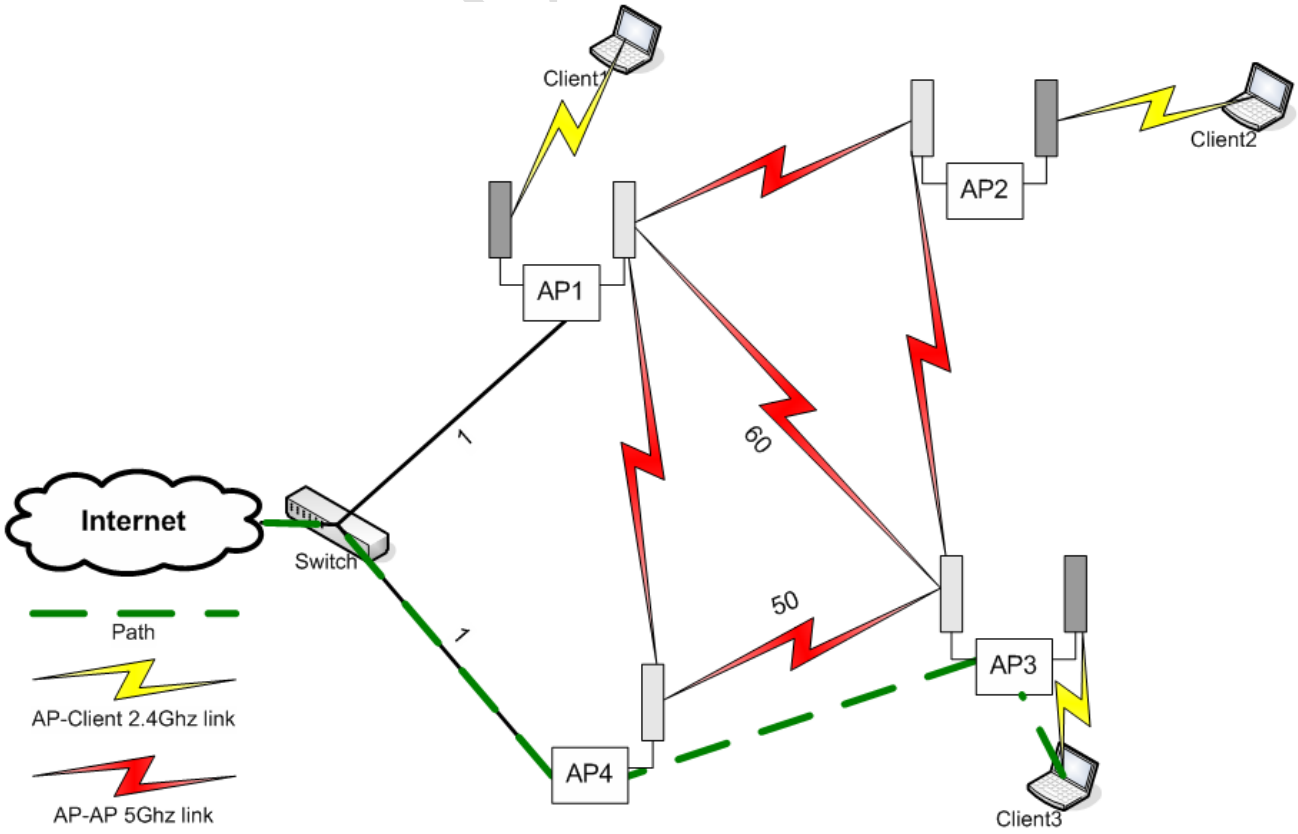
从该图中，我们通过以太网接入 Internet 并连接到 AP1，AP1 使用了两个无线发射天线，深灰色的天线使用 2.4G 面向 client1，而浅灰色的天线通过 5G 频率连接到远程的 AP2，由 AP2 将信号中继并发送给 Client2。

这时在我们远程网络中还有一个 Client3 需要连接，我们可以通过下图：



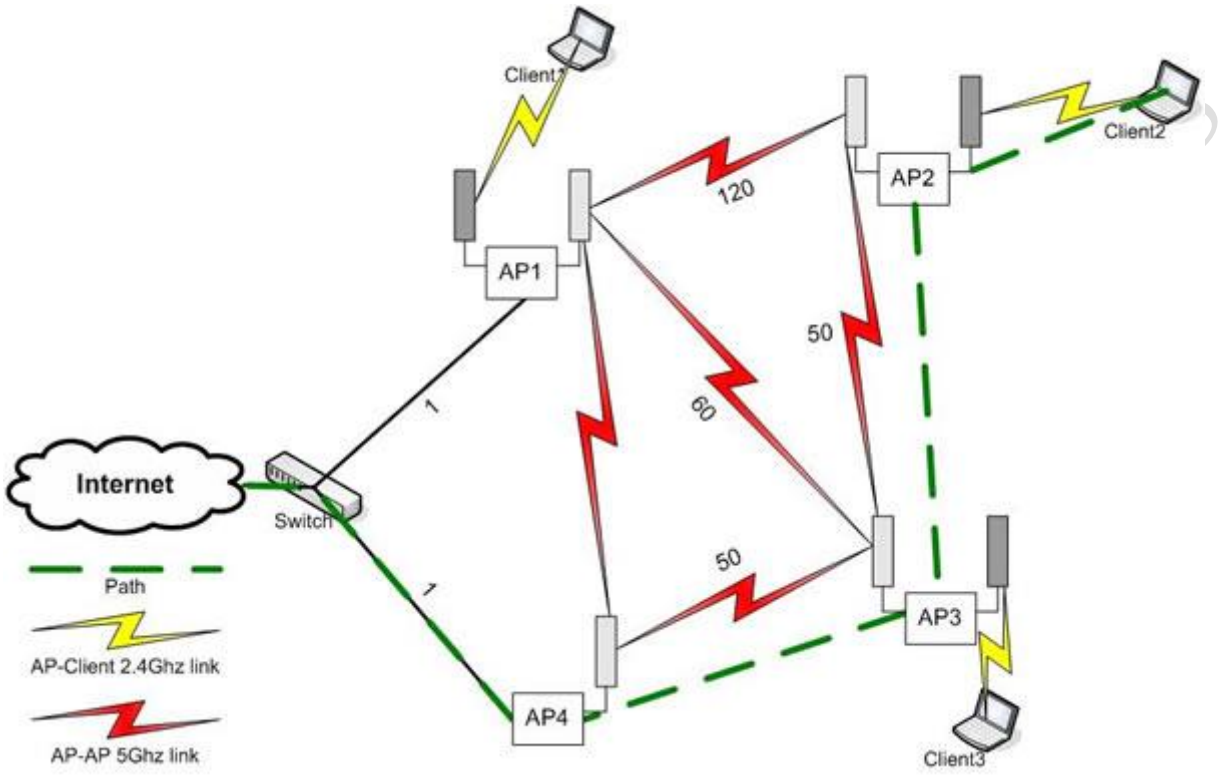
通过增加 AP3 做中继，并将 Internet 信息传送给 Client3，我们通这个拓扑图可以看出在网络的原有基础上只是增加了一个 AP3，而没有增加其他任何设备，而且可以从 AP1 和 AP2 中任意一个设备获取信号。

当我们增加一个设备 AP4 后，我们的网络拓扑将会有完全的改变，如下图：



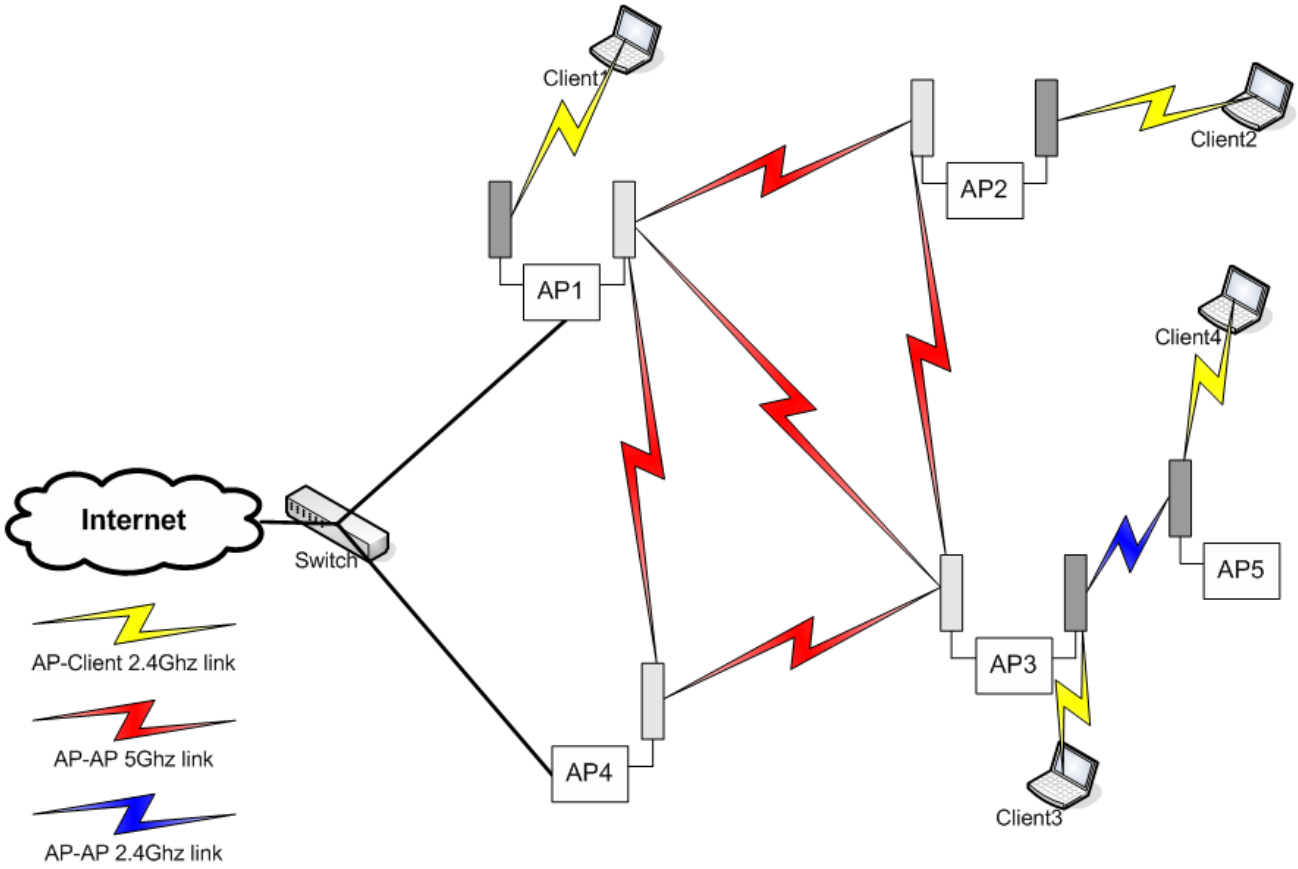
从这个图上我们可以看到，在增加了 AP4 后，他和 AP1 是两个同样接入 Internet 的设备，在拓扑图中的红色连接线代表每个设备之间的通信，也就是说 4 个设备相互之间都可以实现相互互访和线路的切换，当一个设备 AP1 中断后，Internet 资料可以从 AP4 传入整个网络，保证访问 Internet 网络的通畅。

由于在当前的网络中有 4 个设备可以实现相互的访问，访问效率的高低，网络等待时间的大小则成了上网速度的关键，在 MikroTik 的设备中，可以做到根据网络路径的耗用成本大小来计算出最佳的访问路线，如下图：



从这一张图看到，这里从交换机到 AP1 和 AP4 的时间延迟都为 1（因为使用以太网），Client2 需要访问 Internet，我从拓扑图上可以看到，他最近的访问路线是从 AP1 直接到 AP2，但这段线路可能存在故障需要花费 120 的时间延迟，由于 AP4-AP3-AP2 的线路花费是 50+50=100 的时间延迟，这样到 Client2 的线路则会优先选择最低延迟的那条。

在这个网络中 4 个设备构建了一个骨干路，在无线网状网络是可以灵活的拓展你的设备的如下图：



当我们又增加一个用户 Client4 时，我们可以通过在 AP3 下面增加一个 AP5 的中继设备连接到 Client4，这样通过无线设备的增加和变化使得整个网络构建变的灵活可靠。

只是从网络结构上来说，无线网状网的优点应该在于：

- 无论固定组网还是移动组网，都能够迅速按需形成任意拓扑；
- 拓扑遭遇节点高速、高频变换时，无线网状网能够自动调整拓扑并维持连接；
- 能够采用灵活的多跳传输，可随需扩展，非常适合有线不方便或成本很高的场合；

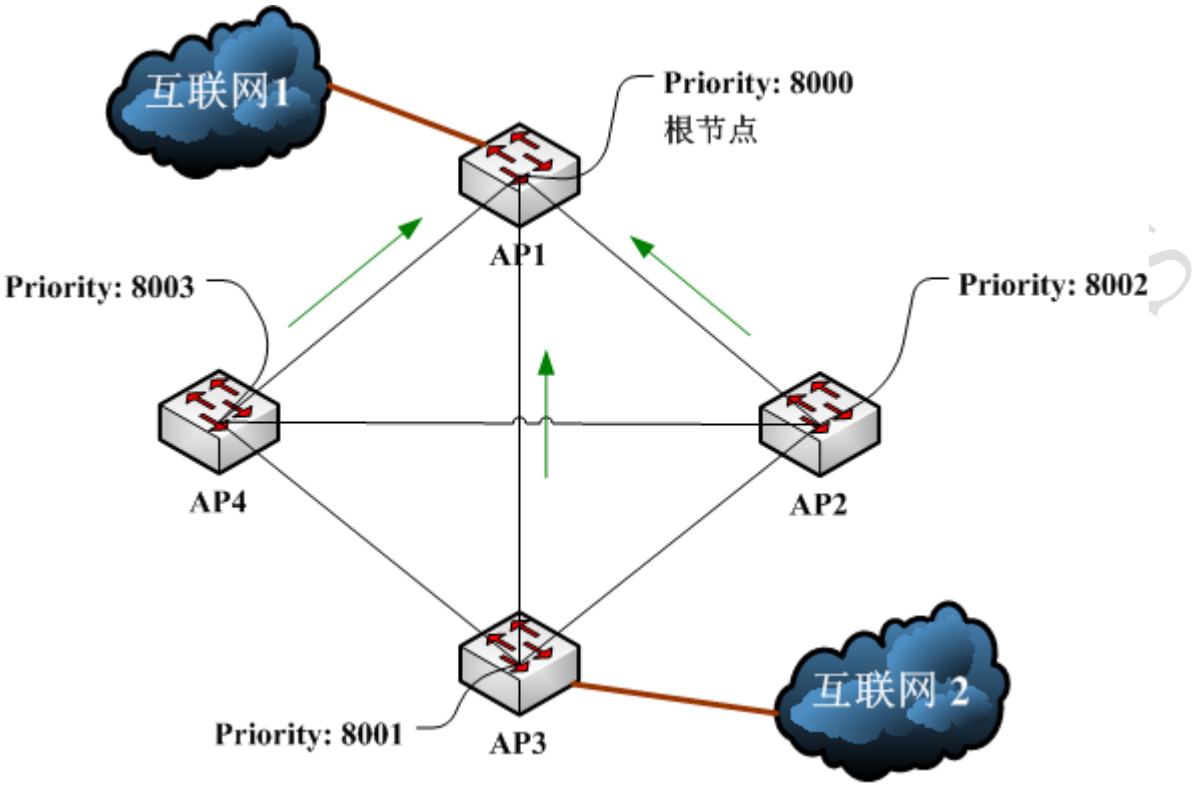
9.2 RSTP MESH 网络配置

RSTP MESH 原理

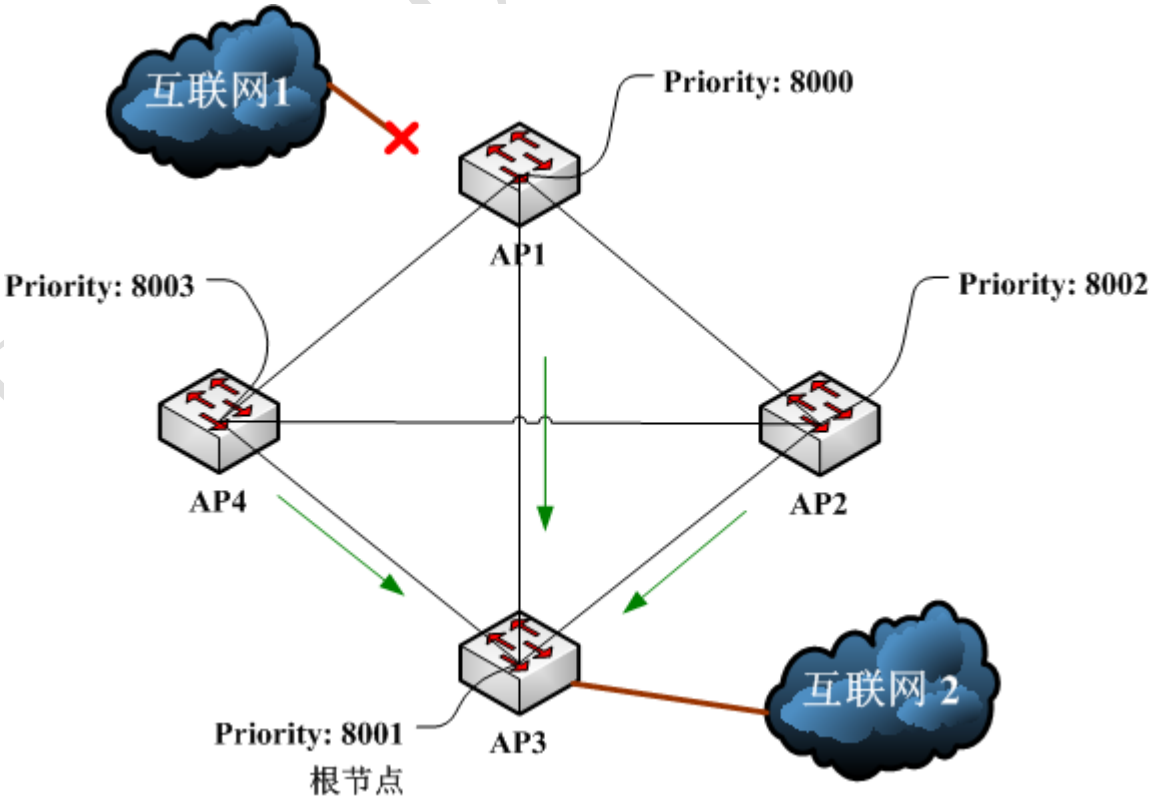
STP 在当拓扑发生变化，新的配置消息要经过一定的时延才能传播到整个网络，这个时延称为 Forward Delay，协议默认值是 15 秒。在所有网桥收到这个变化的消息之前，若旧拓扑结构中处于转发的端口还没有发现自己应该在新的拓扑中停止转发，则可能存在临时环路。为了解决临时环路的问题，生成树使用了一种定时器策略，即在端口从阻塞状态到转发状态中间加上一个只学习 MAC 地址但不参与转发的中间状态，两次状态切换的时间长度都是 Forward Delay，这样就可以保证在拓扑变化的时候不会产生临时环路。但是，这个看似良好的解决方案实际上带来的却是至少两倍 Forward Delay 的收敛时间！

通过最新的 RSTP 收敛速度会更快，MirkoTik 在 RouterOS 3.0 版本后增加了 rstp 协议，这样提高了无线网络的性能。

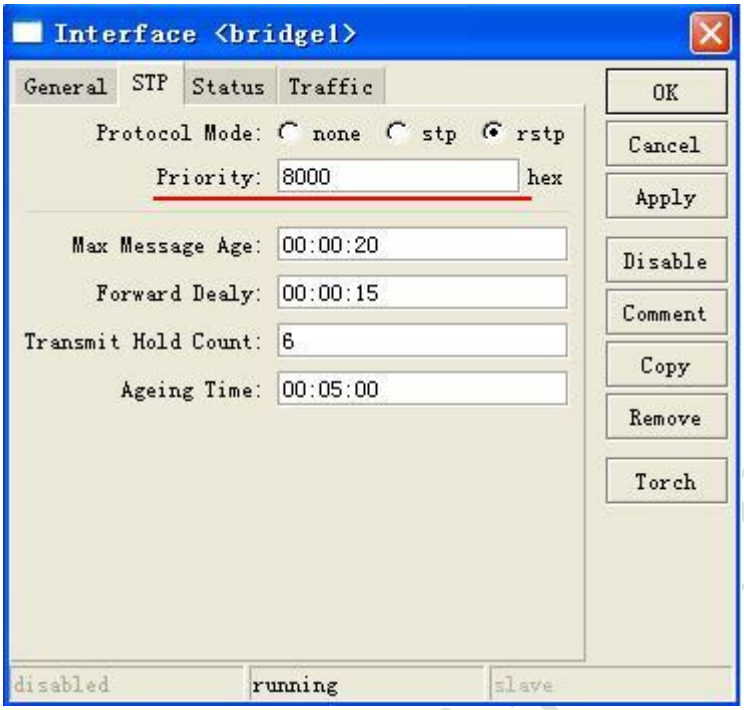
在下面的图中，可以看到，四个 AP 设备，组成了一个 WDS 的网状网络，在 AP1 和 AP3 分别接入的 Internet 网络，每个 AP 设备都会分配一个节点优先级 priority 参数（数字越小，优先级越高），Rstp 运行情况如下下面是一个正常的 rstp 网络拓扑，AP1 的优先级 priority 被设置为 8000，即为整个网络的根节点：



当 AP1 因故障断开后，根节点自动转换到优先级为 8001 的 AP3 上：



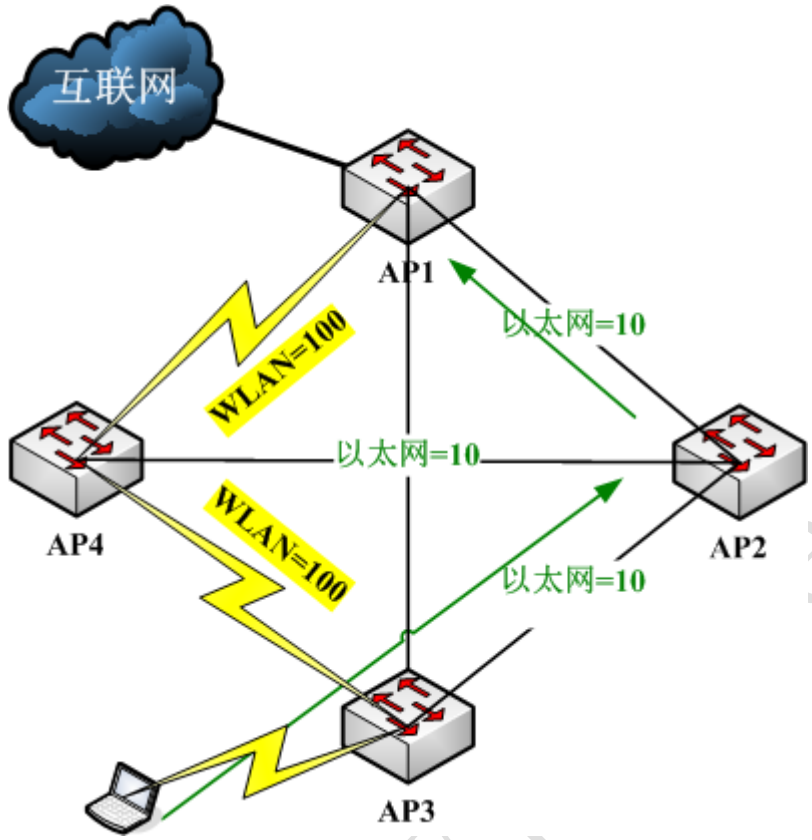
下面是在 RouterOS 上如何配置优先级 priority 参数，RouterOS 的 rstp 默认 priority 为 8000：



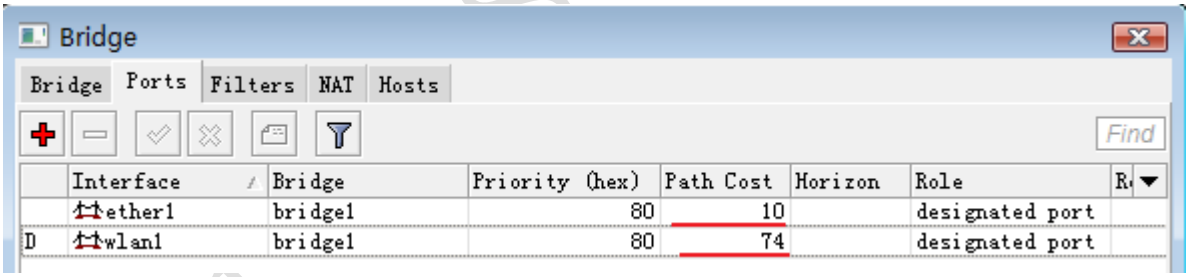
根据每个 AP 的情况，分配不同的优先级参数，这样一个完整的 WDS 网状网络便配置完成。

RSTP 的成本计算

WDS 会通过成本计算，并选择走最佳的路径，如下图我们可以看到，WLAN 默认的成本是 100，而以太网的成本为 10，根据最小值，AP 会选择最低成本的路径：

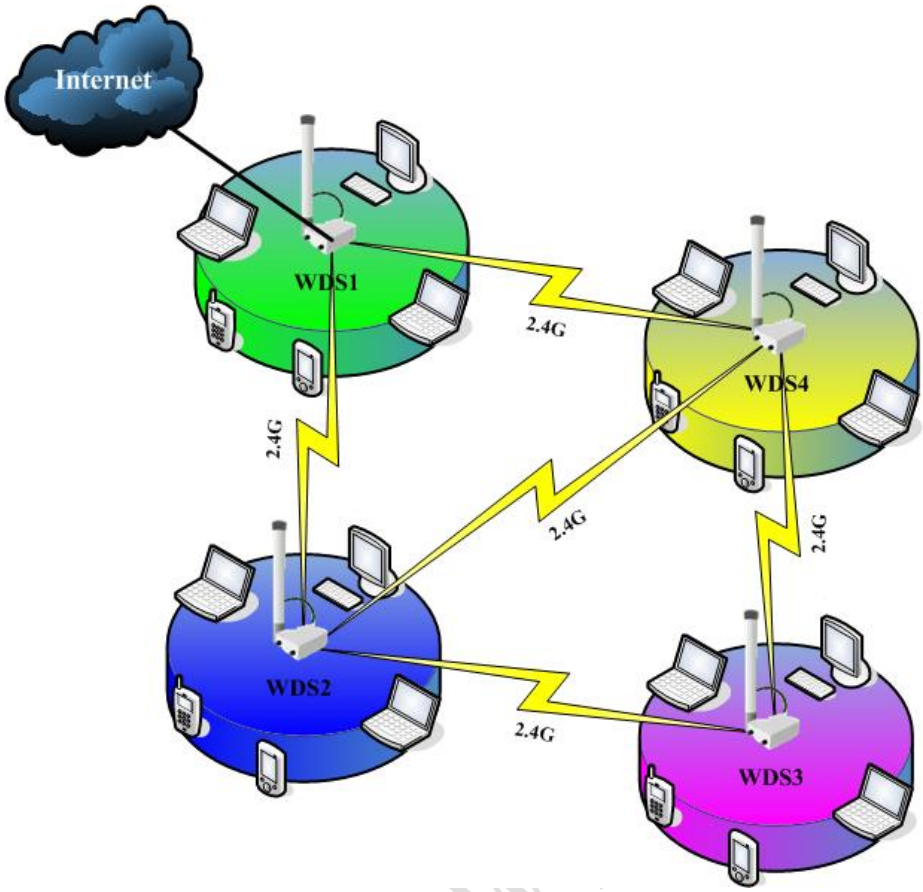


我们可以在 bridge 的 Ports 中看到 Path Cost 参数，ether1 成本为 10，wlan1 成本为 74，系统会自动判断成本最低的路径



9.3 WDS 漫游模式

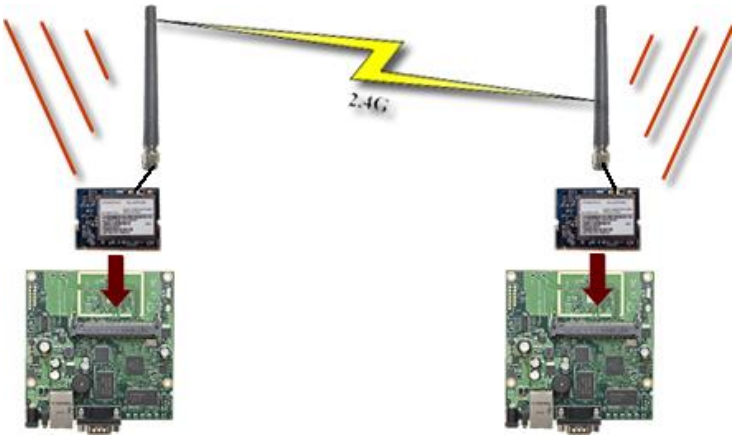
RSTP 的 MESH 最基本的一种方式为 WDS 漫游模式，即每个基站都配置 1 个无线模块，安装 1 个全向天线，对周边的区域进行覆盖，WDS 漫游都采用桥接方式，每个设备的 bridge 都需要启用 RSTP 协议，避免环路的出现，同时实现设备间的冗余。所以设备间的配置都需要将 Mode 设置为 ap-bridge，Band、频率、SSID 需要相同的设置就可以了，如下图：



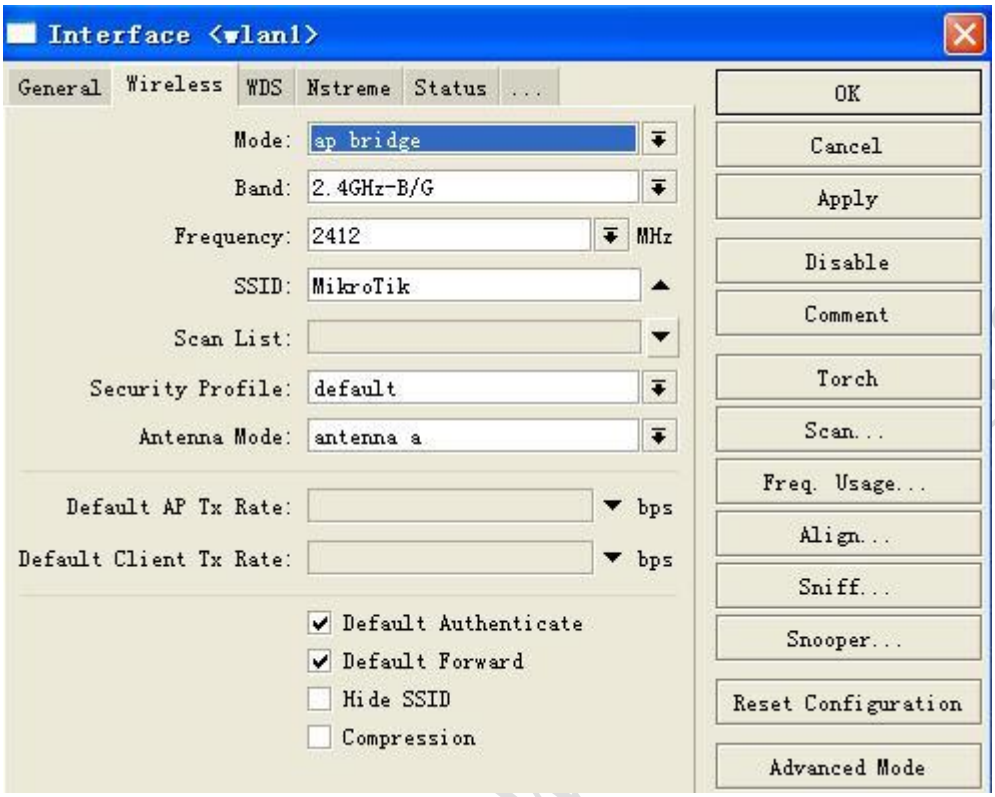
在 RouterOS3.0 中 WDS 选项增加了一组 mesh（无线网状网络）的设置，该参数能让 WDS 无线漫游更好的工作和选择最佳路径。推荐使用 dynamic mesh。

这样的模式仅能用于小范围的网络覆盖，适用于网络访问要求不高的区域，如大型的展会、办公楼层覆盖，以及较小环境的小区，由于这样的模式只有一个模块和全向天线组成，所以 1 个模块既要做无线覆盖，又要做各个基站直接的网络信号传输，降低了无线模块和设备的工作效率。如果要提高这样的网络质量，我们只能通过在每一个 WDS 设备上连接网线，然后骨干数据通过以太网交换，这样会带来一个问题就是网络布线成本的提高。

