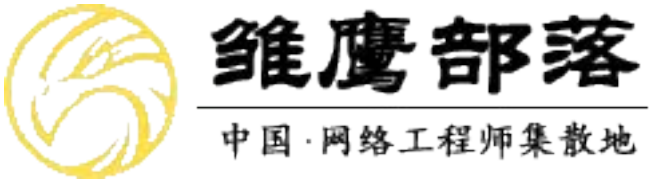


目录

Directed-broadcast（定向广播）

概述

在默认情况下，Cisco 路由器在收到任何广播数据的时候，默认都是丢弃而不转发，在某些时候，有些广播是必须的，比如我们通常使用的 NetBios Name Server 协议，DNS 协议，还有 DHCP 协议。就像 DHCP 协议，当主机在没有地址的情况下，向服务器请求 IP 地址，正因为自己没有 IP 地址，却又不知道服务器在哪，所以需

要使用广播来查找，这时，如果服务器在远程网络，而路由器又拒绝转发广播，那么将给我们的 DHCP 带来麻烦。

要如何才能在这种情况下让网络正常工作呢，那就是让路由器帮我们把广播转发到我们需要到达的目标网络，这样需要到达远程网络的广播，我们称为定向广播，比如我们本地网络是 10.1.1.0/24 的网段，我们需要将广播发到 192.168.1.0/24 的网段，那么只想让广播在 192.168.1.0/24 网段发送，广播地址为 192.168.1.255，但如果直接这样发，是到达不了 192.168.1.0/24 网段的，所以需要得到路由器的允许。

Directed-broadcast

每个网络都是与路由器的某个接口相连的，这个网络需要向远程网络发送广播，首先就是发送到路由器与之相连的接口，那么当路由器从该接口收到广播之后，决定是丢弃还是转发，就在于路由器的接口是否允许定向广播功能，如果我们需要让路由器帮我们转发定向广播，就需要在接口上手工配置 directed-broadcast 转发功能。

配置

1. 配置 Directed-broadcast

（1）定义 ACL

R1(config)#access-list 10 permit any

注：ACL 是用来告诉接口哪些数据包是可以传递定向广播的，加了 ACL 之后，只传递该 ACL

所有允许的数据，如果不加 ACL，默认传递所有定向广播数据。

（2）在接口上开启 directed-broadcast

R1(config)#int f0/0

R1(config-if)#ip directed-broadcast 10

注：IOS 12.0 开始，默认接口上是关闭 directed-broadcast 的。（注意 IOS 版本，请以自身为准。）

Forward-protocol

在路由器接口上开了 directed-broadcast 之后，路由器便能够将该接口上收到的定向广播发往相应的远程网络，比如从 10.1.1.0/24 收到的广播发往 192.168.1.255，但是如果一个目标地址为 255.255.255.255（此广播称为本地广播）的 DHCP 广播请求包，需要发往 DHCP 服务器所在的 192.168.1.0/24 网络（DHCP 服务器地址为 192.168.1.100），路由器会自动将该广播包转到 192.168.1.100 吗？答案当然是不会，那么又如何让路由器在收到目标地址为 255.255.255.255 的广播包，就知道要转发到 192.168.1.100 呢？这就需要在路由器接口上定义 ip helper-address，之后路由器就会将广播转成单播发到 ip helper-address 后面的目标 IP，但是要注意的是，什么样的广播才会被转发到 ip helper-address 后面的目标 IP，是需要根据 forward-protocol 来定义的，如果 forward-protocol 不允许任何协议，那么路由器就不会将任何广播发到 ip helper-address 后面的目标 IP。

forward-protocol 默认允许通过的协议为：

Trivial File Transfer (TFTP) (port 69)

Domain Name System (port 53)

Time service (port 37)

NetBIOS Name Server (port 137)

NetBIOS Datagram Server (port 138)

Boot Protocol (BOOTP) client and server datagrams (ports 67 and 68)

TACACS service (port 49)

所以默认情况下，在接口上加了 ip helper-address，路由器也就只能转发这些默认的端口。

配置

1．配置 forward-protocol

（1）定义可以转发的协议和端口

```
R1(config)#ip forward-protocol udp 3001
```

注：默认是开启 forward-protocol 的，且只能转发默认协议和端口。

（2）在接口下配置 ip helper-address

R1(config)#int f0/0

R1(config-if)# ip helper-address 192.168.1.100

注：默认是关闭 ip helper-address 的。

说明：通过以上配置之后，当路由器从 f0/0 接口上收到目标地址为 255.255.255.255 的广播，并且目标为 UDP 3001 的广播之后，就会转成单播发往 192.168.1.100。如果没有配置 forward-protocol，路由器就只能转发默认的协议和端口。

DHCP

概述

DHCP 应该是一个大家都非常熟悉的协议，可能算不上什么 feature，平时我们使用电脑时，经常会用到，它的功能也是大家所知道的：动态地为主机分配 IP 地址以节省工作量。由此可以看出，DHCP 可以由两个部分组成，即 DHCP 服务器和 DHCP 客户端（通常为我们所使用的 PC）。那么在 Cisco 设备分别扮演 DHCP 服务器和 DHCP 客户端时，与其它设备有什么不同之处呢，下面来详细地介绍。

DHCP 是一个协议，无论它运行在主机上，还是在路由器或交换机上，其运行的规则，就像 TCP/IP 协议一样，是不允许我们使用任何命令更改的，DHCP 前身是 BOOTP 协议，使用的端口号为：服务器端是 UDP 67(cisco 设备上称为 bootps)，客户端是 UDP 68（cisco 设备上称为 bootpc）,CCIE 考生需牢记。当一台主机接入网络后，在没有 IP 地址的情况下，向网络上发送 DHCP 请求获得 IP 地址时，正因为自己还没有 IP 地址，所以发送数据包时，源 IP 为 0.0.0.0,源 MAC 正常，而自己也不可能知道谁是 DHCP 服务器，所以数据包是目标 IP 地址为 255.255.255.255、目标 MAC 地址为 FFFFFF 的广播包。当本地网络中如果存在 DHCP 服务器，那么 DHCP 服务器从某个网卡收到请求后，便向客户端发送一个地址信息，其中包含我们需要使用的 IP 地址，子网掩码，网关，DNS 等信息，同时这些信息会携带一个租约时间（即这个地址客户端可以使用多久，过了这个时间，那么服务器将该地址提供给其它客户端使用），当然，客户端也可以提前结束该地址的使用。

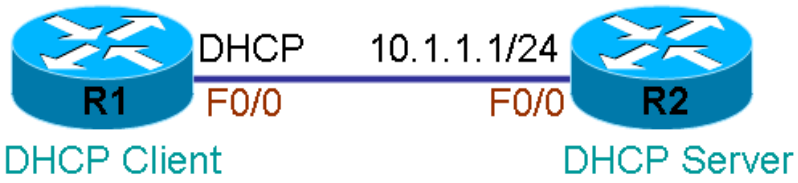
当我们使用的客户端为 windows 主机，其接入网络之后，当网卡配置为 DHCP

获得地址时，就开始向网络中请求地址，先发送一个广播包，等待 1 秒之后，如果没有服务器应答，就开始尝试发送第二个广播包，如果又等了 9 秒没有收到应答，则发送第三个广播包，第三个是等 13 秒，还没有应答，最后再发送一个包，等待 16 秒后，最终在四个广播包没有应答的情况下，也就是从发送第一个请求包到 39 秒之后，默认是放弃请求，但最后也会为网卡自动配上一个私有 IP 地址，地址段为 169.254.0.0/16,并且你会看到网卡连接图标上出现黄色感叹号，状态名为“受限制或无连接”，即使这时网络中出现了 DHCP 服务器，也救不了这台主机，这就是 windows 主机作为 DHCP 客户端的情况。那么使用 Cisco 设备作为 DHCP 客户端的不同之处是什么呢？当 Cisco 设备的接口配置为 DHCP 获得地址时，便向网络中发出广播，如果等待 5 秒都没有收到应答，就再次发送，再等 10 秒，没有收到就再发，然后再等 15 秒,20 秒，25 秒，30 秒 ••• 60 秒，由此可以看出其间隔是随着次数的增加而每次加 5 秒，但尝试到 60 秒后，便不再增加，又会重新回到 5 秒，以次类推，Cisco 设备在得不到地址的情况下，并不会为接口自动配上网段为 169.254.0.0/16 的地址，所以如果网络中之后出现 DHCP 服务器，就会为该设备的接口分发一个 IP 地址。当 DHCP 客户端得到 IP 地址后，因为地址使用是有时间限制的，当这个时间过去一半的时候，客户端会向服务器续约，以请求继续使用该地址，服务器同意后，该地址的使用时间会被刷新，如果在时间过去一半时，续约不成功，便会在总时间过去 75% 的时候再续约一次，如果还不成功，就会放弃该地址的使用权。

服务器与客户端之间并没有 hello 这样的数据包来保持会话状态，所以当一台客户端得到一个 IP 地址的使用权后，中途离开网络，服务器是无法知道的，也就无法将该 IP 地址重新分配给他人使用，所以建议大家在配置服务器时，可以将租约配的越短越好，以免造成一个地址发给客户使用，而这台客户机已经离开了，该 IP 地址还长时间不能重新发给新的客户使用，建议租约配置为 1 分钟，因为一个地址在租约过半时，客户端会续约，也就是可以再次使用同一个地址，所以不用担心一分钟之后，客户端会重新获得别的 IP 地址。当一台 DHCP 客户端收到服务器提供的 IP 地址后，会使用 Gratuitous ARP 来查讯网络，即使用该 IP 地址为目的 IP，目标 MAC 为 FFFFFF 发到网络里，如果有人回答该数据包，则证明该 IP 地址在网络中已经有他人在使用，那么将向 DHCP 服务器重新请求获得别的 IP 地址。（注：Gratuitous ARP 通过正常手段无法关闭）

配置

DHCP 基础



1．配置 DHCP Server

（1）开启 DHCP 功能

r2(config)#service dhcp

（2）配置 DHCP 地址池

r2(config)#ip dhcp pool ccie1　　地址池名为 ccie1

r2(dhcp-config)#network 10.1.1.0 255.255.255.0　可供客户端使用的地址段

r2(dhcp-config)#default-router 10.1.1.1　　网关

r2(dhcp-config)#dns-server 10.1.1.1 10.1.1.2　DNS

r2(dhcp-config)#lease 1 1 1　　租期为 1 天 1 小时 1 分（默认为一天）

r2(config)#ip dhcp pool ccie2　　地址池名为 ccie1

r2(dhcp-config)#network 20.1.1.0 255.255.255.0　可供客户端使用的地址段

r2(dhcp-config)#default-router 20.1.1.1　　网关

r2(dhcp-config)#dns-server 20.1.1.1 20.1.1.2　DNS

r2(dhcp-config)#lease 1 1 1　　租期为 1 天 1 小时 1 分（默认一天）

（3）去掉不提供给客户端的地址

注：因为某些 IP 地址不希望提供给客户端，比如网关地址，所以我们要将这些地址从地址池中移除，这样服务器就不会将这些地址发给客户端使用。

r2(config)#ip dhcp excluded-address 10.1.1.1 10.1.1.10　移除 10.1.1.1 到 10.1.1.10

r2(config)#ip dhcp excluded-address 20.1.1.1 20.1.1.10　移除 20.1.1.1 到 20.1.1.10

2．配置 DHCP Client

（1）配置接口使用 DHCP

r1(config)#int f0/1

r1(config-if)#ip address dhcp

3．查看命令：

（1）在服务器上查看哪些地址分配给了哪些主机：

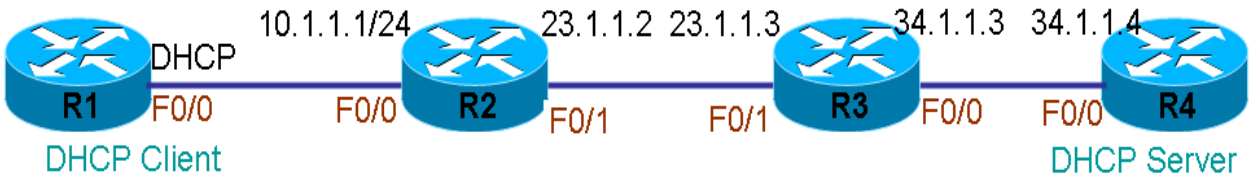
R2#Show ip dhcp binding

4．查看结果

查看 DHCP Client 会看到接口 F0/0 的 IP 地址为 10.1.1.11 并且产生一条指向 10.1.1.1 的默认路由（换成 PC 就会变成网关是 10.1.1.1），路由器并不需要得到 DNS。

在这里，DHCP Server 上明明配了两个地址池，网段分别为 10.1.1.0/24 和 20.1.1.0/24,为什么客户端向服务器请求地址的时候，服务器就偏偏会把 10.1.1.0/24 网段的地址发给客户，而不会错把 20.1.1.0/24 网段的地址发给客户呢。这是因为服务器从哪个接口收到 DHCP 请求，就只能向客户端发送地址段和接收接口地址相同的网段，如果不存在相同网段，就会丢弃请求数据包。图中接收接口地址为 10.1.1.1，而地址池 ccie1 中的网段 10.1.1.0/24 正好和接收接口是相同网段，所以向客户端发送了 IP 地址 10.1.1.11。

DHCP 中继



如图中所示，当 R1 的接口配置为 DHCP 获得地址后，那么将从 F0/0 发出目的地为 255.255.255.255 的广播请求包，如果 R2 为 DHCP 服务器，便会响应客户端，但它不是 DHCP 服务器，因此 R2 收到此广播包后便默认丢弃该请求包。而真正的 DHCP 服务器是 R4，R1 的广播包又如何能到达 R4 这台服务器呢，R4 又如何向 R1 客户端发送正确的 IP 地址呢。

路由器是不能够转发广播的，因此，除非能够让 R2 将客户端的广播包单播发向 R4 这台服务器。 我们的做法就是让 R2 将广播包通过单播继续前转到 R4 这台服务器，称为 DHCP 中继，通过 IP help-address 功能来实现。

1. R2 配置

（1）配置将 DHCP 广播前转到 34.1.1.4

注：IP help-address 功能默认能够前转 DHCP 协议，所以无需额外添加。

```
R2(config)#int f0/0
R2(config-if)#ip helper-address 34.1.1.4
```

2. 配置 DHCP Server:

（1）开启 DHCP 功能

```
R4(config)#service dhcp
```

（2）配置 DHCP 地址池

```
R4(config)#ip dhcp pool ccie1                地址池名为 ccie1
R4(dhcp-config)#network 10.1.1.0 255.255.255.0    可供客户端使用的地址段
R4(dhcp-config)#default-router 10.1.1.1          网关
```

R4(config)#ip dhcp pool ccie2　　地址池名为 ccie1

R4(dhcp-config)#network 34.1.1.0 255.255.255.0　　可供客户端使用的地址段

R4(dhcp-config)#default-router 34.1.1.4　　网关

（3）去掉不提供给客户端的地址

R4(config)#ip dhcp excluded-address 10.1.1.1 10.1.1.10　移除 10.1.1.1 到 10.1.1.10

R4(config)#ip dhcp excluded-address 34.1.1.1 34.1.1.10　移除 20.1.1.1 到 20.1.1.10

（4）配置正确地址池的路由

R4(config)#ip route 10.1.1.0 255.255.255.0 34.1.1.3

注：　R3 无需做任何配置！

3．查看结果

查看 DHCP Client 会看到接口 F0/0 的 IP 地址为 10.1.1.11，那么 DHCP 服务器 R4 又是根据什么来判断出客户端需要的是哪个网段的 IP 地址呢，为什么还是没有错把 34.1.1.0/24 网段的地址发给客户呢。不是说服务器从哪个接口收到请求，就把这个接口相同网段的地址发给客户端吗？按照之前的理论，应该是发送 34.1.1.0/24 的地址给客户啊。在这里，能够指导服务器发送正确 IP 地址给客户端，是因为有一个被称为 option 82 的选项，这个选项只要 DHCP 请求数据包被中继后便会自动添加，此选项,中继路由器会在里面的 giaddr 位置写上参数，这个参数，就是告诉服务器，客户端需要哪个网段的 IP 地址才能正常工作。中继路由器从哪个接口收到客户的 DHCP 请求，就在 option 82 的 giaddr 位置写上该接收接口的 IP 地址，然后服务器根据 giaddr 位置上的 IP 地址，从地址池中选择一个与该 IP 地址相同网段的地址给客户，如果没有相应地址池，则放弃响应，所以，服务器 R4 能够正确发送 10.1.1.0/24 的地址给客户，正是因为 R2 在由于 IP help-address 的影响下，将 giaddr 的参数改成了自己接收接口的地址，即将 giaddr 参数改成了 10.1.1.1，通过 debug 会看到如下过程：

*Mar100:28:36.666: DHCPD: setting giaddr to 10.1.1.1.

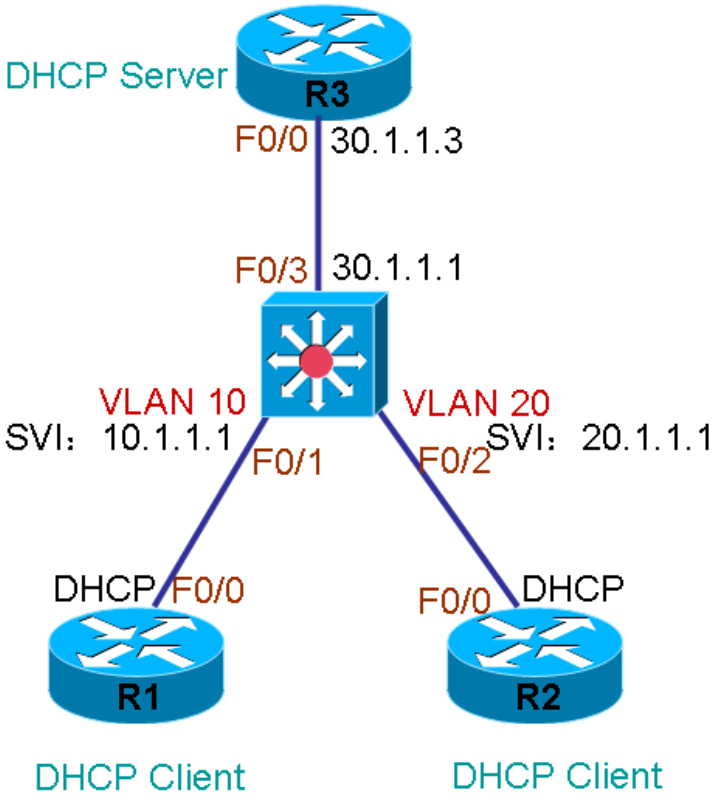
*Mar100:28:36.666:DHCPD:BOOTREQUESTfrom0063.6973.636f.2d30.3031.322e.

