

# H3C S5130S-SI[LI]&S5120V2-SI[LI]&S5110V2-SI& S5000V3-EI&S5000E-X&S3100V3-SI 系列以太网交换机

## 网络管理和监控配置指导

新华三技术有限公司  
<http://www.h3c.com>

资料版本：6W103-20190822  
产品版本：Release 612x 系列

**Copyright © 2019 新华三技术有限公司及其许可者 版权所有，保留一切权利。**

未经本公司书面许可，任何单位和个人不得擅自摘抄、复制本书内容的部分或全部，并不得以任何形式传播。

除新华三技术有限公司的商标外，本手册中出现的其它公司的商标、产品标识及商品名称，由各自权利人拥有。

# 前言

本配置指导主要介绍网络管理和监控相关功能的原理及具体配置。通过这些功能，您可以对网络进行管理和监控，包括查看系统信息、对网络流量进行统计、对网络质量进行分析、对网络内所有具有时钟的设备进行时钟同步，并可以使用 **ping**、**tracert**、**debug** 等命令来检查、调试当前网络的连接情况。

前言部分包含如下内容：

- [读者对象](#)
- [本书约定](#)
- [资料意见反馈](#)

## 读者对象

本手册主要适用于如下工程师：

- 网络规划人员
- 现场技术支持与维护人员
- 负责网络配置和维护的网络管理员

## 本书约定

### 1. 命令行格式约定

格 式	意 义
<b>粗体</b>	命令行关键字（命令中保持不变、必须照输的部分）采用 <b>加粗</b> 字体表示。
<i>斜体</i>	命令行参数（命令中必须由实际值进行替代的部分）采用 <i>斜体</i> 表示。
[ ]	表示用“[ ]”括起来的部分在命令配置时是可选的。
{ x   y   ... }	表示从多个选项中仅选取一个。
[ x   y   ... ]	表示从多个选项选取一个或者不选。
{ x   y   ... } *	表示从多个选项中至少选取一个。
[ x   y   ... ] *	表示从多个选项选取一个、多个或者不选。
&<1-n>	表示符号&前面的参数可以重复输入1~n次。
#	由“#”号开始的行表示为注释行。






### 2. 图形界面格式约定

格 式	意 义
< >	带尖括号“< >”表示按钮名，如“单击<确定>按钮”。
[ ]	带方括号“[ ]”表示窗口名、菜单名和数据表，如“弹出[新建用户]窗口”。

格 式	意 义
/	多级菜单用“/”隔开。如[文件/新建/文件夹]多级菜单表示[文件]菜单下的[新建]子菜单下的[文件夹]菜单项。

### 3. 各类标志

本书还采用各种醒目标志来表示在操作过程中应该特别注意的地方，这些标志的意义如下：

 警告	该标志后的注释需给予格外关注，不当的操作可能会对人身造成伤害。
 注意	提醒操作中应注意的事项，不当的操作可能会导致数据丢失或者设备损坏。
 提示	为确保设备配置成功或者正常工作而需要特别关注的操作或信息。
 说明	对操作内容的描述进行必要的补充和说明。
 窍门	配置、操作、或使用设备的技巧、小窍门。

### 4. 图标约定

本书使用的图标及其含义如下：

	该图标及其相关描述文字代表一般网络设备，如路由器、交换机、防火墙等。
	该图标及其相关描述文字代表一般意义下的路由器，以及其他运行了路由协议的设备。
	该图标及其相关描述文字代表二、三层以太网交换机，以及运行了二层协议的设备。
	该图标及其相关描述文字代表无线控制器、无线控制器业务板和有线无线一体化交换机的无线控制引擎设备。
	该图标及其相关描述文字代表无线接入点设备。
	该图标及其相关描述文字代表无线终结单元。
	该图标及其相关描述文字代表无线终结者。
	该图标及其相关描述文字代表无线Mesh设备。
	该图标代表发散的无线射频信号。
	该图标代表点到点的无线射频信号。
	该图标及其相关描述文字代表防火墙、UTM、多业务安全网关、负载均衡等安全设备。



该图标及其相关描述文字代表防火墙插卡、负载均衡插卡、NetStream插卡、SSL VPN插卡、IPS插卡、ACG插卡等安全插卡。

## 5. 示例约定

由于设备型号不同、配置不同、版本升级等原因，可能造成本手册中的内容与用户使用的设备显示信息不一致。实际使用中请以设备显示的内容为准。

本手册中出现的端口编号仅作参考，并不代表设备上实际具有此编号的端口，实际使用中请以设备上存在的端口编号为准。

## 资料意见反馈

如果您在使用过程中发现产品资料的任何问题，可以通过以下方式反馈：

E-mail: [info@h3c.com](mailto:info@h3c.com)

感谢您的反馈，让我们做得更好！

# 目 录

1 系统维护与调试.....	1-1
1.1 Ping功能.....	1-1
1.1.1 Ping功能简介.....	1-1
1.1.2 Ping操作.....	1-1
1.1.3 Ping功能典型配置举例 .....	1-2
1.2 Tracert功能 .....	1-2
1.2.1 Tracert功能简介 .....	1-2
1.2.2 配置准备.....	1-3
1.2.3 Tracert操作 .....	1-4
1.2.4 Tracert功能典型配置举例 .....	1-4
1.3 系统调试.....	1-5
1.3.1 系统调试简介 .....	1-5
1.3.2 系统调试操作 .....	1-6

# 1 系统维护与调试

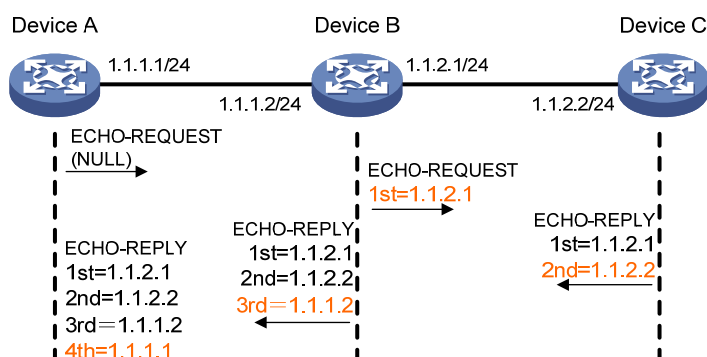
## 1.1 Ping功能

### 1.1.1 Ping功能简介

通过使用 Ping 功能，用户可以检查指定地址的设备是否可达，测试链路是否通畅。

Ping 功能是基于 ICMP (Internet Control Message Protocol, 互联网控制消息协议) 协议来实现的：源端向目的端发送 ICMP 回显请求 (ECHO-REQUEST) 报文后，根据是否收到目的端的 ICMP 回显应答 (ECHO-REPLY) 报文来判断目的端是否可达，对于可达的目的端，再根据发送报文个数、接收到响应报文个数以及 Ping 过程报文的往返时间来判断链路的质量。

图1-1 Ping 原理示意图



Ping功能也可以通过 `ping -r` 命令查看到链路的具体路由，如 [图 1-1](#) 所示，原理为：

- (2) 源端 (Device A) 发送 RR 选项 (ICMP 报文中的一个字段) 为空的 ICMP 回显请求给目的端 (Device C)。
- (3) 中间设备 (Device B) 将自己出接口的 IP 地址 (1.1.2.1) 添加到 ICMP 回显请求报文的 RR 选项中，并转发该报文。
- (4) 目的端收到请求报文后，发送 ICMP 回显响应报文，响应报文会拷贝请求报文的 RR 选项，并将自己出接口的 IP 地址 (1.1.2.2) 添加到 RR 选项中。
- (5) 中间设备将自己出接口的 IP 地址 (1.1.1.2) 添加到 RR 选项中，并转发该报文。

源端收到 ICMP 回显响应报文，将自己入接口的 IP 地址 (1.1.1.1) 添加到 RR 选项中。最后得到，Device A 到 Device C 具体路由为 1.1.1.1 <-> { 1.1.1.2; 1.1.2.1 } <-> 1.1.2.2。

### 1.1.2 Ping操作

可在任意视图下执行以下操作。

- 检查 IPv4 网络中的指定地址是否可达。

```
ping [ ip ] [ -a source-ip | -c count | -f | -h ttl | -i interface-type  
interface-number | -m interval | -n | -p pad | -q | -r | -s packet-size | -t  
timeout | -tos tos | -v ] * host
```

若网络传输速度较慢，在使用本命令时，可适当增大超时时间 **-t** 参数的值。

- 检查 IPv6 网络中的指定地址是否可达。

```
ping ipv6 [ -a source-ipv6 | -c count | -i interface-type  
interface-number | -m interval | -q | -s packet-size | -t timeout | -tc  
traffic-class | -v ] * host
```

若网络传输速度较慢，在使用本命令时，可适当增大超时时间 **-t** 参数的值。

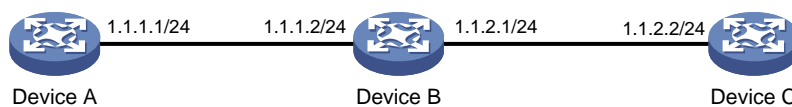
### 1.1.3 Ping功能典型配置举例

#### 1. 组网需求

检查 Device A 与 Device C 之间是否路由可达，如果路由可达，需要了解 Device A 到 Device C 的路由细节。

#### 2. 组网图

图1-2 Ping 应用组网图



#### 3. 配置步骤

# 使用 **ping** 命令查看 Device A 和 Device C 之间路由是否可达。

```
<DeviceA> ping 1.1.2.2
Ping 1.1.2.2 (1.1.2.2): 56 data bytes, press CTRL_C to break
56 bytes from 1.1.2.2: icmp_seq=0 ttl=254 time=2.137 ms
56 bytes from 1.1.2.2: icmp_seq=1 ttl=254 time=2.051 ms
56 bytes from 1.1.2.2: icmp_seq=2 ttl=254 time=1.996 ms
56 bytes from 1.1.2.2: icmp_seq=3 ttl=254 time=1.963 ms
56 bytes from 1.1.2.2: icmp_seq=4 ttl=254 time=1.991 ms

--- Ping statistics for 1.1.2.2 ---
5 packet(s) transmitted, 5 packet(s) received, 0.0% packet loss
round-trip min/avg/max/std-dev = 1.963/2.028/2.137/0.062 ms
```

以上显示信息表明 Device A 给 Device C 发送了 5 个 ICMP 报文，收到 5 个 ICMP 报文，没有报文丢失，路由可达。

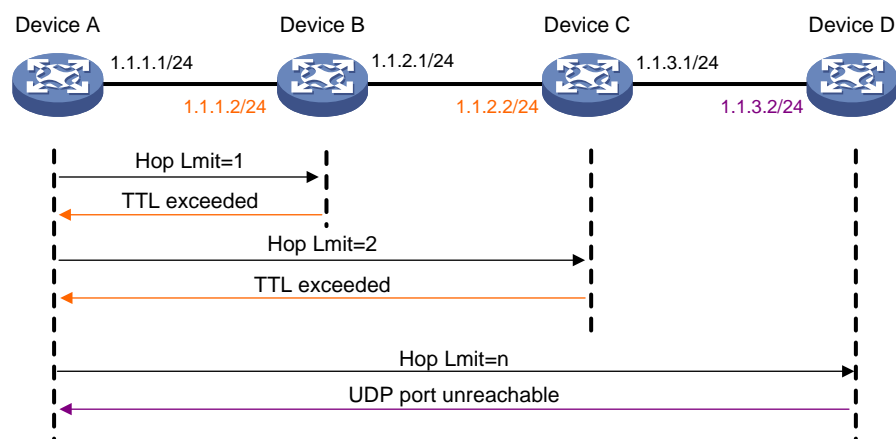
## 1.2 Tracert功能

### 1.2.1 Tracert功能简介

通过使用 **Tracert** 功能，用户可以查看 IP 报文从源端到达目的端所经过的三层设备，从而检查网络连接是否可用。当网络出现故障时，用户可以使用该功能分析出现故障的网络节点。



图1-3 Tracert 原理示意图



Tracert功能也是基于ICMP协议来实现的，如 图 1-3 所示，Tracert功能的原理为：

- (1) 源端（Device A）向目的端（Device D）发送一个 IP 数据报文，TTL 值为 1，报文的 UDP 端口号是目的端的任何一个应用程序都不可能使用的端口号；
- (2) 第一跳（即该报文所到达的第一个三层设备，Device B）回应一个 TTL 超时的 ICMP 错误消息（该报文中含有第一跳的 IP 地址 1.1.1.2），这样源端就得到了第一个三层设备的地址（1.1.1.2）；
- (3) 源端重新向目的端发送一个 IP 数据报文，TTL 值为 2；
- (4) 第二跳（Device C）回应一个 TTL 超时的 ICMP 错误消息，这样源端就得到了第二个三层设备的地址（1.1.2.2）；
- (5) 以上过程不断进行，直到该报文到达目的端，因目的端没有应用程序使用该 UDP 端口，目的端返回一个端口不可达的 ICMP 错误消息（携带了目的端的 IP 地址 1.1.3.2）；
- (6) 当源端收到这个端口不可达的 ICMP 错误消息后，就知道报文已经到达了目的端，从而得到数据报文从源端到目的端所经历的路径（1.1.1.2； 1.1.2.2； 1.1.3.2）。

## 1.2.2 配置准备

IPv4 网络环境：

- 需要在中间设备（源端与目的端之间的设备）上开启 ICMP 超时报文发送功能。如果中间设备是 H3C 设备，需要在设备上执行 **ip ttl-expires enable** 命令（该命令的详细介绍请参见“三层技术-IP 业务命令参考”中的“IP 性能优化”）。
- 需要在目的端开启 ICMP 目的不可达报文发送功能。如果目的端是 H3C 设备，需要在设备上执行 **ip unreachable enable** 命令（该命令的详细介绍请参见“三层技术-IP 业务命令参考”中的“IP 性能优化”）。

IPv6 网络环境：

- 需要在中间设备（源端与目的端之间的设备）上开启设备的 ICMPv6 超时报文的发送功能。如果中间设备是 H3C 设备，需要在设备上执行 **ipv6 hoplimit-expires enable** 命令（该命令的详细介绍请参见“三层技术-IP 业务命令参考”中的“IPv6 基础”）。

- 需要在目的端开启设备的 ICMPv6 目的不可达报文的发送功能。如果目的端是 H3C 设备，需要在设备上执行 **ipv6 unreachable enable** 命令（该命令的详细介绍请参见“三层技术-IP 业务命令参考”中的“IPv6 基础”）。

### 1.2.3 Tracert操作

可在任意视图下执行以下操作。

- 在 IPv4 网络环境查看源端到目的端的路由。  
`tracert [ -a source-ip | -f first-ttl | -m max-ttl | -p port | -q packet-number | -t tos | -w timeout ] * host`
- 在 IPv6 网络环境查看源端到目的端的路由。  
`tracert ipv6 [ -f first-hop | -m max-hops | -p port | -q packet-number | -t traffic-class | -w timeout ] * host`

### 1.2.4 Tracert功能典型配置举例

#### 1. 组网需求

Device A 使用 Telnet 登录 Device C 失败，现需要确认 Device A 与 Device C 之间是否路由可达，如果路由不可达，需要确定故障的网络节点。

#### 2. 组网图

图1-4 Tracert 应用组网图



#### 3. 配置步骤

(1) 在 Device A、Device B 和 Device C 上分别配置 IP 地址，IP 地址值如 [图 1-4](#) 所示。

(2) 在 Device A 上配置一条静态路由。

```

<DeviceA> system-view
[DeviceA] ip route-static 0.0.0.0 0.0.0.0 1.1.1.2
  
```

(3) 使用 **ping** 命令查看 Device A 和 Device C 之间路由是否可达。

```

<DeviceA> ping 1.1.2.2
Ping 1.1.2.2 (1.1.2.2): 56 data bytes, press CTRL_C to break
Request time out
Request time out
Request time out
Request time out
Request time out

--- Ping statistics for 1.1.2.2 ---
5 packet(s) transmitted, 0 packet(s) received, 100.0% packet loss
  
```

(4) 路由不可达，使用 **tracert** 命令确定故障的网络节点。

# 在 Device B 上开启 ICMP 超时报文发送功能。

```
[DeviceB] ip ttl-expires enable
```

# 在 Device C 上开启 ICMP 目的不可达报文发送功能。

```
[DeviceC] ip unreachable enable
```

# 在 Device A 上使用 **tracert** 命令确定故障的网络节点。

```
[DeviceA] tracert 1.1.2.2
```

```
traceroute to 1.1.2.2 (1.1.2.2), 30 hops at most, 40 bytes each packet, press CTRL_C to break
```

```
 1  1.1.1.2 (1.1.1.2) 1 ms 2 ms 1 ms
```

```
 2  * * *
```

```
 3  * * *
```

```
 4  * * *
```

```
 5
```

```
[DeviceA]
```

从上面结果可以看出，Device A 和 Device C 之间路由不可达。Device A 发往 Device C 的报文已经到达 Device B，Device B 和 Device C 之间的连接出了问题。此时可以在 Device A 和 Device C 上使用 **debugging ip icmp** 命令打开 ICMP 报文的调试开关，查看设备有没有收发指定的 ICMP 报文。或者使用 **display ip routing-table** 查看有没有到对端的路由。

## 1.3 系统调试

### 1.3.1 系统调试简介

设备提供了种类丰富的调试功能。设备支持的大部分功能模块，系统都提供了相应的调试信息，帮助用户对错误进行诊断和定位。

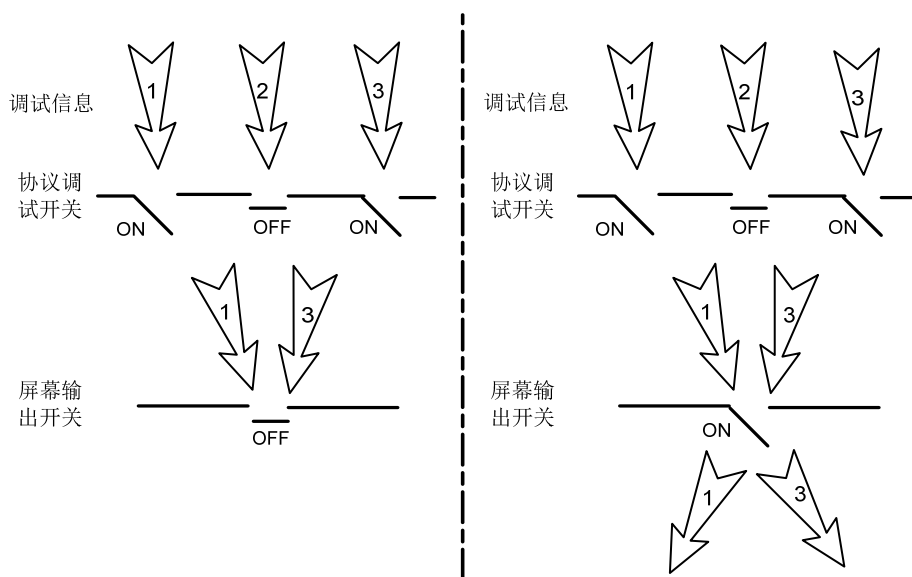
调试信息的输出可以由两个开关控制：

- 模块调试开关，控制是否生成某模块的调试信息。
- 屏幕输出开关，控制是否在某个用户屏幕上显示调试信息。屏幕输出开关可以使用 **terminal monitor** 和 **terminal logging level** 命令打开，**terminal monitor** 和 **terminal logging level** 命令的详细介绍请参见“网络管理与监控命令参考”中的“信息中心”。

如 [图 1-5](#) 所示：假设设备可以为 1、2、3 三个模块提供调试信息，用户只有将两个开关都打开，调试信息才会在终端显示出来。

在控制台上显示是最常用的调试信息输出方式，用户还可以将调试信息发送到别的输出方向，具体配置请参见“网络管理与监控配置指导”中的“信息中心”。

图1-5 系统调试开关关系图



## 1.3.2 系统调试操作

### 1. 配置限制和指导

**debugging** 命令一般在维护人员进行网络故障诊断时使用。由于调试信息的输出会影响系统的运行效率，所以建议在需要进行网络故障诊断时根据需要打开某个功能模块的调试开关，不要同时打开多个功能模块的调试开关。在调试结束后，建议使用 **undo debugging all** 命令关闭所有模块的调试开关。

### 2. 配置步骤

- (1) 打开指定模块的调试开关。

```
debugging module-name [ option ]
```

缺省情况下，所有模块的调试开关均处于关闭状态。

该命令在用户视图下执行。

- (2) （可选）显示已经打开的调试开关。

```
display debugging [ module-name ]
```

该命令可在任意视图下执行。

# 目 录

1 NQA .....	1-1
1.1 NQA简介 .....	1-1
1.1.1 NQA工作机制 .....	1-1
1.1.2 支持联动功能 .....	1-2
1.1.3 支持阈值告警功能 .....	1-2
1.1.4 NQA模板 .....	1-3
1.2 NQA配置任务简介 .....	1-3
1.3 配置NQA服务器 .....	1-3
1.4 使能NQA客户端功能 .....	1-4
1.5 在NQA客户端上配置NQA测试组 .....	1-4
1.5.1 NQA测试组配置任务简介 .....	1-4
1.5.2 配置ICMP-echo测试 .....	1-5
1.5.3 配置ICMP-jitter测试 .....	1-6
1.5.4 配置DHCP测试 .....	1-7
1.5.5 配置DNS测试 .....	1-8
1.5.6 配置FTP测试 .....	1-8
1.5.7 配置HTTP测试 .....	1-10
1.5.8 配置UDP-jitter测试 .....	1-11
1.5.9 配置SNMP测试 .....	1-12
1.5.10 配置TCP测试 .....	1-13
1.5.11 配置UDP-echo测试 .....	1-14
1.5.12 配置UDP-tracert测试 .....	1-15
1.5.13 配置Voice测试 .....	1-17
1.5.14 配置DLSw测试 .....	1-19
1.5.15 配置Path-jitter测试 .....	1-19
1.5.16 配置NQA测试组通用参数 .....	1-21
1.5.17 配置联动功能 .....	1-22
1.5.18 配置阈值告警功能 .....	1-22
1.5.19 配置NQA统计功能 .....	1-24
1.5.20 配置NQA历史记录功能 .....	1-25
1.5.21 在NQA客户端上调度NQA测试组 .....	1-26
1.6 在NQA客户端上配置NQA模板 .....	1-26
1.6.1 配置限制和指导 .....	1-26

1.6.2 NQA模板配置任务简介 .....	1-26
1.6.3 配置ICMP类型的NQA模板 .....	1-27
1.6.4 配置DNS类型的NQA模板 .....	1-28
1.6.5 配置TCP类型的NQA模板 .....	1-30
1.6.6 配置TCP Half Open类型的NQA模板 .....	1-31
1.6.7 配置UDP类型的NQA模板 .....	1-32
1.6.8 配置HTTP类型的NQA模板 .....	1-33
1.6.9 配置HTTPS类型的NQA模板 .....	1-35
1.6.10 配置FTP类型的NQA模板 .....	1-37
1.6.11 配置RADIUS类型的NQA模板 .....	1-38
1.6.12 配置SSL类型的NQA模板 .....	1-39
1.6.13 配置NQA模板通用参数 .....	1-40
1.7 NQA显示和维护 .....	1-41
1.8 NQA典型配置举例 .....	1-42
1.8.1 ICMP-echo测试配置举例 .....	1-42
1.8.2 ICMP-jitter测试配置举例 .....	1-43
1.8.3 DHCP测试配置举例 .....	1-46
1.8.4 DNS测试配置举例 .....	1-47
1.8.5 FTP测试配置举例 .....	1-48
1.8.6 HTTP测试配置举例 .....	1-49
1.8.7 UDP-jitter测试配置举例 .....	1-51
1.8.8 SNMP测试配置举例 .....	1-53
1.8.9 TCP测试配置举例 .....	1-55
1.8.10 UDP-echo测试配置举例 .....	1-56
1.8.11 UDP-tracert测试配置举例 .....	1-57
1.8.12 Voice测试配置举例 .....	1-59
1.8.13 DLSw测试配置举例 .....	1-61
1.8.14 Path-jitter测试配置举例 .....	1-63
1.8.15 NQA联动配置举例 .....	1-64
1.8.16 ICMP类型的NQA模板配置举例 .....	1-67
1.8.17 DNS类型的NQA模板配置举例 .....	1-68
1.8.18 TCP类型的NQA模板配置举例 .....	1-69
1.8.19 TCP Half Open类型的NQA模板配置举例 .....	1-69
1.8.20 UDP类型的NQA模板配置举例 .....	1-70
1.8.21 HTTP类型的NQA模板配置举例 .....	1-71
1.8.22 HTTPS类型的NQA模板配置举例 .....	1-72

1.8.23 FTP类型的NQA模板配置举例 .....	1-73
1.8.24 RADIUS类型的NQA模板配置举例.....	1-74
1.8.25 SSL类型的NQA模板配置举例 .....	1-74

# 1 NQA

## 1.1 NQA简介

NQA（Network Quality Analyzer，网络质量分析）通过发送探测报文，对链路状态、网络性能、网络提供的服务及服务质量进行分析，并为用户提供标识当前网络性能和服务质量的参数，如时延、抖动时间、TCP 连接建立时间、FTP 连接建立时间和文件传输速率等。利用 NQA 的分析结果，用户可以及时了解网络的性能状况，针对不同的网络性能进行相应处理并对网络故障进行诊断和定位。

### 1.1.1 NQA工作机制

图1-1 NQA 测试典型组网图



如 [图 1-1](#) 所示，NQA 测试的典型组网中包括以下两部分：

- NQA 测试的源端设备：又称为 NQA 客户端，负责发起 NQA 测试，并统计探测结果。NQA 测试组在 NQA 客户端上创建。NQA 测试组是一组测试参数的集合，如测试类型、测试目的地址、测试目的端口等。NQA 测试组由一个管理员名称和一个操作标签来标识。管理员通过 NQA 测试组来实现对 NQA 测试的管理和调度。在一台设备上可以创建多个 NQA 测试组，可以同时启动多个 NQA 测试组进行测试。
- NQA 测试的目的端设备：负责接收、处理和响应 NQA 客户端发来的探测报文。
  - 在进行 TCP、UDP-echo、UDP-jitter 和 Voice 类型测试时，必须在目的端设备上配置 NQA 服务器功能，开启指定 IP 地址和端口上的监听服务。此时，目的端设备又称为 NQA 服务器。当 NQA 服务器接收到客户端发送给指定 IP 地址和端口的探测报文后，将对其进行处理，并发送响应报文。
  - 在其他类型的测试中，目的端设备只要能够处理 NQA 客户端发送的探测报文即可，不需要配置 NQA 服务器功能。例如，在 FTP 测试中，目的端设备上需要配置 FTP 服务器相关功能，以便处理客户端发送的 FTP 报文，而无需配置 NQA 服务器功能。

NQA 测试的过程为：

- (1) NQA 客户端构造指定测试类型的探测报文，并发送给目的端设备；
- (2) 目的端设备收到探测报文后，回复应答报文；
- (3) NQA 客户端根据是否收到应答报文，以及接收应答报文的时间，计算报文丢失率、往返时间等。

启动 NQA 测试组后，每隔一段时间进行一次测试，测试的时间间隔由 **frequency** 命令来设定。一次 NQA 测试由若干次连续的探测组成，探测的次数由 **probe count** 命令来设定。



对于 Voice 和 Path-jitter 测试，一次测试中只能进行一次探测，不能通过配置修改测试中探测的次数。

### 1.1.2 支持联动功能

联动功能是指在监测模块、Track 模块和应用模块之间建立关联，实现这些模块之间的联合动作。联动功能利用监测模块对链路状态、网络性能等进行监测，并通过 Track 模块将监测结果及时通知给应用模块，以便应用模块进行相应的处理。联动功能的详细介绍，请参见“可靠性配置指导”中的“Track”。

NQA 可以作为联动功能的监测模块，对 NQA 探测结果进行监测，当连续探测失败次数达到一定数目时，就通过 Track 模块触发应用模块进行相应的处理。

目前，NQA 可以通过 Track 模块建立关联的应用模块包括：

- VRRP（Virtual Router Redundancy Protocol，虚拟路由器冗余协议）
- 静态路由
- 策略路由
- 流量重定向
- Smart Link

以静态路由为例，用户配置了一条静态路由，下一跳为 192.168.0.88。通过在 NQA、Track 模块和静态路由模块之间建立联动，可以实现静态路由有效性的判断：

- (1) 通过 NQA 监测地址 192.168.0.88 是否可达。
- (2) 如果 192.168.0.88 可达，则认为该静态路由有效，NQA 不通知 Track 模块改变 Track 项的状态；如果 NQA 发现 192.168.0.88 不可达，则通知 Track 模块改变 Track 项的状态。
- (3) Track 模块将改变后的 Track 项状态通知给静态路由模块。静态路由模块据此可以判断该静态路由项是否有效。

### 1.1.3 支持阈值告警功能

NQA 可以对探测结果进行监测，在本地记录监测结果，或通过 Trap 消息将监测结果通知给网络管理系统，以便网络管理员了解 NQA 测试运行结果和网络性能。

NQA 阈值告警功能支持的监测对象及对应的测试类型，如 [表 1-1](#) 所示。

表1-1 NQA 阈值告警功能支持的监测对象及对应的测试类型

监测对象	支持的测试类型
探测持续时间	ICMP-echo、DHCP、DNS、FTP、HTTP、SNMP、TCP、UDP-echo和DLSw测试类型
探测失败次数	ICMP-echo、DHCP、DNS、FTP、HTTP、SNMP、TCP、UDP-echo和DLSw测试类型
报文往返时间	ICMP-jitter、UDP-jitter和Voice测试类型
丢弃报文数目	ICMP-jitter、UDP-jitter和Voice测试类型
源到目的或目的到源的单向抖动时间	ICMP-jitter、UDP-jitter和Voice测试类型
源到目的或目的到源的单向时延	ICMP-jitter、UDP-jitter和Voice测试类型

监测对象	支持的测试类型
ICPIF（Calculated Planning Impairment Factor，计算计划损伤元素）值 ICPIF的详细介绍请参见“ <a href="#">1.5.13 配置Voice测试</a> ”	Voice测试类型
MOS（Mean Opinion Scores，平均意见得分）值 MOS的详细介绍请参见“ <a href="#">1.5.13 配置Voice测试</a> ”	Voice测试类型

#### 1.1.4 NQA模板

NQA 模板是一组测试参数的集合（如测试目的地址、测试目的端口、测试目标服务器的 URL 等）。NQA 模板供外部特性调用，可以为外部特性提供测试数据，以便其进行相应处理。NQA 模板通过模板名唯一标识。在一台设备上可以创建多个 NQA 模板。

### 1.2 NQA配置任务简介

NQA 配置任务如下：

(1) [配置NQA服务器](#)

在进行 TCP、UDP-echo、UDP-jitter 和 Voice 类型测试前，必须在目的端设备上进行本配置。进行其他类型测试时，不需要进行本配置。

(2) [使能NQA客户端功能](#)

(3) 配置 NQA 测试组和模板

请至少选择以下一项任务进行配置：

- [在NQA客户端上配置NQA测试组](#)
- [在NQA客户端上配置NQA模板](#)

NQA 测试组配置完毕后，通过调度测试组就可以进行测试操作；NQA 模板配置完毕后并不启动测试，需要从外部特性调用 NQA 模板后，设备自动创建 NQA 测试组并启动 NQA 测试。

### 1.3 配置NQA服务器

#### 1. 配置限制和指导

在进行 TCP、UDP-echo、UDP-jitter 和 Voice 类型测试前，必须在目的端设备上进行本配置。进行其他类型测试时，不需要进行本配置。

在一个 NQA 服务器上可以配置多个 TCP（或 UDP）监听服务，每个监听服务对应一个监听的 IP 地址和一个端口号。

配置的监听 IP 地址和端口号必须与 NQA 客户端上配置的目的 IP 地址和目的端口号一致，且不能与已有的 TCP（或 UDP）监听服务冲突。

#### 2. 配置步骤

(1) 进入系统视图。

```
system-view
```

(2) 开启 NQA 服务器功能。

**nqa server enable**

缺省情况下，NQA 服务器功能处于关闭状态。

- (3) 在 NQA 服务器上配置 TCP 监听服务。

**nqa server tcp-connect ip-address port-number [ tos tos ]**

仅 TCP 测试类型下必须进行本配置。

- (4) 在 NQA 服务器上配置 UDP 监听服务。

**nqa server udp-echo ip-address port-number [ tos tos ]**

仅 UDP-echo、UDP-jitter 和 Voice 测试类型下必须进行本配置。

## 1.4 使能NQA客户端功能

- (1) 进入系统视图。

**system-view**

- (2) 使能 NQA 客户端功能。

**nqa agent enable**

缺省情况下，NQA 客户端功能处于开启状态。

只有使能 NQA 客户端功能后，NQA 客户端的相关配置才会生效。

## 1.5 在NQA客户端上配置NQA测试组

### 1.5.1 NQA测试组配置任务简介

NQA 测试组配置任务如下：

- (1) 配置 NQA 测试组

- [配置ICMP-echo测试](#)
- [配置ICMP-jitter测试](#)
- [配置DHCP测试](#)
- [配置DNS测试](#)
- [配置FTP测试](#)
- [配置HTTP测试](#)
- [配置UDP-jitter测试](#)
- [配置SNMP测试](#)
- [配置TCP测试](#)
- [配置UDP-echo测试](#)
- [配置UDP-tracert测试](#)
- [配置Voice测试](#)
- [配置DLSw测试](#)
- [配置Path-jitter测试](#)

- (2) （可选）[配置NQA测试组通用参数](#)

- (3) （可选）[配置联动功能](#)

- (4) (可选) [配置阈值告警功能](#)
- (5) (可选) [配置NQA统计功能](#)
- (6) (可选) [配置NQA历史记录功能](#)
- (7) [在NQA客户端上调度NQA测试组](#)

## 1.5.2 配置ICMP-echo测试

### 1. 功能简介

ICMP-echo 测试利用 ICMP 协议，根据是否接收到应答报文判断目的主机的可达性。ICMP-echo 测试的功能与 **ping** 命令类似，但 ICMP-echo 测试中可以指定测试的下一跳设备。在源端和目的端设备之间存在多条路径时，通过配置下一跳设备可以指定测试的路径。并且，与 **ping** 命令相比，ICMP-echo 测试输出的信息更为丰富。

对于 ICMP-echo 测试，一次探测操作是指向目的端发送一个探测报文。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 创建 NQA 测试组，并进入 NQA 测试组视图。

```
nqa entry admin-name operation-tag
```

- (3) 配置测试类型为 ICMP-echo，并进入测试类型视图。

```
type icmp-echo
```

- (4) 配置探测报文的地址。

(IPv4 网络)

```
destination ip ip-address
```

(IPv6 网络)

```
destination ipv6 ipv6-address
```

缺省情况下，未配置探测报文的地址。

- (5) 配置探测报文的源地址。请选择其中一项进行配置。

- 使用指定接口的 IP 地址作为探测报文的源 IP 地址。

```
source interface interface-type interface-number
```

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

**source interface** 命令指定的接口必须为 up 状态。

- 配置探测报文的源 IPv4 地址。

```
source ip ip-address
```

缺省情况下，以报文发送接口的主 IPv4 地址作为探测报文中的源 IPv4 地址。

该命令指定的源地址必须是设备上接口的 IP 地址，且接口为 up 状态，否则测试将会失败。

- 配置探测报文的源 IPv6 地址。

```
source ipv6 ipv6-address
```

缺省情况下，以报文发送接口的 IPv6 地址作为探测报文中的源 IPv6 地址。

该命令指定的源地址必须是设备上接口的 IPv6 地址，且接口为 up 状态，否则测试将会失败。

(6) 配置探测报文出接口或下一跳 IP 地址。请选择其中一项进行配置。

- 配置探测报文出接口。

**out interface** *interface-type interface-number*

缺省情况下，设备通过查询路由表信息确认探测报文出接口。

- 配置探测报文的下一跳 IPv4 地址。

**next-hop ip** *ip-address*

缺省情况下，未配置探测报文的下一跳 IPv4 地址。

- 配置探测报文的下一跳 IPv6 地址。

**next-hop ipv6** *ipv6-address*

缺省情况下，未配置探测报文的下一跳 IPv6 地址。

(7) （可选）配置探测报文中的填充内容大小。

**data-size** *size*

缺省情况下，探测报文中的填充内容大小为 100 字节。

(8) （可选）配置探测报文的填充字符串。

**data-fill** *string*

探测报文的填充内容为十六进制 00010203040506070809。

### 1.5.3 配置ICMP-jitter测试

#### 1. 功能简介

语音、视频等实时性业务对时延抖动（Delay jitter）的要求较高。通过 ICMP-jitter 测试，可以获得网络的单向和双向时延抖动，从而判断网络是否可以承载实时性业务。

ICMP-jitter 测试的过程如下：

- (1) 源端以一定的时间间隔向目的端发送探测报文。
- (2) 目的端收到探测报文后，为它打上时间戳，并把带有时间戳的报文发送给源端。
- (3) 源端收到报文后，根据报文上的时间戳，计算出时延抖动，从而清晰地反映出网络状况。时延抖动的计算方法为相邻两个报文的目的端接收时间间隔减去这两个报文的发送时间间隔。

对于 ICMP-jitter 测试，一次探测操作是指向目的端连续发送多个探测报文，发送探测报文的个数由 **probe packet-number** 命令来设定。

#### 2. 配置限制和指导

**display nqa history** 命令的显示信息无法反映 ICMP-jitter 测试的结果，如果了解 ICMP-jitter 测试的结果，建议通过 **display nqa result** 命令查看最近一次 NQA 测试的当前结果，或通过 **display nqa statistics** 命令查看 NQA 测试的统计信息。

进行本测试前需保证网络时钟的 NTP 同步。有关 NTP 的详细介绍请参见“网络管理和监控配置指导”的“NTP”。

#### 3. 配置步骤

- (1) 进入系统视图。

#### **system-view**

- (2) 创建 NQA 测试组，进入 NQA 测试组视图。

**nqa entry** *admin-name operation-tag*

- (3) 配置测试类型为 ICMP-jitter，并进入测试类型视图。

**type icmp-jitter**

- (4) 配置探测报文的目的 IP 地址。

**destination ip** *ip-address*

缺省情况下，未配置探测报文的目的 IP 地址。

- (5) 配置一次 ICMP-jitter 探测中发送探测报文的个数。

**probe packet-number** *packet-number*

缺省情况下，一次 ICMP-jitter 探测中发送 10 个探测报文。

- (6) 配置 ICMP-jitter 测试中发送探测报文的时间间隔。

**probe packet-interval** *packet-interval*

缺省情况下，ICMP-jitter 测试中发送探测报文的时间间隔为 20 毫秒。

- (7) 配置 ICMP-jitter 测试中等待响应报文的超时时间。

**probe packet-timeout** *packet-timeout*

缺省情况下，ICMP-jitter 测试中等待响应报文的超时时间为 3000 毫秒。

- (8) 配置探测报文的源 IP 地址。

**source ip** *ip-address*

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

该命令指定的源地址必须是设备上接口的 IP 地址，且接口为 up 状态，否则测试将会失败。

## 1.5.4 配置DHCP测试

### 1. 功能简介

DHCP 测试主要用来测试网络上的 DHCP 服务器能否响应客户端请求，以及为客户端分配 IP 地址所需的时间。

NQA 客户端模拟 DHCP 中继转发 DHCP 请求报文向 DHCP 服务器申请 IP 地址的过程，DHCP 服务器进行 DHCP 测试的接口 IP 地址不会改变。DHCP 测试完成后，NQA 客户端会主动发送报文释放申请到的 IP 地址。

对于 DHCP 测试，一次探测操作是指完成一次向 DHCP 服务器申请一个 IP 地址。

### 2. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 创建 NQA 测试组，并进入 NQA 测试组视图。

**nqa entry** *admin-name operation-tag*

- (3) 配置测试类型为 DHCP，并进入测试类型视图。

**type dhcp**

- (4) 配置探测报文的目的 IP 地址。

**destination ip** *ip-address*

缺省情况下，未配置探测报文的目的 IP 地址。

- (5) 配置探测报文出接口。

**out interface** *interface-type interface-number*

缺省情况下，设备通过查询路由表信息确认探测报文出接口。

- (6) 配置探测报文的源 IP 地址。

**source ip** *ip-address*

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

该命令指定的源地址必须是设备上接口的 IP 地址，且接口为 up 状态，否则测试将会失败。

## 1.5.5 配置DNS测试

### 1. 功能简介

DNS 测试主要用来测试 NQA 客户端是否可以通过指定的 DNS 服务器将域名解析为 IP 地址，以及域名解析过程需要的时间。

DNS 测试只是模拟域名解析的过程，设备上不会保存要解析的域名与 IP 地址的对应关系。

对于 DNS 测试，一次探测操作是指完成一次将一个域名解析为 IP 地址。

### 2. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 创建 NQA 测试组，并进入 NQA 测试组视图。

**nqa entry** *admin-name operation-tag*

- (3) 配置测试类型为 DNS，并进入测试类型视图。

**type dns**

- (4) 配置探测报文的目的 IP 地址。

**destination ip** *ip-address*

缺省情况下，未配置探测报文的目的 IP 地址。

- (5) 配置要解析的域名。

**resolve-target** *domain-name*

缺省情况下，没有配置要解析的域名。

## 1.5.6 配置FTP测试

### 1. 功能简介

FTP 测试主要用来测试 NQA 客户端是否可以与指定的 FTP 服务器建立连接，以及与 FTP 服务器之间传送文件的时间，从而判断 FTP 服务器的连通性及性能。

在进行 FTP 测试之前，需要获取 FTP 用户的用户名和密码。

对于 FTP 测试，一次探测操作是指完成一次向 FTP 服务器上传或下载一个文件。



## 2. 配置限制和指导

进行 **put** 操作时，若配置了 **filename**，发送数据前判断 **filename** 指定的文件是否存在，如果存在则上传该文件，如果不存在则探测失败。

进行 **get** 操作时，如果 FTP 服务器上没有以 **url** 中所配置的文件名为名称的文件，则测试不会成功。进行 **get** 操作时，设备上不会保存从服务器获取的文件。

进行 **get**、**put** 操作时，请选用较小的文件进行测试，如果文件较大，可能会因为超时而导致测试失败，或由于占用较多的网络带宽而影响其他业务。

## 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 创建 NQA 测试组，并进入 NQA 测试组视图。

```
nqa entry admin-name operation-tag
```

- (3) 配置测试类型为 FTP，并进入测试类型视图。

```
type ftp
```

- (4) 配置 FTP 登录用户名。

```
username username
```

缺省情况下，未配置 FTP 登录用户名。

- (5) 配置 FTP 登录密码。

```
password { cipher | simple } string
```

缺省情况下，未配置 FTP 登录密码。

- (6) 配置探测报文的源 IP 地址。

```
source ip ip-address
```

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

该命令指定的源地址必须是设备上接口的 IP 地址，且接口为 up 状态，否则测试将会失败。

- (7) 配置 FTP 测试的数据传输方式。

```
mode { active | passive }
```

缺省情况下，FTP 测试的数据传输方式为主动方式。

- (8) 配置 FTP 测试的操作类型。

```
operation { get | put }
```

缺省情况下，FTP 操作方式为 **get** 操作，即从 FTP 服务器获取文件。

- (9) 配置 FTP 测试访问的网址。

```
url url
```

缺省情况下，没有配置 FTP 测试访问的网址。

**url** 可以设置为 **ftp://host/filename** 或 **ftp://host:port/filename**。当 FTP 测试的操作类型为 **get** 方式时，必须在 **url** 中配置 **filename** 指定从 FTP 服务器获取的文件名。

- (10) 配置 FTP 服务器和客户端传送文件的文件名。

```
filename filename
```

缺省情况下，未配置 FTP 服务器和客户端之间传送文件的文件名。



当 FTP 测试的操作类型为 **put** 方式时，必须配置本命令来指定向 FTP 服务器传送的文件。  
当 FTP 测试的操作类型为 **get** 方式时，不以此命令为准。

## 1.5.7 配置HTTP测试

### 1. 功能简介

HTTP 测试主要用来测试 NQA 客户端是否可以与指定的 HTTP 服务器建立连接，以及从 HTTP 服务器获取数据所需的时间，从而判断 HTTP 服务器的连通性及性能。

HTTP 测试支持如下操作类型：

- **get**：从 HTTP 服务器获取数据。
- **post**：向 HTTP 服务器提交数据。
- **raw**：向 HTTP 服务器发送 RAW 请求报文。

对于 HTTP 测试，一次探测操作是指完成一次相应操作类型的功能。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 创建 NQA 测试组，并进入 NQA 测试组视图。

```
nqa entry admin-name operation-tag
```

- (3) 配置测试类型为 HTTP，并进入测试类型视图。

```
type http
```

- (4) 配置 HTTP 测试访问的网址。

```
url url
```

缺省情况下，没有配置 HTTP 测试访问的网址。

**url** 配置形式为 `http://host/resource` 或 `http://host:port/resource`。

- (5) 配置 HTTP 登录用户名。

```
username username
```

缺省情况下，未配置 HTTP 登录用户名。

- (6) 配置 HTTP 登录密码。

```
password { cipher | simple } string
```

缺省情况下，未配置 HTTP 登录密码。

- (7) 配置 HTTP 测试所使用的协议版本。

```
version { v1.0 | v1.1 }
```

缺省情况下，HTTP 测试使用的版本为 1.0。

- (8) 配置 HTTP 测试的操作类型。

```
operation { get | post | raw }
```

缺省情况下，HTTP 操作方式为 **get** 操作。如果 HTTP 操作方式为 **raw** 操作，则向服务器发送的探测报文的内容为 **raw-request** 视图中的内容。

- (9) 配置 HTTP 测试请求报文。

- a. 进入 raw-request 视图。

#### **raw-request**

输入 **raw-request** 命令进入 raw-request 视图，每次进入视图原有报文内容清除。

##### b. 配置 HTTP 测试请求报文内容。

逐个字符输入或拷贝粘贴请求报文内容。

缺省情况下，未配置 HTTP 测试请求报文内容。

##### c. 保存输入内容并退回测试类型视图。

#### **quit**

当配置 HTTP 测试的操作类型为 **raw** 时，必须完成此操作且保证发送的测试报文正确有效。

#### (10) 配置探测报文的源 IP 地址。

**source ip ip-address**

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

该命令指定的源地址必须是设备上接口的 IP 地址，且接口为 up 状态，否则测试将会失败。

## 1.5.8 配置UDP-jitter测试

### 1. 功能简介

语音、视频等实时性业务对时延的抖动时间（Delay jitter）的要求较高。通过 UDP-jitter 测试，可以获得网络的单向和双向抖动的时间，从而判断网络是否可以承载实时性业务。

UDP-jitter 测试的过程如下：

- (1) 源端以一定的时间间隔向目的端发送探测报文。
- (2) 目的端收到探测报文后，为它打上时间戳，并把带有时间戳的报文发送给源端。
- (3) 源端收到报文后，根据报文上的时间戳，计算出抖动时间，从而清晰地反映出网络状况。抖动时间的计算方法为相邻两个报文的目的端接收时间间隔减去这两个报文的发送时间间隔。

对于 UDP-jitter 测试，一次探测操作是指连续发送多个探测报文，发送探测报文的个数由 **probe packet-number** 命令来设定。

UDP-jitter测试需要NQA服务器和客户端配合才能完成。进行UDP-jitter测试之前，必须保证NQA服务器端配置了UDP监听功能，配置方法请参见“[1.3 配置NQA服务器](#)”。

### 2. 配置限制和指导

建议不要对知名端口，即 1~1023 之间的端口，进行 UDP-jitter 测试，否则可能导致 NQA 测试失败或该知名端口对应的服务不可用。

**display nqa history** 命令的显示信息无法反映 UDP-jitter 测试的结果，如果了解 UDP-jitter 测试的结果，建议通过 **display nqa result** 命令查看最近一次 NQA 测试的当前结果，或通过 **display nqa statistics** 命令查看 NQA 测试的统计信息。

进行本测试前需保证网络时钟的 NTP 同步。有关 NTP 的详细介绍请参见“网络管理和监控配置指导”的“NTP”。

### 3. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 创建 NQA 测试组，并进入 NQA 测试组视图。

**nqa entry** *admin-name operation-tag*

- (3) 配置测试类型为 UDP-jitter，并进入测试类型视图。

**type** **udp-jitter**

- (4) 配置探测报文的目的 IP 地址。

**destination ip** *ip-address*

缺省情况下，未配置探测报文的目的 IP 地址。

必须与 NQA 服务器上配置的监听服务的 IP 地址一致。

- (5) 配置测试操作的目的端口。

**destination port** *port-number*

缺省情况下，未配置测试操作的目的端口号。

必须与 NQA 服务器上配置的监听服务的端口号一致。

- (6) 配置探测报文的源 IP 地址。

**source ip** *ip-address*

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

该命令指定的源地址必须是设备上接口的 IP 地址，且接口为 up 状态，否则测试将会失败。

- (7) 配置探测报文的源端口。

**source port** *port-number*

缺省情况下，系统自动选择设备当前空闲的端口号。

- (8) 配置一次 UDP-jitter 探测中发送探测报文的个数。

**probe packet-number** *packet-number*

缺省情况下，一次 UDP-jitter 探测中发送 10 个探测报文。

- (9) 配置 UDP-jitter 测试中发送探测报文的时间间隔。

**probe packet-interval** *interval*

缺省情况下，UDP-jitter 测试中发送探测报文的时间间隔为 20 毫秒。

- (10) 配置 UDP-jitter 测试中等待响应报文的超时时间。

**probe packet-timeout** *timeout*

缺省情况下，UDP-jitter 测试中等待响应报文的超时时间为 3000 毫秒。

- (11) （可选）配置探测报文中的填充内容的大小。

**data-size** *size*

缺省情况下，探测报文中的填充内容大小为 100 字节。

- (12) （可选）配置探测报文的填充字符串。

**data-fill** *string*

探测报文的填充内容为十六进制 00010203040506070809。

## 1.5.9 配置SNMP测试

### 1. 功能简介

SNMP 测试主要测试从 NQA 客户端向 SNMP Agent 设备发出一个 SNMP 协议查询，根据能否收到应答报文判断 SNMP Agent 上提供的 SNMP 服务是否可用。

对于 SNMP 测试，一次探测操作是指发送三个 SNMP 协议报文，分别对应 SNMPv1、SNMPv2c 和 SNMPv3 三个版本。

## 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 创建 NQA 测试组，并进入 NQA 测试组视图。

```
nqa entry admin-name operation-tag
```