在 MikroTik Wireless 选项中我们通过配置多个 MikroTik 无线设备构建一个 WDS 漫游的无线网络。这里我们主要通过 802.11bg 的 2.4G 频段构建 Mesh 网络，每个 AP 安装一个无线模块，并采用 2.4G 的 bg 协议同时做 AP 间的连接传输又做网络的覆盖，如下图，我们通过 RB411A 安装 1 个无线模块，使用全向天线做设备间的传输，又作用户网络的覆盖：

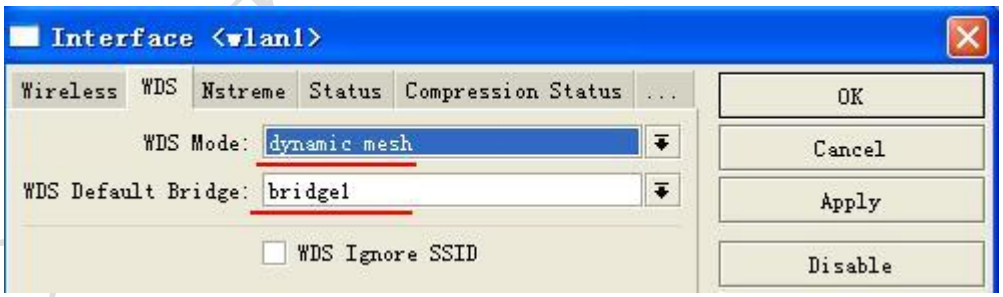


配置这样的网络时，我们需要将同一个区域内所有 MikroTik AP 设备 Mode 配置为 ap-bridge，然后设置相同频段，相同频率，配置为 WDS 模式，然后配置桥模式，如下图：



这里我们配置参数:： Mode:：ap-bridge

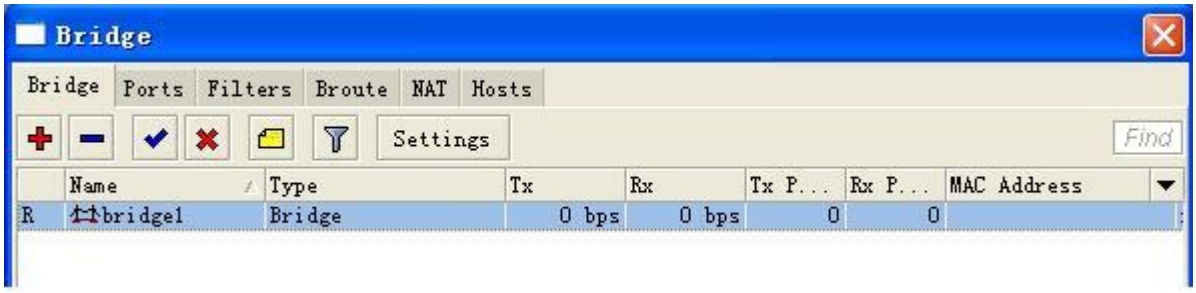
- Band：2.4GHz-B/G
- Frequency：2412
- SSID：MikroTik



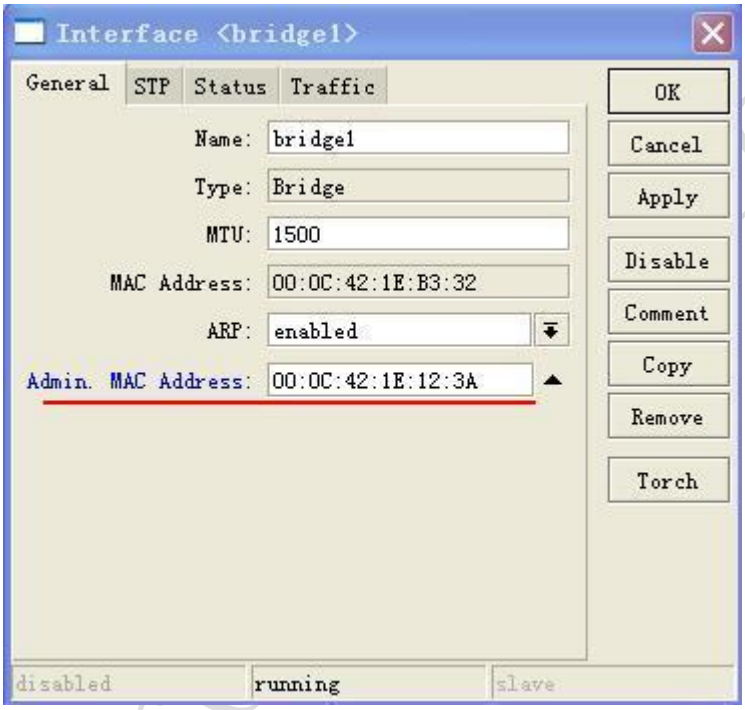
WDS 参数，我们配置 WDS Mode 为 dynamic mesh，并添加到默认 bridge 为 bridge1。采用 dynamic mesh 在 3.0 版本中要比在 dynamic 模式下运行的更稳定快速。

以上是无线网卡的配置参数，在同一区域内的 MikroTik AP 设备无线参数几乎都是相同的。但为了更好的让区域内的多个 AP 在转发数据时，达到最优路径我们需要通过 bridge 的 rstp 来完成。这里我们需要了解如何配置 bridge 中的 rstp：

首先我们在 Bridge 选项中添加一个 bridge1:



这里我们需要对 bridge1 的的 MAC 地址配置，因为在建立 bridge 后，桥接接口会自动产生 MAC 地址，可能在运行时候，MAC 地址会自动变换，为了保证网络质量的稳定我在这里配置一个静态 MAC 地址给 bridge1。

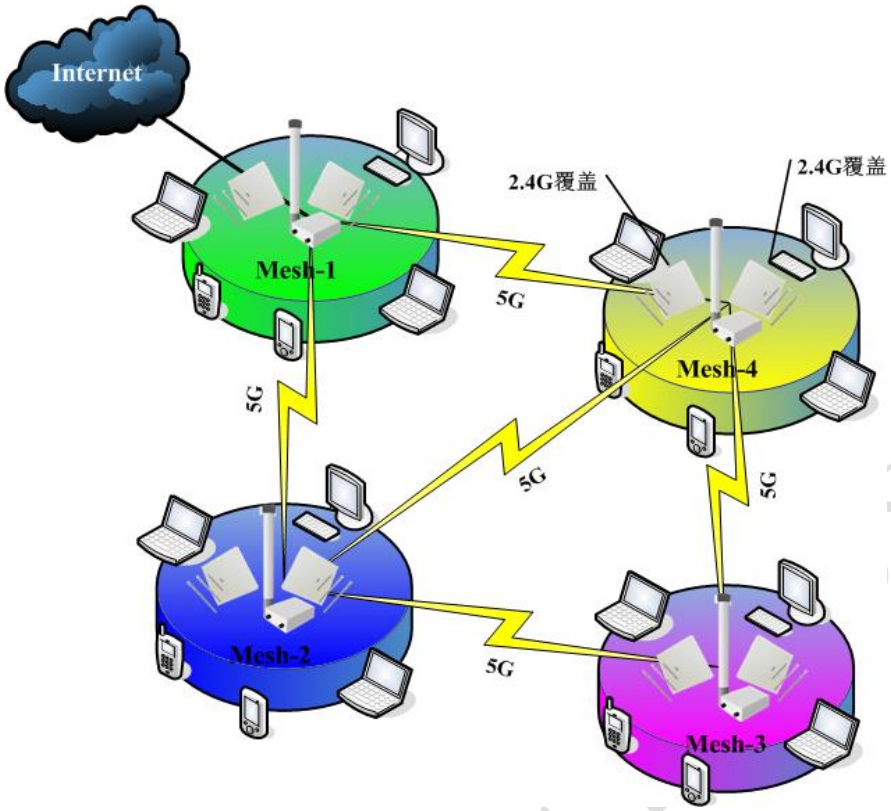


采用静态 MAC 地址有助于 RSTP Mesh 网络的稳定，我们可以自己设置一个 MAC 地址，或者使用设备上任何接口的 MAC 在 bridge 上。

注意：采用这样的 WDS 网络配置，一般使用于小型的无线漫游网络（如多层的办公室、大型的展厅和小范围的用户覆盖），AP 数量在 5-8 个左右的网络。采用这样的 WDS 模式受到连接用户数的限制，如果用户过多会造成大量的无线干扰，降低网络连接效率，这样的网络一个 AP 的用户连接数最好限制在 15-20 个以内。

9.4 多接口 Mesh 网络事例

多接口 Mesh 是解决前一例覆盖中只适用 1 张无线模块既做覆盖又做传输时带来的无线网卡效率过低的问题，在做城区或者小区覆盖时，由于全向天线效果不太理想，我们会考虑使用定向的扇区天线，但扇区天线覆盖范围有限，但我们可以通过增加无线网卡接口与天线弥补这个问题，RB433/AH、RB493/AH、RB600 等都支持扩展 3 个或者 4 个无线接口。多接口的 Mesh 以采用 3 个接口的无线模块为例，我们通过 1 个无线模块为骨干传输使用，保证骨干有足够的带宽，其余 2 个对周边区域进行覆盖，增加覆盖面积和增强信号。



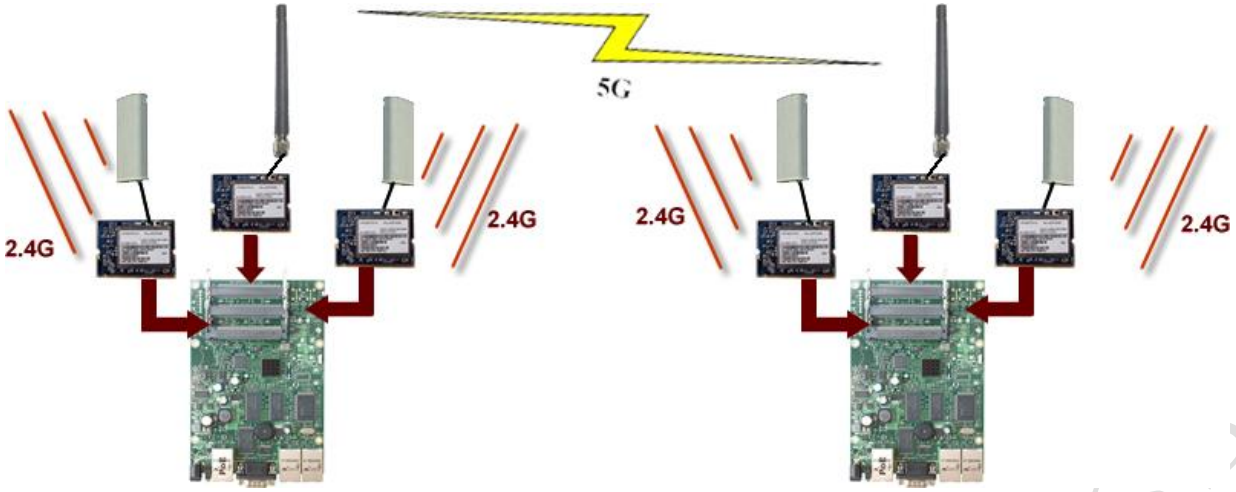
用多接口的无线 Mesh 覆盖，能弥补信号的盲点，增强信号。其好处在能减少设备的投入，普通的 AP 一般只能提供 1-2 个无线模块，但这样在覆盖中的效果相对较弱，如果需要弥补信号，则需要增加 AP 设备，但如果通过多接口的 RouterBOARD 产品，则只需要增加无线模块和天线，大大节约了成本，并增强的无线设备的可管理性。在配制中，我们同样采用 RSTP 与 WDS 结合的方式，即 AP1 的互联网出口为网络的根节点，即设置高的 RSTP 优先级，其余的 AP 则根据需要设置优先级别。、

多接口 Mesh 无线配制：

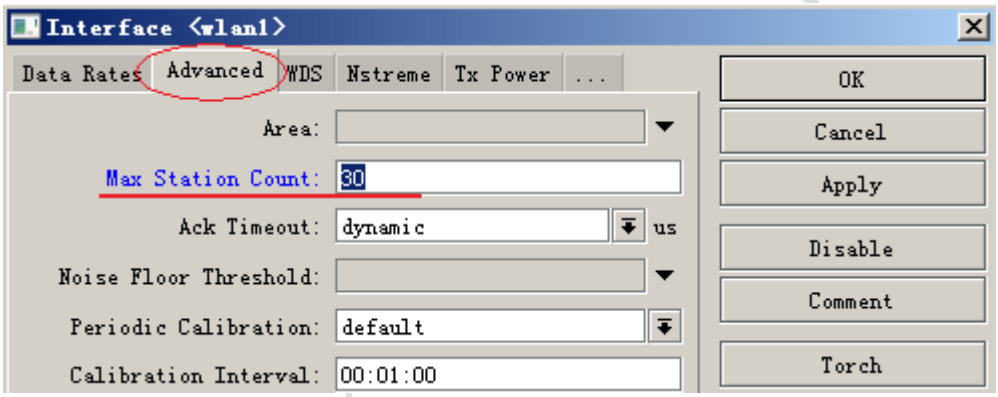
我们将所有的无线模块设置为 AP-bridge 模式，只是骨干和覆盖模块发射频段不同，一般骨干我们采用 5G 模式，2.4G 用于覆盖。

- 骨干与其他骨干间采用相同的发射频率和 SSID，模式为 ap-bridge，但骨干与覆盖的 SSID 不同。
- 覆盖之间的 SSID 相同，发射频率可以不同，以避免频道重复出现的干扰。
- 覆盖的 AP 需设定一个可连接的终端值，限制过多使用者拥挤到一个覆盖 AP 上，避免增加干扰和堵塞。
- 将三个无线接口定义到一个 bridge 中，并设置 RSTP 的优先级和相应的参数。

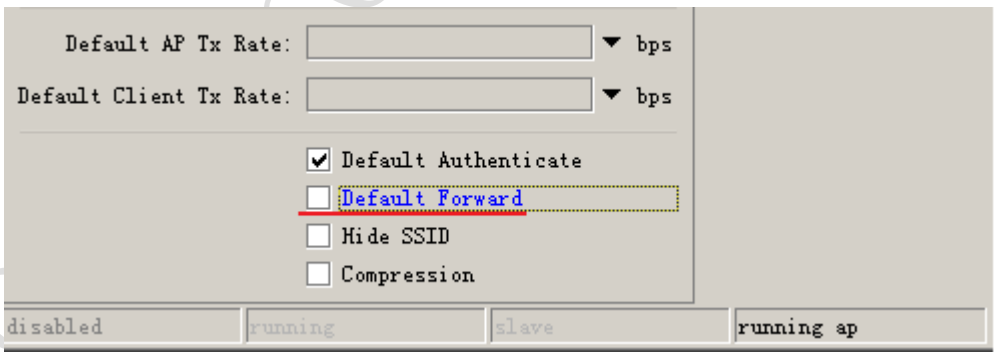
我们通过在一个 RB433 上增加 3 个无线模块，一个用于骨干连接，另外两个则用于网络覆盖：



我们在 wireless 的覆盖无线网卡上配制终端连接数，一般配置 30 个终端比较合适：



在无线的 wireless 参数种，我们关闭掉 Default Forward 参数，禁止无线终端之间相互通信：



以上两种配置，能提高无线覆盖的效率和稳定性。

9.5 Station 模式下通过脚本切换 AP 基站

关于 RouterOS 的无线 RB 设备作为 station（即终端设备）时，在多个相同 SSID 的 AP 基站间如何漫游，很多人完全归结于 AP 设置，其实 AP 设置占了一部分，作为终端设备如何选取 AP 基站是很关键的，在 AP 基站间做切换，什么时候切换都不是 AP 决定的，而是终端设备自己选择。切换 AP 信号最通俗的理解就是信号强度，当信号强度无法满足需要的时候，我们就需要让终端设备切换到其他信号较强的 AP 上。

根据信号强度我们可以通过判断当前连接 AP 基站的信号值，如果低于多少，我们作出要求 station 终端设备切换的操作。无线设备的网卡初始化连接时，周围存在多个相同 SSID 的 AP 基站，始终会选择信号强度最好的 AP 基站，因此根据这个特性，可以理解为将 RB 无线网卡初始化一次（不是复位），也就将当前信号连接从 registration-table 中删除，重新选择新的信号。

当然删除信号操作，在 RouterOS 内部肯定不是手动操作的，必须通过周期性的执行脚本完成，脚本读取当前信号强度，判断是否超过阀门值，超过后，就执行禁用和启用网卡操作，实现切换的目的。这样的漫游肯定会造成网络等待时间，因为承载网络的物理信号都会中断，肯定会掉包，只是看初始化和建立连接的时间会有多长，根据之前的经验一般会掉 2-3 个包左右。

下面脚本仅供参考：

```
:loacl sig
:set sig [/interface wireless registration-table get [/interface wireless
registration-table find interface="wlan1"] signal-strength ]
:set sig [:pick $sig 0 [:find $sig "d" ]]
:if ($sig < -80) do={
  /interface wireless registration-table remove [find interface=wlan1]
}
```

脚本运行，需要在 RouterOS 计划任务在完成，设置周期运行时间，最小单位为“秒”，配置进入/system scheduler（配置操作可以参考《RouterOS 入门到精通》第 39 章节）

9.6 基于 Connect-list 的 Station 无线漫游

无线设备的网卡在初始化连接时，周围存在多个相同 SSID 的 AP 基站，始终会选择信号强度最好的 AP 基站，这里我考虑使用 connect-list 去完成这个操作，通过 connect-list 完成对 AP 基站的匹配（事先要输入 AP 基站的 MAC 地址）信号强度，当连接 AP 基站的信号强度低于一定设定值后，会断开无线连接，这样 station 会尝试连接到 connect-list 中其他 AP 基站，且信号范围在给定的值内。

- 测试的设备：RouterBOARD 设备 RB951Ui-2HnD
- RouterOS 版本：v6.38.1
- 基本配置情况：2 台做 ap-bridge，1 台做 station-wds，所有设备建立 rstp 的 WDS 漫游网络，2 台设备 ap-bridge 无线参数配置相同，启用 wds-dynamic，添加到 bridge1
- 测试环境：室内

根据以上的配置，下面通过配置脚本的形式给出

AP1 配置参数

```
/interface bridge
add name=bridge1 priority=0x8001 protocol-mode=rstp
/interface bridge port
add bridge=bridge1 interface=ether1

/ip address
add address=192.168.11.50/24 interface=bridge1 network=192.168.11.0
```

```
/ip route
add distance=1 gateway=192.168.11.1

/interface wireless
set [ find default-name=wlan1 ] area=mik band=2ghz-b/g/n channel-width=20/40mh
   disabled=no mode=ap-bridge radio-name=AP1 ssid=mik1 wds-default-bridg
bridge1 wds-mode=dynamic-mesh wireless-protocol=802.11
```

AP2 配置参数

```
/interface bridge
add name=bridge1 priority=0x8002 protocol-mode=rstp
/interface bridge port
add bridge=bridge1 interface=ether1

/ip address
add address=192.168.11.51/24 interface=bridge1 network=192.168.11.0

/ip route
add distance=1 gateway=192.168.11.1

/interface wireless
set [ find default-name=wlan1 ] area=mik band=2ghz-b/g/n channel-width=20/40mh
   disabled=no mode=ap-bridge radio-name=AP2 ssid=mik1 wds-default-bridg
bridge1 wds-mode=dynamic-mesh wireless-protocol=802.11
```

station 配置参数

```
/interface bridge
add name=bridge1 priority=0x8010 protocol-mode=rstp
/interface bridge port
add bridge=bridge1 interface=ether1
/ip address
add address=192.168.11.55/24 interface=bridge1 network=192.168.11.0

/ip route
add distance=1 gateway=192.168.11.1

/interface wireless
set [ find default-name=wlan1 ] area=mik band=2ghz-b/g/n disabled=no \
   mode=station-wds radio-name=Client ssid=mik1 wds-default-bridge=bridge1
\
   wds-mode=dynamic-mesh wireless-protocol=802.11
/interface wireless connect-list
add interface=wlan1 mac-address=E4:8D:8C:60:B6:CD security-profile=default
\
```

```
    signal-range=-50..1
add interface=wlan1 mac-address=E4:8D:8C:BD:14:D1 security-profile=default
\
    signal-range=-50..1
```

测试结果：

在不使用脚本判断网卡信号强度低于多少的情况下，通过 connect-list 判断信号来切换 AP 是可行的，切换时会丢 2-3 个包。注意由于是室内环境测试，signal-rang 设置为-50 到 1dBm 较高范围，因此如果是实际的应用场景，需要更加实地信号勘察后作出 signal-rang 的配置。

以上测试是基于 802.11 协议，当改为 nv2 协议后，connect-list 切换会失效，修改 connect-list 参数时，wireless 应用导致 CPU 100%（问题已经回馈给 mikrotik，并得到回复在后续版本会修正），如果禁用启用网卡方式切换，会丢 38 个包

9.7 HWMP+ Mesh 无线网状网络

HWMP+是 MikroTik 为无线网状网络 Mesh 定义的 2 层路由协议。基于 IEEE802.11s 草案 Hybrid Wireless Mesh Protocol (HWMP)，能用于替代 STP 生成树协议确保环路的最优路径。 HWMP+ 协议并不能兼容 HWMP 的 IEEE 802.11s 草案。

注：这种分布式系统不仅能应用到无线分布系统（WDS）。HWMP+网状网络同样也支持以太网接口的网状网络，因此你可以用于简单的以太网分布系统，或者同时连接 WDS 和以太网。

RouterOS Mesh 选项基于 HWMP，Mesh 与 RSTP 的无线组网区别在于，HWMP 是基于跳跃级数选择路径，而 RSTP 则是根据路径成本开销选择路径

interface mesh 属性

PREQ 路由请求

PREP 路由应答

admin-mac (MAC 地址，默认: 00:00:00:00:00:00) – 管理分配的 MAC 地址，当 auto-mac 设置为 disable 后起作用。
arp (disabled | enabled | proxy-arp | reply-only；默认: enabled) – 地址解析协议设置
auto-mac (boolean，默认: no) –如设置为禁用，这时 admin-mac 将会被要求设置 mesh interface 上，否则会使用一些端口的地址。
hwmp-default-hoplimit (integer: 1..255) –路由协议包产生最大的跳跃总数，在一个 HWMP+数据报被发送后，达到最大跳跃限制数，将会被自动丢弃。
hwmp-prep-lifetime (time，默认: 5m) –为创建线路从收到 PREP 或 PREQ 信息的生存时间
hwmp-preq-destination-only (布尔值，默认: yes) -- 是否只有目的地可以响应 hwmp + preq 讯息
hwmp-preq-reply-and-forward (布尔值，默认: yes) --是否中间节点应该发出 hwmp + preq 消息后，响应它。应用于仅当 hwmp-preq-destination-only 被禁用
hwmp-preq-retries (整型，默认: 2) – 当地址无法到达情况下，多长时间重试探测指定 MAC 地址的路径。
hwmp-preq-waiting-time (时间，默认: 4s) – 多长时间等待一个响应第一个 PREQ 信息。

hwmp-rann-interval (时间，默认: 10s) – 间隔多长发送 HWMP+ RANN 信息
hwmp-rann-lifetime (时间，默认: 1s) – 为创建路径从收到 RANN 信息的生存时间
hwmp-rann-propagation-delay (数字，默认: 50) – 多久前等待发送 RANN 信息。值为百分之一秒计算 (100cs = 1sec)
mesh-portal (布尔值，默认: no) – 是否设定这个接口为一个 Mesh 网络的入口。
mtu (数字，默认: 1500) – 最大传输单元
name (字符) – 接口名称
reoptimize-paths (布尔值，默认: no) – 是否定期向外发送 PREQ 信息询问网络中的 MAC 地址，如果网络拓扑经常变动基于 Turing 设置是非常有用的。注意：如果没有接收到一重新优化 PREQ 信息，将保持现有的路径（直到探测超时）

/interface mesh port 属性。

hello-interval (时间，默认: 10s) – 发送 HWMP+ Hello 信息最大时间间隔。只能用于以太网卡类型的端口。
interface (interface name) – 接口名称，那一个接口包括在 Mesh 中
mesh (interface name) – 属于那一个 Mesh 界面
path-cost (整型: 0..65535; 默认: 10) – 接口的路径成本，通过最佳路径决定使用的路由协议。
port-type (WDS | auto | ethernet | wireless) – 使用的端口类型

- **auto** – 根据接口类型自动决定端口使用的类型
- **WDS** – 一个无线分布式系统接口，一种点对点无线连接。远程 MAC 地址通过无线连接数据得知。
- **ethernet** – 远程 MAC 地址通过每次 HWMP+的 Hello 信息、接收到的源 MAC 地址或者转发的传输数据中学习到
- **wireless** – 远程 MAC 地址获取通过无线连接数据

port-type-used (只读, wireless | WDS | ethernet-mesh | ethernet-bridge | ethernet-mixed) – 端口类型和确认的状态

/interface mesh fdb **属性：** 只读状态下的 Mesh 接口转发数据库(FDB).

mac-address (MAC 地址) – MAC 地址相对应的 FDB 记录项目
seqnum (整型) – 序列编号使用到路由协议中避免环路
type (local | outsider | direct | mesh | neighbor | larval | unknown) – FDB 记录项的类型

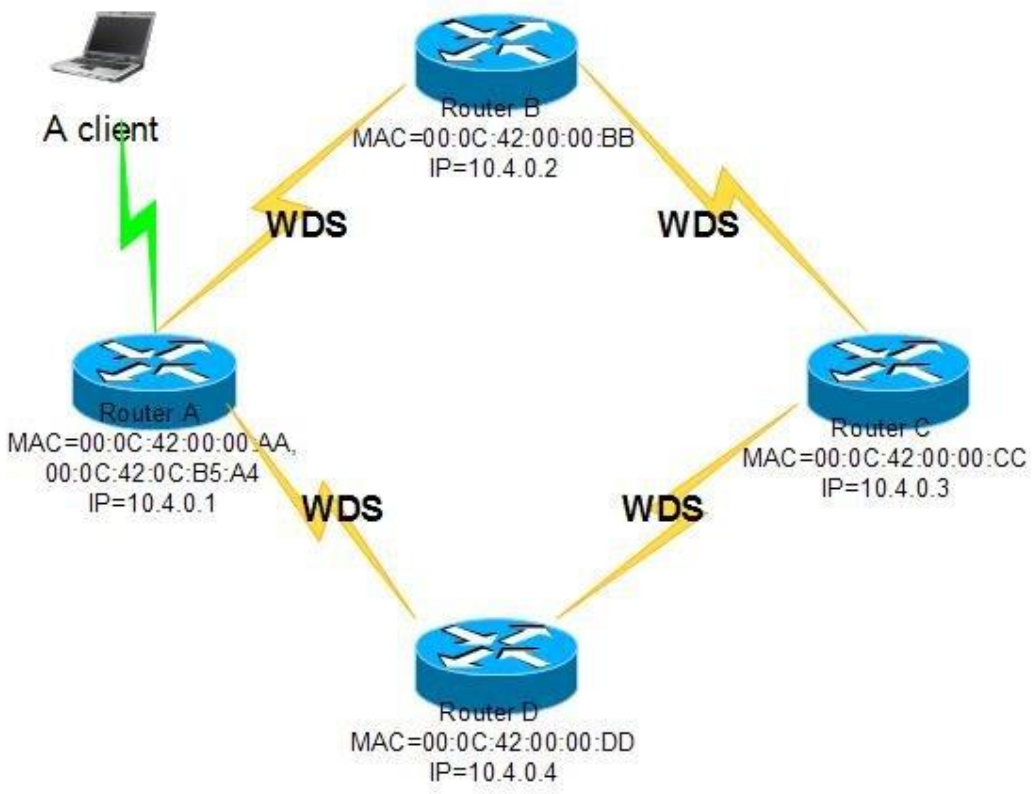
- **local** – MAC 地址属于本地路由器
- **outsider** – MAC 地址属于 Mesh 网络外部的设备
- **direct** – MAC 地址属于在一个 Mesh 网络中一个接口上的无线客户端
- **mesh** – MAC 地址属于一个设备到达了 Mesh 网络，可以是内部或者外部网络
- **neighbor** – MAC 地址属于一个 Mesh 路由器，是一个直接的邻居路由器
- **larval** – MAC 地址属于一个未知到达 Mesh 网络
- **unknown** – MAC 地址属于一个未知的设备

mesh (interface name) – 属于这个 Mesh 界面的 FDB 项
on-interface (interface name) – Mesh 端口用于传输转发，一种 next-hop 值
lifetime (time) – 如果记录项没有使用传输转发，则会定义生存时间。
age (time) – FDB 项的时期
metric (integer) – metric 值是设置通过路由协议决定最佳路径

使用 **wds-default-cost** 和 **wds-cost-range** 无线接口参数会被路由协议使用， 该 WDS 的成本将被用来作为 **path-cost** 端口动态添加到网格接口。

应用实例

使用 wds-default-cost 和 wds-cost-range 无线接口参数会被路由协议使用， 该 WDS 的成本将被用来作为 path-cost 端口动态添加到网格接口。



这个事例使用静态 WDS 连接，当无线连接被启动，会自动添加到 Mesh 端口中。两个不同的发射频率会被使用：一个为 AP 间的通信链路，一个为客户端的 AP 覆盖。因此一个设备至少需要两个无线网卡接口。

下面的配置适用于所有的 AP：

```
/interface mesh add disabled=no
/interface mesh port add interface=wlan1 mesh=mesh1
/interface mesh port add interface=wlan2 mesh=mesh1
```

用于 AP 间互联的 interface

```
/interface wireless set wlan1 disabled=no ssid=mesh frequency=2437
band=2.4ghz-b/g mode=ap-bridge wds-mode=static-mesh
wds-default-bridge=mesh1
```

用于客户端连接的 interface

```
/interface wireless set wlan2 disabled=no ssid=mesh-clients frequency=5180
band=5ghz mode=ap-bridge
```

为每一个 AP 配置一个静态的 WDS 接口连接

```
/interface wireless wds add disabled=no master-interface=wlan1
name=<descriptive name of remote end> wds-address=<MAC address of remote end>
```

注意：这里的 WDS 接口设置需要手动，因为我们采用的是静态 WDS 模式 ，如果你使用 wds-mode=dynamic-mesh，所有的 WDS 接口将会自动创建。

在真实环境中最好需要注意无线连接的安全问题。可以使用/interface wireless security-profile.