6439.6639.2e63.3638.302d.4661.302f.30 forwarded to 34.1.1.4.

从上面 debug 信息可以看到 R2 是将 giaddr 改成 10.1.1.1 后发中继发向 34.1.1.4 的，需要知道的是，经过中继后发来的 DHCP 请求包如果 giaddr 位置不是某个 IP 地址而是 0.0.0.0 的话，服务器是丢弃该请求而不提供 IP 地址的。

注：当服务器上存在 10.1.1.0/24 网段的地址池时，服务器要将该地址池发送给客户，就必须存在到达 10.1.1.0 网段的路由(默认路由也行)，并且客户端必须位于该路由的方向，如果方向不对，该地址池也是不能够发给客户使用的。

不同 VLAN 分配不同地址



如图 3 中所示，两个 DHCP 客户端分别位于交换机上两个不同的 VLAN，交换机上的 VLAN 接口将作为他们的网关，R3 是 DHCP 服务器，这两个客户端必须得到不同网段的地址，否则无法与外网通信，在这种情况下，服务器 R3 也必须正确为 R1 分配 10.1.1.0/24 网段的地址，必须为 R2 分配 20.1.1.0/24 的地址，配置如下：

1. 配置 DHCP Server

（1）开启 DHCP 功能

R3(config)#service dhcp

（2）配置 DHCP 地址池

R3(config)#ip dhcp pool ccie1　　地址池名为 ccie1

R3(dhcp-config)#network 10.1.1.0 255.255.255.0　　可供客户端使用的地址段

R3(dhcp-config)#default-router 10.1.1.1　　网关

R3(config)#ip dhcp pool ccie2　　地址池名为 ccie1

R3(dhcp-config)#network 20.1.1.0 255.255.255.0　　可供客户端使用的地址段

R3(dhcp-config)#default-router 20.1.1.1　　网关

（3）去掉不提供给客户端的地址

R3(config)#ip dhcp excluded-address 10.1.1.1 10.1.1.10　移除 10.1.1.1 到 10.1.1.10

R3(config)#ip dhcp excluded-address 20.1.1.1 20.1.1.10　移除 20.1.1.1 到 20.1.1.10

（4）配置正确地址池的路由

R3(config)#ip route 10.1.1.0 255.255.255.0 30.1.1.1

R3(config)#ip route 20.1.1.0 255.255.255.0 30.1.1.1

2. 配置交换机

（1）配置相应接口信息

sw(config)#vlan 10

sw(config-vlan)#exit

```
sw(config)#vlan 20
sw(config-vlan)#exit
sw(config)#int f0/1
sw(config-if)#switchport mode access
sw(config-if)#switchport access vlan 10
sw(config-if)#exit
sw(config)#int f0/2
sw(config-if)#switchport mode access
sw(config-if)#switchport access vlan 20
sw(config-if)#exit
sw(config)#int vlan 10
sw(config-if)#ip address 10.1.1.1 255.255.255.0
sw(config-if)#ip helper-address 30.1.1.3          单播前转 DHCP 广播到 30.1.1.3
sw(config-if)#exit
sw(config)#int vlan 20
sw(config-if)#ip address 20.1.1.1 255.255.255.0
sw(config-if)#ip helper-address 30.1.1.3          单播前转 DHCP 广播到 30.1.1.3
```

3．配置 DHCP Client

（1）配置 R1

```
r1(config)#int f0/1
r1(config-if)#ip address dhcp
```

(2)配置 R2

r2(config)#int f0/1

r1(config-if)#ip address dhcp

4. 查看结果：

按上述配置完之后，客户端 R1 的 F0/0 便能够收到地址 10.1.1.11,客户端 R2 便能够收到地址 20.1.1.11，然后就可以全网通信。在上述的情况下，服务器 R3 能够正确为 R1 分配 10.1.1.0/24 网段的地址，能够正确为 R2 分配 20.1.1.0/24 网段的地址，同样也是因为交换机在收到 R1 的 DHCP 广播包后，将 giaddr 的参数改成了 10.1.1.1，收到 R2 的广播包后，将 giaddr 的参数改成了 20.1.1.1，所以最后服务器 R3 能够根据 giaddr=10.1.1.1 的包分配 10.1.1.0/24 的地址，根据 giaddr=20.1.1.1 的包分配 20.1.1.0/24 的地址。

IP 与 MAC 地址绑定

在配置 DHCP 时，地址池中除了移除掉的 IP 地址之外，所有的地址都会按顺序分配给客户，所以客户机得到的 IP 地址是无法固定的，有时需要每次固定为某些 PC 分配相同的 IP 地址，那么这时就可以配置 DHCP 服务器以静态将 IP 地址和某些 MAC 绑定，只有相应的 MAC 地址才能获得相应的 IP 地址。在 Cisco 设备上静态将 IP 与 MAC 绑定的方法为，需要将某个 IP 地址绑定给 MAC 地址，就为该 IP 地址单独创建地址池，称为 host pool，地址池中需要注明 IP 地址和掩码位数，并且附上一个 MAC 地址，以后这个 IP 地址就只分配给这个 MAC 地址，所以 host pool 只能有一个 IP 地址和一个 MAC 地址，如果需要为多个客户绑定 IP 和 MAC，就必须得单独为每个客户都配置各自的 host pool，还要注意的是，在 host pool 中，MAC 地址的表示方法和平常不一样，比如一个主机网卡的 MAC 地址为 aabb.ccdd.eeff，在地址池中，需要在前面加上 01（01 表示为以太网类型），结果为 01aa.bbcc.ddee.ff

1. 配置 host pool:

（1） 配置 pool 名

r1(config)#ip dhcp poo ccie

（2）配置 IP 地址

r1(dhcp-config)#host 10.1.1.100 /24

（3）配置与该 IP 地址对应的 MAC 地址

r1(dhcp-config)#client-identifier 01aa.bbcc.ddee.ff

2. 查看配置结果：

（1）查看服务器地址分配状态

r1#sh ip dhcp binding

Bindings from all pools not associated with VRF:

IP address Client-ID/ Lease expiration Type

Hardware address/

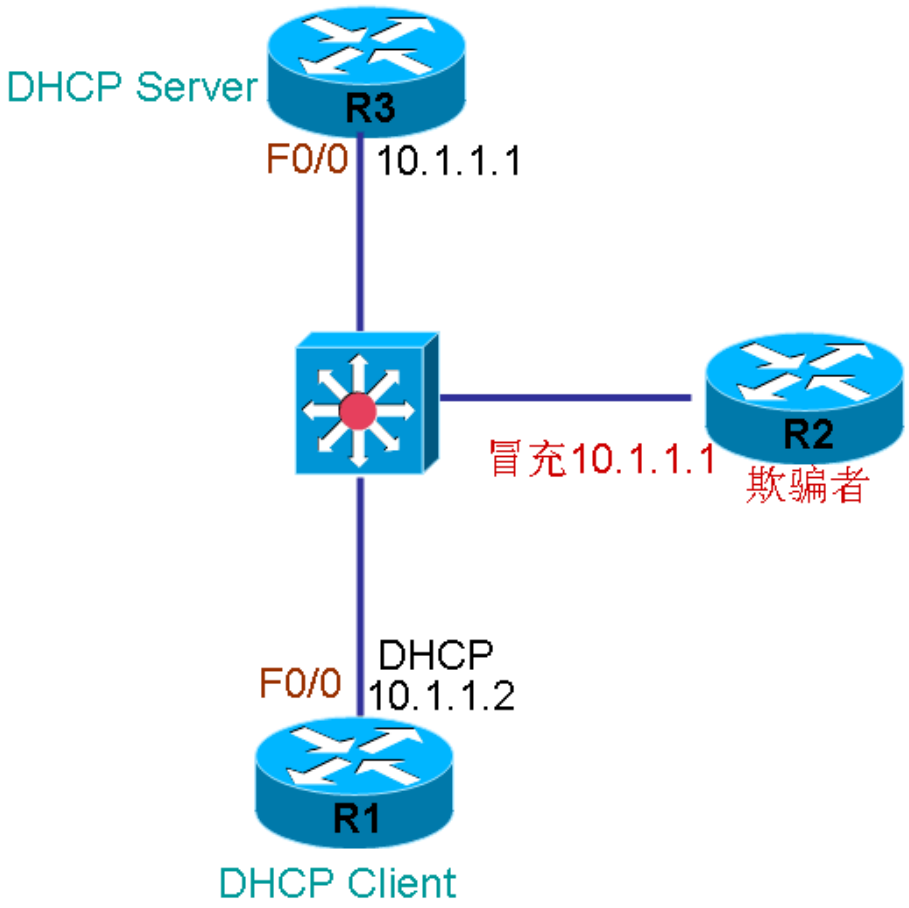
User name

10.1.1.100 01aa.bbcc.ddee.ff Infinite Manual

r1#

说明：从以上结果可以看出，IP 地址 10.1.1.100 已经手工与 MAC 地址 aabb.ccdd.eeff 做了绑定，以后只要 MAC 地址为 aabb.ccdd.eeff 的客户端请求 IP 地址时，才能获得 IP 地址 10.1.1.100。

DHCP 安全 ARP



Cisco 设计的 DHCP 安全 ARP 也许不是绝对的安全，但也起到了一定的作用，原本设计为一个需要计费的公共热点 PVLAN（公共无线场所），如图 4 中所示，R3 为 DHCP 服务器，为付费的 R1 提供正确 IP 地址以提供网络服务，当服务器 R3 为客户端 R1 提供 IP 地址 10.1.1.2 之后，就已经记住了它的 MAC 地址，在正常情况下，如果 R1 退出，服务器是不知道的，并且当网络中有欺骗者接入后，也可冒充 10.1.1.2 这个地址进行上网，当然 R1 和 R2 的 MAC 地址肯定是不一样的，如果这时服务器 R3 由于自动更新 ARP 表的 MAC 地址，就能够顺利让 R2 上网。

基于上述原因，需要在服务器 R3 和客户端 R1 之间提供某种安全机制，即服务器定期 ARP 讯问 10.1.1.2 是否还存在，在讯问时，只有 R1 能够回答。

在完成这种机制，需要两个 feature 来支持，第一个是 Update Arp，在地址池模式下开启，这个 feature 便是定期讯问网络中 DHCP 客户端的;第二个是 Authorized ARP（ARP 授权）,只能在以太网接口下开启，功能是禁止该接口下通过 ARP 自动更新和学习 MAC 地址，这样一来，接口下将不能有手动配置 IP 的设备接入，因为手工配置 IP 接入后，服务器不会更新自己的 ARP 表，也就无法完成到新设备的二层 MAC 地址封装，也就无法和新设备进行通信，只有合法的 DHCP 客户端才能正常通

信，所以，如果为远程客户端分配 IP 地址，就无法做这样的保护，并且到远程客户端的下一跳必须是自己的客户端，因为如果不是，是无法通信的，因为 ARP 不存在到它的条目。

1．配置安全 ARP：

（1）开启 DHCP 功能

R3(config)#service dhcp

（2）配置 DHCP 地址池

R3(config)#ip dhcp pool ccie1　　地址池名为 ccie1

R3(dhcp-config)#network 10.1.1.0 255.255.255.0　　可供客户端使用的地址段

R3(dhcp-config)#default-router 10.1.1.1　　网关

R3(dhcp-config)#update arp　　开启定期 ARP 讯问

（3）去掉不提供给客户端的地址

R3(config)#ip dhcp excluded-address 10.1.1.1 10.1.1.10　移除 10.1.1.1　到 10.1.1.10

（4）在接口下开启 Authorized ARP

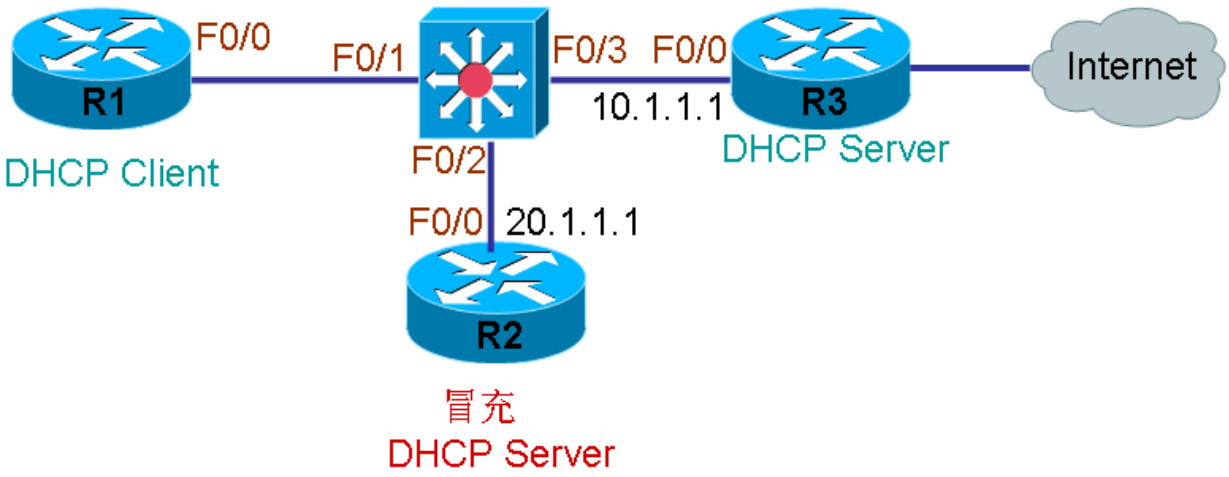
R3(config)#int f0/0

R3(config-if)#Router(config-if)# arp authorized　　禁止动态更新 ARP

R3(config-if)# arp timeout 60　　60 秒客户无应答则删除 ARP 条 目

说明：通过以上配置之后，当 DHCP 客户端从服务器获得 IP 地址后，服务器便会定期查讯该 IP 地址，如果 60 秒没有回答，便从 ARP 表中删除该条目。

DHCP 监听



如图 5 中所示，客户端 R1 只有正确从服务器 R3 中获得 10.1.1.0/24 网段的 IP 地址才能够正确上网，如果当网络中出现另外一台错误的 DHCP 服务器（图中 R2），R2 向客户端 R1 发出 20.1.1.0/24 的地址，那么将导致 R1 网络中断，在这样的情况下，就需要禁止不合法的 DHCP 服务器向网络中提供 DHCP 服务，这就需要 DHCP 监听（DHCP Snooping）。DHCP Snooping 是在交换机上完成的，如上图中，只要告诉交换机，只有 F0/3 发来的 DHCP 应答地址才转发给客户端，其它接口发来的应答地址统统被丢弃。要做到这一点，就要告诉交换机，F0/3 接口是它可能信任的 DHCP 地址，其它接口都是不可信的，不能提供 DHCP 应答，那么在实现这个功能时，就需要将交换机上的接口分为可信任接口和不可信任接口两种，默认交换机全为不可信任接口，也就是说交换机开启 DHCP Snooping 之后，没有任何一个接口上的 DHCP 服务器能提供服务。在交换机上配置 DHCP Snooping 时，必须指明在哪个 VLAN 上进行监听，其它没有监听的 VLAN 不受上述规则限制。

1．交换机上配置 DHCP Snooping

注：交换机上所有接口全部划入 VLAN1

（1）在交换机上开启 DHCP Snooping

sw(config)#ip dhcp snooping　　开启 DHCP Snooping

sw(config)#ip dhcp snooping vlan 1　　在交换机上启用 DHCP Snooping

（2）将相应接口变为信任接口（默认全部为不可信）

sw(config-if)#ip dhcp snooping trust

2. 查看命令：

（1）查看 dhcp snooping

Sw#sh ip dhcp snooping

说明：通过以上配置之后，只有交换机 F0/3 接口上（信任接口）的设备能够应答 DHCP 请求，而其它所有接口，比如 R2 过来的 DHCP 应答是会被丢弃的。但是你会发现，在这之后，R1 还是无法获得服务器 R3 发来的 DHCP 地址。这是因为开了 DHCP Snooping 的交换机默认会产生中继效果，即将 DHCP 请求包的 giaddr 的参数改成 0.0.0.0，交换机的这种中继效果是无法关闭的，当一个服务器收到中继后并且将 giaddr 设置为 0.0.0.0 而不是 IP 地址的请求包时，默认是要丢弃该数据包而不作应答的，所以服务器 R3 丢弃了该请求数据包。要让客户 R1 能够正常收到 DHCP 提供的 IP 地址，就要让 DHCP 服务器对即使 giaddr 为 0.0.0.0 的请求包也作出应答。配置如下：

R3(config-if)#ip dhcp relay information trusted

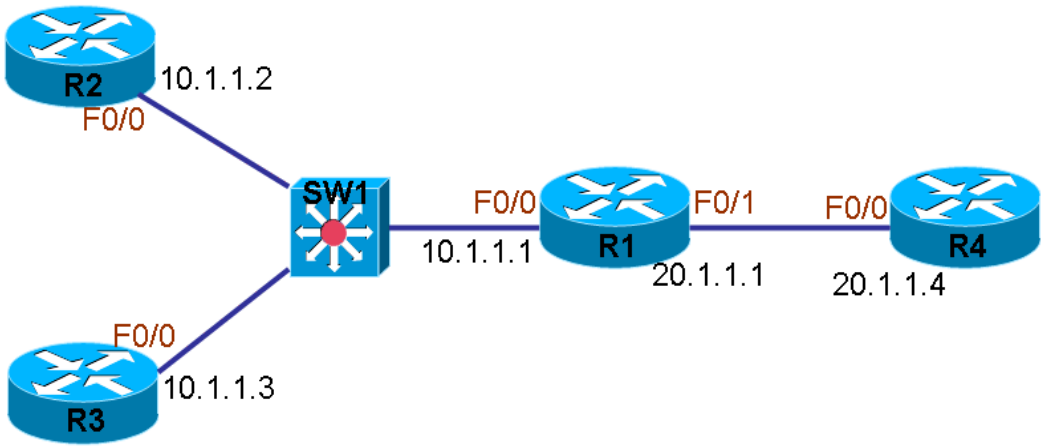
最后，从上图中，如果 R3 本身还不是 DHCP 服务器，如果 DHCP 服务器还在远程网络，需要 R3 提供中继并转发该请求包到服务器的话，那么 R3 除了在接口下配置 ip dhcp relay information trusted 之外，还必须配置 ip helper-address，两者缺一不可。