- (3) 配置测试类型为 SNMP，并进入测试类型视图。

```
type snmp
```

- (4) 配置探测报文的地址。

```
destination ip ip-address
```

缺省情况下，未配置探测报文的地址 IP 地址。

- (5) 配置探测报文的源 IP 地址。

```
source ip ip-address
```

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

该命令指定的源地址必须是设备上接口的 IP 地址，且接口为 up 状态，否则测试将会失败。

- (6) 配置探测报文的源端口。

```
source port port-number
```

缺省情况下，系统自动选择设备当前空闲的端口号。

- (7) 配置用于 SNMPv1 或者 SNMPv2c 探测报文的团体名。

```
community read { cipher | simple } community-name
```

缺省情况下，SNMPv1 或者 SNMPv2c 探测报文使用的团体名为 public。

该命令配置的团体名必须为 SNMP Agent 上已配置具有读权限的团体名。

## 1.5.10 配置TCP测试

### 1. 功能简介

TCP 测试用来测试客户端和服务端指定端口之间是否能够建立 TCP 连接，以及建立 TCP 连接所需的时间，从而判断服务端指定端口上提供的服务是否可用，及服务性能。

TCP测试需要NQA服务器和客户端配合才能完成。在TCP测试之前，需要在NQA服务器端配置TCP监听功能，配置方法请参见“[1.3 配置NQA服务器](#)”。

对于 TCP 测试，一次探测操作是指建立一次 TCP 连接。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 创建 NQA 测试组，并进入 NQA 测试组视图。

```
nqa entry admin-name operation-tag
```

- (3) 配置测试类型为 TCP，并进入测试类型视图。

```
type tcp
```

- (4) 配置探测报文的目的地址。

**destination ip** *ip-address*

缺省情况下，未配置探测报文的目的 IP 地址。

必须与 NQA 服务器上配置的监听服务的 IP 地址一致。

- (5) 配置测试操作的目的端口。

**destination port** *port-number*

缺省情况下，未配置测试操作的目的端口号。

必须与 NQA 服务器上配置的监听服务的端口号一致。

- (6) 配置探测报文的源 IP 地址。

**source ip** *ip-address*

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

该命令指定的源地址必须是设备上接口的 IP 地址，且接口为 up 状态，否则测试将会失败。

### 1.5.11 配置UDP-echo测试

#### 1. 功能简介

UDP-echo 测试可以用来测试客户端和服务器指定 UDP 端口之间的连通性以及 UDP 报文的往返时间。

UDP-echo测试需要NQA服务器和客户端配合才能完成。在进行UDP-echo测试之前，需要在NQA服务器端配置UDP监听功能，配置方法请参见“[1.3 配置NQA服务器](#)”。

对于 UDP-echo 测试，一次探测操作是指发送一个探测报文。

#### 2. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 创建 NQA 测试组，并进入 NQA 测试组视图。

**nqa entry** *admin-name operation-tag*

- (3) 配置测试类型为 UDP-echo，并进入测试类型视图。

**type udp-echo**

- (4) 配置探测报文的目的地址。

**destination ip** *ip-address*

缺省情况下，未配置探测报文的目的 IP 地址。

必须与 NQA 服务器上配置的监听服务的 IP 地址一致。

- (5) 配置测试操作的目的端口。

**destination port** *port-number*

缺省情况下，未配置测试操作的目的端口号。

必须与 NQA 服务器上配置的监听服务的端口号一致。

- (6) 配置探测报文的源 IP 地址。

**source ip** *ip-address*

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

该命令指定的源地址必须是设备上接口的 IP 地址，且接口为 up 状态，否则测试将会失败。

- (7) 配置探测报文的源端口。

**source port** *port-number*

缺省情况下，系统自动选择设备当前空闲的端口号。

- (8) （可选）配置探测报文中的填充内容大小。

**data-size** *size*

缺省情况下，探测报文中的填充内容大小为 100 字节。

- (9) （可选）配置探测报文的填充字符串。

**data-fill** *string*

探测报文的填充内容为十六进制 00010203040506070809。

## 1.5.12 配置UDP-tracert测试

### 1. 功能简介

UDP-tracert 测试可以用来发现源端到目的端之间的路径信息。UDP-tracert 测试和普通 Tracert 流程一致，由源端发送一个目的端口不可达的报文，当目的端收到该报文后，会回复源端一个端口不可达报文，以便使源端知道 Tracert 测试结束。

对于 UDP-tracert 测试，一次探测操作是指一个特定 TTL 值的节点发送一个探测报文。

### 2. 配置限制和指导

UDP-tracert 测试不支持在 IPv6 网络中使用，如果要测试 IPv6 网络中目的主机的可达性，可以使用 **tracert ipv6** 命令。**tracert ipv6** 命令的详细介绍，请参见“网络管理和监控命令参考”中的“系统维护与调试”。

### 3. 配置准备

配置 UDP-tracert 测试需要在中间设备（源端与目的端之间的设备）上开启 ICMP 超时报文发送功能。如果中间设备是 H3C 设备，需要在设备上执行 **ip ttl-expires enable** 命令（该命令的详细介绍请参见“三层技术-IP 业务命令参考”中的“IP 性能优化”）。

需要在目的端开启 ICMP 目的不可达报文发送功能。如果目的端是 H3C 设备，需要在设备上执行 **ip unreachable enable** 命令（该命令的详细介绍请参见“三层技术-IP 业务命令参考”中的“IP 性能优化”）。

### 4. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 创建 NQA 测试组，并进入 NQA 测试组视图。

**nqa entry** *admin-name operation-tag*

- (3) 配置测试类型为 UDP-tracert，并进入测试类型视图。

**type udp-tracert**

- (4) 配置探测报文的目的主机名或目的 IP 地址。请选择其中一项进行配置。

- 配置探测报文的目的主机名。

**destination host** *host-name*

缺省情况下，未配置探测报文的目的主机名。

- 配置探测报文的目的 IP 地址。

**destination ip** *ip-address*

缺省情况下，未配置探测报文的目的 IP 地址。

- (5) 配置测试操作的目的端口。

**destination port** *port-number*

缺省情况下，测试操作的目的端口号为 33434。

该端口必须是对端设备上未启用的端口，这样对端设备会回复目的端口不可达的 ICMP 差错报文。

- (6) 配置探测报文的出接口。

**out interface** *interface-type interface-number*

缺省情况下，设备通过查询路由表信息确认探测报文出接口。

- (7) 配置探测报文的源 IP 地址。请选择其中一项进行配置。

- 配置使用指定接口的 IP 地址作为探测报文的源 IP 地址。

**source interface** *interface-type interface-number*

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

该命令指定的接口必须为 up 状态，否则测试会失败。

- 配置探测报文的源 IP 地址。

**source ip** *ip-address*

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

该命令指定的源地址必须是设备上接口的 IP 地址，且接口为 up 状态，否则测试会失败。

- (8) 配置探测报文的源端口。

**source port** *port-number*

缺省情况下，系统自动选择设备当前空闲的端口号。

- (9) 配置测试最大连续失败次数。

**max-failure** *times*

缺省情况下，最大失败次数为 5。

- (10) 配置发送的探测报文的初始跳数。

**init-ttl** *value*

缺省情况下，UDP-tracert 测试中探测报文初始跳数为 1。

- (11) （可选）配置探测报文中的填充内容大小。

**data-size** *size*

缺省情况下，探测报文中的填充内容大小为 100 字节。

- (12) （可选）配置探测的禁止报文分片功能。

**no-fragment** *enable*

缺省情况下，禁止报文分片功能处于关闭状态。



## 1.5.13 配置Voice测试

### 1. 功能简介

Voice 测试主要用来测试 VoIP（Voice over IP，在 IP 网络上传送语音）网络情况，统计 VoIP 网络参数，以使用户根据网络情况进行相应的调整。

Voice 测试的过程如下：

- (1) 源端（NQA 客户端）以一定的时间间隔向目的端（NQA 服务器）发送 G.711 A 律、G.711  $\mu$  律或 G.729 A 律编码格式的语音数据包。
- (2) 目的端收到语音数据包后，为它打上时间戳，并把带有时间戳的数据包发送给源端。
- (3) 源端收到数据包后，根据数据包上的时间戳等信息，计算出抖动时间、单向延迟等网络参数，从而清晰地反映出网络状况。

对于 Voice 测试，一次探测操作是指连续发送多个探测报文，发送探测报文的个数由 `probe packet-number` 命令来设定。

除了抖动时间等参数，Voice 测试还可以计算出反映 VoIP 网络状况的语音参数值：

- ICPIF（Calculated Planning Impairment Factor，计算计划损伤元素）：用来量化网络中语音数据的衰减，由单向网络延迟和丢包率等决定。数值越大，表明语音网络质量越差。
- MOS（Mean Opinion Scores，平均意见得分）：语音网络的质量得分。MOS 值的范围为 1～5，该值越高，表明语音网络质量越好。通过计算网络中语音数据的衰减——ICPIF 值，可以估算出 MOS 值。

用户对语音质量的评价具有一定的主观性，不同用户对语音质量的容忍程度不同，因此，衡量语音质量时，需要考虑用户的主观因素。对语音质量容忍程度较强的用户，可以通过 `advantage-factor` 命令配置补偿因子，在计算 ICPIF 值时将减去该补偿因子，修正 ICPIF 和 MOS 值，以便在比较语音质量时综合考虑客观和主观因素。

Voice测试需要NQA服务器和客户端配合才能完成。进行Voice测试之前，必须保证NQA服务器端配置了UDP监听功能，配置方法请参见“[1.3 配置NQA服务器](#)”。

### 2. 配置限制和指导

建议不要对知名端口，即 1～1023 之间的端口，进行 Voice 测试，否则可能导致 NQA 测试失败或该知名端口对应的服务不可用。

`display nqa history` 命令的显示信息无法反映 Voice 测试的结果，如果了解 Voice 测试的结果，建议通过 `display nqa result` 命令查看最近一次 NQA 测试的当前结果，或通过 `display nqa statistics` 命令查看 NQA 测试的统计信息。

进行本测试前需保证网络时钟的 NTP 同步。有关 NTP 的详细介绍请参见“网络管理和监控配置指导”的“NTP”。

### 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 创建 NQA 测试组，并进入 NQA 测试组视图。

```
nqa entry admin-name operation-tag
```

- (3) 配置测试类型为 Voice，并进入测试类型视图。

```
type voice
```



- (4) 配置探测报文的目的 IP 地址。

**destination ip** *ip-address*

缺省情况下，未配置探测报文的目的 IP 地址。

必须与 NQA 服务器上配置的监听服务的 IP 地址一致。

- (5) 配置测试操作的目的端口。

**destination port** *port-number*

缺省情况下，未配置测试操作的目的端口号。

必须与 NQA 服务器上配置的监听服务的端口号一致。

- (6) 配置探测报文的源 IP 地址。

**source ip** *ip-address*

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

该命令指定的源地址必须是设备上接口的 IP 地址，且接口为 up 状态，否则测试将会失败。

- (7) 配置探测报文的源端口。

**source port** *port-number*

缺省情况下，系统自动选择设备当前空闲的端口号。

- (8) 配置 Voice 测试的基本参数。

- 配置 Voice 测试的编码格式。

**codec-type** { **g711a** | **g711u** | **g729a** }

缺省情况下，语音编码格式为 G.711 A 律。

- 配置用于计算 MOS 值和 ICPIF 值的补偿因子。

**advantage-factor** *factor*

缺省情况下，补偿因子取值为 0。

- (9) 配置 Voice 测试的探测参数。

- 配置一次 Voice 探测中发送探测报文的个数。

**probe packet-number** *packet-number*

缺省情况下，一次 Voice 探测中发送 1000 个探测报文。

- 配置 Voice 探测中发送探测报文的时间间隔。

**probe packet-interval** *interval*

缺省情况下，Voice 探测中发送探测报文的时间间隔为 20 毫秒。

- 配置 Voice 测试中等待响应报文的超时时间。

**probe packet-timeout** *timeout*

缺省情况下，Voice 测试中等待响应报文的超时时间为 5000 毫秒。

- (10) 配置探测报文中的填充内容。

- a. 配置探测报文中的填充内容大小。

**data-size** *size*

缺省情况下，探测报文中的填充内容大小与配置的编码格式有关，编码格式为 **g.711a** 和 **g.711u** 时缺省报文大小为 172 字节，**g.729a** 时为 32 字节。

- b. （可选）配置探测报文中的填充字符串。

**data-fill** *string*

探测报文的填充内容为十六进制 00010203040506070809。

## 1.5.14 配置DLSw测试

### 1. 功能简介

DLSw 测试主要用来测试 DLSw 设备的响应时间。

对于 DLSw 测试，一次探测操作是指建立一次 DLSw 连接。

### 2. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 创建 NQA 测试组，并进入 NQA 测试组视图。

**nqa entry** *admin-name operation-tag*

- (3) 配置测试类型为 DLSw，并进入测试类型视图。

**type dlsw**

- (4) 配置探测报文的目的 IP 地址。

**destination ip** *ip-address*

缺省情况下，未配置探测报文的目的 IP 地址。

- (5) 配置探测报文的源 IP 地址。

**source ip** *ip-address*

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

该命令指定的源地址必须是设备上接口的 IP 地址，且接口为 up 状态，否则测试将会失败。

## 1.5.15 配置Path-jitter测试

### 1. 功能简介

Path-jitter 测试可以作为 UDP-jitter 测试的一种补充，用于在抖动比较大的情况下，进一步探测中间路径的网络质量，以便查找出网络质量差的具体路段。Path-jitter 测试项对每一条路径记录结果，在路径上的每一跳均记录抖动值、正向抖动值和负向抖动值。

Path-jitter 测试的过程如下：

- (1) NQA 客户端使用 **tracert** 机制发现到达目的地址的路径信息。
- (2) NQA 客户端根据 **tracert** 结果，逐跳使用 ICMP 机制探测从本机至该跳设备的路径上报文是否有丢失，同时计算该跳路径的时延和抖动时间等信息。

对于 Path-jitter 测试，一次探测操作分为两个步骤：首先通过 **tracert** 探路获取到达目的地址的路径（最大为 64 跳）；再根据 **tracert** 结果，分别向路径上的每一跳发送多个 ICMP-echo 探测报文，发送探测报文的个数由 **probe packet-number** 命令来设定。

### 2. 配置准备

配置 Path-jitter 测试需要在中间设备（源端与目的端之间的设备）上开启 ICMP 超时报文发送功能。如果中间设备是 H3C 设备，需要在设备上执行 **ip ttl-expires enable** 命令（该命令的详细介绍请参见“三层技术-IP 业务命令参考”中的“IP 性能优化”）。需要在目的端开启 ICMP 目的不

可达报文发送功能。如果目的端是 H3C 设备，需要在设备上执行 **ip unreachable enable** 命令（该命令的详细介绍请参见“三层技术-IP 业务命令参考”中的“IP 性能优化”）。

### 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 创建 NQA 测试组，并进入 NQA 测试组视图。

```
nqa entry admin-name operation-tag
```

- (3) 配置测试类型为 Path-jitter，并进入测试类型视图。

```
type path-jitter
```

- (4) 配置探测报文的目的 IP 地址。

```
destination ip ip-address
```

缺省情况下，未配置探测报文的目的 IP 地址。

- (5) 配置探测报文的源 IP 地址。

```
source ip ip-address
```

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

该命令指定的源地址必须是设备上接口的 IP 地址，且接口为 up 状态，否则探测将会失败。

- (6) 配置 Path-jitter 测试的探测参数。

- 配置一次 Path-jitter 探测中发送探测报文的个数。

```
probe packet-number packet-number
```

缺省情况下，一次 Path-jitter 探测中发送 10 个 ICMP 探测报文。

- 配置 Path-jitter 测试中发送探测报文的时间间隔。

```
probe packet-interval interval
```

缺省情况下，Path-jitter 测试中发送探测报文的时间间隔为 20 毫秒。

- 配置 Path-jitter 测试中等待响应报文的超时时间。

```
probe packet-timeout timeout
```

缺省情况下，Path-jitter 测试中等待响应报文的超时时间为 3000 毫秒。

- (7) （可选）配置松散路由。

```
lsr-path ip-address&<1-8>
```

缺省情况下，未配置松散路由。

通过本命令配置松散路由，在 **tracert** 过程使用该配置进行探路，NQA 客户端根据该松散路由计算时延和抖动时间。

- (8) （可选）配置仅对目的地址探测。

```
target-only
```

缺省情况下，未配置仅对目的地址探测，Path-jitter 测试中会逐跳进行探测。

- (9) （可选）配置探测报文中的填充内容大小。

```
data-size size
```

缺省情况下，探测报文中的填充内容大小为 100 字节。

- (10) （可选）配置探测报文的填充字符串。

**data-fill** *string*

探测报文的填充内容为十六进制 00010203040506070809。

## 1.5.16 配置NQA测试组通用参数

### 1. 配置限制和指导

NQA 测试组的通用参数，只对当前测试组中的测试有效。

除特别说明外，所有测试类型都可以根据实际情况选择配置下列通用参数。

通用参数中路径服务质量测试目前仅支持 **description** 及 **tos** 命令。

### 2. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 进入已配置测试类型的 NQA 测试组视图。

**nqa entry** *admin-name operation-tag*

- (3) 配置测试组的描述信息。

**description** *text*

缺省情况下，未配置描述信息。

- (4) 配置测试组连续两次测试开始时间的时间间隔。

**frequency** *interval*

缺省情况下，Voice、Path-jitter 测试中连续两次测试开始时间的时间间隔为 60000 毫秒；其他类型的测试为 0 毫秒，即只进行一次测试。当到达指定的时间间隔而上次测试尚未完成时，不启动新一轮测试。

- (5) 配置一次 NQA 测试中进行探测的次数。

**probe count** *times*

缺省情况下，对于 UDP-tracert 测试类型，对于一个 TTL 值的节点发送的探测报文次数为 3 次；其他类型的 NQA 测试一次测试中的探测次数为 1 次。

Voice 和 Path-jitter 测试中探测次数只能为 1，不支持该命令。

- (6) 配置 NQA 探测超时时间。

**probe timeout** *timeout*

缺省情况下，探测的超时时间为 3000 毫秒。

ICMP-jitter、UDP-jitter、Voice 和 Path-jitter 测试不能配置该参数。

- (7) 配置探测报文在网络中可以经过的最大跳数。

**ttl** *value*

缺省情况下，UDP-tracert 测试探测报文在网络中可以经过的最大跳数为 30 跳。其他类型的探测报文在网络中可以经过的最大跳数为 20 跳。

DHCP 和 Path-jitter 测试不能配置该参数。

- (8) 配置 NQA 探测报文 IP 报文头中服务类型域的值。

**tos** *value*

缺省情况下，NQA 探测报文 IP 报文头中服务类型域的值为 0。

- (9) 启动路由表旁路功能。

**route-option bypass-route**

缺省情况下，路由表旁路功能处于关闭状态。

DHCP 和 Path-jitter 测试不能配置该参数。

测试目的端使用 IPv6 地址时，本命令配置无效。

## 1.5.17 配置联动功能

### 1. 功能简介

联动功能是通过建立联动项，对当前所在测试组中的探测进行监测，当连续探测失败次数达到阈值时，就触发配置的动作类型。

### 2. 配置限制和指导

ICMP-jitter、UDP-jitter、UDP-tracert、Voice、Path-jitter 和路径服务质量测试不支持联动功能。

### 3. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 进入已配置测试类型的 NQA 测试组视图。

**nqa entry admin-name operation-tag**

- (3) 建立联动项。

**reaction item-number checked-element probe-fail threshold-type  
consecutive consecutive-occurrences action-type trigger-only**

联动项创建后，不能再通过 **reaction** 命令修改该联动项的内容。

- (4) 退回系统视图。

**quit**

- (5) 配置 Track 与 NQA 联动。

配置方法请参见“可靠性配置指导”中的“Track”

- (6) 配置 Track 与应用模块联动。

配置方法请参见“可靠性配置指导”中的“Track”

## 1.5.18 配置阈值告警功能

### 1. 功能简介

NQA 通过创建阈值告警项，并在阈值告警项中配置监测的对象、阈值类型及触发的动作，来实现阈值告警功能。

NQA 阈值告警功能支持的阈值类型包括：

- 平均值（**average**）：监测一次测试中探测结果的平均值，如果平均值不在指定的范围内，则该监测对象超出阈值。例如，监测一次测试中探测持续时间的平均值。
- 累计数目（**accumulate**）：监测一次测试中探测结果不在指定范围内的累计数目，如果累计数目达到或超过设定的值，则该监测对象超出阈值。

- 连续次数 (**consecutive**): NQA 测试组启动后, 监测探测结果连续不在指定范围内的次数, 如果该次数达到或超过设定的值, 则该监测对象超出阈值。

NQA 阈值告警功能可以触发如下动作:

- **none**: 只在本地记录监测结果, 以便通过显示命令查看, 不向网络管理系统发送 Trap 消息。
- **trap-only**: 不仅在本地记录监测结果, 当阈值告警项的状态改变时, 还向网络管理系统发送 Trap 消息。采用本动作时, 需要通过 **snmp-agent target-host** 命令配置 Trap 消息的目的地址。**snmp-agent target-host** 命令的详细介绍, 请参见“网络管理和监控命令参考”中的“SNMP”。
- **trigger-only**: 在显示信息中记录监测结果的同时, 触发其他模块联动。

阈值告警项包括 **invalid**、**over-threshold** 和 **below-threshold** 三种状态:

- NQA 测试组未启动时, 阈值告警项的状态为 **invalid**。
- NQA 测试组启动后, 每次测试或探测结束时, 检查监测的对象是否超出指定类型的阈值。如果超出阈值, 则阈值告警项的状态变为 **over-threshold**; 如果未超出阈值, 则状态变为 **below-threshold**。

## 2. 配置限制和指导

Path-jitter 和路径服务质量测试不支持配置阈值告警功能。

## 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入已配置测试类型的 NQA 测试组视图。

```
nqa entry admin-name operation-tag
```

- (3) 配置在指定条件下向网管服务器发送 Trap 消息。

```
reaction trap { path-change | probe-failure consecutive-probe-failures
| test-complete | test-failure [ accumulate-probe-failures ] }
```

缺省情况下, 不向网管服务器发送 Trap 消息。

ICMP-jitter、UDP-jitter、Voice 测试只支持 **reaction trap test-complete**。

UDP-tracert 测试不支持 **probe-failure** 和 **accumulate-probe-failures** 参数。

- (4) 创建阈值告警组。请至少选择其中一项进行配置。

- 创建监测探测持续时间的阈值告警组。

```
reaction item-number checked-element probe-duration threshold-type
{ accumulate accumulate-occurrences | average | consecutive
consecutive-occurrences } threshold-value upper-threshold
lower-threshold [ action-type { none | trap-only } ]
```

除 ICMP-jitter、UDP-jitter、UDP-tracert 和 Voice 测试外, 均支持。

- 创建监测探测失败次数的阈值告警组。

```
reaction item-number checked-element probe-fail threshold-type
{ accumulate accumulate-occurrences | consecutive
consecutive-occurrences } [ action-type { none | trap-only } ]
```

除 ICMP-jitter、UDP-jitter、UDP-tracert 和 Voice 测试外，均支持。

- 创建监测报文往返时延的阈值告警组。

```
reaction item-number checked-element rtt threshold-type { accumulate  
accumulate-occurrences | average } threshold-value upper-threshold  
lower-threshold [ action-type { none | trap-only } ]
```

仅 ICMP-jitter、UDP-jitter 和 Voice 测试支持。

- 创建监测每次测试中丢包数的阈值告警组。

```
reaction item-number checked-element packet-loss threshold-type  
accumulate accumulate-occurrences [ action-type { none | trap-only } ]
```

仅 ICMP-jitter、UDP-jitter 和 Voice 测试支持。

- 创建监测单向抖动时间的阈值告警组。

```
reaction item-number checked-element { jitter-ds | jitter-sd }  
threshold-type { accumulate accumulate-occurrences | average }  
threshold-value upper-threshold lower-threshold [ action-type { none  
| trap-only } ]
```

仅 ICMP-jitter、UDP-jitter 和 Voice 测试支持。

- 创建监测单向时延的阈值告警组。

```
reaction item-number checked-element { owd-ds | owd-sd }  
threshold-value upper-threshold lower-threshold
```

仅 ICMP-jitter、UDP-jitter 和 Voice 测试支持。

- 创建监测 Voice 测试 ICPIF 值的阈值告警组。

```
reaction item-number checked-element icpif threshold-value  
upper-threshold lower-threshold [ action-type { none | trap-only } ]
```

仅 Voice 测试支持。

- 创建监测 Voice 测试 MOS 值的阈值告警组。

```
reaction item-number checked-element mos threshold-value  
upper-threshold lower-threshold [ action-type { none | trap-only } ]
```

仅 Voice 测试支持。

DNS 测试不支持发送 Trap 消息，即对于 DNS 测试，触发动作只能配置为 none。

## 1.5.19 配置NQA统计功能

### 1. 功能简介

NQA 将在指定时间间隔内完成的 NQA 测试归为一组，计算该组测试结果的统计值，这些统计值构成一个统计组。通过 **display nqa statistics** 命令可以显示该统计组的信息。

当 NQA 设备上保留的统计组数目达到最大值时，如果形成新的统计组，保存时间最久的统计组将被删除。

统计组具有老化功能，即统计组保存一定时间后，将被删除。

### 2. 配置限制和指导

- UDP-tracert 和路径服务质量测试不支持 NQA 统计功能。



- 如果通过 **frequency** 命令指定连续两次测试开始时间的时间间隔为 0，则不生成统计组信息。

### 3. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 进入已配置测试类型的 NQA 测试组视图。

**nqa entry admin-name operation-tag**

- (3) 配置对测试结果进行统计的时间间隔。

**statistics interval interval**

缺省情况下，对测试结果进行统计的时间间隔为 60 分钟。

- (4) 配置能够保留的最大统计组个数。

**statistics max-group number**

缺省情况下，能够保留的最大统计组数为 2。

最大统计组个数为 0 时，不进行统计。

- (5) 配置统计组的保留时间。

**statistics hold-time hold-time**

缺省情况下，统计组的保留时间为 120 分钟。

## 1.5.20 配置NQA历史记录功能

### 1. 功能简介

开启 NQA 测试组的历史记录保存功能后，系统将记录 NQA 测试的历史信息，通过 **display nqa history** 命令可以查看该测试组的历史记录信息。

### 2. 配置限制和指导

ICMP-jitter、UDP-jitter、Voice、Path-jitter 和路径服务质量测试不支持配置历史记录功能。

### 3. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 进入已配置测试类型的 NQA 测试组视图。

**nqa entry admin-name operation-tag**

- (3) 开启 NQA 测试组的历史记录保存功能。

**history-record enable**

缺省情况下，UDP-tracert 类型测试组的历史记录保存功能处于开启状态，其他类型的 NQA 测试组的历史记录保存功能处于关闭状态。

- (4) 配置 NQA 测试组中历史记录保存时间。

**history-record keep-time keep-time**

缺省情况下，NQA 测试组中历史记录的保存时间为 120 分钟。

历史记录保存时间达到配置的值后，该历史记录将被删除。



- (5) 配置在一个测试组中能够保存的最大历史记录个数。

**history-record number** *number*

缺省情况下，一个测试组中能够保存的最大历史记录个数为 50。

如果历史记录个数超过设定的最大数目，则最早的历史记录将会被删除。

## 1.5.21 在NQA客户端上调度NQA测试组

### 1. 功能简介

通过本配置，可以设置测试组进行测试的启动时间和持续时间。

系统时间在<启动时间>到<启动时间+持续时间>范围内时，测试组进行测试。执行 **nqa schedule** 命令时：

- 如果系统时间尚未到达启动时间，则到达启动时间后，启动测试；
- 如果系统时间在<启动时间>到<启动时间+持续时间>之间，则立即启动测试；
- 如果系统时间已经超过<启动时间+持续时间>，则不会启动测试。

通过 **display clock** 命令可以查看系统的当前时间。

### 2. 配置限制和指导

测试组被调度后就不能再进入该测试组视图和测试类型视图。

对于已启动的测试组或已经完成测试的测试组，不受系统时间调整的影响，只有等待测试的测试组受系统时间调整的影响。

### 3. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 在 NQA 客户端上调度 NQA 测试组。

```
nqa schedule admin-name operation-tag start-time { hh:mm:ss  
[ yyyy/mm/dd | mm/dd/yyyy ] | now } lifetime { lifetime | forever }  
[ recurring ]
```

## 1.6 在NQA客户端上配置NQA模板

### 1.6.1 配置限制和指导

对于 NQA 各类型模板，某些测试参数既可以由外部特性提供，也可以手工直接进行配置。若同时通过以上两种方式获取到测试参数，则以手工配置的测试信息为准。

### 1.6.2 NQA模板配置任务简介

NQA 模板配置任务如下：

- (1) 配置 NQA 模板

- [配置ICMP类型的NQA模板](#)
- [配置DNS类型的NQA模板](#)
- [配置TCP类型的NQA模板](#)

- [配置TCP Half Open类型的NQA模板](#)
  - [配置UDP类型的NQA模板](#)
  - [配置HTTP类型的NQA模板](#)
  - [配置HTTPS类型的NQA模板](#)
  - [配置FTP类型的NQA模板](#)
  - [配置RADIUS类型的NQA模板](#)
  - [配置SSL类型的NQA模板](#)
- (2) (可选) [配置NQA模板通用参数](#)

### 1.6.3 配置ICMP类型的NQA模板

#### 1. 功能简介

ICMP 类型的 NQA 模板为外部特性提供 ICMP 类型的测试，外部特性通过引用该模板来启动 ICMP 测试，并根据是否接收到 ICMP 应答报文判断目的主机的可达性。ICMP 类型的 NQA 模板支持 IPv4 和 IPv6 网络。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 创建 ICMP 类型的 NQA 模板，并进入模板视图。

```
nqa template icmp name
```

- (3) 配置测试操作的目的地址。

(IPv4 网络)

```
destination ip ip-address
```

(IPv6 网络)

```
destination ipv6 ipv6-address
```

缺省情况下，未配置探测报文的目的地址。

- (4) 配置探测报文的源地址。请选择其中一项进行配置。

- 使用指定接口的 IP 地址作为探测报文的源 IP 地址。

```
source interface interface-type interface-number
```

缺省情况下，以报文发送接口的主 IP 地址作为探测报文中的源 IP 地址。

**source interface** 命令指定的接口必须为 up 状态。

- 配置探测报文的源 IPv4 地址。

```
source ip ip-address
```

缺省情况下，以报文发送接口的主 IPv4 地址作为探测报文中的源 IPv4 地址。

该命令指定的源地址必须是设备上接口的 IP 地址，且接口为 up 状态，否则测试将会失败。

- 配置探测报文的源 IPv6 地址。

```
source ipv6 ipv6-address
```

缺省情况下，以报文发送接口的 IPv6 地址作为探测报文中的源 IPv6 地址。

该命令指定的源地址必须是设备上接口的 IPv6 地址，且接口为 up 状态，否则测试将会失败。

- (5) 配置探测报文的下一跳地址。

(IPv4 网络)

**next-hop ip** *ip-address*

(IPv6 网络)

**next-hop ipv6** *ipv6-address*

缺省情况下，未配置探测报文的下一跳地址。

- (6) 配置每次探测结束时都将探测结果发送给外部特性。

**reaction trigger per-probe**

缺省情况下，连续探测成功或失败 3 次时，NQA 客户端会把探测成功或失败的消息发送给外部特性，使外部特性利用 NQA 测试的结果进行相应处理。

**reaction trigger per-probe** 命令与 **reaction trigger probe-pass** 命令作用相同，多次执行这两条命令时，最后一次执行的命令生效。

**reaction trigger per-probe** 命令与 **reaction trigger probe-fail** 命令作用相同，多次执行这两条命令时，最后一次执行的命令生效。

- (7) (可选) 配置探测报文中的填充内容大小。

**data-size** *size*

缺省情况下，探测报文中的填充内容大小为 100 字节。

- (8) (可选) 配置探测报文的填充字符串。

**data-fill** *string*

探测报文的填充内容为十六进制 00010203040506070809。

## 1.6.4 配置DNS类型的NQA模板

### 1. 功能简介

DNS 类型的 NQA 模板为外部特性提供 DNS 类型的测试。外部特性通过引用该模板来启动 DNS 测试，NQA 客户端向指定的 DNS 服务器发送 DNS 请求报文，NQA 客户端通过是否收到应答及应答报文的合法性来确定服务器的状态。DNS 类型的 NQA 模板支持 IPv4 和 IPv6 网络。

在 DNS 类型的 NQA 模板视图下，用户可以配置期望返回的地址。如果 DNS 服务器返回的 IP 地址中包含了期望地址，则该 DNS 服务器为真实的服务器，测试成功；否则，测试失败。

### 2. 配置准备

在进行 DNS 测试之前，需要在 DNS 服务器上创建域名和地址的映射关系。DNS 服务器配置方法，请参见 DNS 服务器相关资料。

### 3. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 创建 DNS 类型的 NQA 模板，并进入模板视图。

**nqa template dns** *name*

- (3) 配置测试操作的地址。

(IPv4 网络)

**destination ip** *ip-address*

(IPv6 网络)

**destination ipv6** *ipv6-address*

缺省情况下，未配置探测报文的地址。

- (4) 配置测试操作的端口。

**destination port** *port-number*

缺省情况下，测试操作的端口号为 53。

- (5) 配置探测报文的源地址。

(IPv4 网络)

**source ip** *ip-address*

缺省情况下，以报文发送接口的主 IPv4 地址作为探测报文中的源 IPv4 地址。

该命令指定的源地址必须是设备上接口的 IPv4 地址，且接口为 up 状态，否则测试将会失败。

(IPv6 网络)

**source ipv6** *ipv6-address*

缺省情况下，以报文发送接口的 IPv6 地址作为探测报文中的源 IPv6 地址。

该命令指定的源地址必须是设备上接口的 IPv6 地址，且接口为 up 状态，否则测试将会失败。

- (6) 配置探测报文的源端口。

**source port** *port-number*

缺省情况下，未配置探测报文的源端口号。

- (7) 配置要解析的域名。

**resolve-target** *domain-name*

缺省情况下，没有配置要解析的域名。

- (8) 配置域名解析类型。

**resolve-type** { **A** | **AAAA** }

缺省情况下，域名解析类型为 A 类型。

其中 A 类型表示将域名解析为 IPv4 地址，AAAA 类型表示将域名解析为 IPv6 地址。

- (9) (可选) 配置用户期望返回的地址。

(IPv4 网络)

**expect ip** *ip-address*

(IPv6 网络)

**expect ipv6** *ipv6-address*

缺省情况下，未设定期望返回的地址。

## 1.6.5 配置TCP类型的NQA模板

### 1. 功能简介

TCP 类型的 NQA 模板为外部特性提供 TCP 类型测试，外部特性通过引用该模板，测试客户端和服务端指定端口之间能否建立 TCP 连接。

在 TCP 类型的 NQA 模板视图下，用户可以配置期望的应答内容。如果用户未配置期望的应答内容，则 NQA 客户端与服务器间只建立 TCP 连接。

TCP 测试需要 NQA 服务器和客户端配合才能完成。在 TCP 测试之前，需要在 NQA 服务器端配置 TCP 监听功能。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 创建 TCP 类型的 NQA 模板，并进入模板视图。

```
nqa template tcp name
```

- (3) 配置测试操作的目的地址。

(IPv4 网络)

```
destination ip ip-address
```

(IPv6 网络)

```
destination ipv6 ipv6-address
```

缺省情况下，未配置探测报文的目的地址。

必须与 NQA 服务器上配置的监听服务的 IP 地址一致。

- (4) 配置测试操作的目的端口。

```
destination port port-number
```

缺省情况下，未配置测试操作的目的端口号。

必须与 NQA 服务器上配置的监听服务的端口号一致。

- (5) 配置探测报文的源地址。

(IPv4 网络)

```
source ip ip-address
```

缺省情况下，以报文发送接口的主 IPv4 地址作为探测报文中的源 IPv4 地址。

该命令指定的源地址必须是设备上接口的 IPv4 地址，且接口为 up 状态，否则测试将会失败。

(IPv6 网络)

```
source ipv6 ipv6-address
```

缺省情况下，以报文发送接口的 IPv6 地址作为探测报文中的源 IPv6 地址。

该命令指定的源地址必须是设备上接口的 IPv6 地址，且接口为 up 状态，否则测试将会失败。

- (6) (可选) 配置探测报文的填充字符串。

```
data-fill string
```

探测报文的填充内容为十六进制 00010203040506070809。

- (7) (可选) 配置用户期望的应答内容。

**expect data** *expression* [ **offset number** ]

缺省情况下, 未配置期望的应答内容。

仅当 **data-fill** 和 **expect data** 命令都配置时, 进行期望应答内容的检查, 否则不做检查。

## 1.6.6 配置TCP Half Open类型的NQA模板

### 1. 功能简介

TCP Half Open 类型的 NQA 模板为外部特性提供 TCP Half Open 类型测试。作为 TCP 测试的补充, TCP Half Open 测试不需要指定目的端端口。当外部特性的现有 TCP 连接无法得到对端应答时, 可以引用 TCP Half Open 模板进行测试。开启测试后, NQA 客户端将主动向对端发出 TCP ACK 报文, 以是否能收到对端返回的 RST 报文来判断对端的 TCP 服务是否可用。

### 2. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 创建 TCP Half Open 类型的 NQA 模板, 并进入模板视图。

**nqa template tcp-half-open** *name*

- (3) 配置测试操作的目的地址。

(IPv4 网络)

**destination ip** *ip-address*

(IPv6 网络)

**destination ipv6** *ipv6-address*

缺省情况下, 未配置探测报文的目的地址。

必须与 NQA 服务器上配置的监听服务的 IP 地址一致。

- (4) 配置探测报文的源地址。

(IPv4 网络)

**source ip** *ip-address*

缺省情况下, 以报文发送接口的主 IPv4 地址作为探测报文中的源 IPv4 地址。

该命令指定的源地址必须是设备上接口的 IPv4 地址, 且接口为 up 状态, 否则测试将会失败。

(IPv6 网络)

**source ipv6** *ipv6-address*

缺省情况下, 以报文发送接口的 IPv6 地址作为探测报文中的源 IPv6 地址。

该命令指定的源地址必须是设备上接口的 IPv6 地址, 且接口为 up 状态, 否则测试将会失败。

- (5) 配置探测报文的下一跳地址。

(IPv4 网络)

**next-hop ip** *ip-address*

(IPv6 网络)

**next-hop ipv6** *ipv6-address*

缺省情况下，未配置探测报文的下一跳地址。

- (6) 配置每次探测结束时都将探测结果发送给外部特性。

**reaction trigger per-probe**

缺省情况下，连续探测成功或失败 3 次时，NQA 客户端会把探测成功或失败的消息发送给外部特性，使外部特性利用 NQA 测试的结果进行相应处理。

**reaction trigger per-probe** 命令与 **reaction trigger probe-pass** 命令作用相同，多次执行这两条命令时，最后一次执行的命令生效。

**reaction trigger per-probe** 命令与 **reaction trigger probe-fail** 命令作用相同，多次执行这两条命令时，最后一次执行的命令生效。

## 1.6.7 配置UDP类型的NQA模板

### 1. 功能简介

UDP 类型的 NQA 模板为外部特性提供 UDP 类型测试，外部特性通过引用该模板，测试客户端和服务器指定端口之间 UDP 传输的联通性。NQA 客户端通过处理服务器端的应答报文，判断服务器指定端口上提供的服务是否可用。

在 UDP 类型的 NQA 模板视图下，用户可以配置期望的应答内容。如果用户未配置期望的应答内容，则 NQA 客户端只要收到合法的回应报文就认为探测成功。

UDP 测试需要 NQA 服务器和客户端配合才能完成。在进行 UDP 测试前，需要在 NQA 服务器端配置 UDP 监听服务。

### 2. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 创建 UDP 类型的 NQA 模板，并进入模板视图。

**nqa template udp** *name*

- (3) 配置测试操作的目的地址。

(IPv4 网络)

**destination ip** *ip-address*

(IPv6 网络)

**destination ipv6** *ipv6-address*

缺省情况下，未配置探测报文的目地址。

必须与 NQA 服务器上配置的监听服务的 IP 地址一致。

- (4) 配置测试操作的目的端口。

**destination port** *port-number*

缺省情况下，未配置测试操作的目的端口号。

必须与 NQA 服务器上配置的监听服务的端口号一致。

- (5) 配置探测报文的源地址。

(IPv4 网络)

**source ip** *ip-address*

缺省情况下，以报文发送接口的主 IPv4 地址作为探测报文中的源 IPv4 地址。

该命令指定的源地址必须是设备上接口的 IPv4 地址，且接口为 up 状态，否则测试将会失败。

(IPv6 网络)

**source ipv6** *ipv6-address*

缺省情况下，以报文发送接口的 IPv6 地址作为探测报文中的源 IPv6 地址。

该命令指定的源地址必须是设备上接口的 IPv6 地址，且接口为 up 状态，否则测试将会失败。

- (6) (可选) 配置探测报文的填充字符串。

**data-fill** *string*

探测报文的填充内容为十六进制 00010203040506070809。

在未配置此命令情况下配置 **expect data** 命令则会探测失败。

- (7) (可选) 配置探测报文中的填充内容大小。

**data-size** *size*

缺省情况下，探测报文中的填充内容大小为 100 字节。

- (8) (可选) 配置用户期望的应答内容。

**expect data** *expression* [ **offset** *number* ]

缺省情况下，未配置期望的应答内容。

仅当 **data-fill** 和 **expect data** 命令都配置时，进行期望应答内容的检查，否则不做检查。

## 1.6.8 配置HTTP类型的NQA模板

### 1. 功能简介

HTTP 类型的 NQA 模板为外部特性提供 HTTP 类型测试，外部特性通过引用该模板，测试 NQA 客户端是否可以与指定的 HTTP 服务器建立连接，以及从 HTTP 服务器获取数据所需的时间，从而判断 HTTP 服务器的连通性及性能。

在 HTTP 类型的 NQA 模板中，用户可以配置期望返回的数据。通过该功能用户可以判断 HTTP 服务器应答报文的合法性。当应答报文中存在“Content-Length”字段，且配置了 **expect data** 命令时，设备将进行期望应答内容的检查。

在 HTTP 类型的 NQA 模板中，用户可以配置应答状态码。应答状态码是由 3 位十进制数组成的字段，它包含 HTTP 服务器的状态信息，用户可以根据该状态码了解 HTTP 服务器的状态。状态码的第一位表示状态码的类型。

### 2. 配置准备

在进行 HTTP 测试之前，需要完成 HTTP 服务器的配置。

### 3. 配置步骤

- (1) 进入系统视图。



## **system-view**

- (2) 创建 HTTP 类型的 NQA 模板，并进入模板视图。

**nqa template http** *name*

- (3) 配置 HTTP 测试访问的目的网址。

**url** *url*

缺省情况下，没有配置 HTTP 测试访问的网址。

*url* 配置形式为 **http://host/resource** 或 **http://host:port/resource**。

- (4) 配置 HTTP 登录用户名。

**username** *username*

缺省情况下，未配置 HTTP 登录用户名。

- (5) 配置 HTTP 登录密码。

**password** { **cipher** | **simple** } *string*

缺省情况下，未配置 HTTP 登录密码。

- (6) 配置 HTTP 所使用的协议版本。

**version** { **v1.0** | **v1.1** }

缺省情况下，HTTP 使用的版本为 **v1.0**。

- (7) 配置 HTTP 的操作方式。

**operation** { **get** | **post** | **raw** }

缺省情况下，HTTP 操作方式为 **get** 操作。

如果 HTTP 操作方式为 **raw** 操作，则向服务器发送的探测报文的内容为 **raw-request** 视图中的内容。

- (8) 配置 HTTP 测试请求报文。

- a. 进入 **raw-request** 视图。

**raw-request**

输入 **raw-request** 命令进入 **raw-request** 视图，每次进入视图原有报文内容清除。

- b. 配置 HTTP 测试请求报文内容。

逐个字符输入或拷贝粘贴请求报文内容。

缺省情况下，未配置 HTTP 测试请求报文内容。

- c. 保存输入内容并退回测试类型视图。

**quit**

当配置 HTTP 测试的操作类型为 **raw** 时，必须完成此操作且保证发送的测试报文正确有效。

- (9) 配置探测报文的源地址。

(IPv4 网络)

**source ip** *ip-address*

缺省情况下，以报文发送接口的主 IPv4 地址作为探测报文中的源 IPv4 地址。

该命令指定的源地址必须是设备上接口的 IPv4 地址，且接口为 **up** 状态，否则测试将会失败。

(IPv6 网络)

**source ipv6** *ipv6-address*

缺省情况下，以报文发送接口的 IPv6 地址作为探测报文中的源 IPv6 地址。

该命令指定的源地址必须是设备上接口的 IPv6 地址，且接口为 up 状态，否则测试将会失败。

- (10) （可选）配置期望的应答状态码。

**expect status** *status-list*

缺省情况下，未配置期望的应答状态码。

- (11) （可选）配置期望的应答内容。

**expect data** *expression* [ **offset** *number* ]

缺省情况下，未配置期望的应答内容。

## 1.6.9 配置HTTPS类型的NQA模板

### 1. 功能简介

HTTPS（Hypertext Transfer Protocol Secure，超文本传输协议的安全版本）是支持 SSL（Secure Sockets Layer，安全套接字层）协议的 HTTP 协议，通过 SSL 为 HTTP 协议提供安全保证。HTTPS 类型的 NQA 模板为外部特性提供 HTTPS 类型测试，外部特性通过引用该模板，测试 NQA 客户端是否可以与指定的 HTTPS 服务器建立连接，以及从 HTTPS 服务器获取数据所需的时间，从而判断 HTTPS 服务器的连通性及性能。

在 HTTPS 类型的 NQA 模板中，用户可以配置期望返回的数据。通过该功能用户可以判断 HTTPS 服务器应答报文的合法性。当应答报文中存在“Content-Length”字段，且配置了 **expect data** 命令时，设备将进行期望应答内容的检查。

在 HTTPS 类型的 NQA 模板中，用户可以配置应答状态码。应答状态码是由 3 位十进制数组成的字段，它包含 HTTPS 服务器的状态信息，用户可以根据该状态码了解 HTTPS 服务器的状态。状态码的第一位表示状态码的类型。

### 2. 配置准备

在进行 HTTPS 测试之前，需要在测试客户端完成 SSL 客户端策略配置，以及在目的端完成 HTTPS 服务器的配置。SSL 客户端策略的配置方法请参见“安全配置指导”中的“SSL”。

### 3. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 创建 HTTPS 类型的 NQA 模板，并进入模板视图。

**nqa template https** *name*

- (3) 配置 HTTPS 测试访问的目的网址。

**url** *url*

缺省情况下，没有配置 HTTPS 测试访问的网址。

*url* 参数的格式为 **https://host/resource** 或 **https://host:port/resource**。

- (4) 配置 HTTPS 登录用户名。

**username** *username*

缺省情况下，未配置 HTTPS 登录用户名。

- (5) 配置 HTTPS 登录密码。

```
password { cipher | simple } string
```

缺省情况下，未配置 HTTPS 登录密码。

- (6) 绑定 SSL 客户端策略。

```
ssl-client-policy policy-name
```

缺省情况下，未绑定 SSL 客户端策略。

- (7) 配置 HTTPS 所使用的协议版本。

```
version { v1.0 | v1.1 }
```

缺省情况下，HTTPS 使用的版本为 v1.0。

- (8) 配置 HTTPS 的操作方式。

```
operation { get | post | raw }
```

缺省情况下，HTTPS 操作方式为 **get** 操作。

如果 HTTP 操作方式为 **raw** 操作，则向服务器发送的探测报文的内容为 **raw-request** 视图中的内容。

- (9) 配置 HTTPS 测试请求报文。

- a. 进入 **raw-request** 视图。

```
raw-request
```

输入 **raw-request** 命令进入 **raw-request** 视图，每次进入视图原有报文内容清除。

- b. 配置 HTTPS 测试请求报文内容。

逐个字符输入或拷贝粘贴请求报文内容。

缺省情况下，未配置 HTTPS 测试请求报文内容。

- c. 保存输入内容并退回测试类型视图。

```
quit
```

当配置 HTTPS 测试的操作类型为 **raw** 时，必须完成此操作且保证发送的测试报文正确有效。

- (10) 配置探测报文的源地址。

(IPv4 网络)

```
source ip ip-address
```

缺省情况下，以报文发送接口的主 IPv4 地址作为探测报文中的源 IPv4 地址。

该命令指定的源地址必须是设备上接口的 IPv4 地址，且接口为 **up** 状态，否则测试将会失败。

(IPv6 网络)

```
source ipv6 ipv6-address
```

缺省情况下，以报文发送接口的 IPv6 地址作为探测报文中的源 IPv6 地址。

该命令指定的源地址必须是设备上接口的 IPv6 地址，且接口为 **up** 状态，否则测试将会失败。

- (11) (可选) 配置期望的应答内容。

**expect data** *expression* [ **offset** *number* ]

缺省情况下，未配置期望的应答内容。

- (12) （可选）配置期望的应答状态码。

**expect status** *status-list*

缺省情况下，未配置期望的应答状态码。

## 1.6.10 配置FTP类型的NQA模板

### 1. 功能简介

FTP 类型的 NQA 模板为外部特性提供 FTP 类型测试，外部特性通过引用该模板，与指定的 FTP 服务器建立连接，以及与 FTP 服务器之间传送文件的时间，从而判断 FTP 服务器的连通性及性能。在进行 FTP 测试之前，需要在 FTP 服务器上进行相应的配置，包括 FTP 客户端登录 FTP 服务器的用户名、密码等。FTP 服务器的配置方法，请参见“基础配置指导”中的“FTP 和 TFTP”。