在路由器 A 上的结果（现在有一个客户端连接到 Wlan2 ）：

```
[admin@A] > /interface mesh pr
Flags: X - disabled, R - running
 0  R name="mesh1" mtu=1500 arp=enabled mac-address=00:0C:42:0C:B5:A4
     auto-mac=yes
     admin-mac=00:00:00:00:00:00 mesh-portal=no hwmp-default-hoplimit=32
     hwmp-preq-waiting-time=4s hwmp-preq-retries=2
     hwmp-preq-destination-only=yes
     hwmp-preq-reply-and-forward=yes hwmp-prep-lifetime=5m
     hwmp-rann-interval=10s
     hwmp-rann-propagation-delay=1s hwmp-rann-lifetime=22s

[admin@A] > interface mesh port p detail
Flags: X - disabled, I - inactive, D - dynamic
 0    interface=wlan1 mesh=mesh1 path-cost=10 hello-interval=10s
     port-type=auto port-type-used=wireless
 1    interface=wlan2 mesh=mesh1 path-cost=10 hello-interval=10s
      port-type=auto port-type-used=wireless
 2  D interface=router_B mesh=mesh1 path-cost=105 hello-interval=10s
      port-type=auto port-type-used=WDS
 3  D interface=router_D mesh=mesh1 path-cost=76 hello-interval=10s
      port-type=auto port-type-used=WDS
```

FDB (转发数据库 Forwarding Database)在当前状态下包含的本地 MAC 地址信息，Mesh 节点能能到达的本地界面和探测到的 Mesh 邻居：

```
[admin@A] /interface mesh> fdb print
Flags: A - active, R - root
   MESH       TYPE     MAC-ADDRESS       ON-INTERFACE       LIFETIME     AGE
A  mesh1      local    00:0C:42:00:00:AA                              3m17s
A  mesh1      neighbor 00:0C:42:00:00:BB router_B                       1m2s
A  mesh1      neighbor 00:0C:42:00:00:DD router_D                      3m16s
```

```
A  mesh1      direct   00:0C:42:0C:7A:2B wlan2                          2m56s
A  mesh1      local    00:0C:42:0C:B5:A4                              2m56s

[admin@A] /interface mesh> fdb print detail
Flags: A - active, R - root
 A  mac-address=00:0C:42:00:00:AA type=local age=3m21s mesh=mesh1 metric=0
    seqnum=4294967196
 A  mac-address=00:0C:42:00:00:BB type=neighbor on-interface=router_B
      age=1m6s
   mesh=mesh1 metric=132 seqnum=4294967196
 A  mac-address=00:0C:42:00:00:DD type=neighbor on-interface=router_D
      age=3m20s
    mesh=mesh1 metric=79 seqnum=4294967196
 A  mac-address=00:0C:42:0C:7A:2B type=direct on-interface=wlan2 age=3m
      mesh=mesh1
    metric=10 seqnum=0
 A  mac-address=00:0C:42:0C:B5:A4 type=local age=3m mesh=mesh1 metric=0
      seqnum=0
```

测试 ping :

```
[admin@A] > /ping 00:0C:42:00:00:CC
00:0C:42:00:00:CC 64 byte ping time=108 ms
00:0C:42:00:00:CC 64 byte ping time=51 ms
00:0C:42:00:00:CC 64 byte ping time=39 ms
00:0C:42:00:00:CC 64 byte ping time=43 ms
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 39/60.2/108 ms
```

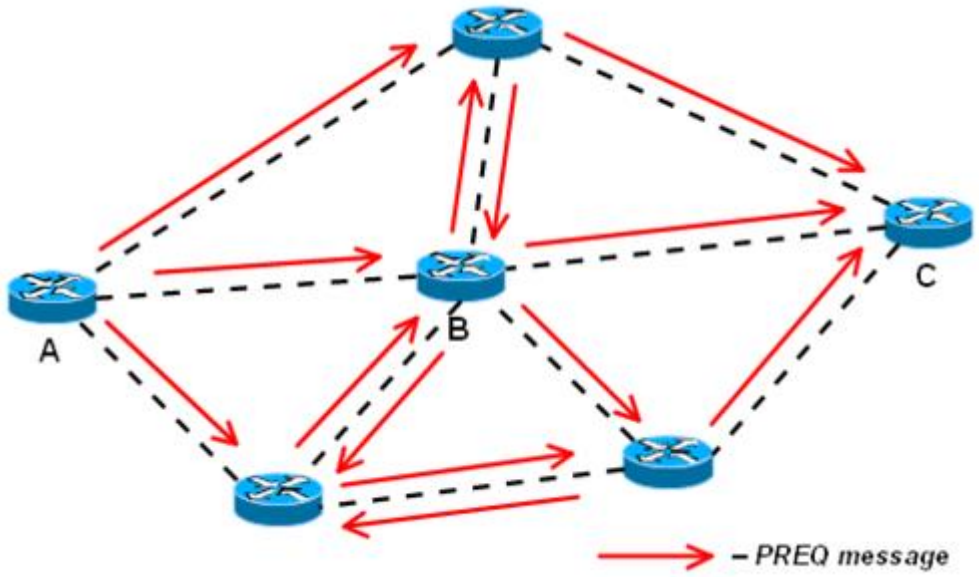
Router A 必须探测到 Router C 的路径，因此第一个 ping 包延迟稍微大一点。

同样我们也可以通过 IP 层的 ping 检测网络 A：

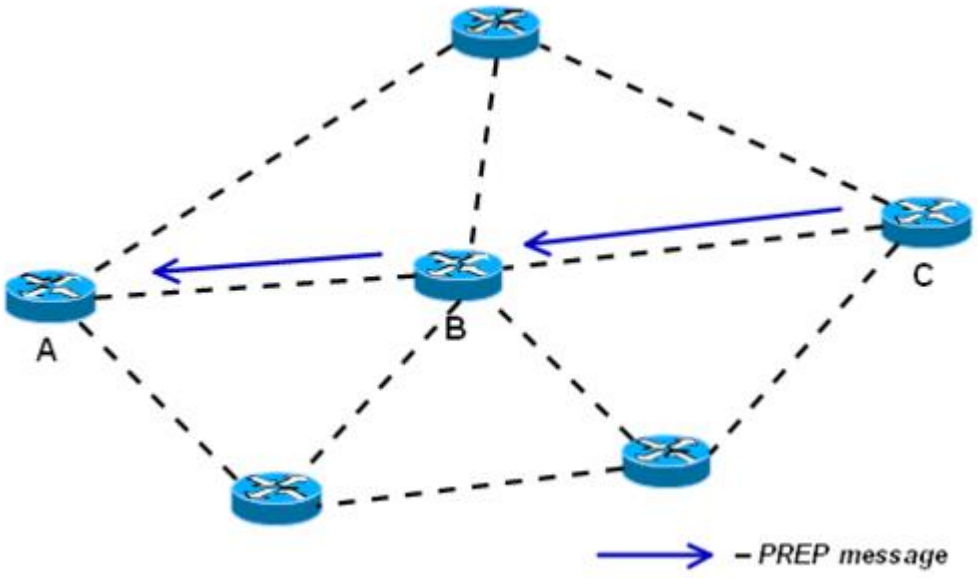
```
[admin@A] > /ping 10.4.0.3
10.4.0.3 64 byte ping: ttl=64 time=163 ms
10.4.0.3 64 byte ping: ttl=64 time=46 ms
10.4.0.3 64 byte ping: ttl=64 time=48 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 46/85.6/163 ms
```

HWMP 协定特性

1、反应方式



Router A 想要探测到 C 的路径



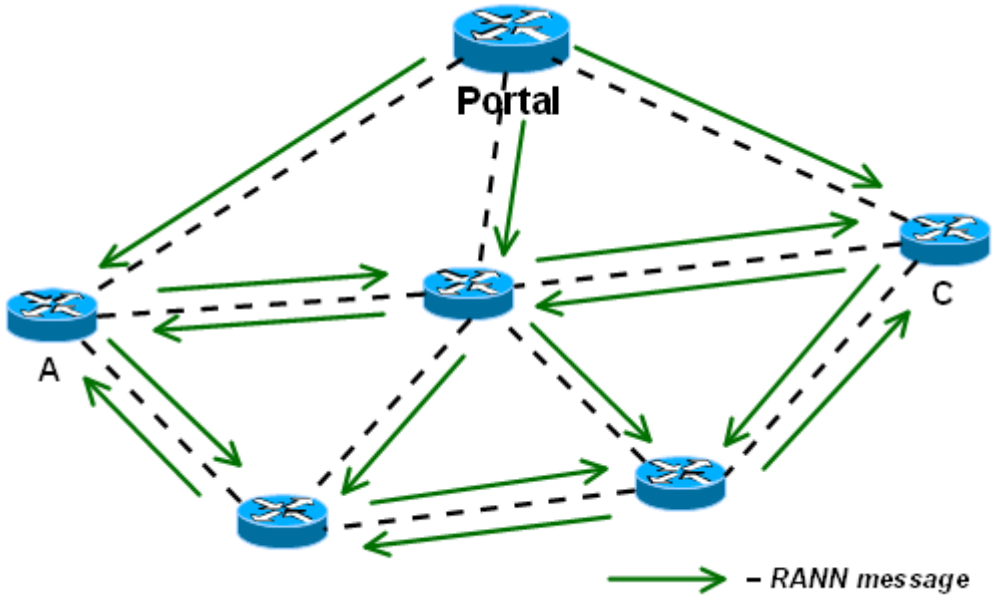
Router C 发送单播资料回复 A

在反应方式中 HWMP+是非常类似 AODV（Ad-hoc On-demand Distance Vector 按英文字面意思是自组网 按需 距离 向量网络，AODV 各移动节点并不持续维护实时描述整个网络拓扑的路由表,仅在有业务到达时才查找建立支持该业务交换的路由,从而节省了大量未必有效的路由管理控制开销）。

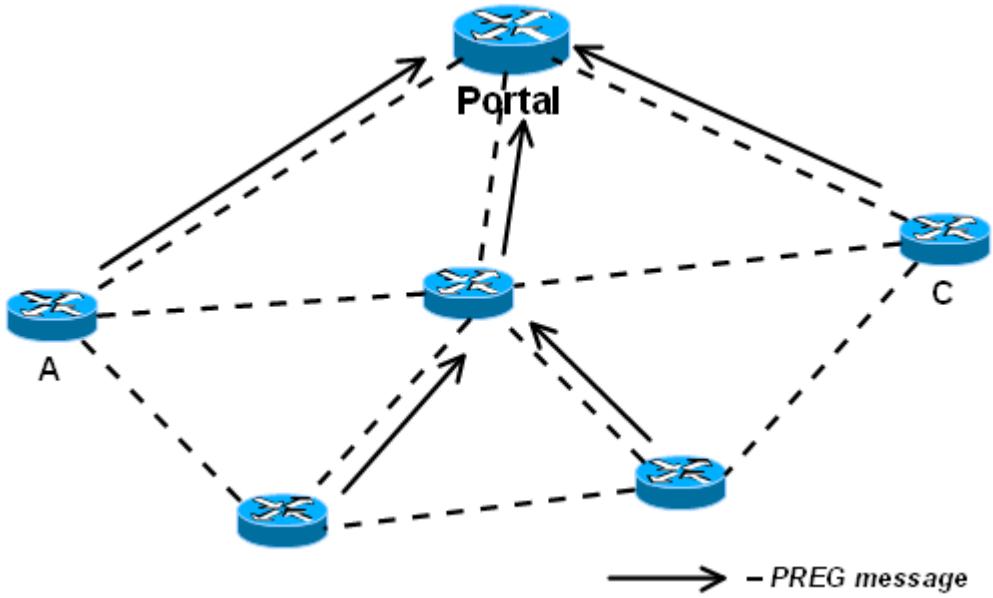
HWMP+在一经要求的情况下所有路径被探测到，通过在网络中不断发送路径请求信息(PREQ)。目的节点或者在路径上的路由器会回复路径信息(PREP)。注意：如果目标地址属于一个客户终端，该 AP 会为下面的客户终端提供代理(例如：答复 PREQ 以他的名义)。

这种模式适用于移动网络，或大部分的通信发生的内部 mesh 节点。

2、主动方式



根节点通过不断发送 RANN 信息



内部节点回应 PREG

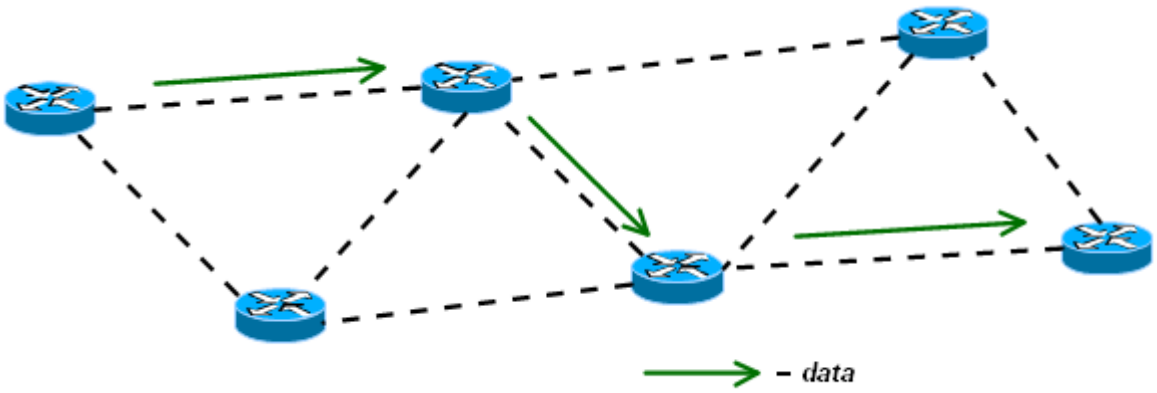
在主动方式下一些路由器配置为入口（protal），一般入口代表路由器有接口连接到其它的网络。

在网络中入口通过发送根消息（RANN）会宣布他是出入点。内部节点会响应一个路径注册信息（PREG），这样的结果是入口节点作为路径树的根节点。

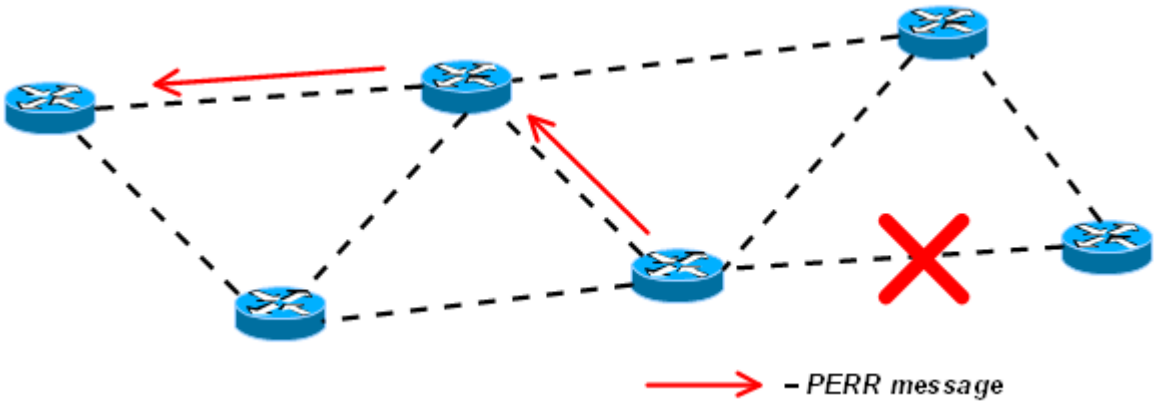
路径为入口将充当一种默认路由，如果一个内部路由没有找到指定的目的路径，将会把所有数据发送到最近的入口节点，如果可能，入口将作为代理路由寻找路径。这个可以引导向最佳的路径，除非数据被指定到入口节点本身，或者一些已有的外部网络接口入口节点。

主动方式更适合当多数传输在两个内部 Mesh 网络之间和存在多个入口节点。

3、拓扑变动探测



数据流路径



当连接消失，错误的上行数据

HWMP+ 使用路径错误信息(PERR)通告一个连接消失，这个信息会发送到所有的上行数据流节点返回到数据源，源节点接收到 PERR 后会重启路径探测。

第十章 CAPsMAN

Controlled Access Point system Manager(CAPsMAN)，即 AP 控制系统， 能集中管理无线网络，根据需要也能做数据处理。在 CAPsMAN 系统中包含数个 AP（**CAPsMAN 代表集中管理系统，CAP 代表被管理 AP**），CAPsMAN 为他们提供无线连接管理，维护客户端验证和数据转发

10.1 介绍

需要功能包：wireless-fp

当 CAP 被控制管理，将建立与 CAPsMAN 管理端的连接，通常 AP 自身控制的客户端，将交由 CAPsMAN 管理包括客户端的验证等。CAP 仅仅是维护无线连接层面的加密和译码，根据配置，数据将转发到管理端进行集中处理，后再转发到本地的 CAP。

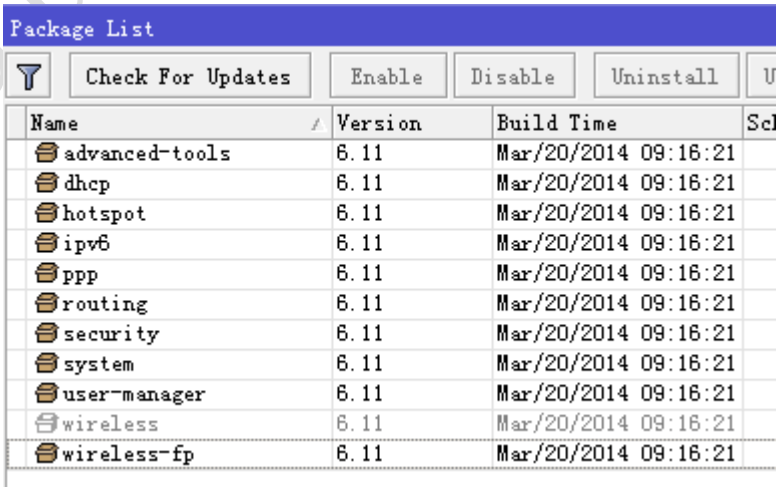
CAPs 管理功能

- RADIUS MAC 验证
- WPA/WPA2 安全加密
- TBA

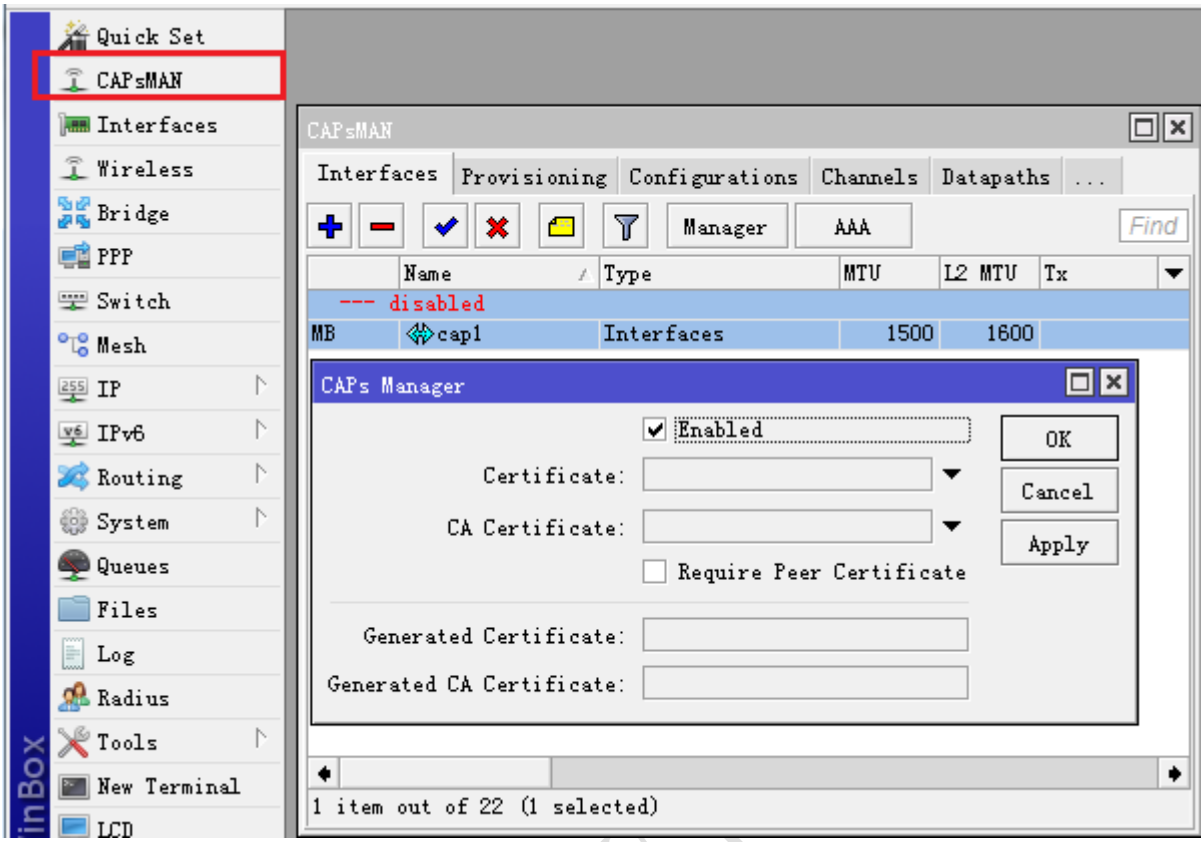
CAPs Manager 特点

- Nstreme AP 支持
- Nv2 AP 支持
- TBA

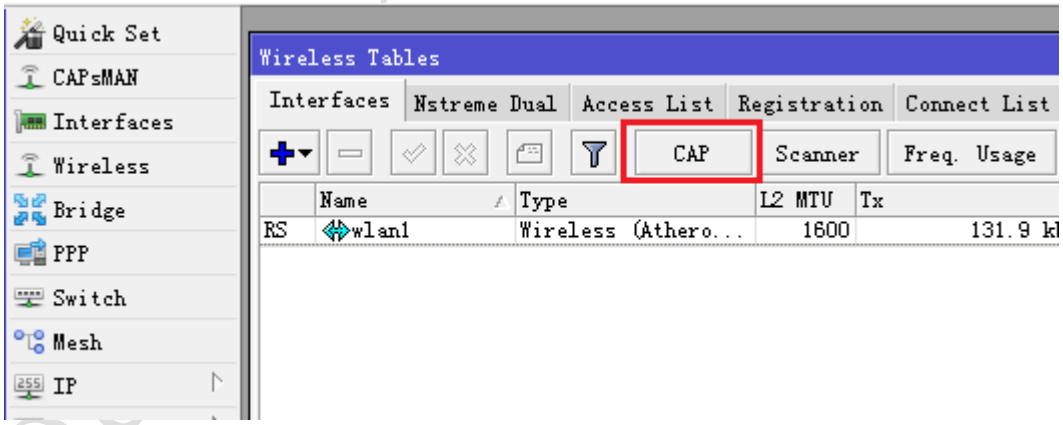
CAPsMAN 的功能包可以独立安装到任何平台的 RouterOS v6.11 以上版本，CAPsMAN 功能还在测试中，所以功能包为独立的 wireless-fp-6.11-mipsbe.npk，安装后将替代原理的 wireless 功能包，功能包升级后 wireless 功能包被自动禁用，wireless-fp 启用（该功能以后可能会合并）。



我们可以在 winbox 中看到 CAPsMAN 菜单：



进入 wireless 菜单，可以看到 CAP 选项



即 CAPsMAN 为 CAP 管理器菜单，CAP 为被管理网卡菜单

10.2 CAP 连接到 CAPsMAN

CAPsMAN 系统要为无线网络提供管理控制，需要至少一个 CAP 必须与 CAPsMAN 建立连接。一个管理连接建立可以使用 MAC 或 IP 层协议和安全的 DTLS。

通常一个 CAP 能传递客户端数据连接到 CAPsMAN 管理器，但数据连接并不安全，因此需要考虑数据安全的加密，例如 IPSec 或其他加密隧道。CAP 连接到 CAPsMAN 过程如下：

1、CAP 连接到 CAPsMAN 能基于二层和三层（MAC 层和 IP 层）建立连接

MAC 层连接特性：

- 没有 IP 配置到 CAP
- CAP 和 CAPsMAN 必须在相同二层网络中，二层交换或虚拟网络（二层隧道，例如 EoIP）

IP 层连接（UDP）特性：

- 如果需要能穿透 NAT
- CAP 必须通过 IP 协议连接到 CAPsMAN

注意：基于三层连接时，如果 CAP 与 CAPsMAN 没有在相同二层网络，必须为 CAPsMAN 分配 IP 地址，且两端路由可达。

2、为了与 CAPsMAN 建立连接，CAP 会执行一个探测操作，在探测周期里，CAP 会试图连接 CAPsMAN，并创建一个可以运行的 CAPsMAN 列表。CAP 连接一个可运行的 CAPsMAN 会进行如下操作：

- 配置管理 IP 地址
- 从 DHCP 服务器获取 CAPsMAN IP 地址
- 配置接口能通过 IP 和 MAC 层广播

当获取 CAPsMAN 列表建立后，CAP 选择一个 CAPsMAN 连接基于以下规则：

- 如果 **caps-man-names** 参数指定管理名称（**/system identity** 作为 CAPsMAN）CAP 将优先选择。如果该参数为空，将连接其他 CAPsMAN。
- MAC 层连接优先级高于 IP 层连接

3、当 CAPsMAN 管理被选择后，CAP 尝试建立 DTLS 连接，以下是验证模式：

- 没有证书在 CAP 和 CAPsMAN，不用验证
- 只有 CAPsMAN 管理器能配置证书 - CAP 会检查 CAPsMAN 的证书，这时 CAPsMAN 必须配置为 **require-peer-certificate=no**，CAP 无需配置证书，即可以验证
- CAP 和 CAPsMAN 都配置证书，证书相互验证

4、在 DTLS 连接建立后，CAP 会检查 CAPsMAN 证书提供的 **CommonName** 字段
caps-man-certificate-common-names,这个字段参数列表包含被允许的 CommonName 值。如果这个列表不为空，CAPsMAN 必须配置证书，如果列表为空，CAP 不会检查 CommonName 字段。

CAP 自动锁定 CAPsMAN

CAP 能配置自动锁定特定的 CAPsMAN， 锁定被执行通过记录 CAPsMAN 的证书 CommonName 字段，CAP 锁定并检查所有后续的 CommonName 连接 ，这个功能执行使用证书的 CommonName 字段。

锁定功能启用通过一下命令：

```
[admin@CAP] > /interface wireless cap set lock-to-caps-man=yes
```

10.3 Datapath 配置

Datapath 设置对各 CAP 资料相关的转发，在 CAPsMAN datapath 中设置 datapath 策略，路径为 **/caps-manager datapath** 或者直接在 configuration 中设置，也能在 interface 中配置

Datapath 有两个主要转发模式：

- **local forwarding mode**（本地转发模式），即由 CAP 通过无线本地接口转发数据
- **manager forwarding mode**（管理转发模式）, CAP 将所有接收到资料发送到 CAPsMAN，CAPsMAN 处理后再发送数据到 CAP。在这个模式包括客户端到客户端之间转发都通过 CAPsMAN 控制

转发模式配置针对的是每张无线网卡，而非每台设备，因此如果一个 CAP 有 2 张无线网卡，一张可颐配置为本地转发模式，一直为管理转发模式。这也同样适用于 Virtual-AP 网卡，不管是 Virtual-AP 还是其主无线网卡（master-interface）都可以选择不同的转发模式

大多情况下 datapath 设置会选择管理转发模式，因为本地转发模式 CAPsMAN 不能统一控制 CAP 的数据转发，对于统一的 CAP 群控制和用户数据转发有好处

下面是 datapath 设置：

- bridge – 添加到指定的 bridge 接口，即将 CAP 加入指定的桥接，前提是 bridge 接口已经配置
- bridge-cost – 当 bridge 端口添加，可设置该端口成本开销值
- bridge-horizon – 当 bridge 端口添加，可设置水平分割桥接预防桥接环路
- client-to-client-forwarding – 客户端到客户端连接到无线网卡后，用于控制他们之间的数据转发，在本地转发模式该功能由 CAP 自行处理，管理转发模式则有 CAPsMAN 处理
- local-forwarding – 控制转发模式
- openflow-switch – 当 openflow 协议配置后，可添加到 OpenFlow 交换器中
- vlan-id – 当 vlan 模式启用，且用 vlan 标签，这时设置的 vlan id 将分配到网卡界面上
- vlan-mode –指定 vlan 模式，即 vlan 设置为打标签、还是不打标签

Local Forwarding 模式

在这个模式下无线网卡作为 CAP 处理类似于一个普通的 AP，并直接将正常数据转发。CAPsMAN 将不参加数据转发和处理任何数据帧，只控制无线网卡的配置和客户端连接和分配处理

当无线网卡启用为 CAP，配置将会被修改，即状态和一些相关参数会与 CAPsMAN 相关联（例如：mac-address、arp 和 mtu），注意无线网卡关联的配置不会应用到实际的无线网卡配置中，而是通过 CAPsMAN 控制 ：

```
[admin@CAP] /interface wireless> pr
Flags: X - disabled, R - running
 0  R ;;; managed by CAPsMAN
     ;;; channel: 5180/20-Ceee/ac, SSID: master, local forwarding
     name="wlan2" mtu=1500 mac-address=00:03:7F:48:CC:07 arp=enabled
     interface-type=Atheros AR9888 mode=ap-bridge ssid="merlin"
     frequency=5240 band=5ghz-a/n channel-width=20/40mhz-eC scan-list=default
     ...
```

Virtual-AP 网卡在 local forwarding 模式下将显示为启用和动态的 Virtual-AP 网卡：

```
[admin@CAP] /interface> pr
Flags: D - dynamic, X - disabled, R - running, S - slave
 #    NAME                            TYPE        MTU L2MTU  MAX-L2MTU
 ...
 2  RS ;;; managed by CAPsMAN
      ;;; channel: 5180/20-Ceee/ac, SSID: master, local forwarding
      wlan2                           wlan       1500  1600
 3 DRS ;;; managed by CAPsMAN
      ;;; SSID: slave, local forwarding
      wlan6                           wlan       1500  1600
 ...
[admin@CAP] /interface> wireless pr
Flags: X - disabled, R - running
 ...
 2  R ;;; managed by CAPsMAN
     ;;; SSID: slave, local forwarding
     name="wlan6" mtu=1500 mac-address=00:00:00:00:00:00 arp=enabled
     interface-type=virtual-AP master-interface=wlan2
```

事实上 Virtual-AP 网卡是动态添加，某些时候设置为静态有助于 CAP 数据转发，例如分配 IP 地址到 Virtual-AP 网卡，这样的配置不会应用到 Master 无线网卡

为有助于数据转发配置，CAP 能配置到一个 bridge 中，且 bridge 能自动添加 CAP 端口到桥接交换中，当然前提是 CAPsMAN 的 bridge 已经启用，这个配置在**/interface wireless cap** 菜单下配置

Manager Forwarding 模式

在这个模式 CAP 发送所有无线到接收/发送数据到 CAPsMAN，CAPsMAN 控制所有的 CAP 的配置和转发数据，包括客户端到客户端之间的转发，Virtual-AP 网卡可以选择禁用 CAP 模式.