IP Accounting（记账）

有时需要在路由器上查看某台主机通过路由器的流量是多少，这时就需要路由器能够记录下该主机的数据量。主机之间进行通信时，发出的数据包都有源 IP 和目的 IP 共两个 IP 地址，路由器在记录流量时，可根据这些 IP 地址来定义要记录的流量。虽然通信时数据包有两个地址，但路由器只需要定义一个地址即可，这个地址不需要说明是源还是目的，也就是说一个数据包无论是源 IP 匹配还是目的 IP 匹配，都会被路由器记录，但是在记录条目里，会同时写上流量发生的源 IP 和目的 IP。在记录流量时，单位是 Byte,其中包含了数据包的包头和数据大小。在路由器开启记账功能后，会影响到路由器的工作性能，请慎用。需要注意的是，路由器只能记录从接口出去的流量，并且从自己发出的流量和发往自己的流量是不能记录的。路由器记录的每一条包含一个源 IP 和一个目标 IP，这一对称为一个条目，路由器能记录的

条目数量是可以随意更改的，默认最多能存储 512 条。

配置



1. 配置 IP Accounting

（1）在路由器接口下直接开启 IP Accounting

r1(config)#int f0/0

r1(config-if)#ip accounting

注：只能记录从接口 f0/0 出去的数据包

2. 测试结果

（1）在 R2 上 ping 20.1.1.4，以测试 R2 到 R4 的数据不作记录，只有 R4 返回 R2 的才记录。

r2#ping 20.1.1.4

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 20.1.1.4, timeout is 2 seconds:

!!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/8 ms

（2）再从 R4 ping R2

r4#ping 10.1.1.2

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 10.1.1.2, timeout is 2 seconds:

!!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 8/8/12 ms

(3)查看 R1 上的流量记录

r1#sh ip accou

*Mar 1 00:08:53.750: %SYS-5-CONFIG_I: Configured from console by console

r1#sh ip accounting

Source	Destination	Packets	Bytes
20.1.1.4	10.1.1.2	10	500

Accounting data age is 0

说明：可以看出，R2 到 R4 的流量没有做记录，因为没有目的为 20.1.1.4 的记录，只看到有 R4 到 R2 的流量记录，因为有目的为 10.1.1.2 的记录。

（4）再测试 R1 的接口 f0/0 是否还会记录别的流量

r3#ping 20.1.1.4

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 20.1.1.4, timeout is 2 seconds:

!!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/8 ms

r3#

r1#sh ip accounting

Source	Destination	Packets	Bytes
20.1.1.4	10.1.1.2	10	1000
20.1.1.4	10.1.1.3	5	500

Accounting data age is 2

说明：从以上数据表明，R1 会记录从接口 f0/0 出去的任何流量

（5）再测试到 R1 的流量

r3#ping 20.1.1.1

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 20.1.1.4, timeout is 2 seconds:

!!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/8 ms

r3#

r1#sh ip accounting

Source	Destination	Packets	Bytes
20.1.1.4	10.1.1.2	10	1000
20.1.1.4	10.1.1.3	5	500

Accounting data age is 2

说明：发现没有记录到 R1（即 IP 地址为 20.1.1.1）的流量，说明从 R1 发起的流量和发到 R1 的流量是不做记录的。

3．配置单独记录到某固定 IP 的流量

（1）配置 R1 只记录到 R3(10.1.1.3)的流量

r1(config)#ip accounting-list 10.1.1.3 0.0.0.0 （后面为通配符掩码 wildcard）

（2）在接口上应用记账功能

r1(config)#int f0/0

r1(config-if)#ip accounting output-packets

注：命令 ip accounting output-packets 和 ip accounting 功能相同。

（3）测试 R3 到 R4 的流量

r3#ping 20.1.1.4

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 20.1.1.4, timeout is 2 seconds:

!!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/8 ms

（4）测试 R2 到 R4 的流量

R2#ping 20.1.1.4

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 20.1.1.4, timeout is 2 seconds:

!!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/8 ms

（5）查看流量记录

r1#sh ip accounting

Source	Destination	Packets	Bytes
20.1.1.4	10.1.1.3	5	500

Accounting data age is 1

说明：从结果中看出，只要数据包中有 10.1.1.3 这个地址，便会记录，其它统统不作记录。

NetFlow

概述

在需要对 Cisco 设备上查看数据的流量传输情况的时候，我们可以用到的技术是 IP Accounting，但是可以看出这个技术记录出的流量是非常简洁的，只能看到数据包的量，却看不到数据包的协议。下面有一项技术在 IP Accounting 的基础上增加了一些有用的附加功能，它不仅能记录数据的数量，并且还可以记录出协议等信息，这就是 NetFlow。

Cisco 开发的 NetFlow 共 4 个版本，分别是 ver 1 ，ver 5，ver 8，ver 9，它们的特点是：

V9 不向后兼容 v8 和 v5，需要独立开启。

V8 只支持聚合缓存，不支持新 feature。

V5 只支持主缓存，不支持新 feature。

V1，不要使用，建议用 5 和 9。

Netflow 在网络设备上抓包，在每台设备上独立进行，不需要所有设备开启。

配置 Netflow 的条件：

（1）必须先开启路由功能，

（2）CEF 必开

（3）并且会消耗额外 CPU 和内存资源。

Netflow 抓的包包含：

IP-to-IP

IP-to- MPLS

Frame Relay

ATM

egress (outgoing)

下面 7 个参数相同即被认为是同一流，作相同记录，否则另作记录

（1）Source IP address

（2）Destination IP address

（3）Source port number

（4）Destination port number

（5）Layer 3 protocol type

（6）Type of service (ToS)

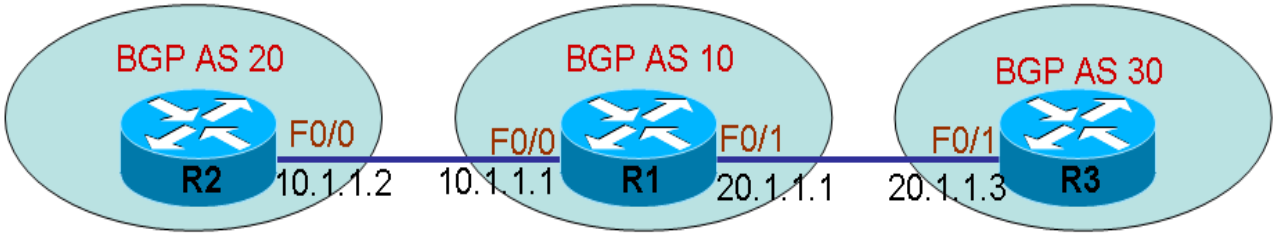
（7）Input logical interface

Netflow 抓到的包可以向远程主机发送，发送时使用协议 UDP，端口号默认为 9991，这些远程主机的 IP 和端口号可以随意更改。

配置

说明：Netflow 是基于接口开启的，在某个接口开启后，就在相应接口的相应方向抓取数据包，

在接口上开启 Netflow 时，有先决条件需要打开。



1. 全局开启 CEF（必开）

R1(config)ip cef

2. 接口开启 Netflow（可开多个接口）

（1）早于 IOS 版本 12.2(14)S, 12.0(22)S, or 12.2(15)T 的

r1(config)#int f0/0

r1(config-if)#ip route-cache flow

（2）等于或晚于 IOS 版本 12.2(14)S, 12.0(22)S, or 12.2(15)T 的

r1(config)#int f0/0

r1(config-if)#ip flow ingress

3. 定义远程采集主机，(最多两台)

（1）定义远程 IP 地址，设备将抓过的数据包发向远程主机（默认端口为 UDP 9991）

R1(config)Ip flow-export destination 10.1.1.2 90

(2)配置数据包源地址

R1(config)# ip flow-export source f0/0

4．定义 Netflow 版本

（1）全局定义 Netflow 版本（默认版本 1，不同 IOS 支持的版本不同）

r1(config)#Ip flow-export ver 5

5．查看效果：

（1）在 r2 上 ping R3 的 20.1.1.3

r2#ping 20.1.1.3

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 20.1.1.3, timeout is 2 seconds:

!!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 4/5/12 ms

r2#

(2)在 R1 上查看记录：

r1#sh ip cache flow

IP packet size distribution (10 total packets):

1-32 64 96 128 160 192 224 256 288 320 352 384 416 448 480

.000 .000 .000 1.00 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

512 544 576 1024 1536 2048 2560 3072 3584 4096 4608

.000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

IP Flow Switching Cache, 278544 bytes

2 active, 4094 inactive, 2 added

6 ager polls, 0 flow alloc failures

Active flows timeout in 30 minutes

Inactive flows timeout in 15 seconds

last clearing of statistics never

Protocol Total Flows Packets Bytes Packets Active(Sec) Idle(Sec)

-------- Flows /Sec /Flow /Pkt /Sec /Flow /Flow

SrcIf SrcIPaddress DstIf DstIPaddress Pr SrcP DstP Pkts

Fa0/0 10.1.1.2 Fa0/1 20.1.1.3 01 0000 0800 5

Fa0/1 20.1.1.3 Fa0/0 10.1.1.2 01 0000 0000 5

说明：从以上结果可以看出，拥有很详细的数据包记录。

（3）查看更详细

r1#sh ip cache verbose flow

IP packet size distribution (10 total packets):

1-32 64 96 128 160 192 224 256 288 320 352 384 416 448 480

.000 .000 .000 1.00 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

512 544 576 1024 1536 2048 2560 3072 3584 4096 4608

.000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

IP Flow Switching Cache, 278544 bytes

0 active, 4096 inactive, 2 added

32 ager polls, 0 flow alloc failures

Active flows timeout in 30 minutes

Inactive flows timeout in 15 seconds

last clearing of statistics never

Protocol	Total Flows	Flows /Sec	Packets /Flow	Bytes /Pkt	Packets /Sec	Active(Sec) /Flow	Idle(Sec) /Flow
ICMP	2	0.0	5	100	0.0	0.0	15.0
Total:	2	0.0	5	100	0.0	0.0	15.0

SrcIf Port Msk AS	SrcIPaddress	DstIf Port Msk AS	DstIPaddress NextHop	Pr TOS Flgs B/Pk	Pkts Active

r1#

并且看到协议也有所记录。

(4).可清除总记录

R1#clear ip flow stats

6. 聚合参数

说明：可以将数据包中某些需要的数据聚合后显示，比如 BGP AS 号码，或者前缀等信息

（1）配置聚合 BGP AS（事先配好 BGP）

r1(config)#ip flow-aggregation cache as　　进入聚合配置模式

r1(config-flow-cache)#cache entries 2000　　配置可聚合的最多条目

r1(config-flow-cache)#cache timeout inactive 200　　配置不活动会话的超时时间

r1(config-flow-cache)#cache timeout active 50　　配置活动会话的时间

r1(config-flow-cache)#enabled　　开启

（2）配置记录中 Origin-as 在源和目的中包含 as

R1(config)#ip flow-export version 5 peer-as

7．查看结果：

(1)从 R2 ping r3

R2#ping 20.1.1.3

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 20.1.1.3, timeout is 2 seconds:

!!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/8 ms

R2#

（2）查看记录：

r1#sh ip cache flow aggregation as

IP Flow Switching Cache, 132104 bytes

4 active, 1996 inactive, 4 added

398 ager polls, 0 flow alloc failures

Active flows timeout in 50 minutes

Inactive flows timeout in 200 seconds

Src If	Src AS	Dst If	Dst AS	Flows	Pkts	B/Pk	Active
Fa0/1	30	Fa0/0	0	1	10	100	0.9
Fa0/0	0	Fa0/1	30	1	10	100	0.9

r1#

说明：可以看到拥有的 AS 记录情况。

WCCP

概述

WCCP 被称为网页信息缓存协议，目的在于通过缓存用户曾经访问过的网页数据，在用户下次访问相同网页时读取缓存信息来加快用户的访问速度。

WCCP 的工作原理为每当用户访问一个页面时，WCCP 就将这些页面进行缓存保留，当用户下次请求网页数据时，路由器将用户的请求发到引擎，如果缓存中有相同网页的备份，则直接从缓存中发给用户，提高速度；但是如果没有，就自己重新请求远程网页，然后再发给用户。

配置

1．开启 WCCP：

（1）启用 WCCP：

Router(config)# ip wccp web-cache

（2）开启 WCCP ver2

(共两个版本，默认为 ver2)

router(config)#ip wccp version 2

（3）在接口上启用：

Router(config)# interface f0/0

Router(config-if)# ip wccp web-cache redirect out

注：在 f0/0 接口的 out 方向启用，表明用户接在 f0/0 上，路由器只将从网站服务器发回的数

据从 f0/0 发给用户时，才会缓存。

（4）限制 WCCP 缓存哪些 WEB 服务器过来的数据包（可选配置）：

例：比如只接收服务器 10.1.1.1（即 ACL10 过来的数据）发来的数据：

router(config)# access-list 10 permit host 10.1.1.1

router(config)# ip wccp web-cache group-list 10

（5）限制到哪些主机或哪些服务器的不被缓存（可选配置）：

例：从服务器发回到目标主机 192.168.196.51 的不缓存，其它全部缓存

Router(config)# access-list 100 deny ip any host 192.168.196.51

Router(config)# access-list 100 permit ip any any

Router(config)# ip wccp web-cache redirect-list 100

（6）缓存组播能力，事先需要路由器配置组播（可选配置）：

例：让路由器缓存到组播目标为 224.2.2.2 的数据

Router(config)# ip wccp web-cache group-address 224.2.2.2

Router(config)# interface interface-number f0/0

Router(config-if)# ip wccp web-cache group-listen

DRP （Director Response Protocol）

导向应答协议

概述

当大型网络里面有多台分布的服务器时，用户在请求服务器数据的时候，可能无法定位与自己最近的服务器，那么这个时候可能需要在网络里放置导向器,导向器的功能就是将用户的请求信息重新定位（重定向）到与之最近的服务器。所以导向器要能够查讯网络里路由器上的 BGP 和 IGP 路由信息，这就需要路由器提供支持，如果路由器能够完全提供这种路由信息的支持，就必须配置成 DRP Server Agent（DRP 服务器代理）。并且建议为网络里所有路由器均配置为 DRP 服务器代理，因为导向器每次将用户的请求定向到服务器代理后，他们将都能够向用户提供响应，而用户便能够采用最快响应的数据，而忽略其它所有。

配置路由器为 DRP 服务器代理，让网内的导向器能够有效的指导和分发流量，这些导向器需要查讯 BGP 和 IGP 的路由信息，所以需要 DRP 协议。

路由器要成为服务器代理，必须满足以下要求：

1.必须在拓朴上和导向器是邻近的且路由上是可达的。

2.应该具有全网的路由信息，IGP 和 BGP 都不能少，须保证自己全网可达。

3.保证能够访问所有导向器。

配置

1.打 DRP agent.

```
r1(config)#ip drp server
```

2.写 ACL 允许哪些能访问

（1）定义 ACL

r1(config)#access-list 10 permit 10.1.1.1

（2）加上 ACL:

r1(config)#ip drp access-group 10

3.使用认证以提供安全：

（1）定义 Key Chain

r1(config)#key chain ccie

r1(config-keychain)#key 1

r1(config-keychain-key)#key-string cisco

r1(config-keychain-key)#exit

r1(config-keychain)#exit

(2)使用认证

r1(config)#ip drp authentication key-chain ccie

4.查看结果：

r1#sh ip drp

IP Event Dampening(IP 事件惩罚)

概述

在网络中，有时会因为一些外在的或者人为的因素，可能引起设备接口翻动（一会儿可用，一会儿不可用），这些接口状态的变化将导致路由器重新计算最优路径，并且将这些路由信息通告给邻居，从而影响的不仅是自身的路由计算和系统资源消耗，还将影响到整个网络的所有设备重新计算路由而消耗巨大的系统资源，这样的

结果将是不可想象的。

IP Event Dampening 监控接口的状态，在接口状态经常性变化，不稳定的时候，就抑制路由协议对这些接口的计算，并且也限制在这些接口上建立路由邻居，直到这些接口回复稳定状态为止。那么，IP Event Dampening 认为什么样的接口为稳定状态，什么样的接口为经常性变化的呢，在这里，有一套自己的标准，这些标准包含四个参数，分别为：

(1)Suppress Threshold（抑制阀值），　　1 to 20000; the default is 2000.

(2)Half-Life Period （半衰期）　　1 to 30 seconds. The default is 5 seconds.