### 2. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 创建 FTP 类型的 NQA 模板，并进入模板视图。

**nqa template ftp** *name*

- (3) 配置 FTP 登录用户名。

**username** *username*

缺省情况下，未配置 FTP 登录用户名。

- (4) 配置 FTP 登录密码。

**password** { **cipher** | **simple** } *string*

缺省情况下，未配置 FTP 登录密码。

- (5) 配置探测报文的源地址。

（IPv4 网络）

**source ip** *ip-address*

缺省情况下，以报文发送接口的主 IPv4 地址作为探测报文中的源 IPv4 地址。

该命令指定的源地址必须是设备上接口的 IPv4 地址，且接口为 up 状态，否则测试将会失败。

（IPv6 网络）

**source ipv6** *ipv6-address*

缺省情况下，以报文发送接口的 IPv6 地址作为探测报文中的源 IPv6 地址。

该命令指定的源地址必须是设备上接口的 IPv6 地址，且接口为 up 状态，否则测试将会失败。

- (6) 配置 FTP 的数据传输方式。

**mode** { **active** | **passive** }

缺省情况下，FTP 数据传输方式为主动方式。

- (7) 配置 FTP 的操作类型。

**operation** { **get** | **put** }

缺省情况下，FTP 操作方式为 **get** 操作，即从 FTP 服务器获取文件。

- (8) 配置 FTP 测试访问的目的网址。

**url** *url*

缺省情况下，没有配置 FTP 测试访问的网址。

*url* 可以设置为 **ftp://host/filename** 或 **ftp://host:port/filename**。当 FTP 测试的操作类型为 **get** 方式时，必须在 *url* 中配置 *filename* 指定从 FTP 服务器获取的文件名。

- (9) 配置 FTP 服务器和客户端传送文件的文件名。

**filename** *filename*

缺省情况下，未配置 FTP 服务器和客户端之间传送文件的文件名。

当 FTP 测试的操作类型为 **put** 方式时，必须配置本命令来指定向 FTP 服务器传送的文件。

当 FTP 测试的操作类型为 **get** 方式时，不以此命令为准。

## 1.6.11 配置RADIUS类型的NQA模板

### 1. 功能简介

RADIUS 类型的 NQA 模板为外部特性提供 RADIUS 类型测试，外部特性通过引用该模板来启动 RADIUS 测试，来检测 RADIUS 服务器的业务可用性。

RADIUS 服务器是一种提供认证、授权和计费功能的服务器，RADIUS 类型的 NQA 模板检测过程选择了最基本的 RADIUS 认证过程：

- (1) NQA 客户端根据配置的用户名和密码，向 RADIUS 服务器发送认证请求包（Access-Request），其中的密码在共享密钥 Key 的参与下利用 MD5 算法进行加密处理。
- (2) RADIUS 服务器对用户名和密码进行认证，如果认证成功，RADIUS 服务器向 NQA 客户端发送认证接受包（Access-Accept）；如果认证失败，则返回认证拒绝包（Access-Reject）。
- (3) 当 NQA 客户端收到 RADIUS 服务器发出的认证接受包后，则表示 RADIUS 服务器是健康的；否则，该 RADIUS 服务器被认为无法成功提供服务。

### 2. 配置准备

RADIUS 测试需要 RADIUS 服务器和 NQA 客户端配合才能完成。进行 RADIUS 探测时，要求 RADIUS 服务器存在探测使用的用户信息，并配置与 NQA 客户端相同的密钥（Key）。RADIUS 服务器配置方法，请参见“安全使用指导”中的“AAA”。

### 3. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 创建 RADIUS 类型的 NQA 模板，并进入模板视图。

**nqa template radius** *name*

- (3) 配置测试操作的目的地址。

（IPv4 网络）

**destination ip** *ip-address*

（IPv6 网络）

**destination ipv6** *ipv6-address*

缺省情况下，未配置探测报文的目的地址。

- (4) 配置测试操作的目的端口。

**destination port** *port-number*

缺省情况下，测试操作的目的端口号为 1812。

- (5) 配置 RADIUS 用户名。

**username** *username*

缺省情况下，未配置 RADIUS 用户名。

- (6) 配置 RADIUS 密码。

**password** { **cipher** | **simple** } *string*

缺省情况下，未配置 RADIUS 密码。

- (7) 配置 RADIUS 认证使用的共享密钥。

**key** { **cipher** | **simple** } *string*

缺省情况下，未配置 RADIUS 认证使用的共享密钥。

- (8) 配置探测报文的源地址。

(IPv4 网络)

**source ip** *ip-address*

缺省情况下，以报文发送接口的主 IPv4 地址作为探测报文中的源 IPv4 地址。

该命令指定的源地址必须是设备上接口的 IPv4 地址，且接口为 up 状态，否则测试将会失败。

(IPv6 网络)

**source ipv6** *ipv6-address*

缺省情况下，以报文发送接口的 IPv6 地址作为探测报文中的源 IPv6 地址。

该命令指定的源地址必须是设备上接口的 IPv6 地址，且接口为 up 状态，否则测试将会失败。

## 1.6.12 配置SSL类型的NQA模板

### 1. 功能简介

SSL 类型的 NQA 模板为外部特性提供 SSL 类型测试，外部特性通过引用该模板，测试 NQA 客户端是否可以与指定的 SSL 服务器建立 SSL 连接，从而通过 SSL 连接建立的时间判断服务器的连通性及性能。

### 2. 配置准备

在进行 SSL 测试之前，需要在测试客户端完成 SSL 客户端策略配置。SSL 客户端策略配置方法请参见“安全配置指导”中的“SSL”。

### 3. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 创建 SSL 类型的 NQA 模板，并进入模板视图。

**nqa template ssl** *name*

- (3) 配置测试操作的地址。

(IPv4 网络)

**destination ip** *ip-address*

(IPv6 网络)

**destination ipv6** *ipv6-address*

缺省情况下，未配置探测报文的地址。

- (4) 配置测试操作的端口。

**destination port** *port-number*

缺省情况下，未配置测试操作的端口号。

- (5) 绑定 SSL 客户端策略。

**ssl-client-policy** *policy-name*

缺省情况下，未绑定 SSL 客户端策略。

- (6) 配置探测报文的源地址。

(IPv4 网络)

**source ip** *ip-address*

缺省情况下，以报文发送接口的主 IPv4 地址作为探测报文中的源 IPv4 地址。

该命令指定的源地址必须是设备上接口的 IPv4 地址，且接口为 up 状态，否则测试将会失败。

(IPv6 网络)

**source ipv6** *ipv6-address*

缺省情况下，以报文发送接口的 IPv6 地址作为探测报文中的源 IPv6 地址。

该命令指定的源地址必须是设备上接口的 IPv6 地址，且接口为 up 状态，否则测试将会失败。

## 1.6.13 配置NQA模板通用参数

### 1. 配置限制和指导

NQA 模板的通用参数，只对当前模板的测试有效。

除特别说明外，所有类型 NQA 模板都可以根据实际情况选择配置下列通用参数。

### 2. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 进入已存在的 NQA 模板视图。

**nqa template { arp | dns | ftp | http | https | icmp | ssl | tcp |  
tcphalfopen | udp } name**

- (3) 配置 NQA 模板的描述信息。

**description** *text*

缺省情况下，未配置模板的信息。

- (4) 配置连续两次探测开始时间的时间间隔。
- frequency interval**
- 缺省情况下，连续两次探测开始时间的时间间隔为 5000 毫秒。
- 如果到达 **frequency** 指定的时间间隔时，上次探测尚未完成，则不启动新一轮探测。
- (5) 配置每次探测超时时间。
- probe timeout timeout**
- 缺省情况下，探测的超时时间为 3000 毫秒。
- (6) 配置探测报文在网络中可以经过的最大跳数。
- ttl value**
- 缺省情况下，探测报文在网络中可以经过的最大跳数为 20 跳。
- ARP 类型的 NQA 模板不支持配置本命令。
- (7) 配置 NQA 探测报文 IP 报文头中服务类型域的值。
- tos value**
- 缺省情况下，NQA 探测报文 IP 报文头中服务类型域的值为 0。
- ARP 类型的 NQA 模板不支持配置本命令。
- (8) 配置连续探测成功的次数，当连续探测成功次数达到命令配置的数值时，NQA 客户端会把探测成功的消息发送给外部特性，使外部特性利用 NQA 测试的结果进行相应处理。
- reaction trigger probe-pass count**
- 缺省情况下，连续探测成功 3 次时，NQA 客户端会把探测成功的消息发送给外部特性，使外部特性利用 NQA 测试的结果进行相应处理。
- (9) 配置连续探测失败的次数，当连续探测失败次数达到命令配置的数值时，NQA 客户端会把探测失败的消息发送给外部特性，使外部特性利用 NQA 测试的结果进行相应处理。
- reaction trigger probe-fail count**
- 缺省情况下，连续探测失败 3 次时，NQA 客户端会把探测失败的消息发送给外部特性，是外部特性利用 NQA 测试的结果进行相应处理。

## 1.7 NQA显示和维护

在完成上述配置后，在任意视图下执行 **display** 命令可以显示配置后 NQA 的运行情况，通过查看显示信息验证配置的效果。

表1-2 NQA 显示和维护

操作	命令
显示NQA测试组的历史记录	<b>display nqa history</b> [ <i>admin-name</i> <i>operation-tag</i> ]
显示NQA阈值告警功能的当前监测结果	<b>display nqa reaction counters</b> [ <i>admin-name</i> <i>operation-tag</i> [ <i>item-number</i> ] ]
显示最近一次NQA测试的当前结果	<b>display nqa result</b> [ <i>admin-name</i> <i>operation-tag</i> ]
显示NQA测试的统计信息	<b>display nqa statistics</b> [ <i>admin-name</i> <i>operation-tag</i> ]
显示服务器的状态信息	<b>display nqa server</b>



## 1.8 NQA典型配置举例

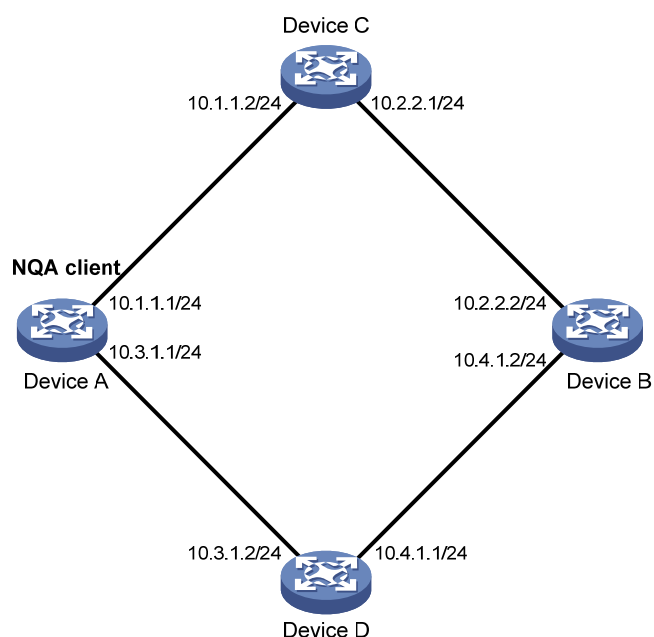
### 1.8.1 ICMP-echo测试配置举例

#### 1. 组网需求

使用 NQA 的 ICMP-echo 测试功能，测试本端（Device A）发送的报文是否可以经过指定的下一跳设备（Device C）到达指定的目的端（Device B），以及报文的往返时间。

#### 2. 组网图

图1-2 ICMP-echo 测试组网图



#### 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

# 创建 ICMP-echo 类型的 NQA 测试组（管理员为 admin，操作标签为 test1），并配置探测报文的目的地址为 10.2.2.2。

```
<DeviceA> system-view
[DeviceA] nqa entry admin test1
[DeviceA-nqa-admin-test1] type icmp-echo
[DeviceA-nqa-admin-test1-icmp-echo] destination ip 10.2.2.2
```

# 配置下一跳地址为 10.1.1.2，以便测试报文经过指定的下一跳设备（Device C）到达目的端，而不是通过 Device D 到达目的端。

```
[DeviceA-nqa-admin-test1-icmp-echo] next-hop ip 10.1.1.2
```

# 配置可选参数：一次 NQA 测试中探测的次数为 10，探测的超时时间为 500 毫秒，测试组连续两次测试开始时间的时间间隔为 5000 毫秒。

```
[DeviceA-nqa-admin-test1-icmp-echo] probe count 10
[DeviceA-nqa-admin-test1-icmp-echo] probe timeout 500
[DeviceA-nqa-admin-test1-icmp-echo] frequency 5000
# 开启 NQA 历史记录保存功能，并配置一个测试组中能够保存的最大历史记录个数为 10。
[DeviceA-nqa-admin-test1-icmp-echo] history-record enable
[DeviceA-nqa-admin-test1-icmp-echo] history-record number 10
[DeviceA-nqa-admin-test1-icmp-echo] quit
# 启动 ICMP-echo 测试操作，并一直进行测试。
[DeviceA] nqa schedule admin test1 start-time now lifetime forever
# 测试执行一段时间后，停止 ICMP-echo 测试操作。
[DeviceA] undo nqa schedule admin test1
```

#### 4. 验证配置

# 显示 ICMP-echo 测试中最后一次测试的当前结果。

```
[DeviceA] display nqa result admin test1
NQA entry (admin admin, tag test1) test results:
    Send operation times: 10          Receive response times: 10
    Min/Max/Average round trip time: 2/5/3
    Square-Sum of round trip time: 96
    Last succeeded probe time: 2011-08-23 15:00:01.2
Extended results:
    Packet loss ratio: 0%
    Failures due to timeout: 0
    Failures due to internal error: 0
    Failures due to other errors: 0
```

# 显示 ICMP-echo 测试的历史记录。

```
[DeviceA] display nqa history admin test1
NQA entry (admin admin, tag test1) history records:
  Index      Response      Status      Time
  ---
  370         3      Succeeded  2011-08-23 15:00:01.2
  369         3      Succeeded  2011-08-23 15:00:01.2
  368         3      Succeeded  2011-08-23 15:00:01.2
  367         5      Succeeded  2011-08-23 15:00:01.2
  366         3      Succeeded  2011-08-23 15:00:01.2
  365         3      Succeeded  2011-08-23 15:00:01.2
  364         3      Succeeded  2011-08-23 15:00:01.1
  363         2      Succeeded  2011-08-23 15:00:01.1
  362         3      Succeeded  2011-08-23 15:00:01.1
  361         2      Succeeded  2011-08-23 15:00:01.1
```

以上显示信息表示，Device A 发送的报文可以通过 Device C 到达 Device B；测试过程中未发生丢包；报文的最小、最大、平均往返时间分别为 2 毫秒、5 毫秒和 3 毫秒。

## 1.8.2 ICMP-jitter测试配置举例

### 1. 组网需求

使用 NQA 的 ICMP-jitter 测试功能，测试本端（Device A）和指定目的端（Device B）之间传送报文的时延抖动。

## 2. 组网图

图1-3 ICMP-jitter 测试组网图



## 3. 配置步骤

- (1) 配置各接口的 IP 地址。（配置过程略）
- (2) 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）
- (3) 配置 Device A

# 创建 ICMP-jitter 类型的 NQA 测试组（管理员为 admin，操作标签为 test1）。

```
<DeviceA> system-view
```

```
[DeviceA] nqa entry admin test1
```

```
[DeviceA-nqa-admin-test1] type icmp-jitter
```

# 配置测试操作的探测报文的目的地址为 10.2.2.2。

```
[DeviceA-nqa-admin-test1-icmp-jitter] destination ip 10.2.2.2
```

# 配置可选参数：测试组连续两次测试开始时间的时间间隔为 1000 毫秒。

```
[DeviceA-nqa-admin-test1-icmp-jitter] frequency 1000
```

```
[DeviceA-nqa-admin-test1-icmp-jitter] quit
```

# 启动 ICMP-jitter 测试操作，并一直进行测试。

```
[DeviceA] nqa schedule admin test1 start-time now lifetime forever
```

# 测试执行一段时间后，停止 ICMP-jitter 测试操作。

```
[DeviceA] undo nqa schedule admin test1
```

## 4. 验证配置

# 显示 ICMP-jitter 测试中最后一次测试的当前结果。

```
[DeviceA] display nqa result admin test1
```

NQA entry (admin admin, tag test1) test results:

Send operation times: 10                      Receive response times: 10

Min/Max/Average round trip time: 1/2/1

Square-Sum of round trip time: 13

Last packet received time: 2015-03-09 17:40:29.8

Extended results:

Packet loss ratio: 0%

Failures due to timeout: 0

Failures due to internal error: 0

Failures due to other errors: 0

Packets out of sequence: 0

Packets arrived late: 0

ICMP-jitter results:

RTT number: 10

Min positive SD: 0

Min positive DS: 0

Max positive SD: 0

Max positive DS: 0

```

Positive SD number: 0
Positive SD sum: 0
Positive SD average: 0
Positive SD square-sum: 0
Min negative SD: 1
Max negative SD: 1
Negative SD number: 1
Negative SD sum: 1
Negative SD average: 1
Negative SD square-sum: 1
One way results:
Max SD delay: 1
Min SD delay: 1
Number of SD delay: 1
Sum of SD delay: 1
Square-Sum of SD delay: 1
Lost packets for unknown reason: 0

Positive DS number: 0
Positive DS sum: 0
Positive DS average: 0
Positive DS square-sum: 0
Min negative DS: 2
Max negative DS: 2
Negative DS number: 1
Negative DS sum: 2
Negative DS average: 2
Negative DS square-sum: 4
Max DS delay: 2
Min DS delay: 2
Number of DS delay: 1
Sum of DS delay: 2
Square-Sum of DS delay: 4

# 显示 ICMP-jitter 测试的统计结果。
[DeviceA] display nqa statistics admin test1
NQA entry (admin admin, tag test1) test statistics:
NO. : 1
Start time: 2015-03-09 17:42:10.7
Life time: 156 seconds
Send operation times: 1560          Receive response times: 1560
Min/Max/Average round trip time: 1/2/1
Square-Sum of round trip time: 1563
Extended results:
Packet loss ratio: 0%
Failures due to timeout: 0
Failures due to internal error: 0
Failures due to other errors: 0
Packets out of sequence: 0
Packets arrived late: 0
ICMP-jitter results:
RTT number: 1560
Min positive SD: 1
Max positive SD: 1
Positive SD number: 18
Positive SD sum: 18
Positive SD average: 1
Positive SD square-sum: 18
Min negative SD: 1
Max negative SD: 1
Negative SD number: 24
Negative SD sum: 24
Negative SD average: 1
Negative SD square-sum: 24
Min positive DS: 1
Max positive DS: 2
Positive DS number: 46
Positive DS sum: 49
Positive DS average: 1
Positive DS square-sum: 55
Min negative DS: 1
Max negative DS: 2
Negative DS number: 57
Negative DS sum: 58
Negative DS average: 1
Negative DS square-sum: 60
One way results:

```

Max SD delay: 1	Max DS delay: 2
Min SD delay: 1	Min DS delay: 1
Number of SD delay: 4	Number of DS delay: 4
Sum of SD delay: 4	Sum of DS delay: 5
Square-Sum of SD delay: 4	Square-Sum of DS delay: 7
Lost packets for unknown reason: 0	

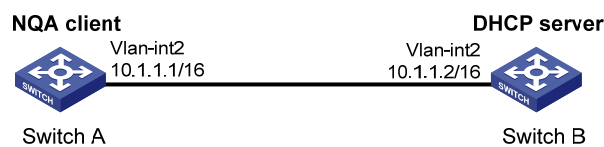
### 1.8.3 DHCP测试配置举例

#### 1. 组网需求

使用 NQA 的 DHCP 测试功能, 测试 Switch A 从 DHCP 服务器 Switch B 申请到 IP 地址所需的时间。

#### 2. 组网图

图1-4 配置 DHCP 组网图



#### 3. 配置步骤

# 创建 DHCP 类型的 NQA 测试组（管理员为 admin，操作标签为 test1），并指定进行 DHCP 测试中探测报文的目的地址为 10.1.1.2。

```

<SwitchA> system-view
[SwitchA] nqa entry admin test1
[SwitchA-nqa-admin-test1] type dhcp
[SwitchA-nqa-admin-test1-dhcp] destination ip 10.1.1.2
  
```

# 开启 NQA 测试组的历史记录保存功能。

```

[SwitchA-nqa-admin-test1-dhcp] history-record enable
[SwitchA-nqa-admin-test1-dhcp] quit
  
```

# 启动 DHCP 测试操作，并一直进行测试。

```

[SwitchA] nqa schedule admin test1 start-time now lifetime forever
  
```

# 测试执行一段时间后，停止 DHCP 测试操作。

```

[SwitchA] undo nqa schedule admin test1
  
```

#### 4. 验证配置

# 显示 DHCP 测试中最后一次测试的当前结果。

```

[SwitchA] display nqa result admin test1
NQA entry (admin admin, tag test1) test results:
  Send operation times: 1          Receive response times: 1
  Min/Max/Average round trip time: 512/512/512
  Square-Sum of round trip time: 262144
  Last succeeded probe time: 2011-11-22 09:56:03.2
Extended results:
  Packet loss ratio: 0%
  Failures due to timeout: 0
  Failures due to internal error: 0
  
```

```
Failures due to other errors: 0
```

# 显示 DHCP 测试的历史记录。

```
[SwitchA] display nqa history admin test1
```

```
NQA entry (admin admin, tag test1) history records:
```

Index	Response	Status	Time
1	512	Succeeded	2011-11-22 09:56:03.2

以上显示信息表示，Switch A 可以从 DHCP 服务器获取 IP 地址，获取 IP 地址所需的时间为 512 毫秒。

## 1.8.4 DNS测试配置举例

### 1. 组网需求

使用 NQA 的 DNS 测试功能，测试 Device A 是否可以通过指定的 DNS 服务器将域名 host.com 解析为 IP 地址，并测试域名解析所需的时间。

### 2. 组网图

图1-5 配置 DNS 组网图



### 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

# 创建 DNS 类型的 NQA 测试组（管理员为 admin，操作标签为 test1）。

```
<DeviceA> system-view
```

```
[DeviceA] nqa entry admin test1
```

```
[DeviceA-nqa-admin-test1] type dns
```

# 配置探测报文的目的地址为 DNS 服务器的 IP 地址 10.2.2.2，要解析的域名为 host.com。

```
[DeviceA-nqa-admin-test1-dns] destination ip 10.2.2.2
```

```
[DeviceA-nqa-admin-test1-dns] resolve-target host.com
```

# 开启 NQA 测试组的历史记录保存功能。

```
[DeviceA-nqa-admin-test1-dns] history-record enable
```

```
[DeviceA-nqa-admin-test1-dns] quit
```

# 启动 DNS 测试操作，并一直进行测试。

```
[DeviceA] nqa schedule admin test1 start-time now lifetime forever
```

# 测试执行一段时间后，停止 DNS 测试操作。

```
[DeviceA] undo nqa schedule admin test1
```

### 4. 验证配置

# 显示 DNS 测试中最后一次测试的当前结果。

```
[DeviceA] display nqa result admin test1
```

```
NQA entry (admin admin, tag test1) test results:
```

```

Send operation times: 1          Receive response times: 1
Min/Max/Average round trip time: 62/62/62
Square-Sum of round trip time: 3844
Last succeeded probe time: 2011-11-10 10:49:37.3
Extended results:
Packet loss ratio: 0%
Failures due to timeout: 0
Failures due to internal error: 0
Failures due to other errors: 0

```

# 显示 DNS 测试的历史记录。

```
[DeviceA] display nqa history admin test1
```

NQA entry (admin admin, tag test1) history records:

Index	Response	Status	Time
1	62	Succeeded	2011-11-10 10:49:37.3

以上显示信息表示，Device A 可以通过指定的 DNS 服务器将域名 host.com 解析为 IP 地址，域名解析所需的时间为 62 毫秒。

## 1.8.5 FTP测试配置举例

### 1. 组网需求

使用 NQA 的 FTP 测试功能，测试 Device A 是否可以和指定的 FTP 服务器 Device B 建立连接，以及往 FTP 服务器上传一个文件的时间。登录 FTP 服务器的用户名为 admin，密码为 systemtest，要传送到服务器的文件名为 config.txt。

### 2. 组网图

图1-6 配置 FTP 组网图



### 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

# 创建 FTP 类型的 NQA 测试组（管理员为 admin，操作标签为 test1）。

```
<DeviceA> system-view
```

```
[DeviceA] nqa entry admin test1
```

```
[DeviceA-nqa-admin-test1] type ftp
```

# 配置测试操作的目的地址为 FTP 服务器的 IP 地址 10.2.2.2。

```
[DeviceA-nqa-admin-test1-ftp] url ftp://10.2.2.2
```

# 配置探测报文的源 IP 地址为 10.1.1.1。

```
[DeviceA-nqa-admin-test1-ftp] source ip 10.1.1.1
```

# 配置测试执行的操作为向 FTP 服务器上传文件 config.txt。

```
[DeviceA-nqa-admin-test1-ftp] operation put
```

```
[DeviceA-nqa-admin-test1-ftp] filename config.txt
# 配置 FTP 操作的登录用户名为 admin。
[DeviceA-nqa-admin-test1-ftp] username admin
# 配置 FTP 操作的登录密码为 systemtest。
[DeviceA-nqa-admin-test1-ftp] password simple systemtest
# 开启 NQA 测试组的历史记录保存功能。
[DeviceA-nqa-admin-test1-ftp] history-record enable
[DeviceA-nqa-admin-test1-ftp] quit
# 启动 FTP 测试操作，并一直进行测试。
[DeviceA] nqa schedule admin test1 start-time now lifetime forever
# 测试执行一段时间后，停止 FTP 测试操作。
[DeviceA] undo nqa schedule admin test1
```

#### 4. 验证配置

```
# 显示 FTP 测试中最后一次测试的当前结果。
[DeviceA] display nqa result admin test1
NQA entry (admin admin, tag test1) test results:
    Send operation times: 1                Receive response times: 1
    Min/Max/Average round trip time: 173/173/173
    Square-Sum of round trip time: 29929
    Last succeeded probe time: 2011-11-22 10:07:28.6
Extended results:
    Packet loss ratio: 0%
    Failures due to timeout: 0
    Failures due to disconnect: 0
    Failures due to no connection: 0
    Failures due to internal error: 0
    Failures due to other errors: 0
```

```
# 显示 FTP 测试的历史记录。
[DeviceA] display nqa history admin test1
NQA entry (admin admin, tag test1) history records:
    Index      Response      Status      Time
    1          173          Succeeded   2011-11-22 10:07:28.6
```

以上显示信息表示，Device A 可以和指定的 FTP 服务器 Device B 建立连接，向 FTP 服务器上传一个文件的时间是 173 毫秒。

### 1.8.6 HTTP测试配置举例

#### 1. 组网需求

使用 NQA 的 HTTP 测试功能，测试是否可以和指定的 HTTP 服务器之间建立连接，以及从 HTTP 服务器获取数据的时间。



## 2. 组网图

图1-7 HTTP 测试组网图



## 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

# 创建 HTTP 类型的 NQA 测试组（管理员为 **admin**，操作标签为 **test1**）。

```
<DeviceA> system-view
```

```
[DeviceA] nqa entry admin test1
```

```
[DeviceA-nqa-admin-test1] type http
```

# 配置 HTTP 测试服务器的 IP 地址为 10.2.2.2，访问的网址为 **/index.htm**。

```
[DeviceA-nqa-admin-test1-http] url http://10.2.2.2/index.htm
```

# 配置 HTTP 测试的操作方式为 **get** 操作。（**get** 操作为缺省操作方式，因此，可以不执行本配置）

```
[DeviceA-nqa-admin-test1-http] operation get
```

# 配置 HTTP 测试使用的版本为 **1.0**。（缺省情况下使用的版本为 **1.0**，因此，可以不执行本配置）

```
[DeviceA-nqa-admin-test1-http] version v1.0
```

# 开启 NQA 测试组的历史记录保存功能。

```
[DeviceA-nqa-admin-test1-http] history-record enable
```

```
[DeviceA-nqa-admin-test1-http] quit
```

# 启动 HTTP 测试操作，并一直进行测试。

```
[DeviceA] nqa schedule admin test1 start-time now lifetime forever
```

# 测试执行一段时间后，停止 HTTP 测试操作。

```
[DeviceA] undo nqa schedule admin test1
```

## 4. 验证配置

# 显示 HTTP 测试中最后一次测试的当前结果。

```
[DeviceA] display nqa result admin test1
```

NQA entry (admin admin, tag test1) test results:

Send operation times: 1                      Receive response times: 1

Min/Max/Average round trip time: 64/64/64

Square-Sum of round trip time: 4096

Last succeeded probe time: 2011-11-22 10:12:47.9

Extended results:

Packet loss ratio: 0%

Failures due to timeout: 0

Failures due to disconnect: 0

Failures due to no connection: 0

Failures due to internal error: 0

Failures due to other errors: 0

# 显示 HTTP 测试的历史记录。

```
[DeviceA] display nqa history admin test1
NQA entry (admin admin, tag test1) history records:
```

Index	Response	Status	Time
1	64	Succeeded	2011-11-22 10:12:47.9

以上显示信息表示，Device A 可以和指定的 HTTP 服务器 Device B 建立连接，从 HTTP 服务器获取数据的时间为 64 毫秒。

## 1.8.7 UDP-jitter测试配置举例

### 1. 组网需求

使用 NQA 的 UDP-jitter 测试功能，测试本端（Device A）和指定目的端（Device B）的端口 9000 之间传送报文的抖动时间。

### 2. 组网图

图1-8 UDP-jitter 测试组网图



### 3. 配置步骤

- (1) 配置各接口的 IP 地址。（配置过程略）
- (2) 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）
- (3) 配置 Device B

# 使能 NQA 服务器，配置监听的 IP 地址为 10.2.2.2，UDP 端口号为 9000。

```
<DeviceB> system-view
[DeviceB] nqa server enable
[DeviceB] nqa server udp-echo 10.2.2.2 9000
```

- (4) 配置 Device A

# 创建 UDP-jitter 类型的 NQA 测试组（管理员为 admin，操作标签为 test1）。

```
<DeviceA> system-view
[DeviceA] nqa entry admin test1
[DeviceA-nqa-admin-test1] type udp-jitter
# 配置测试操作的探测报文的目的地址为 10.2.2.2，目的端口号为 9000。
[DeviceA-nqa-admin-test1-udp-jitter] destination ip 10.2.2.2
[DeviceA-nqa-admin-test1-udp-jitter] destination port 9000
# 配置可选参数：测试组连续两次测试开始时间的时间间隔为 1000 毫秒。
[DeviceA-nqa-admin-test1-udp-jitter] frequency 1000
[DeviceA-nqa-admin-test1-udp-jitter] quit
# 启动 UDP-jitter 测试操作，并一直进行测试。
[DeviceA] nqa schedule admin test1 start-time now lifetime forever
# 测试执行一段时间后，停止 UDP-jitter 测试操作。
[DeviceA] undo nqa schedule admin test1
```

#### 4. 验证配置

# 显示 UDP-jitter 测试中最后一次测试的当前结果。

```
[DeviceA] display nqa result admin test1
NQA entry (admin admin, tag test1) test results:
    Send operation times: 10                Receive response times: 10
    Min/Max/Average round trip time: 15/32/17
    Square-Sum of round trip time: 3235
    Last packet received time: 2011-05-29 13:56:17.6
Extended results:
    Packet loss ratio: 0%
    Failures due to timeout: 0
    Failures due to internal error: 0
    Failures due to other errors: 0
    Packets out of sequence: 0
    Packets arrived late: 0
UDP-jitter results:
    RTT number: 10
    Min positive SD: 4                      Min positive DS: 1
    Max positive SD: 21                    Max positive DS: 28
    Positive SD number: 5                  Positive DS number: 4
    Positive SD sum: 52                    Positive DS sum: 38
    Positive SD average: 10                Positive DS average: 10
    Positive SD square-sum: 754            Positive DS square-sum: 460
    Min negative SD: 1                    Min negative DS: 6
    Max negative SD: 13                    Max negative DS: 22
    Negative SD number: 4                  Negative DS number: 5
    Negative SD sum: 38                    Negative DS sum: 52
    Negative SD average: 10                Negative DS average: 10
    Negative SD square-sum: 460            Negative DS square-sum: 754
One way results:
    Max SD delay: 15                      Max DS delay: 16
    Min SD delay: 7                      Min DS delay: 7
    Number of SD delay: 10                Number of DS delay: 10
    Sum of SD delay: 78                    Sum of DS delay: 85
    Square-Sum of SD delay: 666            Square-Sum of DS delay: 787
    SD lost packets: 0                    DS lost packets: 0
    Lost packets for unknown reason: 0
```

# 显示 UDP-jitter 测试的统计结果。

```
[DeviceA] display nqa statistics admin test1
NQA entry (admin admin, tag test1) test statistics:
NO. : 1
    Start time: 2011-05-29 13:56:14.0
    Life time: 47 seconds
    Send operation times: 410                Receive response times: 410
    Min/Max/Average round trip time: 1/93/19
    Square-Sum of round trip time: 206176
Extended results:
```

```

Packet loss ratio: 0%
Failures due to timeout: 0
Failures due to internal error: 0
Failures due to other errors: 0
Packets out of sequence: 0
Packets arrived late: 0
UDP-jitter results:
RTT number: 410
Min positive SD: 3
Max positive SD: 30
Positive SD number: 186
Positive SD sum: 2602
Positive SD average: 13
Positive SD square-sum: 45304
Min negative SD: 1
Max negative SD: 30
Negative SD number: 181
Negative SD sum: 181
Negative SD average: 13
Negative SD square-sum: 46994
Min positive DS: 1
Max positive DS: 79
Positive DS number: 158
Positive DS sum: 1928
Positive DS average: 12
Positive DS square-sum: 31682
Min negative DS: 1
Max negative DS: 78
Negative DS number: 209
Negative DS sum: 209
Negative DS average: 14
Negative DS square-sum: 3030
One way results:
Max SD delay: 46
Min SD delay: 7
Number of SD delay: 410
Sum of SD delay: 3705
Square-Sum of SD delay: 45987
SD lost packets: 0
Lost packets for unknown reason: 0
Max DS delay: 46
Min DS delay: 7
Number of DS delay: 410
Sum of DS delay: 3891
Square-Sum of DS delay: 49393
DS lost packets: 0

```

## 1.8.8 SNMP测试配置举例

### 1. 组网需求

使用 NQA 的 SNMP 测试功能，测试从 Device A 发出 SNMP 协议查询报文到收到 SNMP Agent（Device B）响应报文所用的时间。

### 2. 组网图

图1-9 SNMP 配置测试组网图



### 3. 配置步骤

- (1) 配置各接口的 IP 地址。（配置过程略）
- (2) 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）
- (3) 配置 SNMP Agent（Device B）

# 启动 SNMP Agent 服务，设置 SNMP 版本为 all、只读团体名为 public、读写团体名为 private。

```
<DeviceB> system-view
[DeviceB] snmp-agent sys-info version all
[DeviceB] snmp-agent community read public
[DeviceB] snmp-agent community write private
```

#### (4) 配置 Device A

# 创建 SNMP 类型的测试组（管理员为 admin，操作标签为 test1），并配置测试操作的探测报文的地址为 SNMP Agent 的 IP 地址 10.2.2.2。

```
<DeviceA> system-view
[DeviceA] nqa entry admin test1
[DeviceA-nqa-admin-test1] type snmp
[DeviceA-nqa-admin-test1-snmp] destination ip 10.2.2.2
```

# 开启 NQA 测试组的历史记录保存功能。

```
[DeviceA-nqa-admin-test1-snmp] history-record enable
[DeviceA-nqa-admin-test1-snmp] quit
```

# 启动测试操作，并一直进行测试。

```
[DeviceA] nqa schedule admin test1 start-time now lifetime forever
```

# 测试执行一段时间后，停止 SNMP 测试操作。

```
[DeviceA] undo nqa schedule admin test1
```

## 4. 验证配置

# 显示 SNMP 测试中最后一次测试的当前结果。

```
[DeviceA] display nqa result admin test1
```

NQA entry (admin admin, tag test1) test results:

```
Send operation times: 1          Receive response times: 1
Min/Max/Average round trip time: 50/50/50
Square-Sum of round trip time: 2500
Last succeeded probe time: 2011-11-22 10:24:41.1
```

Extended results:

```
Packet loss ratio: 0%
Failures due to timeout: 0
Failures due to internal error: 0
Failures due to other errors: 0
```

# 显示 SNMP 测试的历史记录。

```
[DeviceA] display nqa history admin test1
```

NQA entry (admin admin, tag test1) history records:

Index	Response	Status	Time
1	50	Succeeded	2011-11-22 10:24:41.1

以上显示信息表示，Device A 可以和 SNMP Agent（Device B）建立连接，从 Device A 发出一个 SNMP 协议查询报文到收到 SNMP Agent 响应报文所用的时间为 50 毫秒。

## 1.8.9 TCP测试配置举例

### 1. 组网需求

使用 NQA 的 TCP 测试功能，测试本端（Device A）和指定目的端（Device B）的端口 9000 之间建立 TCP 连接所需的时间。

### 2. 组网图

图1-10 TCP 测试组网图



### 3. 配置步骤

- (1) 配置各接口的 IP 地址。（配置过程略）
- (2) 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）
- (3) 配置 Device B

# 使能 NQA 服务器，配置监听的 IP 地址为 10.2.2.2，TCP 端口号为 9000。

```
<DeviceB> system-view
[DeviceB] nqa server enable
[DeviceB] nqa server tcp-connect 10.2.2.2 9000
```

- (4) 配置 Device A

# 创建 TCP 类型的测试组（管理员为 admin，操作标签为 test1）。

```
<DeviceA> system-view
[DeviceA] nqa entry admin test1
[DeviceA-nqa-admin-test1] type tcp
```

# 配置探测报文的目的地址为 10.2.2.2，目的端口号为 9000。

```
[DeviceA-nqa-admin-test1-tcp] destination ip 10.2.2.2
[DeviceA-nqa-admin-test1-tcp] destination port 9000
```

# 开启 NQA 测试组的历史记录保存功能。

```
[DeviceA-nqa-admin-test1-tcp] history-record enable
[DeviceA-nqa-admin-test1-tcp] quit
```

# 启动测试操作，并一直进行测试。

```
[DeviceA] nqa schedule admin test1 start-time now lifetime forever
```

# 测试执行一段时间后，停止 TCP 测试操作。

```
[DeviceA] undo nqa schedule admin test1
```

### 4. 验证配置

# 显示 TCP 测试中最后一次测试的当前结果。

```
[DeviceA] display nqa result admin test1
```

NQA entry (admin admin, tag test1) test results:

```
Send operation times: 1          Receive response times: 1
Min/Max/Average round trip time: 13/13/13
Square-Sum of round trip time: 169
```

```

Last succeeded probe time: 2011-11-22 10:27:25.1
Extended results:
  Packet loss ratio: 0%
  Failures due to timeout: 0
  Failures due to disconnect: 0
  Failures due to no connection: 0
  Failures due to internal error: 0
  Failures due to other errors: 0

```

# 显示 TCP 测试的历史记录。

```
[DeviceA] display nqa history admin test1
```

NQA entry (admin admin, tag test1) history records:

Index	Response	Status	Time
1	13	Succeeded	2011-11-22 10:27:25.1

以上显示信息表示，Device A 可以与 Device B 的端口 9000 建立 TCP 连接，建立连接所需的时间为 13 毫秒。

## 1.8.10 UDP-echo测试配置举例

### 1. 组网需求

使用 NQA 的 UDP-echo 测试功能，测试本端（Device A）和指定目的端（Device B）的端口 8000 之间 UDP 协议报文的往返时间。

### 2. 组网图

图1-11 UDP-echo 测试组网图



### 3. 配置步骤

- (1) 配置各接口的 IP 地址。（配置过程略）
- (2) 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）
- (3) 配置 Device B

# 使能 NQA 服务器，配置监听的 IP 地址为 10.2.2.2，UDP 端口号为 8000。

```

<DeviceB> system-view
[DeviceB] nqa server enable
[DeviceB] nqa server udp-echo 10.2.2.2 8000

```

- (4) 配置 Device A

# 创建 UDP-echo 类型的测试组（管理员为 admin，操作标签为 test1）。

```

<DeviceA> system-view
[DeviceA] nqa entry admin test1
[DeviceA-nqa-admin-test1] type udp-echo

```

# 配置探测报文的目的地址为 10.2.2.2，目的端口号为 8000。

```
[DeviceA-nqa-admin-test1-udp-echo] destination ip 10.2.2.2
```

```
[DeviceA-nqa-admin-test1-udp-echo] destination port 8000
# 开启 NQA 测试组的历史记录保存功能。
[DeviceA-nqa-admin-test1-udp-echo] history-record enable
[DeviceA-nqa-admin-test1-udp-echo] quit
# 启动测试操作，并一直进行测试。
[DeviceA] nqa schedule admin test1 start-time now lifetime forever
# 测试执行一段时间后，停止 UDP-echo 测试操作。
[DeviceA] undo nqa schedule admin test1
```

#### 4. 验证配置

# 显示 UDP-echo 测试中最后一次测试的当前结果。

```
[DeviceA] display nqa result admin test1
NQA entry (admin admin, tag test1) test results:
  Send operation times: 1          Receive response times: 1
  Min/Max/Average round trip time: 25/25/25
  Square-Sum of round trip time: 625
  Last succeeded probe time: 2011-11-22 10:36:17.9
Extended results:
  Packet loss ratio: 0%
  Failures due to timeout: 0
  Failures due to internal error: 0
  Failures due to other errors: 0
```

# 显示 UDP-echo 测试的历史记录。

```
[DeviceA] display nqa history admin test1
NQA entry (admin admin, tag test1) history records:
  Index      Response      Status      Time
  1          25            Succeeded   2011-11-22 10:36:17.9
```

以上显示信息表示，Device A 和 Device B 的端口 8000 之间 UDP 协议报文的往返时间为 25 毫秒。

### 1.8.11 UDP-tracert测试配置举例

#### 1. 组网需求

使用 NQA 的 UDP-tracert 测试功能，探测本端（Device A）到指定目的端（Device B）之间经过的路径信息。

#### 2. 组网图

图1-12 UDP-tracert 测试组网图



#### 3. 配置步骤

- (1) 配置各接口的 IP 地址。（配置过程略）
- (2) 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）



(3) 在中间设备上配置 **ip ttl-expires enable** 命令，在 Device B 上配置 **ip unreachable enable** 命令。

(4) 配置 Device A

# 创建 UDP-tracert 类型的测试组（管理员为 admin，操作标签为 test1）。

```
<DeviceA> system-view
```

```
[DeviceA] nga entry admin test1
```

```
[DeviceA-nga-admin-test1] type udp-tracert
```

# 配置测试操作的地址为 10.2.2.2，目的端口号为 33434。（目的端口号为 33434 是缺省操作方式，因此，可以不执行本配置）

```
[DeviceA-nga-admin-test1-udp-tracert] destination ip 10.2.2.2
```

```
[DeviceA-nga-admin-test1-udp-tracert] destination port 33434
```

# 配置可选参数：对一个 TTL 值的节点的探测次数为 3（探测次数为 3 是缺省操作方式，因此，可以不执行本配置），探测的超时时间为 500 毫秒，测试组连续两次测试开始时间的时间间隔为 5000 毫秒。

```
[DeviceA-nga-admin-test1-udp-tracert] probe count 3
```

```
[DeviceA-nga-admin-test1-udp-tracert] probe timeout 500
```

```
[DeviceA-nga-admin-test1-udp-tracert] frequency 5000
```

# 配置 UDP-tracert 测试的出接口为 GigabitEthernet1/0/1。

```
[DeviceA-nga-admin-test1-udp-tracert] out interface gigabitethernet 1/0/1
```

# 开启 UDP-tracert 测试的禁止报文分片功能。

```
[DeviceA-nga-admin-test1-udp-tracert] no-fragment enable
```

# 配置最大连续失败次数为 6 次，配置初始 TTL 为 1

```
[DeviceA-nga-admin-test1-udp-tracert] max-failure 6
```

```
[DeviceA-nga-admin-test1-udp-tracert] init-ttl 1
```

# 启动测试操作，并一直进行测试。

```
[DeviceA] nga schedule admin test1 start-time now lifetime forever
```

# 测试执行一段时间后，停止 UDP-tracert 测试操作。

```
[DeviceA] undo nga schedule admin test1
```

#### 4. 验证配置

# 显示 UDP-tracert 测试中最后一次测试的当前结果。

```
[DeviceA] display nga result admin test1
```

NQA entry (admin admin, tag test1) test results:

Send operation times: 6                      Receive response times: 6

Min/Max/Average round trip time: 1/1/1

Square-Sum of round trip time: 1

Last succeeded probe time: 2013-09-09 14:46:06.2

Extended results:

Packet loss in test: 0%

Failures due to timeout: 0

Failures due to internal error: 0

Failures due to other errors: 0

UDP-tracert results:

TTL	Hop IP	Time
1	3.1.1.1	2013-09-09 14:46:03.2

2 10.2.2.2 2013-09-09 14:46:06.2

# 显示 UDP-tracert 测试的历史记录。

```
[DeviceA] display nqa history admin test1
```

NQA entry (admin admin, tag test1) history records:

Index	TTL	Response	Hop IP	Status	Time
1	2	2	10.2.2.2	Succeeded	2013-09-09 14:46:06.2
1	2	1	10.2.2.2	Succeeded	2013-09-09 14:46:05.2
1	2	2	10.2.2.2	Succeeded	2013-09-09 14:46:04.2
1	1	1	3.1.1.1	Succeeded	2013-09-09 14:46:03.2
1	1	2	3.1.1.1	Succeeded	2013-09-09 14:46:02.2
1	1	1	3.1.1.1	Succeeded	2013-09-09 14:46:01.2

## 1.8.12 Voice测试配置举例

### 1. 组网需求

使用 NQA 的 Voice 测试功能，测试本端（Device A）和指定的目的端（Device B）之间传送语音报文的抖动时间和网络语音质量参数。

### 2. 组网图

图1-13 Voice 测试组网图



### 3. 配置步骤

- (1) 配置各接口的 IP 地址。（配置过程略）
- (2) 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）
- (3) 配置 Device B

# 使能 NQA 服务器，配置监听的 IP 地址为 10.2.2.2，UDP 端口号为 9000。

```
<DeviceB> system-view
[DeviceB] nqa server enable
[DeviceB] nqa server udp-echo 10.2.2.2 9000
```

- (4) 配置 Device A

# 创建 Voice 类型的 NQA 测试组（管理员为 admin，操作标签为 test1）。

```
<DeviceA> system-view
[DeviceA] nqa entry admin test1
[DeviceA-nqa-admin-test1] type voice
```

# 配置探测报文的目的地址为 10.2.2.2，目的端口号为 9000。

```
[DeviceA-nqa-admin-test1-voice] destination ip 10.2.2.2
[DeviceA-nqa-admin-test1-voice] destination port 9000
[DeviceA-nqa-admin-test1-voice] quit
```

# 启动 Voice 测试操作，并一直进行测试。

```
[DeviceA] nqa schedule admin test1 start-time now lifetime forever
```

# 测试执行一段时间后，停止 Voice 测试操作。

```
[DeviceA] undo nqa schedule admin test1
```

#### 4. 验证配置

# 显示 Voice 测试中最后一次测试的当前结果。

```
[DeviceA] display nqa result admin test1
```

NQA entry (admin admin, tag test1) test results:

Send operation times: 1000                      Receive response times: 1000

Min/Max/Average round trip time: 31/1328/33

Square-Sum of round trip time: 2844813

Last packet received time: 2011-06-13 09:49:31.1

Extended results:

Packet loss ratio: 0%

Failures due to timeout: 0

Failures due to internal error: 0

Failures due to other errors: 0

Packets out of sequence: 0

Packets arrived late: 0

Voice results:

RTT number: 1000

Min positive SD: 1

Min positive DS: 1

Max positive SD: 204

Max positive DS: 1297

Positive SD number: 257

Positive DS number: 259

Positive SD sum: 759

Positive DS sum: 1797

Positive SD average: 2

Positive DS average: 6

Positive SD square-sum: 54127

Positive DS square-sum: 1691967

Min negative SD: 1

Min negative DS: 1

Max negative SD: 203

Max negative DS: 1297

Negative SD number: 255

Negative DS number: 259

Negative SD sum: 759

Negative DS sum: 1796

Negative SD average: 2

Negative DS average: 6

Negative SD square-sum: 53655

Negative DS square-sum: 1691776

One way results:

Max SD delay: 343

Max DS delay: 985

Min SD delay: 343

Min DS delay: 985

Number of SD delay: 1

Number of DS delay: 1

Sum of SD delay: 343

Sum of DS delay: 985

Square-Sum of SD delay: 117649

Square-Sum of DS delay: 970225

SD lost packets: 0

DS lost packets: 0

Lost packets for unknown reason: 0

Voice scores:

MOS value: 4.38

ICPIF value: 0

# 显示 Voice 测试的统计结果。

```
[DeviceA] display nqa statistics admin test1
```

NQA entry (admin admin, tag test1) test statistics:

NO. : 1

Start time: 2011-06-13 09:45:37.8

```

Life time: 331 seconds
Send operation times: 4000          Receive response times: 4000
Min/Max/Average round trip time: 15/1328/32
Square-Sum of round trip time: 7160528
Extended results:
  Packet loss ratio: 0%
  Failures due to timeout: 0
  Failures due to internal error: 0
  Failures due to other errors: 0
Packets out of sequence: 0
  Packets arrived late: 0
Voice results:
  RTT number: 4000
  Min positive SD: 1                Min positive DS: 1
  Max positive SD: 360              Max positive DS: 1297
  Positive SD number: 1030          Positive DS number: 1024
  Positive SD sum: 4363             Positive DS sum: 5423
  Positive SD average: 4            Positive DS average: 5
  Positive SD square-sum: 497725    Positive DS square-sum: 2254957
  Min negative SD: 1               Min negative DS: 1
  Max negative SD: 360              Max negative DS: 1297
  Negative SD number: 1028          Negative DS number: 1022
  Negative SD sum: 1028             Negative DS sum: 1022
  Negative SD average: 4            Negative DS average: 5
  Negative SD square-sum: 495901    Negative DS square-sum: 5419
One way results:
  Max SD delay: 359                 Max DS delay: 985
  Min SD delay: 0                   Min DS delay: 0
  Number of SD delay: 4             Number of DS delay: 4
  Sum of SD delay: 1390             Sum of DS delay: 1079
  Square-Sum of SD delay: 483202    Square-Sum of DS delay: 973651
  SD lost packets: 0                DS lost packets: 0
  Lost packets for unknown reason: 0
Voice scores:
  Max MOS value: 4.38               Min MOS value: 4.38
  Max ICPIF value: 0                Min ICPIF value: 0