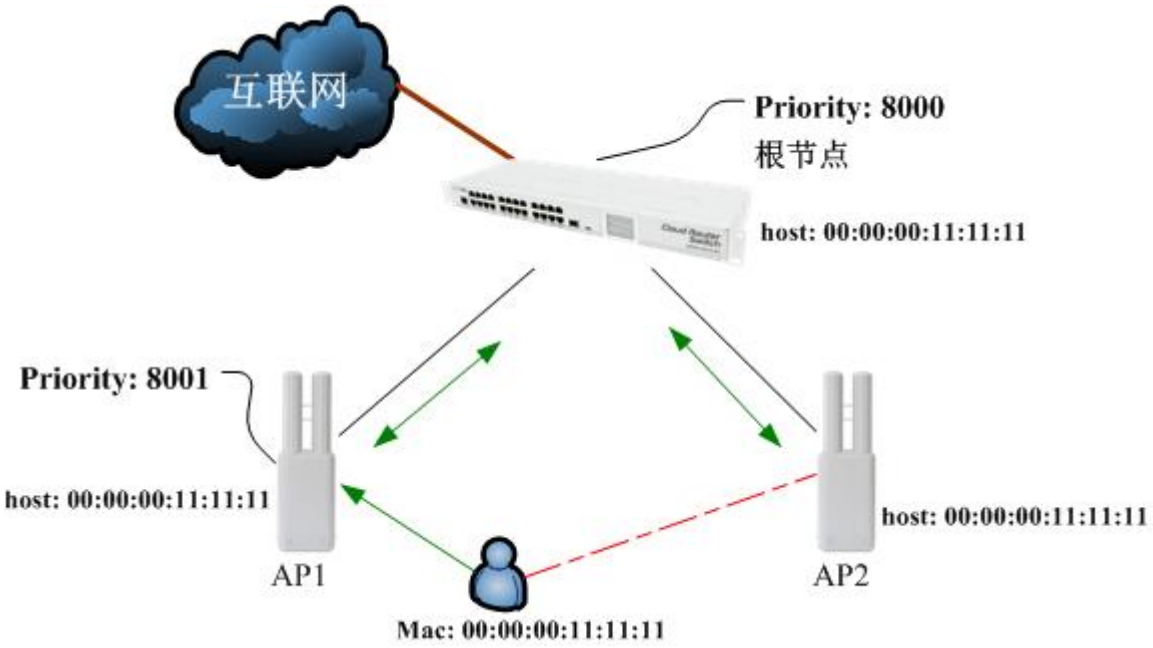
```
 ...
 1 X  ;;; managed by CAPsMAN
     ;;; channel: 5180/20-Ceee/ac, SSID: master, manager forwarding
     name="wlan2" mtu=1500 mac-address=00:03:7F:48:CC:07 arp=enabled
     interface-type=Atheros AR9888 mode=ap-bridge ssid="merlin"
 ...
```

Datapath 实例

关于 Datapath 的使用，这里我们以 Manger Forwarding mod（管理转发模式）来讲解下，管理转发模式不仅仅是希望统一管理 AP，也希望通过各个 CAPsMAN 将各个无线访问节点的客户端进行集中控制和漫游等，特别是漫游方面。其实 CAPsMAN 的 datapath 可以实现这部分，但仍然通过建立 bridge 后将用户都放到一个二层交换下，通过 RSTP 的模式来完成。

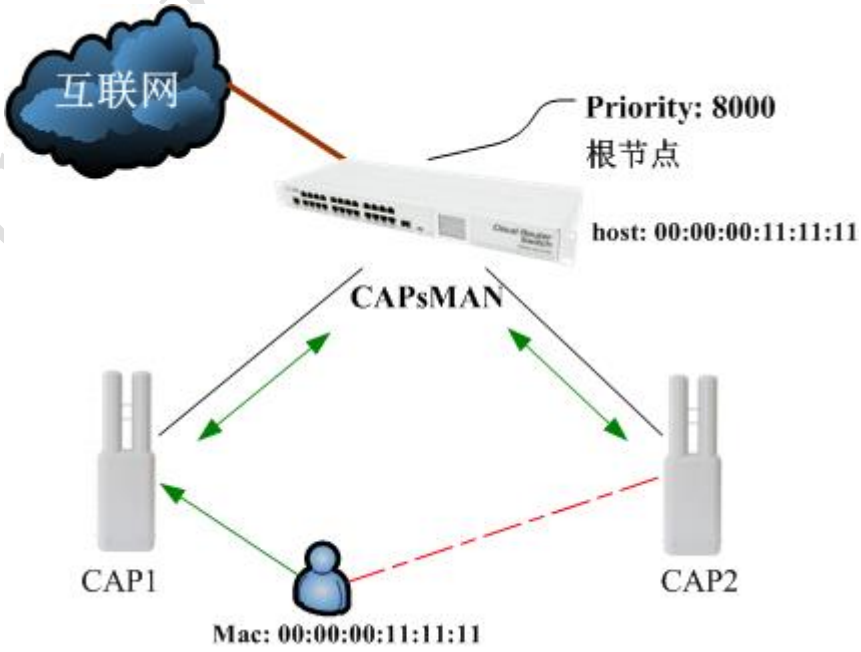
通常我们建立可漫游的无线网络，是才采用透明桥的 RSTP 网络，即将所有 AP 设置为透明桥接，连接到一个根节点的桥下，根节点的桥作为三层网关，并通过 DHCP 分配 IP 地址，或建立 nat 和热点网关等等。

如下图，两个 AP，分别都是独立的，都建立一个 bridge 和自己的优先级，有自己的 hosts 列表，当用户 mac：00:00:00:11:11:11 连接后，这个二层网络没划分 VLAN，所以每个设备都能学习到该用户 MAC，这样当用户在无线网络中切换时，用户几乎无感觉基本就在 1～2 个包的延迟（注意无线漫游切换选择取决于用户终端设备，而非 AP）



而现在 CAPsMAN 也是类似的 RSTP 网络，但不是每个 AP 都有自己的 bridge 和 hosts 列表，由于建立了 CAPsMAN 管理，下面的 CAP 会将所有无线资料发送到 CAPsMAN 处理，所以 bridge 只有一个，就是 CAPsMAN 管理器本身，可以理解为 CAPsMAN 就是一个大交换机。

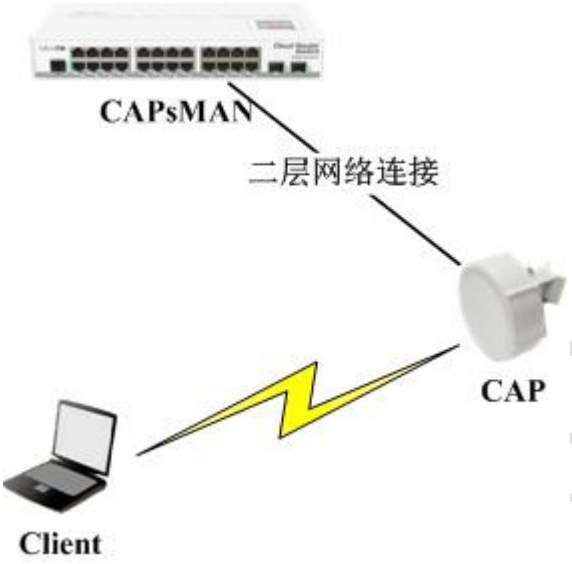
如下图，CAP1 和 CAP2 除了建立 WiFi 的无线连接，没有其他任何操作，bridge 建立在 CAPsMAN 上，即 hosts 管理则由 CAPsMAN 完成。



可以理解为，以前的 AP 网络是，每个 AP 都是一个交换机，都上联到上层的一台汇聚交换机，而现在的 CAPsMAN 则是所有无线终端都连接这台汇聚交换机，中间就不存在其他任何“AP 交换机”。

10.4 CAPsMAN 实例

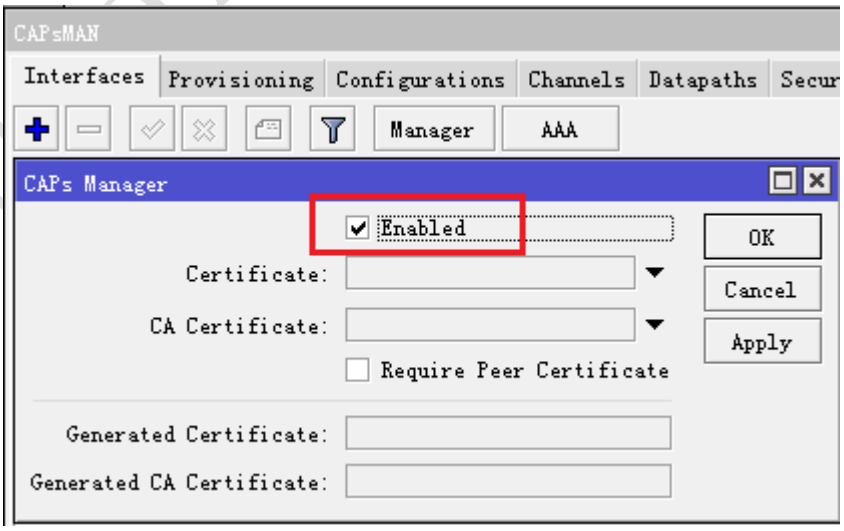
下面是一个简单的 CAPs 联网实例，假设我们网络中只有一个 CAPsMAN 和一个 CAP，他们之间通过二层网络连接，即采用二层 MAC 建立互连，无证书验证。



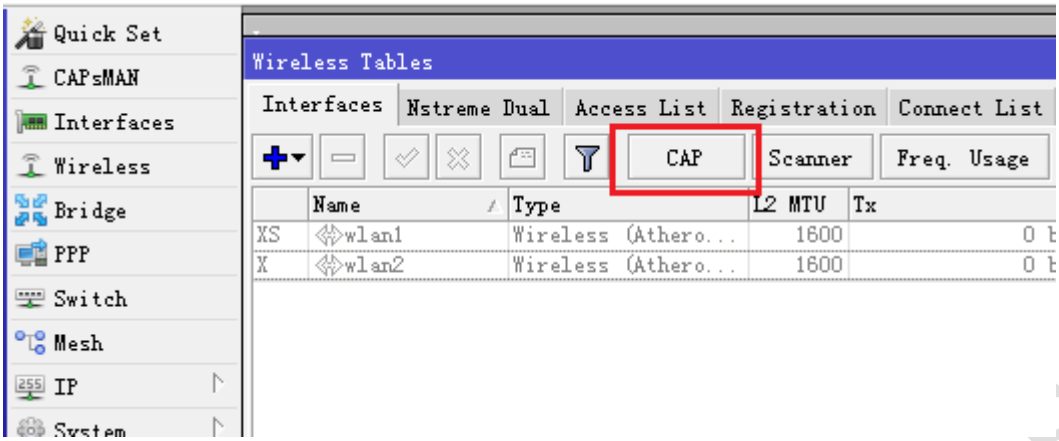
之前的介绍 CAP 与 CAPsMAN 连接，可以通过 MAC 层和 IP 层连接，CAP 也具备自动搜索 CAPsMAN 功能，也可以通过 caps-man-names 连接，这里 CAPsMAN 与 CAP 连接基于二层，且这个二层网络仅有一台 CAPsMAN，所以我们的建立变的简单。

启用 CAPsMAN

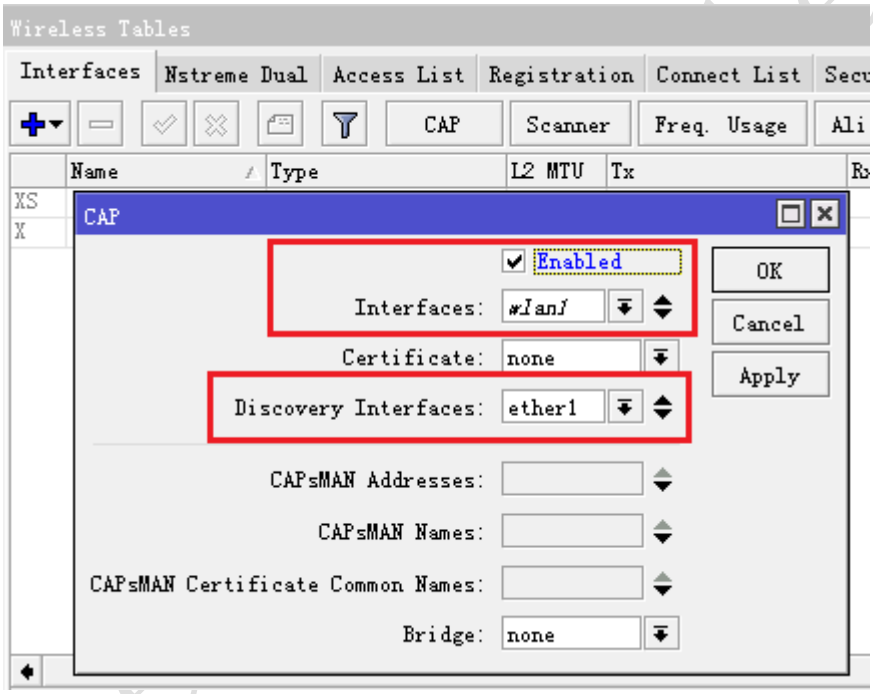
由于 CAPsMAN 与 CAP 基于二层网络互连，且仅只有一个 CAPsMAN，所以我们仅启用 CAPsMAN，CAPsMAN 功能可以安装到任何 RouterOS 平台上，我用 RB750 作为 CAPsMAN 管理器。



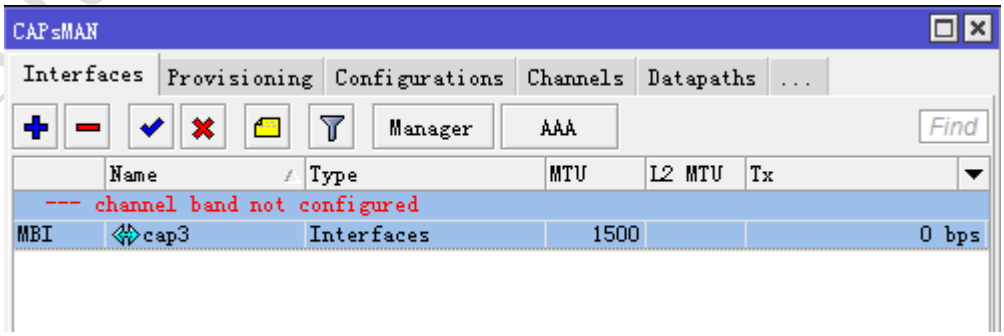
启用 CAPsMAN 后，需要将 CAP 连接到 CAPsMAN 上，这里我们是一台 RB411，通过 ether1 连接到 RB750，首先我们需要在 CAP 上开启功能，进入 wireless 选项，选择 CAP 菜单



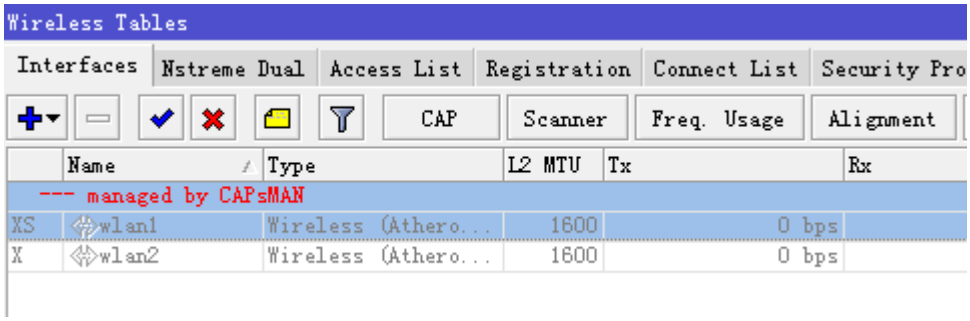
进入 CAP 菜单后，我们开启 CAP 功能，选择 wlan1 为被管理网卡，探测 CAPsMAN 网卡为 ether1



配置完成后,CAP 开始自动搜索二层网络内的 CAPsMAN 管理器，大约几秒钟后 CAP 连接上 CAPsMAN，如下图

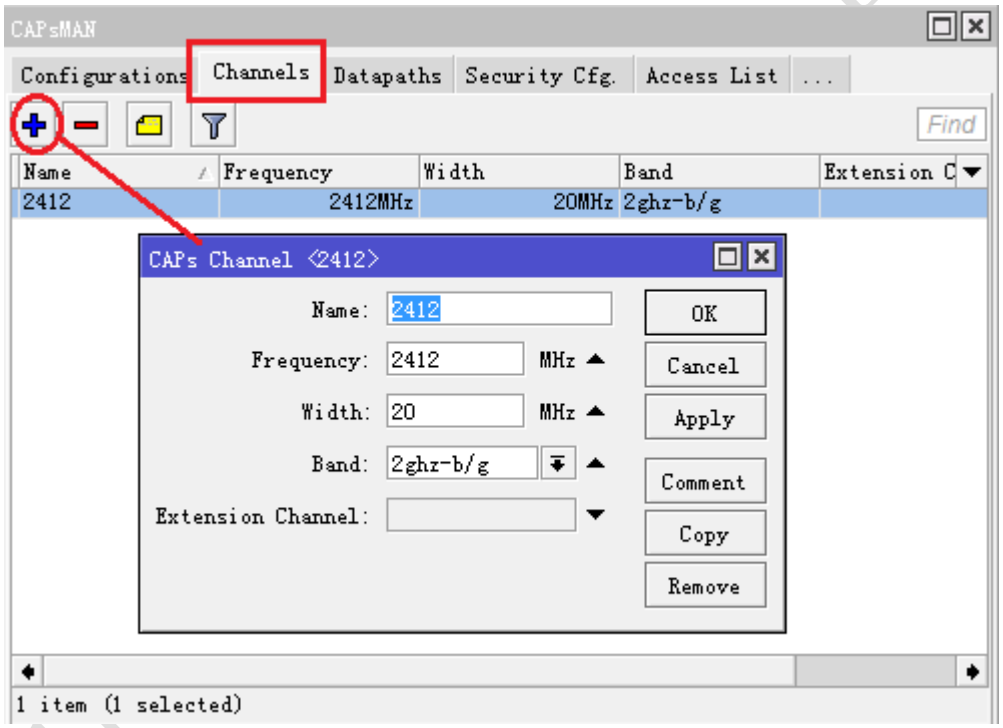


但当前标识状态看 cap3 为 MBI，M 代表主设备，B 代表被绑定，I 代表未启动，因为 cap3 没有配置无线相关参数，我们再来看看 CAP 的状态

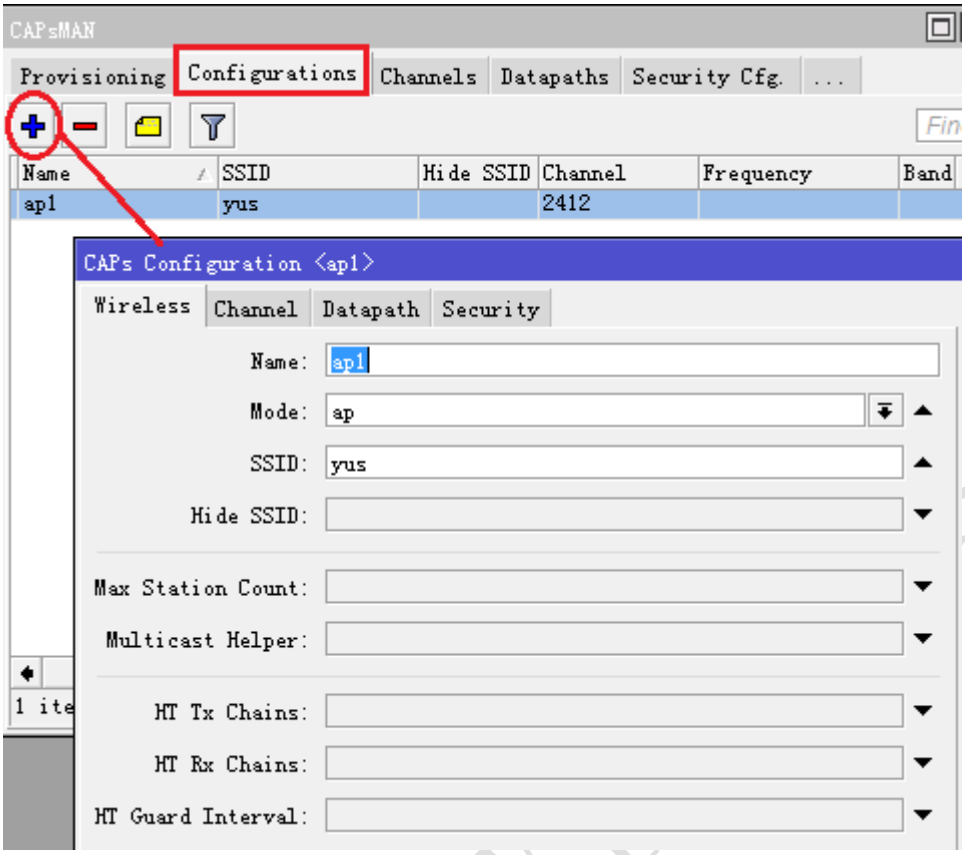


从上图中可以看到 managed by CAPsMAN，即被 CAPsMAN 管理，wlan1 无线网卡处于禁用状态，是无法被本地的 RouterOS 管理。

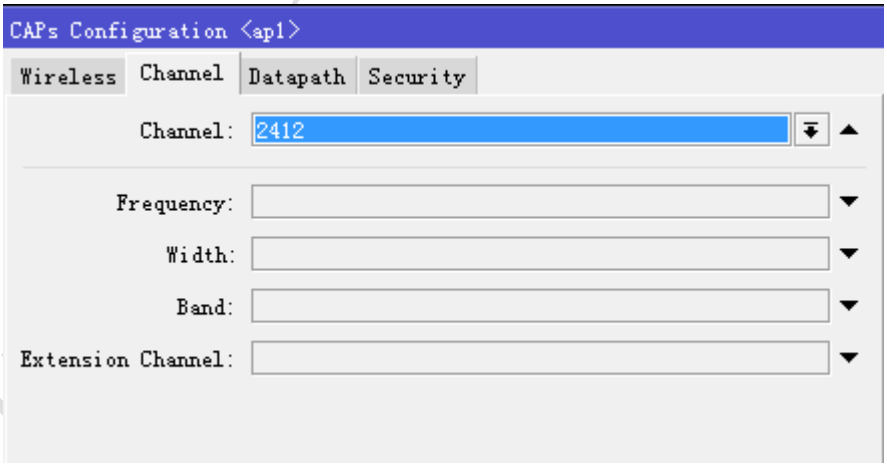
跟着我们要为 CAPsMAN 连接的 CAP 配置无线参数，连接的 wlan1 是一张 802.11bg 网卡，我们需要在 CAPsMAN 配置相关的 bg 无线参数，首先定义频道，取名 2412，频段 2ghz-b/g，发射频率 2412MHz，带宽 20MHz。



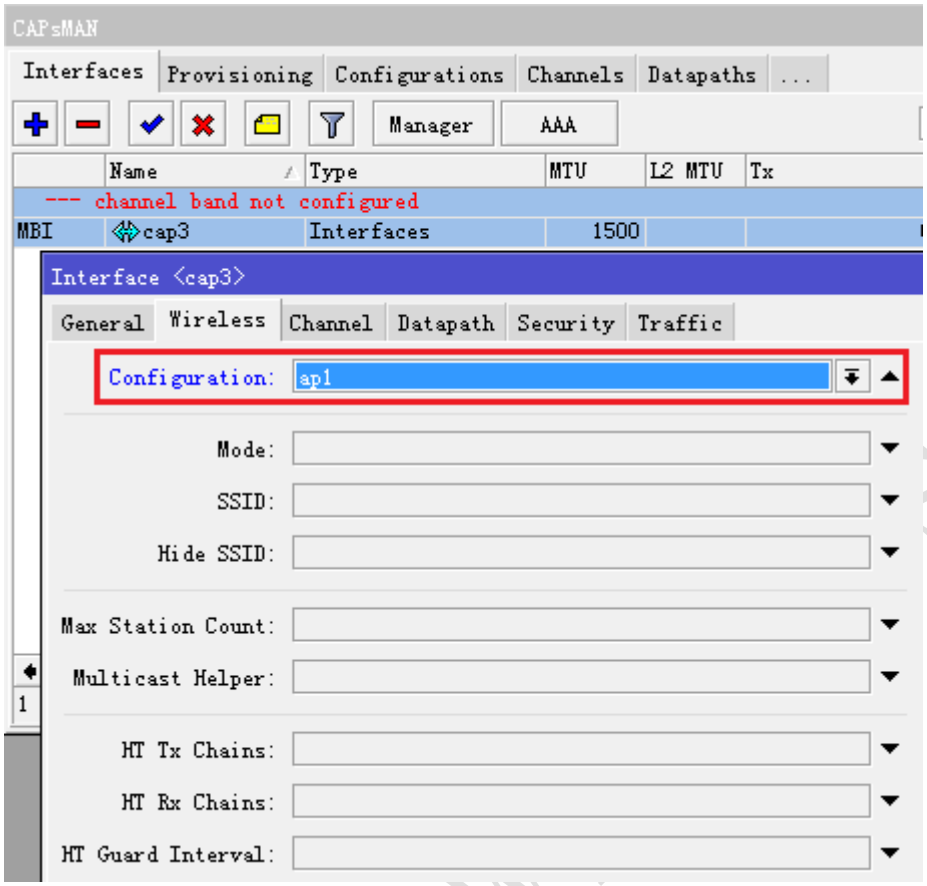
定义配置规则组，进入 configurations 下，添加一个规则组取名 ap1，mod 为 ap，SSID 设置为 yus



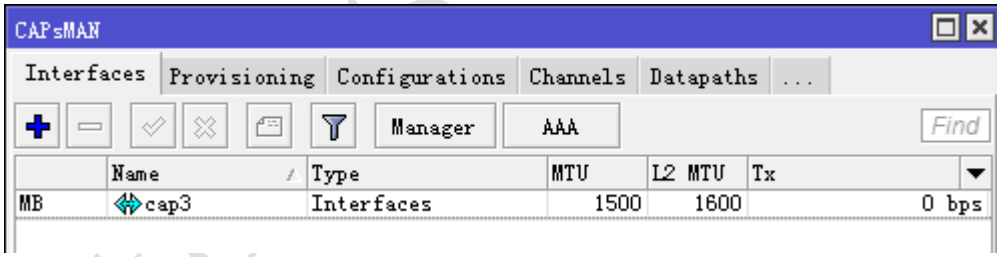
选择 Channel 设置频率，我们将之前设置的 2412 频率选择上



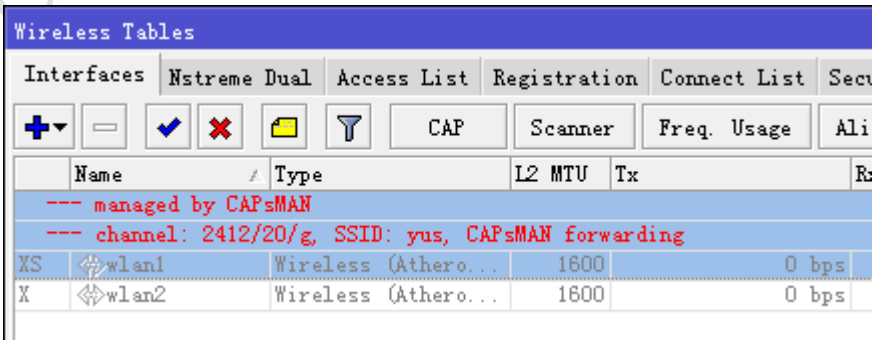
当配置规则组完成后，返回 interface 菜单下，选择 cap3，进入 wireless 菜单，直接选择 configuration 为 ap1



配置完成后，我们可以看下 cap3 的状态，已经没有 I 标识



再看看 CAP 状态，清楚的显示 wlan1 的频率和 SSID 等信息



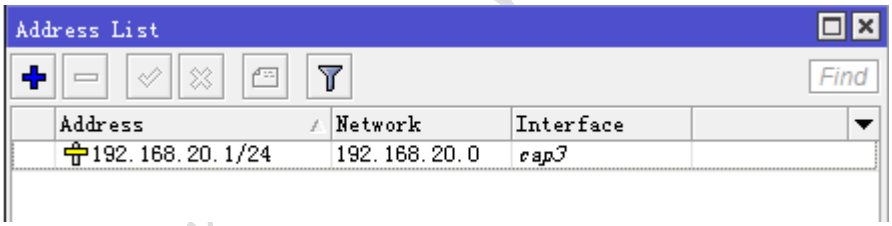
通过计算机搜索到 yus 的无线信号，没有设置加密所以不安全

这样基本的无线已经配置完成，但对于这个无线的 CAPsMAN 系统仅仅完成一半，通常情况下我们需要为客户端分配 IP，不管是热点认证还是直接上网，都要让客户端自动获取 IP（特殊环境除外，如固定 IP），下一步肯定是要做 DHCP 服务器，

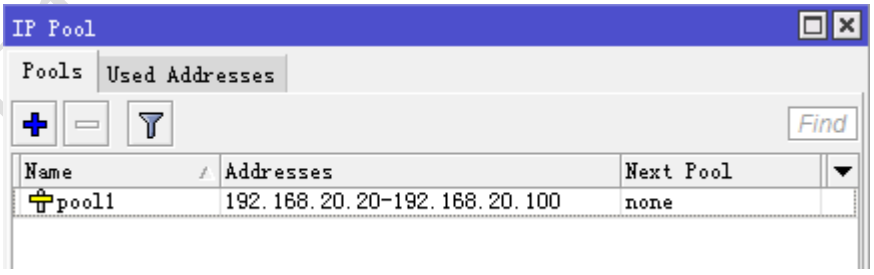
DHCP 服务器不是在 CAP 上做，而是在 CAPsMAN 上，因为 CAP 的 wlan1 网卡不在接受本地 RouterOS 的管理，已经从属于 CAPsMAN 的 RouterOS 上，cap3 已经是 CAPsMAN 的 RouterOS 的一张虚拟无线网卡，所以我们的 DHCP 服务器是在 CAPsMAN 上完成。

DHCP 配置就简单过下：

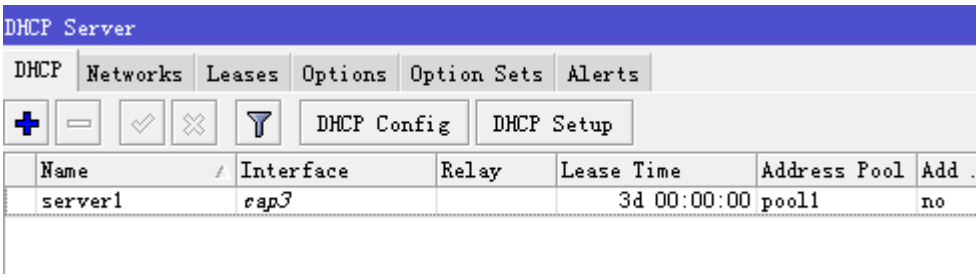
在 ip address 中为 cap3 分配 ip 地址 192.168.20.1/24



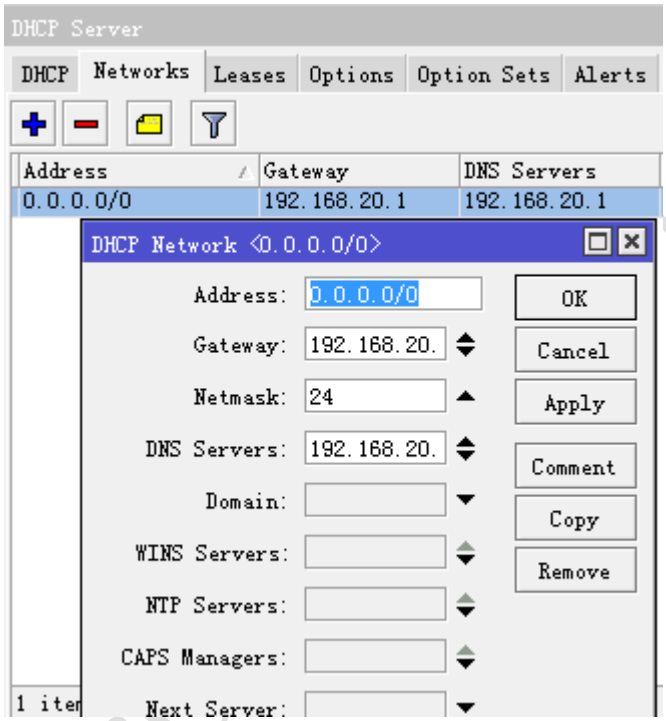
在 ip pool 建立地址池



建立 DHCP 服务器到 cap3



设置 DHCP 网络获取的网关、子网和 DNS



DHCP 建立完成后，我们可以获取到 cap3 无线分配的 IP 地址

第十一章 WLAN 的认证服务应用

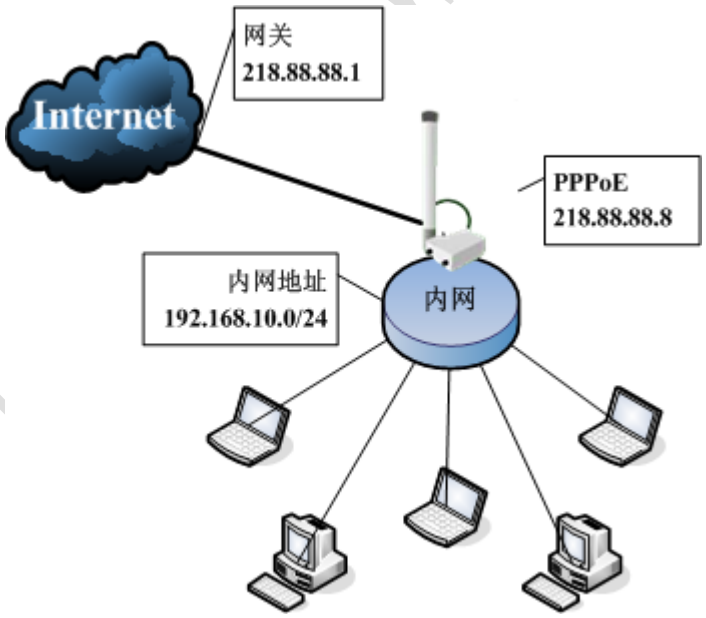
在 WLAN 中我们可能会涉及到网络认证功能，如 PPPoE 认证和 Hotspot 热点认证等，在这些应用中都是要求客户通过账号密码验证，才能连接到互联网或者访问指定的数据。这里我们主要介绍 PPPoE 和 Hotspot 的认证方式。

注：PPPoE 主要基于二层链路认证，要求 WLAN 网络基于桥接模式传输的，如中间有路由结构的 WLAN 的设备，就无法透传到路由后的网络。Hotspot 则可以基于二层和三层网络的传输，

什么地方选择什么样的认证方式，需要特定的考虑，当前主流的认证方式是 Hotspot 热点认证，因为这种认证采用基于 web 的 http 验证更加方便快捷，特别适合在人流量大的地方选择 Hotspot 热点认证，如酒店、图书馆、咖啡厅、公园、火车站和机场等，而相对固定的网络可以考虑 PPPoE 认证，但 PPPoE 认证对 pad 或手机终端设备有影响，因为他们默认是不支持 PPPoE 拨号的，这就需要对特点的网络环境进行考察评估，选择相应的认证方式。（关于 PPPoE 和 Hotspot 认证详细配置可以参考**《RouterOS 中文网络教程》**）

11.1 基于 PPPoE 的 WLAN 认证

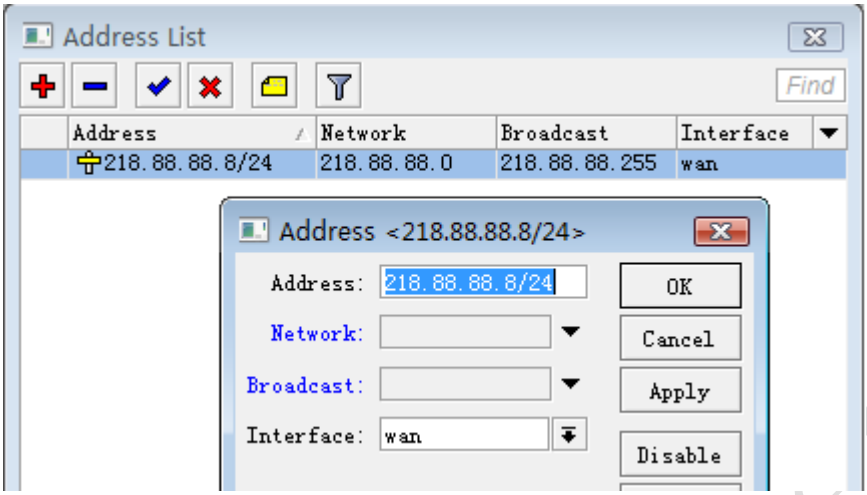
建立 PPPoE 认证我们一般是通过 WDS 桥接模式构建 WLAN 网络，因为只有二层链路才能透传 PPPoE 数据，构建网络如下：