(3)Reuse Threshold （重新使用阀值）　　1 to 20000 default value is 1000 penalties

(4)Maximum Suppress Time （最大抑制时间） 1 to 20000 seconds.deis 20 seconds (即 4 倍的半衰期)

(1))Suppress Threshold（抑制阀值）

当一个接口由于翻动而要被 IP Event Dampening 抑制住，这接口的惩罚值必须累加到一定的数额才行这个数额就是 Suppress Threshold（抑制阀值），默认是 2000，范围是 1-20000。

(2)Half-Life Period （半衰期）

当一个接口的惩罚值到达抑制阀值被抑制住后，自己的抑制阀值会随着时间的流逝而慢慢降低，这个下降的速度由 Half-Life Period （半衰期）来控制，也就是每过去一个半衰期的时间，惩罚值的数额就降为总数额的一半，默认半衰期为 5 秒，范围是 1-30 少，比如一个接口的惩罚值为 2000，5 秒钟过去后，这个值就为 2000 的一半，即 1000。

(3)Reuse Threshold （重新使用阀值）

当一个接口被抑制住后，如果还要重新被路由协议接受或重新使用，这个接口的惩罚值必须降到一定的数额才行，这个数额就是 Reuse Threshold （重新使用阀值） ,默认为 1000，范围是 1-20000。

(4)Maximum Suppress Time （最大抑制时间）

接口每经过一个 UP 和 down 的状态，就被认为是翻动一次，每翻动一次，惩罚值就会加 1000，但是为了防止一个接口由于翻动次数过多，而真正等到稳定之后，由于抑制时间过长而不能重新被使用的可能，所以定义了最大抑制值，但定义的不是值，而是一个时间，这个时间意为一个接口被抑制住后，最多过多少时间可以再次被使用，默认为 20 秒，即为半衰期的 4 倍。

在使用 IP Event Dampening 时，会有一些限制，这些限制为：不支持子接口的监控，不支持虚拟接口（即 Virtual Templates）的监控，

当惩罚出现后，与之接口的路由将不出现在路由表中，（包括静态路由）

协议包含：RIP, OSPF, EIGRP, IS-IS, and BGP:，HSRP，CLNS

配置

1.在接口下开启 Dampening

例：配置 半衰期为 30 秒，重新使用阀值为 500，抑制阀值为 1000，最大抑制时间为 100 秒

```
r1(config)#int f0/0

r1(config-if)#dampening 30 500 1000 100
```

2. 查看配置

```
r1#show interface dampening
```

3.测试效果

（1）让接口翻动，即让接口 shutdown，再 up

```
r1(config)#int f0/0

r1(config-if)#shutdown

r1(config-if)#no shutdown
```

（2）查看状态

r1#sh dampening interface

1 interface is configured with dampening.

1 interface is being suppressed.

No features are using interface dampening.

r1#

r1#sh int dampening

FastEthernet0/0

Flaps Penalty Supp ReuseTm HalfL ReuseV SuppV MaxSTm MaxP Restart

1 811 TRUE 21 30 500 1000 100 5039 0

r1#

说明：从上面可以看到显示有 1 个接口已被抑制，接口已翻动一次，当前还剩惩罚值为 811,离重新使用时间还剩 21 秒，半衰期为 30 秒，重新使用阀值为 500，抑制阀值为 1000，最大抑制时间为 100 秒。

Core dump（核心崩溃）

概述

核心崩溃是路由器检测到自己发生了一些无法恢复的错误但需要重启自己才能正常工作，并非所有错误都会造成路由器核心崩溃。当出现核心崩溃，路由器重启自己时，其实所有接口都是能够正常工作的，也是能够发送数据包的，但这个时候它并不会为用户转发数据包，只会转发系统必须的数据包。

既然能够在接口上发送数据包，路由器便考虑将自己内存中所有的数据内容复制到远程服务器，这时内容中所有的内容将打包成一个二进制文件，称为 core dump file，这个文件可能会很能够大；远程服务器的类型包括(FTP), (TFTP), 和 Remote Copy Protocol (RCP,建议使用 FTP，如果不指定,默认使用 TFTP。当远程服务器为路由器时，发生核心崩溃的路由器只将文件的前 16MB 传送过去，即使对方内容再大也不会完全传送过去。

配置

1. 配置 core dump

（1）指定核心崩溃时的传输协议 （默认为 TFTP）

r1(config)#exception protocol ftp

（2）定义 FTP 的用户名和密码

r1(config)#ip ftp username ccie

r1(config)#ip ftp password cisco

（3）定义远程的服务器

r1(config)#exception dump 10.1.1.1

（4）定义核心崩溃时将内存中文件打包的名字（默认为 主机名-core）

r1(config)#exception core-file dumpfile

2. 测试

注:此测试只是测试传送文件，并不测试系统崩溃）

R1#write core

Remote host [10.1.1.1]?

Base name of core files to write [dumpfile]?

writing uncompressed ftp://10.77.233.129/cdfile1

Writing cdfile1 !!!

!!!!!!!!!!!!!!!!!!!!!!!!!

GLBP （Gateway Load Balancing Protocol ）

网关负载均衡协议

概述

GLBP 是思科的私有协议，在一般高端设备才会支持。GLBP 和 HSRP、VRRP 的目的是一样的，用多台路由器为 LAN 提供网关负载能力，但是 HSRP 和 VRRP 是将多台路由器配成一个组并虚拟成一个 IP 地址，用户将数据发到这个 IP 地址，组内只有一台活动路由器，其它都是备份路由器，只有活动路由器才能转发用户的数据包，而备份路由器只有当活动路由器不可用时才有机会为用户提供数据转发。可以看出，HSRP and VRRP 的备份路由器在平时是没有使用的，要想让所有的路由器都被使用，只有配置多个组，并且用户的主机都要分别指到相应的组 IP。

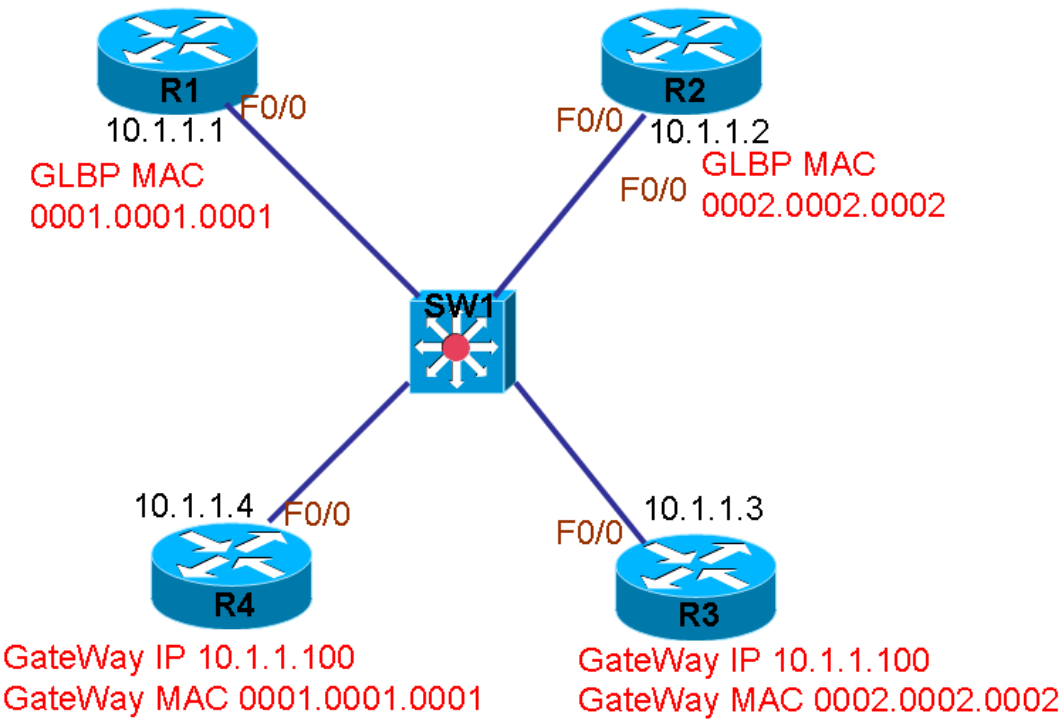
考虑到以上问题，GLBP 在将多台路由器配置成一个组时，也同时向用户提供一个单一的 IP 地址，额外增加的功能是让组内所有的成员都能够为用户提供数据转发，而不存在空闲的路由器。做法是在为用户提供单一 IP 的同时，每台路由器都为用户提供不同的 MAC 地址，这样，用户发到组 IP 的数据包就成功地被分担各个路由器上了。而组内的各个成员之间也有 hello 数据包来进行交流，3 秒发一个，使用组播地址 224.0.0.102, UDP 协议，端口号 3222。

在 GLBP 组中，选择一台路由器做为 active virtual gateway (AVG),其它路由器做为 AVG 的备份，在 AVG 不可用时顶替 AVG 的角色。在组中，AVG 的任务只是为各成员分配虚拟 MAC 地址，因为大家都是同一个 IP，只要大家的 MAC 地址不同，就可以同时为用户提供数据转发，用户的数据包目标 MAC 发到谁身上，谁就转发相应的数据包。

这些能够为用户提供数据转发的路由器被称为 active virtual forwarders (AVFs) ，

所以 AVG 也是 AVF，

图例说明 GLBP：



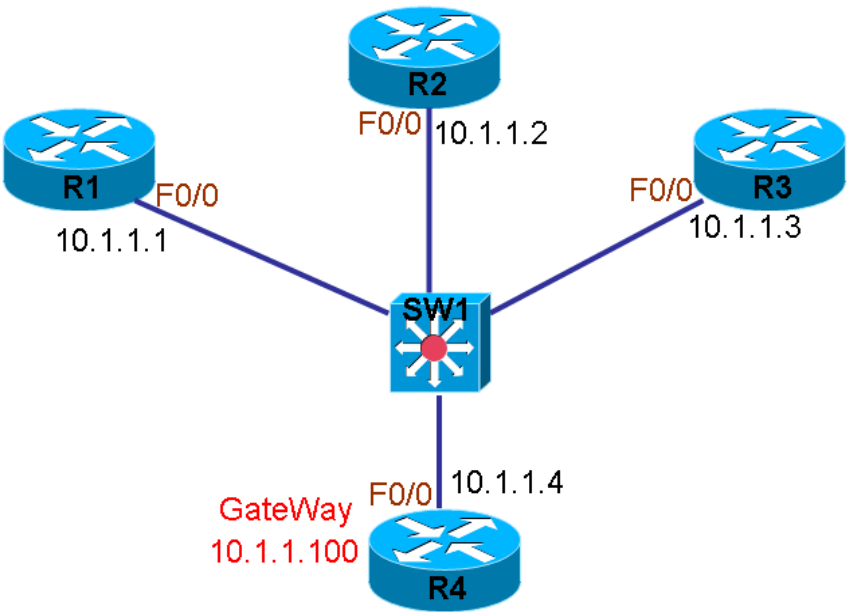
从上图中，R1 和 R2 配置为 GLBP 中成员，组 IP 为 10.1.1.100,而 R3 和 R4 为主机，两台主机的网关 IP 都为 10.1.1.100，但是由于 GLBP 中两个组成员的虚拟 MAC 分别为 0001.0001.0001 和 0002.0002.0002,所以虽然主机的网关 IP 都是相同的，只要他们映射的 10.1.1.100 的 MAC 不一样，即可发送到不同的 GLBP 成员。现在 R3 学习到 10.1.1.100 的 MAC 为 0002.0002.0002，所以最终它发向 10.1.1.100 的数据包将被 R2 接收，而 R4 学习到 10.1.1.100 的 MAC 为 0001.0001.0001，所以最终它发向 10.1.1.100 的数据包将被 R1 接收。这样因为 GLBP 中组成员的 IP 虽然相同，但 MAC 不相同，所以能够为用户实现负载均衡。

GLBP 组最多可以有 024 virtual routers 和四个虚拟 MAC 地址，主的为活动网关，其它都停在 listen state 状态，可以定义优先级来竞选： 1 到 255 ，然后是 higher IP address

GLBP 的抢夺模式是关闭的。一台路由器转发用户流量的多少，可以利用 weighting 来控制，也可以 track 一个接口，接口失效就降低相应权重。

组内所有成员必须配置相同的组号，并且如果是 VLAN 不同，那么组号也必须是不同的。

配置



1. 配置 GLBP

说明：配置 R1，R2，R3 为 GLBP 组成员

（1）配置 R1（并配置为 AVG，即优先级最高）

R1(config)# interface fastethernet 0/0

R1(config-if)#ip add 10.1.1.1 255.255.255.0

R1(config-if)# glbp 10 ip 10.1.1.100

R1(config-if)# glbp 10 priority 200　　配置优先级为 200，使其成为 AVG，默认 100。

R1(config-if)# glbp 10 preempt

（2）可选配置

R1(config-if)#glbp 10 weighting 200　　配置权重，影响分担流量

R1(config-if)#glbp 10 weighting track 1 decrement 30　配置监控信息

(3)配置 R2

R2(config)# interface fastethernet 0/0

R2(config-if)#ip add 10.1.1.2 255.255.255.0

R2(config-if)# glbp 10 ip 10.1.1.100

R2(config-if)# glbp 10 preempt

(4)配置 R3

R3(config)# interface fastethernet 0/0

R3(config-if)#ip add 10.1.1.3 255.255.255.0

R3(config-if)# glbp 10 ip 10.1.1.100

R3(config-if)# glbp 10 preempt

2．查看结果：

（1）查看 AVG（R1）的状态

r1#sh glbp

FastEthernet0/0 - Group 10

State is Active

2 state changes, last state change 00:05:52

Virtual IP address is 10.1.1.100

Hello time 3 sec, hold time 10 sec

Next hello sent in 1.910 secs

Redirect time 600 sec, forwarder time-out 14400 sec

Preemption enabled, min delay 0 sec

Active is local

Standby is 10.1.1.3, priority 100 (expires in 9.366 sec)

Priority 200 (configured)

Weighting 100 (default 100), thresholds: lower 1, upper 100

Load balancing: round-robin

There are 3 forwarders (1 active)

Forwarder 1

State is Active

1 state change, last state change 00:05:42

MAC address is 0007.b400.0a01 (default)

Owner ID is 0013.1a2f.0680

Redirection enabled

Preemption enabled, min delay 30 sec

Active is local, weighting 100

Forwarder 2

State is Listen

MAC address is 0007.b400.0a02 (learnt)

Owner ID is 0012.da88.3560

Redirection enabled, 599.563 sec remaining (maximum 600 sec)

Time to live: 14399.563 sec (maximum 14400 sec)

Preemption enabled, min delay 30 sec

Active is 10.1.1.2 (primary), weighting 100 (expires in 9.559 sec)

Forwarder 3

State is Listen

MAC address is 0007.b400.0a03 (learnt)

Owner ID is 0013.19f8.8be0

Redirection enabled, 598.625 sec remaining (maximum 600 sec)

Time to live: 14398.625 sec (maximum 14400 sec)

Preemption enabled, min delay 30 sec

Active is 10.1.1.3 (primary), weighting 100 (expires in 8.625 sec)

r1#

说明：可以看出 R1 状态为 Active，即自己为 AVG，且自己的虚拟 MAC 为：0007.b400.0a01，其它 AVF 的 MAC 分别为 0007.b400.0a02 和 0007.b400.0a03。

（2）查看 R2

r2#

r2#sh gl

r2#sh glbp

FastEthernet0/0 - Group 10

State is Listen

Virtual IP address is 10.1.1.100

Hello time 3 sec, hold time 10 sec

Next hello sent in 1.105 secs

Redirect time 600 sec, forwarder time-out 14400 sec

Preemption enabled, min delay 0 sec

Active is 10.1.1.1, priority 200 (expires in 9.715 sec)

Standby is 10.1.1.3, priority 100 (expires in 7.171 sec)

Priority 100 (default)

Weighting 100 (default 100), thresholds: lower 1, upper 100

Load balancing: round-robin

There are 3 forwarders (1 active)

Forwarder 1

State is Listen

MAC address is 0007.b400.0a01 (learnt)

Owner ID is 0013.1a2f.0680

Time to live: 14399.711 sec (maximum 14400 sec)

Preemption enabled, min delay 30 sec

Active is 10.1.1.1 (primary), weighting 100 (expires in 9.711 sec)

Forwarder 2

State is Active

1 state change, last state change 00:02:55

MAC address is 0007.b400.0a02 (default)

```
  Owner ID is 0012.da88.3560
  Preemption enabled, min delay 30 sec
  Active is local, weighting 100
 Forwarder 3
  State is Listen
  MAC address is 0007.b400.0a03 (learnt)
  Owner ID is 0013.19f8.8be0
  Time to live: 14397.712 sec (maximum 14400 sec)
  Preemption enabled, min delay 30 sec
  Active is 10.1.1.3 (primary), weighting 100 (expires in 7.604 sec)
r2#
```

说明：从上可以看出 R2 的状态为 Listen，因为还有个 R3，虽然优先级相同，但对方 IP 地址比自己大，所以对方状态是 Standby,即为 AVG 的备份。

（3）查看 R3（AVG 的备份）

```
r3#sh glbp
FastEthernet0/0 - Group 10
  State is Standby
   1 state change, last state change 00:03:01
  Virtual IP address is 10.1.1.100
  Hello time 3 sec, hold time 10 sec
   Next hello sent in 1.750 secs
  Redirect time 600 sec, forwarder time-out 14400 sec
```

Preemption enabled, min delay 0 sec

Active is 10.1.1.1, priority 200 (expires in 8.294 sec)

Standby is local

Priority 100 (default)

Weighting 100 (default 100), thresholds: lower 1, upper 100

Load balancing: round-robin

Group members:

0012.da88.3560 (10.1.1.2)

0013.19f8.8be0 (10.1.1.3) local

0013.1a2f.0680 (10.1.1.1)

There are 3 forwarders (1 active)

Forwarder 1

State is Listen

MAC address is 0007.b400.0a01 (learnt)

Owner ID is 0013.1a2f.0680

Time to live: 14398.290 sec (maximum 14400 sec)

Preemption enabled, min delay 30 sec

Active is 10.1.1.1 (primary), weighting 100 (expires in 7.304 sec)

Forwarder 2

State is Listen

MAC address is 0007.b400.0a02 (learnt)

```
  Owner ID is 0012.da88.3560
  Time to live: 14398.690 sec (maximum 14400 sec)
  Preemption enabled, min delay 30 sec
  Active is 10.1.1.2 (primary), weighting 100 (expires in 8.690 sec)
 Forwarder 3
  State is Active
    1 state change, last state change 00:03:09
  MAC address is 0007.b400.0a03 (default)
  Owner ID is 0013.19f8.8be0
  Preemption enabled, min delay 30 sec
  Active is local, weighting 100
r3#
```

3．测试结果

（1）在 R4 上通过 telnet 10.1.1.100，看最终的结果是登陆哪里：

```
r4#telnet 10.1.1.100
Trying 10.1.1.100 ... Open
r1>
```