```

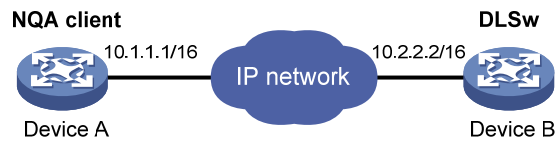
### 1.8.13 DLSw测试配置举例

#### 1. 组网需求

使用 NQA 的 DLSw 测试功能，测试 DLSw 设备的响应时间。

## 2. 组网图

图1-14 DLSw 测试组网图



## 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

# 创建 DLSw 类型的测试组（管理员为 **admin**，操作标签为 **test1**），并配置探测报文的目的地址为 10.2.2.2。

```
<DeviceA> system-view
[DeviceA] nqa entry admin test1
[DeviceA-nqa-admin-test1] type dls
[DeviceA-nqa-admin-test1-dls] destination ip 10.2.2.2
```

# 开启 NQA 测试组的历史记录保存功能。

```
[DeviceA-nqa-admin-test1-dls] history-record enable
[DeviceA-nqa-admin-test1-dls] quit
```

# 启动测试操作，并一直进行测试。

```
[DeviceA] nqa schedule admin test1 start-time now lifetime forever
```

# 测试执行一段时间后，停止 DLSw 测试操作。

```
[DeviceA] undo nqa schedule admin test1
```

## 4. 验证配置

# 显示 DLSw 测试中最后一次测试的当前结果。

```
[DeviceA] display nqa result admin test1
NQA entry (admin admin, tag test1) test results:
  Send operation times: 1          Receive response times: 1
  Min/Max/Average round trip time: 19/19/19
  Square-Sum of round trip time: 361
  Last succeeded probe time: 2011-11-22 10:40:27.7
Extended results:
  Packet loss ratio: 0%
  Failures due to timeout: 0
  Failures due to disconnect: 0
  Failures due to no connection: 0
  Failures due to internal error: 0
  Failures due to other errors: 0
```

# 显示 DLSw 测试的历史记录。

```
[DeviceA] display nqa history admin test1
NQA entry (admin admin, tag test1) history records:
  Index      Response      Status      Time
  1          19           Succeeded   2011-11-22 10:40:27.7
```

以上显示信息表示，DLSw 设备的响应时间为 19 毫秒。

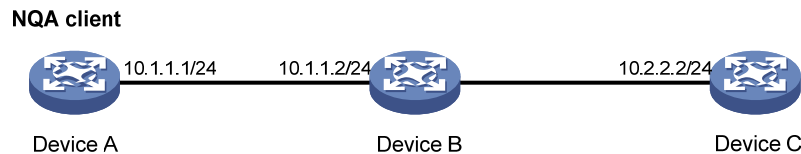
## 1.8.14 Path-jitter测试配置举例

### 1. 组网需求

使用 NQA 的 Path-jitter 测试功能，测试本端（Device A）到指定目的端（Device C）间的网络质量情况。

### 2. 组网图

图1-15 Path-jitter 测试组网图



### 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

# 在 Device B 上配置 **ip ttl-expires enable** 命令，在设备 C 上配置 **ip unreachable enable** 命令。

# 创建 Path-jitter 类型的 NQA 测试组（管理员为 admin，操作标签为 test1），并配置探测报文的目的地址为 10.2.2.2。

```
<DeviceA> system-view
[DeviceA] nqa entry admin test1
[DeviceA-nqa-admin-test1] type path-jitter
[DeviceA-nqa-admin-test1-path-jitter] destination ip 10.2.2.2
```

# 配置可选参数：测试组连续两次测试开始时间的时间间隔为 10000 毫秒。

```
[DeviceA-nqa-admin-test1-path-jitter] frequency 10000
[DeviceA-nqa-admin-test1-path-jitter] quit
```

# 启动 Path-jitter 测试操作，并一直进行测试。

```
[DeviceA] nqa schedule admin test1 start-time now lifetime forever
```

# 测试执行一段时间后，停止 Path-jitter 测试操作。

```
[DeviceA] undo nqa schedule admin test1
```

### 4. 验证配置

# 显示 Path-jitter 测试中最后一次测试的当前结果。

```
[DeviceA] display nqa result admin test1
NQA entry (admin admin, tag test1) test results:
Hop IP 10.1.1.2
Basic Results
  Send operation times: 10          Receive response times: 10
  Min/Max/Average round trip time: 9/21/14
  Square-Sum of round trip time: 2419
Extended Results
```

```

Failures due to timeout: 0
Failures due to internal error: 0
Failures due to other errors: 0
Packets out of sequence: 0
Packets arrived late: 0
Path-Jitter Results
Jitter number: 9
  Min/Max/Average jitter: 1/10/4
Positive jitter number: 6
  Min/Max/Average positive jitter: 1/9/4
  Sum/Square-Sum positive jitter: 25/173
Negative jitter number: 3
  Min/Max/Average negative jitter: 2/10/6
  Sum/Square-Sum positive jitter: 19/153

Hop IP 10.2.2.2
Basic Results
  Send operation times: 10          Receive response times: 10
  Min/Max/Average round trip time: 15/40/28
  Square-Sum of round trip time: 4493
Extended Results
  Failures due to timeout: 0
  Failures due to internal error: 0
  Failures due to other errors: 0
  Packets out of sequence: 0
  Packets arrived late: 0
Path-Jitter Results
Jitter number: 9
  Min/Max/Average jitter: 1/10/4
Positive jitter number: 6
  Min/Max/Average positive jitter: 1/9/4
  Sum/Square-Sum positive jitter: 25/173
Negative jitter number: 3
  Min/Max/Average negative jitter: 2/10/6
  Sum/Square-Sum positive jitter: 19/153

```

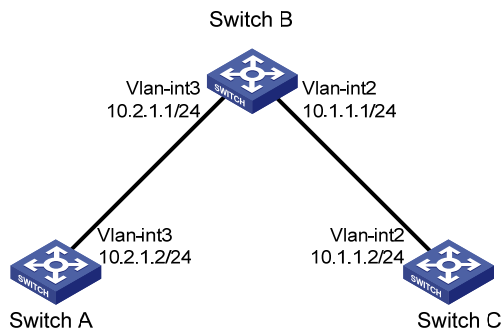
## 1.8.15 NQA联动配置举例

### 1. 组网需求

- Switch A 到达 Switch C 的静态路由下一跳为 Switch B。
- 在 Switch A 上通过静态路由、Track 与 NQA 联动，对到达 Switch C 的静态路由有效性进行实时判断。

## 2. 组网图

图1-16 NQA 联动配置组网图



## 3. 配置步骤

(1) 配置各接口的 IP 地址。（配置过程略）

(2) 在 Switch A 上配置静态路由，并与 Track 项关联。

# 配置到达 Switch C 的静态路由下一跳地址为 10.2.1.1，并配置静态路由与 Track 项 1 关联。

```
<SwitchA> system-view
[SwitchA] ip route-static 10.1.1.2 24 10.2.1.1 track 1
```

(3) 在 Switch A 上配置 NQA 测试组

# 创建管理员名为 admin、操作标签为 test1 的 NQA 测试组。

```
[SwitchA] nqa entry admin test1
```

# 配置测试类型为 ICMP-echo。

```
[SwitchA-nqa-admin-test1] type icmp-echo
```

# 配置目的地址为 10.2.1.1。

```
[SwitchA-nqa-admin-test1-icmp-echo] destination ip 10.2.1.1
```

# 测试频率为 100ms。

```
[SwitchA-nqa-admin-test1-icmp-echo] frequency 100
```

# 配置联动项 1（连续失败 5 次触发联动）。

```
[SwitchA-nqa-admin-test1-icmp-echo] reaction 1 checked-element probe-fail
threshold-type consecutive 5 action-type trigger-only
```

```
[SwitchA-nqa-admin-test1-icmp-echo] quit
```

# 启动 ICMP-echo 探测操作，并一直进行测试。

```
[SwitchA] nqa schedule admin test1 start-time now lifetime forever
```

(4) 在 Switch A 上配置 Track 项

# 配置 Track 项 1，关联 NQA 测试组（管理员为 admin，操作标签为 test1）的联动项 1。

```
[SwitchA] track 1 nqa entry admin test1 reaction 1
```

## 4. 验证配置

# 显示 Switch A 上 Track 项的信息。

```
[SwitchA] display track all
```

Track ID: 1

State: Positive

Duration: 0 days 0 hours 0 minutes 0 seconds

Notification delay: Positive 0, Negative 0 (in seconds)



```

Tracked object:
  NQA entry: admin test1
  Reaction: 1

```

# 显示 Switch A 的路由表。

```
[SwitchA] display ip routing-table
```

```
Destinations : 13          Routes : 13
```

Destination/Mask	Proto	Pre	Cost	NextHop	Interface
0.0.0.0/32	Direct	0	0	127.0.0.1	InLoop0
10.1.1.0/24	Static	60	0	10.2.1.1	Vlan3
10.2.1.0/24	Direct	0	0	10.2.1.2	Vlan3
10.2.1.0/32	Direct	0	0	10.2.1.2	Vlan3
10.2.1.2/32	Direct	0	0	127.0.0.1	InLoop0
10.2.1.255/32	Direct	0	0	10.2.1.2	Vlan3
127.0.0.0/8	Direct	0	0	127.0.0.1	InLoop0
127.0.0.0/32	Direct	0	0	127.0.0.1	InLoop0
127.0.0.1/32	Direct	0	0	127.0.0.1	InLoop0
127.255.255.255/32	Direct	0	0	127.0.0.1	InLoop0
224.0.0.0/4	Direct	0	0	0.0.0.0	NULL0
224.0.0.0/24	Direct	0	0	0.0.0.0	NULL0
255.255.255.255/32	Direct	0	0	127.0.0.1	InLoop0

以上显示信息表示，NQA 测试的结果为下一跳地址 10.2.1.1 可达（Track 项状态为 Positive），配置的静态路由生效。

# 在 Switch B 上删除 VLAN 接口 3 的 IP 地址。

```

<SwitchB> system-view
[SwitchB] interface vlan-interface 3
[SwitchB-Vlan-interface3] undo ip address

```

# 显示 Switch A 上 Track 项的信息。

```

[SwitchA] display track all
Track ID: 1
  State: Negative
  Duration: 0 days 0 hours 0 minutes 0 seconds
  Notification delay: Positive 0, Negative 0 (in seconds)
  Tracked object:
    NQA entry: admin test1
    Reaction: 1

```

# 显示 Switch A 的路由表。

```
[SwitchA] display ip routing-table
```

```
Destinations : 12          Routes : 12
```

Destination/Mask	Proto	Pre	Cost	NextHop	Interface
0.0.0.0/32	Direct	0	0	127.0.0.1	InLoop0
10.2.1.0/24	Direct	0	0	10.2.1.2	Vlan3
10.2.1.0/32	Direct	0	0	10.2.1.2	Vlan3
10.2.1.2/32	Direct	0	0	127.0.0.1	InLoop0

10.2.1.255/32	Direct	0	0	10.2.1.2	Vlan3
127.0.0.0/8	Direct	0	0	127.0.0.1	InLoop0
127.0.0.0/32	Direct	0	0	127.0.0.1	InLoop0
127.0.0.1/32	Direct	0	0	127.0.0.1	InLoop0
127.255.255.255/32	Direct	0	0	127.0.0.1	InLoop0
224.0.0.0/4	Direct	0	0	0.0.0.0	NULL0
224.0.0.0/24	Direct	0	0	0.0.0.0	NULL0
255.255.255.255/32	Direct	0	0	127.0.0.1	InLoop0

以上显示信息表示，NQA 测试的结果为下一跳地址 10.2.1.1 不可达（Track 项状态为 **Negative**），配置的静态路由无效。

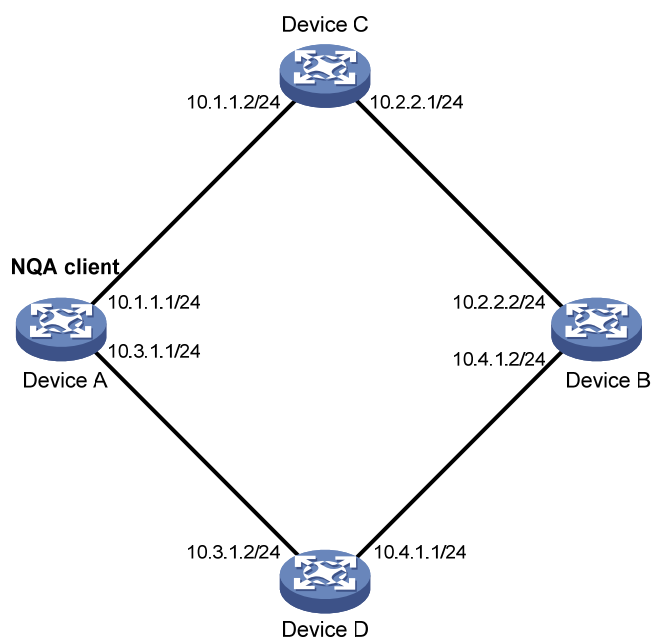
## 1.8.16 ICMP类型的NQA模板配置举例

### 1. 组网需求

外部特性通过引用 ICMP 类型的 NQA 模板，测试本端（Device A）发送的报文是否可以到达指定的目的端（Device B）。

### 2. 组网图

图1-17 ICMP 类型的 NQA 模板配置组网图



### 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

# 创建 ICMP 类型的 NQA 模板，模板名为 icmp，并配置操作中探测报文的目的地址为 10.2.2.2。

```

<DeviceA> system-view
[DeviceA] nqa template icmp icmp
[DeviceA-ngatplt-icmp-icmp] destination ip 10.2.2.2

```

# 配置 ICMP 一次探测的超时时间为 500 毫秒，连续两次探测开始时间的时间间隔为 3000 毫秒。

```
[DeviceA-nqatplt-icmp-icmp] probe timeout 500
```

```
[DeviceA-nqatplt-icmp-icmp] frequency 3000
```

# 配置确定节点有效前需要连续探测成功的次数为 2。当连续探测成功次数达到 2 次时，NQA 客户端把探测成功的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-icmp-icmp] reaction trigger probe-pass 2
```

# 配置确定节点失效需要连续探测失败的次数为 2。当连续探测失败次数达到 2 次时，NQA 客户端把探测失败的消息发送给外部特性，是外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-icmp-icmp] reaction trigger probe-fail 2
```

## 1.8.17 DNS类型的NQA模板配置举例

### 1. 组网需求

外部特性通过引用 DNS 类型的 NQA 模板，测试 Device A 是否可以通过指定的 DNS 服务器将域名 host.com 解析为 IP 地址。

### 2. 组网图

图1-18 DNS 类型的 NQA 模板配置组网图



### 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

# 创建 DNS 类型的 NQA 模板，模板名为 dns。

```
<DeviceA> system-view
```

```
[DeviceA] nqa template dns dns
```

# 配置操作中探测报文的目的地址为 DNS 服务器的 IP 地址 10.2.2.2，要解析的域名为 host.com，解析类型为 A，用户期望返回的 IP 地址为 3.3.3.3。

```
[DeviceA-nqatplt-dns-dns] destination ip 10.2.2.2
```

```
[DeviceA-nqatplt-dns-dns] resolve-target host.com
```

```
[DeviceA-nqatplt-dns-dns] resolve-type A
```

```
[DeviceA-nqatplt-dns-dns] expect ip 3.3.3.3
```

# 配置确定节点有效前需要连续探测成功的次数为 2。当连续探测成功次数达到 2 次时，NQA 客户端把探测成功的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-dns-dns] reaction trigger probe-pass 2
```

# 配置确定节点失效需要连续探测失败的次数为 2。当连续探测失败次数达到 2 次时，NQA 客户端把探测失败的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-dns-dns] reaction trigger probe-fail 2
```

## 1.8.18 TCP类型的NQA模板配置举例

### 1. 组网需求

外部特性通过引用 TCP 类型的 NQA 模板，测试本端（Device A）和服务器（Device B）的端口之间能否建立 TCP 连接，并处理服务器端的应答数据。

### 2. 组网图

图1-19 TCP 类型的 NQA 模板配置组网图



### 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

#### (1) 配置 Device B

# 使能 NQA 服务器，配置监听的 IP 地址为 10.2.2.2，TCP 端口号为 9000。

```
<DeviceB> system-view
[DeviceB] nqa server enable
[DeviceB] nqa server tcp-connect 10.2.2.2 9000
```

#### (2) 配置 Device A

# 创建 TCP 类型的 NQA 模板，模板名为 tcp。

```
<DeviceA> system-view
[DeviceA] nqa template tcp tcp
[DeviceA-nqatplt-tcp-tcp] destination ip 10.2.2.2
[DeviceA-nqatplt-tcp-tcp] destination port 9000
```

# 配置确定节点有效前需要连续探测成功的次数为 2。当连续探测成功次数达到 2 次时，NQA 客户端把探测成功的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-tcp-tcp] reaction trigger probe-pass 2
```

# 配置确定节点失效需要连续探测失败的次数为 2。当连续探测失败次数达到 2 次时，NQA 客户端把探测失败的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-tcp-tcp] reaction trigger probe-fail 2
```

## 1.8.19 TCP Half Open类型的NQA模板配置举例

### 1. 组网需求

外部特性通过引用 TCP Half Open 类型的 NQA 模板，测试本端（Device A）和服务器（Device B）的 TCP 服务是否可用。

## 2. 组网图

图1-20 TCP Half Open 类型的 NQA 模板配置组网图



## 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

### (1) 配置 Device A

# 创建 TCP Half Open 类型的 NQA 模板，模板名为 test。

```
<DeviceA> system-view
```

```
[DeviceA] nqa template tcphalfopen test
```

# 配置 TCP Half Open 探测报文的目的地址为 10.2.2.2。

```
[DeviceA-nqatplt-tcphalfopen-test] destination ip 10.2.2.2
```

# 配置连续探测成功的次数为 2。当连续探测成功次数达到 2 次时，NQA 客户端把探测成功的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-tcphalfopen-test] reaction trigger probe-pass 2
```

# 配置连续探测失败的次数为 2。当连续探测失败次数达到 2 次时，NQA 客户端把探测失败的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-tcphalfopen-test] reaction trigger probe-fail 2
```

## 1.8.20 UDP类型的NQA模板配置举例

### 1. 组网需求

外部特性通过引用 UDP 类型的 NQA 模板，测试本端（Device A）和服务器（Device B）的端口之间的 UDP 报文交互，并处理服务器端的应答数据。

## 2. 组网图

图1-21 UDP 类型的 NQA 模板配置组网图



## 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

### (1) 配置 Device B

# 使能 NQA 服务器，配置监听的 IP 地址为 10.2.2.2，UDP 端口号为 9000。

```

<DeviceB> system-view
[DeviceB] nqa server enable
[DeviceB] nqa server udp-echo 10.2.2.2 9000

```

## (2) 配置 Device A

# 创建 UDP 类型的 NQA 模板，模板名为 udp。

```

<DeviceA> system-view
[DeviceA] nqa template udp udp

```

# 配置 UDP 探测报文的目的地址为 10.2.2.2，目的端口号为 9000。

```

[DeviceA-nqatplt-udp-udp] destination ip 10.2.2.2
[DeviceA-nqatplt-udp-udp] destination port 9000

```

# 配置确定节点有效前需要连续探测成功的次数为 2。当连续探测成功次数达到 2 次时，NQA 客户端把探测成功的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```

[DeviceA-nqatplt-udp-udp] reaction trigger probe-pass 2

```

# 配置确定节点失效需要连续探测失败的次数为 2。当连续探测失败次数达到 2 次时，NQA 客户端把探测失败的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```

[DeviceA-nqatplt-udp-udp] reaction trigger probe-fail 2

```

## 1.8.21 HTTP类型的NQA模板配置举例

### 1. 组网需求

外部特性通过引用 HTTP 类型的 NQA 模板，测试是否可以和指定的 HTTP 服务器之间建立连接，以及能否从 HTTP 服务器获取数据。

### 2. 组网图

图1-22 HTTP 类型的 NQA 模板配置组网图



### 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

# 创建 HTTP 类型的 NQA 模板，模板名为 http。

```

<DeviceA> system-view
[DeviceA] nqa template http http

```

# 配置 HTTP 测试的网址为 https://10.2.2.2/index.htm。

```

[DeviceA-nqatplt-http-http] url http://10.2.2.2/index.htm

```

# 配置 HTTP 测试的操作方式为 get 操作。（get 操作为缺省操作方式，因此，可以不执行本配置）

```

[DeviceA-nqatplt-http-http] operation get

```

# 配置确定节点有效前需要连续探测成功的次数为 2。当连续探测成功次数达到 2 次时，NQA 客户端把探测成功的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-http-http] reaction trigger probe-pass 2
```

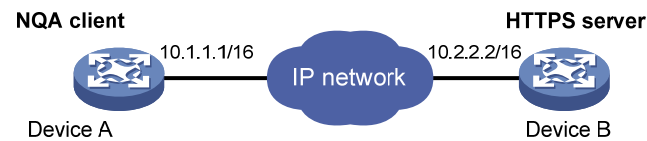
## 1.8.22 HTTPS类型的NQA模板配置举例

### 1. 组网需求

外部特性通过引用 HTTPS 类型的 NQA 模板，测试是否可以和指定的 HTTPS 服务器之间建立连接，以及能否从 HTTPS 服务器获取数据。

### 2. 组网图

图1-23 HTTPS 类型的 NQA 模板配置组网图



### 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

# 在 Device A 上配置 SSL 客户端策略，确保客户端与服务器端可以建立 SSL 安全连接。（配置过程略）

# 创建 HTTPS 类型的 NQA 模板，模板名为 test。

```
<DeviceA> system-view
```

```
[DeviceA] nqa template https https
```

# 配置 HTTPS 测试的网址为 https://10.2.2.2/index.htm。

```
[DeviceA-nqatplt-https-https] url https://10.2.2.2/index.htm
```

# 配置 HTTPS 绑定的 SSL 客户端策略为 abc。

```
[DeviceA-nqatplt-https-https] ssl-client-policy abc
```

# 配置 HTTPS 测试的操作方式为 get 操作。（get 操作为缺省操作方式，可不执行本配置）

```
[DeviceA-nqatplt-https-https] operation get
```

# 配置 HTTPS 测试使用的版本为 1.0。（缺省情况下使用的版本为 1.0，可不执行本配置）

```
[DeviceA-nqatplt-https-https] version v1.0
```

# 配置连续探测成功的次数为 2。当连续探测成功次数达到 2 次时，NQA 客户端把探测成功的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-https-https] reaction trigger probe-pass 2
```

# 配置连续探测失败的次数为 2。当连续探测失败次数达到 2 次时，NQA 客户端把探测失败的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-https-https] reaction trigger probe-fail 2
```

# 配置确定节点失效需要连续探测失败的次数为 2。当连续探测失败次数达到 2 次时，NQA 客户端把探测失败的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-https-https] reaction trigger probe-fail 2
```

## 1.8.23 FTP类型的NQA模板配置举例

### 1. 组网需求

外部特性通过引用 FTP 类型的 NQA 模板，测试 Device A 是否可以和指定的 FTP 服务器 Device B 建立连接，以及能否往 FTP 服务器上传文件。登录 FTP 服务器的用户名为 **admin**，密码为 **systemtest**，要传送到服务器的文件名为 **config.txt**。

### 2. 组网图

图1-24 FTP 类型的 NQA 模板配置组网图



### 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

# 创建 FTP 类型的 NQA 模板，模板名为 **ftp**。

```
<DeviceA> system-view
```

```
[DeviceA] nqa template ftp ftp
```

# 配置操作的目的地址为 FTP 服务器的 IP 地址 **10.2.2.2**。

```
[DeviceA-nqatplt-ftp-ftp] url ftp://10.2.2.2
```

# 配置探测报文的源 IP 地址为 **10.1.1.1**。

```
[DeviceA-nqatplt-ftp-ftp] source ip 10.1.1.1
```

# 配置执行的操作为向 FTP 服务器上传文件 **config.txt**。

```
[DeviceA-nqatplt-ftp-ftp] operation put
```

```
[DeviceA-nqatplt-ftp-ftp] filename config.txt
```

# 配置登录 FTP 服务器的用户名为 **admin**。

```
[DeviceA-nqatplt-ftp-ftp] username admin
```

# 配置登录 FTP 服务器的密码为 **systemtest**。

```
[DeviceA-nqatplt-ftp-ftp] password simple systemtest
```

# 配置确定节点有效前需要连续探测成功的次数为 **2**。当连续探测成功次数达到 **2** 次时，NQA 客户端把探测成功的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-ftp-ftp] reaction trigger probe-pass 2
```

# 配置确定节点失效需要连续探测失败的次数为 **2**。当连续探测失败次数达到 **2** 次时，NQA 客户端把探测失败的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-ftp-ftp] reaction trigger probe-fail 2
```



## 1.8.24 RADIUS类型的NQA模板配置举例

### 1. 组网需求

外部特性通过引用 RADIUS 类型的 NQA 模板，测试 Device A 是否可以和指定的 RADIUS 服务器 Device B 建立连接，并检测 Device B 是否提供服务。RADIUS 用户名为 admin，RADIUS 密码为 systemtest，RADIUS 认证使用的共享密钥为 123456。

### 2. 组网图

图1-25 RADIUS 类型的 NQA 模板配置组网图



### 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

# 配置 RADIUS 服务器 Device B。（配置步骤略）

# 创建 RADIUS 类型的 NQA 模板，模板名为 radius。

```
<DeviceA> system-view
```

```
[DeviceA] nqa template radius radius
```

# 配置 RADIUS 探测报文的目的地址为 10.2.2.2。

```
[DeviceA-nqatplt-radius-radius] destination ip 10.2.2.2
```

# 配置 RADIUS RADIUS 用户名为 admin，RADIUS 密码为明文 systemtest。

```
[DeviceA-nqatplt-radius-radius] username admin
```

```
[DeviceA-nqatplt-radius-radius] password simple systemtest
```

# 配置 RADIUS 用于 RADIUS 认证的共享密钥为明文 123456。

```
[DeviceA-nqatplt-radius-radius] key simple 123456
```

# 配置确定节点有效前需要连续探测成功的次数为 2。当连续探测成功次数达到 2 次时，NQA 客户端把探测成功的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-radius-radius] reaction trigger probe-pass 2
```

# 配置确定节点失效需要连续探测失败的次数为 2。当连续探测失败次数达到 2 次时，NQA 客户端把探测失败的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-radius-radius] reaction trigger probe-fail 2
```

## 1.8.25 SSL类型的NQA模板配置举例

### 1. 组网需求

外部特性通过引用 SSL 类型的 NQA 模板，本端（Device A）通过引用指定的 SSL 客户端策略与 SSL 服务器（Device B）建立 SSL 连接，从而测试 SSL 客户端和服务器的连通性和性能。

## 2. 组网图

图1-26 SSL 类型的 NQA 模板配置组网图



## 3. 配置步骤

# 配置各接口的 IP 地址。（配置过程略）

# 配置静态路由或动态路由协议，确保各设备之间路由可达。（配置过程略）

# 在 Device A 上配置 SSL 客户端策略，确保客户端与服务器端可以建立 SSL 安全连接。（配置过程略）

### (1) 配置 Device A

# 创建 SSL 类型的 NQA 模板，模板名为 ssl。

```
<DeviceA> system-view
[DeviceA] nqa template ssl ssl
```

# 配置 SSL 探测报文的目的地址为 10.2.2.2，目的端口号为 9000。

```
[DeviceA-nqatplt-ssl-ssl] destination ip 10.2.2.2
[DeviceA-nqatplt-ssl-ssl] destination port 9000
```

# 配置 SSL 绑定的 SSL 客户端策略为 abc。

```
[DeviceA-nqatplt-ssl-ssl] ssl-client-policy abc
```

# 配置连续探测成功的次数为 2。当连续探测成功次数达到 2 次时，NQA 客户端把探测成功的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-ssl-ssl] reaction trigger probe-pass 2
```

# 配置连续探测失败的次数为 2。当连续探测失败次数达到 2 次时，NQA 客户端把探测失败的消息发送给外部特性，使外部特性能利用 NQA 测试的结果进行相应处理。

```
[DeviceA-nqatplt-ssl-ssl] reaction trigger probe-fail 2
```

# 目 录

1 NTP.....	1-1
1.1 NTP简介.....	1-1
1.1.1 NTP应用场合.....	1-1
1.1.2 NTP基本工作原理.....	1-1
1.1.3 NTP网络结构及时钟层数.....	1-2
1.1.4 NTP的工作模式.....	1-3
1.1.5 NTP安全功能.....	1-4
1.1.6 协议规范.....	1-6
1.2 NTP配置限制和指导.....	1-6
1.3 NTP配置任务简介.....	1-6
1.4 开启NTP服务.....	1-7
1.5 配置不同工作模式下的NTP.....	1-7
1.5.1 配置客户端/服务器模式下的NTP.....	1-7
1.5.2 配置对等体模式下的NTP.....	1-8
1.5.3 配置广播模式下的NTP.....	1-8
1.5.4 配置组播模式下的NTP.....	1-9
1.6 配置本地时钟作为参考时钟.....	1-10
1.7 配置NTP服务的访问控制权限.....	1-10
1.8 配置NTP验证功能.....	1-11
1.8.1 配置客户端/服务器模式的NTP验证功能.....	1-11
1.8.2 配置对等体模式的NTP验证功能.....	1-12
1.8.3 配置广播模式的NTP验证功能.....	1-14
1.8.4 配置组播模式的NTP验证功能.....	1-15
1.9 控制NTP报文的收发.....	1-17
1.9.1 配置NTP报文的源接口.....	1-17
1.9.2 关闭接口接收NTP报文功能.....	1-18
1.9.3 配置动态会话的最大数目.....	1-18
1.9.4 配置NTP报文的DSCP优先级.....	1-19
1.10 NTP显示和维护.....	1-19
1.11 NTP典型配置举例.....	1-20
1.11.1 配置NTP客户端/服务器模式.....	1-20
1.11.2 配置IPv6 NTP客户端/服务器模式.....	1-21
1.11.3 配置NTP对等体模式.....	1-22

1.11.4 配置IPv6 NTP对等体模式 .....	1-24
1.11.5 配置NTP广播模式 .....	1-25
1.11.6 配置NTP组播模式 .....	1-27
1.11.7 配置IPv6 NTP组播模式 .....	1-30
1.11.8 配置带验证功能的NTP客户端/服务器模式.....	1-33
1.11.9 配置带验证功能的NTP广播模式 .....	1-35
<b>2 SNTP.....</b>	<b>2-1</b>
2.1 SNTP简介.....	2-1
2.1.1 SNTP的工作模式 .....	2-1
2.1.2 协议规范 .....	2-1
2.2 SNTP配置限制和指导.....	2-1
2.3 SNTP配置任务简介.....	2-1
2.4 开启SNTP服务.....	2-2
2.5 为SNTP客户端指定NTP服务器.....	2-2
2.6 配置SNTP验证功能.....	2-2
2.7 SNTP显示和维护.....	2-3
2.8 SNTP典型配置举例.....	2-4
2.8.1 SNTP基本组网配置举例 .....	2-4

# 1 NTP

## 1.1 NTP简介

在大型的网络中，如果依靠管理员手工配置来修改网络中各台设备的系统时间，不但工作量巨大，而且也不能保证时间的精确性。**NTP**（**Network Time Protocol**，网络时间协议）可以用来在分布式时间服务器和客户端之间进行时间同步，使网络内所有设备的时间保持一致，并提供较高的时间同步精度。**NTP** 采用的传输层协议为 **UDP**，使用的 **UDP** 端口号为 **123**。

---



说明

这里的“分布式”指的是运行 **NTP** 的设备既可以与其他设备的时间同步，又可以作为时间服务器为其他设备提供时间同步。

---

### 1.1.1 NTP应用场合

**NTP** 主要应用于需要网络中所有设备的时间保持一致的场合，比如：

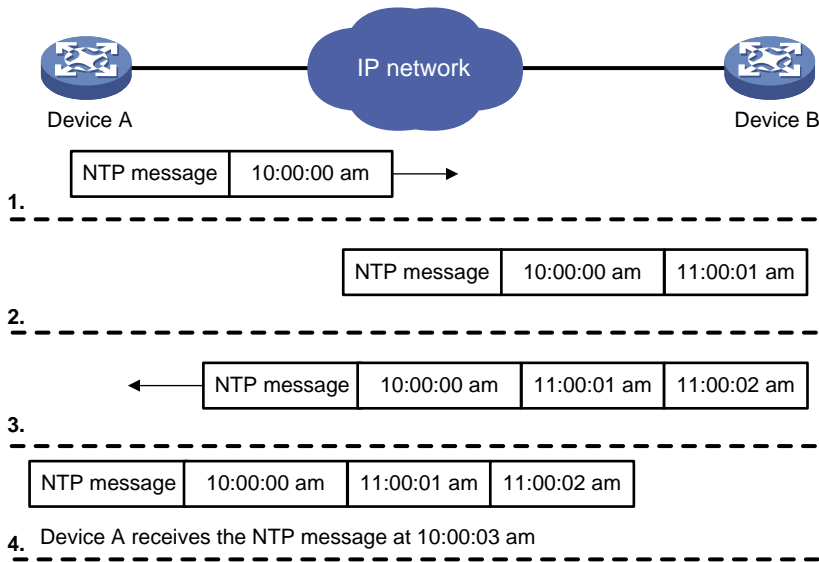
- 需要以时间作为参照依据，对从不同设备采集来的日志信息、调试信息进行分析的网络管理系统。
- 对设备时间一致性有要求的计费系统。
- 多个系统协同处理同一个比较复杂的事件的场合。此时，为保证正确的执行顺序，多个系统的时间必须保持一致。

### 1.1.2 NTP基本工作原理

**NTP**的基本工作原理如 [图 1-1](#) 所示。**Device A**和**Device B**通过网络相连，**Device A**和**Device B**的时间不同，需要通过**NTP**实现时间的自动同步。为便于理解，作如下假设：

- 在 **Device A** 和 **Device B** 的时间同步之前，**Device A** 的时间设定为 10:00:00 am，**Device B** 的时间设定为 11:00:00 am。
- **Device B** 作为 **NTP** 时间服务器，即 **Device A** 与 **Device B** 的时间同步。
- **NTP** 报文从 **Device A** 到 **Device B**、从 **Device B** 到 **Device A** 单向传输所需要的时间均为 1 秒。
- **Device B** 处理 **NTP** 报文所需的时间是 1 秒。

图1-1 NTP 基本工作原理图



Device A 和 Device B 时间同步的工作过程如下：

- (1) Device A 发送一个 NTP 报文给 Device B，该报文带有它离开 Device A 时的时间戳，该时间戳为 10:00:00 am（T1）。
- (2) 当此 NTP 报文到达 Device B 时，Device B 在 NTP 报文上增加该报文到达 Device B 时的时间戳，该时间戳为 11:00:01 am（T2）。
- (3) 当此 NTP 报文离开 Device B 时，Device B 再在 NTP 报文上增加该报文离开 Device B 时的时间戳，该时间戳为 11:00:02 am（T3）。
- (4) 当 Device A 接收到该响应报文时，Device A 的本地时间为 10:00:03 am（T4）。

至此，Device A 可以根据上述时间戳计算两个重要的参数：

- NTP 报文的往返时延  $Delay = (T4 - T1) - (T3 - T2) = 2 \text{ 秒}$ 。
- Device A 相对 Device B 的时间差  $Offset = ((T2 - T1) + (T3 - T4)) / 2 = 1 \text{ 小时}$ 。

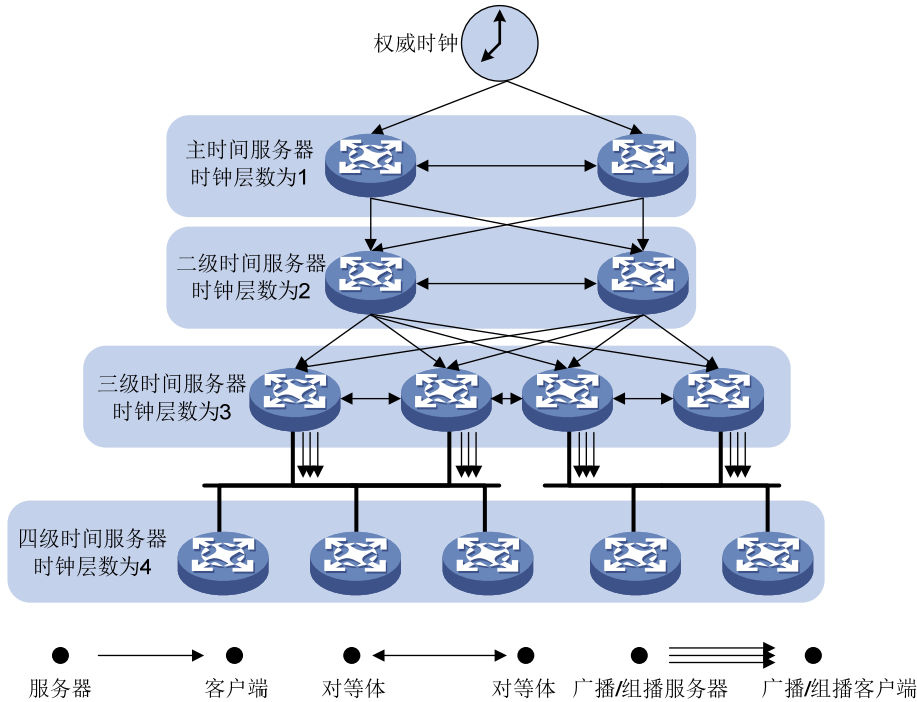
这样，Device A 就能够根据这些信息来设定自己的时间，使之与 Device B 的时间同步。

以上内容只是对 NTP 工作原理的一个粗略描述，详细内容请参阅相关的协议规范。

1.1.3 NTP网络结构及时钟层数

NTP 通过时钟层数来定义时钟的准确度。时钟层数的取值范围为 1~16，取值越小，时钟准确度越高。层数为 1~15 的时钟处于同步状态；层数为 16 的时钟处于未同步状态。

图1-2 NTP 网络结构



如 图 1-2 所示，实际网络中，通常将从权威时钟（如原子时钟）获得时间同步的NTP服务器的层数设置为 1，并将其作为主时间服务器，为网络中其他设备的时钟提供时间同步。网络中的设备与主时间服务器的NTP距离，即NTP同步链上NTP服务器的数目，决定了设备上时钟的层数。例如，从主时间服务器获得时间同步的设备的时钟层数为 2，即比主时间服务器的时钟层数大 1；从时钟层数为 2 的时间服务器获得时间同步的设备的时钟层数为 3，以此类推。

为了保证时间的准确性和可靠性，可以为一台设备指定多个时间服务器，设备根据时钟层数等参数进行时钟过滤和选择，从多个时间服务器中选择最优的时钟，与其同步。设备选中的时钟称为参考时钟。时钟优选过程的详细介绍，请参阅相关的协议规范。

在某些网络中，例如无法与外界通信的孤立网络，网络中的设备无法与权威时钟进行时间同步。此时，可以从该网络中选择一台时钟较为准确的设备，指定该设备与本地时钟进行时间同步，即采用本地时钟作为参考时钟，使得该设备的时钟处于同步状态。该设备作为时间服务器为网络中的其他设备提供时间同步，从而实现整个网络的时间同步。

### 1.1.4 NTP的工作模式

NTP 支持以下几种工作模式：

- 客户端/服务器模式
- 对等体模式
- 广播模式
- 组播模式

用户可以根据需要选择一种或几种工作模式进行时间同步。各种模式的详细介绍，如 表 1-1 所示。本文中 NTP 服务器或服务器指的是客户端/服务器模式中工作在服务器模式的设备；时间服务器指的是所有能够提供时间同步的设备，包括 NTP 服务器、NTP 对等体、广播服务器和组播服务器。

表1-1 NTP 模式介绍

模式	工作过程	时间同步方向	应用场合
客户端/服务器模式	<p>客户端上需要手工指定NTP服务器的地址。客户端向NTP服务器发送NTP时间同步报文。NTP服务器收到报文后会自动工作在服务器模式，并回复应答报文</p> <p>如果客户端可以从多个时间服务器获取时间同步，则客户端收到应答报文后，进行时钟过滤和选择，并与优选的时钟进行时间同步</p>	<ul style="list-style-type: none"> <li>客户端能够与 NTP 服务器的时间同步</li> <li>NTP 服务器无法与客户端的时间同步</li> </ul>	如 <a href="#">图1-2</a> 所示，该模式通常用于下级的设备从上级的时间服务器获取时间同步
对等体模式	<p>主动对等体（Symmetric active peer）上需要手工指定被动对等体（Symmetric passive peer）的地址。主动对等体向被动对等体发送NTP时间同步报文。被动对等体收到报文后会自动工作在被动对等体模式，并回复应答报文</p> <p>如果主动对等体可以从多个时间服务器获取时间同步，则主动对等体收到应答报文后，进行时钟过滤和选择，并与优选的时钟进行时间同步</p>	<ul style="list-style-type: none"> <li>主动对等体和被动对等体的时间可以互相同步</li> <li>如果双方的时钟都处于同步状态，则层数大的时钟与层数小的时钟的时间同步</li> </ul>	如 <a href="#">图1-2</a> 所示，该模式通常用于同级的设备间互相同步，以便在同级的设备间形成备份。如果某台设备与所有上级时间服务器的通信出现故障，则该设备仍然可以从同级的时间服务器获得时间同步
广播模式	<p>广播服务器周期性地向广播地址255.255.255.255发送NTP时间同步报文。广播客户端侦听来自广播服务器的广播报文，根据接收的广播报文将设备的时间与广播服务器的时间进行同步</p> <p>广播客户端接收到广播服务器发送的第一个NTP报文后，会与广播服务器进行报文交互，以获得报文的往返时延，为时间同步提供必要的参数。之后，只有广播服务器单方向发送报文</p>	<ul style="list-style-type: none"> <li>广播客户端能够与广播服务器的时间同步</li> <li>广播服务器无法与广播客户端的时间同步</li> </ul>	<p>广播服务器广播发送时间同步报文，可以同时同步同一个子网中多个广播客户端的时间。如<a href="#">图1-2</a>所示，使用同一个时间服务器为同一个子网中的大量设备提供时间同步时，可以使用广播模式，以简化网络配置</p> <p>由于只有广播服务器单方向发送报文，广播模式的时间准确度不如客户端/服务器模式和对等体模式</p>
组播模式	<p>组播服务器周期性地向指定的组播地址发送NTP时间同步报文。客户端侦听来自服务器的组播报文，根据接收的组播报文将设备的时间与组播服务器的时间进行同步</p>	<ul style="list-style-type: none"> <li>组播客户端能够与组播服务器的时间同步</li> <li>组播服务器无法与组播客户端的时间同步</li> </ul>	<p>组播模式对广播模式进行了扩展，组播服务器可以同时为同一子网、不同子网的多个组播客户端提供时间同步</p> <p>组播模式的时间准确度不如客户端/服务器模式和对等体模式</p>

### 1.1.5 NTP安全功能

为了提高时间同步的安全性，NTP 提供了 NTP 服务的访问控制权限和 NTP 验证功能。

#### 1. NTP服务的访问控制权限

本功能是指利用 ACL 限制对端设备对本地设备上 NTP 服务的访问控制权限。NTP 服务的访问控制权限从高到低依次为 **peer**、**server**、**synchronization**、**query**。



- **peer:** 完全访问权限。该权限既允许对端设备向本地设备的时间同步，对本地设备进行控制查询（查询 NTP 的一些状态，比如告警信息、验证状态、时间服务器信息等），同时本地设备也可以向对端设备的时间同步。
- **server:** 服务器访问与查询权限。该权限允许对端设备向本地设备的时间同步，对本地设备进行控制查询，但本地设备不会向对端设备的时间同步。
- **synchronization:** 仅具有访问服务器的权限。该权限只允许对端设备向本地设备的时间同步，但不能进行控制查询。
- **query:** 仅具有控制查询的权限。该权限只允许对端设备对本地设备的 NTP 服务进行控制查询，但是不能向本地设备的时间同步。

当设备接收到 NTP 服务请求时，会按照权限从高到低的顺序依次进行匹配。匹配原则为：

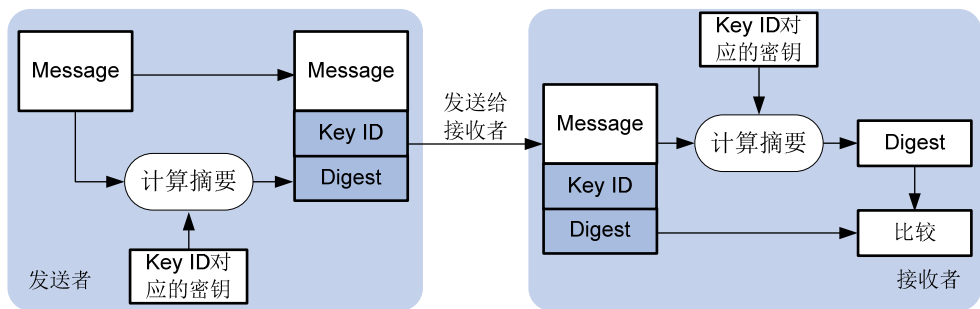
- 如果没有指定权限应用的 ACL 或权限应用的 ACL 尚未创建，则继续匹配下一个权限。
- 如果所有的权限都没有应用 ACL 或权限应用的 ACL 尚未创建，则所有对端设备对本地设备 NTP 服务的访问控制权限均为 **peer**。
- 如果存在应用了 ACL 的权限，且该 ACL 已经创建，则只有 NTP 服务请求匹配了某个权限应用的 ACL 中的 **permit** 规则，发送该 NTP 服务请求的对端设备才会具有该访问控制权限。其他情况下（NTP 服务请求匹配某个权限应用的 ACL 中的 **deny** 规则或没有匹配任何权限的任何规则），发送该 NTP 服务请求的对端设备不具有任何权限。

配置 NTP 服务的访问控制权限，仅提供了一种最小限度的安全措施，更安全的方法是使用 NTP 验证功能。

## 2. NTP验证功能

在一些对时间同步的安全性要求较高的网络中，运行 NTP 协议时需要使用 NTP 验证功能。NTP 验证功能可以用来验证接收到的 NTP 报文的合法性。只有报文通过验证后，设备才会接收该报文，并从中获取时间同步信息；否则，设备会丢弃该报文。从而，保证设备不会与非法的时间服务器进行时间同步，避免时间同步错误。

图1-3 NTP 验证功能示意图



如 图 1-3 所示，NTP验证功能的工作过程为：

- (1) NTP 报文发送者利用密钥 ID 标识的密钥对 NTP 报文进行 MD5/HMAC 运算，并将计算出来的摘要信息连同 NTP 报文和密钥 ID 一起发送给接收者。
- (2) 接收者接收到该 NTP 报文后，根据报文中的密钥 ID 找到对应的密钥，并利用该密钥对报文进行 MD5/HMAC 运算。接收者将运算结果与报文中的摘要信息比较，依据比较结果，有以下三种情况：

- 比较结果不相同，则丢弃该报文。
- 比较结果相同，且本地无需建立NTP会话，则应答该报文。建立会话的详细介绍，请参见“[配置动态会话的最大数目](#)”。
- 比较结果相同，且本地需要建立或已存在NTP会话，则检查NTP报文发送者是否有权在本端使用该密钥 ID，检查通过，则接收该报文；否则，丢弃该报文。

### 1.1.6 协议规范

与 NTP 相关的协议规范有：

- RFC 1305: Network Time Protocol (Version 3) Specification, Implementation and Analysis
- RFC 5905: Network Time Protocol Version 4: Protocol and Algorithms Specification

## 1.2 NTP配置限制和指导

配置 NTP 时，需要注意：

- 设备上不能同时配置 NTP 和 SNTP 功能。
- NTP 功能中所指的“接口”为三层接口。
- 为保证时间同步的准确性，不建议用户在组网中配置两个或者两个以上的时钟源，以免造成时钟震荡，甚至出现无法同步的情况。
- 建议用户不要在聚合成员口上进行 NTP 相关配置。
- 用户需要保证通过 `clock protocol` 命令，配置以 NTP 方式获取系统时间。有关 `clock protocol` 命令的详细介绍，请参见“基础配置命令参考”中的“设备管理”。

## 1.3 NTP配置任务简介

NTP 配置任务如下：

- (1) [开启NTP服务](#)
- (2) [配置不同工作模式下的NTP](#)
  - [配置客户端/服务器模式下的NTP](#)
  - [配置对等体模式下的NTP](#)
  - [配置广播模式下的NTP](#)
  - [配置组播模式下的NTP](#)
- (3) （可选）[配置本地时钟作为参考时钟](#)
- (4) （可选）[配置NTP服务的访问控制权限](#)
- (5) （可选）[配置NTP验证功能](#)
  - [配置客户端/服务器模式的NTP验证功能](#)
  - [配置对等体模式的NTP验证功能](#)
  - [配置广播模式的NTP验证功能](#)
  - [配置组播模式的NTP验证功能](#)
- (6) （可选）[控制NTP报文的收发](#)
  - [配置NTP报文的源接口](#)

- [关闭接口接收NTP报文功能](#)
- [配置动态会话的最大数目](#)
- [配置NTP报文的DSCP优先级](#)

## 1.4 开启NTP服务

### 1. 配置准备

NTP 服务与 SNTP 服务互斥，同一时刻只能开启其中一个服务。因此，在开启 NTP 服务器之前，请确保 SNTP 服务关闭。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启 NTP 服务。

```
ntp-service enable
```

缺省情况下，NTP 服务处于关闭状态。

## 1.5 配置不同工作模式下的NTP

### 1.5.1 配置客户端/服务器模式下的NTP

#### 1. 配置限制和指导

- 当设备采用客户端/服务器模式时，需要在客户端上指定服务器的地址。
- 当服务器端的时钟层数大于或等于客户端的时钟层数时，客户端将不会与其同步。
- 可以通过多次执行 **ntp-service unicast-server** 命令和 **ntp-service ipv6 unicast-server** 命令为设备指定多个服务器。
- 服务器需要通过与其他设备同步或配置本地时钟作为参考时钟等方式，使得自己的时钟处于同步状态，否则客户端不会将自己的时间与服务器的时间同步。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 为设备指定 NTP 服务器。

(IPv4 网络)

```
ntp-service unicast-server { server-name | ip-address }  
[ authentication-keyid keyid | maxpoll maxpoll-interval | minpoll  
minpoll-interval | priority | source interface-type interface-number |  
version number ] *
```

(IPv6 网络)

```
ntp-service ipv6 unicast-server { server-name | ipv6-address }  
[ authentication-keyid keyid | maxpoll maxpoll-interval | minpoll
```