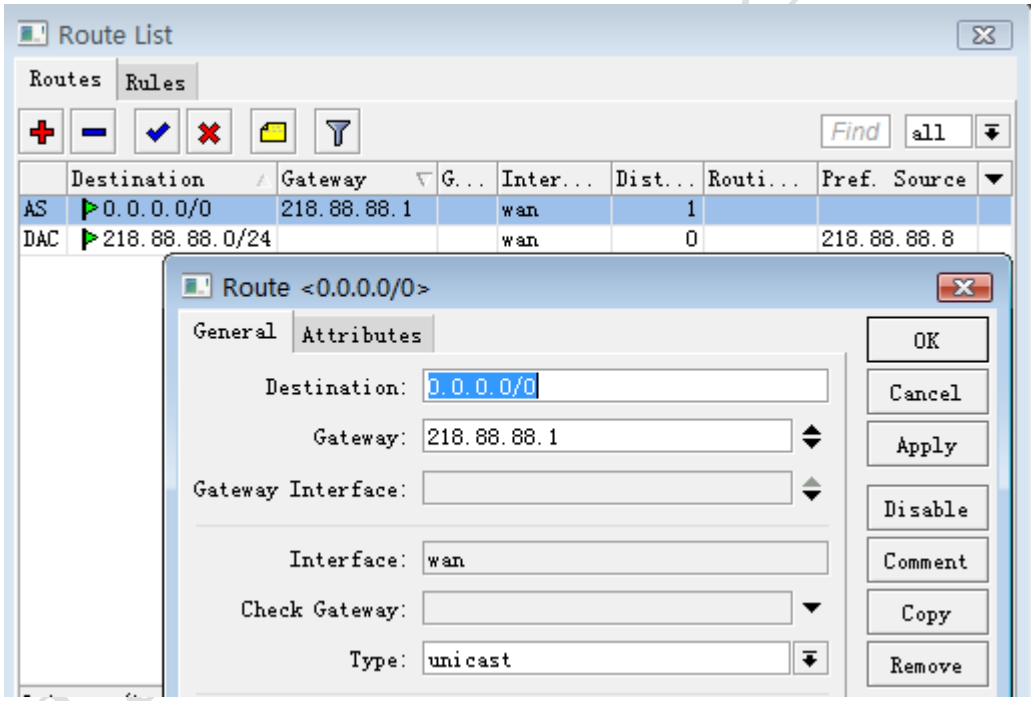
wan 口的外网连接 IP 地址是 218.88.88.8/24，内网因为是 PPPoE 拨号认证，我们可以不用在内网接口设置 IP 地址，只需要建立 PPPoE 服务器分配建立隧道连接的 IP 即可，构建 PPPoE 的 WLAN 认证网络步骤如下：

1、 配置基本网络参数 IP 地址、网关、nat 和地址池，并配置 wlan 的连接；
2、 在 PPP 中添加 PPPoE 服务器；
3、 配置 PPPoE 的 Profiles 用户组规则，并在 Secrets 中添加用户账号；
4、 测试 PPPoE 拨号连接。

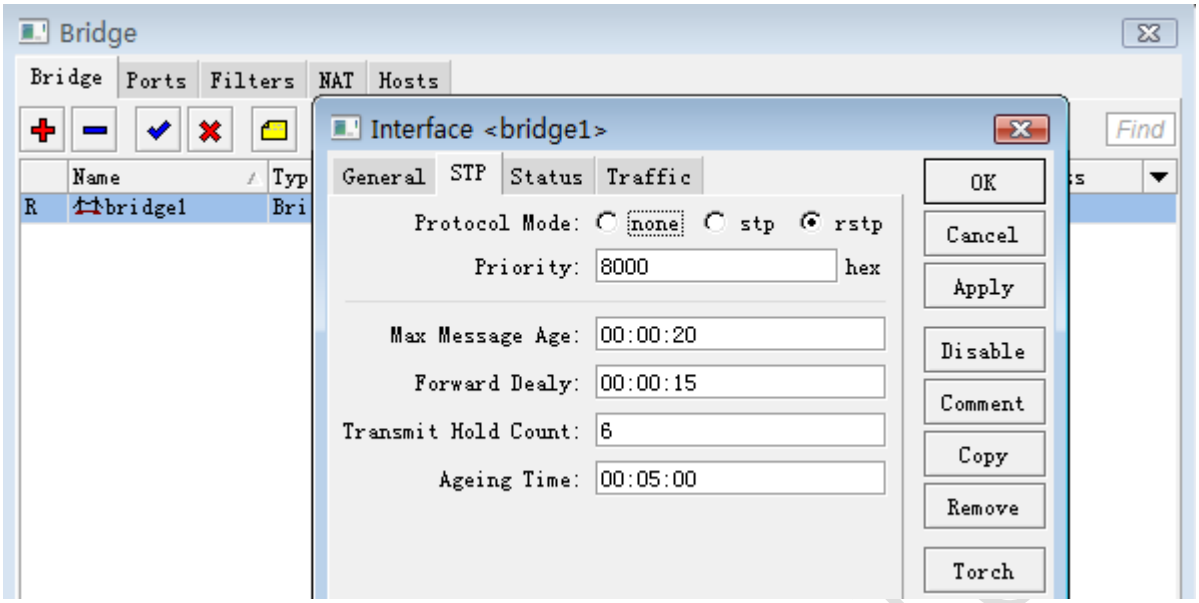
步骤 1：我们先添加 wan 口的外网 IP 地址：



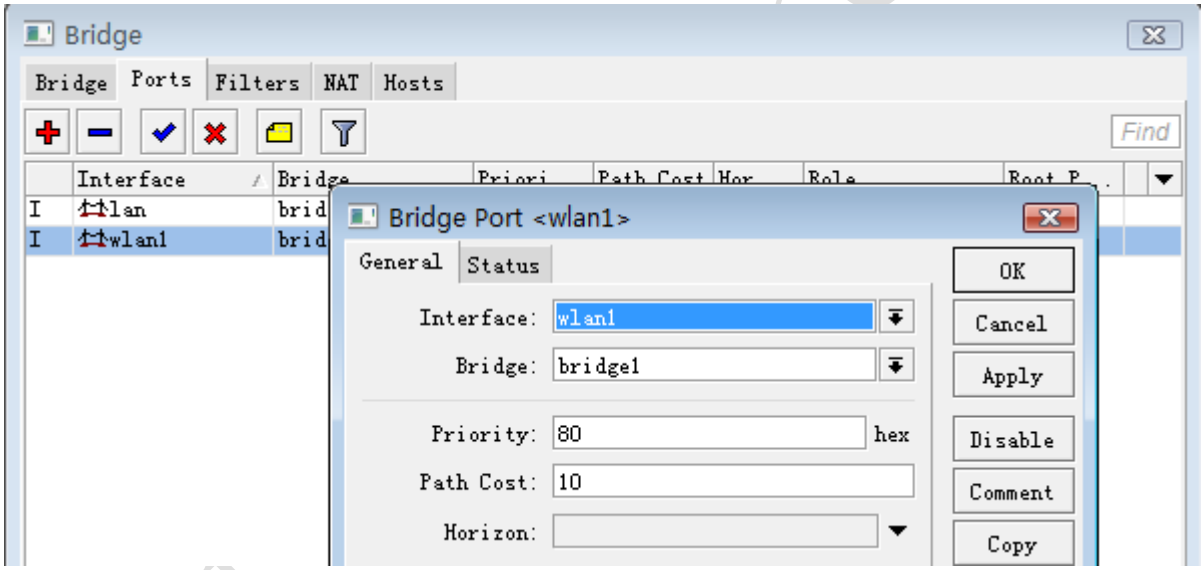
接着我们进入 ip route 配置路由，网关为 218.88.88.1



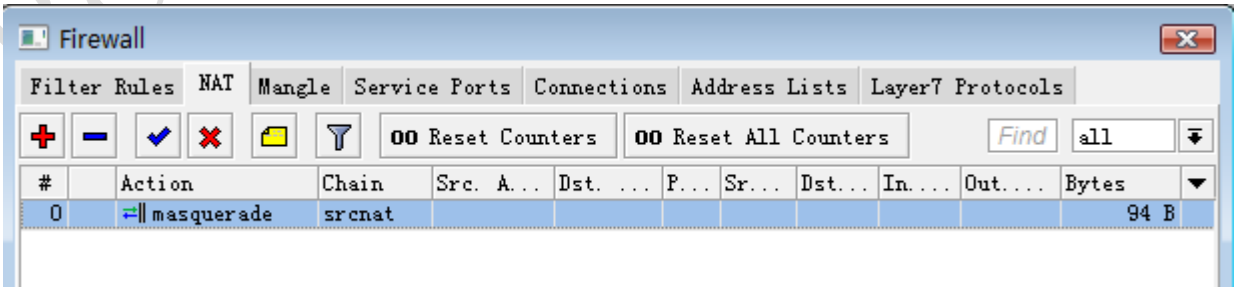
进入 bridge 在我们添加桥接，并设置 rstp 的参数，将 lan 口和 wlan1 设置到 bridge1 中：



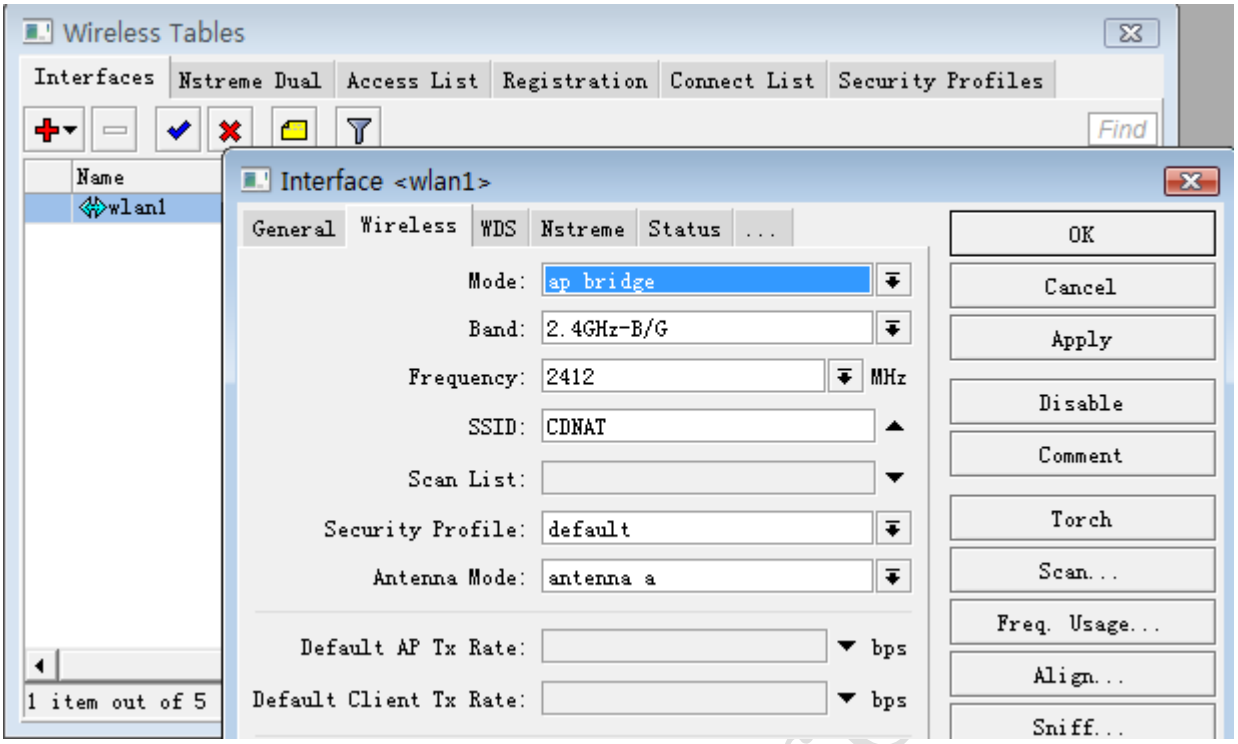
将 lan 和 wlan1 添加如 bridge1 中：



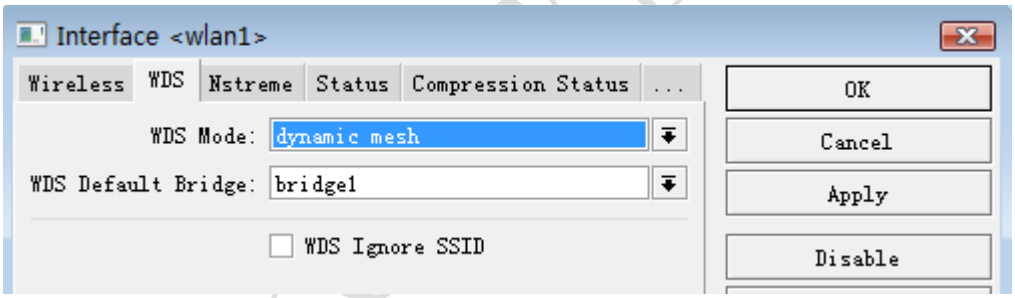
之后我们进入 ip firewall nat 配置 src-nat 的伪装策略 action=masquerade，用于隐藏内部的私有 IP 地址连接上网，如下图：



接下来配置 wlan1 的无线模块，我们这里以 2.4G-bg 为主，配置 mode=ap-bridge、Band=2.4G-B/G、Frequency=2412、SSID=CDNAT、WDS-Mode=dynamic-mesh、wds-default-bridge=bridge1，配置如下图：



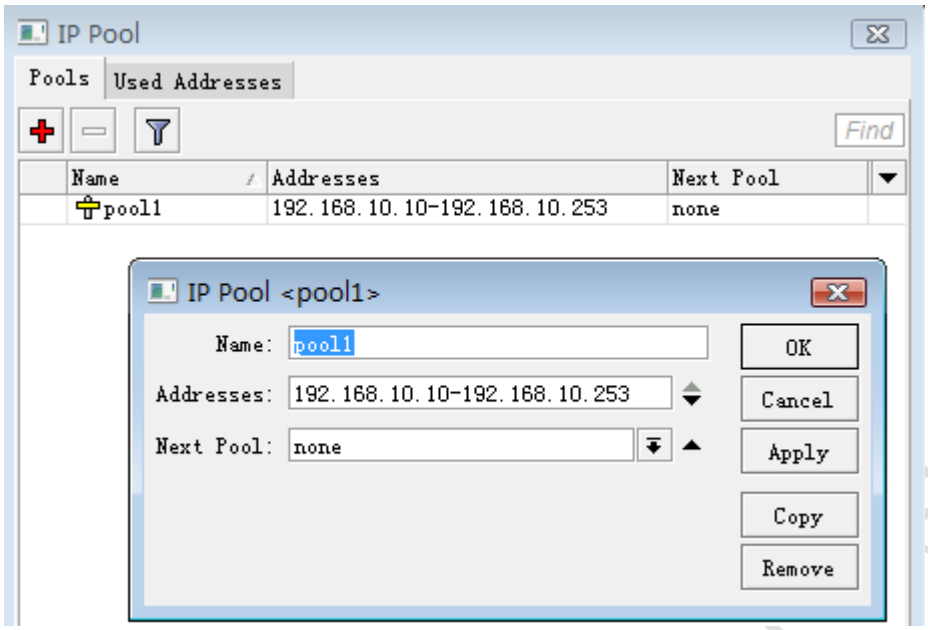
在 wlan1 配置 WDS 选项，设置 WDS-Mode=Dynamic-Mesh，比将其添加到 bridge1 中：



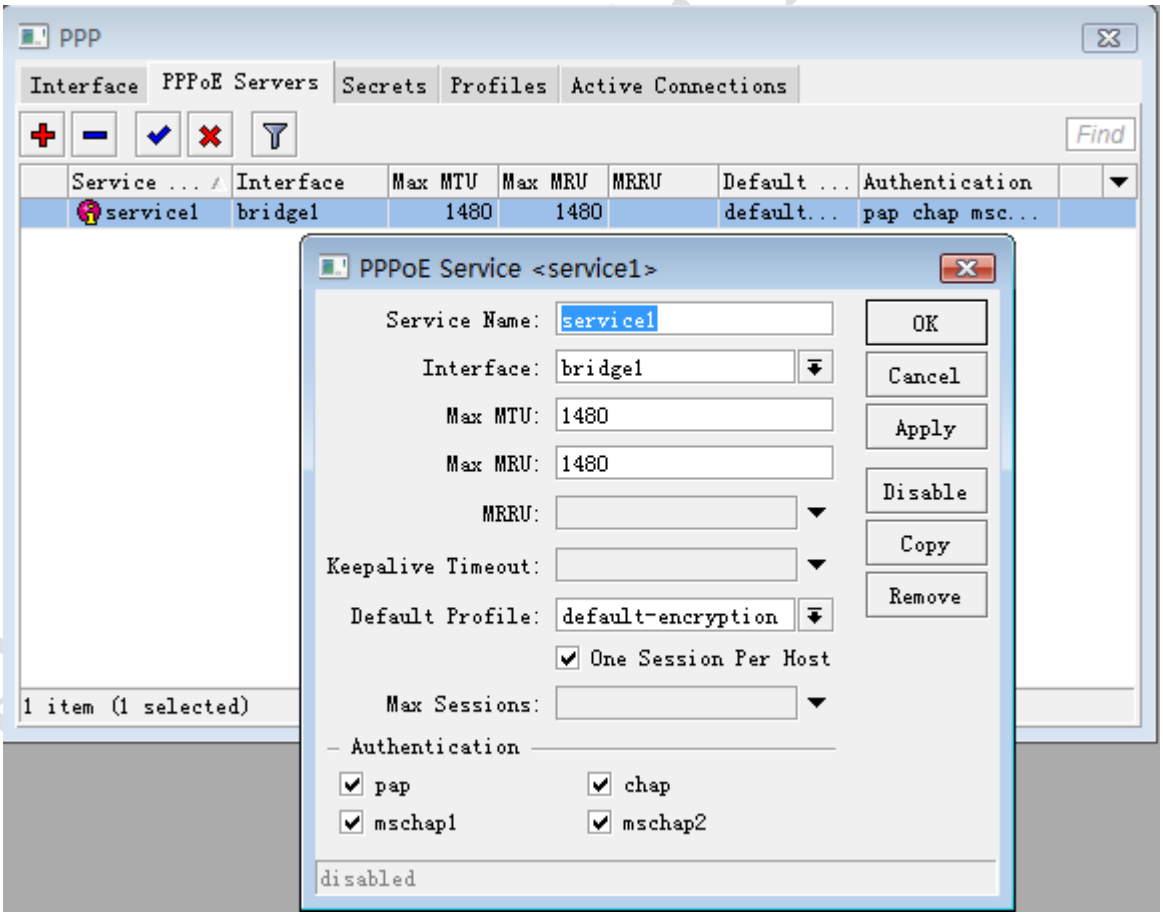
当 WLAN 无线配置完后，配置 PPPoE 服务，PPPoE 服务器参数如下：

1、 用户分配地址段：192.168.10.10-192.168.10.253
2、 用户网关地址：192.168.10.1
3、 DNS 服务器：61.139.2.69
4、 用户带宽为：下行：2Mbps，上行：1Mbps

首先我们需要进入 ip pool 中添加 PPPoE 分配给用户的地址池：



步骤 2：配置完基本参数后，现在进入 ppp 的 PPPoE-server 选项，启用 PPPoE 服务器，将服务器的 interface 设置到 bridge1 上，Default-Profile 选择 default-encryption，并将 one-session-per-host 打上勾，验证方式 Authentication 都选择上：



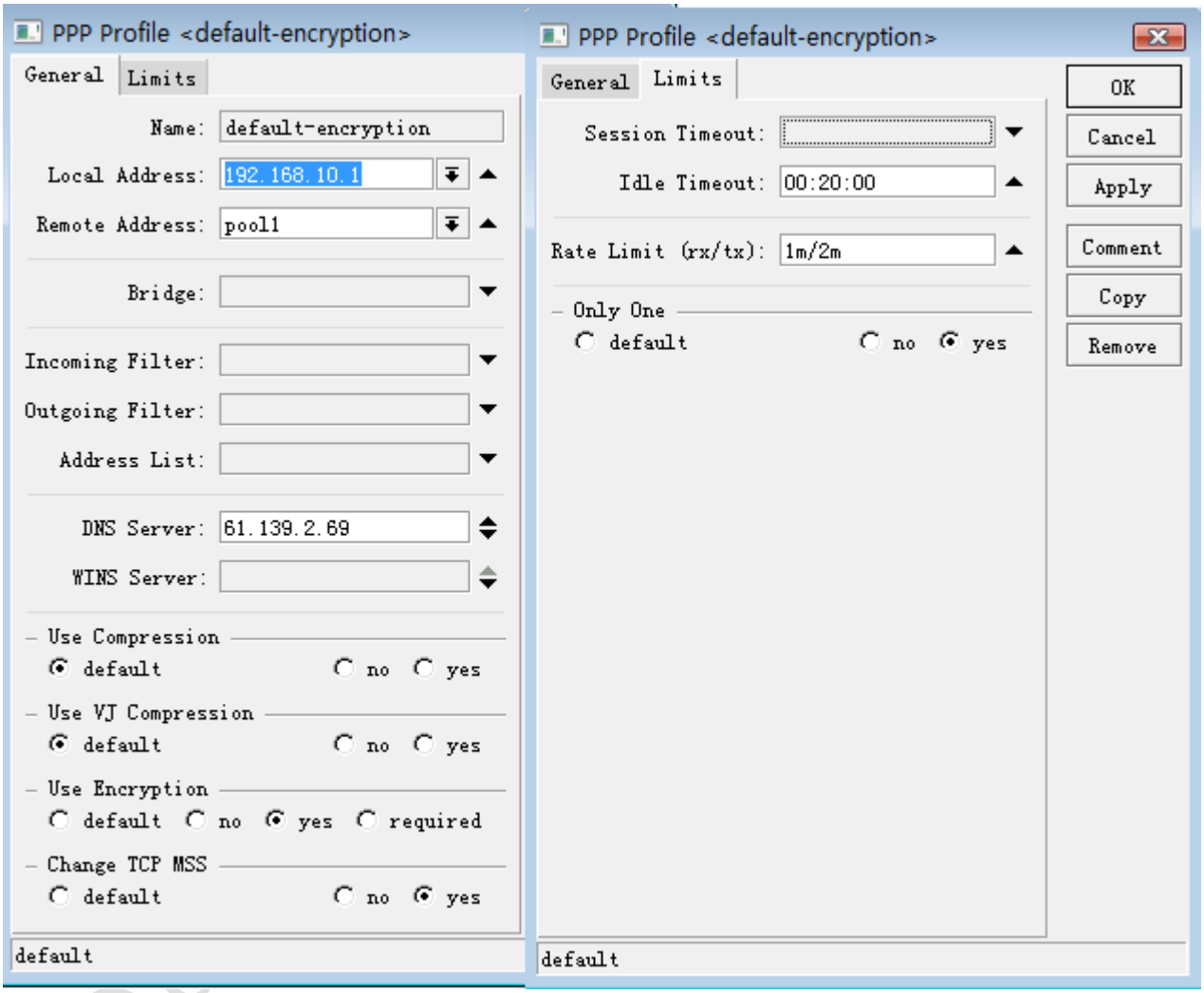
步骤 3：进入 profile 中配置 profiles 规则，这个是配置用户组规则，即不同的使用者类型分配一个 profile 类型，这里我们使用默认的 default-encryption 规则

1、 Local-address 为本地路由器的网关地址，

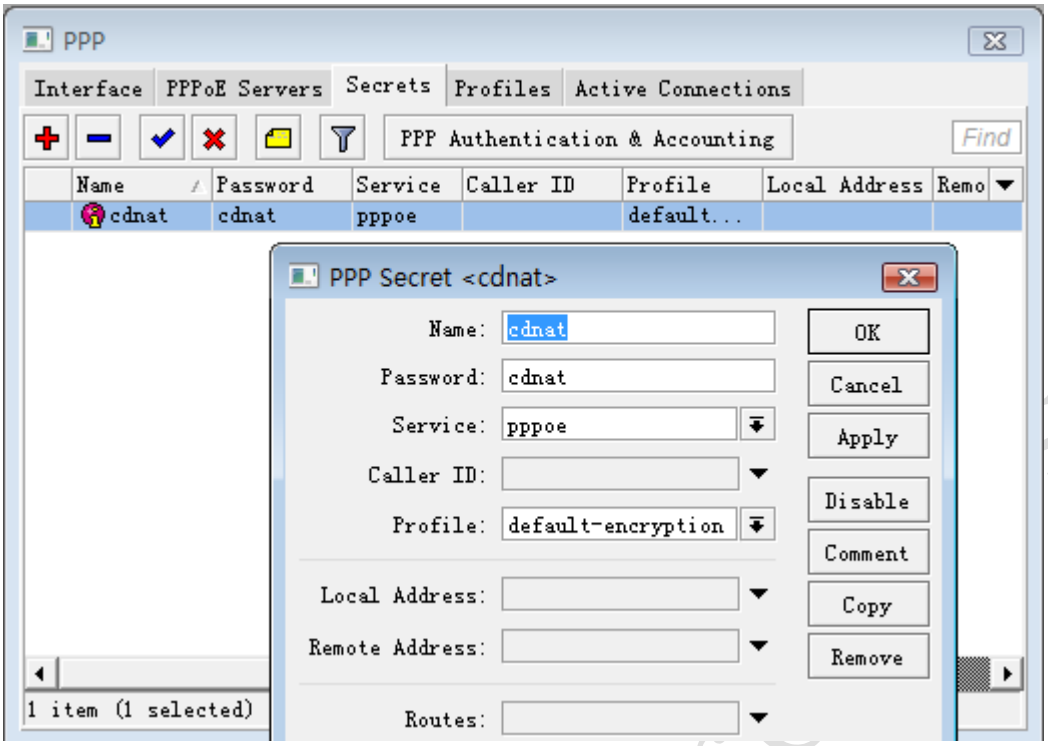
2、 Remote-address 为给客户端分配的 IP 地址，

3、 DNS-server 填写相应的 DNS 服务器
4、 User-Encryption 使用加密方式，在 windows 下预设是要求 PPPoE 拨号是加密的
5、 Idle-Timeout：空闲超时时间，即当客户端在一段时间内没有任何数据流量后，就断开连接。
6、 Rate-Limit：用户带宽，rx/tx（上行/下行）
7、 Only-one：即用户账号是否唯一

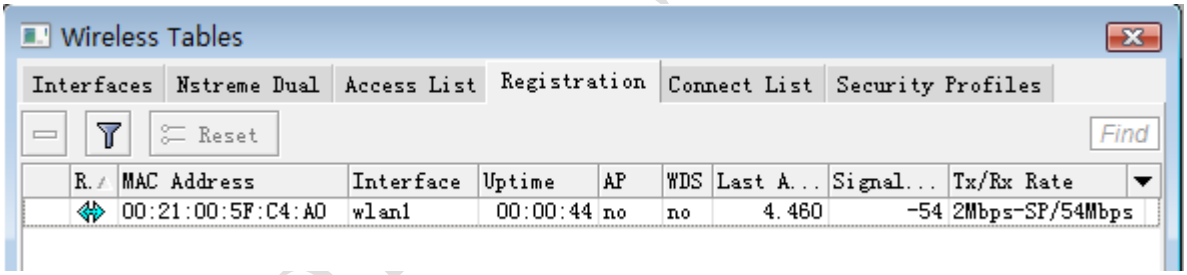
我们在配置 profile 时，设置 Local-address=192.168.10.1，Remote-address=pool1，DNS-Server=61.139.2.69，Idle-Timeout=20 分钟，Rate-limit=1m/2m，only-one=yes



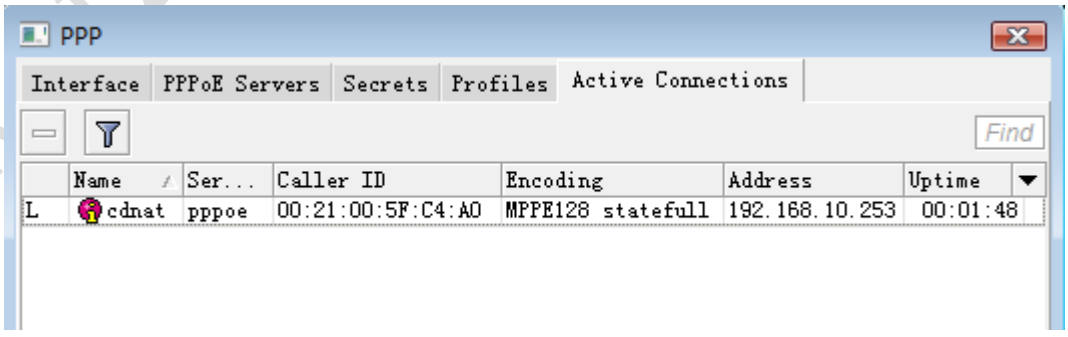
进入 secrets 配置账号，我们新添加一个账号为 cdnat，密码为 cdnat。选择服务类型为 pppoe，并选择 Profile 类型为 default-encryption，如下图：



步骤 4：这样 PPPoE 服务器建立完成，我们可以通过拨号连接测试一下，首先我们通过笔记本的无线网络卡连接到 MikroTik 的 AP 设备上，连接后，我们可以在 wireless 中的 Registration 中查看连接情况和信号强度：



在 windows 上配置完 PPPoE 拨号连接后，我们通过拨号连接到 PPPoE 服务，我们可以在 PPP 中的 Active 中看到连接成功的 PPPoE 账号：



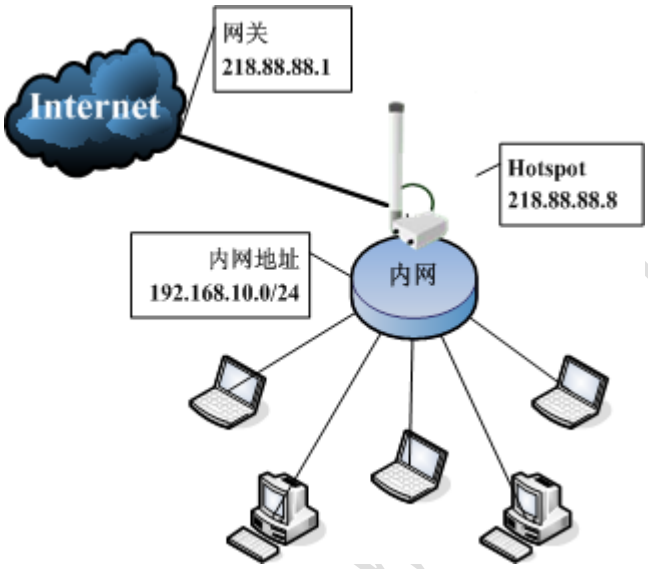
11.2 基于 Hotspot 的 WLAN 认证

Hotspot 的 WLAN 认证和 PPPoE 不同在于，PPPoE 是基于二层链路的认证，只能在二层网络传输如果有三层设备（路由器）就不无法穿透。而 Hotspot 是基于三层的 IP 网络验证，能被二层数据透传，Hotspot

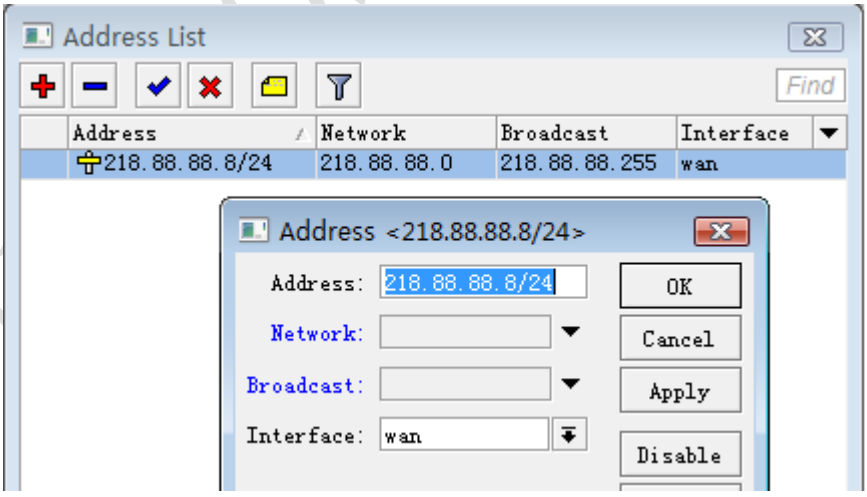
支持二层和三层的 IP 地址认证，所以能在二层和三层中传输，而 Hotspot 能基于 MAC 的 UPNP 认证，这种认证方式只能在二层中传输。我们来看看 Hotspot 的两种验证方式：