说明：可以看出，结果是 telnet 上 R1。

（2）查看 10.1.1.100 的 ARP

```
r4#sh arp
```

Protocol Address Age (min) Hardware Addr Type Interface

Internet 10.1.1.4 - 00e0.1e60.7c86 ARPA Ethernet0

Internet 10.1.1.100 0 0007.b400.0a01 ARPA Ethernet0

r4#

说明：可以看出，10.1.1.100 的 MAC 为 0007.b400.0a01，即为 R1（AVG）的 MAC

（3）清除 ARP，再次测试 telnet 10.1.1.100 的结果，因为 GLBP 中有三个成员，所以会负载，这次应该 telnet 到第二台了。

r1#clear arp-cache

r1#

r4#telnet 10.1.1.100

Trying 10.1.1.100 ... Open

r2>

说明：可以看出，已经不再是到 R1，而是到 R2，说明负载成功。

r4#sh arp

Protocol Address Age (min) Hardware Addr Type Interface

Internet 10.1.1.2 0 0012.da88.3560 ARPA Ethernet0

Internet 10.1.1.4 - 00e0.1e60.7c86 ARPA Ethernet0

Internet 10.1.1.100 0 0007.b400.0a02 ARPA Ethernet0

r4#

说明：查看 ARP 表也发现，第二次得到 10.1.1.100 的 MAC 为 0007.b400.0a02，说明 GLBP 中的组成员是为用户分担数据的。

（4）清除 ARP，再测试一次：

r1#clear arp-cache

r1#

r4#telnet 10.1.1.100

Trying 10.1.1.100 ... Open

R3>

说明：看到结果 telnet 到了 R3，说明 GLBP 中负载成功。

4．配置 GLBP 认证

说明：认证由于 IOS 版本的原因，可能支持明文和 MD5 认证，可使用直接输入密码，也可使用 key chain 的方法。

（1）为 R1 配置 GLBP 认证密码：

r1(config-if)#glbp 10 authentication text cisco　密码为 cisco

（2）为 R2 配置 BLGP 认证密码

R2(config-if)#glbp 10 authentication text ccie　密码为 ccie,密码同其它不一样

（3）为 R3 配置 BLGP 认证密码

R3(config-if)#glbp 10 authentication text cisco　密码为 cisco

5.查看 GLBP 认证后的结果

说明：由于 R2 配置的密码与其它成员和 AVG 不一样，查看时会发现 R2 不在此组中。

（1）在 R3 上查看 GLBP 组状态

```
r3#sh glbp
FastEthernet0/0 - Group 10
  State is Standby
    4 state changes, last state change 00:00:42
  Virtual IP address is 10.1.1.100
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 2.595 secs
  Redirect time 600 sec, forwarder time-out 14400 sec
  Authentication text "cisco"
  Preemption enabled, min delay 0 sec
  Active is 10.1.1.1, priority 200 (expires in 8.597 sec)
  Standby is local
  Priority 100 (default)
  Weighting 100 (default 100), thresholds: lower 1, upper 100
  Load balancing: round-robin
  Group members:
    0013.19f8.8be0 (10.1.1.3) local
    0013.1a2f.0680 (10.1.1.1)
  There are 3 forwarders (2 active)
  Forwarder 1
    State is Listen
```

2 state changes, last state change 00:00:52

MAC address is 0007.b400.0a01 (learnt)

Owner ID is 0013.1a2f.0680

Time to live: 14397.055 sec (maximum 14400 sec)

Preemption enabled, min delay 30 sec

Active is 10.1.1.1 (primary), weighting 100 (expires in 7.055 sec)

Forwarder 2

State is Active

1 state change, last state change 00:00:44

MAC address is 0007.b400.0a02 (learnt)

Owner ID is 0012.da88.3560

Time to live: 14311.038 sec (maximum 14359 sec)

Preemption enabled, min delay 30 sec

Active is local, weighting 100

Forwarder 3

State is Active

1 state change, last state change 00:08:12

MAC address is 0007.b400.0a03 (default)

Owner ID is 0013.19f8.8be0

Preemption enabled, min delay 30 sec

Active is local, weighting 100

r3#

SLA （Service Level Agreements）

概述

SLA 这个技术，算是一个非常实用，功能相当广泛的技术，但是 SLA 许多功能，许多结果，需要特定的软件才能读懂，需要特定的人才能分析。那么我们就介绍一些我们能够读懂，我们能够分析的功能。

如果有一个人在上海，他从未去过南京，但是他现在想要测试从上海到南京的路程上要花多少时间，很显然，只要有人亲自从上海去一趟南京，这个时间就能够计算出来，并且，从上海往南京多跑几趟，再算个平均值，那么这样计算出来的时间将更为理想。正好这个想测试从上海到南京之间的路程要花多少时间的人，听说上海有朋友要去南京买东西，这样他就抓住了一个测试的机会，他就去跟朋友说这个事，比如他的朋友 10 点从上海出发，那么这个人记下了出发时间，然后等到他朋友从南京回到上海后，这个人记下回到上海的时间是 15 点，如果这个人把 15 点减去 10，等于 5,然后再除以 2，结果为 2.5，那么他认为从上海到南京的路程要花 2.5 个小时，这样计算的结果是否准确呢。很明显，这样计算出来的路程是很马虎的，也是不可靠的，因为可能他的朋友在南京买东西就花去了一个小时，也就是说他的朋友从上海去南京再回来，路程上花掉的时间是 4 个小时，而不是 5 个小时，中间的路程时间就应该是 4 小时除以 2，结果等于 2 小时，所以按之前的算法算出 2.5 小时是错误的。那么又怎样才能正确计算出路程应该是 2 小时呢，这也很明显，那就是在他朋友 10 点从上海出发时，记下这个出发时间，然后他朋友到达南京时，应该是 12 点，他的朋友也应该把这个时间记下来，等他朋友在南京买到东西，准备离开南京时，应该是 13 点，那么这个时间也应该做记录，最后回到上海是 15 点，再记下这个到达时间，最后得出来的，才是真实的路程时间。所以中间买东西的误差，应该用科学的方法去除。

以上是生活的真实例子，如果拿到网络中，用来测试我们网络上从一个主机到另外一个主机之间的网络速度，又应该是什么样的方法呢，又应该用什么样的方法好呢？比如，要测试从一台主机到目标为 cisco.com 的主机，中间跨越了许多网络设备，我们想要测试从这台主机到 cisco.com 之间网络中的传输该用去多少时间，中间的网络性能到底如何，我们该用什么样的方法呢？假如我们用常用的方法“ping”，我们从这台主机去 ping cisco.com，结果看到花去的时间为 20ms，那么我们认为从

这台主机到 cisco.com 中间的网络需要 20ms 除以 2 等于 10ms，那我们就真的错了，因为我们算出的这 10ms，根本就不是中间网络需要使用的时间，也不能因此而认为网络的性能是差的。也许 cisco.com 这台主机正处于繁忙状态，或者是 CPU 负荷不够，CPU 处理 ICMP 数据需要 10ms，那么这样一来，我们应该计算出，两台主机之间的网络传输数据需要多少时间呢？很明显，应该是来回的时间 20ms 减去 CPU 繁忙时处理 ICMP 的时间 10ms，结果为剩下的 10ms 除以 2,应该为 5ms，所以我们之前并没有算出网络的时间和性能。那么我们在测试网络时间和性能的时候，能不能像上面测试上海到南京路程那样，在分别出发的时候，下记时间，然后让朋友在南京到达和出发时，也记下时间呢。网络设备能不能这样为我们记下时间呢，答案是不能。既然不能，那就说明我们不能准确的计算网络延时，这个时候，如果要让网络设备像我们测试路程记录时间一样来测试网络，就可以利用 SLA 这个技术，SLA 正是在我们测试网络中两个节点之间的数据时，为数据记录下我们需要的重要参数，这就是 SLA 完成的，我们在发数据从设备上发出去，只要我们正确配置 SLA，便可记录参数，但是，数据到达目标之后，目标是否又能协助我们添加重要参数呢？如果需要对方这样做，就必须将对方配置为 SLA responder。需要说明的是，SLA 在测试网络的时候，用到的数据可以由我们任意定义，包括数据的协议，端口，并且这些测试的数据，是程序在后台发起的，不需要我们人工干预，我们需要做的只是定义数据类型，和数据发起时间，以及频率，其它的，交给后台处理。

配置

根据图 1 中的拓朴，来测试 R1 到 R3 之间的网络延迟和性能，按照理论，我们需要在 R1 上配置好 SLA，并且需要将 R3 配置成为 SLA responder,否则结果有如掩耳盗铃。

1. 配置 R1 的 SLA

（1）定义 SLA 的数据类型，发起时间，频率

注：定义会话号码 1：

R1(config)#ip sla monitor 1

（2）定义数据类型为 TCP 23，（其它协议也可用，但需网络允许），源端口可不配：

R1(config-sla-monitor)#type tcpConnect dest-ipaddr 13.1.1.3 dest-port 23 source-ipaddr 13.1.1.1**（3）定义频率为 60 秒一次：**

R1(config-sla-monitor-tcp)#frequency 60

定义会话发起时间为现在，并且在没有人工干预的情况下永不停止：

R1(config)#ip sla monitor schedule 1 start-time now life forever

2.配置目标为 SLA responder

r3(config)#ip sla monitor responder

3．检查测试结果：

（1）查看 R1

R1#sh ip sla monitor statistics

Round trip time (RTT) Index 2

Latest RTT: 8 ms

Latest operation start time: *01:25:07.252 UTC Fri Mar 1 2002

Latest operation return code: OK

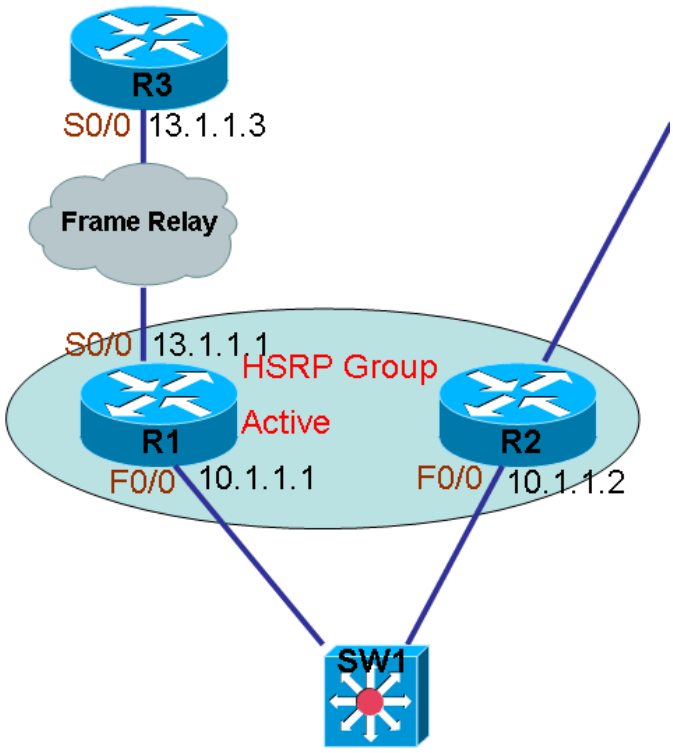
Number of successes: 1

Number of failures: 0

Operation time to live: Forever

R1#

说明：从结果可以看出，测试成功次数为 1 次，并且可以看到时间为 8ms，如果对方不配 SLA，将会失败。



来看图 2，在 LAN 中，R1 和 R2 为 HSRP，为 LAN 中主机提供网关备份，我们配置 R1 为主网关，即 Active,R1 负责将 LAN 中主机的数据包从 S0/0 送出去，如果当 R1 的 S0/0 失效之后，理所当然主网关应该变成 R2，由 R2 为 LAN 内的主机提供数据转发。在通常情况下，我们配置 R1 的 HSRP 时，可以监控 S0/0 的接口状态，当接口 down 掉之后，降低自身优先级而后让出 Active 的角色。虽然我们已经为 HSRP 监控了接口 S0/0 的状态，可是当 S0/0 与对端网络失去连接后，R1 并不会因此而降低优先级来让出 Active 的角色，因为 S0/0 出去是帧中继网络，就算对方接口有什么变化，因为 S0/0 是和帧中继交换机相连的，所以 S0/0 接口没有出现接口 down 的状态，从而导致 R1 的 S0/0 已经不能通信了，但自身还是 HSRP 的 Active 状态，而导致用户不能与外界通信。

最理想的做法就是，R1 不能单纯的只监控 S0/0 接口的状态，不能听认定接口状态 down 了，网络才失去连接，而应该主动去尝试是否可以从 S0/0 发送数据包出去得到远程主机的回应，如果数据包从 S0/0 发出去，并无主机回应，那么就认为 S0/0

已不能与外界通信，这时就应该降低优先级来让出 Active 的角色。那么要怎样才能测试从 S0/0 发数据包出去测试与远程主机的连通性呢，在本图中，我们完全可以利用之前的 SLA，让 SLA 从 R1 向 13.1.1.3 发出 ICMP 数据包，正常情况下是应该得到对方的回应的（必须在对方放开 ICMP 的通信），如果 13.1.1.3 不对 ICMP 作出回应，我们便认为 S0/0 已失去网络连通性。这样的组合，Cisco 称为 Object Tracking,即基于目标的跟踪。

配置

1．配置 Object Tracking：

（1）配置到 13.1.1.3 的 SLA，并利用 ICMP 作为数据协议：

r1(config)#ip sla monitor 5

r1(config-sla-monitor)#type echo protocol ipIcmpEcho 13.1.1.1 source-ipaddr 13.1.1.3

r1(config-sla-monitor)#exit

r1(config)#ip sla monitor schedule 5 start-time now life forever

（2）配置 Track 组:

r1(config)#track 10 rtr 5　　10 为 track 组号码，5 为 SLA 会话号

2．将 Track 组应用到 HSRP：

（1）在接口应用 track 组 10

r1(config)#int f0/0

r1(config-if)#standby 1 track 10

3．检查结果

说明：如果 R1 能 ping 通 13.1.1.3，就说明 Track 组状态是 up 的，那么 HSRP 就不会降低优先级，只要 ping 不通 13.1.1.3，Track 组状态 down 了，HSRP 就会降低优先级成为 Standby 状态。

（1）检查 SLA 状态

r1#sh ip sla monitor statistics

Round trip time (RTT) Index 5

Latest RTT: 35 ms

Latest operation start time: *01:54:30.942 UTC Fri Mar 1 2002

Latest operation return code: OK

Number of successes: 25

Number of failures: 0

Operation time to live: Forever

说明 SLA 正常

（2）检查 Track 组状态

r1#sh track

Track 10

Response Time Reporter 5 state

State is Up

1 change, last change 00:04:04

Latest operation return code: OK

Latest RTT (millisecs) 36

Tracked by:

HSRP FastEthernet0/0 1

r1#

说明：Track 组正常，则 HSRP 角色正常。

（3）检查 HSRP 角色：

```
r1#sh standby
FastEthernet0/0 - Group 1
  State is Active
    4 state changes, last state change 00:00:07
  Virtual IP address is 10.1.1.100
  Active virtual MAC address is 0000.0c07.ac01
    Local virtual MAC address is 0000.0c07.ac01 (v1 default)
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 1.950 secs
  Preemption enabled
  Active router is local
  Standby router is unknown
  Priority 105 (configured 105)
    Track interface Serial0/0 state Up decrement 10
    Track object 10 state Up decrement 10
  IP redundancy name is "hsrp-Fa0/0-1" (default)
r1#
```

说明：HSRP 角色正常。

4．测试 SLA 状态失效后，HSRP 的反应：

（1）测试 R1 到对端 ICMP 断开

r1#sh ip sla monitor statistics

Round trip time (RTT) Index 5

Latest RTT: NoConnection/Busy/Timeout

Latest operation start time: *01:58:50.941 UTC Fri Mar 1 2002

Latest operation return code: Timeout

Number of successes: 50

Number of failures: 1

Operation time to live: Forever

r1#

说明：可以看出 ICMP 失败了

（2）查看 track 组状态：

r1#sh track

Track 10

Response Time Reporter 5 state

State is Down

2 changes, last change 00:00:37

Latest operation return code: Timeout

Tracked by:

HSRP FastEthernet0/0 1

```
r1#
```

说明：由于 SLA 失效，所以 Track 组也 down 了，那么 HSRP 就应该降低优先级而改变角色身份 。

（3）查看 HSRP 角色状态：

```
r1#sh standby
FastEthernet0/0 - Group 1
  State is Standby
    3 state changes, last state change 00:01:18
  Virtual IP address is 10.1.1.100
  Active virtual MAC address is 0000.0c07.ac01
    Local virtual MAC address is 0000.0c07.ac01 (v1 default)
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 2.083 secs
  Preemption enabled
  Active router is 10.1.1.2, priority 100 (expires in 8.065 sec)
  Standby router is local
  Priority 95 (configured 105)
    Track interface Serial0/0 state Up decrement 10
    Track object 10 state Down decrement 10
  IP redundancy name is "hsrp-Fa0/0-1" (default)
r1#
```

说明：可以看出 HSRP 已将优先级从 105 降到 95，从 Active 降到了 Standby 状态。

除加 Track 组额外功能

我们不仅可以在 HSRP 中应用 Track，利用 Track 的状态来做出自己的决定。有时我们可以在其它方面也利用 Track 组的功能，比如我们写路由的时候，某条路由也可以调用 Track 组的状态，当 Track 状态 down 后，相应的路由也消失。下面我们以静态路由为例，来看看当 Track 组状态 down 后，静态路由消息的例子。

1．监控接口的路由能力

说明：监控某个接口是否能够传送数据包，如果没有传送数据包的能力，那么 Track 组状态将 down，也就是我们把接口的 IP 地址去掉，Track 组即 down

（1）配置 Track 接口

r1(config)#track 1 interface s0/0 ip routing

（2）去掉 S0/0 的接口 IP

r1(config)#int s0/0

r1(config-if)#no ip address

（3）查看接口 IP

r1#sh ip int s0/0

Serial0/0 is up, line protocol is up

Internet protocol processing disabled

r1#

（4）查看 Track 组状态：

r1#sh track

Track 1

Interface Serial0/0 ip routing

IP routing is Down (ip disabled, no ip addr)

1 change, last change 00:03:09

说明：从上可以看出，接口没有数据传递功能后，Track 组变成 Down 状态。

如果被静态路由调用，那么相应静态路由将消失。

2 监控接口 line-protocol 状态

说明：如果接口的 line-protocol 为 down，那么 Track 组即 down，只要 line-protocol 是 up 的，不管有没有接口 IP 地址，不管是否有路由能力，track 组的状态都为 up。