```
minpoll-interval | priority | source interface-type interface-number ]  
*
```

缺省情况下，未指定 NTP 服务器。

## 1.5.2 配置对等体模式下的NTP

### 1. 配置限制和指导

- 当设备采用对等体模式时，需要在主动对等体上指定被动对等体的地址。
- 被动对等体上需要执行 **ntp-service enable** 命令来开启 NTP 服务，否则被动对等体不会处理来自主动对等体的 NTP 报文。
- 主动对等体和被动对等体的时钟至少要有有一个处于同步状态，否则它们的时间都将无法同步。
- 可以通过多次执行 **ntp-service unicast-peer** 命令或 **ntp-service ipv6 unicast-peer** 命令为设备指定多个被动对等体。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 指定设备的被动对等体。

(IPv4 网络)

```
ntp-service unicast-peer { peer-name | ip-address }  
[ authentication-keyid keyid | maxpoll maxpoll-interval | minpoll  
minpoll-interval | priority | source interface-type interface-number ]  
version number ] *
```

(IPv6 网络)

```
ntp-service ipv6 unicast-peer { peer-name | ipv6-address }  
[ authentication-keyid keyid | maxpoll maxpoll-interval | minpoll  
minpoll-interval | priority | source interface-type interface-number ]  
*
```

缺省情况下，未指定被动对等体。

## 1.5.3 配置广播模式下的NTP

### 1. 配置限制和指导

- 当设备采用广播模式时，广播服务器端和广播客户端上都需要进行配置。
- 配置 NTP 的广播模式之前，广播服务器需要通过与其他设备同步或配置本地时钟作为参考时钟等方式，使得自己的时钟处于同步状态，否则广播客户端不会将自己的时间与广播服务器的时间同步。

### 2. 配置广播客户端

- (1) 进入系统视图。

```
system-view
```

- (2) 进入要接收 NTP 广播报文的接口。

```
interface interface-type interface-number
```

- (3) 配置设备工作在 NTP 广播客户端模式。

```
ntp-service broadcast-client
```

缺省情况下，未配置 NTP 工作模式。

执行本命令后，设备将通过当前接口接收 NTP 广播报文。

### 3. 配置广播服务器

- (1) 进入系统视图。

```
system-view
```

- (2) 进入要发送 NTP 广播报文的接口。

```
interface interface-type interface-number
```

- (3) 配置设备工作在 NTP 广播服务器模式。

```
ntp-service broadcast-server [ authentication-keyid keyid | version number ] *
```

缺省情况下，未配置 NTP 工作模式。

执行本命令后，设备将通过当前接口周期性发送 NTP 广播报文。

## 1.5.4 配置组播模式下的NTP

### 1. 配置限制和指导

- 设备采用组播模式时，在组播服务器端和组播客户端上都需要进行本配置。
- 配置 NTP 的组播模式之前，组播服务器需要通过与其他设备同步或配置本地时钟作为参考时钟等方式，使得自己的时钟处于同步状态，否则组播客户端不会将自己的时间与组播服务器的时间同步。

### 2. 配置组播客户端

- (1) 进入系统视图。

```
system-view
```

- (2) 进入要接收 NTP 组播报文的接口。

```
interface interface-type interface-number
```

- (3) 配置设备工作在 NTP 组播客户端模式。

(IPv4 网络)

```
ntp-service multicast-client [ ip-address ]
```

(IPv6 网络)

```
ntp-service ipv6 multicast-client ipv6-address
```

缺省情况下，未配置 NTP 工作模式。

执行本命令后，设备将通过当前接口接收 NTP 组播报文。

### 3. 配置组播服务器

- (1) 进入系统视图。

**system-view**

- (2) 进入要发送 NTP 组播报文的接口。

**interface** *interface-type interface-number*

- (3) 配置设备工作在 NTP 组播服务器模式。

(IPv4 网络)

**ntp-service multicast-server** [ *ip-address* ] [ **authentication-keyid** *keyid* | **ttl** *ttl-number* | **version** *number* ] \*

(IPv6 网络)

**ntp-service ipv6 multicast-server** *ipv6-address* [ **authentication-keyid** *keyid* | **ttl** *ttl-number* ] \*

缺省情况下，未配置 NTP 工作模式。

执行本命令后，设备将通过当前接口周期性发送 NTP 组播报文。

## 1.6 配置本地时钟作为参考时钟

### 1. 功能简介

本配置用来指定设备与本地时钟进行时间同步，使得该设备的时钟处于同步状态。

### 2. 配置限制和指导

- 配置本地时钟作为参考时钟后，本地设备的时钟将处于同步状态，可以作为时间服务器为网络中其他设备的时钟提供时间同步。如果本地设备的时钟不正确，则会导致网络中设备的时间错误，请谨慎使用本配置。
- 设备断电重启后，系统时间会变成系统初始化的时间。建议不要在设备上配置本地时钟作为参考时钟，并且不要将设备指定为时间服务器。
- 不同的设备在时钟精度方面存在差异，一个网段中建议只配置一个精度最高参考时钟，否则可能出现网络震荡，导致时钟无法同步。

### 3. 配置准备

配置本地时钟作为参考时钟之前，建议先校准本地系统时间。

### 4. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 配置本地时钟作为参考时钟。

**ntp-service refclock-master** [ *ip-address* ] [ *stratum* ]

缺省情况下，设备未采用本地时钟作为参考时钟。

## 1.7 配置NTP服务的访问控制权限

### 1. 配置准备

在配置对本地设备 NTP 服务的访问控制权限时，需要创建并配置与访问权限关联的 ACL。ACL 的配置方法请参见“ACL 和 QoS 配置指导”中的“ACL”。

2. 配置步骤

- (1) 进入系统视图。  
`system-view`
- (2) 配置对端设备对本地设备 NTP 服务的访问控制权限。  
(IPv4 网络)  
`ntp-service { peer | query | server | synchronization } acl  
ipv4-acl-number`  
(IPv6 网络)  
`ntp-service ipv6 { peer | query | server | synchronization } acl  
ipv4-acl-number`  
缺省情况下，对端设备对本地设备 NTP 服务的访问控制权限为 **peer**（完全访问权限）。

1.8 配置NTP验证功能

1.8.1 配置客户端/服务器模式的NTP验证功能

1. 配置限制和指导

客户端和服务端上需要配置相同的密钥 ID、验证算法及密钥值，并且保证对端有权在本端使用该密钥 ID 进行验证，否则会导致 NTP 验证失败。

客户端和服务端上进行不同的配置时，NTP 验证结果有所不同，详细介绍请参见 [表 1-2](#)。其中，表格中的“-”表示不管此项是否配置。

表1-2 客户端和服务端上进行不同配置时的 NTP 验证结果

客户端			服务器		结果
身份验证	关联密钥	关联密钥存在且为可信密钥	身份验证	关联密钥存在且为可信密钥	
是	是	是	是	是	身份验证成功
是	是	是	是	否	身份验证失败
是	是	是	否	-	身份验证失败
是	是	否	-	-	身份验证失败
是	否	-	-	-	不进行身份验证
否	-	-	-	-	不进行身份验证

2. 配置客户端的NTP验证

- (1) 进入系统视图。  
`system-view`
- (2) 开启 NTP 身份验证功能。  
`ntp-service authentication enable`  
缺省情况下，NTP 身份验证功能处于关闭状态。

- (3) 配置 NTP 身份验证密钥。

```
ntp-service authentication-keyid keyid authentication-mode  
{ hmac-sha-1 | hmac-sha-256 | hmac-sha-384 | hmac-sha-512 | md5 } { cipher  
| simple } string [ acl ipv4-acl-number | ipv6 acl ipv6-acl-number ] *
```

缺省情况下，未配置 NTP 身份验证密钥。

- (4) 配置指定密钥为可信密钥。

```
ntp-service reliable authentication-keyid keyid
```

缺省情况下，未指定可信密钥。

- (5) 将指定密钥与对应的 NTP 服务器关联。

(IPv4 网络)

```
ntp-service unicast-server { server-name | ip-address }  
authentication-keyid keyid
```

(IPv6 网络)

```
ntp-service ipv6 unicast-server { server-name | ipv6-address }  
authentication-keyid keyid
```

### 3. 配置服务器端的NTP验证

- (1) 进入系统视图。

```
system-view
```

- (2) 开启 NTP 身份验证功能。

```
ntp-service authentication enable
```

缺省情况下，NTP 身份验证功能处于关闭状态。

- (3) 配置 NTP 身份验证密钥。

```
ntp-service authentication-keyid keyid authentication-mode  
{ hmac-sha-1 | hmac-sha-256 | hmac-sha-384 | hmac-sha-512 | md5 } { cipher  
| simple } string [ acl ipv4-acl-number | ipv6 acl ipv6-acl-number ] *
```

缺省情况下，未配置 NTP 身份验证密钥。

- (4) 配置指定密钥为可信密钥。

```
ntp-service reliable authentication-keyid keyid
```

缺省情况下，未指定可信密钥。

## 1.8.2 配置对等体模式的NTP验证功能

### 1. 配置限制和指导

主动对等体和被动对等体上需要配置相同的密钥 ID、验证算法及密钥值，并且保证对端有权在本端使用该密钥 ID 进行验证，否则会导致 NTP 验证失败。

主动对等体和被动对等体上进行不同的配置时，NTP 验证结果有所不同，详细介绍请参见 [表 1-3](#)。其中，表格中的“-”表示不管此项是否配置。



表1-3 主动对等体和被动对等体上进行不同配置时的 NTP 验证结果

主动对等体				被动对等体		结果
身份验证	关联密钥	关联密钥存在且为可信密钥	时钟层数	身份验证	关联密钥存在且为可信密钥	
是	是	是	-	是	是	身份验证成功
是	是	是	-	是	否	身份验证失败
是	是	是	-	否	-	身份验证失败
是	否	-	-	是	-	身份验证失败
是	否	-	-	否	-	不进行身份验证
否	-	-	-	是	-	身份验证失败
否	-	-	-	否	-	不进行身份验证
是	是	否	大于被动对等体	-	-	身份验证失败
是	是	否	小于被动对等体	是	-	身份验证失败
是	是	否	小于被动对等体	否	-	不进行身份验证

## 2. 配置主动对等体的NTP验证

- (1) 进入系统视图。

```
system-view
```

- (2) 开启 NTP 身份验证功能。

```
ntp-service authentication enable
```

缺省情况下，NTP 身份验证功能处于关闭状态。

- (3) 配置 NTP 身份验证密钥。

```
ntp-service authentication-keyid keyid authentication-mode
{ hmac-sha-1 | hmac-sha-256 | hmac-sha-384 | hmac-sha-512 | md5 } { cipher
| simple } string [ acl ipv4-acl-number | ipv6 acl ipv6-acl-number ] *
```

缺省情况下，未配置 NTP 身份验证密钥。

- (4) 配置指定密钥为可信密钥。

```
ntp-service reliable authentication-keyid keyid
```

缺省情况下，未指定可信密钥。

- (5) 将指定密钥与对应的被动对等体关联。

(IPv4 网络)

```
ntp-service unicast-peer { ip-address | peer-name }
authentication-keyid keyid
```

(IPv6 网络)

```
ntp-service ipv6 unicast-peer { ipv6-address | peer-name }
authentication-keyid keyid
```

### 3. 配置被动对等体的NTP验证

- (1) 进入系统视图。

```
system-view
```

- (2) 开启 NTP 身份验证功能。

```
ntp-service authentication enable
```

缺省情况下，NTP 身份验证功能处于关闭状态。

- (3) 配置 NTP 身份验证密钥。

```
ntp-service authentication-keyid keyid authentication-mode
{ hmac-sha-1 | hmac-sha-256 | hmac-sha-384 | hmac-sha-512 | md5 } { cipher
| simple } string [ acl ipv4-acl-number | ipv6 acl ipv6-acl-number ] *
```

缺省情况下，未配置 NTP 身份验证密钥。

- (4) 配置指定密钥为可信密钥。

```
ntp-service reliable authentication-keyid keyid
```

缺省情况下，未指定可信密钥。

## 1.8.3 配置广播模式的NTP验证功能

### 1. 配置限制和指导

广播服务器和广播客户端上需要配置相同的密钥 ID、验证算法及密钥值，并且保证对端有权在本端使用该密钥 ID 进行验证，否则会导致 NTP 验证失败。

广播客户端和广播服务器上进行不同的配置时，NTP 验证结果有所不同，详细介绍请参见 [表 1-4](#)。其中，表格中的“-”表示不管此项是否配置。

表1-4 广播客户端和广播服务器上不同配置时的 NTP 验证结果

广播服务器			广播客户端		结果
身份验证	关联密钥	关联密钥存在且为可信密钥	身份验证	关联密钥存在且为可信密钥	
是	是	是	是	是	身份验证成功
是	是	是	是	否	身份验证失败
是	是	是	否	-	身份验证失败
是	是	否	是	-	身份验证失败
是	是	否	否	-	不进行身份验证
是	否	-	是	-	身份验证失败
是	否	-	否	-	不进行身份验证
否	-	-	是	-	身份验证失败
否	-	-	否	-	不进行身份验证

## 2. 配置广播客户端的NTP验证

- (1) 进入系统视图。

```
system-view
```

- (2) 开启 NTP 身份验证功能。

```
ntp-service authentication enable
```

缺省情况下，NTP 身份验证功能处于关闭状态。

- (3) 配置 NTP 身份验证密钥。

```
ntp-service authentication-keyid keyid authentication-mode  
{ hmac-sha-1 | hmac-sha-256 | hmac-sha-384 | hmac-sha-512 | md5 } { cipher  
| simple } string [ acl ipv4-acl-number | ipv6 acl ipv6-acl-number ] *
```

缺省情况下，未配置 NTP 身份验证密钥。

- (4) 配置指定密钥为可信密钥。

```
ntp-service reliable authentication-keyid keyid
```

缺省情况下，未指定可信密钥。

## 3. 配置广播服务器端的NTP验证

- (1) 进入系统视图。

```
system-view
```

- (2) 开启 NTP 身份验证功能。

```
ntp-service authentication enable
```

缺省情况下，NTP 身份验证功能处于关闭状态。

- (3) 配置 NTP 身份验证密钥。

```
ntp-service authentication-keyid keyid authentication-mode  
{ hmac-sha-1 | hmac-sha-256 | hmac-sha-384 | hmac-sha-512 | md5 } { cipher  
| simple } string [ acl ipv4-acl-number | ipv6 acl ipv6-acl-number ] *
```

缺省情况下，未配置 NTP 身份验证密钥。

- (4) 配置指定密钥为可信密钥。

```
ntp-service reliable authentication-keyid keyid
```

缺省情况下，未指定可信密钥。

- (5) 进入接口视图。

```
interface interface-type interface-number
```

- (6) 将指定密钥与对应的广播服务器关联。

```
ntp-service broadcast-server authentication-keyid keyid
```

缺省情况下，广播服务器没有与密钥关联。

## 1.8.4 配置组播模式的NTP验证功能

### 1. 配置限制和指导

组播服务器和组播客户端上需要配置相同的密钥 ID、验证算法及密钥值，并且保证对端有权在本端使用该密钥 ID 进行验证，否则会导致 NTP 验证失败。

组播客户端和组播服务器上进行不同的配置时，NTP验证结果有所不同，详细介绍请参见 [表 1-5](#)。其中，表格中的“-”表示不管此项是否配置。

表1-5 组播客户端和组播服务器上不同配置时的 NTP 验证结果

组播服务器			组播客户端		结果
身份验证	关联密钥	关联密钥存在且为可信密钥	身份验证	关联密钥存在且为可信密钥	
是	是	是	是	是	身份验证成功
是	是	是	是	否	身份验证失败
是	是	是	否	-	身份验证失败
是	是	否	是	-	身份验证失败
是	是	否	否	-	不进行身份验证
是	否	-	是	-	身份验证失败
是	否	-	否	-	不进行身份验证
否	-	-	是	-	身份验证失败
否	-	-	否	-	不进行身份验证

## 2. 配置组播客户端的NTP验证

- (1) 进入系统视图。

```
system-view
```

- (2) 开启 NTP 身份验证功能。

```
ntp-service authentication enable
```

缺省情况下，NTP 身份验证功能处于关闭状态。

- (3) 配置 NTP 身份验证密钥。

```
ntp-service authentication-keyid keyid authentication-mode  
{ hmac-sha-1 | hmac-sha-256 | hmac-sha-384 | hmac-sha-512 | md5 } { cipher  
| simple } string [ acl ipv4-acl-number | ipv6 acl ipv6-acl-number ] *
```

缺省情况下，未配置 NTP 身份验证密钥。

- (4) 配置指定密钥为可信密钥。

```
ntp-service reliable authentication-keyid keyid
```

缺省情况下，未指定可信密钥。

## 3. 配置组播服务器端的NTP验证

- (1) 进入系统视图。

```
system-view
```

- (2) 开启 NTP 身份验证功能。

```
ntp-service authentication enable
```

缺省情况下，NTP 身份验证功能处于关闭状态。

- (3) 配置 NTP 身份验证密钥。

```
ntp-service authentication-keyid keyid authentication-mode  
{ hmac-sha-1 | hmac-sha-256 | hmac-sha-384 | hmac-sha-512 | md5 } { cipher  
| simple } string [ acl ipv4-acl-number | ipv6 acl ipv6-acl-number ] *
```

缺省情况下，未配置 NTP 身份验证密钥。

- (4) 配置指定密钥为可信密钥。

```
ntp-service reliable authentication-keyid keyid
```

缺省情况下，未指定可信密钥。

- (5) 进入接口视图。

```
interface interface-type interface-number
```

- (6) 将指定密钥与对应的组播服务器关联。

(IPv4 网络)

```
ntp-service multicast-server [ ip-address ] authentication-keyid keyid
```

(IPv6 网络)

```
ntp-service ipv6 multicast-server ipv6-address authentication-keyid  
keyid
```

缺省情况下，组播服务器没有与密钥关联。

## 1.9 控制NTP报文的收发

### 1.9.1 配置NTP报文的源接口

#### 1. 配置限制和指导

- 如果指定了 NTP 报文的源接口，则设备在主动发送 NTP 报文时，NTP 报文的源地址为指定的源接口的地址。建议将 Loopback 接口指定为源接口，以避免设备上某个接口的状态变化而导致 NTP 报文无法接收。
- 设备对接收到的 NTP 请求报文进行应答时，应答报文的源地址始终为接收到的 NTP 请求报文的目地址。
- 如果在命令 **ntp-service unicast-server**/**ntp-service ipv6 unicast-server** 或 **ntp-service unicast-peer**/**ntp-service ipv6 unicast-peer** 中指定了 NTP 报文的源接口，则优先使用 **ntp-service unicast-server**/**ntp-service ipv6 unicast-server** 或 **ntp-service unicast-peer**/**ntp-service ipv6 unicast-peer** 命令指定的接口作为 NTP 报文的源接口。
- 如果在接口视图下配置了 **ntp-service broadcast-server** 或 **ntp-service multicast-server**/**ntp-service ipv6 multicast-server**，则 NTP 广播或组播模式报文的源接口为配置了 **ntp-service broadcast-server** 或 **ntp-service multicast-server**/**ntp-service ipv6 multicast-server** 命令的接口。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置 NTP 报文的源接口。

(IPv4 网络)

```
ntp-service source interface-type interface-number
```

(IPv6 网络)

```
ntp-service ipv6 source interface-type interface-number
```

缺省情况下，未指定 NTP 报文的源接口。

## 1.9.2 关闭接口接收NTP报文功能

### 1. 功能简介

启动 NTP 服务后，缺省情况下所有接口都可以接收 NTP 报文。如果出于安全性、简化网络管理等方面的考虑，不希望设备为某个接口对应网段内的对端设备提供时间同步，或不希望设备从某个接口对应网段内的对端设备获得时间同步，则可以在该接口上执行本配置，使该接口关闭接收 NTP 报文功能。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入接口视图。

```
interface interface-type interface-number
```

- (3) 关闭接口接收 NTP 报文功能。

(IPv4 网络)

```
undo ntp-service inbound enable
```

(IPv6 网络)

```
undo ntp-service ipv6 inbound enable
```

缺省情况下，接口接收 NTP 报文。

## 1.9.3 配置动态会话的最大数目

### 1. 功能简介

本功能用来限制动态会话的数目，以避免设备上维护过多的动态会话，占用过多的系统资源。

NTP 会话分为两种：

- 静态会话：用户手动配置 NTP 相关命令而建立的会话。
- 动态会话：NTP 协议运行过程中建立的临时会话，若系统长期（大约 12 分钟）没有报文交互就会删除该临时会话。

各种模式中，会话的建立情况如下：

- 客户端/服务器模式中，在客户端上指定了 NTP 服务器后，客户端上会建立一个静态会话，服务器端在收到报文之后只是被动的响应报文，而不会建立会话（包括静态和动态会话）。
- 对等体模式中，在主动对等体上指定了被动对等体后，主动对等体上会建立静态会话，被动对等体端会建立动态会话。

- 广播模式和组播模式中，在广播/组播服务器端上会建立静态会话，而在广播/组播客户端上会建立动态会话。

## 2. 配置限制和指导

设备同一时间内最多可以建立的会话数目为 128 个，其中包括静态会话数和动态会话数。

## 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置 NTP 动态会话的最大数目。

```
ntp-service max-dynamic-sessions number
```

缺省情况下，NTP 动态会话的最大数目为 100。

## 1.9.4 配置NTP报文的DSCP优先级

### 1. 功能简介

DSCP 优先级用来体现报文自身的优先等级，决定报文传输的优先程度。通过本配置可以指定 NTP/IPv6 NTP 服务器发送的 NTP 报文的 DSCP 优先级。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置 NTP 报文的 DSCP 优先级。

- （IPv4 网络）

```
ntp-service dscp dscp-value
```

缺省情况下，IPv4 NTP 报文的 DSCP 优先级为 48。

- （IPv6 网络）

```
ntp-service ipv6 dscp dscp-value
```

缺省情况下，IPv6 NTP 报文的 DSCP 优先级为 56。

## 1.10 NTP显示和维护

在完成上述配置后，在任意视图下执行 **display** 命令可以显示配置后 NTP 的运行情况，通过查看显示信息验证配置的效果。

表1-6 NTP 显示和维护

操作	命令
显示NTP服务的所有IPv6会话信息	<b>display ntp-service ipv6 sessions</b> [ <b>verbose</b> ]
显示NTP服务的所有IPv4会话信息	<b>display ntp-service sessions</b> [ <b>verbose</b> ]
显示NTP服务的状态信息	<b>display ntp-service status</b>
显示从本地设备回溯到主时间服务器的各个NTP时间服务器的简要信息	<b>display ntp-service trace</b> [ <b>source</b> <i>interface-type interface-number</i> ]

## 1.11 NTP典型配置举例

### 1.11.1 配置NTP客户端/服务器模式

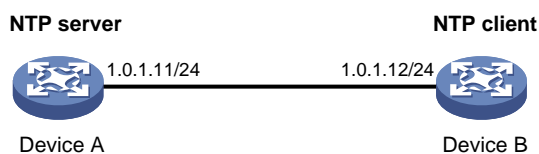
#### 1. 组网需求

为了通过 NTP 实现 Device B 与 Device A 的时间同步，要求：

- 在 Device A 上设置本地时钟作为参考时钟，层数为 2；
- 配置 Device B 工作在客户端模式，指定 Device A 为 NTP 服务器。

#### 2. 组网图

图1-4 配置 NTP 客户端/服务器模式组网图



#### 3. 配置步骤

(1) 按照 图 1-4 配置各接口的IP地址，并确保路由可达，具体配置过程略。

(2) 配置 Device A

# 开启 NTP 服务。

```
<DeviceA> system-view
[DeviceA] ntp-service enable
# 设置本地时钟作为参考时钟，层数为 2。
[DeviceA] ntp-service refclock-master 2
```

(3) 配置 Device B

# 开启 NTP 服务。

```
<DeviceB> system-view
[DeviceB] ntp-service enable
# 配置通过 NTP 协议获取时间。
[DeviceB] clock protocol ntp
# 设置 Device A 为 Device B 的 NTP 服务器。
[DeviceB] ntp-service unicast-server 1.0.1.11
```

#### 4. 验证配置

# 完成上述配置后，Device B 向 Device A 进行时间同步。同步后查看 Device B 的 NTP 状态。可以看出，Device B 已经与 Device A 同步，层数比 Device A 的层数大 1，为 3。

```
[DeviceB] display ntp-service status
Clock status: synchronized
Clock stratum: 3
System peer: 1.0.1.11
Local mode: client
```



```

Reference clock ID: 1.0.1.11
Leap indicator: 00
Clock jitter: 0.000977 s
Stability: 0.000 pps
Clock precision: 2^-19
Root delay: 0.00383 ms
Root dispersion: 16.26572 ms
Reference time: d0c6033f.b9923965 Wed, Dec 29 2010 18:58:07.724
# 查看 Device B 的 NTP 服务的所有 IPv4 会话信息，可以看到 Device B 与 Device A 建立了会话。
[DeviceB] display ntp-service sessions
      source          reference      stra reach poll  now offset  delay disper
*****
[12345]1.0.1.11      127.127.1.0      2      1    64    15    -4.0 0.0038 16.262
Notes: 1 source(master), 2 source(peer), 3 selected, 4 candidate, 5 configured.
Total sessions: 1

```

## 1.11.2 配置IPv6 NTP客户端/服务器模式

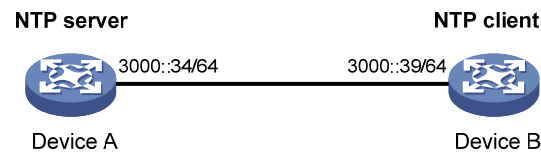
### 1. 组网需求

为了通过 IPv6 NTP 实现 Device B 与 Device A 的时间同步，要求：

- 在 Device A 上设置本地时钟作为参考时钟，层数为 2；
- 配置 Device B 工作在客户端模式，指定 Device A 为 IPv6 NTP 服务器。

### 2. 组网图

图1-5 配置 IPv6 NTP 客户端/服务器模式组网图



### 3. 配置步骤

(1) 按照 图 1-5 配置各接口的IP地址，并确保路由可达，具体配置过程略。

(2) 配置 Device A

# 开启 NTP 服务。

```

<DeviceA> system-view
[DeviceA] ntp-service enable

```

# 设置本地时钟作为参考时钟，层数为 2。

```

[DeviceA] ntp-service refclock-master 2

```

(3) 配置 Device B

# 开启 NTP 服务。

```

<DeviceB> system-view
[DeviceB] ntp-service enable

```

# 配置通过 NTP 协议获取时间。

```

[DeviceB] clock protocol ntp

```

# 设置 Device A 为 Device B 的 IPv6 NTP 服务器。

```
[DeviceB] ntp-service ipv6 unicast-server 3000::34
```

#### 4. 验证配置

# 完成上述配置后，Device B 向 Device A 进行时间同步。同步后查看 Device B 的 NTP 状态。可以看出，Device B 已经与 Device A 同步，层数比 Device A 的层数大 1，为 3。

```
[DeviceB] display ntp-service status
```

```
Clock status: synchronized
```

```
Clock stratum: 3
```

```
System peer: 3000::34
```

```
Local mode: client
```

```
Reference clock ID: 163.29.247.19
```

```
Leap indicator: 00
```

```
Clock jitter: 0.000977 s
```

```
Stability: 0.000 pps
```

```
Clock precision: 2^-19
```

```
Root delay: 0.02649 ms
```

```
Root dispersion: 12.24641 ms
```

```
Reference time: d0c60419.9952fb3e Wed, Dec 29 2010 19:01:45.598
```

# 查看 Device B 的 NTP 服务的所有 IPv6 会话信息，可以看到 Device B 与 Device A 建立了会话。

```
[DeviceB] display ntp-service ipv6 sessions
```

```
Notes: 1 source(master), 2 source(peer), 3 selected, 4 candidate, 5 configured.
```

```
Source: [12345]3000::34
```

```
Reference: 127.127.1.0
```

```
Clock stratum: 2
```

```
Reachabilities: 15
```

```
Poll interval: 64
```

```
Last receive time: 19
```

```
Offset: 0.0
```

```
Roundtrip delay: 0.0
```

```
Dispersion: 0.0
```

```
Total sessions: 1
```

### 1.11.3 配置NTP对等体模式

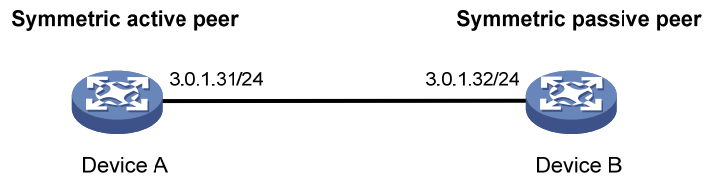
#### 1. 组网需求

网络中存在时间服务器 Device A。为了通过 NTP 实现 Device B 与 Device A 进行时间同步，要求：

- 在 Device A 上设置本地时钟作为参考时钟，层数为 2；
- 配置 Device A 工作在对等体模式，指定 Device B 为被动对等体，即 Device A 为主动对等体，Device B 为被动对等体。

## 2. 组网图

图1-6 配置 NTP 对等体模式组网图



## 3. 配置步骤

(1) 按照 [图 1-6](#) 配置各接口的IP地址，并确保路由可达，具体配置过程略。

(2) 配置 Device B

# 开启 NTP 服务。

```
<DeviceB> system-view
[DeviceB] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[DeviceB] clock protocol ntp
```

(3) 配置 Device A

# 开启 NTP 服务。

```
<DeviceA> system-view
[DeviceA] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[DeviceA] clock protocol ntp
```

# 设置本地时钟作为参考时钟，层数为 2。

```
[DeviceA] ntp-service refclock-master 2
```

# 设置 Device B 为被动对等体。Device A 处于主动对等体模式。

```
[DeviceA] ntp-service unicast-peer 3.0.1.32
```

## 4. 验证配置

# 完成上述配置后，Device B 选择 Device A 作为参考时钟，与 Device A 进行时间同步。同步后查看 Device B 的状态。可以看出，Device B 已经与 Device A 同步，层数比 Device A 的层数大 1，为 3。

```
[DeviceB] display ntp-service status
Clock status: synchronized
Clock stratum: 3
System peer: 3.0.1.31
Local mode: sym_passive
Reference clock ID: 3.0.1.31
Leap indicator: 00
Clock jitter: 0.000916 s
Stability: 0.000 pps
Clock precision: 2^-19
Root delay: 0.00609 ms
Root dispersion: 1.95859 ms
```

Reference time: 83aec681.deb6d3e5 Wed, Jan 8 2014 14:33:11.081

# 查看 Device B 的 NTP 服务的 IPv4 会话信息，可以看到 Device B 与 Device A 建立了会话。

[DeviceB] display ntp-service sessions

source	reference	strata	reach	poll	now	offset	delay	disper
*****								
[12]3.0.1.31	127.127.1.0	2	62	64	34	0.4251	6.0882	1392.1

Notes: 1 source(master), 2 source(peer), 3 selected, 4 candidate, 5 configured.

Total sessions: 1

## 1.11.4 配置IPv6 NTP对等体模式

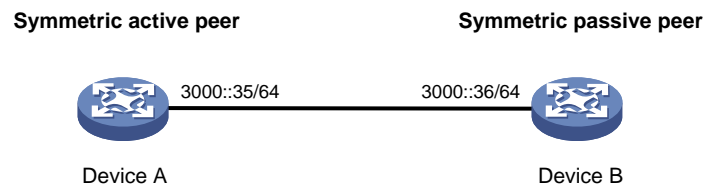
### 1. 组网需求

网络中存在时间服务器 Device A。为了通过 IPv6 NTP 实现 Device B 与 Device A 进行时间同步，要求：

- 在 Device A 上设置本地时钟作为参考时钟，层数为 2；
- 配置 Device A 工作在对等体模式，指定 Device B 为被动对等体，即 Device A 为主动对等体，Device B 为被动对等体。

### 2. 组网图

图1-7 配置 IPv6 NTP 对等体模式组网图



### 3. 配置步骤

(1) 按照 图 1-7 配置各接口的IP地址，并确保路由可达，具体配置过程略。

(2) 配置 Device B

# 开启 NTP 服务。

```
<DeviceB> system-view
[DeviceB] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[DeviceB] clock protocol ntp
```

(3) 配置 Device A

# 开启 NTP 服务。

```
<DeviceA> system-view
[DeviceA] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[DeviceA] clock protocol ntp
```

# 设置本地时钟作为参考时钟，层数为 2。

```
[DeviceA] ntp-service refclock-master 2
```

# 设置 Device B 为 IPv6 被动对等体。Device A 处于主动对等体模式。

```
[DeviceA] ntp-service ipv6 unicast-peer 3000::36
```

#### 4. 验证配置

# 完成上述配置后，Device B 选择 Device A 作为参考时钟，与 Device A 进行时间同步。同步后查看 Device B 的状态。可以看出，Device B 已经与 Device A 同步，层数比 Device A 的层数大 1，为 3。

```
[DeviceB] display ntp-service status
```

```
Clock status: synchronized
```

```
Clock stratum: 3
```

```
System peer: 3000::35
```

```
Local mode: sym_passive
```

```
Reference clock ID: 251.73.79.32
```

```
Leap indicator: 11
```

```
Clock jitter: 0.000977 s
```

```
Stability: 0.000 pps
```

```
Clock precision: 2^-19
```

```
Root delay: 0.01855 ms
```

```
Root dispersion: 9.23483 ms
```

```
Reference time: d0c6047c.97199f9f Wed, Dec 29 2010 19:03:24.590
```

# 查看 Device B 的 NTP 服务的 IPv6 会话信息，可以看到 Device B 与 Device A 建立了会话。

```
[DeviceB] display ntp-service ipv6 sessions
```

```
Notes: 1 source(master), 2 source(peer), 3 selected, 4 candidate, 5 configured.
```

```
Source: [1234]3000::35
```

```
Reference: 127.127.1.0
```

```
Clock stratum: 2
```

```
Reachabilities: 15
```

```
Poll interval: 64
```

```
Last receive time: 19
```

```
Offset: 0.0
```

```
Roundtrip delay: 0.0
```

```
Dispersion: 0.0
```

```
Total sessions: 1
```

### 1.11.5 配置NTP广播模式

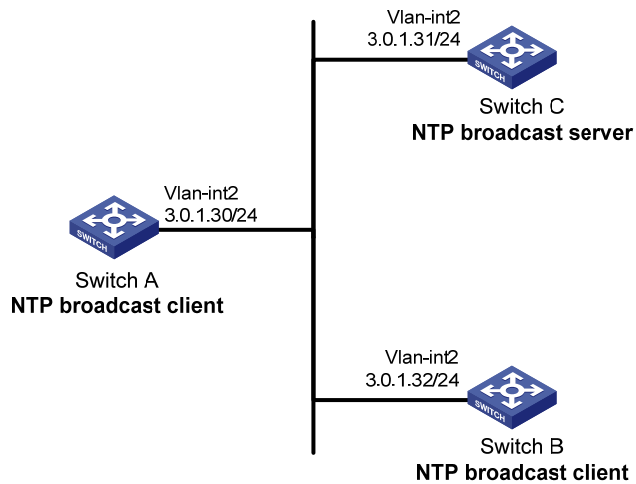
#### 1. 组网需求

为了实现 Switch C 作为同一网段中多个设备的时间服务器，同时同步多个设备的时间，要求：

- 在 Switch C 上设置本地时钟作为参考时钟，层数为 2；
- Switch C 工作在广播服务器模式，从 VLAN 接口 2 向外广播发送 NTP 报文；
- Switch A 和 Switch B 工作在广播客户端模式，分别从各自的 VLAN 接口 2 监听 NTP 广播报文。

## 2. 组网图

图1-8 配置 NTP 广播模式组网图



## 3. 配置步骤

(1) 按照 图 1-8 配置各接口的IP地址，并确保路由可达，具体配置过程略。

(2) 配置 Switch C

# 开启 NTP 服务。

```
<SwitchC> system-view
```

```
[SwitchC] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[SwitchC] clock protocol ntp
```

# 设置本地时钟作为参考时钟，层数为 2。

```
[SwitchC] ntp-service refclock-master 2
```

# 设置 Switch C 为广播服务器，从 VLAN 接口 2 发送广播报文。

```
[SwitchC] interface vlan-interface 2
```

```
[SwitchC-Vlan-interface2] ntp-service broadcast-server
```

(3) 配置 Switch A

# 开启 NTP 服务。

```
<SwitchA> system-view
```

```
[SwitchA] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[SwitchA] clock protocol ntp
```

# 设置 Switch A 为广播客户端，从 VLAN 接口 2 监听广播报文。

```
[SwitchA] interface vlan-interface 2
```

```
[SwitchA-Vlan-interface2] ntp-service broadcast-client
```

(4) 配置 Switch B

# 开启 NTP 服务。

```
<SwitchB> system-view
```

```
[SwitchB] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[SwitchB] clock protocol ntp
# 设置 Switch B 为广播客户端，从 VLAN 接口 2 监听广播报文。
[SwitchB] interface vlan-interface 2
[SwitchB-Vlan-interface2] ntp-service broadcast-client
```

#### 4. 验证配置

# Switch A 和 Switch B 接收到 Switch C 发出的广播报文后，与其同步。以 Switch A 为例，同步后查看 Switch A 的状态。可以看出，Switch A 已经与 Switch C 同步，层数比 Switch C 的层数大 1，为 3。

```
[SwitchA-Vlan-interface2] display ntp-service status
Clock status: synchronized
Clock stratum: 3
System peer: 3.0.1.31
Local mode: bclient
Reference clock ID: 3.0.1.31
Leap indicator: 00
Clock jitter: 0.044281 s
Stability: 0.000 pps
Clock precision: 2^-19
Root delay: 0.00229 ms
Root dispersion: 4.12572 ms
Reference time: d0d289fe.ec43c720 Sat, Jan 8 2011 7:00:14.922
```

# 查看 Switch A 的 NTP 服务的 IPv4 会话信息，可以看到 Switch A 与 Switch C 建立了会话。

```
[SwitchA-Vlan-interface2] display ntp-service sessions
      source          reference      stra reach poll  now offset  delay disper
*****
[1245]3.0.1.31        127.127.1.0        2      1    64  519   -0.0 0.0022 4.1257
Notes: 1 source(master),2 source(peer),3 selected,4 candidate,5 configured.
Total sessions: 1
```

### 1.11.6 配置NTP组播模式

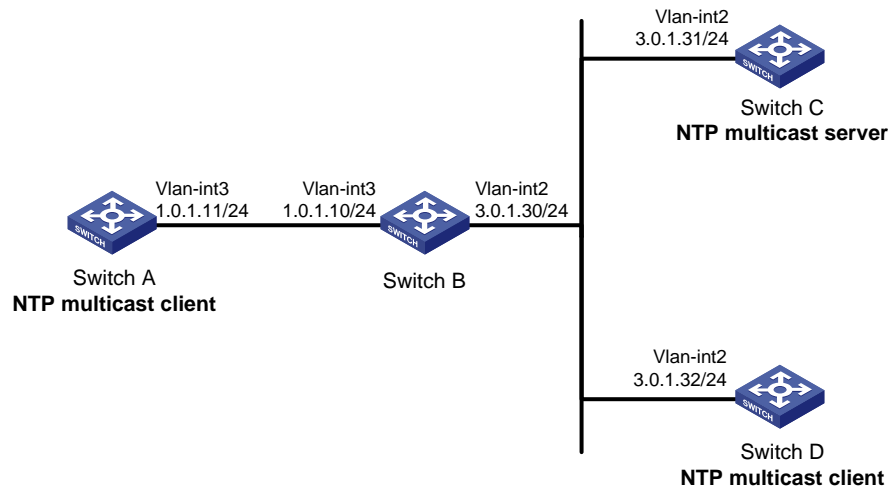
#### 1. 组网需求

为了实现 Switch C 作为不同网段中多个设备的时间服务器，同时同步多个设备的时间，要求：

- 在 Switch C 上设置本地时钟作为参考时钟，层数为 2；
- Switch C 工作在组播服务器模式，从 VLAN 接口 2 向外组播发送 NTP 报文；
- Switch A 和 Switch D 工作在组播客户端模式，Switch A 从 VLAN 接口 3 监听 NTP 组播报文，Switch D 从 VLAN 接口 2 监听 NTP 组播报文。

## 2. 组网图

图1-9 配置 NTP 组播模式组网图



## 3. 配置步骤

(1) 按照 图 1-9 配置各接口的IP地址，并确保路由可达，具体配置过程略。

(2) 配置 Switch C

# 开启 NTP 服务。

```
<SwitchC> system-view
```

```
[SwitchC] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[SwitchC] clock protocol ntp
```

# 设置本地时钟作为参考时钟，层数为 2。

```
[SwitchC] ntp-service refclock-master 2
```

# 设置 Switch C 为组播服务器，从 VLAN 接口 2 发送组播报文。

```
[SwitchC] interface vlan-interface 2
```

```
[SwitchC-Vlan-interface2] ntp-service multicast-server
```

(3) 配置 Switch D

# 开启 NTP 服务。

```
<SwitchD> system-view
```

```
[SwitchD] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[SwitchD] clock protocol ntp
```

# 设置 Switch D 为组播客户端，从 VLAN 接口 2 监听组播报文。

```
[SwitchD] interface vlan-interface 2
```

```
[SwitchD-Vlan-interface2] ntp-service multicast-client
```

(4) 验证配置一

# 由于 Switch D 和 Switch C 在同一个网段，不需要配置组播功能，Switch D 就可以收到 Switch C 发出的组播报文，并与其同步。同步后查看 Switch D 的状态。可以看出，Switch D 已经与 Switch C 同步，层数比 Switch C 的层数大 1，为 3。



```
[SwitchD-Vlan-interface2] display ntp-service status
```

```
Clock status: synchronized
Clock stratum: 3
System peer: 3.0.1.31
Local mode: bclient
Reference clock ID: 3.0.1.31
Leap indicator: 00
Clock jitter: 0.044281 s
Stability: 0.000 pps
Clock precision: 2^-19
Root delay: 0.00229 ms
Root dispersion: 4.12572 ms
Reference time: d0d289fe.ec43c720 Sat, Jan 8 2011 7:00:14.922
```

# 查看 Switch D 的 NTP 服务的所有 IPv4 会话信息，可以看到 Switch D 与 Switch C 建立了会话。

```
[SwitchD-Vlan-interface2] display ntp-service sessions
```

```
          source          reference      stra reach poll  now offset  delay disper
*****
[1245]3.0.1.31          127.127.1.0          2      1  64  519   -0.0 0.0022 4.1257
Notes: 1 source(master),2 source(peer),3 selected,4 candidate,5 configured.
Total sessions: 1
```

#### (5) 配置 Switch B

由于 Switch A 与 Switch C 不在同一网段，所以 Switch B 上需要配置组播功能，否则 Switch A 收不到 Switch C 发出的组播报文。

# 配置组播功能。

```
<SwitchB> system-view
[SwitchB] multicast routing
[SwitchB-mrib] quit
[SwitchB] interface vlan-interface 2
[SwitchB-Vlan-interface2] pim dm
[SwitchB-Vlan-interface2] quit
[SwitchB] vlan 3
[SwitchB-vlan3] port gigabitethernet 1/0/1
[SwitchB-vlan3] quit
[SwitchB] interface vlan-interface 3
[SwitchB-Vlan-interface3] igmp enable
[SwitchB-Vlan-interface3] igmp static-group 224.0.1.1
[SwitchB-Vlan-interface3] quit
[SwitchB] igmp-snooping
[SwitchB-igmp-snooping] quit
[SwitchB] interface gigabitethernet 1/0/1
[SwitchB-GigabitEthernet1/0/1] igmp-snooping static-group 224.0.1.1 vlan 3
```

#### (6) 配置 Switch A

# 开启 NTP 服务。

```
<SwitchA> system-view
[SwitchA] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[SwitchA] clock protocol ntp
```

# 设置 Switch A 为组播客户端，从 VLAN 接口 3 监听组播报文。

```
[SwitchA] interface vlan-interface 3
```

```
[SwitchA-Vlan-interface3] ntp-service multicast-client
```

#### (7) 验证配置二

# 同步后查看 Switch A 的状态。可以看出，Switch A 已经与 Switch C 同步，层数比 Switch C 的层数大 1，为 3。

```
[SwitchA-Vlan-interface3] display ntp-service status
```

```
Clock status: synchronized
```

```
Clock stratum: 3
```

```
System peer: 3.0.1.31
```

```
Local mode: bclient
```

```
Reference clock ID: 3.0.1.31
```

```
Leap indicator: 00
```

```
Clock jitter: 0.165741 s
```

```
Stability: 0.000 pps
```

```
Clock precision: 2^-19
```

```
Root delay: 0.00534 ms
```

```
Root dispersion: 4.51282 ms
```

```
Reference time: d0c61289.10b1193f Wed, Dec 29 2010 20:03:21.065
```

# 查看 Switch A 的 NTP 服务的所有 IPv4 会话信息，可以看到 Switch A 与 Switch C 建立了会话。

```
[SwitchA-Vlan-interface3] display ntp-service sessions
```

source	reference	stra	reach	poll	now	offset	delay	disper
*****								
[1234]3.0.1.31	127.127.1.0	2	247	64	381	-0.0	0.0053	4.5128

```
Notes: 1 source(master),2 source(peer),3 selected,4 candidate,5 configured.
```

```
Total sessions: 1
```

### 1.11.7 配置IPv6 NTP组播模式

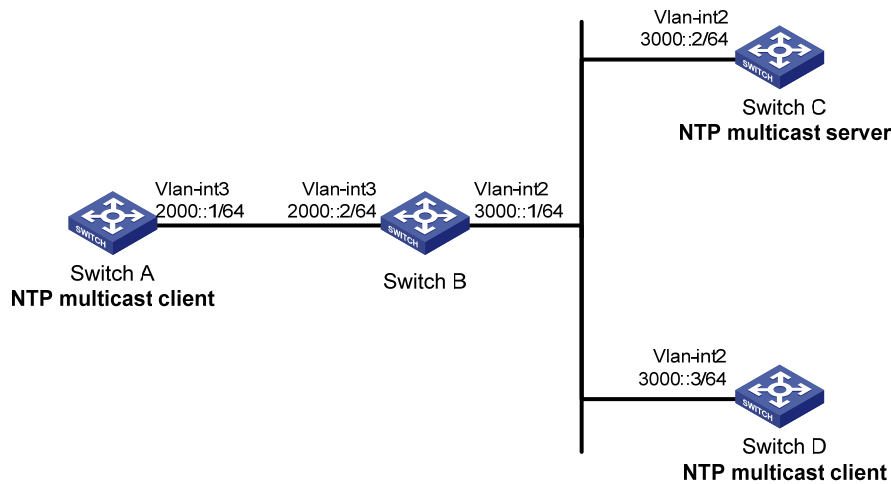
#### 1. 组网需求

为了实现 Switch C 作为不同网段中多个设备的时间服务器，同时同步多个设备的时间，要求：

- 在 Switch C 上设置本地时钟作为参考时钟，层数为 2；
- Switch C 工作在 IPv6 组播服务器模式，从 VLAN 接口 2 向外组播发送 IPv6 NTP 报文；
- Switch A 和 Switch D 工作在 IPv6 组播客户端模式，Switch A 从 VLAN 接口 3 监听 IPv6 NTP 组播报文，Switch D 从 VLAN 接口 2 监听 IPv6 NTP 组播报文。

## 2. 组网图

图1-10 配置 IPv6 NTP 组播模式组网图



## 3. 配置步骤

(1) 按照 [图 1-10](#) 配置各接口的IP地址，并确保路由可达，具体配置过程略。

(2) 配置 Switch C

# 开启 NTP 服务。

```
<SwitchC> system-view
```

```
[SwitchC] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[SwitchC] clock protocol ntp
```

# 设置本地时钟作为参考时钟，层数为 2。

```
[SwitchC] ntp-service refclock-master 2
```

# 设置 Switch C 为 IPv6 组播服务器，从 VLAN 接口 2 向组播地址 FF24::1 发送 NTP 报文。

```
[SwitchC] interface vlan-interface 2
```

```
[SwitchC-Vlan-interface2] ntp-service ipv6 multicast-server ff24::1
```

(3) 配置 Switch D

# 开启 NTP 服务。

```
<SwitchD> system-view
```

```
[SwitchD] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[SwitchD] clock protocol ntp
```

# 设置 Switch D 为 IPv6 组播客户端，在 VLAN 接口 2 监听目的地址为 FF24::1 的 NTP 组播报文。

```
[SwitchD] interface vlan-interface 2
```

```
[SwitchD-Vlan-interface2] ntp-service ipv6 multicast-client ff24::1
```

(4) 验证配置一

# 由于 Switch D 和 Switch C 在同一个网段，不需要配置 IPv6 组播功能，Switch D 就可以收到 Switch C 发出的 IPv6 组播报文，并与其同步。同步后查看 Switch D 的状态。Switch D 已经与 Switch C 同步，层数比 Switch C 的层数大 1，为 3。

```
[SwitchD-Vlan-interface2] display ntp-service status
Clock status: synchronized
Clock stratum: 3
System peer: 3000::2
Local mode: bclient
Reference clock ID: 165.84.121.65
Leap indicator: 00
Clock jitter: 0.000977 s
Stability: 0.000 pps
Clock precision: 2^-19
Root delay: 0.00000 ms
Root dispersion: 8.00578 ms
Reference time: d0c60680.9754fb17 Wed, Dec 29 2010 19:12:00.591
```

# 查看 Switch D 的 NTP 服务的所有 IPv6 会话信息，可以看到 Switch D 与 Switch C 建立了会话。

```
[SwitchD-Vlan-interface2] display ntp-service ipv6 sessions
Notes: 1 source(master), 2 source(peer), 3 selected, 4 candidate, 5 configured.
```

```
Source: [1234]3000::2
Reference: 127.127.1.0          Clock stratum: 2
Reachabilities: 111            Poll interval: 64
Last receive time: 23          Offset: -0.0
Roundtrip delay: 0.0           Dispersion: 0.0

Total sessions: 1
```

## (5) 配置 Switch B

由于 Switch A 与 Switch C 不在同一网段，所以 Switch B 上需要配置 IPv6 组播功能，否则 Switch A 收不到 Switch C 发出的 IPv6 组播报文。

# 配置 IPv6 组播功能。