基于 IP 的验证：根据客户端配置或者获取正确的 IP 地址和网关，对客户的账号和密码进行认证上网。

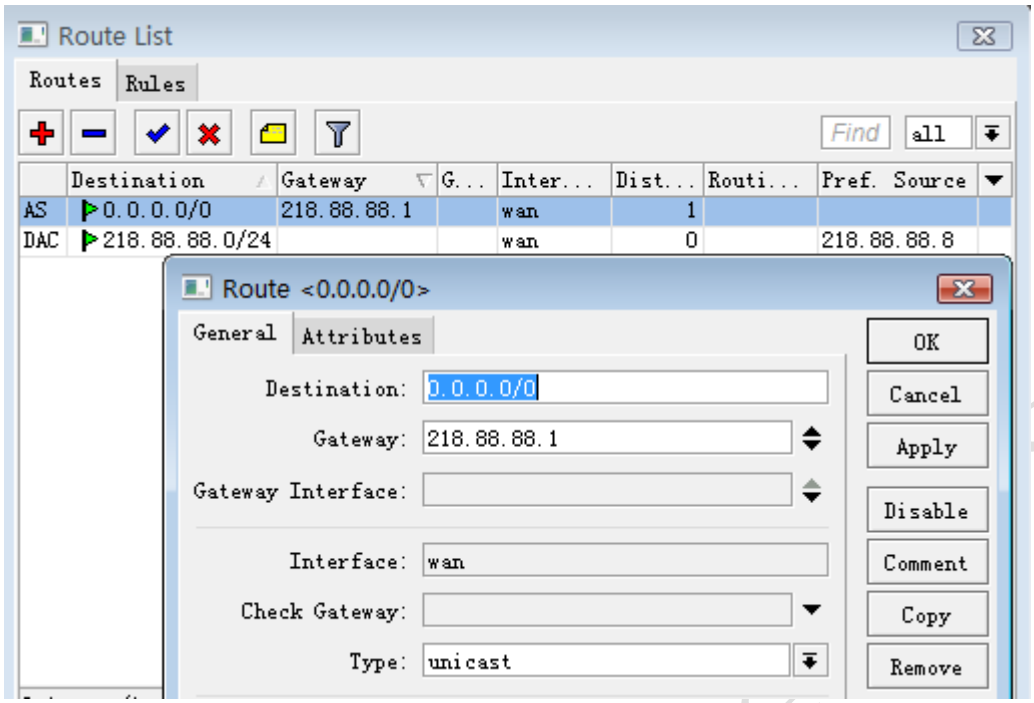
基于 MAC 的 UPNP 验证：UPNP 即插即用连接，Hotspot 服务器会在同一局域网内发送 ARP 广播，告诉局域网内的所有主机自己的网关设备，并为在线的主机分配一个虚拟的 IP 地址，这样客户主机在没有配置正确的 IP 地址情况下也能连接到 Hotspot 网关服务器，并认证上网。



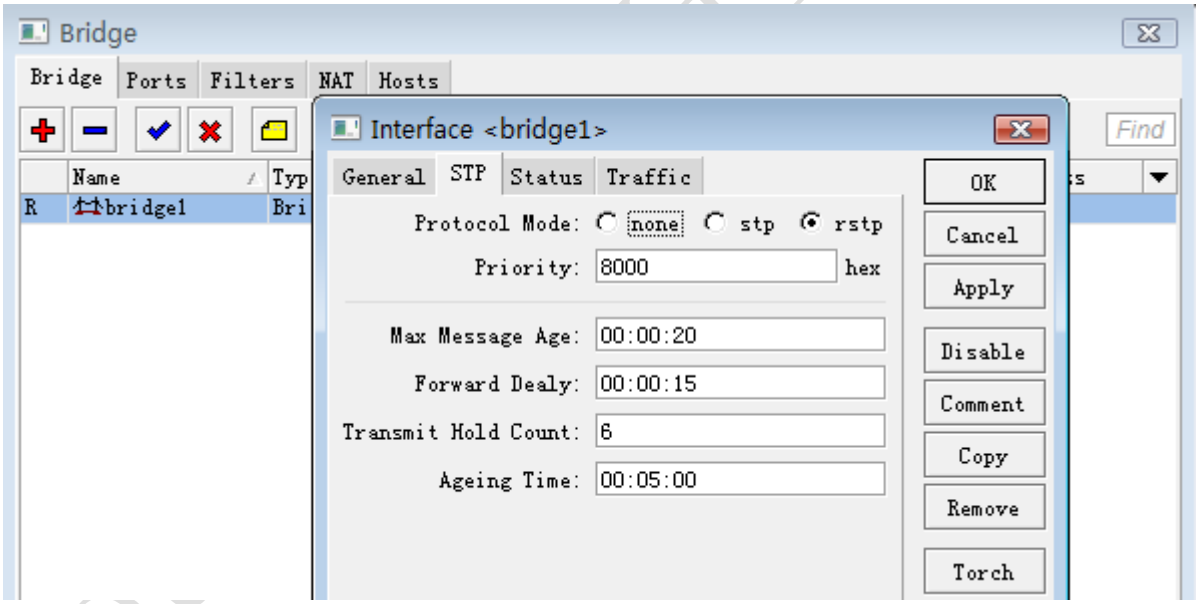
wan 口的外网连接 IP 地址是 218.88.88.8/24，内网 IP 段为 192.168.10.0/24，我们先添加 wan 口的外网 IP 地址：



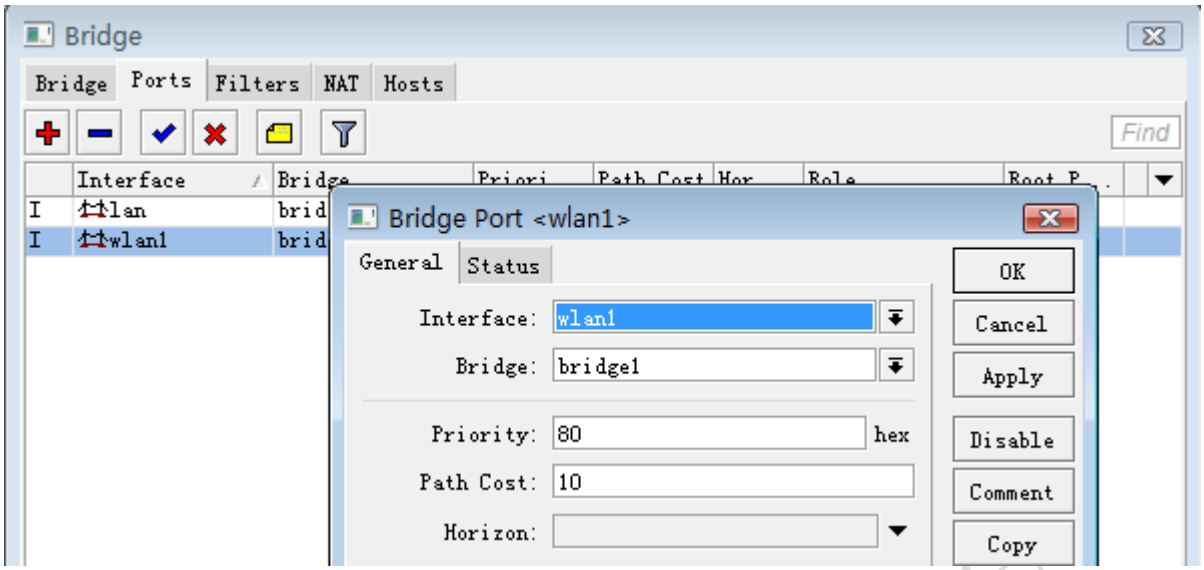
接着我们进入 ip route 配置路由，网关为 218.88.88.1



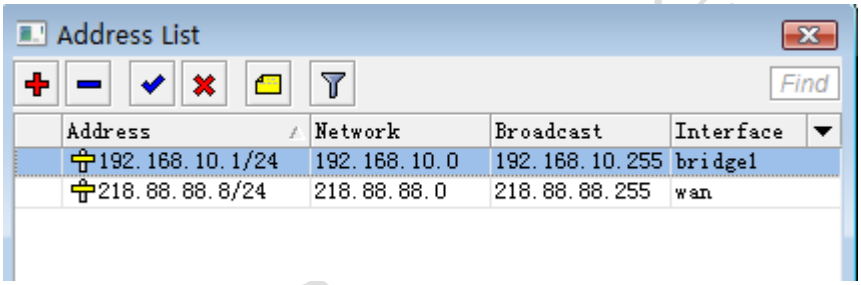
进入 bridge 在我们添加桥接，并设置 rstp 的参数，将 lan 口和 wlan1 设置到 bridge1 中：



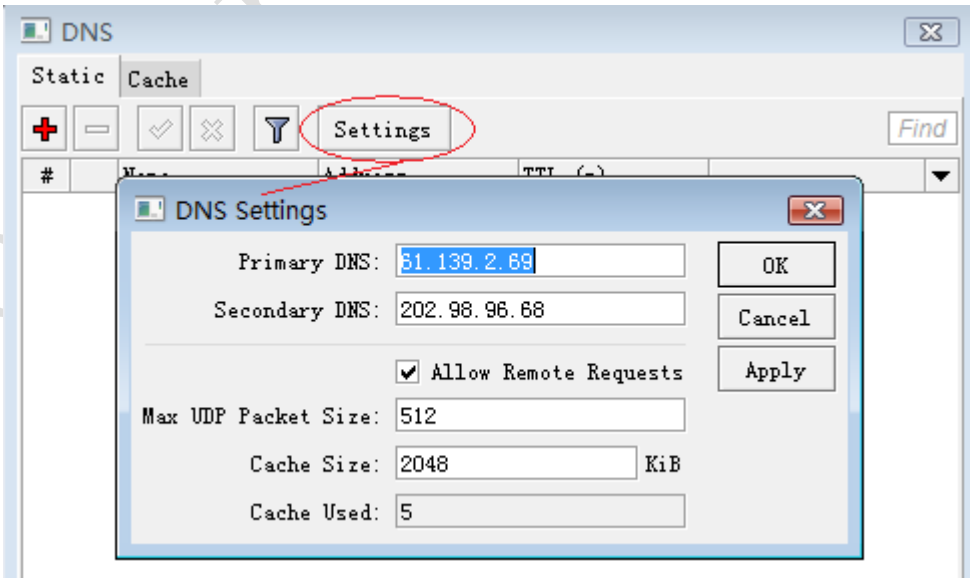
将 lan 和 wlan1 添加如 bridge1 中：



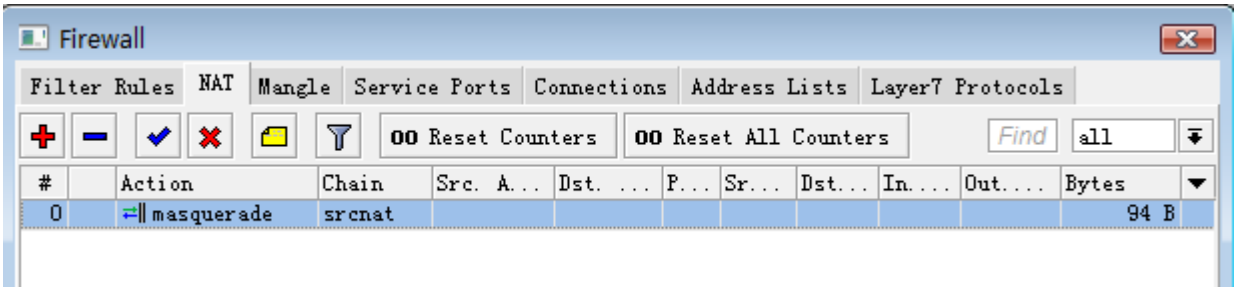
我们分配内网 IP 地址 192.168.10.1/24 设置到 bridge1 上



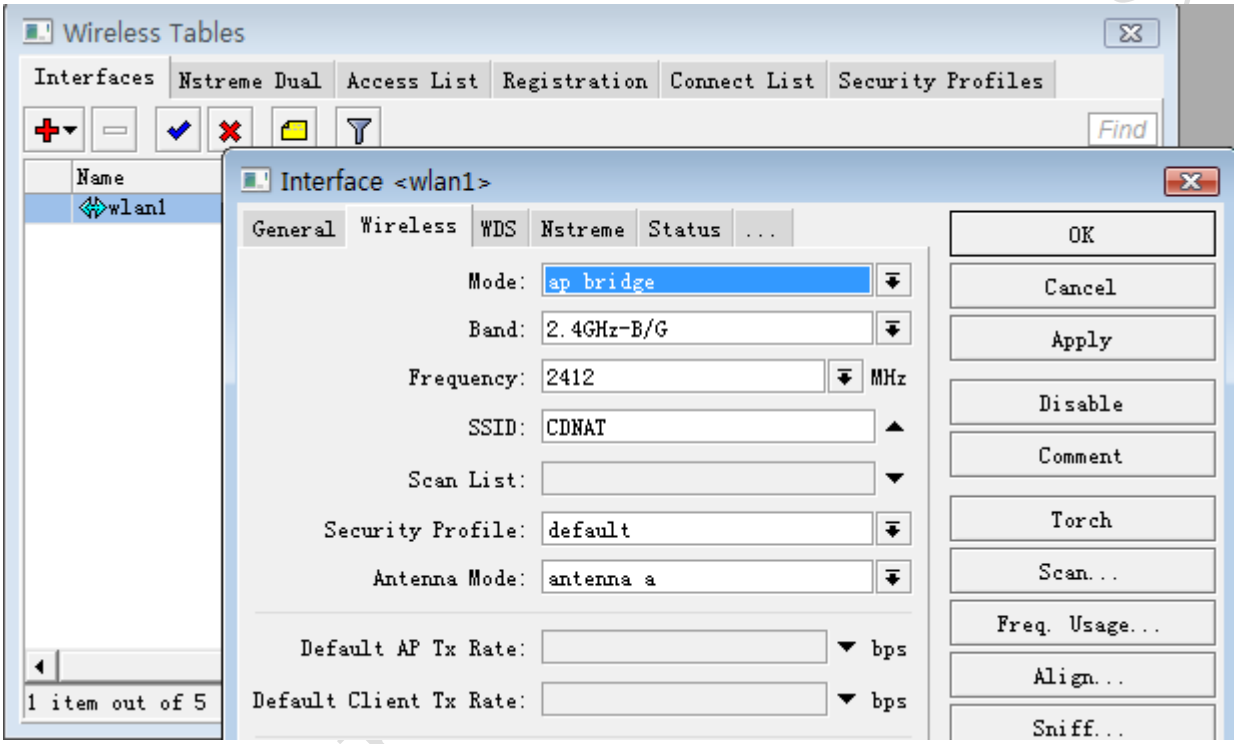
进入 ip dns 配置 DNS 服务，因为 Hotspot 要求通过本机的 DNS 解析后才能跳转到认证页面，这里我们配置 DNS 服务分别为 61.139.2.69 和 202.98.96.98，将 allow-remote-request 参数选择上，该功能是启用 DNS 缓存：



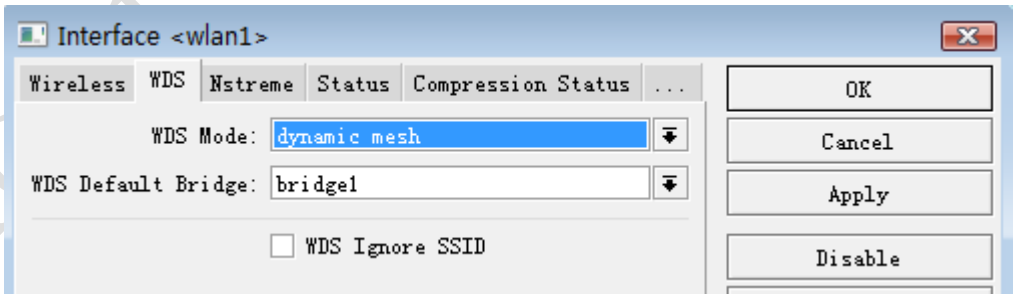
之后我们进入 ip firewall nat 配置 src-nat 的伪装策略 action=masquerade，用于隐藏内部的私有 IP 地址连接上网，如下图：



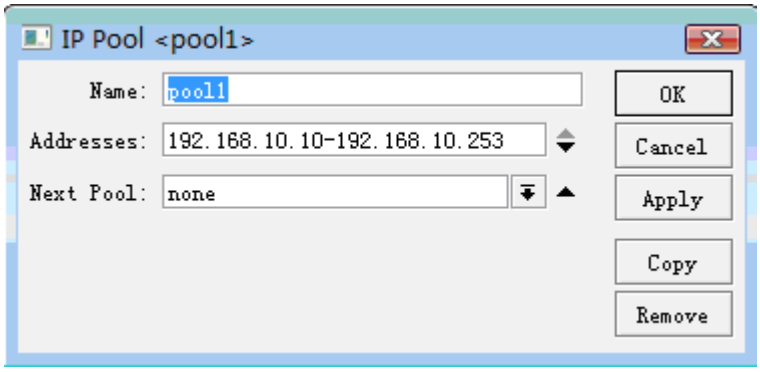
接下来配置 wlan1 的无线模块，我们这里以 2.4G-bg 为主，配置 mode=ap-bridge、Band=2.4G-B/G、Frequency=2412、SSID=CDNAT、WDS-Mode=dynamic-mesh、wds-default-bridge=bridge1，配置如下图：



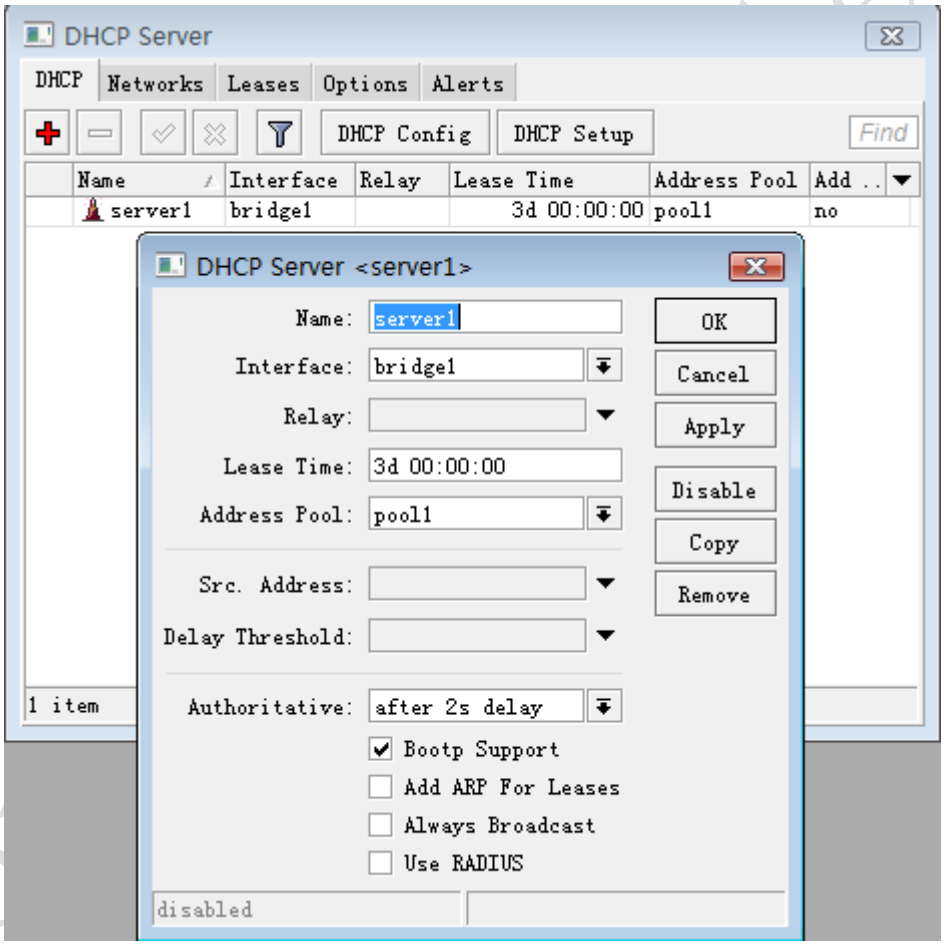
在 wlan1 配置 WDS 选项，设置 WDS-Mode=Dynamic-Mesh，比将其添加到 bridge1 中：



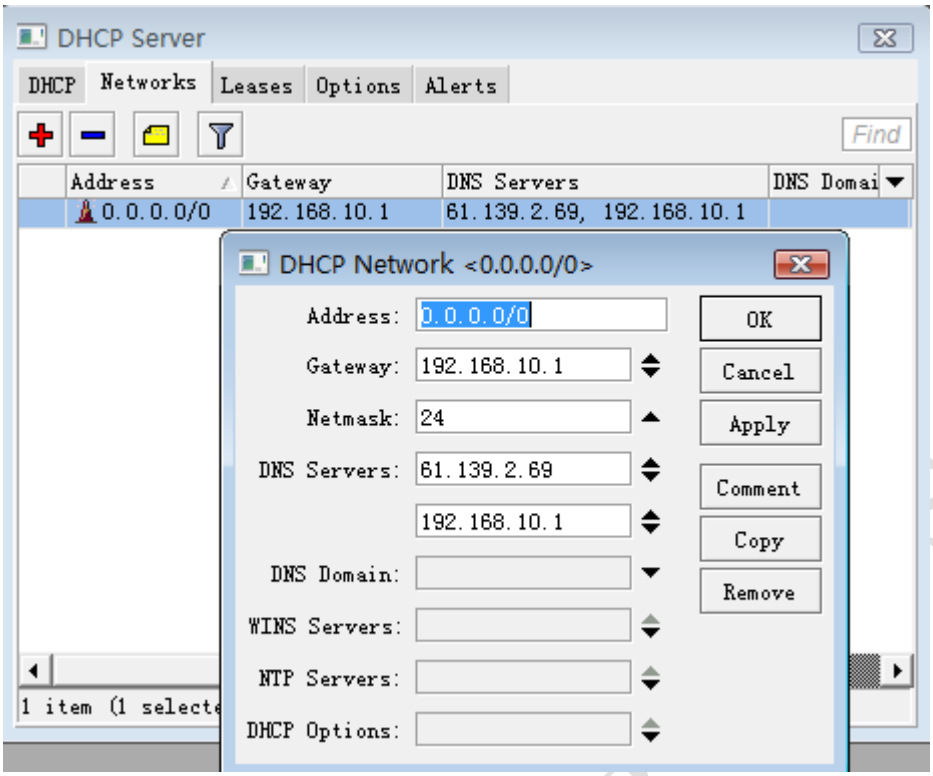
配置完基本参数后，首先我们添加地址池用于 Hotspot 的 IP 地址分配：



我们配置好地址池后，我们需要建立 DHCP 服务器，当用户没有设置 IP 地址的情况下，可以通过 Hotspot 服务器获取到动态的 IP 地址，我们进入 ip dhcp-server，选择 Interface=bridge1，给用户分配的地址池 Address-pool=pool1



之后进入 DHCP-Server 的 Networks 配置分配给客户的网关和 DNS 服务，分配网关地址是 192.168.10.1，DNS 服务设置为 61.139.2.69 和 192.168.10.1（在 ip dns 中启用了 dns 缓存功能）：

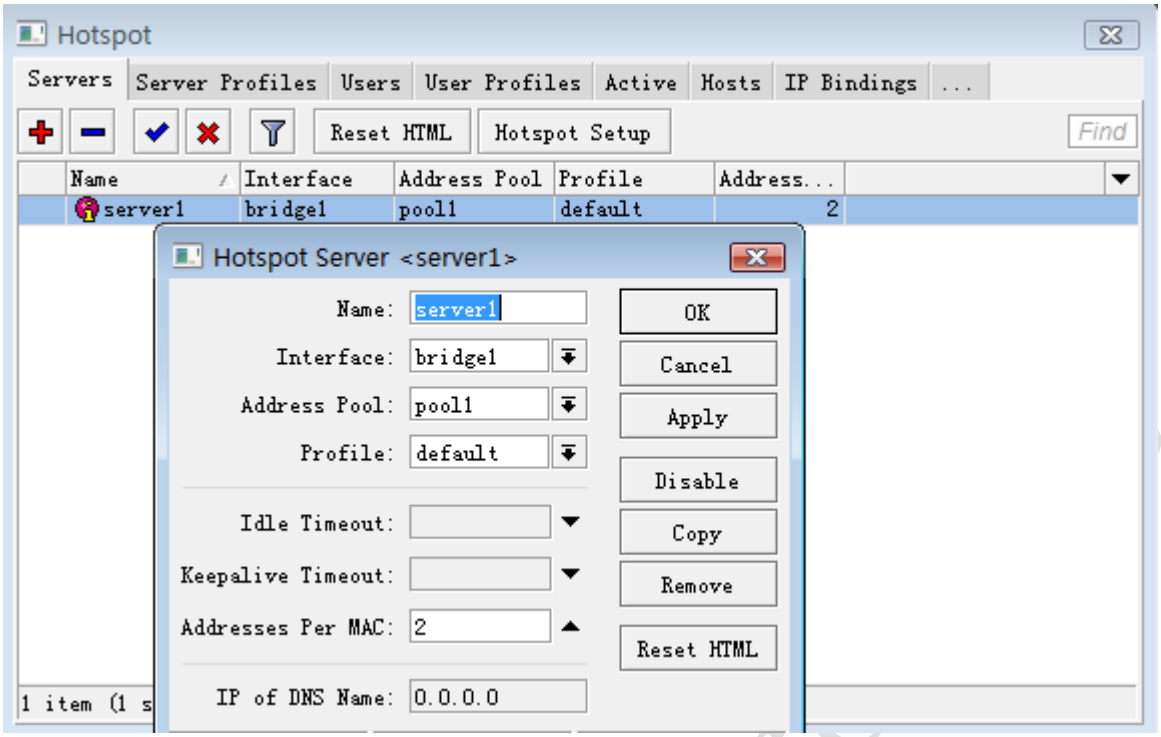


配置完 DHCP 服务器后，我们进入 ip hotspot 目录下配置 Hotspot 服务，配置 Hotspot 服务器步骤如下：

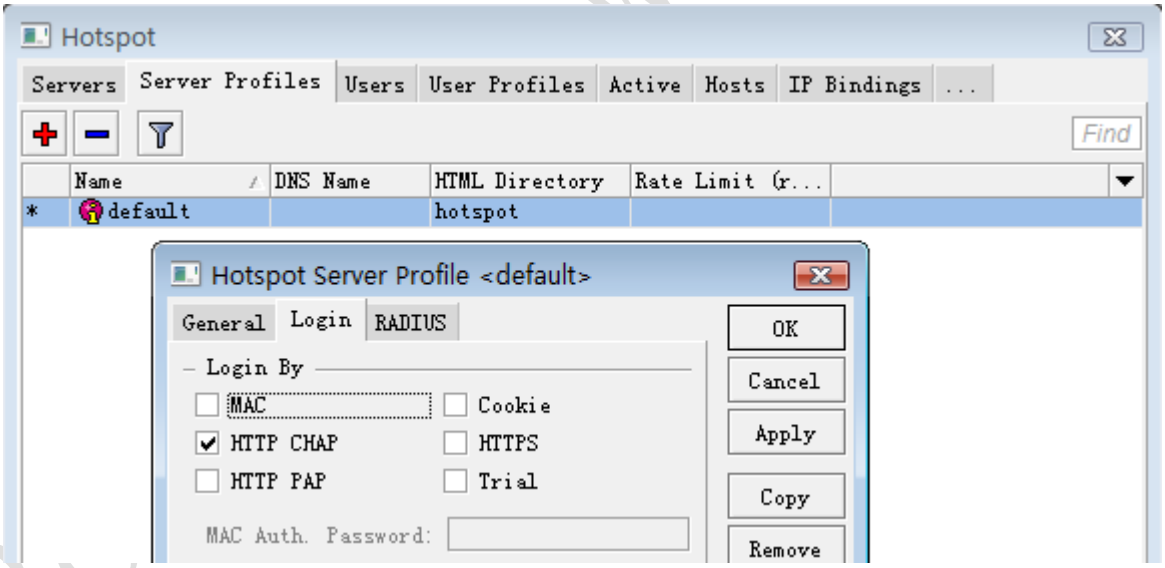
1、 进入 Server 添加 Hotspot 服务器，在 Server-Profile 中配置服务器组规则；
2、 进入 User-Profile 目录下配置用户组规则，包括共享用户数、空闲时间和带宽；
3、 在 User 目录下添加用户账号和密码；
4、 通过 IE 浏览器登陆 Hotspot 认证页面，并认证登陆。

步骤 1：配置 Hotspot 服务器

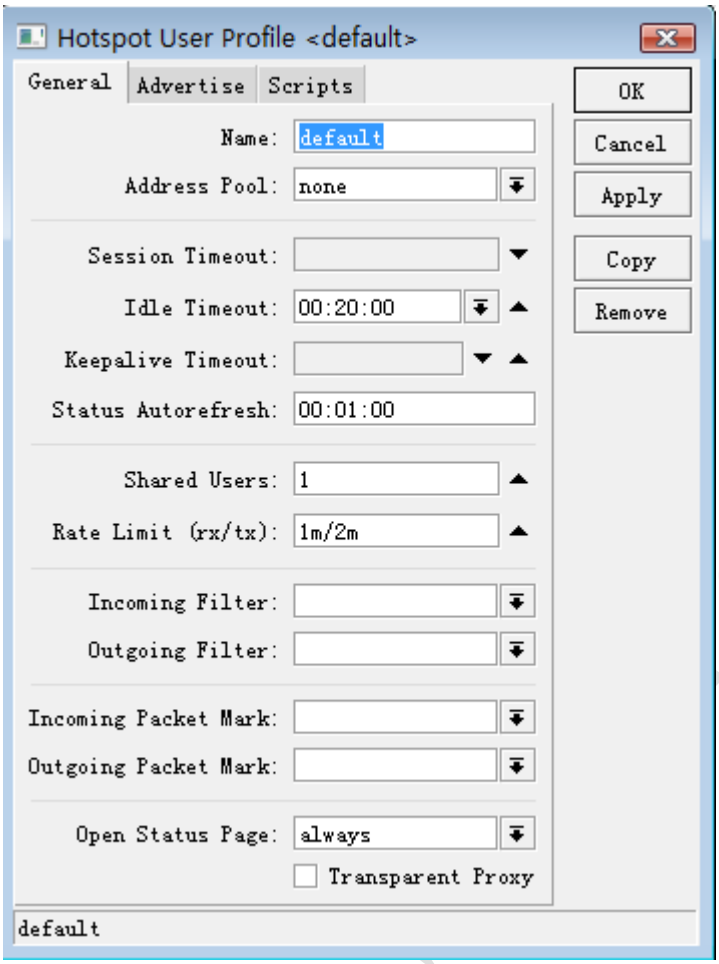
我们进入 ip hotspot server 目录下，添加服务器 Server1，将认证接口定义到 bridge1 上，并配置 Pool 地址池用于 UPNP 用户的 IP 地址分配，设置 Address-per-mac=2（允许 1 个 IP 地址邦定 MAC 地址的数量，降低 IP 地址仿真多个 MAC 攻击的可能性）。



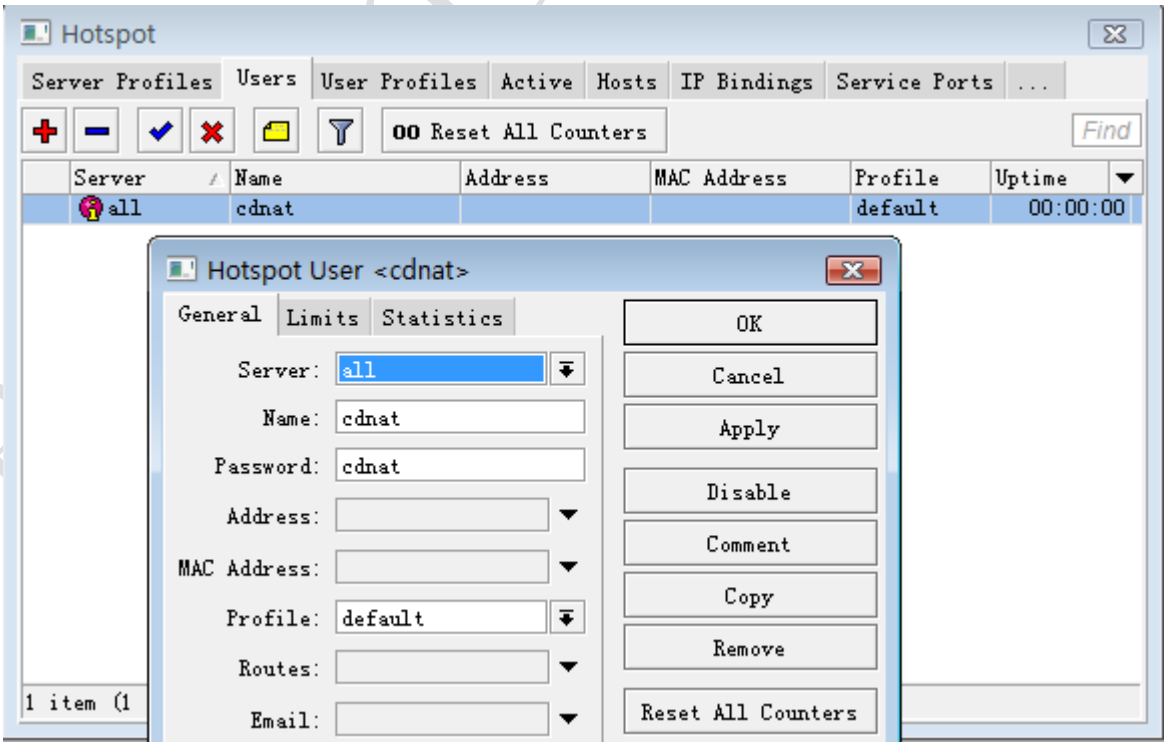
步骤 2：进入 server profiles 配置，服务器组规则，我们选用默认的 default 规则，我们将 Login 方式修改为仅有 HTTP-CHAP 协议：