（1）配置 track 接口的 line-protocol

r1(config)#track 2 interface s0/0 line-protocol

（2）查看接口信息：

r1#sh ip interface s0/0

Serial0/0 is up, line protocol is up

Internet protocol processing disabled

r1#

说明：可以看出接口 S0/0 line protocol 是 up 的，虽然没有路由能力

（3）查看 Track 组状态：

r1#sh track

Track 2

Interface Serial0/0 line-protocol

Line protocol is Up

1 change, last change 00:01:25

说明：可以看出，Track 组此时只关心接口的 line-protocol 状态，并不状态接口的路由能力。

3 监控路由表中路由条目

说明：监控路由表中是否有某些路由，如果被监控的路由条目在路由表中存在，则 Track 组 UP，被监控路由消失，则 Track 组 down:

（1）写出一条被监控路由作测试

r1(config)#ip route 100.1.1.0 255.255.255.0 s0/0

（2）将 100.1.1.0/24 作为监控目标：

r1(config)#track 3 ip route 100.1.1.0/24 reachability

（3）添加路由

说明：最后写一条静态路由，如果 100.1.1.0 这条路由消失（也就是 Track 组 down），则自己也跟着消失：

r1(config)#ip route 30.1.1.0 255.255.255.0 s0/0 track 3

（4）查看是否路由都存在：

r1#sh ip route

Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP

D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area

N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2

E1 - OSPF external type 1, E2 - OSPF external type 2

i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2

```
        ia - IS-IS inter area, * - candidate default, U - per-user static route
        o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     100.0.0.0/24 is subnetted, 1 subnets
S       100.1.1.0 is directly connected, Serial0/0
     13.0.0.0/24 is subnetted, 1 subnets
C       13.1.1.0 is directly connected, Serial0/0
     30.0.0.0/24 is subnetted, 1 subnets
S       30.1.1.0 is directly connected, Serial0/0
r1#
```

说明：可以看出，自身路由 30.1.1.0/24 是存在的，以及 100.1.1.0/24 也是存在的。

（5）查看 track 组状态

```
r1#sh track
Track 3
  IP route 100.1.1.0 255.255.255.0 reachability
  Reachability is Up (static)
    3 changes, last change 00:00:04
  First-hop interface is Loopback3
  Tracked by:
```

STATIC-IP-ROUTING 0

（5）测试 100.1.1.0/24 消失

r1(config)#no ip route 100.1.1.0 255.255.255.0 s0/0

（6）查看 Track 组状态：

r1#sh track

Track 3

IP route 100.1.1.0 255.255.255.0 reachability

Reachability is Down (no route)

2 changes, last change 00:00:09

First-hop interface is unknown

Tracked by:

STATIC-IP-ROUTING 0

说明：100.1.1.0/24 消失，track 组也消失。

（7）查看路由 30.1.1.0/24

r1#sh ip route

Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP

D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area

N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2

E1 - OSPF external type 1, E2 - OSPF external type 2

i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2

ia - IS-IS inter area, * - candidate default, U - per-user static route

```
        o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     13.0.0.0/24 is subnetted, 1 subnets
C       13.1.1.0 is directly connected, Serial0/0
r1#
```

说明：因为路由 100.1.1.0/24 消失，Track 组也 down，所以 30.1.1.0/24 也从路由表中消失。

NTP（网络时间协议）

概述

某些时候，我们需要在 Cisco 设备上做一些基于时间的策略或访问控制，让这些策略或控制在特定的时间内生效，所以设备上必须存在着准确的时间。但是如果手工给设备配好时间，当设备因为某些原因重启后，时间将被刷新到出厂时的时间，这样就影响到我们所做的策略或服务。这时我们就需要设备能够借助于远程时间服务器上的时间来同步自己的本地时间，让设备在正常工作时，本地的时间和远程时间服务器的时间保持一致。

本地设备的时间和远程时间服务器即使能够同步，也会存在毫秒级的误差，如果自己和远程时间服务器同步，那么别人再和自己同步，就意味着别人的时间误差可能更大。在这里，时间的精准度就会有高低，Cisco 设备的 NTP 把这样的精准度高低称为 stratum，如果 stratum 值越大，就表示精准度越差，stratum 值越小表示精准度就越好。比如远程一台时间服务器的 stratum 是 2，本地设备和它同步后，自己的 stratum 就是 3，精准度就差了一些，如果这时别的设备再和自己同步，那么它得到的 stratum 就是 4，精准度就意味着更差。

Cisco 设备即可以做为 NTP 客户端，即自己和远程时间服务器同步，也可作为

NTP 服务器，即向别的设备提供自己的时间，让别的设备和自己的时间同步，如果将 Cisco 设备作为 NTP 服务器，默认的 stratum 是 8，就表示远程设备和自己同步后，stratum 就是 9。

时间和时区

如果一台 Cisco 设备需要做 NTP 时间服务器，就得先为自己配上时间，但是，因为设备可能出现在全球的任何一个国家，而可能各个国家的时区是不一样的，所以这时还需要为设备配置时区，至于每个国家用哪个时区，不在本篇讨论范围内，请自行查阅各国相关时区和时差，中国使用东 8 区时。

配置

1. 配置时间

（1）为设备配置时区：

R1(config)#clock timezone GMT +8　配置时区为东 8 区时

（2）为设备配置时间：

r1#clock set 20:00:00 1 oct　2008　　配置时间为 2008 年 10 月 1 日 20 点整

注：此时间为东 8 区时 2008 年 10 月 1 日 20 点整，如果将时区更新，设备会自行计算时差将时间调整到对应时区的时间。

2. 查看结果：

（1）r1#show clock

2. 配置 NTP

（1）配置 NTP 服务器：

注：配置 master 和 stratum(默认为 8)

R1(config)#ntp master 5　　stratum 为 5

（2）配置 NTP 数据包的源地址：

注：此地址为数据发出时的源地址，并不影响 NTP 时间同步

R1(config)#ntp source Loopback0

3.配置 NTP Client

（1）指定 NTP 服务器地址

R2(config)# ntp server 10.1.1.1

(2)配置 clock timezone

R2(config)# clock timezone GMT +8

（3）查看结果：

R2#show clock

说明：看本地时间和服务器的时间是否一致。

NTP 认证

服务器和客户端之间可以使用 MD5 来提供安全认证，只有双方在密码相同的情况下，时间才能同步，双方可以同时配置多个 key，号码也可以不一样，但当前使用的的 key 密码必须是相同的，否则时间不能同步。

配置

1．配置 NTP 服务器：

（1）开启认证

R1(config)# ntp authenticate

(2)配置密码

R1(config)# ntp authentication-key 5 md5 cisco

(3)使用某个密码

注：在 key 只有一个的情况下，可以不配

R1(config)#ntp trusted-key 5

2．配置 NTP Client;

(1) 开启认证

R2(config)# ntp authenticate

（2）配置密码

R2(config)# ntp authentication-key 20 md5 cisco

（3）使用某个密码

R2(config)#ntp trusted-key 20

（4）打开对服务器的密码使用，让发送给服务器的数据中携带密码

R2(config)#ntp server 10.1.1.1 key 20

3.查看结果

（1）未同步的：

R2#sh ntp status

Clock is unsynchronized, stratum 16, no reference clock

R2#show ntp association detail

12.0.0.1 configured, insane, invalid, unsynced, stratum 16

（2）已同步的：

R2#sh ntp status

Clock is synchronized, stratum 6, reference is 127.127.7.1

r2#sh ntp associations detail

10.1.1.1 configured, authenticated, our_master, sane, valid, stratum 6

Summer-time 夏令时

概述

众所周知，在冬天，如果某个地方是早上 7 点钟天亮，晚上是 6 点钟天黑，那么到了夏天，这个地方就很可能是早上 5 点钟就天亮了，并且晚上 7 点才天黑，如果人们一般是 7 点半起床，9 点上班的话，那么可以相象，人们绝对可以在夏天的时候 6 点半起床，甚至可以更早，因为天早已经亮了，人们可以大大利用白天的时间。 但是如果因为到了夏天而让人们的学习和工作提前一个小时，可能会让人产生厌恶情绪。因此，就利用调时钟的方法，在夏天到来的时间，将时间往前拨快一小时，也就是时间明明还是早上 5 点，咱们就把时钟调到 6 点，大家到了夏天都遵守这个规则，这样一来，大家夏天就可以每天多利用一小时。但是到了冬天，时钟还是应该回归到夏天之前的时间，也就是时间往后拨慢 1 小时。但是这钟时钟拨快和拨慢，当然并不是人们手工去执行，而只要时钟支持这种夏令时机制，就会自行在相应的季节里调整自己的时间。

那么究竟在什么样的一个时间范围内，才需要让时间调快一小时呢，这段时间范围就称为夏令时范围。中国已经没有这样的机制，而现在 Cisco 所在的美国夏令时范围是从每年 3 月的第二个星期日，到 11 月的第一个星期日。这样的夏令时范围，在 Cisco 设备上，可以随意配置，当时钟进入夏令时范围，时间会马上往前跳过一小时，比如夏令时范围是 3 月 1 日早上 9 点，那么时间到了 3 月 1 日早上 9 点，

下一分钟就不再是 9 点零 1 分，而应该是 10 点零 1 分。在结束的时候，如果是 10 月 1 日早上 9 点结束，那么到了 10 月 1 日早上 9 点，下一分钟就不再是 9 点零 1 分，而应该是 8 点零 1 分。

配置

1.配置夏令时

注：在配置之前，应该配好正确的时区和时间

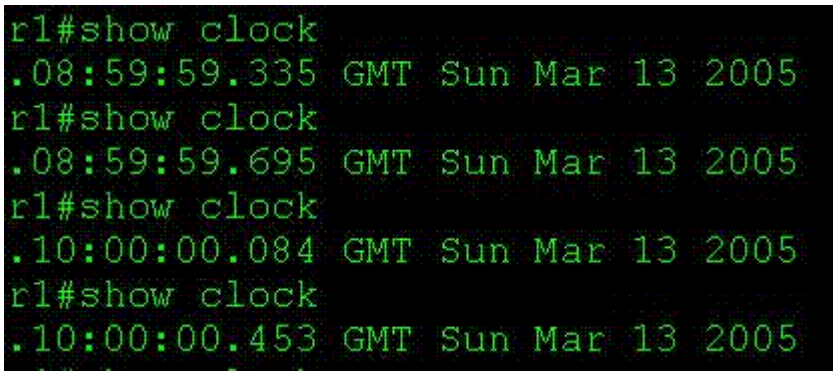
（1）配置夏令时范围

```
r1(config)#clock summer-time GMT recurring 2 sun march 9:00 2 sun nov 9:00
```

说明：夏令时范围从 3 月的第二个周日上午 9:00 到 11 月第二个周日的上午 9:00

2 查看结果：

（1）查看进入夏令时的时间变动



说明：从上面可以看出，时间从 8:59 过了之后，按正常情况应该是 9:00，可是因为受进入夏令时的影响，时间直接加快一小时，跳到了 10:00

（2）查看结束夏令时的时间变动

说明：从上面可以看出，时间从 8:59 过了之后，按正常情况应该是 9:00，可是因为受结束夏令时的影响，时间直接变慢一小时，跳到了 8:00。

Syslog and local logging

Local logging

在 Cisco 设备上往往会发生许多事件，有些是我们预料中的，是我们自己配置的，有些是无法预料的故障。如果当我们使用中的设备出现某些故障时，我们需要知道设备究竟发生了哪些事件，或者曾经发生过哪些事件，以便我们能够排错故障，这时，就需要设备有记录事件的功能。Logging 功能就能够帮助我们记录设备上发生的大大小小的事件。

设备上所发生的事件是有等级之分的，有时我们并不需要设备将任何事件都记录下来，那么我们就可以随意设定哪种级别的事件是我们需要记录的，哪些是不需要记录的。Cisco 将设备上的事件级别分为 0 到 7 总共 8 个级别，分别是 0 级代表最严重的事件，0 (emergencies)， 1(alerts)，2(critical)，3(errors)，4，(warnings) ，5 (notifications)， 6(informational) ，

7 (debugging) ，0 是最严重的，如果我们指定记录 5 级别的，那么 0 到 5 级别的会全部记录。

我们可以将事件记录到设备本地存储器中，称为 buffered，也可以将事件记录到远程服务器中。如果要记录到远程服务器，远程服务器的空间大小在 Cisco 设备上是无法控制的，但是如果将事件记录在设备本地存储器，就可以定义存储器的空

间大小 ，默认为 4096 bytes。存放在本地存储器时，我们查看的时候，事件的排列是由上往下的，上面的日志是最老的，也就是最早发生的，最下面的日志是最新的，如果存储器满了，那么最新的日志将替换最老的日志。

当设备出现故障时，我们需要查找曾经发生的件事和原因，有时要根据事件发生的时间来判断，那么在设备记录事件时，就应该为每条发生的件事打上相应的时间戳。在默认情况下，设备会为事件写上发生的时间，但是这个时间并不是设备本地设置的时间，这样也就不利于我们排错，所以，我们需要手工告诉设备将时间戳写成本机的时间。

在我们登陆到设备上进行配置的时候，如果引起设备发生一些事件，我们希望设备能够给我们一些反馈信息，也就是能跳出一个提示信息。因为我们可以通过 console 登陆设备，也可以 telnet 登陆，但设备默认是 console 登陆设备的是能够看见日志提示的，而 telnet 登陆的是无法看见的，如果要让 telnet 方式登陆也看见日志信息，也需要手工进行配置。

配置

1．配置 logging:

（1）开启 logging 服务（默认是开启的）

r1(config)#logging on

（2）开启本地记录功能

r1(config)#logging buffered

（3）定义本地存储器的空间大小（单位 Byte）

r1(config)#logging buffered 9090

(4)定义可以存储的日志级别（默认 7 debugging）

r1(config)#logging buffered errors 改为记录日志的级别为 3 errors

(5)定义在 console 下面是否弹出日志提示(默认开启)

r1(config)#logging console 定义 console 下弹出日志

（6）定义 telnet 方式登陆设备时也能看到弹出日志（默认关闭）

r1#terminal monitor

（7）定义能够传到 telnet 下的事件等级（默认为 7 debugging）

r1(config)#logging monitor errors 定义能够传到 telnet 下的事件等级为 3 errors

（8）定义时间戳（默认非本地时间）

r1(config)#service timestamps log datetime msec show-timezone localtime

注：在记录时间戳时，精确到毫秒级，并且使用本地时间，附带时区信息。

（9）查看日志：

r1#sh logging

syslog

有时我们需要将设备发生的事件记录到远程服务器上，这时就需要为本地设备定义远程服务器的 IP 地址，并且在远程服务器上，需要知道远程的存储类型，称为 facility，要事先了解清楚，通常我们使用的有 local0 local1 local2 local3 local4 local5 local6 local7 共 8 个类型，默认为 local7。

配置

1．配置远程记录日志

（1）配置远程服务器地址

r1(config)#logging host 10.1.1.1 定义远程服务器 IP 为 10.1.1.1

（2）定义远程存储类型(默认为 local7)

r1(config)#logging facility local6 定义远程存储类型为 local6

（3）定义发送到远程服务器事件等级（默认为 6 informational）

r1(config)#logging trap 7

FTP&TFTP

概述

在很多时候，我们需要从别的地方往 Cisco 设备上传送文件，或者从 Cisco 设备上往别的地方传送文件。在传输文件时，我们可以使用的协议有很多，在这里需要介绍的是两个传输协议，第一个是大家非常熟悉的 FTP，第二个是简单文件传输协议（TFTP）。也许 FTP 在平时用的非常多，但是到了 Cisco 设备上，并不是如此。Cisco 设备可以利用 FTP 协议从别的 FTP 服务器上往本地传送文件，也可以将自身配置成一台 FTP 服务器，为别人提供文件传送。要让 Cisco 设备成为一台 FTP 服务器，共享出自己存储器中的文件让别人拷贝，这需要特定的 IOS 支持，而 Cisco 从远程 FTP 服务器往本地拷贝文件，几乎所有的 IOS 都支持。另外一个 TFTP 协议，也几乎是所有的 IOS 都支持的协议，TFTP 和 FTP 的功能是相同的，但是 TFTP 不需要认证，而 FTP 在传送文件之前，可以采用认证。Cisco 设备可以将自己配置成一台 TFTP 服务器，共享出自己的文件和目录，也可以从远程 TFTP 服务器往本地拷贝文件。

配置

1．配置 FTP Server

注：要配置 Cisco 设备成为一台 FTP 服务器，需要特定 IOS 支持

（1）开启 FTP server 功能：

Router(config)# ftp-server enable

（2）指定 FTP 的共享目录

如：共享出 flash 中某文件

Router(config)# ftp-server topdir flash:abc.bin

2. 配置 FTP Client

说明：此配置目的在于让 Cisco 设备能够从远程 FTP 服务器往本地拷贝文件，即配置用户名和密码

（1）配置远程 FTP 服务器用户名

Router(config)# ip ftp username ccie　　配置远程 FTP 服务器的用户名为 ccie

（2） 配置远程 FTP 服务器密码

Router(config)# ip ftp password cisco　　配置远程 FTP 服务器的用户名为 cisco

（3）配置本地 FTP 源 IP（可选配置）

Router(config)# ip ftp source-interface

3 配置 TFTP

说明：将 Cisco 设备配置成一台 TFTP 服务器，以共享出自己的目录和文件

（1）配置 Cisco 设备为 TFTP 服务器（需要指明文件名）

R1(config)#tftp-server flash:abc.bin

HTTP&HTTPS

概述

我们在对 Cisco 设备进行配置和管理时，可以通过 Consolde 连接上去，也可以通过 VTY 上去，还有 TTY，最后还有 AUX 拨号上去，不管我们通过以上哪种方式连上去管理，我们能看到的都是命令行界面（CLI 界面）。除了以上的管理方式外，我们还可以为设备配置图形界面的管理方式，也就是通常我们用的网页浏览器去连接

设备进行管理，既然要使用网页浏览器来管理设备，那就要使用的 HTTP 协议。

要想使用网页浏览器通过 HTTP 协议对 Cisco 设备进行图形化界面的管理，这时就需要设备开启 HTTP 服务，默认是关闭的。一般情况下，我们使用的 HTTP 都是使用 TCP 端口号 80 进行连接的，这种方式的传输也只是明文的，如果我们需要让数据安全的传输，可以使用安全的 HTTP，即 HTTPS，端口号为 443，这两种 HTTP 方式，Cisco 设备是否完全支持，取决于 IOS 文件。