```
<SwitchB> system-view
[SwitchB] ipv6 multicast routing
[SwitchB-mrib6] quit
[SwitchB] interface vlan-interface 2
[SwitchB-Vlan-interface2] ipv6 pim dm
[SwitchB-Vlan-interface2] quit
[SwitchB] vlan 3
[SwitchB-vlan3] port gigabitethernet 1/0/1
[SwitchB-vlan3] quit
[SwitchB] interface vlan-interface 3
[SwitchB-Vlan-interface3] mld enable
[SwitchB-Vlan-interface3] mld static-group ff24::1
[SwitchB-Vlan-interface3] quit
[SwitchB] mld-snooping
[SwitchB-mld-snooping] quit
```

```
[SwitchB] interface gigabitethernet 1/0/1
[SwitchB-GigabitEthernet1/0/1] mld-snooping static-group ff24::1 vlan 3
```

#### (6) 配置 Switch A

# 开启 NTP 服务。

```
<SwitchA> system-view
[SwitchA] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[SwitchA] clock protocol ntp
```

# 设置 Switch A 为 IPv6 组播客户端，在 VLAN 接口 3 监听目的地址为 FF24::1 的 NTP 组播报文。

```
[SwitchA] interface vlan-interface 3
[SwitchA-Vlan-interface3] ntp-service ipv6 multicast-client ff24::1
```

#### (7) 验证配置二

# 同步后查看 Switch A 的状态。可以看出，Switch A 已经与 Switch C 同步，层数比 Switch C 的层数大 1，为 3。

```
[SwitchA-Vlan-interface3] display ntp-service status
Clock status: synchronized
Clock stratum: 3
System peer: 3000::2
Local mode: bclient
Reference clock ID: 165.84.121.65
Leap indicator: 00
Clock jitter: 0.165741 s
Stability: 0.000 pps
Clock precision: 2^-19
Root delay: 0.00534 ms
Root dispersion: 4.51282 ms
Reference time: d0c61289.10b1193f Wed, Dec 29 2010 20:03:21.065
```

# 查看 Switch A 的 NTP 服务的 IPv6 会话信息，可以看到 Switch A 与 Switch C 建立了会话。

```
[SwitchA-Vlan-interface3] display ntp-service ipv6 sessions
Notes: 1 source(master), 2 source(peer), 3 selected, 4 candidate, 5 configured.
```

```
Source:    [124]3000::2
Reference: 127.127.1.0      Clock stratum: 2
Reachabilities: 2          Poll interval: 64
Last receive time: 71      Offset: -0.0
Roundtrip delay: 0.0       Dispersion: 0.0
```

```
Total sessions: 1
```

### 1.11.8 配置带验证功能的NTP客户端/服务器模式

#### 1. 组网需求

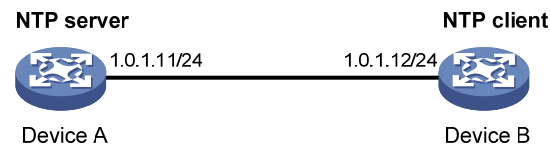
为了通过 NTP 实现 Device B 与 Device A 的时间同步，并保证时间同步的安全性，要求：

- 在 Device A 上设置本地时钟作为参考时钟，层数为 2；

- Device B 工作在客户端模式，指定 Device A 为 NTP 服务器；
- Device A 和 Device B 上同时配置 NTP 验证。

## 2. 组网图

图1-11 配置带身份验证的 NTP 客户端/服务器模式组网图



## 3. 配置步骤

(1) 按照 图 1-11 配置各接口的 IP 地址，并确保路由可达，具体配置过程略。

(2) 配置 Device A 的本地时钟作为参考时钟

# 开启 NTP 服务。

```
<DeviceA> system-view
[DeviceA] ntp-service enable
```

# 设置本地时钟作为参考时钟，层数为 2。

```
[DeviceA] ntp-service refclock-master 2
```

(3) 配置 Device B

# 开启 NTP 服务。

```
<DeviceB> system-view
[DeviceB] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[DeviceB] clock protocol ntp
```

# 在 Device B 上启动 NTP 验证功能。

```
[DeviceB] ntp-service authentication enable
```

# 创建编号为 42 的 NTP 验证密钥，密钥值为 aNiceKey，以明文形式输入。

```
[DeviceB] ntp-service authentication-keyid 42 authentication-mode md5 simple aNiceKey
```

# 配置编号为 42 的密钥为可信密钥。

```
[DeviceB] ntp-service reliable authentication-keyid 42
```

# 设置 Device A 为 Device B 的 NTP 服务器，并将该服务器与编号为 42 的密钥关联。

```
[DeviceB] ntp-service unicast-server 1.0.1.11 authentication-keyid 42
```

以上配置将使得 Device B 与 Device A 进行时间同步，但由于 Device A 没有开启 NTP 身份验证，所以，Device B 还是无法与 Device A 同步。

(4) 在 Device A 上配置 NTP 验证功能。

# 在 Device A 上启动 NTP 验证功能。

```
[DeviceA] ntp-service authentication enable
```

# 创建编号为 42 的 NTP 验证密钥，密钥值为 aNiceKey，以明文形式输入。

```
[DeviceA] ntp-service authentication-keyid 42 authentication-mode md5 simple aNiceKey
```

# 配置编号为 42 的密钥为可信密钥。

```
[DeviceA] ntp-service reliable authentication-keyid 42
```

## 4. 验证配置

# 完成上述配置后，Device B 可以与 Device A 的时间同步。同步后查看 Device B 的状态。可以看出，Device B 已经与 Device A 同步，层数比 Device A 的层数大 1，为 3。

```
[DeviceB] display ntp-service status
Clock status: synchronized
Clock stratum: 3
System peer: 1.0.1.11
Local mode: client
Reference clock ID: 1.0.1.11
Leap indicator: 00
Clock jitter: 0.005096 s
Stability: 0.000 pps
Clock precision: 2^-19
Root delay: 0.00655 ms
Root dispersion: 1.15869 ms
Reference time: d0c62687.ab1bba7d Wed, Dec 29 2010 21:28:39.668
```

# 查看 Device B 的 NTP 服务的所有 IPv4 会话信息，可以看到 Device B 与 Device A 建立了会话。

```
[DeviceB] display ntp-service sessions
      source      reference      stra reach poll  now offset  delay disper
*****
[1245]1.0.1.11      127.127.1.0      2      1      64 519   -0.0 0.0065   0.0
Notes: 1 source(master),2 source(peer),3 selected,4 candidate,5 configured.
Total sessions: 1
```

## 1.11.9 配置带验证功能的NTP广播模式

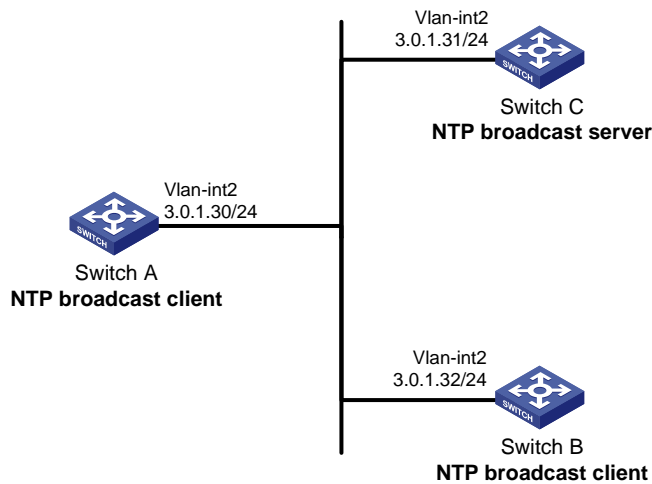
### 1. 组网需求

Switch C 作为同一网段中多个设备的时间服务器，同时同步多个设备的时间。Switch A 和 Switch B 要求对时间服务器进行验证，以保证时间同步的安全性。为了实现上述需求，要求：

- 在 Switch C 上设置本地时钟作为参考时钟，层数为 3；
- Switch C 工作在广播服务器模式，从 VLAN 接口 2 向外广播发送 NTP 报文；
- Switch A 和 Switch B 工作在广播客户端模式，从 VLAN 接口 2 监听 NTP 广播报文；
- 在 Switch A、Switch B 和 Switch C 上配置 NTP 验证功能。

## 2. 组网图

图1-12 配置带身份验证的 NTP 广播模式组网图



## 3. 配置步骤

(1) 按照 [图 1-12](#) 配置各接口的IP地址，并确保路由可达，具体配置过程略。

(2) 配置 Switch A

# 开启 NTP 服务。

```
<SwitchA> system-view
```

```
[SwitchA] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[SwitchA] clock protocol ntp
```

# 开启 NTP 验证功能，创建 ID 为 88 的 NTP 验证密钥，密钥值为 123456，以明文形式输入，并将密钥 88 指定为可信密钥。

```
[SwitchA] ntp-service authentication enable
```

```
[SwitchA] ntp-service authentication-keyid 88 authentication-mode md5 simple 123456
```

```
[SwitchA] ntp-service reliable authentication-keyid 88
```

# 设置 Switch A 为 NTP 广播客户端，从 VLAN 接口 2 监听广播报文。

```
[SwitchA] interface vlan-interface 2
```

```
[SwitchA-Vlan-interface2] ntp-service broadcast-client
```

(3) 配置 Switch B

# 开启 NTP 服务。

```
<SwitchB> system-view
```

```
[SwitchB] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[SwitchB] clock protocol ntp
```

# 开启 NTP 验证功能，创建 ID 为 88 的 NTP 验证密钥，密钥值为 123456，以明文形式输入，并将密钥 88 指定为可信密钥。

```
[SwitchB] ntp-service authentication enable
```

```
[SwitchB] ntp-service authentication-keyid 88 authentication-mode md5 simple 123456
```

```
[SwitchB] ntp-service reliable authentication-keyid 88
```



# 设置 Switch B 为 NTP 广播客户端，从 VLAN 接口 2 监听广播报文。

```
[SwitchB] interface vlan-interface 2
[SwitchB-Vlan-interface2] ntp-service broadcast-client
```

(4) 配置 Switch C 作为 NTP 广播服务器

# 开启 NTP 服务。

```
<SwitchC> system-view
[SwitchC] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[SwitchC] clock protocol ntp
```

# 设置本地时钟作为参考时钟，层数为 3。

```
[SwitchC] ntp-service refclock-master 3
```

# 设置 Switch C 为 NTP 广播服务器，从 VLAN 接口 2 向外发送广播报文。

```
[SwitchC] interface vlan-interface 2
[SwitchC-Vlan-interface2] ntp-service broadcast-server
[SwitchC-Vlan-interface2] quit
```

(5) 验证配置一

# 由于 Switch A 和 Switch B 上开启了 NTP 验证功能，Switch C 上没有开启 NTP 验证功能。因此，Switch A 和 Switch B 无法与 Switch C 的时间同步。以 Switch B 为例，查看 NTP 服务的状态。

```
[SwitchB-Vlan-interface2] display ntp-service status
Clock status: unsynchronized
Clock stratum: 16
Reference clock ID: none
```

(6) 在 Switch C 上配置 NTP 验证功能

# 在 Switch C 上开启 NTP 验证功能，创建 ID 为 88 的 NTP 验证密钥，密钥值为 123456，以明文形式输入，并将密钥 88 指定为可信密钥。

```
[SwitchC] ntp-service authentication enable
[SwitchC] ntp-service authentication-keyid 88 authentication-mode md5 simple 123456
[SwitchC] ntp-service reliable authentication-keyid 88
```

# 设置 Switch C 为 NTP 广播服务器并指定关联的密钥编号为 88。

```
[SwitchC] interface vlan-interface 2
[SwitchC-Vlan-interface2] ntp-service broadcast-server authentication-keyid 88
```

(7) 验证配置二

# 在 Switch C 上开启 NTP 验证功能后，Switch A 和 Switch B 可以与 Switch C 的时间同步。以 Switch B 为例，查看 NTP 服务的状态信息，可以看到 Switch B 已经与 Switch C 同步，层数比 Switch C 的层数大 1，为 4。

```
[SwitchB-Vlan-interface2] display ntp-service status
Clock status: synchronized
Clock stratum: 4
System peer: 3.0.1.31
Local mode: bclient
Reference clock ID: 3.0.1.31
Leap indicator: 00
Clock jitter: 0.006683 s
```

```
Stability: 0.000 pps
Clock precision: 2^-19
Root delay: 0.00127 ms
Root dispersion: 2.89877 ms
Reference time: d0d287a7.3119666f Sat, Jan 8 2011 6:50:15.191
```

# 查看 Switch B 的 NTP 服务的所有 IPv4 会话信息，可以看到 Switch B 与 Switch C 建立了连接。

```
[SwitchB-Vlan-interface2] display ntp-service sessions
      source           reference      stra reach poll  now offset  delay disper
*****
[1245]3.0.1.31         127.127.1.0          3      3   64   68   -0.0 0.0000   0.0
Notes: 1 source(master),2 source(peer),3 selected,4 candidate,5 configured.
Total sessions: 1
```

# 2 SNTP

## 2.1 SNTP简介

NTP 时间同步过程中需要进行复杂的时钟优选运算，时间同步速度较慢，并且占用较多的系统资源。SNTP（Simple NTP，简单 NTP）是由 RFC 4330 定义的客户端版本的简单 NTP，采用与 NTP 相同的报文格式及交互过程，但简化了 NTP 的时间同步过程，以牺牲时间精度为代价实现了时间的快速同步，并减少了占用的系统资源。在时间精度要求不高的情况下，可以使用 SNTP 来实现时间同步。

### 2.1.1 SNTP的工作模式

SNTP 只支持客户端/服务器模式，在模式中提供客户端功能，即作为客户端，从 NTP 服务器获得时间同步，不能作为服务器为其他设备提供时间同步。

如果同时为 SNTP 客户端指定了多个 NTP 服务器，则 SNTP 客户端根据如下方法选择与哪个服务器的时间同步：

- (1) 优先选择时钟层数值最小的 NTP 服务器。
- (2) 如果时钟层数相同，则选择接收到的第一个 NTP 报文对应的 NTP 服务器。

### 2.1.2 协议规范

与 SNTP 相关的协议规范有：

- RFC 4330: Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI

## 2.2 SNTP配置限制和指导

配置 SNTP 时，需要注意：

- 设备上不能同时配置 NTP 和 SNTP 功能。
- 用户需要保证通过 `clock protocol` 命令，配置以 NTP 方式设置系统时间。有关 `clock protocol` 命令的详细介绍，请参见“基础配置命令参考”中的“设备管理”。

## 2.3 SNTP配置任务简介

SNTP 配置任务如下：

- (1) [开启SNTP服务](#)
- (2) [为SNTP客户端指定NTP服务器](#)
- (3) （可选）[配置SNTP验证功能](#)

## 2.4 开启SNTP服务

### 1. 配置限制和指导

NTP 服务与 SNTP 服务互斥，同一时刻只能开启其中一个服务。因此，在开启 SNTP 服务器之前，请确保 NTP 服务关闭。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启 SNTP 服务。

```
sntp enable
```

缺省情况下，SNTP 服务处于关闭状态。

## 2.5 为SNTP客户端指定NTP服务器

### 1. 配置限制和指导

NTP 服务器的时钟只有处于同步状态时，才能作为时间服务器为 SNTP 客户端提供时间同步。当 NTP 服务器的时钟层数大于或等于客户端的时钟层数时，客户端将不会与其同步。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 为设备指定 NTP 服务器。

(IPv4 网络)

```
sntp unicast-server { server-name | ip-address } [ authentication-keyid keyid | source interface-type interface-number | version number ] *
```

缺省情况下，未指定 IPv4 NTP 服务器。

可以通过多次执行本命令配置多个 NTP 服务器。**authentication-keyid** 参数用来将指定密钥与对应的 NTP 服务器关联。使用验证功能时，需要指定本参数。

(IPv6 网络)

```
sntp ipv6 unicast-server { server-name | ipv6-address }  
[ authentication-keyid keyid | source interface-type interface-number ]  
*
```

缺省情况下，未指定 IPv6 NTP 服务器。

可以通过多次执行本命令配置多个 IPv6 NTP 服务器。**authentication-keyid** 参数用来将指定密钥与对应的 NTP 服务器关联。使用验证功能时，需要指定本参数。

## 2.6 配置SNTP验证功能

### 1. 功能简介

在一些对时间同步安全性要求较高的网络中，运行 SNTP 协议时需要启用验证功能。通过客户端和服务端端的身验证，保证客户端只与通过验证的服务器进行时间同步，提高了网络安全性。

2. 配置限制和指导

- 在 NTP 服务器和 SNTP 客户端上都需要开启验证功能。
- NTP服务器和SNTP客户端上必须配置相同的验证密钥（包括密钥ID、验证算法及密钥值），并将密钥设为可信密钥。NTP服务器上验证功能的配置方法，请参见“[1.8.1 配置客户端/服务器模式的NTP验证功能](#)”。
- 在客户端需要将指定密钥与对应的 NTP 服务器关联，并保证服务端有权在本端使用该密钥 ID 进行验证。
- 如果客户端没有成功启用 SNTP 验证功能，不论服务器端是否开启验证功能，客户端均可以与服务器端同步。

3. 配置步骤

- (1) 进入系统视图。
- ```
system-view
```
- (2) 开启 SNTP 身份验证功能。
- ```
sntp authentication enable
```
- 缺省情况下，SNTP 身份验证功能处于关闭状态。
- (3) 配置 SNTP 身份验证密钥。
- ```
sntp authentication-keyid keyid authentication-mode { hmac-sha-1 |  
hmac-sha-256 | hmac-sha-384 | hmac-sha-512 | md5 } { cipher | simple }  
string [ acl ipv4-acl-number | ipv6 acl ipv6-acl-number ] *
```
- 缺省情况下，未配置 SNTP 身份验证密钥。
- (4) 配置指定密钥为可信密钥。
- ```
sntp reliable authentication-keyid keyid
```
- 缺省情况下，未指定可信密钥。
- (5) 将指定密钥与对应的 NTP 服务器关联。
- （IPv4 网络）
- ```
sntp unicast-server { server-name | ip-address } authentication-keyid  
keyid
```
- （IPv6 网络）
- ```
sntp ipv6 unicast-server { server-name | ipv6-address }  
authentication-keyid keyid
```
- 缺省情况下，未指定 NTP 服务器。

2.7 SNTP显示和维护

在完成上述配置后，在任意视图下执行 **display** 命令可以显示配置后 SNTP 的运行情况，通过查看显示信息验证配置的效果。

表2-1 SNTP 显示和维护

操作	命令
显示SNTP服务维护的IPv6会话信息	<b>display sntp ipv6 sessions</b>

操作	命令
显示SNTP服务维护的IPv4会话信息	<code>display sntp sessions</code>

## 2.8 SNTP典型配置举例

### 2.8.1 SNTP基本组网配置举例

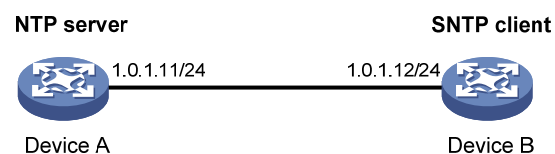
#### 1. 组网需求

Device B 对时间精度要求不高。为了实现 Device B 与 Device A 的时间同步，要求：

- 在 Device A 上设置本地时钟作为参考时钟，层数为 2；
- Device B 工作在 SNTP 客户端模式，指定 Device A 为 NTP 服务器。
- Device B 要求对 NTP 服务器进行验证，以保证时间同步的安全性。

#### 2. 组网图

图2-1 SNTP 配置组网图



#### 3. 配置步骤

(1) 按照 [图 2-1](#) 配置各接口的IP地址，并确保路由可达，具体配置过程略。

(2) 配置 Device A

# 开启 NTP 服务。

```
<DeviceA> system-view
[DeviceA] ntp-service enable
```

# 配置通过 NTP 协议获取时间。

```
[DeviceA] clock protocol ntp
```

# 设置本地时钟作为参考时钟，层数为 2。

```
[DeviceA] ntp-service refclock-master 2
```

# 在 Device A 上启动 NTP 验证功能。

```
[DeviceA] ntp-service authentication enable
```

# 创建编号为 10 的 NTP 验证密钥，密钥值为 aNiceKey，以明文形式输入。

```
[DeviceA] ntp-service authentication-keyid 10 authentication-mode md5 simple aNiceKey
```

# 设置编号为 10 的密钥为可信密钥。

```
[DeviceA] ntp-service reliable authentication-keyid 10
```

(3) 配置 Device B

# 开启 SNTP 服务。

```
<DeviceB> system-view
[DeviceB] sntp enable
```

# 配置通过 NTP 协议获取时间。

```
[DeviceB] clock protocol ntp
```

# 在 Device B 上启动 SNTP 验证功能。

```
[DeviceB] sntp authentication enable
```

# 创建编号为 10 的 SNTP 验证密钥，密钥值为 aNiceKey，以明文形式输入。

```
[DeviceB] sntp authentication-keyid 10 authentication-mode md5 simple aNiceKey
```

# 设置编号为 10 的密钥为可信密钥。

```
[DeviceB] sntp reliable authentication-keyid 10
```

# 设置 Device A 为 Device B 的 NTP 服务器，并将该服务器与编号为 10 的密钥关联。

```
[DeviceB] sntp unicast-server 1.0.1.11 authentication-keyid 10
```

#### 4. 验证配置

# 查看 Device B 的 SNTP 会话信息，可以看到 Device B 与 Device A 建立了会话，并且处于已同步状态。

```
[DeviceB] display sntp sessions
```

SNTP server	Stratum	Version	Last receive time
1.0.1.11	2	4	Tue, May 17 2011 9:11:20.833 (Synced)

# 目 录

1 PoE .....	1-1
1.1 PoE简介 .....	1-1
1.1.1 PoE系统组成 .....	1-1
1.1.2 协议规范 .....	1-1
1.2 PoE配置限制和指导 .....	1-2
1.3 PoE配置任务简介 .....	1-2
1.4 PoE配置准备 .....	1-2
1.5 开启PoE功能 .....	1-2
1.6 开启设备热重启后主动复位PoE端口供电功能 .....	1-4
1.7 检测PD .....	1-4
1.7.1 开启非标准PD检测功能 .....	1-4
1.7.2 配置对PD支持标准的检测方式 .....	1-4
1.8 配置PoE功率 .....	1-5
1.9 配置PoE优先级策略 .....	1-5
1.10 配置PoE监控 .....	1-6
1.11 通过PoE Profile配置PoE接口 .....	1-6
1.11.1 配置限制和指导 .....	1-6
1.11.2 定义PoE Profile .....	1-7
1.11.3 应用PoE Profile .....	1-7
1.12 在线升级PSE固件 .....	1-8
1.13 PoE显示和维护 .....	1-8
1.14 PoE典型配置举例 .....	1-9
1.14.1 PoE基本组网配置举例 .....	1-9
1.15 常见配置错误举例 .....	1-10
1.15.1 配置PoE接口优先级为critical不成功 .....	1-10
1.15.2 应用PoE Profile到PoE接口不成功 .....	1-10



# 1 PoE

## 1.1 PoE简介

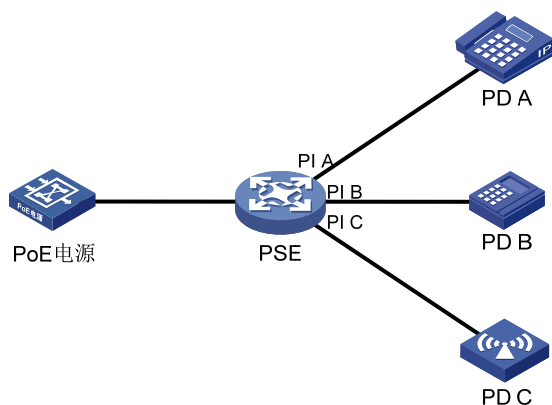
PoE (Power over Ethernet, 以太网供电), 又称远程供电, 是指设备通过以太网电口, 利用双绞线对外接 PD (Powered Device, 受电设备) 进行远程供电。

### 1.1.1 PoE 系统组成

PoE系统如 [图 1-1](#) 所示, 包括PoE电源、PSE (Power Sourcing Equipment, 供电设备)、PI (Power Interface, 电源接口) 和PD。

- PoE 电源: 为整个 PoE 系统供电。
- PSE: 直接给 PD 供电的设备。根据设备中携带的 PSE 的数量可以分为单 PSE 设备和多 PSE 设备。
  - 单 PSE 设备: 设备中只携带了一块 PSE, 通常将整个设备视为一个 PSE。
  - 多 PSE 设备: 设备中携带了多块 PSE, 系统使用 PSE ID 来识别不同 PSE, 使用 **display poe device** 命令可以查看 PSE ID 和接口板槽位号的对应关系。
- PI: 具备 PoE 供电能力的以太网接口, 也称为 PoE 接口。
- PD: 接受 PSE 供电的设备, 如 IP 电话、无线 AP (Access Point, 接入点)、便携设备充电器、刷卡机、网络摄像头等。PD 设备在接受 PoE 电源供电的同时, 可以连接其他电源, 进行电源冗余备份。

图1-1 PoE 系统示意图



### 1.1.2 协议规范

与 PoE 相关的协议规范有:

- 802.3af-2003 - IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD)

Access Method and Physical Layer Specifications - Data Terminal Equipment (DTE) Power Via Media Dependent Interface (MDI)

- 802.3at-2009 - IEEE Standard for Information technology-- Local and metropolitan area networks-- Specific requirements-- Part 3: CSMA/CD Access Method and Physical Layer Specifications Amendment 3: Data Terminal Equipment (DTE) Power via the Media Dependent Interface (MDI) Enhancements

## 1.2 PoE配置限制和指导

开启 PoE 接口远程供电功能，以及 PoE 接口供电优先级、最大功率、远程供电模式的配置有两种方式：

- 直接在接口上配置。
- 基于 Profile 配置。将 PoE Profile 应用到多个 PoE 接口，这些接口具有相同的 PoE 特性；如果 PD 的接入接口变更，则把原接口上应用的 PoE Profile 应用到新的接入接口即可，不再需要重新逐条配置，从而方便了网管人员对 PoE 特性的配置。

对于同一 PoE 接口下的同一 PoE 配置参数，只能选择一种方式进行配置，先配置的生效。如果对命令行已经配置过的参数再通过 PoE Profile 配置，则 PoE Profile 会应用失败；反之亦然。

## 1.3 PoE配置任务简介

PoE 配置任务如下：

- (1) [开启PoE功能](#)
- (2) （可选）[开启设备热重启后主动复位PoE端口供电功能](#)
- (3) （可选）[检测PD](#)
- (4) （可选）[配置PoE功率](#)
- (5) （可选）[配置PoE优先级策略](#)
- (6) （可选）[配置PoE监控](#)
- (7) （可选）[在线升级PSE固件](#)

如需通过配置Profile开启PoE接口远程供电功能，以及PoE接口供电优先级、最大功率，请参见“[通过PoE Profile配置PoE接口](#)”。

## 1.4 PoE配置准备

在配置 PoE 功能前，请确保 PoE 电源和 PSE 已经处于正常工作状态，否则，可能无法进行 PoE 配置或者配置的 PoE 功能不能生效。

## 1.5 开启PoE功能

### 1. 功能简介

开启 PoE 接口的 PoE 功能后，系统即可为 PD 预留功率，并为 PoE 接口下挂的 PD 供电。

开启 PoE 接口的 PoE 功能时：

- 如果该 PoE 接口的加入不会导致 PSE 功率过载（当 PSE 上已经获得供电的端口的最大功率之和大于 PSE 最大功率时，系统将认为 PSE 功率过载，则允许该 PoE 接口为下挂的 PD 供电。
- 如果该 PoE 接口的加入会导致 PSE 功率过载，则由该 PoE 接口是否开启 PoE 接口优先级策略决定（PoE 接口优先级策略的详细介绍请参见“[1.9 配置 PoE 优先级策略](#)”）：
  - 如果该 PoE 接口没有开启 PoE 接口优先级策略，则不允许该 PoE 接口为下挂的 PD 供电；
  - 如果该 PoE 接口已开启 PoE 接口优先级策略，PD 能否获得供电由 PoE 接口的优先级决定。

PoE 接口远程供电有两种模式：

- 信号线供电模式：PSE 使用 3/5 类双绞线中传输数据所用的线对（1、2、3、6）向 PD 传输数据的同时传输直流电。
- 空闲线供电模式：PSE 使用 3/5 类双绞线中没有被使用的线对（4、5、7、8）向 PD 来传输直流电。

## 2. 配置限制和指导

本系列设备 PoE 接口仅支持信号线供电模式。

PSE 和 PD 必须使用相同的模式才能正常供电。如果 PSE 和 PD 支持的供电模式不同（例如 PSE 不支持空闲线供电式，PD 只支持空闲线供电），则需要转接才能给 PD 供电。

## 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 PoE 接口视图。

```
interface interface-type interface-number
```

- (3) 开启 PoE 接口远程供电功能。

```
poe enable
```

缺省情况下：

- 对于 R6126P12 版本，PoE 接口远程供电功能处于关闭状态。
- 对于 R6126P20 及以上版本，设备采用出厂配置启动时，PoE 接口远程供电功能处于开启状态。设备采用空配置启动时，PoE 接口远程供电功能处于关闭状态。

关于空配置启动和缺省配置启动的详细介绍，请参见“基础配置指导”中的“配置文件管理”。

- (4) （可选）配置 PoE 接口连接 PD 的描述信息。

```
poe pd-description text
```

缺省情况下，未配置 PoE 接口连接 PD 的描述信息。

## 1.6 开启设备热重启后主动复位PoE端口供电功能

### 1. 功能简介

设备热重启（通过执行 **reboot** 命令重启）过程中，设备与 PD 的数据连接已中断，但 PoE 端口将继续向 PD 供电。设备重启完成后，PD 可能无法再次主动与设备建立数据连接。配置该功能后，设备热重启时主动对 PoE 端口断电并再次供电，此时 PD 将与设备重新建立数据连接。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启设备热重启后主动复位 PoE 端口供电功能。

```
poe reset enable
```

缺省情况下，设备热重启后主动复位 PoE 端口供电功能处于关闭状态。

## 1.7 检测PD

### 1.7.1 开启非标准 PD 检测功能

#### 1. 功能简介

PD 设备分为标准 PD 和非标准 PD，标准 PD 是指符合 IEEE 802.3af 和 IEEE 802.3at 标准的 PD 设备。PSE 只对标准 PD 供电。只有在开启非标准 PD 检测功能后，PSE 才对非标准 PD 供电。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启非标准 PD 检测功能。

```
poe legacy enable pse pse-id
```

缺省情况下，未开启非标准 PD 检测功能。

### 1.7.2 配置对 PD 支持标准的检测方式

#### 1. 功能简介

设备支持以下对 PD 支持标准的检测方式：

- **none**：不检测，只要与 PD 设备连接无短路和断路，PoE 接口即可对 PD 供电。
- **simple**：简单检测方式，PD 满足 IEEE 802.af 或 802.at 最低要求即可对 PD 供电。
- **strict**：严格检测方式，PD 需要满足 IEEE 802.af 或 802.at 所有要求时，才会对 PD 供电。

如需检测非标准的 PD，请先使用 **poe legacy enable** 命令开启非标准 PD 检测功能。

#### 2. 配置限制和指导

PoE 接口连接非 PD 设备时，请勿配置 **none** 参数，否则，设备会尝试为非 PD 设备供电，可能造成非 PD 设备元器件损坏。

仅 Release 6127 及以上版本支持本功能。

### 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 PoE 接口视图。

```
interface interface-type interface-number
```

- (3) 配置对 PD 支持标准的检测方式。

```
poe detection-mode { none | simple | strict }
```

缺省情况下，对 PD 支持标准的检测方式为 **strict**。

## 1.8 配置PoE功率

### 1. 功能简介

PoE 接口的最大功率是指 PoE 接口能够提供给下挂 PD 的最大功率。当 PD 要求的功率大于 PoE 接口的最大功率时，则不会给 PD 供电。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 PoE 接口视图。

```
interface interface-type interface-number
```

- (3) 配置 PoE 接口的最大供电功率。

```
poe max-power max-power
```

缺省情况下，PoE 接口的最大功率为 30000。

## 1.9 配置PoE优先级策略

### 1. 功能简介

PoE 接口优先级策略用于 PSE 过载时，决定是否给 PD 供电。

PoE 接口的优先级按照高低顺序为：Critical、High 和 Low。

PoE 接口对 PD 供电时：

- 没有开启 PoE 接口优先级策略，PSE 功率过载时：
  - 配置 PSE 最大功率时，如果接入新的 PD 导致 PSE 功率过载，则不对新接入的 PD 供电；如果正在受电的 PD 功率增大导致 PSE 功率过载，则停止对该 PD 供电。
  - 没有配置 PSE 最大功率时，触发 PoE 电源自我保护机制，停止给所有的 PD 供电。
- 开启 PoE 接口优先级策略时：
  - 如果因正在受电的 PD 功率增大导致 PSE 功率过载，则停止对该 PD 供电。
  - 如果接入新的 PD 导致 PSE 功率过载，优先级高的 PoE 接口下挂的 PD 优先得到供电。
  - 如果新接入 PD 对应的 PoE 接口优先级与正在供电的 PoE 接口优先级相同，则正在供电的 PoE 接口优先级高，此时不给新接入的 PD 供电。
  - 如果多个 PoE 接口优先级相同，接口编号小的优先级高，优先为下挂 PD 供电。

## 2. 配置限制和指导

配置 PoE 接口优先级为 **Critical** 时，PSE 最大功率值减该 PSE 中优先级为 **Critical** 的 PoE 接口的最大功率值之差必须大于 PoE 接口最大功率。否则，配置该接口配置 **Critical** 优先级失败。

低优先级 PoE 接口被抢占时，该接口下挂的 PD 断电，PoE 接口的配置不变。

## 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启 PoE 接口优先级策略。

```
poe pd-policy priority
```

缺省情况下，PoE 接口优先级策略处于关闭状态。

- (3) 进入 PoE 接口视图。

```
interface interface-type interface-number
```

- (4) 配置 PoE 接口优先级。

```
poe priority { critical | high | low }
```

缺省情况下，PoE 接口优先级为 **low**。

## 1.10 配置PoE监控

### 1. 功能简介

当 PSE 在当前功率利用率首次超过或低于配置的功率阈值时，系统将生成告警信息，发送给设备的 SNMP 模块，通过设置 SNMP 中告警信息的发送参数，来决定告警信息输出的相关属性。有关告警信息的详细介绍，请参见“网络管理和监控配置指导”中的“SNMP”。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置 PSE 的功率告警阈值。

```
poe utilization-threshold value pse pse-id
```

缺省情况下，PSE 的功率告警阈值为 80%。

## 1.11 通过PoE Profile配置PoE接口

### 1.11.1 配置限制和指导

不能通过重复执行 **apply poe-profile** 或 **apply poe-profile interface** 命令修改 PoE Profile。如需修改 PoE Profile，请先取消 PoE profile 在 PoE 接口的应用，修改 PoE Profile 后，再将 PoE profile 应用到 PoE 接口。

**poe max-power** *max-power* 命令和 **poe priority** { **critical** | **high** | **low** } 必须使用同一种方式配置，即都通过命令行配置或都通过 PoE Profile 配置。

### 1.11.2 定义 PoE Profile

- (1) 进入系统视图。

**system-view**

- (2) 创建 PoE Profile，并进入 PoE Profile 视图。

**poe-profile** *profile-name* [ *index* ]

缺省情况下，未配置 PoE Profile。

- (3) 开启 PoE 接口远程供电功能。

**poe enable**

缺省情况下，PoE 接口远程供电功能处于关闭状态。

- (4) （可选）配置 PoE 接口的最大功率。

**poe max-power** *max-power*

缺省情况下，PoE 接口的最大功率为 30000。

- (5) （可选）配置 PoE 接口供电优先级。

**poe priority** { **critical** | **high** | **low** }

缺省情况下，PoE 接口供电优先级为 **low**。

仅在开启接口功率管理优先级策略的情况下生效。

### 1.11.3 应用 PoE Profile

#### 1. 配置限制和指导

设备支持在系统视图下应用 PoE Profile 和在接口视图下应用 PoE Profile:

- 在系统视图下应用 PoE Profile 时，可以将 PoE Profile 应用到多个 PoE 接口。
- 在接口视图下应用 PoE Profile 时，只能将 PoE Profile 应用到当前 PoE 接口。

对于同一个 PoE 接口，通过系统视图应用 PoE Profile 和通过接口视图应用 PoE Profile 配置效果相同，最后一次执行的命令生效。

一个 PoE Profile 可以应用于多个 PoE 接口。

一个 PoE 接口下只能应用一个 PoE Profile。

#### 2. 通过系统视图应用 PoE Profile

- (1) 进入系统视图。

**system-view**

- (2) 将 PoE Profile 应用到 PoE 接口。

**apply poe-profile** { **index** *index* | **name** *profile-name* } **interface** *interface-range*

缺省情况下，未将 PoE profile 应用到 PoE 接口。

#### 3. 通过接口视图应用 PoE Profile

- (1) 进入系统视图。

**system-view**

- (2) 进入 PoE 接口视图。

```
interface interface-type interface-number
```

- (3) 将 PoE Profile 应用到当前 PoE 接口。

```
apply poe-profile { index index | name profile-name }
```

缺省情况下，未将 PoE profile 应用到当前 PoE 接口。

## 1.12 在线升级PSE固件

### 1. 功能简介

用户可以通过以下操作在线升级 PSE 固件，在线升级指的是不用重启 PSE 就能完成升级。升级有两种模式：

- **refresh** 模式：在原 PSE 固件的基础上进行更新升级，升级过程简单快速。一般情况下使用该模式来升级 PSE 固件。
- **full** 模式：将原 PSE 固件删除，再安装新 PSE 固件。PSE 固件被损坏的情况下（表现为所有的 PoE 命令执行不成功），可用 **full** 模式进行升级，使软件恢复。

### 2. 配置限制和指导

如果 PSE 固件的升级过程因设备重启而中断，重启完成后用 **full** 方式升级失败时，请将设备断电重启后再用 **full** 方式升级即可升级成功。

### 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 在线升级 PSE 固件。

```
poe update { full | refresh } filename [ pse pse-id ]
```

## 1.13 PoE显示和维护

在完成上述配置后，在任意视图下执行 **display** 命令可以显示配置后 PoE 的运行情况，通过查看显示信息验证配置的效果。

表1-1 PoE 显示和维护

操作	命令
显示PSE的相关信息	<b>display poe device</b> [ slot slot-number ]
显示设备指定PoE接口的供电状态	<b>display poe interface</b> [ interface-type interface-number ]
显示PoE接口的功率信息	<b>display poe interface power</b> [ interface-type interface-number ]
显示PSE的信息	<b>display poe pse</b> [ pse-id ]
显示指定PSE上PoE接口的供电状态	<b>display poe pse pse-id interface</b>
显示指定PSE上PoE接口的功率信息	<b>display poe pse pse-id interface power</b>
显示PoE Profile的相关信息	<b>display poe-profile</b> [ index index   name profile-name ]



操作	命令
显示指定PoE接口当前生效的PoE Profile配置项和应用的信息	<b>display poe-profile interface</b> <i>interface-type interface-number</i>

## 1.14 PoE典型配置举例

### 1.14.1 PoE 基本组网配置举例

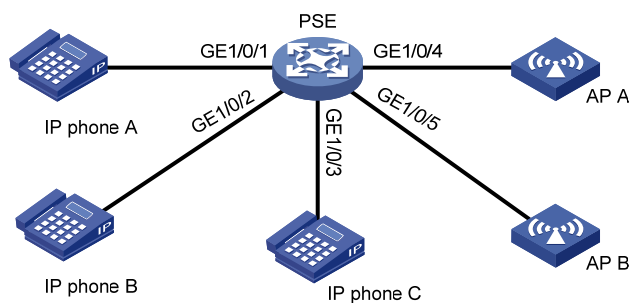
#### 1. 组网需求

设备通过 PoE 接口为 PD 设备供电。

- GigabitEthernet1/0/1、GigabitEthernet1/0/2、GigabitEthernet1/0/3 接入 IP 电话。
- GigabitEthernet1/0/4、GigabitEthernet1/0/5 接入 AP 设备。
- IP 电话的供电优先级高于 AP 设备。当 PSE 功率过载时，优先给 IP 电话供电。
- GigabitEthernet1/0/5 下接的 AP B 的功率最大不能超过 9000 毫瓦。

#### 2. 组网图

图1-2 PoE 组网图



#### 3. 配置步骤

# 开启 PoE 接口优先级策略。

```
<PSE> system-view
[PSE] poe pd-policy priority
```

# 开启 PoE 接口 GigabitEthernet1/0/1、GigabitEthernet1/0/2、GigabitEthernet1/0/3 的远程供电功能，并且供电优先级为 critical。

```
[PSE] interface gigabitethernet 1/0/1
[PSE-GigabitEthernet1/0/1] poe enable
[PSE-GigabitEthernet1/0/1] poe priority critical
[PSE-GigabitEthernet1/0/1] quit
[PSE] interface gigabitethernet 1/0/2
[PSE-GigabitEthernet1/0/2] poe enable
[PSE-GigabitEthernet1/0/2] poe priority critical
[PSE-GigabitEthernet1/0/2] quit
[PSE] interface gigabitethernet 1/0/3
[PSE-GigabitEthernet1/0/3] poe enable
```

```
[PSE-GigabitEthernet1/0/3] poe priority critical
[PSE-GigabitEthernet1/0/3] quit
```

# 开启 PoE 接口 GigabitEthernet1/0/4 和 GigabitEthernet1/0/5 的远程供电功能，并配置 PoE 接口 GigabitEthernet1/0/5 的最大供电功率为 9000 毫瓦。

```
[PSE] interface gigabitethernet 1/0/4
[PSE-GigabitEthernet1/0/4] poe enable
[PSE-GigabitEthernet1/0/4] quit
[PSE] interface gigabitethernet 1/0/5
[PSE-GigabitEthernet1/0/5] poe enable
[PSE-GigabitEthernet1/0/5] poe max-power 9000
```

#### 4. 结果验证

配置完成后，IP 电话和 AP 设备被供电，能够正常工作。

## 1.15 常见配置错误举例

### 1.15.1 配置 PoE 接口优先级为 critical 不成功

#### 1. 原因分析

PoE 接口所在 PSE 剩余保证功率小于 PoE 接口的最大供电功率。

PoE 接口的优先级已经通过其他方式进行配置。

#### 2. 解决方法

对于第一种情况可以通过增大 PSE 的最大供电功率来解决，或者在 PSE 剩余保证功率不可调整时减小 PoE 接口的最大供电功率。

对于第二种情况需先取消其他方式的配置。

### 1.15.2 应用 PoE Profile 到 PoE 接口不成功

#### 1. 原因分析

该 PoE Profile 的某些配置项已经通过其他方式进行配置。

该 PoE Profile 的某些配置项不符合 PoE 接口的配置要求。

已经存在 PoE Profile 在该 PoE 接口的应用。

#### 2. 解决方法

对于第一种情况，可以通过取消其他方式的配置来解决。

对于第二种情况，需修改该 PoE Profile 的某些配置项。

对于第三种情况，先取消其他 PoE Profile 在该 PoE 接口的应用。

# 目 录

1 SNMP .....	1-1
1.1 SNMP简介 .....	1-1
1.1.1 SNMP的网络架构 .....	1-1
1.1.2 MIB和MIB视图 .....	1-1
1.1.3 SNMP基本操作 .....	1-2
1.1.4 SNMP版本介绍 .....	1-2
1.1.5 SNMP支持的访问控制方式 .....	1-2
1.2 FIPS相关说明 .....	1-3
1.3 配置任务简介 .....	1-3
1.4 开启SNMP功能 .....	1-3
1.5 配置SNMP版本 .....	1-4
1.6 配置SNMPv1/v2c/v3 版本公共参数 .....	1-4
1.7 配置SNMPv1/v2c团体 .....	1-6
1.7.1 功能简介 .....	1-6
1.7.2 配置限制和指导 .....	1-6
1.7.3 基于名称配置SNMPv1/v2c团体 .....	1-6
1.7.4 基于用户配置SNMPv1/v2c团体 .....	1-6
1.8 配置SNMPv3 组和用户 .....	1-7
1.8.1 配置限制和指导 .....	1-7
1.8.2 非FIPS模式下配置SNMPv3 组和用户 .....	1-7
1.8.3 FIPS模式下配置SNMPv3 组和用户 .....	1-8
1.9 配置SNMP告警 .....	1-9
1.9.1 功能简介 .....	1-9
1.9.2 开启告警功能 .....	1-9
1.9.3 配置告警信息发送参数 .....	1-10
1.10 配置SNMP日志 .....	1-11
1.11 SNMP显示和维护 .....	1-12
1.12 SNMP典型配置举例 .....	1-13
1.12.1 SNMPv1/v2c配置举例 .....	1-13
1.12.2 SNMPv3 配置举例 .....	1-14

# 1 SNMP

## 1.1 SNMP简介

SNMP（Simple Network Management Protocol，简单网络管理协议）广泛用于网络设备的远程管理和操作。SNMP 允许管理员通过 NMS 对网络上不同厂商、不同物理特性、采用不同互联技术的设备进行管理，包括状态监控、数据采集和故障处理。

### 1.1.1 SNMP的网络架构

SNMP 网络架构由三部分组成：NMS、Agent 和 MIB。

- NMS（Network Management System，网络管理系统）是 SNMP 网络的管理者，能够提供友好的人机交互界面，方便网络管理员完成大多数的网络管理工作。
- Agent 是 SNMP 网络的被管理者，负责接收、处理来自 NMS 的 SNMP 报文。在某些情况下，如接口状态发生改变时，Agent 也会主动向 NMS 发送告警信息。
- MIB（Management Information Base，管理信息库）是被管理对象的集合。NMS 管理设备的时候，通常会关注设备的一些参数，比如接口状态、CPU 利用率等，这些参数就是被管理对象，在 MIB 中称为节点。每个 Agent 都有自己的 MIB。MIB 定义了节点之间的层次关系以及对对象的一系列属性，比如对象的名称、访问权限和数据类型等。被管理设备都有自己的 MIB 文件，在 NMS 上编译这些 MIB 文件，就能生成该设备的 MIB。NMS 根据访问权限对 MIB 节点进行读/写操作，从而实现 Agent 的管理。

NMS、Agent和MIB之间的关系如 [图 1-1](#) 所示。

图1-1 NMS、Agent 和 MIB 关系图

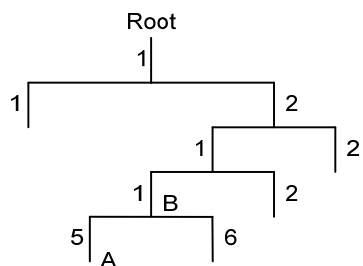


### 1.1.2 MIB和MIB视图

MIB以树状结构进行存储。树的每个节点都是一个被管理对象，它用从根开始的一条路径唯一地识别（OID）。如 [图 1-2](#) 所示，被管理对象B可以用一串数字{1.2.1.1}唯一确定，这串数字是被管理对象的OID（Object Identifier，对象标识符）。

MIB 视图是 MIB 的子集合，将团体名/用户名与 MIB 视图绑定，可以限制 NMS 能够访问的 MIB 对象。当用户配置 MIB 视图包含某个 MIB 子树时，NMS 可以访问该子树的所有节点；当用户配置 MIB 视图不包含某个 MIB 子树时，NMS 不能访问该子树的所有节点。

图1-2 MIB 树结构



### 1.1.3 SNMP基本操作

SNMP 提供四种基本操作：

- **Get 操作：**NMS 使用该操作查询 Agent MIB 中节点的值。
- **Set 操作：**NMS 使用该操作配置 Agent MIB 中节点的值。
- **告警操作：**SNMP 告警包括 Trap 和 Inform 两种。
  - **Trap 操作：**Agent 使用该操作向 NMS 发送 Trap 报文。Agent 不要求 NMS 发送回应报文，NMS 也不会对 Trap 报文进行回应。SNMPv1、SNMPv2c 和 SNMPv3 均支持 Trap 操作。
  - **Inform 操作：**Agent 使用该操作向 NMS 发送 Inform 报文。Agent 要求 NMS 发送回应报文，因此，Inform 报文比 Trap 报文更可靠，但消耗的系统资源更多。如果 Agent 在一定时间内没有收到 NMS 的回应报文，则会启动重发机制。只有 SNMPv2c 和 SNMPv3 支持 Inform 操作。

### 1.1.4 SNMP版本介绍

目前，设备运行于非 FIPS 模式时，支持 SNMPv1、SNMPv2c 和 SNMPv3 三种版本；设备运行于 FIPS 模式时，只支持 SNMPv3 版本。只有 NMS 和 Agent 使用的 SNMP 版本相同时，NMS 才能和 Agent 建立连接。

- **SNMPv1** 采用团体名（Community Name）认证机制。团体名类似于密码，用来限制 NMS 和 Agent 之间的通信。如果 NMS 配置的团体名和被管理设备上配置的团体名不同，则 NMS 和 Agent 不能建立 SNMP 连接，从而导致 NMS 无法访问 Agent，Agent 发送的告警信息也会被 NMS 丢弃。
- **SNMPv2c** 也采用团体名认证机制。SNMPv2c 对 SNMPv1 的功能进行了扩展：提供了更多的操作类型；支持更多的数据类型；提供了更丰富的错误代码，能够更细致地区分错误。
- **SNMPv3** 采用 USM（User-Based Security Model，基于用户的安全模型）认证机制。网络管理员可以配置认证和加密功能。认证用于验证报文发送方的合法性，避免非法用户的访问；加密则是对 NMS 和 Agent 之间的传输报文进行加密，以免被窃听。采用认证和加密功能可以为 NMS 和 Agent 之间的通信提供更高的安全性。

### 1.1.5 SNMP支持的访问控制方式

SNMP 支持的访问控制方式包括：

- VACM（View-based Access Control Model，基于视图的访问控制模型）：将团体名/用户名与指定的 MIB 视图进行绑定，可以限制 NMS 能够访问哪些 MIB 对象，以及对 MIB 对象不同的操作权限。
- RBAC（Role Based Access Control，基于角色的访问控制）：创建团体名/用户名时，可以指定对应的用户角色，通过用户角色下制定的规则，来限制 NMS 能够访问哪些 MIB 对象，以及对 MIB 对象不同的操作权限。
  - 拥有 network-admin 或 level-15 用户角色的 SNMP 团体/用户，可以对所有的 MIB 对象进行读写操作；
  - 拥有 network-operator 用户角色的 SNMP 团体/用户，可以对所有的 MIB 对象进行读操作；
  - 拥有自定义用户角色的 SNMP 团体/用户，可以对角色规则中指定的 MIB 对象进行操作。

对于同一 SNMP 用户名/团体名，只能配置一种控制方式，多次使用两种控制方式配置同一用户名/团体名时，以最后一次的配置方式为准。

RBAC 配置方式限制的是 MIB 节点的读写权限，VACM 配置方式限制的是 MIB 视图的读写权限，而一个视图中通常包括多个 MIB 节点。所以，RBAC 配置方式更精准、更灵活。关于 RBAC 的详细介绍请参见“基础配置”中的“RBAC”。

## 1.2 FIPS相关说明

设备运行于 FIPS 模式时，不支持 SNMPv1/v2c 版本。

设备运行于 FIPS 模式时，本特性部分配置相对于非 FIPS 模式有所变化，具体差异请见本文相关描述。有关 FIPS 模式的详细介绍请参见“安全配置指导”中的“FIPS”。

## 1.3 配置任务简介

SNMP 配置任务如下：

- (1) [开启SNMP功能](#)
- (2) [配置SNMP版本](#)
- (3) 配置 SNMP 基本参数
  - （可选）[配置SNMPv1/v2c/v3 版本公共参数](#)
  - [配置SNMPv1/v2c团体](#)
  - [配置SNMPv3 组和用户](#)
- (4) （可选）[配置SNMP告警](#)
- (5) （可选）[配置SNMP日志](#)

## 1.4 开启SNMP功能

### 1. 配置限制和指导

执行除 `snmp-agent calculate-password` 外任何以 `snmp-agent` 开头的命令，都可以开启 SNMP 功能。

## 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 启动 SNMP 功能。