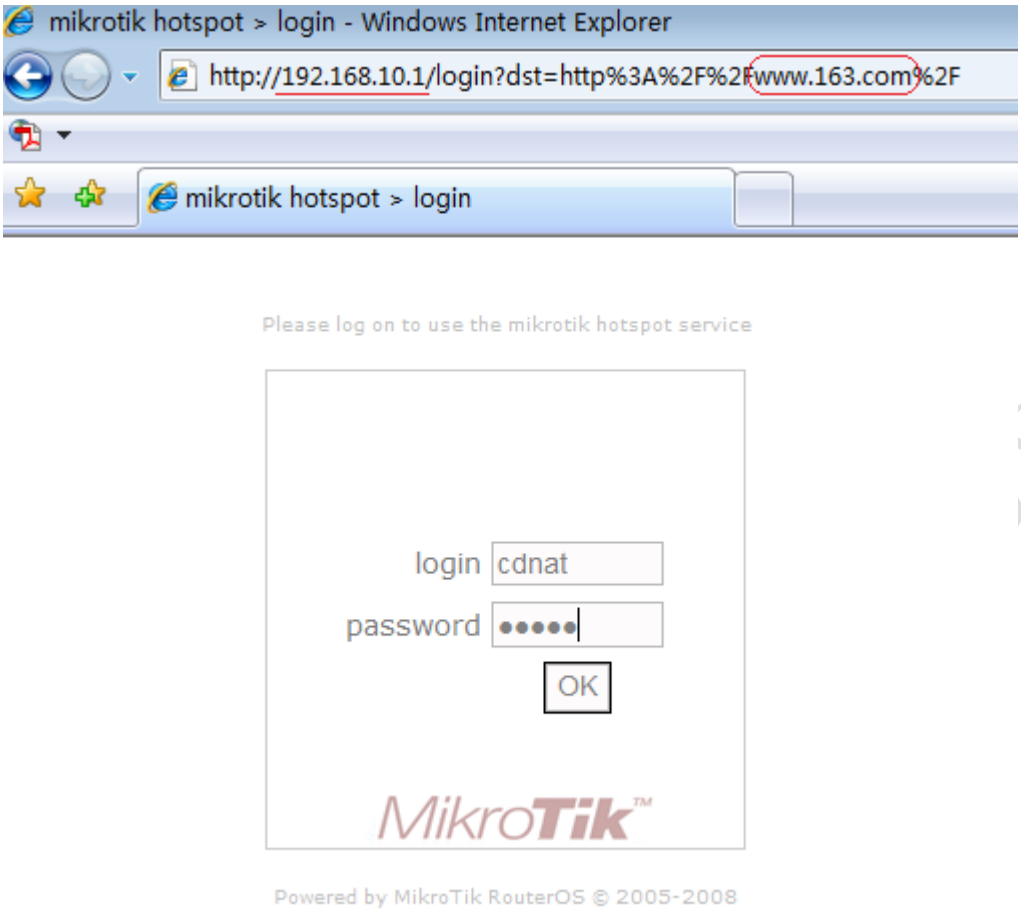
然后进入 User-profile 目录下，配置用户组规则，设置 Idle-Timeout=00:20:00（20 分钟），Shared-User=1（账号只被 1 个用户使用），Rate-limit=1m/2m（上行 1Mbps，下行 2Mbps），配置如下：



步骤 3：进入 Users 目录下，添加用户账号，账号 cdnat，密码 cdnat，用户分组的 Profile 选择 default：

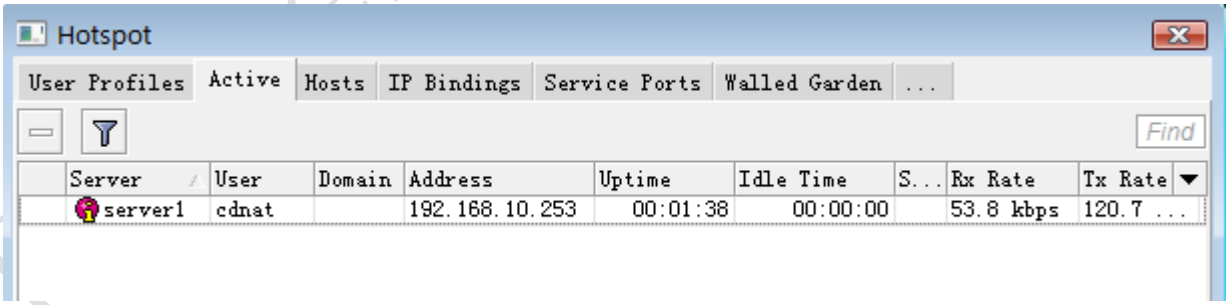


步骤 4：Hotspot 配置基本完成，现在我们可以通过 IE 浏览器登陆到 Hotspot 的认证页面，我们在 IE 浏览器中输入 www.163.com 的网址，当 Hotspot 服务器正确解析道 163 网址的地址，会自动跳转到认证页面，输入我们的用户名和密码认证登陆：



上面的地址栏中，我们可以看到，在输入 www.163.com 被 hotspot 强行跳转到了 192.168.10.1 的认证网关上。该认证页面上可以修改，具体操作请参阅**《RouterOS 中文网络教程》**。

登陆后，我们可以在 active 中看到，当前登陆的使用者账号状态：



注：在即插即用情况下登陆，是在用户没有正确配置当前网络下的静态 IP 地址和网关，通过 Hotspot 服务器的 Pool（地址池）分配一个虚拟的 IP 地址给客户主机，但要求客户注意必须配置静态的 IP 和网关，如果只配置了静态的 IP 地址，则即插即用不会生效。

第十二章 其他无线应用

12.1 无线管理 VLAN 与业务 VLAN

RouterOS 的桥可以透传二层的 VLAN 数据，通过在接入端设备配置 VLAN 接口，分别连接到各个接入点的 VLAN 交换机，这样让每个区域的用户的数据能够通过 VLAN，直接到达我们的接入设备，而不会受到其他二层数据广播的影响，同时也减小了二层的广播风暴，隔离每个区域的数据。在一些运营商会将管理 VLAN 和业务 VLAN 划分开，这样有助于对设备的管理。

关于 RouterOS VLAN 的介绍可以参考 RouterOS 网络教程中的 VLAN 一章，由于大多 RouterBOARD 都基本 Switch 交换功能，但 WLAN 网卡芯片是独立于 Switch 交换逻辑芯片，所以只能通过 bridge 来实现 VLAN 透传

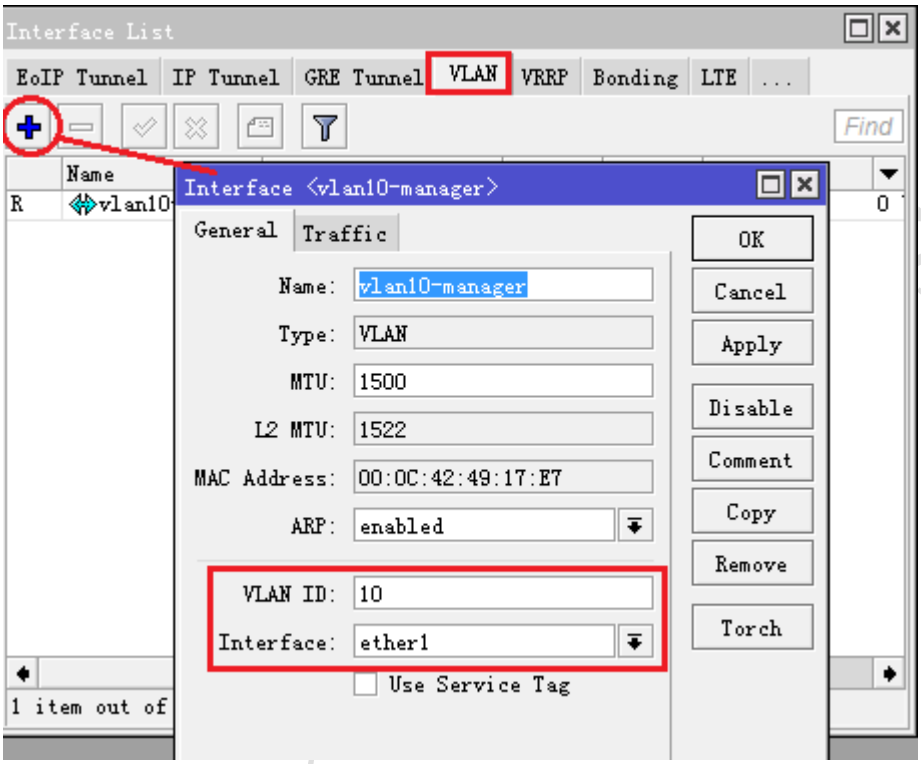
WLAN 网络可以建立 ap-bridge to station-wds 模式的网桥连接，基于透明桥接的方式，我们可以把网络数据透传到远程的节点，ap-bridge 可以建立点对点或者点对多点的模式，根据网络方案的需要可以灵活调整网络结构。

VLAN 配置实例

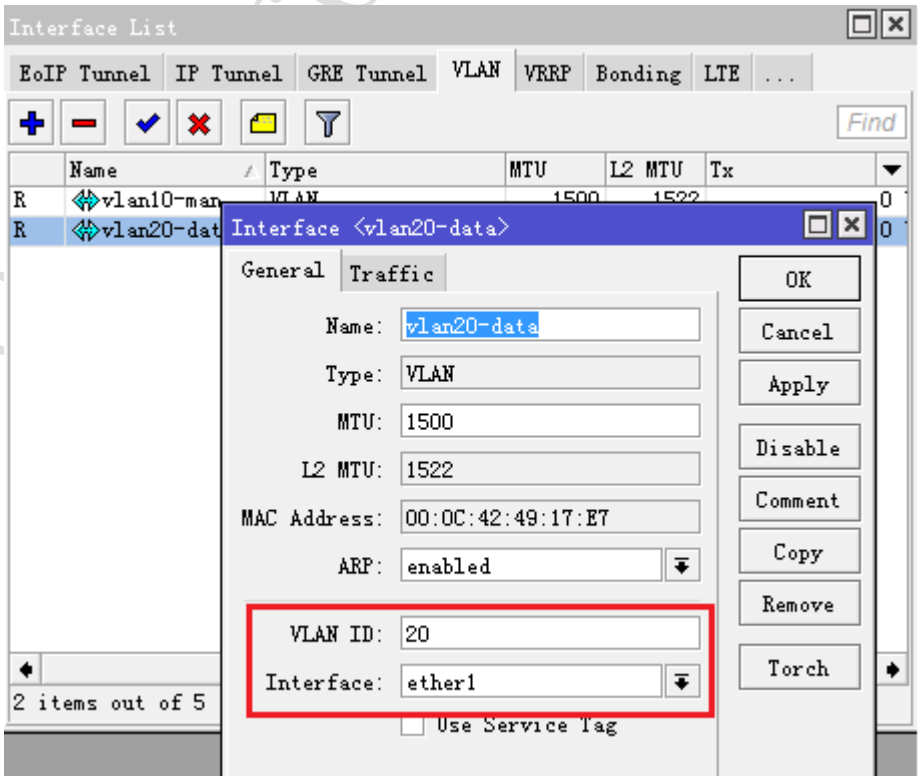
这种方式常见于运营商的 WLAN 网络，即组网时，就将管理网络和业务网络划分开，避免之间相互影响。假设我们有以下网络，vlan 10 为管理 VLAN 负责设备管理，vlan 20 为业务 VLAN 负责用户 PPPoE 或热点认证上网。而上联的三层交换机通过 trunk 模式将两个 vlan 透传给 RouterBOARD 设备。如下图：

RouterBOARD 通过 ether1 的以太网接口连接到三层交换机，这样 ether1 接口需要配置两个 vlan，vlan10 和 vlan20

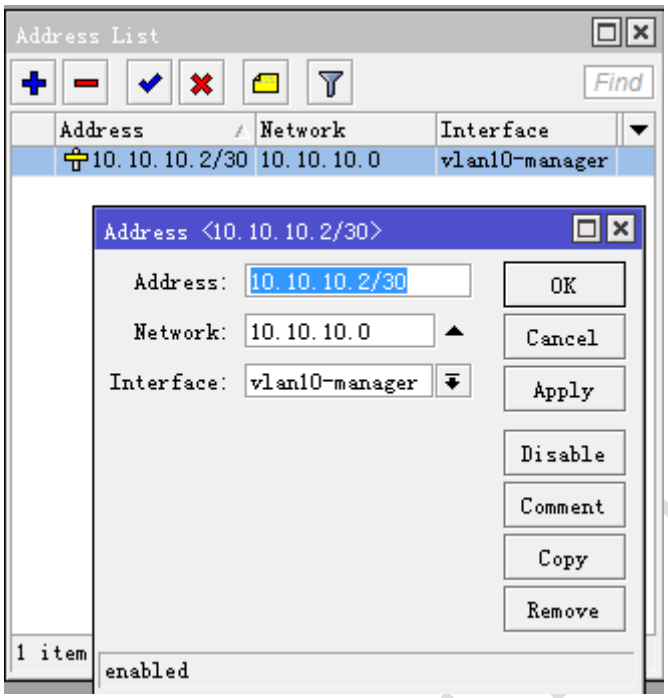
首先进入 interface 菜单，打开 VLAN 设置，添加管理 vlan，取名为 vlan10-manager，设置 VLAN ID 为 10，绑定到 ether1 界面



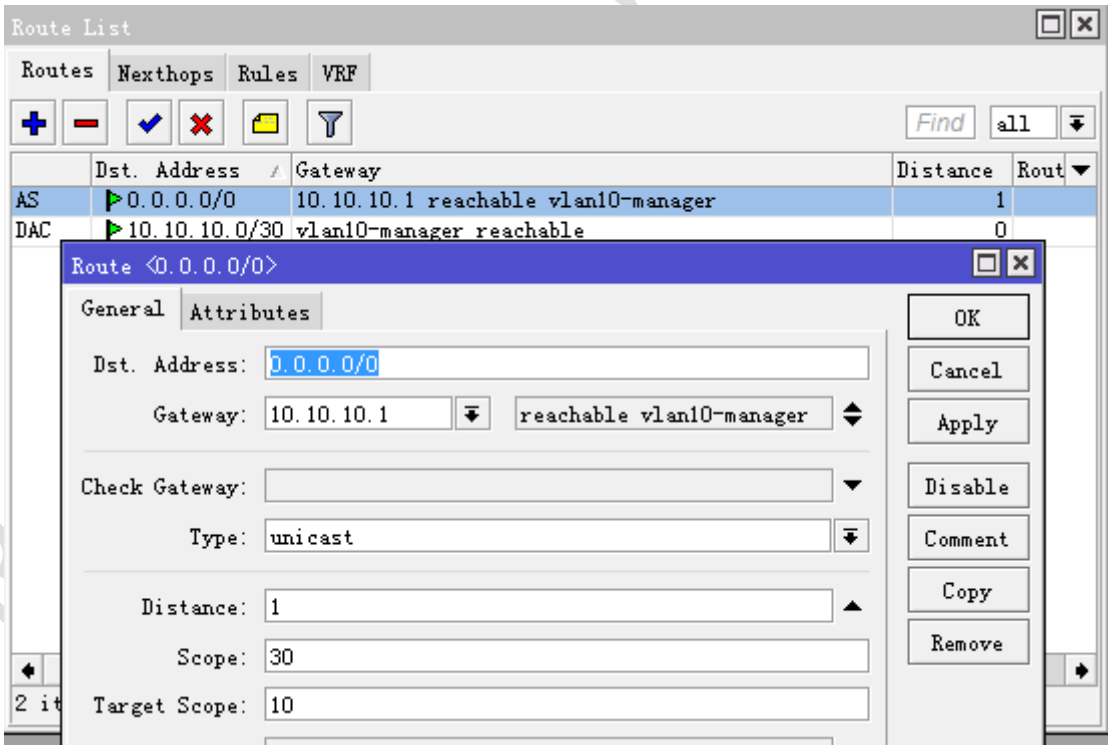
用同样的方法添加业务 VLAN，取名 vlan20-data，设置 VLAN ID 为 20，同样绑定到 ether1



这样的配置类似于其他路由器配置 VLAN 子接口，下面配置管理 VLAN 的 IP 地址，进入 ip address 添加 IP 地址，并配置到 vlan10-manager 上

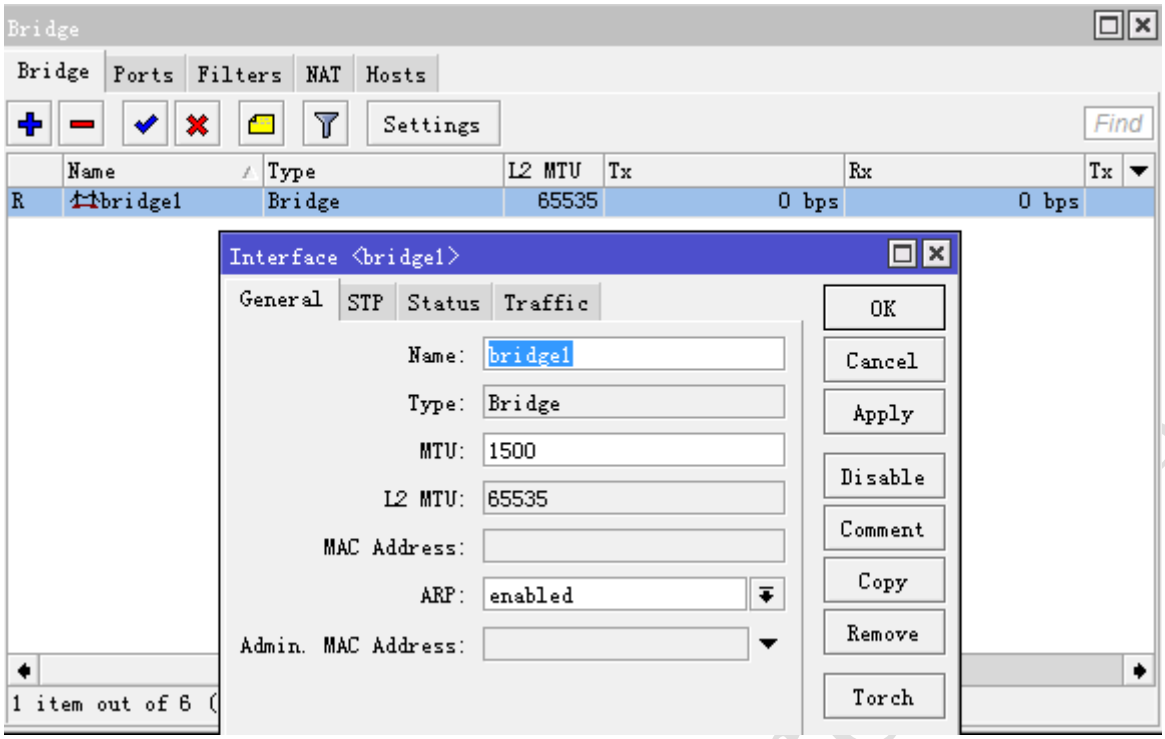


添加默认网关 10.10.10.1

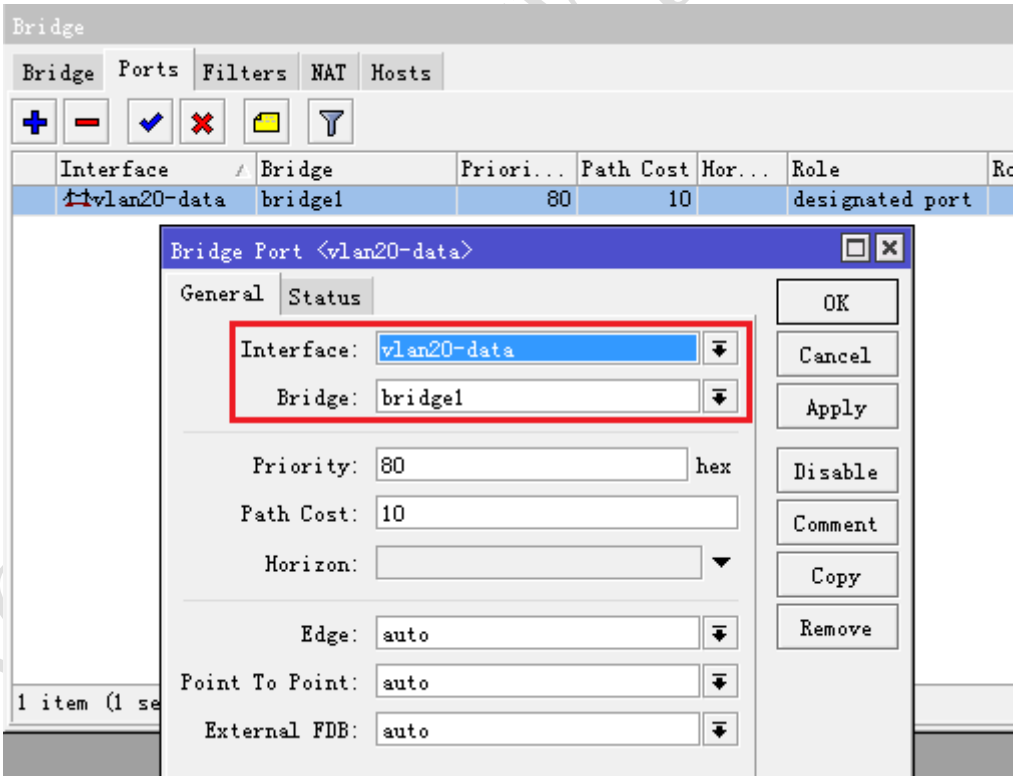


这样管理 VLAN 配置完成，接下来配置业务接口的无线网络透传，

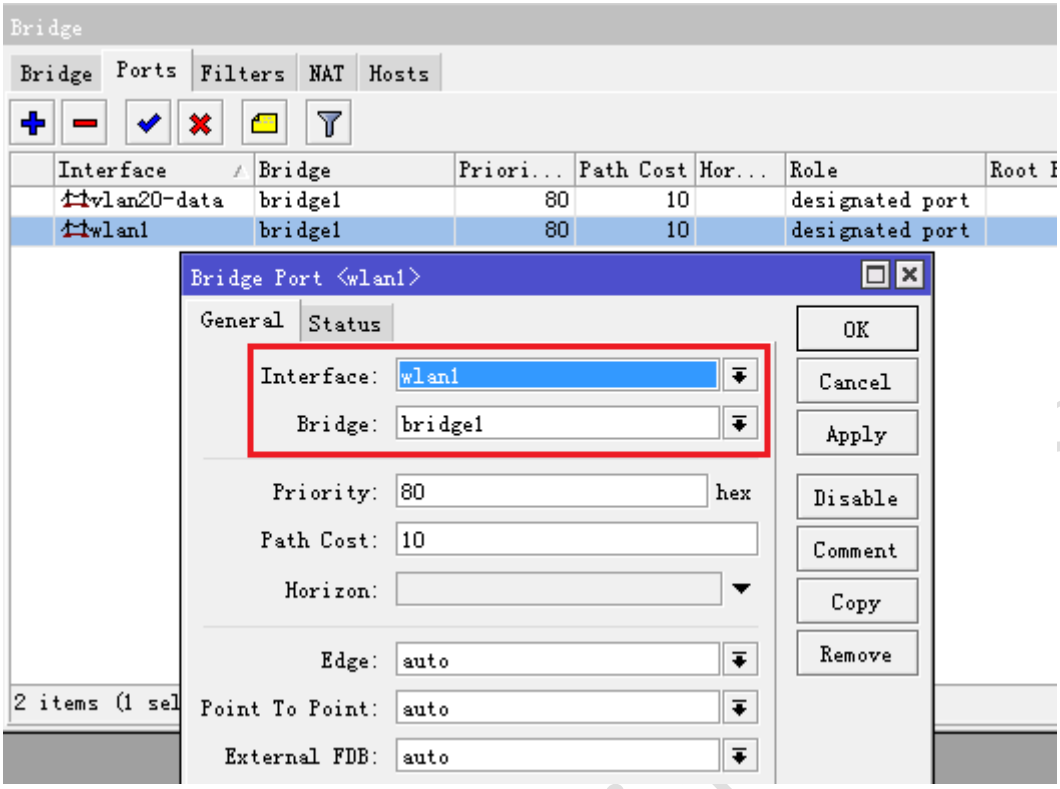
业务 VLAN 实质就是要将 vlan 20 和 wlan1 网卡做透明桥接，这里我们需要添加一个 bridge，进入 bridge 菜单，添加一个 bridge1



进入 ports 将 vlan20-data 和 wlan1 网卡做桥接，下面是添加 vlan20-data 到 bridge1



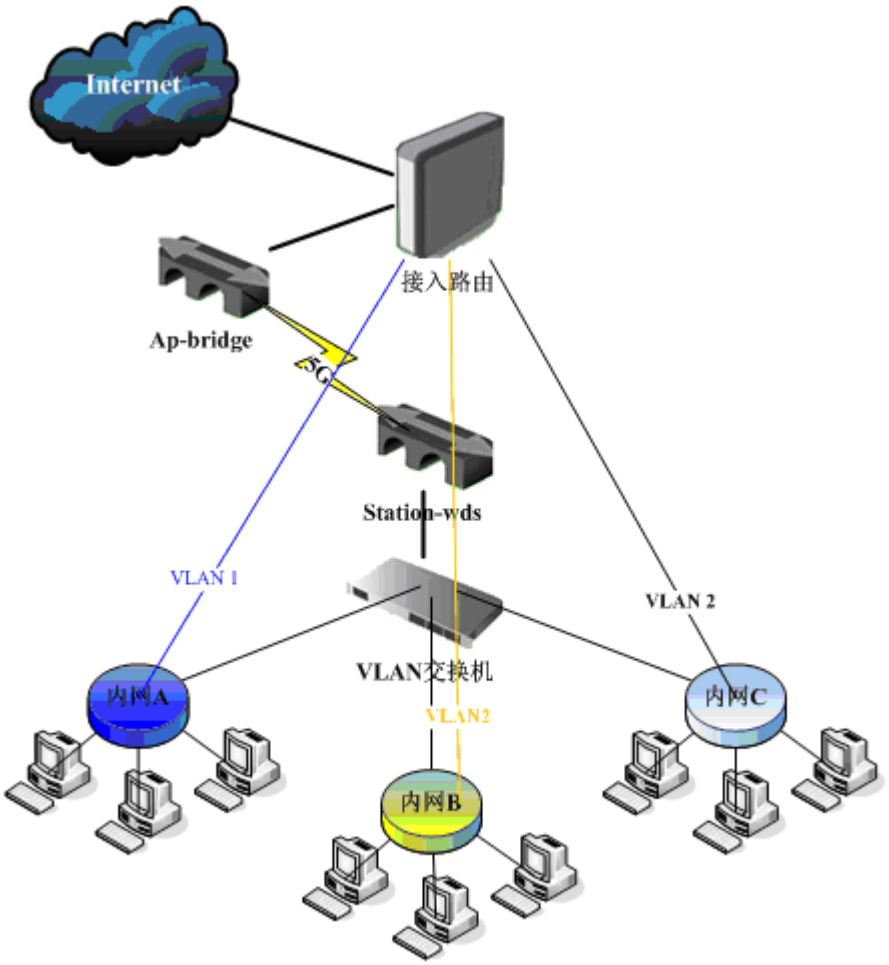
添加 wlan1 到 bridge1



这样配置，就完成了 vlan 10 为管理 VLAN，vlan 20 为业务 VLAN 的配置

VLAN 方案一

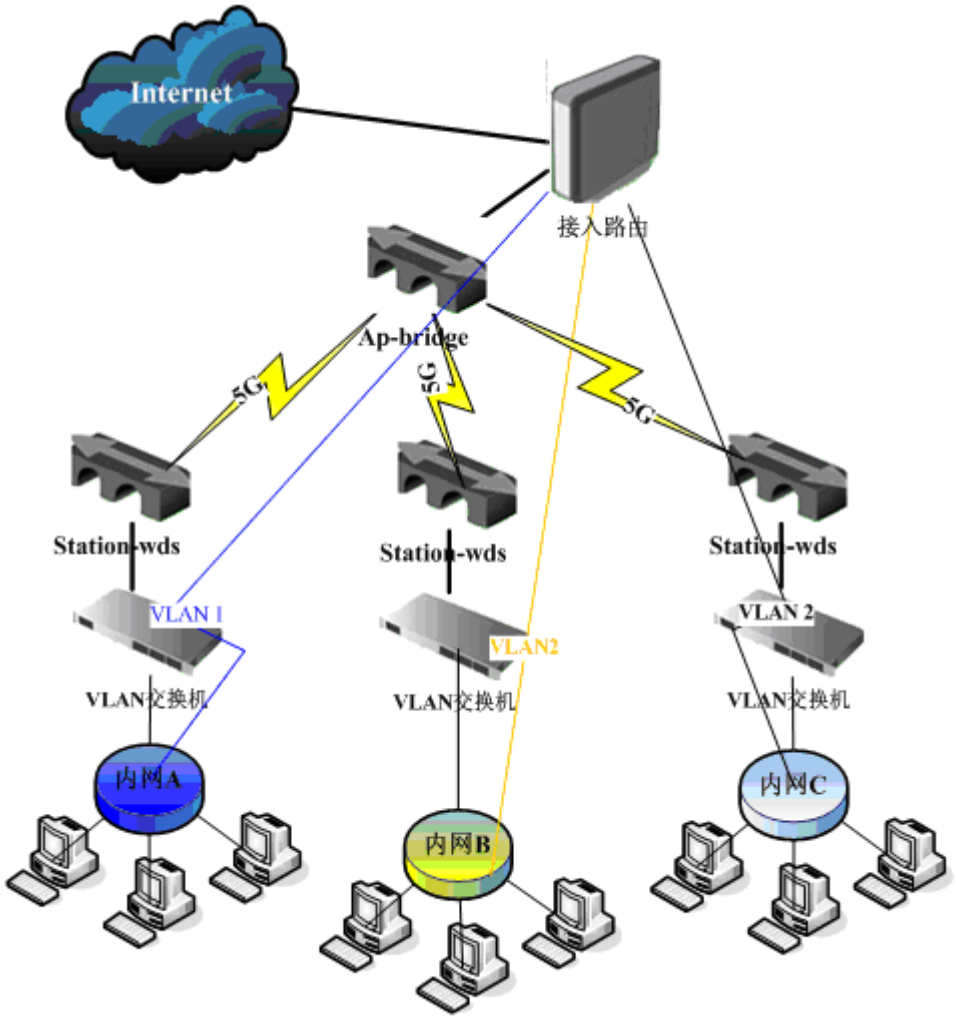
假设我们的网络通过 Internet 连接到接入路由，然后分别通过 2 个点对点的桥接设备连接到远程的 3 个网络，在点对点和 3 个网络中间有一个 VLAN 交换机连接，我们通过接入路由配置 VLAN，分别连接到远程的 3 个不同 VLAN 网络，如下图：



通过这样的网络结构，我们可以为每一个 VLAN 划分一个子网，便于网络的管理。由于建立的是 VLAN 隧道，3 个网络可以直接（VLAN 隧道建立了虚拟链路，通过 ID 区分每个链路）和接入路由通讯，这样的 VLAN 划分我们不仅可以划分不同的子网，也可以建立 3 个独立的 PPPoE 服务或者 Hotspot 服务。