配置

1 配置 HTTP Server

说明：当路由器配置成 HTTP Server 后，我们就可以通过网页浏览器对设备进行管理。

（1）开启 HTTP Server(默认关闭)

R1(config)# ip http server

（2）改变 HTTP 端口号（默认 80）

R1(config)#secure-server ip http port 8080　端口号改为 8080

（3）设定 HTTP 连接的认证方式（默认为 enable 密码）

注：共支持认证方式为 aaa ， enable(默认) ， local ， tacacs，连接时，只需输入相应的认证用户信息即可

R1(config)# ip http authentication local　将认证方式改为 local（即用户名密码认证）

（4）限制访问

注：通过使用 ACL 来限制特定的 IP 地址能够通过 HTTP 连接

R1(config)# ip http access-class 20

（5）限制最大连接数，默认为 5

R1(config)# ip http max-connections 10

（6）测试连接

如测试路由器的IP地址为 192.168.27.66，则在PC网页浏览器输入 http://192.168.27.66:8080/即可看见如下认证框

输入已配置的用户名和密码即可。

（7）查看状态

R1#sh ip http server all

1. 配置 HTTPS Server

说明：以上是配置 HTTP 的方式，因为 HTTP 使用的是明文不安全的连接方式，我们可以将其改为安全的 HTTP 连接方式，即 HTTPS，在开启 HTTPS 时，必须使用 no ip http server 来关闭 HTTP，才能开启 HTTPS

（1）开启 HTTPS Server

```
R1(config)#ip http secure-server
```

其它配置基本和 HTTP 一致，不再重复。

SNMP

概述

当网络到一定规模的时候，对于网络管理员来说，要管理好这个网络，如果纯粹靠着命令行的方式去连接网络设备，去查看设备当前的状态，查看自己需要的信息，这些将变的烦琐而难以实现。在这样的环境下，很多人都在试图寻找一个网络管理软件，如果这个软件能够通过图形化的界面对设备状态进行监测，能够利用图形界面看到当前设备的各种参数，能够随时看到设备上发生的各种事件，这将大大改善管理员的工作效率。其实，这样的网络管理软件已经存在，在 Cisco 厂商主推荐的有 CiscoWorks，CiscoWorks 含有多个版本，既然一个管理软件能够图形化显示设备上的各种软硬件信息，那么在软件与设备之间就必须存在着一种交流，只有设备将自身的各种软硬件信息和状态发送给管理软件，那么管理员才能通过图形化界面看到这一切。在担任管理软件与设备之间交流的协议也就 SNMP（简单网络管理协议）。

从上面可以看出，SNMP 协议为两个组件在服务，这两个组件就是我们的网络设备和我们的网络管理软件，SNMP 正是它们之间的桥梁。要让管理软件成功的利用图形界面为我们显示出设备的一切信息，就必须正确的理解和正确的配置 SNMP。

SNMP 工作在网络设备和网络管理软件这两个组件之间，SNMP 将设备上的相关信息通告给管理软件，在配置 SNMP 时，配置了 SNMP 的网络设备被称为 SNMP 代理，而装了管理软件的主机被称为 SNMP 管理，也就是 SNMP 代理将自己的状态信息发送给 SNMP 管理，SNMP 将其整理后，通过图形界面汇报给用户，并且 SNMP 管理上的相关软件被称为 NMS（网络管理系统）。

在 SNMP 代理与 SNMP 管理之间，SNMP 代理使用 UDP 162,SNMP 管理使用 UDP 161。一个完整的 SNMP 共包含三个组件，除了以上说的 SNMP 代理和 SNMP 管理两个组件之外，还有一个组件就是 MIB，这个 MIB 的具体内容就是，之所有 SNMP 管理能够图形化显示设备的一切，正是因为收到了 SNMP 代理发来的相关数据，而 SNMP 代理的所有硬件信息和软件信息，就全部存储在 MIB 里面。SNMP 代理需要将 MIB 的内容发送给 SNMP 管理，才能对用户真正有用。

SNMP 代理将 MIB 发送给 NMS，可以使用两种数据包，这两种数据包分别是 trap 和 informs。

他们之间的区别是，当设备上发生了各种事件之后，产生的 MIB 信息由 SNMP 代理主动向 NMS 发送，是不需要 NMS 请求的，并且在将 MIB 信息发送比值 NMS 之后，NMS 即使收到了，也不需要向 SNMP 代理发送确认消息，也就是说 SNMP 代理发出去的 trap 是根本就不知道 NMS 是否收到，而自己发送完毕之后，这些 MIB 信息会马上删除，不会驻留在内存里面。

而 informs 和 trap 正好恰恰相反，当设备发生事件后，这些信息并不会主动向 NMS 通告，除非 NMS 发送 requst 来查询，然后才会发出去，但是这些已经发出去的 informs 在发送之后是会保留在内存里面的，当 NMS 收到 informs 之后必须向 SNMP 代理发送确认消息，如果不发送确认消息，SNMP 代理会重复多次发送 informs。从上可以看出 informs 具有可靠性。

网络管理员不仅可以通过 NMS 来查询设备上的各种信息，而且还可以通过 NMS 来更改设备上的配置，如果是要查询设备信息，发送的数据包被称为 get，如果是要更改设备的配置，被称为 set。

SNMP 版本

SNMP 协议目前为止分为 3 个版本，分别是 version 1，version 2c，version3。

因为网络设备上配置 SNMP 之后，就可以利用 NMS 软件对设备进行查看和修改配置，但是为了安全性，所以在 SNMP 代理与 SNMP 管理之间必须存在着某种安全机制，这些安全机制，在各个版本上的实现是不一样的。

版本 1：版本 1 在 SNMP 代理与 SNMP 管理之间提供的安全机制是使用密码的方式，只有拥有相应密码的 NMS，才能够对 SNMP 代理进行操作，这种密码也分了级别，可以分别定义相应密码是否具有读权限或者是否同时具有读和写的权限。这样的密码被称为 community。版本 1 不仅提供密码认证的方式，除此之外，还可以利用 ACL 的方式来限制只有某些主机能够对其进行访问，也就是说即使主机提供了相应的密码，同样不能够对设备进行操作，还必须 IP 地址是被允许的。

版本 2c：版本 2c 的认证方式和版本 1 是一样的。

版本 3：版本 3 提供的认证方式是使用用户名和密码的方式，并且密码可以是 MD5 加密的。

如果要让 SNMP 代理和 SNMP 管理正常的协同工作，必须将双方的版本配置一致。需要注意的是，一台已配置 SNMP 的设备，可以同时允许多个 SNMP 管理对其进行操作，那么这时就可以在设备上分别为每个 SNMP 管理配置各自的 SNMP 版本。

MIB

设备的所有信息都包含在 MIB 中，接口信息在 MIB 中共分三类：

1：ifIndex(ifEntry 1) （接口索引），是接口在 MIB 中的区分唯一性的方法，这个值是大于 0 的。查看可使用命令：show snmp mib ifmib ifindex

2：ifAlias(ifXEntry 18) （接口别名）,是用户给接口定义的名称，方便识别，在接口下 description 定义。 snmp ifmib ifalias long 命令用来扩展最长可用字符为 256（默认 64）。

别名是在 show interfaces 时才看到。

3：ifName (ifXEntry 1)（接口名），即接口的原名，

所有信息在 MIB 中存储时，以词汇表的顺序编排，（即字典上 A-Z）

所有软硬件在 MIB 库里，都有与自己相关联的标识，而这些标识，并不是永久不变的，有的可能因为重启后，就改变了，也可以通过配置使其固定不变。方法可以使用命令

snmp mib persist ，如果要保存，使用 write mib-data 写入 NVRAM。

让接口在 MIB 里固定使用某个信息，全局命令 snmp mib persist circuit 来完成。

Event MIB（MIB 事件）

MIB 认为设备上的软硬件在多大的范围内变化可被认为是事件发生，这些采集方法可分为三种：

Absolute Sampling ，绝对采集也就是定义绝对变化多少之后 ，认为是事件

Delta Sampling 相对采集 定义从多少到多少这样一个变化范围，认为是事件

Changed Sampling

变化采集，只要变化便认为是事件，多用于硬件状态。

这些参数，对配置 RMON 时有用。

配置

注：没有一条特定的开启 SNMP 的命令，全部以 snmp-server 打头的命令便开启了 SNMP。

1.配置 SNMP v1 和 V2

（1）配置 ACL，通过此 ACL，可以限制相应的 IP 才能访问 SNMP 代理：

r1(config)#access-list 10 permit 10.1.1.2　　定义源 IP 为 10.1.1.2 的才能访问

（2）配置密码（即 community）

说明：这个密码必须定义，并且需要说明通过这个密码，能够执行什么样的操作，操作分为读和读写。并且后面可以跟上 ACL 作源 IP 限制。

r1(config)#snmp-server community cisco rw 10

注：源 IP 为 ACL 10 的主机通过密码 cisco 具有读写权限。

（3）配置 trap（也可用 informs 代理）

说明：设备在默认情况下，许多事件是不进行记录的，如果要记录，就需要打开相应的事件记录功能。

r1(config)#snmp-server enable traps bgp

注：配置为记录 BGP 的事件，如果 traps 后面不跟参数，就为开启所有事件记录功能。

（4）配置对 SNMP 管理的 trap（须已配置 trap，否则无效）

说明：当配置完 trap 之后，相应的事件并不会对 SNMP 管理进行反馈，还必须手工指明对哪些主机进行反馈什么样的事件。在这里，对 SNMP 管理反馈的事件必须全局已配，而且还可以在此配置对相应主机要求的密码，SNMP 版本等信息。

r1(config)#snmp-server host 10.1.1.100 version 2c cisco bgp

注：配置主机 10.1.1.100 使用密码 cisco，并且向主机发送 BGP 事件信息，而且 SNMP 版本为 2c。

某些 trap 不需要开，因为本身就是开启的，如果需要开启对接口状态的监控（即 up 和 down 状态的监控），需要使用如下命令：

r1(config)#snmp trap link-status （此命令因 IOS 不同而命令有所不同）

（5）配置关闭系统功能

说明：通过 NMS 默认情况下不能对设备进行重启操作，如需开启重启的权限，需要额外配置：

r1(config)#snmp-server system-shutdown

（6）配置接口与 MIB 绑定：

r1(config)#snmp-server ifindex persist

（7）配置 MIB 所有信息均绑定：

r1(config)#snmp mib persist

r1#write mib-data

说明：虽然没有一条特定的命令来开启 SNMP 服务，但有特定的命令可以关闭所有 SNMP 服务：

r1(config)#no snmp-server

2.查看信息：

（1）查看 SNMP 总信息：

r1#sh snmp

（2）查看与 SNMP 管理之间的会话

r1#sh snmp sessions

（3）查看 MIB 所有软硬件信息

r1#sh snmp mib：

（4）查看所有 MIB 中接口的信息（此信息对 RMON 有用）

r1#sh snmp mib ifmib ifindex

RMON（远程监视）

概述

功能有点像 SNMP，但可以用来增强 SNMP，对 SNMP MIB 中各种信息的变量进行轮询（查询），查询相关事件的变量分为上升门限和下降门限，上升门限即超过一定的值，下降门限即降到一定的值。当 MIB 变量达到我们预期的效果后，就会发送 SNMP trap。

RMON 的组：

警报组

警报组(alarm)是 RMON 中的第 3 组，以指定的时间间隔监控一个特定的 MIB(Management

Information Base)对象，当这个 MIB 对象的值超过一个设定的上限值或低于一个设定的下限值

时，会触发一个警报。警报被当作事件来处理，处理事件的方式可以是记录日志或发送 SNMP Trap 的方式。

事件组

事件组(event)是 RMON 中的第 9 组，决定当由于警报而产生事件时，处理行为是产生一个日志. 记录表项或者一个 SNMP Trap。

取样时，对结果可以用两种方式来判断，分为 delta 和 absolute，关键字 delta 表示取样的值在 MIB 变量中两次取样间值的变化，而关键字 absolute 表示直接使扩 MIB 变量的值作为取样值也相当于平均值。

当配置 RMON 时，需要指定以下一些参数：

log ：(可选)输入这个关键值，则警报产生时，会将这个事件记录到日志中

trap：(可选)输入这个关键值，则警报产生时，会产生一个 SNMP Trap。

community：(可选)发送 Trap 时使用的认证名。

descrīption string：(可选)对这个事件的描述。

sowner string：(可选)标志这个事件的拥有者。

需要说明的是，在配置 RMON 对目标进行监控时，输入目标的标识，这个目标的标识是在 SNMP MIB 中的标识符，也称为 OID，一般 SNMP MIB 里面的标识是我们看不懂的，如果需要理解其含义，可查询 CISCO OID 工具，地址为：

http://tools.cisco.com/Support/SNMP/do/BrowseOID.do?local=en

配置

1. 配置 SNMP（此步略过，详细见前面 SNMP 部分）

2. 配置 RMON

（1）例配置对接口 f0/0 的监控，

r1(config)#rmon alarm 1 ifEntry.1.1 2 absolute rising-threshold 2 falling-threshold 1 owner ccie

说明：配置接口 f0/0(OID: ifEntry.1.1 2,各目标的 OID 请自行查阅)，采样时间为 2 秒，当阀值上升到 2 或降到 1 的时候都产生警报，并注明了标识符为 ccie。

（2）并配置事件发生后通告的内容

r1(config)#rmon event 1 trap cisco description "intover" owner ccie

说明：community 为之前 SNMP 的配置，intover 表示消息，owner 可选标识。

3．查看结果：

（1）查看产生过的警报

r1#show rmon alarms

（2）查看产生过的事件

r1#show rmon events

Embedded Event Manager (EEM)

之前的任何一种网络管理技术，如 SNMP，RMON，在检测到事件发生后，并不能解决问题，这些传统的网管技术只有监测功能，却没有解决故障的功能，因此，为了更有效的管理网络，能够在事件发生时，便采用有效的动作来杜绝网络问题，Cisco 推出更进一步的网管技术—Embedded Event Manager (EEM)。

EEM 在正常工作时，能够定期监视指定的事件，当被监测的事件发生后，EEM 可以产生指定的信息或指定的动作。EEM 如何检测指定的事件，需要指定相应的监测方法和监测标准，当事件发生后，需要产生的信息或执行的动作也需要定义，这

一系列的事件和事件发生后需要执行的动作集合起来称为 EEM policy ；由于 EEM 工作复杂，所以 EEM 需要根据不同的分工定义不同的组件，EEM 共有如下几个组件：

EEM server

相当于 EEM 主程序。

Core Event Publishers (Event Detectors)

也就是 EEM 用于检测事件的组件，负责检测各种定义好的事件，事件的检测可以基于其它网管技术，Event Detectors 会在事件发生时向 Server 报告。

Event Subscribers (Policies)

当 Event Detector 检测到指定的事件发生后，Event Subscribers 便执行指定的动作，动作包括产生特定的消息，或执行特定的命令。

EEM 可以单独使用，也可以和其它网管技术配合使用,在配置 EEM 时，就是配置 EEM Policy，因为 Policy 就是事件和事件发生后需要执行的动作集合，配置 Policy，分两种方式：

Applet

Tool Command Language (Tcl)

其中 Applet 是使用 IOS 的 CLI 来配置的，操作相对简单，而 Tool Command Language (Tcl)是一种编程所使用的脚本工具，比较专业，需要使用外置的第三方 ASCII editor 才能编辑和配置，所以，在理论上，单纯只学习 Cisco 课程应该没有能力编写 ASCII editor 的，CCIE 考试中，目前推算几乎不太可能会考到，而 Applet 却直接就能在设备上进行配置。

由于 Cisco 的 IOS 版本众多，所以 EEM 的版本也是相当多，对于事件的检测和能够执行的动作，会因为 EEM 版本的不同而有所不同，基本上新版本会包含老版本的功

能，对于各个 EEM 版本所支持的检测方法和执行的动作只作统一列举，而不一一列举，目前所有 EEM 的版本和对应的 IOS 版本信息如下：

EEM 1.0

支持的 IOS ： 12.0(26)S 、 12.3(4)T 以及后续版本。

EEM 2.0

支持的 IOS ： 12.2(25)S 以及后续版本。

EEM 2.1

支持的 IOS ： 12.3(14)T, 12.2(18)SXF5, 12.2(28)SB, 12.2(33)SRA 以及后续版本。

EEM 2.2

支持的 IOS ： 12.4(2)T, 12.2(31)SB3, 12.2(33)SRB 以及后续版本。

EEM 2.3

支持的 IOS ： Catalyst 6500 交换机上 12.2(33)SXH 以及后续版本。

EEM 2.4

支持的 IOS ： 12.4(20)T, 12.2(33)SXI, 12.2(33)SRE 以及后续版本。

EEM 3.0

支持的 IOS ： 12.4(22)T, 12.2(33)SRE 以及后续版本。

EEM 3.1

支持的 IOS ： 15.0(1)M 以及后续版本。

对于事件的检测，总体上支持如下一些方式，具体哪个版本支持哪个方式，请以实际 IOS 为准，不在该文档中详细说明，事件检测方式如下：

Application-Specific

CLI

Counter

Custom CLI

Enhanced Object Tracking

GOLD

Interface Counter

IPSLA

NF

None

OIR

Resource

RF

Routing

RPC

SNMP

SNMP Notification

SNMP Object

Syslog

System Manager

Timer

IOSWDSysMon (Cisco IOS watchdog)

WDSysMon (Cisco IOS Software Modularity watchdog)

当事件发生后，能够执行的动作如下：

Execute a CLI command

Generate a CNS event

Generate a prioritized syslog message

Generate an SNMP trap

Manually run an EEM policy

Publish an application-specific event

Read the state of a tracked object

Reload the Cisco IOS software

Request system information

Send a short e-mail

Set or modify a named counter

Set the state of a tracked object

Switch to a secondary RP

对于配置 Policy，只对 Applet 做出介绍，由于 Tool Command Language (Tcl)已经超过范围，不再讨论。

EEM Applet

在配置 Applet 时，共有 3 种配置状态：Event， Action， Set。

Event 用于定义事件标准，当指定的要求发生或阀值触发时，则表示该事件产生。

Action 当事件发生后执行的动作。

Set 是设置变量的，目前只支持_exit_。

在 EEM Applet 配置中，一个 Policy 只支持一个 event，也就是一个 Policy 只能检测一个事件，如果退出 Policy 配置时并没有 event，则会有警告，表示 Applet 没有注册成功；但如果没有 action，事件照样被检测，只是事件发生后不会执行任何动作，一个 Policy 中可以配置多个 action。