```
snmp-agent
```

缺省情况下，SNMP 功能处于关闭状态。

## 1.5 配置SNMP版本

### 1. 配置限制和指导

设备运行于非 FIPS 模式时，支持 SNMPv1、SNMPv2c 和 SNMPv3 三种版本；设备运行于 FIPS 模式时，只支持 SNMPv3 版本。只有 NMS 和设备使用的 SNMP 版本相同时，NMS 才能和 Agent 建立连接。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置设备支持的 SNMP 版本。

（非 FIPS 模式）

```
snmp-agent sys-info version { all | { v1 | v2c | v3 } * }
```

（FIPS 模式）

```
snmp-agent sys-info version { all | v3 }
```

缺省情况下，启用 SNMPv3 版本。

多次执行该命令配置不同的版本时，各配置均生效，设备会和 NMS 协商一个版本进行通信

## 1.6 配置SNMPv1/v2c/v3版本公共参数

### 1. 配置限制和指导

设备出厂时，已分配了全网唯一的 SNMP 引擎 ID。并且，SNMPv3 版本的用户名、密文密码等都和引擎 ID 相关联，如果更改了引擎 ID，则原引擎 ID 下配置的用户名、密码失效。所以，通常情况下，请不要修改设备的 SNMP 引擎 ID。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置设备上接收 SNMP 报文的端口号。

```
snmp-agent port port-number
```

缺省情况下，使用 161 号端口接收 SNMP 报文。

- (3) 配置本设备的 SNMP 引擎 ID。

```
snmp-agent local-engineid engineid
```

缺省情况下，设备引擎 ID 为公司的“企业号+设备信息”。每台设备的设备信息不同，请以设备实际情况为准。

- (4) 配置远端 SNMP 实体的引擎 ID。

```
snmp-agent remote { ipv4-address | ipv6 ipv6-address } engineid engineid
```

缺省情况下，未配置远端 SNMP 实体的引擎 ID。

当设备需要向目的主机（能够解析 Trap 和 Inform 报文的设备，通常为 NMS）发送 SNMPv3 Inform 报文时，该步骤必选。

- (5) 创建或者更新 MIB 视图。

```
snmp-agent mib-view { excluded | included } view-name oid-tree [ mask  
mask-value ]
```

缺省情况下，存在四个 MIB 视图，名称均为 ViewDefault。

- 视图一包含 MIB 子树 iso。
- 视图二不包含子树 snmpUsmMIB。
- 视图三不包含子树 snmpVacmMIB。
- 视图四不包含子树 snmpModules.18。

MIB 视图由视图名和 MIB 子树来唯一确定。视图名相同但包含的子树不同，则认为是不同的视图。

- (6) 配置 SNMP 管理信息。

- 配置设备的维护联系信息。

```
snmp-agent sys-info contact sys-contact
```

缺省情况下，设备的维护联系信息为 New H3C Technologies Co., Ltd.

- 配置设备的物理位置信息。

```
snmp-agent sys-info location sys-location
```

缺省情况下，物理位置信息为 Hangzhou, China。

- (7) 创建 SNMP 上下文。

```
snmp-agent context context-name
```

缺省情况下，不存在 SNMP 上下文。

- (8) 配置 Agent 能处理的 SNMP 报文的最大长度。

```
snmp-agent packet max-size byte-count
```

缺省情况下，Agent 能处理的 SNMP 报文的最大长度为 1500。

- (9) 配置 SNMP 响应报文的 DSCP（Differentiated Services Code Point，差分服务代码点）优先级。

```
snmp-agent packet response dscp dscp-value
```

缺省情况下，SNMP 响应报文的 DSCP 优先级为 0。



## 1.7 配置SNMPv1/v2c团体

### 1.7.1 功能简介

用户可以基于名称配置 SNMPv1/v2c 团体，或者基于用户配置 SNMPv1/v2c 团体。两种配置方式，效果相同。基于用户配置指的是先创建 SNMP 组，再向创建的组中添加 SNMPv1/v2c 用户，SNMPv1/v2c 用户相当于 SNMPv1/v2c 的团体名，在 NMS 上配置的团体名需要跟设备上配置的 SNMPv1/v2c 用户名一致。

### 1.7.2 配置限制和指导

为了安全起见，只有具有 network-admin 或者 level-15 用户角色的用户登录设备后才能创建 SNMP 团体、用户或组。其它角色的用户，即使授权了 SNMP 特性或相应命令的操作权限，也不能执行相应命令。

### 1.7.3 基于名称配置SNMPv1/v2c团体

- (1) 进入系统视图。

```
system-view
```

- (2) 直接创建 SNMP 团体。请选择其中一项进行配置。

- o VACM 方式:

```
snmp-agent community { read | write } [ simple | cipher ] community-name  
[ mib-view view-name ] [ acl { ipv4-acl-number | name ipv4-acl-name } |  
acl ipv6 { ipv6-acl-number | name ipv6-acl-name } ] *
```

- o RBAC 方式:

```
snmp-agent community [ simple | cipher ] community-name user-role  
role-name [ acl { ipv4-acl-number | name ipv4-acl-name } | acl ipv6  
{ ipv6-acl-number | name ipv6-acl-name } ] *
```

- (3) (可选) 创建团体名到 SNMP 上下文的映射。

```
snmp-agent community-map community-name context context-name
```

### 1.7.4 基于用户配置SNMPv1/v2c团体

- (1) 进入系统视图。

```
system-view
```

- (2) 通过创建用户来创建 SNMP 团体。

- a. 创建 SNMPv1/v2c 组:

```
snmp-agent group { v1 | v2c } group-name [ read-view view-name ]  
[ write-view view-name ] [ notify-view view-name ] [ acl  
{ ipv4-acl-number | name ipv4-acl-name } | acl ipv6 { ipv6-acl-number  
| name ipv6-acl-name } ] *
```

- b. 创建 SNMPv1/v2c 用户:

```
snmp-agent usm-user { v1 | v2c } user-name group-name [ acl
{ ipv4-acl-number | name ipv4-acl-name } | acl ipv6 { ipv6-acl-number
| name ipv6-acl-name } ] *
```

- (3) (可选) 创建团体名到 SNMP 上下文的映射。

```
snmp-agent community-map community-name context context-name
```

## 1.8 配置SNMPv3组和用户

### 1.8.1 配置限制和指导

为了安全起见, 只有具有 **network-admin** 或者 **level-15** 用户角色的用户登录设备后才能创建 **SNMP** 用户或组。其它角色的用户, 即使授权了 **SNMP** 特性或相应命令的操作权限, 也不能执行相应命令。**SNMPv3** 使用组来管理用户。**NMS**使用**SNMPv3** 用户名访问设备时, 是否需要认证和加密, 由组的配置决定, 创建用户时, 可以为不同用户配置不同的算法和认证密码、加密密码, 基本配置要求见 [表 1-1](#)。

表1-1 不同安全模式的基本配置要求

安全模型	创建组时必须配置的相关参数	创建用户时必须配置的相关参数	说明
认证加密	<b>privacy</b>	算法、认证密码、加密密码	认证密码和加密密码必须和NMS上的一致才能建立SNMP连接
认证不加密	<b>authentication</b>	算法、认证密码	认证密码必须和NMS上的一致才能建立SNMP连接
不认证不加密	无	无	即便配置了认证密码、加密密码, 建立SNMP连接时也不会用到

### 1.8.2 非FIPS模式下配置SNMPv3 组和用户

- (1) 进入系统视图。

```
system-view
```

- (2) 创建 SNMPv3 组。

```
snmp-agent group v3 group-name [ authentication | privacy ] [ read-view
view-name ] [ write-view view-name ] [ notify-view view-name ] [ acl
{ ipv4-acl-number | name ipv4-acl-name } | acl ipv6 { ipv6-acl-number |
name ipv6-acl-name } ] *
```

- (3) (可选) 为明文密码计算对应的密文密码。

```
snmp-agent calculate-password plain-password mode { 3desmd5 | 3dessha |
aes192md5 | aes192sha | aes256md5 | aes256sha | md5 | sha }
{ local-engineid | specified-engineid engineid }
```

- (4) 创建 SNMPv3 用户。请选择其中一项进行配置。

- VACM 方式:

```
snmp-agent usm-user v3 user-name group-name [ remote { ipv4-address |
ipv6 ipv6-address } ] [ { cipher | simple } authentication-mode { md5 |
sha } auth-password [ privacy-mode { 3des | aes128 | aes192 | aes256 |
des56 } priv-password ] ] [ acl { ipv4-acl-number | name ipv4-acl-name }
| acl ipv6 { ipv6-acl-number | name ipv6-acl-name } ] *
```

- RBAC 方式:

```
snmp-agent usm-user v3 user-name user-role role-name [ remote
{ ipv4-address | ipv6 ipv6-address } ] [ { cipher | simple }
authentication-mode { md5 | sha } auth-password [ privacy-mode { 3des |
aes128 | aes192 | aes256 | des56 } priv-password ] ] [ acl
{ ipv4-acl-number | name ipv4-acl-name } | acl ipv6 { ipv6-acl-number
| name ipv6-acl-name } ] *
```

当设备需要向目的主机发送 SNMPv3 Inform 报文时，**remote** 参数必选。

如果使用 **cipher** 参数，则后面的 **auth-password** 和 **priv-password** 都必须输入并被视为密文密码。

- (5) (可选) 为通过 RBAC 方式创建的 SNMPv3 用户添加角色。

```
snmp-agent usm-user v3 user-name user-role role-name
```

缺省情况下，使用创建 SNMPv3 用户时指定的角色。

### 1.8.3 FIPS模式下配置SNMPv3 组和用户

- (1) 进入系统视图。

```
system-view
```

- (2) 创建 SNMPv3 组。

```
snmp-agent group v3 group-name { authentication | privacy } [ read-view
view-name ] [ write-view view-name ] [ notify-view view-name ] [ acl
{ ipv4-acl-number | name ipv4-acl-name } | acl ipv6 { ipv6-acl-number |
name ipv6-acl-name } ] *
```

- (3) (可选) 为明文密码计算对应的密文密码。

```
snmp-agent calculate-password plain-password mode { aes192sha |
aes256sha | sha } { local-engineid | specified-engineid engineid }
```

- (4) 创建 SNMPv3 用户。请选择其中一项进行配置。

- VACM 方式:

```
snmp-agent usm-user v3 user-name group-name [ remote { ipv4-address |
ipv6 ipv6-address } ] { cipher | simple } authentication-mode sha
auth-password [ privacy-mode { aes128 | aes192 | aes256 } priv-password ]
[ acl { ipv4-acl-number | name ipv4-acl-name } | acl ipv6
{ ipv6-acl-number | name ipv6-acl-name } ] *
```

- RBAC 方式:

```
snmp-agent usm-user v3 user-name user-role role-name [ remote
{ ipv4-address | ipv6 ipv6-address } ] { cipher | simple }
```

```
authentication-mode sha auth-password [ privacy-mode { aes128 | aes192
| aes256 } priv-password ] [ acl { ipv4-acl-number | name ipv4-acl-name }
| acl ipv6 { ipv6-acl-number | name ipv6-acl-name } ] *
```

当设备需要向目的主机发送 SNMPv3 Inform 报文时，**remote** 参数必选。

如果使用 **cipher** 参数，则后面的 **auth-password** 和 **priv-password** 都必须输入并被视为密文密码。

- (5) （可选）为通过 RBAC 方式创建的 SNMPv3 用户添加角色。

```
snmp-agent usm-user v3 user-name user-role role-name
```

缺省情况下，使用创建 SNMPv3 用户时指定的角色。

## 1.9 配置SNMP告警

### 1.9.1 功能简介

SNMP 告警信息包括 Trap 和 Inform 两种，用来告知 NMS 设备上发生了重要事件，比如，用户的登录/退出，接口状态变成 up/down 等。业务模块开启告警功能后，会将生成的告警信息发送给 SNMP 模块，SNMP 根据当前的配置封装成 Trap 和（或）Inform 报文发送。如无特殊说明，本文中的告警信息均指 Trap 和 Inform 两种信息。

### 1.9.2 开启告警功能

#### 1. 配置限制和指导

因为告警信息通常较多，会占用设备内存，影响设备性能，所以建议用户根据需要开启指定模块的告警功能，生成相应的告警信息。

如果要求接口在链路状态发生改变时生成相应的告警信息，需要在全局和接口下均开启接口链路状态变化的告警功能。如果要生成其它模块的告警信息，除了使用 **snmp-agent trap enable** 命令开启告警功能外，还可能需执行各个模块的相关配置，详情请参见各模块的相关描述。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启 SNMP 告警功能。

```
snmp-agent trap enable [ configuration | protocol | standard
[ authentication | coldstart | linkdown | linkup | warmstart ] * | system ]
```

缺省情况下，SNMP 配置告警、标准告警和系统告警功能处于开启状态，其他各模块告警功能是否开启请参见各模块手册。

- (3) 进入接口视图。

```
interface interface-type interface-number
```

- (4) 开启接口链路状态变化的告警功能。

```
enable snmp trap updown
```

缺省情况下，接口状态变化的告警功能处于开启状态。

## 1.9.3 配置告警信息发送参数

### 1. 功能简介

设备第一次发送告警信息时，会检查设备和目的主机是否路由可达。如果可达，则直接发送。如果不可达，则先将告警信息缓存在消息队列里，等路由可达后，再发送。为防止告警信息累积占用太多内存，用户可以配置该队列的长度以及告警信息在队列里的保存时间。

- 如果在告警信息的发送队列满时系统又收到了新的告警信息，则系统会自动删除最先收到的告警信息来保存新的告警信息。
- 如果告警信息的发送队列中的某信息到达了已设定的保存时间，则系统会自动删除该告警信息。

对 linkUp/linkDown 告警信息进行私有扩展后，设备生成和发送的该信息由标准 linkUp/linkDown 告警信息后增加接口描述和接口类型信息构成。

### 2. 配置Trap信息发送参数

- (1) 进入系统视图。

**system-view**

- (2) 配置 Trap 报文的发送参数。

（非 FIPS 模式）

```
snmp-agent target-host trap address udp-domain { ipv4-address | ipv6  
ipv6-address } [ udp-port port-number ] [ dscp dscp-value ] params  
securityname security-string [ v1 | v2c | v3 [ authentication | privacy ] ]
```

（FIPS 模式）

```
snmp-agent target-host trap address udp-domain { ipv4-address | ipv6  
ipv6-address } [ udp-port port-number ] [ dscp dscp-value ] params  
securityname security-string v3 { authentication | privacy }
```

缺省情况下，未配置 Trap 报文的目的主机（能够解析 Trap 和 Inform 报文的设备，通常为 NMS）。

- (3) （可选）配置发送的 Trap 报文的源 IP 地址。

```
snmp-agent trap source interface-type interface-number
```

缺省情况下，使用出接口的 IP 地址作为 Trap 报文的源 IP 地址。

### 3. 配置Inform信息发送参数

- (1) 进入系统视图。

**system-view**

- (2) 配置 Inform 报文的发送参数。

（非 FIPS 模式）

```
snmp-agent target-host inform address udp-domain { ipv4-address | ipv6  
ipv6-address } [ udp-port port-number ] params securityname  
security-string { v2c | v3 [ authentication | privacy ] }
```

（FIPS 模式）

```
snmp-agent target-host inform address udp-domain { ipv4-address | ipv6  
ipv6-address } [ udp-port port-number ] params securityname  
security-string v3 { authentication | privacy }
```

缺省情况下，未配置 Inform 报文的目的主机。

仅 SNMPv2c 和 SNMPv3 版本支持 Inform 报文。

- (3) （可选）配置发送的 Inform 报文的源 IP 地址。

```
snmp-agent inform source interface-type interface-number
```

缺省情况下，使用出接口的 IP 地址作为 Inform 报文的源 IP 地址。

#### 4. 配置告警信息公共发送参数

- (1) 进入系统视图。

```
system-view
```

- (2) 对标准 linkUp/linkDown 告警信息进行私有扩展。

```
snmp-agent trap if-mib link extended
```

缺省情况下，系统发送的 linkUp/linkDown 告警信息的格式为标准格式，不对其进行私有扩展。

如果 NMS 不支持该扩展信息，请关闭私有扩展功能。

- (3) 配置告警信息发送队列的长度。

```
snmp-agent trap queue-size size
```

缺省情况下，告警信息的信息队列最多可以存储 100 条告警信息。

- (4) 配置告警信息的保存时间。

```
snmp-agent trap life seconds
```

缺省情况下，告警信息的保存时间为 120 秒。

## 1.10 配置SNMP日志

### 1. 功能简介

SNMP 日志可以记录 NMS 对 Agent 的 Get 请求、Set 请求和 Set 响应信息，不能记录 Get 响应信息。同时 SNMP 日志可以记录 Agent 对 NMS 的 Trap 和 Inform 操作信息以及 SNMP 认证失败信息。

- 当进行 Get 操作时，Agent 会记录 NMS 用户的 IP 地址、Get 操作的节点名和节点 OID。
- 当进行 Set 操作时，Agent 会记录 NMS 用户的 IP 地址、Set 操作的节点名、节点 OID、节点值以及 Set 操作返回的错误码和错误索引。
- 当进行 Trap 和 Inform 操作时，Agent 会向 NMS 发送告警，Agent 会记录告警相关的信息。
- 当进行 SNMP 认证操作时，如果 Agent 收到来自 NMS 的 SNMP 请求但是没有通过认证，Agent 会记录相关日志。

这些日志将被发送到设备的信息中心，并通过信息中心配置的参数，最终决定 SNMP 日志的输出规则（即是否允许输出以及输出方向）。SNMP 每条日志信息中记录的 node 域（信息内容对应的 MIB 节点名）和 value 域（信息内容对应的 MIB 节点值）的长度之和不能超过 1024 字节，超出的部分将不会被输出。有关信息中心的详细介绍请参见“网络管理和监控配置指导”中的“信息中心”。

### 2. 配置限制和指导

大量的日志记录会占用设备的存储空间，影响设备的性能。正常情况下，建议关闭 SNMP 日志功能。

### 3. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 打开 SNMP 日志开关。

**snmp-agent log { all | authfail | get-operation | set-operation }**

缺省情况下，SNMP 日志开关处于关闭状态。

- (3) 打开 SNMP 告警日志开关。

**snmp-agent trap log**

缺省情况下，SNMP 告警日志功能处于关闭状态。

## 1.11 SNMP显示和维护

在完成上述配置后，在任意视图下执行 **display** 命令，均可以显示配置后 SNMP 的运行情况，通过查看显示信息，来验证配置的效果。

表1-2 SNMP 显示和维护

操作	命令
显示SNMPv1或SNMPv2c团体信息 (FIPS模式下不支持该命令)	<b>display snmp-agent community [ read   write ]</b>
显示SNMP上下文	<b>display snmp-agent context [ context-name ]</b>
显示SNMP组信息	<b>display snmp-agent group [ group-name ]</b>
显示本设备的SNMP引擎ID	<b>display snmp-agent local-engineid</b>
显示SNMP支持的MIB节点信息	<b>display snmp-agent mib-node [ details   index-node   trap-node   verbose ]</b>
显示MIB视图的信息	<b>display snmp-agent mib-view [ exclude   include   viewname view-name ]</b>
显示远端SNMP实体引擎信息	<b>display snmp-agent remote [ { ipv4-address   ipv6 ipv6-address } ]</b>
显示SNMP报文统计信息	<b>display snmp-agent statistics</b>
显示SNMP设备的系统信息	<b>display snmp-agent sys-info [ contact   location   version ] *</b>
显示告警信息队列的基本信息	<b>display snmp-agent trap queue</b>
显示SNMP告警功能的开启状态	<b>display snmp-agent trap-list</b>
显示SNMPv3用户信息	<b>display snmp-agent usm-user [ engineid engineid   username user-name   group group-name ] *</b>



## 1.12 SNMP典型配置举例

### 1.12.1 SNMPv1/v2c配置举例



说明

设备运行于 FIPS 模式时，不支持该配置举例。

SNMPv1 和 SNMPv2c 的配置方法相同，下面以 SNMPv1 为例进行配置。

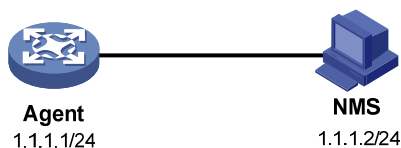
#### 1. 组网需求

NMS与Agent相连，设备的IP地址和掩码如 图 1-3 所示。

NMS 通过 SNMPv1 对 Agent 进行监控管理，Agent 在故障时能够主动向 NMS 发送告警信息。

#### 2. 组网图

图1-3 SNMPv1 配置组网图



#### 3. 配置步骤

##### (1) 配置 Agent

# 配置 Agent 的 IP 地址为 1.1.1.1/24，并确保 Agent 与 NMS 之间路由可达。（配置步骤略）

# 配置 Agent 支持 SNMPv1 版本、只读团体名为 public，读写团体名为 private。

```
<Agent> system-view
```

```
[Agent] snmp-agent sys-info version v1
```

```
[Agent] snmp-agent community read public
```

```
[Agent] snmp-agent community write private
```

# 配置设备的联系人和位置信息，以方便维护。

```
[Agent] snmp-agent sys-info contact Mr.Wang-Tel:3306
```

```
[Agent] snmp-agent sys-info location telephone-closet,3rd-floor
```

# 开启 NMS 告警功能，告警信息发送到主机 1.1.1.2，使用的团体名为 public。

```
[Agent] snmp-agent trap enable
```

```
[Agent] snmp-agent target-host trap address udp-domain 1.1.1.2 params securityname public v1
```

**snmp-agent target-host** 命令中指定的版本必须和 NMS 上运行的 SNMP 版本一致，因此需要将 **snmp-agent target-host** 命令中的版本参数配置为 v1。否则，NMS 无法正确接收告警信息。

##### (2) 配置 NMS

配置 NMS 使用的 SNMP 版本为 SNMPv1，只读团体名为 public，读写团体名为 private。另外，还可以根据需求配置“超时”时间和“重试次数”。具体配置请参考 NMS 的相关手册。





说明

NMS 侧的配置必须和 Agent 侧保持一致，否则无法通信。

### (3) 结果验证

# 通过查询 Agent 上相应的 MIB 节点获取 NULL0 接口的 MTU 值，结果为 1500:

```
Send request to 1.1.1.1/161 ...
Protocol version: SNMPv1
Operation: Get
Request binding:
1: 1.3.6.1.2.1.2.2.1.4.135471
Response binding:
1: Oid=ifMtu.135471 Syntax=INT Value=1500
Get finished
```

# 当使用错误的团体名获取 Agent 上的 MIB 节点信息时，NMS 上将看到认证失败的 Trap 信息，即 authenticationFailure:

```
1.1.1.1/2934 V1 Trap = authenticationFailure
SNMP Version = V1
Community = public
Command = Trap
Enterprise = 1.3.6.1.4.1.43.1.16.4.3.50
GenericID = 4
SpecificID = 0
Time Stamp = 8:35:25.68
```

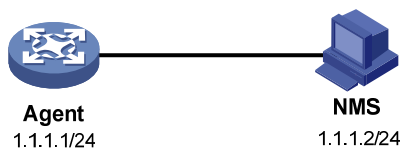
## 1.12.2 SNMPv3 配置举例

### 1. 组网需求

- NMS与Agent相连，设备的IP地址和掩码如 [图 1-4](#) 所示。
- NMS 通过 SNMPv3 只能对 Agent 的 SNMP 报文的相关信息监控管理，Agent 在出现故障时能够主动向 NMS 发送告警信息，NMS 上接收 SNMP 告警信息的默认 UDP 端口号为 162。
- NMS 与 Agent 建立 SNMP 连接时，需要认证，使用的认证算法为 SHA-1，认证密码为 123456TESTauth&!。NMS 与 Agent 之间传输的 SNMP 报文需要加密，使用的加密协议为 AES，加密密码为 123456TESTencr&!。

### 2. 组网图

图1-4 SNMPv3 配置组网图



### 3. 通过RBAC方式配置步骤

#### (1) 配置 Agent

# 配置 Agent 的 IP 地址为 1.1.1.1/24，并确保 Agent 与 NMS 之间路由可达。（配置步骤略）  
# 创建用户角色 test 并配置访问权限：用户只能读节点 snmpMIB（OID 为 1.3.6.1.6.3.1）下的对象（本举例验证时会用到 linkUp 和 linkDown 节点），不可以访问其它 MIB 对象。

```
<Agent> system-view
[Agent] role name test
[Agent-role-test] rule 1 permit read oid 1.3.6.1.6.3.1
# 配置用户角色 test 具有 system（OID 为 1.3.6.1.2.1.1）的读权限与 interfaces（OID 为 1.3.6.1.2.1.2）的读写权限，以便接口状态变化时，Agent 会向 NMS 发送告警信息。
[Agent-role-test] rule 2 permit read oid 1.3.6.1.2.1.1
[Agent-role-test] rule 3 permit read write oid 1.3.6.1.2.1.2
[Agent-role-test] quit
```

# 创建用户 RBACtest，为其绑定用户角色 test，认证算法为 SHA-1，认证密码为 123456TESTauth&！，加密算法为 AES，加密密码是 123456TESTencr&！。

```
[Agent] snmp-agent usm-user v3 RBACtest user-role test simple authentication-mode sha 123456TESTauth&! privacy-mode aes128 123456TESTencr&!
```

# 配置设备的联系人和位置信息，以方便维护。

```
[Agent] snmp-agent sys-info contact Mr.Wang-Tel:3306
[Agent] snmp-agent sys-info location telephone-closet,3rd-floor
```

# 开启 NMS 告警功能，告警信息发送到主机 1.1.1.2，使用的用户名为 RBACtest。

```
[Agent] snmp-agent trap enable
[Agent] snmp-agent target-host trap address udp-domain 1.1.1.2 params securityname RBACtest v3 privacy
```

#### (2) 配置 NMS

# 配置 NMS 使用的 SNMP 版本为 SNMPv3，用户名为 RBACtest，启用认证和加密功能，认证算法为 SHA-1，认证密码为 123456TESTauth&！，加密协议为 AES，加密密码为 123456TESTencr&！。另外，还可以根据需求配置“超时”时间和“重试次数”。具体配置请参考 NMS 的相关手册。



说明

NMS 侧的配置必须和设备侧保持一致，否则无法进行相应操作。

---

### 4. 通过VACM方式配置步骤

#### (1) 配置 Agent

# 配置 Agent 的 IP 地址为 1.1.1.1/24，并确保 Agent 与 NMS 之间路由可达。（配置步骤略）  
# 配置访问权限：用户只能读节点 snmpMIB（OID 为 1.3.6.1.6.3.1）下的对象（本举例验证时会用到 linkUp 和 linkDown 节点），不可以访问其它 MIB 对象。

```
<Agent> system-view
[Agent] undo snmp-agent mib-view ViewDefault
[Agent] snmp-agent mib-view included test snmpMIB
[Agent] snmp-agent group v3 managev3group privacy read-view test
```

# 配置访问权限：配置用户具有 **system**（OID 为 1.3.6.1.2.1.1）和 **interfaces**（OID 为 1.3.6.1.2.1.2）的读写权限，以便接口状态变化时时，Agent 会向 NMS 发送告警信息。（VACM 方式只能将 MIB 视图中包含的所有节点配置为只读属性或者读写属性，不能仅配置部分节点的属性）

```
[Agent] snmp-agent mib-view included test 1.3.6.1.2.1.1
```

```
[Agent] snmp-agent mib-view included test 1.3.6.1.2.1.2
```

```
[Agent] snmp-agent group v3 managev3group privacy read-view test write-view test
```

# 配置 Agent 使用的用户名为 **VACMtest**，认证算法为 **SHA-1**，认证密码为 **123456TESTauth&!**，加密算法为 **AES**，加密密码是 **123456TESTencr&!**。

```
[Agent] snmp-agent usm-user v3 VACMtest managev3group simple authentication-mode sha 123456TESTauth&! privacy-mode aes128 123456TESTencr&!
```

# 配置设备的联系人和位置信息，以方便维护。

```
[Agent] snmp-agent sys-info contact Mr.Wang-Tel:3306
```

```
[Agent] snmp-agent sys-info location telephone-closet,3rd-floor
```

# 开启 NMS 告警功能，告警信息发送到主机 1.1.1.2，使用的用户名为 **VACMtest**。

```
[Agent] snmp-agent trap enable
```

```
[Agent] snmp-agent target-host trap address udp-domain 1.1.1.2 params securityname VACMtest v3 privacy
```

## (2) 配置 NMS

# 配置 NMS 使用的 **SNMP** 版本为 **SNMPv3**，用户名为 **VACMtest**，启用认证和加密功能，认证算法为 **SHA-1**，认证密码为 **123456TESTauth&!**，加密协议为 **AES**，加密密码为 **123456TESTencr&!**。另外，还可以根据需求配置“超时”时间和“重试次数”。具体配置请参考 NMS 的相关手册。



说明

NMS 侧的配置必须和设备侧保持一致，否则无法进行相应操作。

---

## 5. 验证配置

### (1) NMS 使用 RBACtest 用户名访问 Agent

可查询 **sysName** 节点的值，返回结果为 **Agent**。

设置 **sysName** 节点的值 **Sysname**，由于没有权限，操作失败。

在 Agent 上关闭或打开接口，NMS 上将收到 **linkUP**（OID 为 1.3.6.1.6.3.1.1.5.4）或 **linkDown**（OID 为 1.3.6.1.6.3.1.1.5.3）Trap 信息。

### (2) NMS 使用 VACMtest 用户名访问 Agent

可查询 **sysName** 节点的值，返回结果为 **Agent**。

设置 **sysName** 节点的值 **Sysname**，操作成功。

在 Agent 上关闭或打开接口，NMS 上将收到 **linkUP** 或 **linkDown** Trap 信息。

# 目 录

1 RMON.....	1-1
1.1 RMON简介 .....	1-1
1.1.1 RMON工作机制.....	1-1
1.1.2 RMON组.....	1-1
1.1.3 采样类型.....	1-2
1.1.4 协议规范.....	1-3
1.2 配置RMON统计功能 .....	1-3
1.2.1 功能简介.....	1-3
1.2.2 配置RMON以太网统计功能.....	1-3
1.2.3 配置RMON历史统计功能.....	1-3
1.3 配置RMON告警功能 .....	1-4
1.4 RMON显示和维护 .....	1-5
1.5 RMON典型配置举例 .....	1-5
1.5.1 统计功能配置举例.....	1-5
1.5.2 历史统计功能配置举例.....	1-6
1.5.3 告警功能配置举例.....	1-7

# 1 RMON

## 1.1 RMON简介

RMON (Remote Network Monitoring, 远程网络监视) 是一种基于 SNMP 协议的网络管理协议, 通常用来实现管理设备对被管理设备的远程监控和管理。

### 1.1.1 RMON工作机制

用户通过 RMON 功能可以周期性或者持续性统计以太网接口收发报文的个数, 可以监控被管理设备上 MIB 节点的值, 当这些值达到用户设置的阈值, 设备会自动记录日志或者发送告警信息给管理设备, 来实现管理设备对被管理设备的远程监控和管理。

SNMP 是 RMON 实现的基础。RMON 使用 SNMP 告警信息发送机制向管理设备发送告警信息告知告警变量的异常。虽然 SNMP 也定义了告警功能, 但通常用于告知被管理设备上某功能是否运行正常、接口物理状态的变化等, 两者监控的对象、触发条件以及报告的内容均不同。

RMON 是 SNMP 功能的增强。RMON 协议规定达到告警阈值时被管理设备能自动生成告警信息发送给设备的 SNMP 模块, 所以管理设备不需要多次去获取 MIB 变量的值, 进行比较, 从而能够减少管理设备同被管理设备的通讯流量, 达到简便而有力地管理大型互连网络的目的。

### 1.1.2 RMON组

RMON 协议中定义了多个 RMON 组, 设备实现了公有 MIB 中支持的统计组、历史组、事件组、告警组、代理配置组 and 用户历史组。此外, Comware 系统还自定义和实现了扩展告警组, 以增强告警组的功能。其中, 代理配置组 and 用户历史组只支持 MIB 操作。

#### 1. 统计组

统计组规定系统将持续地对端口的各种流量信息进行统计 (目前只支持对以太网端口的统计), 并将统计结果存储在以太网统计表 (etherStatsTable) 中以便管理设备随时查看。在指定接口下创建统计表项成功后, 统计组就对当前接口的报文数进行统计, 它统计的结果是一个连续的累加值。

统计信息包括网络冲突数、CRC 校验错误报文数、过小 (或超大) 的数据报文数、广播、多播的报文数以及接收字节数、接收报文数等。

#### 2. 历史组

历史组规定系统将按指定周期对端口的各种流量信息进行统计, 并将统计结果存储在历史记录表 (etherHistoryTable) 中以便管理设备随时查看。历史组统计的是每个周期内端口接收报文的情况, 统计数据包括带宽利用率、错误包数和总包数等, 周期的长短可以通过命令行来配置。

#### 3. 事件组

事件组用来定义事件索引号及事件的处理方式。事件组定义的事件用于告警组表项和扩展告警组表项中。当监控对象达到告警条件时, 就会触发事件, 事件有如下几种处理方式:

- **Log:** 将事件相关信息 (事件发生的时间、事件的内容等) 记录在本设备 RMON MIB 的事件日志表中, 以便管理设备通过 SNMP Get 操作进行查看。
- **Trap:** 表示事件被触发时, 会生成告警信息发送给设备的 SNMP 模块。

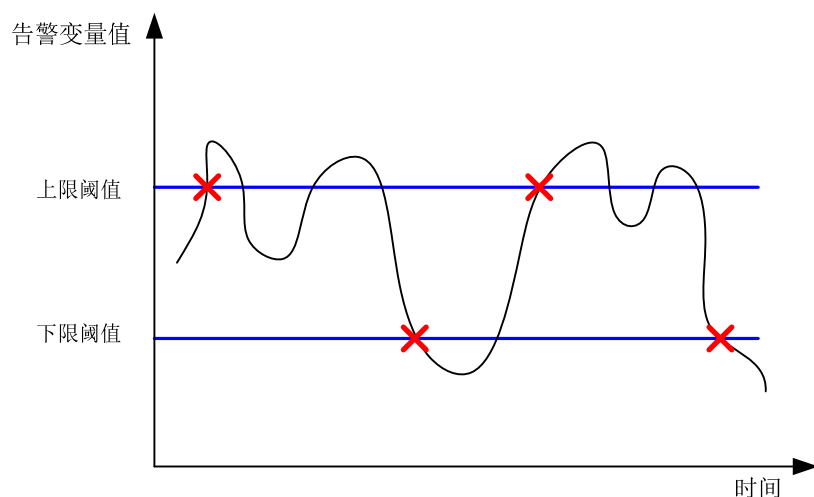
- **Log-Trap:** 表示事件被触发时，既在本设备上记录日志，又会生成告警信息发送给设备的 SNMP 模块。
- **None:** 不做任何处理。

#### 4. 告警组

RMON 告警管理可对指定的告警变量（如端口收到的报文总数 `etherStatsPkts`）进行监视。用户定义了告警表项后，系统会按照定义的时间周期去获取被监视的告警变量的值，当告警变量的值大于或等于上限阈值时，触发一次上限告警事件；当告警变量的值小于或等于下限阈值，触发一次下限告警事件，告警管理将按照事件的定义进行相应的处理。

当告警变量的采样值在同一方向上连续多次超过阈值时，只会在第一次产生告警事件，后面的几次不会产生告警事件。即上限告警和下限告警是交替产生的，出现了一次上限告警，则下一次必为下限告警。如 [图 1-1](#) 所示，告警变量的值（如图中黑色曲线所示）多次超过阈值（如图中蓝色直线所示），产生了多个交叉点，但只有红叉标识的交叉点才会触发告警事件，其它交叉点不会触发告警事件。

图1-1 上下限告警示意图



#### 5. 扩展告警组

扩展告警表项可以对告警变量进行运算，然后将运算结果和配置的阈值比较，实现更为丰富的告警功能。

用户定义了扩展告警表项后，系统对扩展告警表项的处理如下：

- (1) 对定义的扩展告警公式中的告警变量按照定义的时间间隔进行采样。
- (2) 将采样值按照定义的运算公式进行计算。
- (3) 将计算结果和设定的阈值进行比较，大于或等于上限阈值时，触发一次上限告警事件；小于或等于下限阈值，触发一次下限告警事件。

与告警组一样，当扩展告警组的运算结果在同一方向上连续多次超过阈值时，只会在第一次产生告警事件，后面的几次不会产生告警事件，即上限告警和下限告警是交替产生的。

### 1.1.3 采样类型

设备在监控股警组和扩展告警组的告警变量时，支持两种采样类型：

- 绝对值采样：设备按照采样间隔获取告警节点的值，直接用告警节点的值或者计算后获得的值和上限阈值、下限阈值进行比较，来触发对应的事件。
- 变化值采样：设备按照采样间隔对告警节点进行采样，并用当前采样时间点获取的值减去上一个采样时间点获取的值，得到告警节点在采样间隔内的变化值，直接用告警节点的变化值或者计算后得到的值和上限阈值、下限阈值进行比较，来触发对应的事件。

#### 1.1.4 协议规范

与 RMON 相关的协议规范有：

- RFC 4502: Remote Network Monitoring Management Information Base Version 2
- RFC 2819: Remote Network Monitoring Management Information Base Status of this Memo

## 1.2 配置RMON统计功能

### 1.2.1 功能简介

RMON 的统计功能可以通过 RMON 统计组或者 RMON 历史组来实现，但是两者统计的对象不同，请根据实际需要配置。

- RMON 统计组统计的是 RMON 以太网统计表里定义的变量，记录的是从 RMON 统计表项创建到当前阶段变量的累加值。
- RMON 历史组统计的是 RMON 历史记录表里定义的变量，记录的是每个周期内变量的累加值。

### 1.2.2 配置RMON以太网统计功能

(1) 进入系统视图。

```
system-view
```

(2) 进入以太网接口视图。

```
interface interface-type interface-number
```

(3) 创建统计表项。

```
rmon statistics entry-number [ owner text ]
```

每个接口下只能创建一个统计表项。

### 1.2.3 配置RMON历史统计功能

#### 1. 配置限制和指导

历史控制表项的 *entry-number* 必须全局唯一，如果已经在其他接口下使用，则创建操作失败。

同一接口下，可以创建多条历史控制表项，但要求不同表项 *entry-number* 和 *sampling-interval* 的值必须不同，否则创建操作失败。

在创建历史控制表项时，如果指定的 **buckets number** 参数值超出了设备实际支持的历史表容量时，该历史控制表项会被添加，但该表项对应生效的 **buckets number** 的值为设备实际支持的历史表容量。

## 2. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 进入以太网接口视图。

**interface** *interface-type interface-number*

- (3) 创建历史控制表项。

**rmon history** *entry-number* **buckets** *number* **interval** *interval* [ **owner** *text* ]

同一接口下可以创建多个历史控制表项。

## 1.3 配置RMON告警功能

### 1. 配置限制和指导

- 系统不允许创建两个配置完全相同的表项。如果新建表项参数的值和已存在表项对应参数的值完全相同，则创建操作失败。不同表项需要比较的参数不同，请参见 [表 1-1](#)。
- 系统对每种类型表项的总数均进行了限制，具体数目请参见 [表 1-1](#)。当某种类型表项的总数达到系统允许创建的最大数目时，创建操作失败。

表1-1 RMON 配置约束表

表项名	需要比较的参数	最多可创建的表项数
事件表项	事件描述 ( <b>description</b> <i>string</i> )、事件类型 ( <b>log</b> 、 <b>trap</b> 、 <b>logtrap</b> 或 <b>none</b> ) 和团体名 ( <i>security-string</i> )	60
告警表项	告警变量 ( <i>alarm-variable</i> )、采样间隔 ( <i>sampling-interval</i> )、采样类型 ( <b>absolute</b> 或 <b>delta</b> )、上限阈值 ( <i>threshold-value1</i> ) 和下限阈值 ( <i>threshold-value2</i> )	60
扩展告警表项	告警变量公式 ( <i>prialarm-formula</i> )、采样间隔 ( <i>sampling-interval</i> )、采样类型 ( <b>absolute</b> 或 <b>delta</b> )、上限阈值 ( <i>threshold-value1</i> ) 和下限阈值 ( <i>threshold-value2</i> )	50

## 2. 配置准备

如果触发告警事件时，需要向管理设备（NMS）发送告警信息，则在配置 RMON 告警功能之前，必须保证 SNMP Agent 已经正确配置。SNMP Agent 的配置请参见“网络管理和监控配置指导”中的“SNMP”。

## 3. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) （可选）创建事件表项。

**rmon event** *entry-number* [ **description** *string* ] { **log** | **log-trap** *security-string* | **none** | **trap** *security-string* } [ **owner** *text* ]

缺省情况下，事件表中不存在表项。



- (3) 创建告警表项。

```
rmon alarm entry-number alarm-variable sampling-interval { absolute |  
delta } [ startup-alarm { falling | rising | rising-falling } ]  
rising-threshold threshold-value1 event-entry1 falling-threshold  
threshold-value2 event-entry2 [ owner text ]
```

配置时如果 *event-entry* 不存在也可以配置，但触发告警时不会有事件动作。

- (4) 创建扩展告警表项。

```
rmon prialarm entry-number prialarm-formula prialarm-des  
sampling-interval { absolute | delta } [ startup-alarm { falling | rising  
| rising-falling } ] rising-threshold threshold-value1 event-entry1  
falling-threshold threshold-value2 event-entry2 entrytype { forever |  
cycle cycle-period } [ owner text ]
```

配置时如果 *event-entry* 不存在也可以配置，但触发告警时不会有事件动作。

## 1.4 RMON显示和维护

在完成上述配置后，在任意视图下执行 **display** 命令可以显示配置后 RMON 的运行情况，通过查看显示信息，验证配置的效果。

表1-2 RMON 显示和维护

操作	命令
显示RMON告警表项的相关信息	<b>display rmon alarm</b> [ <i>entry-number</i> ]
显示RMON事件表项的相关信息	<b>display rmon event</b> [ <i>entry-number</i> ]
显示事件日志表项的相关信息	<b>display rmon eventlog</b> [ <i>entry-number</i> ]
显示RMON历史控制表及历史采样信息	<b>display rmon history</b> [ <i>interface-type</i> <i>interface-number</i> ]
显示RMON扩展告警表项的相关信息	<b>display rmon prialarm</b> [ <i>entry-number</i> ]
显示RMON统计信息	<b>display rmon statistics</b> [ <i>interface-type</i> <i>interface-number</i> ]

## 1.5 RMON典型配置举例

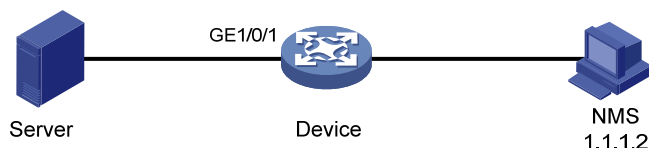
### 1.5.1 统计功能配置举例

#### 1. 组网需求

现需要通过 RMON 统计表对以太网接口 GigabitEthernet1/0/1 接收的报文进行性能统计。管理员随时查看统计数据了解接口接收报文的情况。

## 2. 组网图

图1-2 配置统计功能组网图



## 3. 配置步骤

# 使用 RMON 对接口 GigabitEthernet1/0/1 进行流量统计。

```
<Sysname> system-view
[Sysname] interface gigabitethernet 1/0/1
[Sysname-GigabitEthernet1/0/1] rmon statistics 1 owner user1
```

## 4. 验证配置

完成以上配置后，系统就开始对接口 GigabitEthernet1/0/1 接收的报文进行分类统计，统计的结果可以通过多种方式查看：

- 查看接口的统计信息。

```
<Sysname> display rmon statistics gigabitethernet 1/0/1
EtherStatsEntry 1 owned by user1 is VALID.
  Interface : GigabitEthernet1/0/1<ifIndex.3>
  etherStatsOctets      : 21657      , etherStatsPkts      : 307
  etherStatsBroadcastPkts : 56      , etherStatsMulticastPkts : 34
  etherStatsUndersizePkts : 0      , etherStatsOversizePkts : 0
  etherStatsFragments   : 0      , etherStatsJabbers     : 0
  etherStatsCRCAlignErrors : 0      , etherStatsCollisions  : 0
  etherStatsDropEvents (insufficient resources): 0
  Incoming packets by size:
  64      : 235      , 65-127 : 67      , 128-255 : 4
  256-511: 1      , 512-1023: 0      , 1024-1518: 0
```

- 在 NMS 上通过软件执行 Get 操作，直接获取 MIB 节点的值。

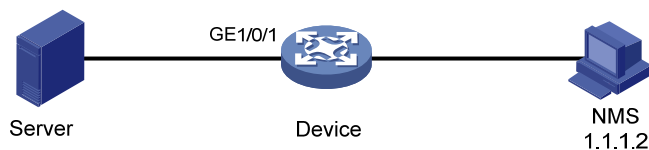
## 1.5.2 历史统计功能配置举例

### 1. 组网需求

现需要每 1 分钟通过 RMON 历史统计表对以太网接口 GigabitEthernet1/0/1 接收的报文进行周期性统计。管理员能够随时了解短时间内接口是否有数据突发。

### 2. 组网图

图1-3 配置历史统计功能组网图



### 3. 配置步骤

# 使用 RMON 对接口 GigabitEthernet1/0/1 进行周期性流量统计。

```
<Sysname> system-view
[Sysname] interface gigabitethernet 1/0/1
[Sysname-GigabitEthernet1/0/1] rmon history 1 buckets 8 interval 60 owner user1
```

### 4. 验证配置

完成以上配置后，系统就开始对接口 GigabitEthernet1/0/1 接收的报文进行周期性分类统计，每 1 分钟统计一次，历史统计列表里会保存最近的 8 次统计的结果，以便管理员查看。统计的结果可以通过以下方式查看：

- 查看接口的历史统计信息。

```
[Sysname-GigabitEthernet1/0/1] display rmon history
HistoryControlEntry 1 owned by user1 is VALID
  Sampled interface      : GigabitEthernet1/0/1<ifIndex.3>
  Sampling interval      : 60(sec) with 8 buckets max
  Sampling record 1 :
    dropevents           : 0           , octets           : 834
    packets              : 8           , broadcast packets : 1
    multicast packets    : 6           , CRC alignment errors : 0
    undersize packets    : 0           , oversize packets   : 0
    fragments            : 0           , jabbers            : 0
    collisions           : 0           , utilization         : 0
  Sampling record 2 :
    dropevents           : 0           , octets           : 962
    packets              : 10          , broadcast packets : 3
    multicast packets    : 6           , CRC alignment errors : 0
    undersize packets    : 0           , oversize packets   : 0
    fragments            : 0           , jabbers            : 0
    collisions           : 0           , utilization         : 0
```

- 在 NMS 上通过软件执行 Get 操作，直接获取 MIB 节点的值。

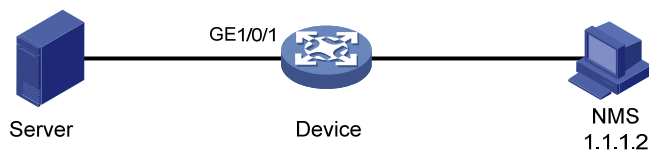
## 1.5.3 告警功能配置举例

### 1. 组网需求

Device 与 Server 和 NMS 相连。现需要对该服务器的流量进行统计，以 5 秒的采样间隔采样，当流量过大或过低时发送相应的告警信息给设备的 SNMP 模块，由 SNMP 模块通知 NMS。

### 2. 组网图

图1-4 配置告警功能组网图



### 3. 配置步骤

# 配置 SNMP Agent。(请注意: 这些参数的值应与 NMS 侧的配置一致, 假设 NMS 上运行 SNMP v1, 访问设备时使用的可读团体名为 public, 可写团体名为 private, NMS 的 IP 地址为 1.1.1.2)

```
<Sysname> system-view
[Sysname] snmp-agent
[Sysname] snmp-agent community read public
[Sysname] snmp-agent community write private
[Sysname] snmp-agent sys-info version v1
[Sysname] snmp-agent trap enable
[Sysname] snmp-agent trap log
[Sysname] snmp-agent target-host trap address udp-domain 1.1.1.2 params securityname public
```

# 使用 RMON 对以太网接口 GigabitEthernet1/0/1 进行统计。

```
[Sysname] interface gigabitethernet 1/0/1
[Sysname-GigabitEthernet1/0/1] rmon statistics 1 owner user1
[Sysname-GigabitEthernet1/0/1] quit
```

# 配置 RMON 告警表项。(当节点 1.3.6.1.2.1.16.1.1.1.4.1 的相对采样值超过 100 时或者低于 50 时, 都触发事件 1—生成告警信息发送给设备的 SNMP 模块)。

```
[Sysname] rmon event 1 trap public owner user1
[Sysname] rmon alarm 1 1.3.6.1.2.1.16.1.1.1.4.1 5 delta rising-threshold 100 1
falling-threshold 50 1 owner user1
```

### 4. 验证配置

# 查看 RMON 告警表信息。