VLAN 方案二

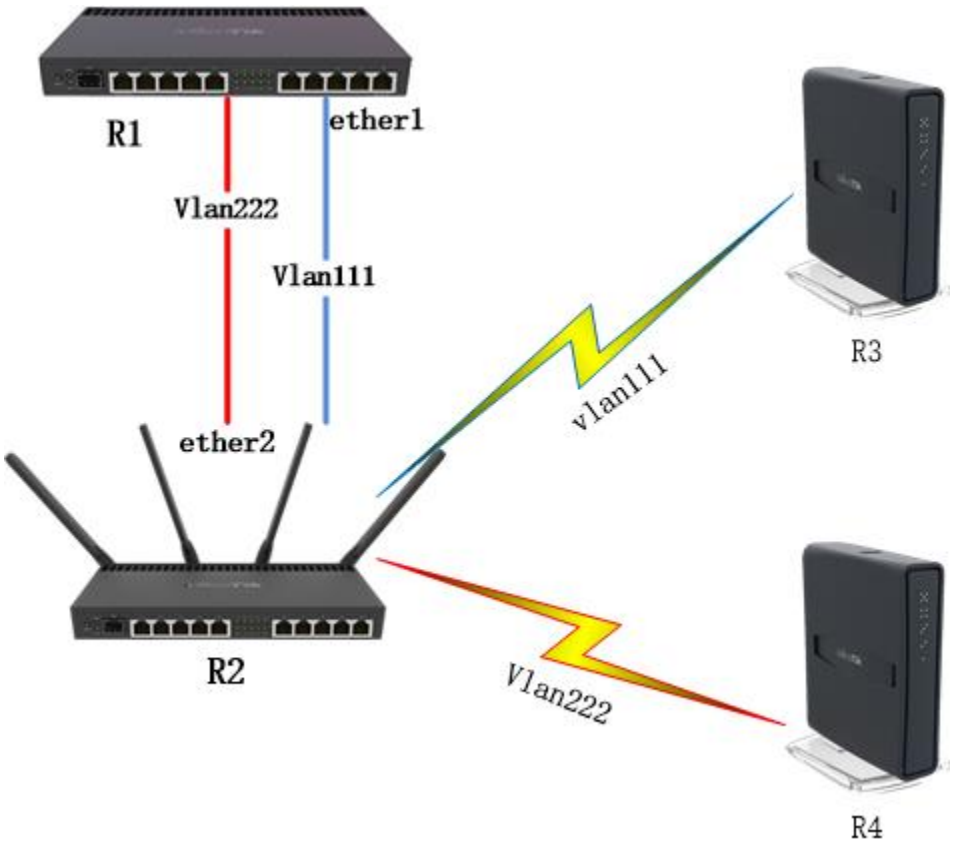
假设我们的无线网络不是点对点的方式，而是点对多点的桥连接，构建与其点对点模式基本相同，同样是通过接入路由建立多 VLAN，分别连接到各网络内的 VLAN 交换机：



以上是对 VLAN 在 WLAN 网络中的一些应用参考。

12.2 基于 vlan-filtering 和 wireless vlan-mode (推荐)

RouterOS 支持基于无线链路的 VLAN 透传。 例如，当我们在一个办公网络中，需要使用一台 RouterOS 设备分隔办公 WiFi 和来访 WiFi 时，可以使用 VLAN 方式做二层隔离，使用 Wlan1 做来访 WiFi，在 Wlan1 上创建虚拟 AP（VirtualAP Wlan2）作为办公 WiFi。 在之前的版本中有相应的解决方案，但 v6.41 后在 bridge 增加了 VLAN Filtering 功能，通过 vlan filtering 能更好实现无线中 vlan 透传，也是官方推荐方法。



从 RouterOS v6.41 开始，Birdge 增加了 VLAN Filtering 功能支持二层 VLAN 转发和 VLAN 标记处理，因此在 v6.41 后建议使用以下配置方案。

R1 路由器:

在路由器 R1 创建指定的 VLAN 接口在对应的 ether1 有线接口上，即 trunk 模式的配置，并配置 IP 地址到 VLAN 接口，启用三层 VLAN。

```
[admin@R1] >
/interface vlan
add interface=ether1 name=vlan111 vlan-id=111
add interface=ether1 name=vlan222 vlan-id=222

/ip address
add address=192.168.1.1/24 interface=vlan111
add address=192.168.2.1/24 interface=vlan222
```

R2 路由器:

我们需要创建 wlan2 虚拟机 AP 接口，并配置 Wlan1 和 Wlan2 的加密规则

```
[admin@R2] >
/interface wireless security-profiles
add authentication-types=wpa-psk,wpa2-psk eap-methods=""
management-protection=allowed mode=dynamic-keys name=\
```

```
vlan111 supplicant-identity="" wpa-pre-shared-key=yusvlan111
wpa2-pre-shared-key=yusvlan111

add authentication-types=wpa-psk,wpa2-psk eap-methods=""
management-protection=allowed mode=dynamic-keys name=\
    vlan222 supplicant-identity="" wpa-pre-shared-key=yusvlan222
wpa2-pre-shared-key=yusvlan222
```

在路由器 R2 的 Wlan1 创建一个虚拟 AP 接口取名 Wlan2，并设置 Wlan1 和 Wlan2 不同的无线加密规则，设置 vlan-mode=use-tag

```
/interface wireless
set [ find default-name=wlan1 ] disabled=no mode=ap-bridge
security-profile=vlan111 ssid=vlan111 vlan-id=111 vlan-mode=use-tag
add disabled=no master-interface=wlan1 name=wlan2 security-profile=vlan222
ssid=vlan222 vlan-id=222 vlan-mode=use-tag
```

路由器 R2 创建桥接，设置 vlan-filtering=yes，添加相应的桥接端口，并在 bridge vlan 中创建对应端口的 vlan id 标签

```
[admin@R2] >
/interface bridge
add fast-forward=no name=bridge1 vlan-filtering=yes

/interface bridge port
add bridge=bridge1 interface=ether2
add bridge=bridge1 interface=wlan1
add bridge=bridge1 interface=wlan2
/interface bridge vlan
add bridge=bridge1 tagged=ether2,wlan1 vlan-ids=111
add bridge=bridge1 tagged=ether2,wlan2 vlan-ids=222
```

提示： 一些 RB 和 CRS 设备集成了交换芯片，能实现以太网二层数据交由交换芯片做线速转发，启用桥接 VLAN filtering 功能后无法实现交换芯片线速转发（除 CRS3xx 系列设备），则会交叉 CPU 处理。由于无线接口没有归属于交换芯片，因此也无法实现硬件转发，只能交给 CPU 处理

R3 路由器:

路由器 R3，添加对应 VLAN111 的 IP 地址到 Wlan1 接口，并创建与 VLAN111 相同的加密配置，设置无线网卡 wlan1 的 mode=station

```
[admin@R3] >
/interface wireless security-profiles
add authentication-types=wpa-psk,wpa2-psk eap-methods=""
management-protection=allowed mode=dynamic-keys name=\
vlan111 supplicant-identity="" wpa-pre-shared-key=yusvlan111
wpa2-pre-shared-key=yusvlan111

[admin@R3] >
/ip address
add address=192.168.1.3/24 interface=wlan1
```

```
/interface wireless
set [ find default-name=wlan1 ] disabled=no mode=station security-profile=vlan111
```

R4 路由器:

路由器 R4，同样添加对应 VLAN222 的 IP 地址到 Wlan1 接口上，并创建与 VLAN222 相同的加密配置，设置无线网卡 wlan1 的 mode=station

```
[admin@R4] >
/interface wireless security-profiles
add authentication-types=wpa-psk,wpa2-psk eap-methods=""
management-protection=allowed mode=dynamic-keys name=\
vlan222 supplicant-identity="" wpa-pre-shared-key=yusvlan222
wpa2-pre-shared-key=yusvlan222

[admin@R4] >
/ip address
add address=192.168.2.4/24 interface=wlan1

/interface wireless
set [ find default-name=wlan1 ] disabled=no mode=station security-profile=vlan222
```

以上配置的也适用于终端笔记本，手机和平板接入，只需要在 R1 的 vlan111 和 vlan222 创建 DHCP 服务

DHCP 服务配置

R1 路由器创建 DHCP 服务，首先创建地址池

```
/ip pool
add name=vlan111 ranges=192.168.1.10-192.168.1.50
add name=vlan222 ranges=192.168.2.10-192.168.2.50
```

R1 上配置 DHCP 服务

```
/ip dhcp-server
add address-pool=vlan111 authoritative=after-2sec-delay disabled=no
interface=vlan111 name=server1
add address-pool=vlan222 authoritative=after-2sec-delay disabled=no
interface=vlan222 name=server2

/ip dhcp-server network
add address=192.168.1.0/24 dns-server=192.168.1.1 gateway=192.168.1.1
netmask=24
```

```
add address=192.168.2.0/24 dns-server=192.168.2.1 gateway=192.168.2.1
netmask=24
```

12.3 LED 配置

RouterOS 允许管理员配置每个 led（发光二极管）的活动状态。例如通过配置 led 显示无线信号强度，通过闪烁显示以太网接口传输活动状态，以及其他的选项等。Led 控制主要针对 RouterBOARD 设备设置。

操作路径: /system leds

下面是 Groove 设备的 led 默认配置：

```
[admin@MikroTik] /system leds> print
Flags: X - disabled
 #   TYPE                    INTERFACE             LEDS
 0   wireless-signal-strength                      led1
                                                   led2
                                                   led3
                                                   led4
                                                   led5
 1   interface-activity      ether1                user-led
```

RB Groove 使用 5 个 led 灯显示无线信号强度，一个显示以太网接口状态

6.0 支持 LED 无线信号强度显示的设备包括 RB400 系列、RB911/711 系列、RB SXT 和 Groove/Metal 系列，根据不同信号强度指定 LED 灯：

- 1 LED – 开启，当无线客户端连接到 AP 的信号强度 >=-89dBm
- 2 LED – 开启，当信号强度 >= -82dBm
- 3 LED - 开启，当信号强度 >= -75dBm
- 4 LED - 开启，当信号强度 >= -68dBm
- 5 LED -开启，当信号强度 >= -61dBm

对于 RB751 和 RB951 系列，5 个以太网接口 LED 和 1 个无线状态 LED

配置属性

属性	描述
disabled (*yes* | *no*; 默认: no)	是否选择禁用
interface (*字符*; 默认:)	接口名称，根据网络接口类型知道 led 显示类型
modem-signal-treshold (整型*[-113..-51]*; 默认:)	模块的信号强度设置
leds (led 列表; 默认:)	Led 列表中用于状态显示，例如：通过 5 个 led 灯显示无线信号强度
type (*ap-cap* | *flash-access* | *interface-activity* |	状态类型：

interface-receive | *interface-status* | *interface-transmit* | *modem-signal* | *wireless-signal-strength* | *wireless-status*;)	ap-cap – 在与 CAPsMAN 初始化状态下 CAP 闪烁，连接完成后稳定 *flash-access* – 当访问闪存 led 闪烁、 *interface-activity* – 界面活动 led 闪烁 *interface-receive* – 接口接收数据 led 闪烁 *interface-status* – 接口连接状态 led 亮起 *interface-transmit* – 接口发送数据 led 闪烁 *modem-signal* – 3G 模块信号 led 闪烁 (USB 或 miniPCIe 界面) *wireless-signal-strength* – 无线信号强度变化 led 灯亮起，需要多 led 支持 *wireless-status* – 无线连接状态 led 灯亮起

CAP 配置实例

在 RB951 上设置 user-led 灯，显示当前 CAP 状态

```
/system leds
add leds=user-led type=ap-cap
```

注意：6.23 开始 led 命令被删除，会直接使用/system leds 下对 led 灯进行控制，所以你在 6.23 版本下无法找到 led 命令，需要修改你的脚本在 RB 规范的类型使用

```
*) leds - removed 'led' command and added support for 'on', 'off' types under
'system leds';
```

在 type 中增加了两个属性 on 和 off，例如控制 user-led

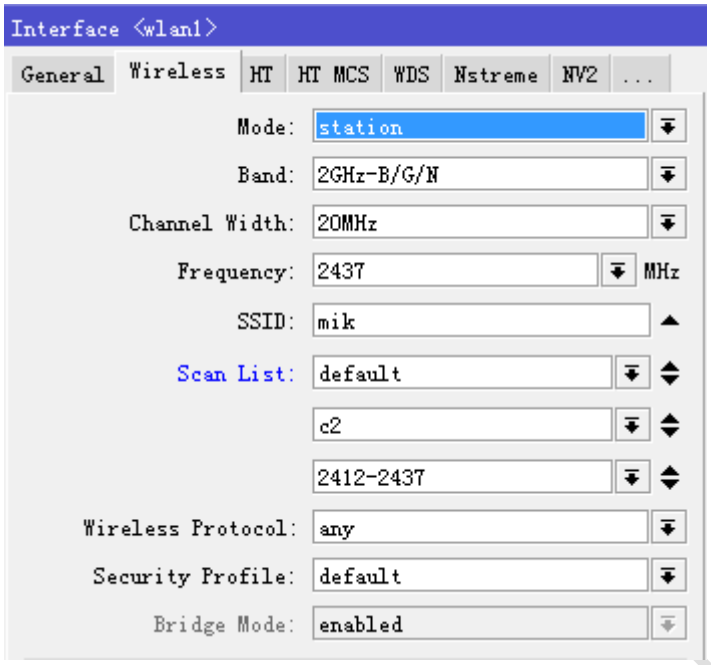
```
/system leds add leds=user-led type=off
```

12.4 Scan-list 搜索列表

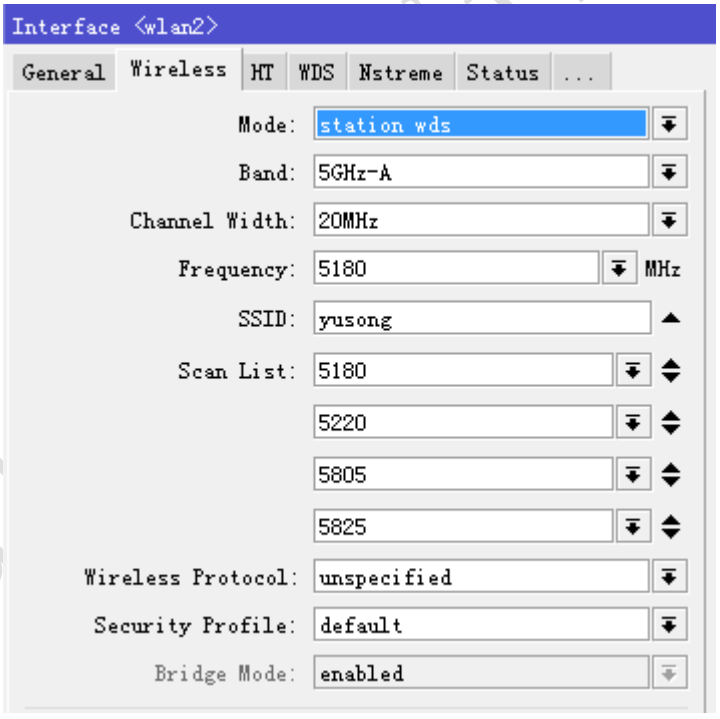
Scan-list 在之前的 Superchannel 介绍时提到，Scan-list 不仅用于指定 superchannel 的频率范围，还可以定义 RouterOS 客户端设备搜索频率范围

Scan-list 默认在 5ghz 频段下，每间隔 20MHz 步进搜索，在 5ghz-turbo 频段下每间隔 40MHz，2.4G 则间隔 5MHz。如果 scan-list 采用手动设定，所有指定的频率范围都会被锁定搜索（例如：scan-list=default,5200-5245,2412-2427 即会使用默认频段搜索，并添加从 5200-5245 或 2412-2427 频率范围.） 当然如果是 scan-list=2412-2437，即只能在 2412-2437 频率范围搜索。

从 RouterOS v6.0 开始对于 winbox 或 webfig 配置 Scan-list 输入多频率、频段范围，会被分隔为多个 Scan-list 选项。使用逗号间隔帧 v6.0 后的 winbox/webfig 不再支援。如下图的 winbox 设置：



如在 802.11a 无线网络规定了频率连接范围，station-wds 客户端设备只能连接 5180、5220、5805 和 5825 四个频率，我们在 v6.0 的网卡配置如下:



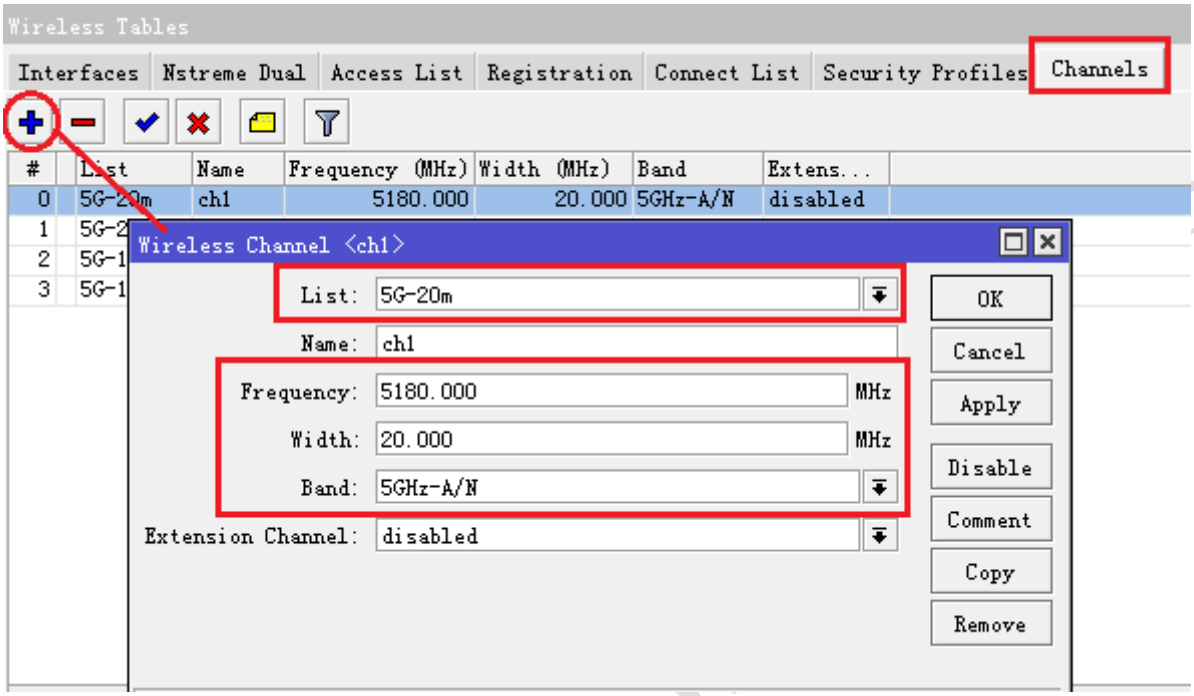
12.5 高级频率搜索 channels

RouterOS 从 v6 开始提供了一个高级的无线频率搜索列表配置- channels，仅支持 Atheros AR92xx 芯片，频率范围为 2192-2734MHz 和 4800-6100MHz，且支持中心频率 0.5MHz 步进，能定制带宽分为从 2.5～30MHz 每 0.5MHz 步进，无线网卡通过 Scan-list 调用该功能，同时 Scan-list 有可以定义搜索范围。

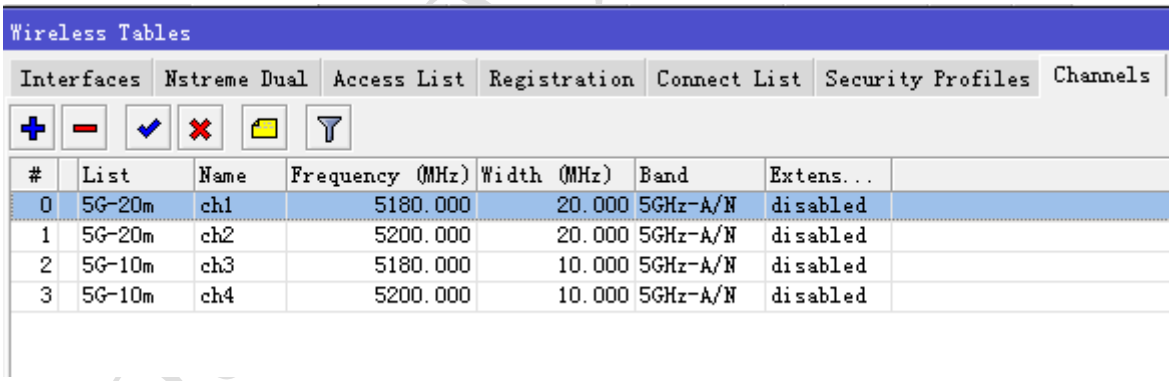
操作路径： /interface wireless channels

对于特殊的网络环境，我们希望设备采用不通的频率（frequency）和带宽（width），我们可以通过 channels 来定义，并配置到 Scan-list 中

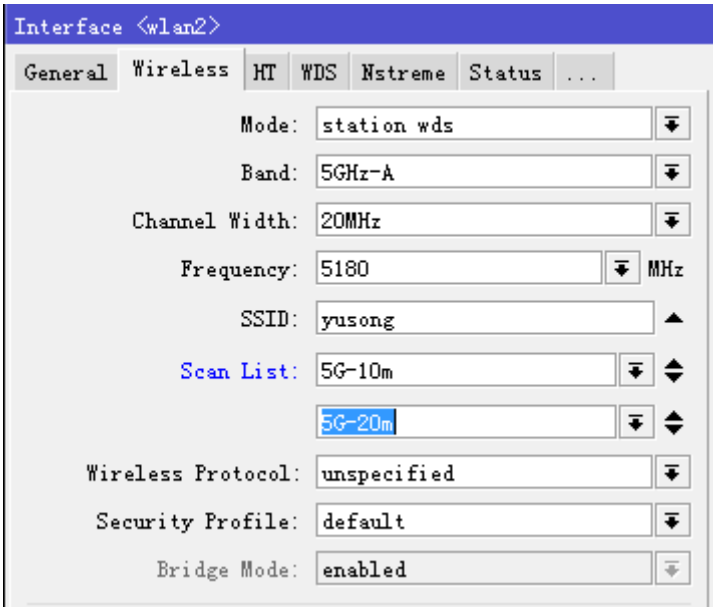
例如下面定义 802.11a 的无线网络环境的搜索范围和带宽，频率 5180 和 5200 分别使用 20Mhz 和 10Mhz 带宽



依次安装各个频率和带宽添加到 channels 下：



添加完成后，我们将 Channels 配置到 Scan-list 中



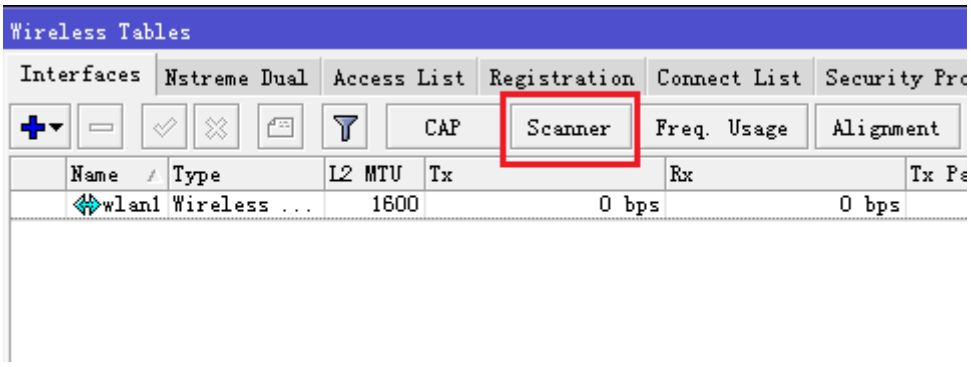
12.6 搜索器 Scaner

Scaner 工具，是用于搜索附近的 AP，搜索 AP 类型取决于无线网卡的型号和设定参数，搜索器便于你找到当前区域中的 AP 节点，并选择连接。当无线网卡设置为 2.4G 时，只能搜索 2.4G 频段中的 AP，同样当设置为 5G 时，只能搜索 5G 频段中的 AP

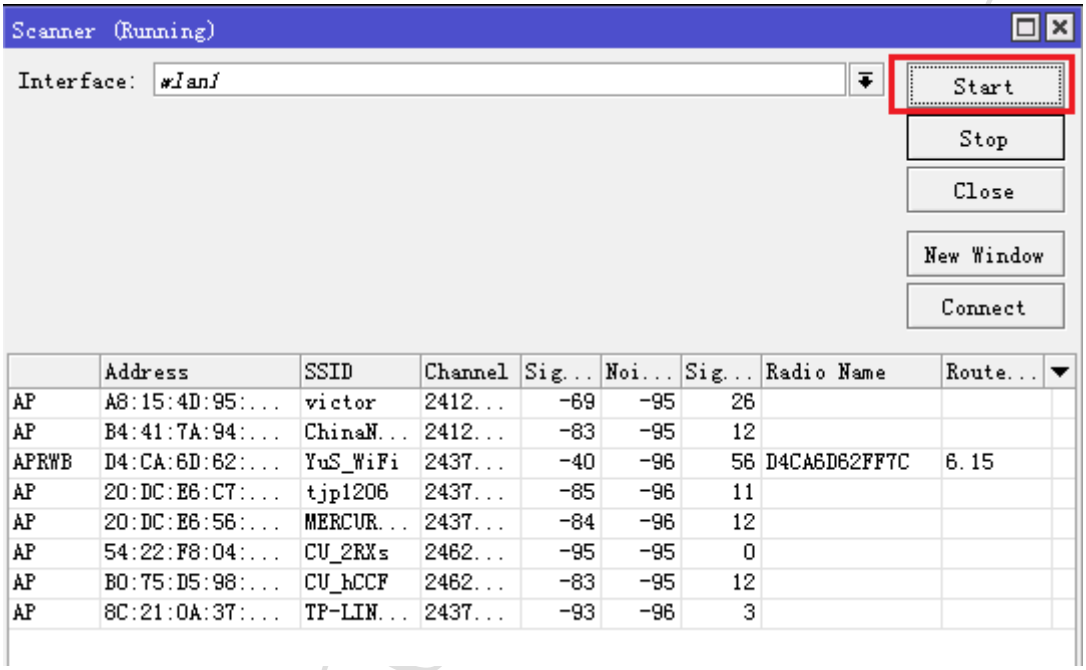
Scaner 搜索器，可以在 interface wireless 中开启，通过 scan 命令，如下面搜索 wlan1 2.4G 网络 AP

```
[admin@MikroTik] /interface wireless> scan wlan1
Flags: A - active, P - privacy, R - routeros-network, N - nstreme, T - tdma,
W - wds, B - bridge
          ADDRESS          SSID      CHANNEL        SIG   NF  SNR
RADIO-NAME
AP      A8:15:4D:95:65:E0 victor     2412/20-Ce/gn  -68  -95  27
AP      B4:41:7A:94:CB:B1 ChinaNe... 2412/20/gn     -71  -95  24
AP      20:DC:E6:56:A0:50 MERCURY... 2437/20-Ce/gn -85  -96  11
APR WB D4:CA:6D:62:FF:7C YuS_WiFi    2437/20/gn     -45  -96  51
D4CA6D62FF7C
AP      20:DC:E6:C7:C7:E8 tjp1206    2437/20-Ce/gn -85  -96  11
AP      28:2C:B2:63:0B:7C MERCURY... 2412/20-Ce/gn -93  -95   2
AP      9C:21:6A:AC:F3:D6 TP-LINK... 2412/20-Ce/gn -89  -95   6
```

Winbox 中可以在 wireless 菜单下找到



打开后，点击 start 开始搜索



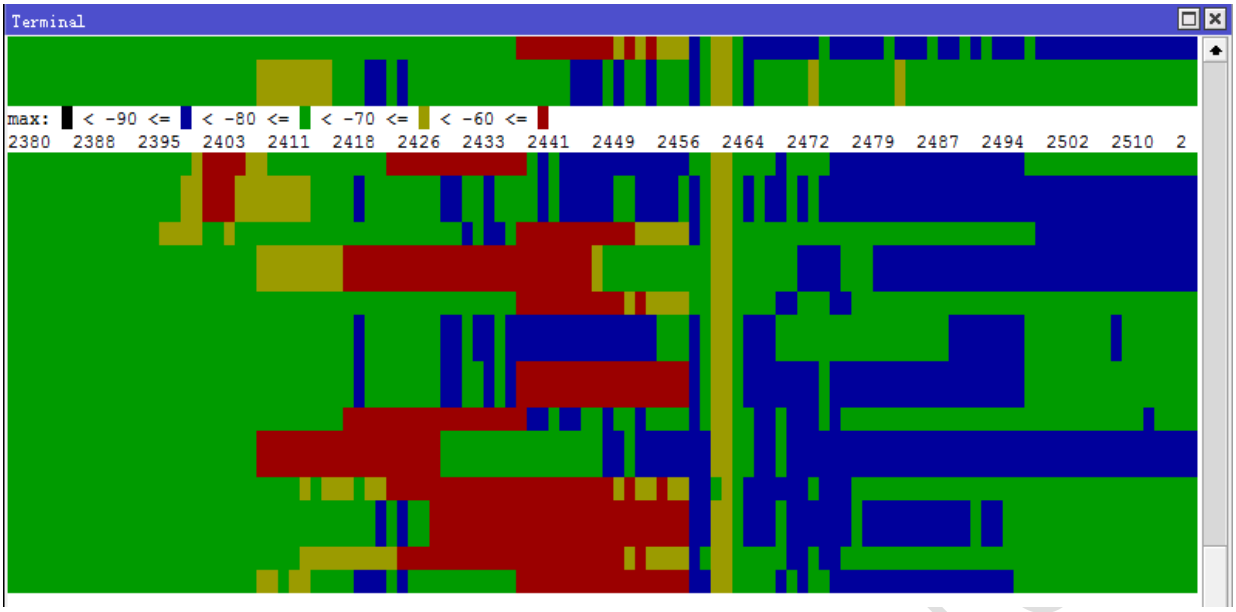
在 Scaner 菜单中，有 Connect 按钮，当你选择好的无线 AP 信号后，可以点 Connect 连接，这样你的无线网卡会自动将 SSID 修改为选定的 AP 的 SSID。

12.7 无线频谱扫描 spectral scan

spectral scan（频谱扫描） 能扫描无线网卡支持的所有频道，并在控制终端绘制出图片。精确度依赖于无线网卡性能。例如 R52n 支持的频道包括 4790～6085、 2182～2549。通过绘制频谱图可以得到周围的频率使用环境，用于分析无线频道情况，便于选择空闲的频道部署自己的网络。

当前支持 Atheros 芯片，如 AR9220, AR9280, AR9223，已经测试通过对无线网卡包括：R52N 和 R2N。

频谱历史（Spectral history）



命令行操作如下：

```
/interface wireless spectral-history <wireless interface name>
```

绘制频谱历史，图例和频率规则每 24 行一组显示，不同的背景颜色代表当前环境下不同的频道功率强度，在上图已经给出了不同颜色的区间值：

- 小于-90 为黑色
- 大于等于-90 小于-80 为蓝色
- 大于等于-80 小于-70 为绿色
- 大于等于-70 小于-60 为黄色
- 大于等于-60 为红色

功率越强代表当前环境下该频道的干扰越强。

频谱扫描 Spectral Scan

连续监测频谱资料，该命令类似'spectral-history'，只是通过显示每个频道的具体信号强度。每行显示一个频谱频率，通过字符图形显示， 通过":"显示平均功率值，平均峰值保持通过"."。

```
Terminal
FREQ  DBM  GRAPH
2382  -74  ::::::::::::::::::::::::::::::::::.
2388  -73  ::::::::::::::::::::::::::::::::::::.
2394  -73  :::::::::::::::::::::::::::::::::::.
2400  -72  ::::::::::::::::::::::::::::::::::::.
2406  -75  :::::::::::::::::::::::::::::::::.......
2411  -73  :::::::::::::::::::::::::::::::::::........
2417  -75  :::::::::::::::::::::::::::::::::......
2423  -78  ::::::::::::::::::::::::::::::.
2429  -77  ::::::::::::::::::::::::::::::.
2435  -80  :::::::::::::::::::::::::::.
2441  -37  ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::.
2446  -36  ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::.
2452  -56  :::::::::::::::::::::::::::::::::::::::::::::::::::::::::.
2458  -56  :::::::::::::::::::::::::::::::::::::::::::::::::::::::::.
2464  -70  :::::::::::::::::::::::::::::::::::::::.
2470  -80  :::::::::::::::::::::::::::.
2476  -80  ::::::::::::::::::::::::::.
2481  -81  :::::::::::::::::::::::::.
2487  -82  ::::::::::::::::::::::::.
2493  -83  ::::::::::::::::::::::.
2499  -74  ::::::::::::::::::::::::::::::::::.
2505  -73  :::::::::::::::::::::::::::::::::::.
2511  -74  ::::::::::::::::::::::::::::::::::.
2516  -75  :::::::::::::::::::::::::::::::::.
-- [Q quit|D dump|C-z pause|down]
```

命令行操作：

```
/interface wireless spectral-scan <wireless interface name>
```

show-interference – 新增一栏目为探测干扰源类型，干扰源类型如下：

- bluetooth-headset
- bluetooth-stereo
- cordless-phone
- microwave-oven
- cwa
- video-bridge
- wifi

参考文献:

http://wiki.mikrotik.com

http://www.routerboard.com

https://ros.tw/wp/

相关网络技术资料