注：如果修改配置，在没有退出配置模式前，是不会生效的。

在配置 Applet 时，每一个 action 动作是有标签的，多个 action 将由标签的顺序来执行，标签可以是字母，也可以是数字，如果是数字，需要写成如 01.0,，02.0 等等，或 1.0，2.0。

配置 EEM

说明： 配置中共包含

1.配置 EEM 监测内存使用率

2.配置 EEM 监测 Enhanced Object Tracking 状态

3.配置 EEM 监测 CPU 利用率

4.配置 EEM 在事件触发时发送简短 E-mail

1.配置 EEM 监测内存使用率:

（1）查看当前内存情况：

```
Router#show processes memory

Processor Pool Total:   30623072 Used:   17889156 Free:   12733916

      I/O Pool Total:    6291456 Used:    4429312 Free:    1862144


 PID TTY  Allocated      Freed    Holding    Getbufs    Retbufs Process
```

```
   0    0    35624856    13980528    18744396         608          78
*Init*

   0    0       12128      122652       12128           0           0
*Sched*

   0    0      248956      895688         564           1           0
*Dead*


Router#
```

说明：从结果中看出，内存总大小为 30623072，空闲大小为 12733916。

（2）查看当前路由协议状态：

```
Router#sh ip protocols


Router#
```

说明：路由器当前没有配置任何路由协议。

（3）配置 EEM 监测内存使用率：

```
Router(config)#event manager applet MEM

Router(config-applet)#event snmp oid 1.3.6.1.4.1.9.9.48.1.1.1.6.1
get-type exact entry-op lt entry-val 30623072 poll-interval 90

Router(config-applet)#action 01.0 cli command "enable"

Router(config-applet)#action 02.0 cli command "conf t"

Router(config-applet)#action 03.0 cli command "router eigrp 100"
```

```
Router(config-applet)#exit
```

说明：EEM 当前监测内存的使用情况，如果空闲大小低于 30623072，则事件被触发，采集间隔为 90 秒一次，如果事件触发后，执行的第一个动作为在命令行下输入命令 enable，执行的第二个动作为在命令行下输入命令 conf t，执行的第三个动作为在命令行下输入命令 router eigrp 100，其实结果就是在事件发生后，自动启用一个 EIGRP 进程，AS 号为 100；结合之前可以得知，内存总大小为 30623072，所以内存空闲空间肯定会小于 30623072，那么该 EEM policy 配置后，事件肯定是被触发的。其中动作标签为 01.0 格式。

（4）查看 EEM Policy 注册情况：

```
Router#show event manager policy registered

No.  Class   Type    Event Type          Trap  Time
Registered           Name

1    applet  system  snmp                Off   Fri Mar 1 00:10:53
2002   MEM

 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1} get_type exact entry_op lt entry_val
{30623072} poll_interval 90.000

 action 01.0 cli command "enable"

 action 02.0 cli command "conf t"

 action 03.0 cli command "router eigrp 100"


Router#
```

说明：正常工作的 EEM Policy 名字为 MEM，并且其它详细信息也能看见。

（5）查看当前内存使用情况：

```
Router#show processes memory

Processor Pool Total:   30623072 Used:   18103504 Free:   12519568
      I/O Pool Total:    6291456 Used:    4429312 Free:    1862144

 PID TTY  Allocated      Freed    Holding    Getbufs    Retbufs Process
   0   0   35624856   13980528   18744396        608         78
*Init*
   0   0      12128     155784      12128          0          0
*Sched*

Router#
```

说明：当前内存空闲大小为 12519568，小于事件定义的 30623072，所以事件肯定已经触发。

（6）再次查看路由协议情况：

```
Router#sh ip protocols

Routing Protocol is "eigrp 100"

  Outgoing update filter list for all interfaces is not set

  Incoming update filter list for all interfaces is not set

  Default networks flagged in outgoing updates

  Default networks accepted from incoming updates

  EIGRP metric weight K1=1, K2=0, K3=1, K4=0, K5=0
```

```
  EIGRP maximum hopcount 100

  EIGRP maximum metric variance 1

  Redistributing: eigrp 100

  EIGRP NSF-aware route hold timer is 240s

  Automatic network summarization is in effect

  Maximum path: 4

  Routing for Networks:

  Routing Information Sources:

    Gateway         Distance      Last Update

  Distance: internal 90 external 170


Router#

Router#
```

说明：由于 EEM 事件被触发，所以自动启用一个 EIGRP 进程，AS 号为 100。

（7）查看 EEM 事件记录：

```
Router#show event manager history events detailed

No.   Time of Event             Event Type          Name

1     Fri Mar 1  00:14:53 2002  snmp                applet: MEM

 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1}

 val {12725680}

2     Fri Mar 1  00:16:23 2002  snmp                applet: MEM
```

```
 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1}

 val {12532448}

Router#
```

说明：结果显示了相应的 EEM 事件触发了两次。

2:配置 EEM 监测 Enhanced Object Tracking 状态

（1）开启接口 F0/0：

```
Router(config)#int f0/0

Router(config-if)#no shutdown

Router(config-if)#exi

Router(config)#exi


Router#sh interfaces f0/0

FastEthernet0/0 is up, line protocol is up

  Hardware is Gt96k FE, address is c000.03d0.0000 (bia c000.03d0.0000)
```

说明：接口 F0/0 处于双 up 状态。

（2）配置 Enhanced Object Tracking，跟踪接口 F0/0 的状态：

```
Router(config)#track 1 interface f0/0 line-protocol
Router(config-track)#exit
Router(config)#exit
```

```
Router#show track
Track 1
  Interface FastEthernet0/0 line-protocol
  Line protocol is Up
    1 change, last change 00:00:06
Router#
```

说明：当前 Enhanced Object Tracking 跟踪接口 F0/0 的 line-protocol 状态，由于接口 F0/0 目前为双 up 状态，所以 Enhanced Object Tracking 的状态也为 up。

（3）配置 EEM 监测 Enhanced Object Tracking 的状态：

```
Router(config)#event manager applet EOT
Router(config-applet)#event track 1 state down
Router(config-applet)#action a cli command "enable"
Router(config-applet)#action b cli command "conf t"
Router(config-applet)#action c cli command "router ospf 100"
Router(config-applet)#exit
```

说明：EEM 监测 track 1 的状态，如果为 down，则事件触发，并且在事件触发后，自动配置 ospf，进程为 100。其中动作标签为字母格式。

（4）查看 EEM Policy 注册情况：

```
Router#sh event manager policy registered

No.  Class   Type    Event Type          Trap  Time
Registered          Name

1    applet  system  snmp                Off   Fri Mar 1 00:13:23
2002   MEM

 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1} get_type exact entry_op lt entry_val
{30623072} poll_interval 90.000

 action 01.0 cli command "enable"

 action 02.0 cli command "conf t"

 action 03.0 cli command "router eigrp 100"


2    applet  system  track               Off   Fri Mar 1 00:22:49
2002   EOT

 track 1 state down

 action a cli command "enable"

 action b cli command "conf t"

 action c cli command "router ospf 100"
```

```
Router#
```

说明：显示了正常工作的 EEM Policy，包含之前的 MEM 和现在的 EOT。

（5）将 Enhanced Object Tracking 的状态变为 down：

```
Router(config)#int f0/0
Router(config-if)#shutdown
Router(config-if)#exit
```

```
Router#show track
Track 1
  Interface FastEthernet0/0 line-protocol
  Line protocol is Down (hw admin-down)
    2 changes, last change 00:00:08
  Tracked by:
    EEM applet EOT
Router#
```

说明：由于接口 F0/0 被关闭，所以 Enhanced Object Tracking 的状态变为 down，也显示了当前 Enhanced Object Tracking 的状态正被 EEM 所监测。

（6）再次查看路由器上的路由协议：

```
Router#sh ip protocols summary

Index Process Name

0     connected

1     static

2     eigrp 100

3     ospf 100

Router#
```

说明：由于 Enhanced Object Tracking 的状态变为 down，EEM 事件被触发，所以自动配置了 OSPF，进程号 100。

（7）查看 EEM 事件记录：

```
Router#show event manager history events

No.   Time of Event             Event Type          Name

1     Fri Mar 1  00:14:53 2002  snmp                applet: MEM

2     Fri Mar 1  00:16:23 2002  snmp                applet: MEM

3     Fri Mar 1  00:17:53 2002  snmp                applet: MEM

4     Fri Mar 1  00:19:23 2002  snmp                applet: MEM

5     Fri Mar 1  00:20:53 2002  snmp                applet: MEM

6     Fri Mar 1  00:22:23 2002  snmp                applet: MEM

7     Fri Mar 1  00:23:53 2002  snmp                applet: MEM

8     Fri Mar 1  00:24:28 2002  track               applet: EOT

Router#
```

说明：结果显示了 Enhanced Object Tracking 引起的 EEM 事件。

3.配置 EEM 监测 CPU 利用率

（1）查看当前 CPU 利用率：

```
Router#show processes cpu

CPU utilization for five seconds: 8%/0%; one minute: 4%; five minutes: 4%

 PID Runtime(ms)    Invoked      uSecs   5Sec   1Min   5Min TTY Process

   1           4         47         85  0.00%  0.00%  0.00%   0 Chunk
Manager

   2          12        312         38  0.08%  0.01%  0.00%   0 Load
Meter

   3       31580       4185       7545  8.35%  3.53%  3.35%   0
Exec


Router#
```

说明：当前 CPU 利用率大于 4%。

（2）配置 EEM 监测 CPU 使用率：

```
Router(config)#event manager applet CPU

Router(config-applet)#event snmp oid 1.3.6.1.4.1.9.9.109.1.1.1.1.3.1
get-type exact entry-op ge entry-val 1 poll-interval 10
```

```
Router(config-applet)#action 1.0 syslog msg "CPU Over"

Router(config-applet)#exit
```

说明：配置 EEM 监测 CPU 的使用率，每 10 秒种采集一次，如果使用率超过 1%，则事件被触发，当事件触发后，自动产生 syslog 消息”CPU Over”，CPU 使用率肯定超过了 1%，所以事件已经触发。

（3）开启 syslog 功能：

```
Router(config)#logging on

Router(config)#logging buffered 10000 7
```

说明：开启 syslog 缓存信息功能。

（4）查看 EEM Policy 注册情况：

```
Router#show event manager policy registered

No.  Class   Type    Event Type          Trap  Time
Registered          Name

1    applet  system  snmp                Off   Fri Mar 1 00:13:23
2002   MEM

 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1} get_type exact entry_op lt entry_val
{30623072} poll_interval 90.000

 action 01.0 cli command "enable"

 action 02.0 cli command "conf t"

 action 03.0 cli command "router eigrp 100"
```

```
2    applet  system  track              Off   Fri Mar 1 00:22:49
2002   EOT

 track 1 state down

 action a cli command "enable"

 action b cli command "conf t"

 action c cli command "router ospf 100"



3    applet  system  snmp               Off   Fri Mar 1 00:40:12
2002   CPU

 oid {1.3.6.1.4.1.9.9.109.1.1.1.1.3.1} get_type exact entry_op ge
entry_val {1} poll_interval 10.000

 action 1.0 syslog msg "CPU Over"



Router#
```

说明：显示了正常工作的 EEM Policy，包含之前的 MEM，EOT 和现在的 CPU。

（5）查看 syslog 状态:

```
Router#sh logging

Syslog logging: enabled (11 messages dropped, 0 messages rate-limited,

                0 flushes, 0 overruns, xml disabled, filtering disabled)

    Console logging: level debugging, 78 messages logged, xml disabled,

                     filtering disabled
```

```
    Monitor logging: level debugging, 0 messages logged, xml disabled,
                     filtering disabled
    Buffer logging: level debugging, 2 messages logged, xml disabled,
                    filtering disabled
    Logging Exception size (4096 bytes)
    Count and timestamp logging messages: disabled

No active filter modules.

ESM: 0 messages dropped

    Trap logging: level informational, 83 message lines logged

Log Buffer (10000 bytes):

*Mar  1 00:40:32.155: %HA_EM-6-LOG: CPU: CPU Over
*Mar  1 00:40:39.395: %SYS-5-CONFIG_I: Configured from console by console
Router#
```

说明：因为 CPU 使用率超过 1%，所以自动产生了 syslog 信息” CPU Over”。

（6）查看 EEM 事件记录：

```
Router#

Router#show event manager history events detailed

No.   Time of Event             Event Type          Name

1     Fri Mar 1  00:34:23 2002  snmp                applet: MEM

 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1}

 val {12133372}

2     Fri Mar 1  00:35:53 2002  snmp                applet: MEM

 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1}

 val {12013372}

3     Fri Mar 1  00:37:23 2002  snmp                applet: MEM

 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1}

 val {12026628}

4     Fri Mar 1  00:38:53 2002  snmp                applet: MEM

 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1}

 val {12020216}

5     Fri Mar 1  00:40:22 2002  snmp                applet: CPU

 oid {1.3.6.1.4.1.9.9.109.1.1.1.1.3.1}

 val {11}

6     Fri Mar 1  00:40:23 2002  snmp                applet: MEM

 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1}
```

```
 val {12014284}
7    Fri Mar 1  00:40:32 2002  snmp                applet: CPU
 oid {1.3.6.1.4.1.9.9.109.1.1.1.1.3.1}
 val {1}
8    Fri Mar 1  00:40:52 2002  snmp                applet: CPU
 oid {1.3.6.1.4.1.9.9.109.1.1.1.1.3.1}
 val {1}
Router#
Router#
```

说明：结果显示了 CPU 利用率触发的事件。

4.配置 EEM 在事件触发时发送简短 E-mail

（1）配置 EEM 在 CPU 利用率超过阀值时发送简短 E-mail:

```
Router(config)#event manager applet EMAIL
Router(config-applet)#event snmp oid 1.3.6.1.4.1.9.9.109.1.1.1.1.3.1
get-type exact entry-op ge entry-val 75 poll-interval 10
Router(config-applet)#action 1.0 cli command "enable"
Router(config-applet)#action 2.0 cli command "show process cpu"
```

```
Router(config-applet)#action 3.0 mail server "192.168.1.146" to
"engineer@cisco.com" from "eem@cisco.com" subject "CPU Alert" body "CPU
Alert 75"
```

说明：EEM Policy 定义了 CPU 利用率在超过 75%时，将命令 show process cpu 的结果发送到邮箱地址"engineer@cisco.com" 邮箱源地址为"eem@cisco.com" ，邮件主题为 "CPU Alert"，正文同时添加"CPU Alert 75"。

（2）查看 EEM Policy 注册情况:

```
Router#show event manager policy registered

No.  Class   Type    Event Type          Trap  Time
Registered           Name

1    applet  system  snmp                Off   Fri Mar 1 00:13:23
2002   MEM

 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1} get_type exact entry_op lt entry_val
{30623072} poll_interval 90.000

 action 01.0 cli command "enable"

 action 02.0 cli command "conf t"

 action 03.0 cli command "router eigrp 100"


2    applet  system  track               Off   Fri Mar 1 00:22:49
2002   EOT

 track 1 state down

 action a cli command "enable"

 action b cli command "conf t"

 action c cli command "router ospf 100"
```

3 applet system snmp Off Fri Mar 1 00:40:12 2002 CPU

oid {1.3.6.1.4.1.9.9.109.1.1.1.1.3.1} get_type exact entry_op ge entry_val {1} poll_interval 10.000

action 1.0 syslog msg "CPU Over"

4 applet system snmp Off Fri Mar 1 00:53:37 2002 EMAIL

oid {1.3.6.1.4.1.9.9.109.1.1.1.1.3.1} get_type exact entry_op ge entry_val {75} poll_interval 10.000

action 1.0 cli command "enable"

action 2.0 cli command "show process cpu"

action 3.0 mail server "192.168.1.146" to "engineer@cisco.com" from "eem@cisco.com" subject "CPU Alert" body "CPU Alert 75"

Router#

说明：结果显示了正常工作中的 EEM policy。

SCP（安全复制协议）

概述

SCP 通过认证的方式为 Cisco 设备的配置和映像复制提供安全，这种安全是基于 SSH 的。

在配置 SCP 之前，必须配置 SSH，认证和授权，所以必须有 RSA 密码。正因为有了 SSH 和 AAA 这样的机制所以能够确定用户是否是合法的。

配置

1．配置 AAA：

注：AAA 是必配，详细 AAA 介绍和配置，请参见本站的安全部分。

（1）开启 AAA

Router (config)# aaa new-model

（2）开启认证并指明认证方式

Router (config)# aaa authentication login default local　认证方式为本地用户数据库

（3）开启授权指明授权方式

Router (config)# aaa authorization exec default local　授权方式为本地用户数据库

（4）创建本地用户数据库

Router (config)# username ccie privilege 15 password cisco　必须为 15 级

2．配置 SSH：

注：SSH 是必配，详细 SSH 介绍和配置，请参见本站的安全部分。

（1）配置域名

r1(config)#ip domain-name cisco.com　　配置域名为 cisco.com

(2) 配置 RSA 密码

r1(config)#crypto key generate rsa

3．配置 SCP

（1）开启 SCP 服务

Router (config)# ip scp server enable

4.检查配置

（1）检查配置

Router# show running-config

5.SCP 测试：

(1)测试从远程路由器向这台开了 SCP 的路由器传送文件

说明：在远程路由器将 flash:下的文件 r1-confg 传送到开了 SCP 服务的路由器，目录是在 flash 下的。注：前面需要跟用户名，但密码只能在提示的时候才输入。

r1#copy flash:r1-confg scp://ccie@10.1.1.2/

Address or name of remote host [10.1.1.2]?

Destination username [ccie]?

Destination filename [r1-confg]?

Writing r1-confg

Password:

!

1054 bytes copied in 6.428 secs (164 bytes/sec)

r1#