```
<Sysname> display rmon alarm 1
AlarmEntry 1 owned by user1 is VALID.
Sample type                : delta
Sampled variable            : 1.3.6.1.2.1.16.1.1.1.4.1<etherStatsOctets.1>
Sampling interval (in seconds) : 5
Rising threshold            : 100(associated with event 1)
Falling threshold           : 50(associated with event 1)
Alarm sent upon entry startup : risingOrFallingAlarm
Latest value                : 0
```

# 查看以太网接口的统计信息。

```
<Sysname> display rmon statistics gigabitethernet 1/0/1
EtherStatsEntry 1 owned by user1 is VALID.
Interface : GigabitEthernet1/0/1<ifIndex.3>
etherStatsOctets      : 57329      , etherStatsPkts      : 455
etherStatsBroadcastPkts : 53      , etherStatsMulticastPkts : 353
etherStatsUndersizePkts : 0      , etherStatsOversizePkts : 0
etherStatsFragments   : 0      , etherStatsJabbers     : 0
etherStatsCRCAlignErrors : 0      , etherStatsCollisions   : 0
etherStatsDropEvents (insufficient resources): 0
Incoming packets by size :
64      : 7      , 65-127 : 413      , 128-255 : 35
256-511: 0      , 512-1023: 0      , 1024-1518: 0
```

当告警事件被触发时, NMS 将收到相应的告警信息。

# 目 录

1 NETCONF .....	1-1
1.1 NETCONF简介 .....	1-1
1.1.1 NETCONF协议结构 .....	1-1
1.1.2 NETCONF报文格式 .....	1-2
1.1.3 如何使用NETCONF .....	1-3
1.1.4 协议规范 .....	1-3
1.2 FIPS相关说明 .....	1-3
1.3 NETCONF配置任务简介 .....	1-3
1.4 建立NETCONF会话 .....	1-4
1.4.1 配置限制和指导 .....	1-4
1.4.2 配置NETCONF会话公共属性 .....	1-5
1.4.3 建立NETCONF over SOAP会话 .....	1-6
1.4.4 建立NETCONF over SSH会话 .....	1-7
1.4.5 建立NETCONF over Telnet或NETCONF over Console会话 .....	1-8
1.4.6 交换能力集 .....	1-8
1.5 获取设备配置信息 .....	1-9
1.5.1 配置限制和指导 .....	1-9
1.5.2 获取运行配置和运行状态 .....	1-9
1.5.3 获取可配置的功能的变量值 .....	1-11
1.5.4 获取NETCONF信息 .....	1-12
1.5.5 获取YANG文件的内容 .....	1-13
1.5.6 获取NETCONF会话信息 .....	1-13
1.5.7 获取运行配置举例 .....	1-14
1.5.8 获取可配置的功能的变量值配置举例 .....	1-15
1.5.9 获取Syslog功能的变量值配置举例 .....	1-17
1.5.10 获取NETCONF会话信息配置举例 .....	1-18
1.6 过滤表和列的信息 .....	1-19
1.6.1 功能简介 .....	1-19
1.6.2 配置限制和指导 .....	1-19
1.6.3 基于表的过滤 .....	1-19
1.6.4 基于列的过滤 .....	1-19
1.6.5 正则表达式过滤配置举例 .....	1-22
1.6.6 条件匹配过滤配置举例 .....	1-23

1.7 配置加锁和解锁.....	1-24
1.7.1 功能简介.....	1-24
1.7.2 配置限制和指导.....	1-25
1.7.3 配置加锁.....	1-25
1.7.4 配置解锁.....	1-25
1.7.5 加锁配置举例.....	1-26
1.8 下发配置.....	1-27
1.8.1 功能简介.....	1-27
1.8.2 配置步骤.....	1-27
1.8.3 下发配置举例.....	1-28
1.9 保存配置.....	1-29
1.9.1 功能简介.....	1-29
1.9.2 配置限制和指导.....	1-29
1.9.3 配置步骤.....	1-29
1.9.4 保存配置举例.....	1-29
1.10 加载配置文件.....	1-30
1.10.1 功能简介.....	1-30
1.10.2 配置限制和指导.....	1-30
1.10.3 配置步骤.....	1-31
1.11 回滚配置.....	1-31
1.11.1 配置限制和指导.....	1-31
1.11.2 回滚配置文件.....	1-31
1.11.3 回滚运行配置.....	1-31
1.12 下发命令行.....	1-36
1.12.1 功能简介.....	1-36
1.12.2 配置限制和指导.....	1-36
1.12.3 命令行操作.....	1-36
1.12.4 命令行操作配置举例.....	1-36
1.13 订阅事件.....	1-37
1.13.1 功能简介.....	1-37
1.13.2 配置限制和指导.....	1-37
1.13.3 订阅Syslog事件.....	1-37
1.13.4 订阅监控事件.....	1-39
1.13.5 订阅模块上报事件.....	1-40
1.13.6 订阅事件配置举例.....	1-41
1.14 关闭NETCONF会话.....	1-42

1.14.1 功能简介 .....	1-42
1.14.2 Kill-session操作 .....	1-42
1.14.3 Kill-session操作配置举例 .....	1-43
1.15 退出XML视图 .....	1-43
<b>2 Comware V7 支持的NETCONF操作 .....</b>	<b>2-1</b>
2.1.1 action .....	2-1
2.1.2 CLI .....	2-1
2.1.3 close-session .....	2-2
2.1.4 edit-config: create .....	2-2
2.1.5 edit-config: delete .....	2-3
2.1.6 edit-config: merge .....	2-3
2.1.7 edit-config: remove .....	2-3
2.1.8 edit-config: replace .....	2-3
2.1.9 edit-config测试处理选项 .....	2-4
2.1.10 edit-config缺省操作选项 .....	2-4
2.1.11 edit-config缺省错误处理选项 .....	2-5
2.1.12 edit-config增量下发 .....	2-6
2.1.13 get .....	2-7
2.1.14 get-bulk .....	2-7
2.1.15 get-bulk-config .....	2-8
2.1.16 get-config .....	2-8
2.1.17 get-sessions .....	2-9
2.1.18 kill-session .....	2-9
2.1.19 load .....	2-9
2.1.20 lock .....	2-10
2.1.21 rollback .....	2-10
2.1.22 save .....	2-10
2.1.23 unlock .....	2-11

# 1 NETCONF

## 1.1 NETCONF简介

NETCONF (Network Configuration Protocol, 网络配置协议) 是一种基于 XML 的网络管理协议, 他提供了一种可编程的、对网络设备进行配置和管理的方法。用户可以通过该协议设置属性、获取属性值、获取统计信息等。这使得他在第三方软件的开发上非常便利, 很容易开发出在混合不同厂商、不同设备的环境下的特殊定制的网管软件。

### 1.1.1 NETCONF协议结构

NETCONF 协议采用分层结构, 分为内容层 (Content)、操作层 (Operations)、RPC (Remote Procedure Call, 远程调用) 层和通信协议层 (Transport Protocol) 等。

表1-1 XML 分层与 NETCONF 分层模型对应关系

NETCONF 分层	XML 分层	说明
内容层	配置数据、状态数据、统计信息等	被管理对象的集合, 可以是配置数据、状态数据、统计信息等 NETCONF协议具体可读写的的数据请参见《NETCONF XML API 手册》
操作层	<get>,<get-config>,<edit-config>...	在RPC中应用的基本的原语操作集, 这些操作组成NETCONF的基本能力 NETCONF全面地定义了对被管理设备的各种基础操作, 设备支持的操作请参见“ <a href="#">2 Comware V7支持的NETCONF操作</a> ”
RPC层	<rpc>,<rpc-reply>	为RPC模块的编码提供了简单的、传输协议无关的机制。通过使用<rpc>和<rpc-reply>元素分别对NETCONF请求和响应数据 (即操作层和内容层的内容) 进行封装
通信协议层	非FIPS模式下: Console/Telnet/SSH/HTTP/HTTPS/TLS FIPS模式下: Console/SSH/HTTP S/TLS	为NETCONF提供面向连接的、可靠的、顺序的数据链路。 非FIPS模式下: <ul style="list-style-type: none"><li>NETCONF 支持 Telnet、SSH 和 Console 等 CLI 登录方式/协议, 即 NETCONF over SSH、NETCONF over Telnet 和 NETCONF over Console</li><li>NETCONF 支持 HTTP 和 HTTPS 协议, 即 NETCONF over HTTP 和 NETCONF over HTTPS</li><li>NETCONF 支持封装成 SOAP (Simple Object Access Protocol, 简单对象访问协议) 报文后通过 HTTP 或 HTTPS 协议传输, 即 NETCONF over SOAP over HTTP 和 NETCONF over SOAP over HTTPS</li></ul> FIPS模式下: <ul style="list-style-type: none"><li>NETCONF 支持 SSH 和 Console 等 CLI 方式/协议, 即 NETCONF over SSH 和 NETCONF over Console</li><li>NETCONF 支持 HTTPS 登录协议, 即 NETCONF over HTTPS</li><li>NETCONF 支持封装成 SOAP 报文后通过 HTTPS 协议传输, 即 NETCONF over SOAP over HTTPS</li></ul>



## 1.1.2 NETCONF报文格式

### 1. NETCONF

NETCONF 命令必须符合 XML 语言的基本格式，格式遵循 RFC 4741。

NETCONF 操作以及可操作的数据项，请参见《NETCONF XML API 手册》。NETCONF 报文的数据合法性都将经过校验才会下发，如果校验失败则会向客户端报错。其中，数据合法性校验通过 XML Schema 的方式完成。

如下为一个 NETCONF 报文示例，用于获取设备上所有接口的所有参数：

```
<?xml version="1.0" encoding="utf-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-bulk>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        <Ifmgr>
          <Interfaces>
            <Interface/>
          </Interfaces>
        </Ifmgr>
      </top>
    </filter>
  </get-bulk>
</rpc>
```

### 2. NETCONF over SOAP

NETCONF over SOAP 之后，NETCONF 报文会放在 SOAP 报文的 BODY 元素里，这些报文除了需要遵循纯 NETCONF 报文的规则外，还需要遵循以下规则：

- SOAP 消息必须用 XML 来编码。
- SOAP 消息必须使用 SOAP Envelope 命名空间。
- SOAP 消息必须使用 SOAP Encoding 命名空间。
- SOAP 消息不能包含 DTD（Document Type Definition，文件类型定义）引用。
- SOAP 消息不能包含 XML 处理指令。

如下为一个 NETCONF over SOAP 报文示例，用于获取设备上所有接口的所有参数：

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Header>
    <auth:Authentication env:mustUnderstand="1"
xmlns:auth="http://www.h3c.com/netconf/base:1.0">
      <auth:AuthInfo>800207F0120020C</auth:AuthInfo>
    </auth:Authentication>
  </env:Header>
  <env:Body>
    <rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <get-bulk>
        <filter type="subtree">
          <top xmlns="http://www.h3c.com/netconf/data:1.0">
            <Ifmgr>
```

```

        <Interfaces>
            <Interface/>
        </Interfaces>
    </Ifmgr>
</top>
</filter>
</get-bulk>
</rpc>
</env:Body>
</env:Envelope>

```

### 1.1.3 如何使用NETCONF

用户可通过以下方式使用 NETCONF 配置和管理设备：

- 通过 Telnet、SSH 和 Console 登录到设备并进入 XML 视图，将合法的 NETCONF 报文直接拷贝、粘贴到命令行提示符处执行，即可实现对设备的配置和管理。
- 通过 HTTP 或 HTTPS 登录到设备的 Web 页面，系统会自动将 Web 页面的配置转换成 NETCONF 指令下发给设备来实现对设备的配置和管理。该方式通过简洁易用的图像化 Web 页面上完成配置，配置结果也会以简洁清晰的方式呈现给用户，整个 NETCONF 的交互过程对用户透明。
- 使用配置工具给设备下发 NETCONF 指令来实现对设备的配置和管理。该方式需要将 NETCONF 指令用 SOAP 封装成通用的格式，以便设备能够正常转换。因此，使用该方式前必须开启 NETCONF over SOAP 功能。

### 1.1.4 协议规范

与 NETCONF、SOAP 协议相关的协议规范有：

- RFC 3339: Date and Time on the Internet: Timestamps
- RFC 4741: NETCONF Configuration Protocol
- RFC 4742: Using the NETCONF Configuration Protocol over Secure SHell (SSH)
- RFC 4743: Using NETCONF over the Simple Object Access Protocol (SOAP)
- RFC 5277: NETCONF Event Notifications
- RFC 5381: Experience of Implementing NETCONF over SOAP
- RFC 5539: NETCONF over Transport Layer Security (TLS)
- RFC 6241: Network Configuration Protocol

## 1.2 FIPS相关说明

设备运行于 FIPS 模式时，本特性部分配置相对于非 FIPS 模式有所变化，具体差异请见本文相关描述。有关 FIPS 模式的详细介绍请参见“安全配置指导”中的“FIPS”。

## 1.3 NETCONF配置任务简介

NETCONF 配置任务如下：

- (1) [建立NETCONF会话](#)
  - a. (可选) [配置NETCONF会话公共属性](#)
  - b. [建立NETCONF over SOAP会话](#)
  - c. [建立NETCONF over SSH会话](#)
  - d. [建立NETCONF over Telnet或NETCONF over Console会话](#)
  - e. [交换能力集](#)
- (2) (可选) [获取设备配置信息](#)
  - o [获取运行配置和运行状态](#)
  - o [获取可配置的功能的变量值](#)
  - o [获取NETCONF信息](#)
  - o [获取YANG文件的内容](#)
  - o [获取NETCONF会话信息](#)
- (3) (可选) [过滤表和列的信息](#)
  - o [基于表的过滤](#)
  - o [基于列的过滤](#)
- (4) (可选) [配置加锁和解锁](#)
  - a. [配置加锁](#)
  - b. [配置解锁](#)
- (5) (可选) [下发配置](#)
- (6) (可选) [配置文件管理](#)
  - o [保存配置](#)
  - o [加载配置文件](#)
  - o [回滚配置](#)
- (7) (可选) [下发命令行](#)
- (8) (可选) [订阅事件](#)
  - o [订阅Syslog事件](#)
  - o [订阅监控事件](#)
  - o [订阅模块上报事件](#)
- (9) (可选) [关闭NETCONF会话](#)
- (10) (可选) [退出XML视图](#)

## 1.4 建立NETCONF会话

### 1.4.1 配置限制和指导

建立 NETCONF 会话后，客户端先与设备进行能力集交互，完成能力集的交互后，设备才会处理客户端发送的其他请求。

多个用户同时配置设备时，可能会导致用户配置与配置结果不一致，因此，请避免多个用户同时配置设备。

设备同一时间内允许建立的最大会话数可以通过 **aaa session-limit** 命令配置，关于该命令的详细描述，请参见“安全配置指导”中的“AAA”。用户数超过上限后，新登录的用户将登录失败。

## 1.4.2 配置NETCONF会话公共属性

### 1. 功能简介

NETCONF 支持共用命名空间和专用命名空间，两种类型的命名空间互不兼容。客户端与设备必须使用相同的命名空间才能建立会话。如果客户端不支持共用命名空间，则需要配置本功能，使设备和客户端建立会话时使用专用命名空间。

- 共用命名空间：各模块共用一个命名空间。缺省情况下，设备使用共用命名空间。使用共用命名空间的报文中，命名空间位于<top>元素中，模块位于<top>元素下，报文示例如下：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-bulk>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        <Ifmgr>
          <Interfaces>
          </Interfaces>
        </Ifmgr>
      </top>
    </filter>
  </get-bulk>
</rpc>
```

- 专用命名空间：每个模块单独使用一个命名空间。使用专用命名空间的报文中无<top>元素，命名空间位于模块名后，报文示例如下：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-bulk>
    <filter type="subtree">
      <Ifmgr xmlns="http://www.h3c.com/netconf/data:1.0-Ifmgr">
        <Interfaces>
        </Interfaces>
      </Ifmgr>
    </filter>
  </get-bulk>
</rpc>
```

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置 NETCONF 会话超时时间。

```
netconf { agent | soap } idle-timeout minute
```

属性	说明
<b>agent</b>	<b>agent</b> 表示如下NETCONF会话类型： <ul style="list-style-type: none"> <li>• NETCONF over SSH</li> <li>• NETCONF over Telnet</li> <li>• NETCONF over Console</li> </ul> 缺省超时时间为0（不超时）
<b>soap</b>	<b>soap</b> 表示如下NETCONF会话类型： <ul style="list-style-type: none"> <li>• NETCONF over SOAP over HTTP</li> <li>• NETCONF over SOAP over HTTPS</li> </ul> 缺省超时时间为10分钟

- (3) 开启 NETCONF 日志功能。

```
netconf log source { all | { agent | soap | web } * } { protocol-operation
{ all | { action | config | get | set | session | syntax | others } * } |
row-operation | verbose }
```

缺省情况下，NETCONF 日志功能处于关闭状态。

- (4) 配置设备使用专用命名空间。

```
netconf capability specific-namespace
```

缺省情况下，设备使用共用命名空间。

配置 NETCONF 使用专用命名空间后，需要重新建立 NETCONF 会话，使客户端和设备使用相同的命名空间。

### 1.4.3 建立NETCONF over SOAP会话

#### 1. 功能简介

NETCONF 支持封装成 SOAP 报文后通过 HTTP 或 HTTPS 协议传输，即 NETCONF over SOAP over HTTP 和 NETCONF over SOAP over HTTPS，使用该功能后，用户可以通过配置工具给设备下发 NETCONF 指令来实现对设备的访问。

#### 2. 配置限制和指导

用户可以配置在 SOAP 请求的<UserName>中携带认证域信息，该信息仅对当前请求生效。通过 **netconf soap domain** 命令配置强制认证域后，SOAP 请求中携带的认证域不生效。

#### 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启 NETCONF over SOAP 功能。

（非 FIPS 模式）

```
netconf soap { http | https } enable
```

（FIPS 模式）

```
netconf soap https enable
```

缺省情况下，NETCONF over SOAP 处于关闭状态。

- (3) 配置设备发送的 SOAP 报文的 DSCP 优先级。

（非 FIPS 模式）

```
netconf soap { http | https } dscp dscp-value
```

（FIPS 模式）

```
netconf soap https dscp dscp-value
```

缺省情况下，设备发送的 SOAP 报文的 DSCP 优先级为 0。

- (4) 配置 NETCONF over SOAP 关联 IPv4 ACL，只有 IPv4 ACL 允许通过的客户端可以与设备建立 NETCONF over SOAP 会话。

（非 FIPS 模式）

```
netconf soap { http | https } acl { ipv4-acl-number | name ipv4-acl-name }
```

（FIPS 模式）

```
netconf soap https acl { ipv4-acl-number | name ipv4-acl-name }
```

缺省情况下，未配置 NETCONF over SOAP 关联 IPv4 ACL。

- (5) 配置 NETCONF 用户的强制认证域。

```
netconf soap domain domain-name
```

缺省情况下，未配置 NETCONF 用户的强制认证域。

配置该功能后，所有用户都会使用强制认证域进行认证。关于认证域的详细介绍请参见“安全配置指导”中的“AAA”。

- (6) 通过配置工具与设备建立 NETCONF over SOAP 会话。关于配置工具的使用方法，具体参见配置工具的配置指导。

## 1.4.4 建立NETCONF over SSH会话

### 1. 配置限制和指导

进入 XML 视图后，用户需要严格按照 NETCONF 报文格式将报文拷贝、粘贴到 XML 视图中，请勿手工输入 NETCONF 报文。

在一次 NETCONF 操作完成前，请勿进行其他任何操作（例如粘贴 NETCONF 报文或输入回车），否则可能导致 NETCONF 配置失败。

在 XML 视图下进行 NETCONF 配置时，NETCONF 报文最后需要添加“]]>]]>”结束符，否则设备无法识别。本手册举例中，为方便识别 XML 格式，均未添加该结束符，实际操作中请自行添加。

### 2. 配置准备

通过命令行建立 NETCONF over SSH 会话前，请先通过 SSH 登录到设备。

### 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启 NETCONF over SSH。

```
netconf ssh server enable
```

缺省情况下，NETCONF over SSH 处于关闭状态。

(3) 配置 NETCONF over SSH 的监听端口。

```
netconf ssh server port port-number
```

缺省情况下，NETCONF over SSH 的监听端口为 830。

(4) 建立 NETCONF over SSH 会话。请选择其中一项进行配置。

- 通过配置工具与设备建立 NETCONF over SSH 会话。关于配置工具的使用方法，具体参见配置工具的配置指导。
- 请依次执行以下命令建立 NETCONF over SSH 会话。

```
quit
```

```
xml
```

进入 XML 视图，即表示成功建立 NETCONF over SSH 会话。

## 1.4.5 建立NETCONF over Telnet或NETCONF over Console会话

### 1. 配置限制和指导

进入 XML 视图后，用户需要严格按照 NETCONF 报文格式将报文拷贝、粘贴到 XML 视图中，请勿手工输入 NETCONF 报文。

在一次 NETCONF 操作完成前，请勿进行其他任何操作（例如粘贴 NETCONF 报文或输入回车），否则可能导致 NETCONF 配置失败。

在 XML 视图下进行 NETCONF 配置时，NETCONF 报文最后需要添加“]]>]]>”结束符，否则设备无法识别。本手册举例中，为方便识别 XML 格式，均未添加该结束符，实际操作中请自行添加。

### 2. 配置准备

建立 NETCONF over Telnet 或 NETCONF over Console 会话前，请先通过 Telnet 或 Console 口登录到设备。

### 3. 配置步骤

请在用户视图下执行本命令，进入 XML 视图。

```
xml
```

进入 XML 视图，即表示成功建立 NETCONF over Telnet 或 NETCONF over Console 会话。

## 1.4.6 交换能力集

### 1. 功能简介

建立 NETCONF 会话后，客户端和设备必须交换各自支持的能力集，双方收到对方的能力集后才可以进行下一步操作。

### 2. 设备发送给客户端的报文

建立 NETCONF 会话后，设备会发送如下报文自动告知客户端支持的 NETCONF 能力集：

```
<?xml version="1.0" encoding="UTF-8"?><hello  
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"><capabilities><capability>urn:ietf:params:netconf:base:1.1</capability><capability>urn:ietf:params:netconf:writable-running</capability><capability>urn:ietf:params:netconf:capability:notification:1.0</capability><cap
```

```
ability>urn:ietf:params:netconf:capability:validate:1.1</capability><capability>urn:ietf:params:netconf:capability:interleave:1.0</capability><capability>urn:h3c:params:netconf:capability:h3c-netconf-ext:1.0</capability></capabilities><session-id>1</session-id></hello>]]>]]>
```

<capabilities>和</capabilities>之间的内容表示设备支持的能力集，以具体设备实际情况为准。  
<session-id>和</session-id>之间的内容表示为本次会话分配的会话 ID，用来唯一标识本次会话。

3. 客户端发送给设备的报文

客户端收到设备发送的能力集协商报文后，需要给设备发送如下格式的报文，告知设备客户端支持哪些 NETCONF 能力集。

Hello 协商报文格式如下：

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      capability-set
    </capability>
  </capabilities>
</hello>
```

属性	说明
capability-set	客户端支持的能力集，由用户定义 一个<capability>和</capability>选项对中只能填写一个能力集，可以使用多个选项对，配置多个能力集

1.5 获取设备配置信息

1.5.1 配置限制和指导

用户执行<get>、<get-bulk>、<get-config>或<get-bulk-config>操作获取设备数据时，如果所获取数据中包含 NETCONF 不能识别的字符，则 NETCONF 将该字符转换为“?”输出到客户端。  
<get><netconf-state/></get>操作不支持数据过滤。  
关于获取设备配置信息操作的详细规则，请参见《NETCONF XML API 手册》。

1.5.2 获取运行配置和运行状态

设备支持通过<get>操作获取设备运行状态和运行配置。  
设备支持通过<get-bulk>操作从指定索引的下一条开始获取 N 条运行状态和运行配置（索引行数据不返回）。  
<get>操作会返回所有符合条件的数据，在某些情况下，会导致获取数据效率不高。<get-bulk>允许用户从固定数据项开始，向后获取指定条目的数据记录。

<get>和<get-bulk>报文的通用格式如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <getoperation>
    <filter>
```



```

    <top xmlns="http://www.h3c.com/netconf/data:1.0">
        指定模块，子模块，表名，列名
    </top>
</filter>
</getoperation>
</rpc>

```

属性	说明
<b>getoperation</b>	可以为get或者get-bulk
<b>filter</b>	<p>用于过滤信息，&lt;filter&gt;中可包括模块名、子模块名、表名和列名：</p> <ul style="list-style-type: none"> <li>如果不指定模块（子模块），则表示全部模块（子模块）。指定模块（子模块）时，则返回数据只包含指定模块（子模块）。</li> <li>如果模块下不指定表，则表示全部表。指定表时，则返回数据只包含指定表</li> <li>如果只指定索引列，则返回的数据包括全部的列。如果同时指定索引列之外的其他列，则返回的数据仅包含索引列和指定的列</li> </ul>

- <get-bulk>操作报文中可以携带 count 和 index 属性。

属性	说明
<b>index</b>	指定索引。如未指定索引，则以第一条操作为索引
<b>count</b>	<p>获取指定数量的运行状态和运行配置信息。count属性遵循如下约定：</p> <ul style="list-style-type: none"> <li>count 属性的位置在可以从 top 下的节点开始，到表节点为止，即：模块节点，表节点这几个位置都能放置 count 属性，其他位置的 count 将不被解释</li> <li>如果 count 放在模块节点上，则报文中指定的子孙节点(表)中没有 count 的都默认 count 属性的值和模块一致</li> <li>如果不指定 count，或者数据表中符合条件的数据记录小于 count，则获取从指定索引开始的所有数据</li> </ul>

如下为一个携带了 count 和 index 属性的<get-bulk>操作报文示例：

```

<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:xc="http://www.h3c.com/netconf/base:1.0">
    <get-bulk>
        <filter type="subtree">
            <top xmlns="http://www.h3c.com/netconf/data:1.0"
xmlns:base="http://www.h3c.com/netconf/base:1.0">
                <Syslog>
                    <Logs xc:count="5">
                        <Log>
                            <Index>10</Index>
                        </Log>
                    </Logs>
                </Syslog>
            </top>
        </filter>
    </get-bulk>
</rpc>

```

- ```

    </get-bulk>
</rpc>

```
- 获取接口数据时,如果 **IfIndex** 元素的值为数字,设备无法识别该值为名称类型还是索引类型。此时,用户可以使用 **valuetype** 指定该值的类型。**valuetype** 取值为:

| 属性           | 说明                                                                                  |
|--------------|-------------------------------------------------------------------------------------|
| <b>name</b>  | 值为名称类型                                                                              |
| <b>index</b> | 值为索引类型                                                                              |
| <b>auto</b>  | 设备先按名称类型进行匹配,如果没有匹配到任何信息,再按照索引类型进行匹配<br>如果不指定 <b>valuetype</b> 的属性,缺省使用 <b>auto</b> |

下面以 **IfIndex** 元素值为 **index** 类型,值为 1 为例:

```

<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <getoperation>
    <filter>
      <top xmlns="http://www.h3c.com/netconf/config:1.0"
xmlns:base="http://www.h3c.com/netconf/base:1.0">
        <VLAN>
          <TrunkInterfaces>
            <Interface>
              <IfIndex base:valuetype="index">1</IfIndex>
            </Interface>
          </TrunkInterfaces>
        </VLAN>
      </top>
    </filter >
  </getoperation>
</rpc>

```

设备收到配置获取请求报文后会将相应属性的值通过如下报文反馈给客户端:

```

<?xml version="1.0"?>
<rpc-reply message-id="100"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    全部配置数据和状态数据
  </data>
</rpc-reply>

```

### 1.5.3 获取可配置的功能的变量值

设备支持通过<get-config>和<get-bulk-config>操作获取设备所有可配置的功能的变量值。

<get-config>和<get-bulk-config>操作报文中可以包含子标签<filter>,用来对要获取的信息进行过滤。

<get-config>和<get-bulk-config>的通用报文格式为:

```

<?xml version="1.0"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>

```

```
<source>
  <running/>
</source>
<filter>
  <top xmlns="http://www.h3c.com/netconf/config:1.0">
    指定模块，子模块，表名，列名
  </top>
</filter>
</get-config>
</rpc>
```

设备收到配置获取请求报文后会将相应配置通过如下报文反馈给客户端：

```
<?xml version="1.0"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    所有指定filter 内的数据
  </data>
</rpc-reply>
```

1.5.4 获取NETCONF信息

<get><netconf-state/></get>报文的通用格式为：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="m-641" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter type='subtree'>
      <netconf-state xmlns='urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring'>
        <getType/>
      </netconf-state>
    </filter>
  </get>
</rpc>
```

其中，*getType* 可以为 **capabilities**、**datastores**、**schemas**、**sessions** 或者 **statistics**：

- 不指定 *getType* 时，该操作则获取 NETCONF 全部信息。
- 当指定 *getType* 时，该操作仅返回相应类型的应答数据。其中，*getType* 取值为：

属性	说明
<b>capabilities</b>	该操作用来获取设备能力集
<b>datastores</b>	该操作用来获取设备中的数据库
<b>schemas</b>	该操作用来获取设备中的YANG文件名称列表
<b>sessions</b>	该操作用来获取设备中的会话信息
<b>statistics</b>	该操作用来获取NETCONF的统计信息

# 设备收到 NETCONF 信息获取请求报文后，将相应属性的值通过如下报文反馈给客户端：

```
<?xml version="1.0"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
```

```

<data>
    全部 NETCONF 全部相关信息
</data>
</rpc-reply>

```

### 1.5.5 获取YANG文件的内容

YANG 文件中保存了设备支持的 NETCONF 操作，用户通过获取、并分析 YANG 文件的内容，从而获知设备支持哪些 NETCONF 操作。

YANG文件集成在设备软件中，命名格式为`yang_identifier@yang_version.yang`，用户不能通过`dir`命令显示YANG文件名，关于如何获取设备上YANG文件名称，详细请参见“[1.5.4 获取NETCONF信息](#)”。

如下报文要求返回名称为 `syslog-data@2017-01-01.yang` 的 YANG 文件中的数据：

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get-schema xmlns='urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring'>
<identifier>syslog-data</identifier>
<version>2017-01-01</version>
<format>yang</format>
</get-schema>
</rpc>

```

设备收到 YANG 文件获取请求报文后，将相应属性的值通过如下报文反馈给客户端：

```

<?xml version="1.0"?>
<rpc-reply message-id="100"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data>
        指定 YANG 文件的内容
    </data>
</rpc-reply>

```

### 1.5.6 获取NETCONF会话信息

使用该功能用户可以获取当前设备的所有 NETCONF 会话信息。

获取 NETCONF 会话信息的格式为：

```

<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <get-sessions/>
</rpc>

```

设备收到命令行指令后会回应客户端，当客户端收到如下报文时，表示命令行执行成功：

```

<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <get-sessions>
        <Session>
            <SessionID>用户会话 ID 信息</SessionID>
            <Line>line 信息</Line>
            <UserName>用户登录名称</UserName>
            <Since>用户登录时间</Since>
            <LockHeld>用户是否持有锁</LockHeld>
        </Session>
    </get-sessions>
</rpc-reply>

```

```

    </Session>
  </get-sessions>
</rpc-reply>

```

## 1.5.7 获取运行配置举例

### 1. 组网需求

获取接口表的一条数据

### 2. 配置步骤

# 进入 XML 视图。

```
<Sysname> xml
```

# 进行能力交换。

请将以下报文拷贝、粘贴到客户端：

```

<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
  </capabilities>
</hello>

```

# 获取接口表的一条数据。

请将以下报文拷贝、粘贴到客户端：

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:web="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-bulk>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0"
xmlns:web="http://www.h3c.com/netconf/base:1.0">
        <Ifmgr>
          <Interfaces web:count="1">
        </Interfaces>
        </Ifmgr>
      </top>
    </filter>
  </get-bulk>
</rpc>

```

### 3. 验证配置

# 如果客户端收到类似如下的报文，则表示操作成功：

```

<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:web="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="100">
  <data>
    <top xmlns="http://www.h3c.com/netconf/data:1.0">
      <Ifmgr>
        <Interfaces>
          <Interface>
            <IfIndex>3</IfIndex>
            <Name>GigabitEthernet1/0/2</Name>
            <AbbreviatedName>GE1/0/2</AbbreviatedName>

```

```

        <PortIndex>3</PortIndex>
        <ifTypeExt>22</ifTypeExt>
        <ifType>6</ifType>
        <Description>GigabitEthernet1/0/2 Interface</Description>
        <AdminStatus>2</AdminStatus>
        <OperStatus>2</OperStatus>
        <ConfigSpeed>0</ConfigSpeed>
        <ActualSpeed>100000</ActualSpeed>
        <ConfigDuplex>3</ConfigDuplex>
        <ActualDuplex>1</ActualDuplex>
    </Interface>
</Interfaces>
</Ifmgr>
</top>
</data>
</rpc-reply>

```

## 1.5.8 获取可配置的功能的变量值配置举例

### 1. 组网需求

获取所有可配置的功能的变量值。

### 2. 配置步骤

# 进入 XML 视图。

```
<Sysname> xml
```

# 进行能力交换。

请将以下报文拷贝、粘贴到客户端：

```

<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
</hello>

```

# 获取模块所有配置数据。

请将以下报文拷贝、粘贴到客户端：

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>
    <source>
      <running/>
    </source>
  </get-config>
</rpc>

```

### 3. 验证配置

# 如果客户端收到类似如下的报文，则表示操作成功：

```

<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:web="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="100">
  <data>
    <top xmlns="http://www.h3c.com/netconf/config:1.0">
      <Ifmgr>
        <Interfaces>
          <Interface>
            <IfIndex>1307</IfIndex>
            <Shutdown>1</Shutdown>
          </Interface>
          <Interface>
            <IfIndex>1308</IfIndex>
            <Shutdown>1</Shutdown>
          </Interface>
          <Interface>
            <IfIndex>1309</IfIndex>
            <Shutdown>1</Shutdown>
          </Interface>
          <Interface>
            <IfIndex>1311</IfIndex>
            <VlanType>2</VlanType>
          </Interface>
          <Interface>
            <IfIndex>1313</IfIndex>
            <VlanType>2</VlanType>
          </Interface>
        </Interfaces>
      </Ifmgr>
      <Syslog>
        <LogBuffer>
          <BufferSize>120</BufferSize>
        </LogBuffer>
      </Syslog>
      <System>
        <Device>
          <SysName>Sysname</SysName>
          <TimeZone>
            <Zone>+11:44</Zone>
            <ZoneName>ABC</ZoneName>
          </TimeZone>
        </Device>
      </System>
    </top>
  </data>
</rpc-reply>

```

## 1.5.9 获取Syslog功能的变量值配置举例

### 1. 组网需求

获取 Syslog 模块的所有配置数据

### 2. 配置步骤

# 进入 XML 视图。

```
<Sysname> xml
```

# 进行能力交换。

请将以下报文拷贝、粘贴到客户端：

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
</hello>
```

# 获取 Syslog 模块的所有配置数据。

请将以下报文拷贝、粘贴到客户端：

```
<rpc message-id="100"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>
    <source>
      <running/>
    </source>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
        <Syslog/>
      </top>
    </filter>
  </get-config>
</rpc>
```

### 3. 验证配置

# 如果客户端收到类似如下的报文，则表示操作成功：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="100">
  <data>
    <top xmlns="http://www.h3c.com/netconf/config:1.0">
      <Syslog>
        <LogBuffer>
          <BufferSize>120</BufferSize>
        </LogBuffer>
      </Syslog>
    </top>
  </data>
</rpc-reply>
```



## 1.5.10 获取NETCONF会话信息配置举例

### 1. 配置需求

获取会话信息。

### 2. 配置步骤

# 进入 XML 视图。

```
<Sysname> xml
```

# 进行能力交换。

请将以下报文拷贝、粘贴到客户端：

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
</hello>
```

# 获取设备上当前存在的 NETCONF 会话的信息。

请将以下报文拷贝、粘贴到客户端：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-sessions/>
</rpc>
```

### 3. 验证配置

# 如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="100">
  <get-sessions>
    <Session>
      <SessionID>1</SessionID>
      <Line>vty0</Line>
      <UserName></UserName>
      <Since>2017-01-07T00:24:57</Since>
      <LockHeld>false</LockHeld>
    </Session>
  </get-sessions>
</rpc-reply>
```

以上信息表明：目前有一个 NETCONF 连接，SessionID 是 1，登录用户类型 vty0，登录时间是 2011-01-05T00:24:57，此用户不持有锁。

## 1.6 过滤表和列的信息

### 1.6.1 功能简介

当用户执行<get>、<get-bulk>、<get-config>或者<get-bulk-config>操作时，在 XML 语言中增加过滤条件，可以使用户只看到自己关心的数据。数据过滤包括基于表的过滤和基于列的过滤：

- 基于表的过滤用于过滤表的信息。
- 基于列的过滤用于过滤单个列的信息。

### 1.6.2 配置限制和指导

基于表的过滤需要在基于列的过滤之前配置。

### 1.6.3 基于表的过滤

#### 1. 功能简介

基于表的过滤中，过滤条件类似命令行，命令空间为 <http://www.h3c.com/netconf/base:1.0>，属性名为 **filter**。关于支持该过滤方式表的详细信息，具体请参见模块的 XML API 手册。

如下要求返回 IPv4 路由表中匹配 IP 地址为 1.1.1.0 和 24 位掩码最长的数据：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:h3c="http://www.h3c.com/netconf/base:1.0">
  <get>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        <Route>
          <Ipv4Routes>
            <RouteEntry h3c:filter="IP 1.1.1.0 MaskLen 24 longer"/>
          </Ipv4Routes>
        </Route>
      </top>
    </filter>
  </get>
</rpc>
```

#### 2. 配置限制和指导

配置基于表的过滤时，查询条件必须写在行的属性上。

### 1.6.4 基于列的过滤

#### 1. 功能简介

基于列的过滤包括严格匹配过滤、正则表达式匹配过滤和条件匹配过滤。

同时指定多种过滤时，只有一个生效，其优先级从高到低依次为：严格匹配过滤、正则表达式匹配过滤和条件匹配过滤。

#### 2. 严格匹配过滤

严格匹配包括两种匹配方式，一种是元素值方式，还有一种是属性名方式。

用户在 XML 语言中直接指定对应的元素值，设备将对这些值进行严格匹配。如果指定了多个元素的值，则返回同时符合这几个条件的数据。只有完全匹配条件才返回成功。

如下为一个 NETCONF 报文示例，用于获取所有状态为 UP 的接口的配置信息：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        <Ifmgr>
          <Interfaces>
            <Interface>
              <AdminStatus>1</AdminStatus>
            </Interface>
          </Interfaces>
        </Ifmgr>
      </top>
    </filter>
  </get>
</rpc>
```

当用户在行的位置上放置的某个属性的名称和当前表的某个列的名称一致，则这个属性的值将与用户下发配置中同名称的列的值进行严格匹配，例如，上面例子用属性名方式的等价 XML 请求为：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0"
xmlns:data="http://www.h3c.com/netconf/data:1.0">
        <Ifmgr>
          <Interfaces>
            <Interface data:AdminStatus="1" />
          </Interfaces>
        </Ifmgr>
      </top>
    </filter>
  </get>
</rpc>
```

以上两个 NETCONF 报文示例说明通过不同的严格匹配方式都可以获取所有状态为 UP 的接口的索引和指定列信息。

### 3. 正则表达式匹配过滤

当过滤条件比较复杂时，可以在指定元素上设置 **regExp** 属性为一个正则表达式，以完成过滤的目的。

支持正则表达式匹配的数据类型有整数、日期和时间、字符串、IPv4 地址、IPv4 掩码、IPv6 地址、MAC 地址、OID 和时区。

如下为一个 NETCONF 报文示例，用于获取接口的描述信息，并要求这些描述信息全部为大写字母，不能有其他字符。

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:h3c="http://www.h3c.com/netconf/base:1.0">
```

```

<get-config>
  <source>
    <running/>
  </source>
  <filter type="subtree">
    <top xmlns="http://www.h3c.com/netconf/config:1.0">
      <Ifmgr>
        <Interfaces>
          <Interface>
            <Description h3c:regExp="[A-Z]*$"/>
          </Interface>
        </Interfaces>
      </Ifmgr>
    </top>
  </filter>
</get-config>
</rpc>

```

#### 4. 条件匹配过滤

由于正则表达仅能够完成字符匹配，对于数值逻辑的判断过滤实现起来比较麻烦，此时，可使用条件匹配过滤功能。

条件匹配通过在元素中增加 **match** 属性完成，属性的值（即过滤条件）可以为数字、字符串。

表1-2 条件匹配命令

操作	命令	说明
大于	<b>match="more:value"</b>	值大于value，支持的数据类型为：日期、数字、字符串
小于	<b>match="less:value"</b>	值小于value，支持的数据类型为：日期、数字、字符串
不小于	<b>match="notLess:value"</b>	值不小于value，支持的数据类型为：日期、数字、字符串
不大于	<b>match="notMore:value"</b>	值不大于value，支持的数据类型为：日期、数字、字符串
等于	<b>match="equal:value"</b>	值等于value，支持的数据类型为：日期、数字、字符串、OID、BOOL
不等于	<b>match="notEqual:value"</b>	值不等于value，支持的数据类型为：日期、数字、字符串、OID、BOOL
包含	<b>match="include:string"</b>	包含字符串string，支持的数据类型为：字符串
不包含	<b>match="exclude:string"</b>	不能包含字符串string，支持的数据类型为：字符串
开始于	<b>match="startWith:string"</b>	以字符串string开头，支持的数据类型为：字符串、OID
结束于	<b>match="endWith:string"</b>	以字符串string结束，支持的数据类型为：字符串

如下为一个 NETCONF 报文示例，用于获取实体扩展信息中 CPU 利用率大于 50%的实体。

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:h3c="http://www.h3c.com/netconf/base:1.0">

```

```

<get>
  <filter type="subtree">
    <top xmlns="http://www.h3c.com/netconf/data:1.0">
      <Device>
        <ExtPhysicalEntities>
          <Entity>
            <CpuUsage h3c:match="more:50"></CpuUsage>
          </Entity>
        </ExtPhysicalEntities>
      </Device>
    </top>
  </filter>
</get>
</rpc>

```

## 1.6.5 正则表达式过滤配置举例

### 1. 组网需求

取 Ifmgr 模块下 Interfaces 表下 Description 列包含 Gigabit 的所有数据。

### 2. 配置步骤

# 进入 XML 视图。

```
<Sysname> xml
```

# 进行能力交换。

请将以下报文拷贝、粘贴到客户端：

```

<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
</hello>

```

# 取 Ifmgr 模块 Interfaces 表下 Description 列包含 Gigabit 的所有数据。

请将以下报文拷贝、粘贴到客户端：

```

<?xml version="1.0"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:h3c="http://www.h3c.com/netconf/base:1.0">
  <get>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        <Ifmgr>
          <Interfaces>
            <Interface>
              <Description h3c:regExp="(Gigabit)+" />
            </Interface>
          </Interfaces>
        </Ifmgr>
      </top>
    </filter>
  </get>
</rpc>

```

```

    </top>
  </filter>
</get>
</rpc>

```

### 3. 验证配置

# 如果客户端收到类似如下的报文，则表示操作成功。

```

<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:h3c="http://www.h3c.com/netconf/base:1.0" message-id="100">
  <data>
    <top xmlns="http://www.h3c.com/netconf/data:1.0">
      <Ifmgr>
        <Interfaces>
          <Interface>
            <IfIndex>2681</IfIndex>
            <Description>GigabitEthernet1/0/1 Interface</Description>
          </Interface>
          <Interface>
            <IfIndex>2685</IfIndex>
            <Description>GigabitEthernet1/0/2 Interface</Description>
          </Interface>
          <Interface>
            <IfIndex>2689</IfIndex>
            <Description>GigabitEthernet1/0/3 Interface</Description>
          </Interface>
        </Interfaces>
      </Ifmgr>
    </top>
  </data>
</rpc-reply>

```

## 1.6.6 条件匹配过滤配置举例

### 1. 组网需求

取 Ifmgr 模块 Interfaces 表下 ifindex 值大于等于 5000 的 Name 列信息。

### 2. 配置步骤

# 进入 XML 视图。

```
<Sysname> xml
```

# 进行能力交换。

请将以下报文拷贝、粘贴到客户端：

```

<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>

```

```
</hello>
```

# 取 Ifmgr 模块 Interfaces 表下 ifindex 值大于等于 5000 的 Name 列信息。

请将以下报文拷贝、粘贴到客户端：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:h3c="http://www.h3c.com/netconf/base:1.0">
  <get>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        <Ifmgr>
          <Interfaces>
            <Interface>
              <IfIndex h3c:match="notLess:5000"/>
              <Name/>
            </Interface>
          </Interfaces>
        </Ifmgr>
      </top>
    </filter>
  </get>
</rpc>
```

### 3. 验证配置

# 如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:h3c="http://www.h3c.com/netconf/base:1.0" message-id="100">
  <data>
    <top xmlns="http://www.h3c.com/netconf/data:1.0">
      <Ifmgr>
        <Interfaces>
          <Interface>
            <IfIndex>7241</IfIndex>
            <Name>NULL0</Name>
          </Interface>
        </Interfaces>
      </Ifmgr>
    </top>
  </data>
</rpc-reply>
```

## 1.7 配置加锁和解锁

### 1.7.1 功能简介

设备支持通过 NETCONF、CLI 和 SNMP 等多种方式配置，当用户管理、维护设备或者定位网络问题时，为防止其他用户修改当前配置、引入干扰，可以使用本特性给当前配置加锁。给当前配置加锁后，只有持有锁的用户可以修改设备的当前配置，其他用户只能执行读取操作，不能进行配置。

只有持有锁的用户可以解锁，解锁后其他用户才可以修改设备的当前配置或另外加锁。如果持有锁的用户的当前连接断开，系统会自动解锁。

### 1.7.2 配置限制和指导

NETCONF 加锁操作将锁定配置数据，即<edit-config>操作中指定的配置数据，其他操作（例如<get>操作）不受锁的限制。

目前设备只支持对当前配置加锁，不能对具体的功能模块进行加锁。

只有持有锁的用户才能解锁，其他用户不能解锁。

### 1.7.3 配置加锁

请将以下报文拷贝、粘贴到客户端，用户即能完成加锁操作：

```
<?xml version="1.0" encoding="UTF-8"?>
  <rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <lock>
      <target>
        <running/>
      </target>
    </lock>
  </rpc>
```

设备收到加锁报文后会回应客户端，当客户端收到如下报文时，表示加锁成功：

```
<?xml version="1.0" encoding="UTF-8"?>
  <rpc-reply message-id="100"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <ok/>
  </rpc-reply>
```

### 1.7.4 配置解锁

请将以下报文拷贝、粘贴到客户端，用户即能完成解锁操作：

```
<?xml version="1.0" encoding="UTF-8"?>
  <rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <unlock>
      <target>
        <running/>
      </target>
    </unlock>
  </rpc>
```

设备收到解锁报文后会回应客户端，当客户端收到如下报文时，表示解锁成功：

```
<?xml version="1.0" encoding="UTF-8"?>
  <rpc-reply message-id="100"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <ok/>
  </rpc-reply>
```



## 1.7.5 加锁配置举例

### 1. 组网需求

给设备加锁，以免其他用户使用 XML 语言修改设备的当前配置。

### 2. 配置步骤

# 进入 XML 视图。

```
<Sysname> xml
```

# 进行能力交换。

请将以下报文拷贝、粘贴到客户端：

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
</hello>
```

# 对当前配置加锁。

请将以下报文拷贝、粘贴到客户端：

```
<?xml version="1.0" encoding="UTF-8"?>
  <rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <lock>
      <target>
        <running/>
      </target>
    </lock>
  </rpc>
```

### 3. 验证配置

# 如果客户端收到如下报文，则表示加锁成功：

```
<?xml version="1.0" encoding="UTF-8"?>
  <rpc-reply message-id="100"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <ok/>
  </rpc-reply>
```

用户加锁成功后，另一客户端发送加锁报文，设备会返回如下报文：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<rpc-error>
  <error-type>protocol</error-type>
  <error-tag>lock-denied</error-tag>
  <error-severity>error</error-severity>
  <error-message xml:lang="en">Lock failed because the NETCONF lock is held by another
session.</error-message>
  <error-info>
    <session-id>1</session-id>
  </error-info>
```

```
</rpc-error>
</rpc-reply>
```

以上报文表明加锁失败，session-id 是 1 的用户已经持有锁。

## 1.8 下发配置

### 1.8.1 功能简介

设备支持通过<edit-config>操作下发配置，该操作支持如下配置选项：merge、create、replace、remove、delete、缺省操作选项、缺省错误处理选项、测试处理、增量下发，关于这些选项的详细描述请参见“[2 Comware V7 支持的NETCONF操作](#)”。

### 1.8.2 配置步骤

当需要向设备下发配置时，请使用如下格式的 NETCONF 报文：

```
<?xml version="1.0"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target><running></running></target>
  <error-option>
    error-option
  </error-option>
  <config>
    <top xmlns="http://www.h3c.com/netconf/config:1.0">
      指定模块名，子模块名，列名，表名
    </top>
  </config>
</edit-config>
</rpc>
```

error-option 为下发配置过程中遇到错误时的处理方式，取值为：

属性	说明
stop-on-error	停止下发配置
continue-on-error	继续下发配置
rollback-on-error	回滚到该<edit-config>操作前的配置 配置回滚时缺省不能下发<edit-config>操作。因为回滚到该<edit-config>操作前的配置时，如果回滚时间超过客户端等待时间，客户端会认为<edit-config>操作下发失败，此时客户端将重新下发<edit-config>操作，设备再次触发回滚，如此反复，则会导致CPU/内存资源耗尽，设备重启 如需回滚配置的同时执行<edit-config>操作，请先执行<action>操作将DisableEditConfigWhenRollback的属性修改为false

设备收到 edit-config 请求后会回应客户端，当客户端收到如下报文时，表示设置成功：

```
<?xml version="1.0">
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
```

```
</rpc-reply>
```

用户还可以通过<get>操作可以查看属性的当前值是否和 edit-config 操作设置的值一致。

### 1.8.3 下发配置举例

#### 1. 组网需求

修改 syslog 模块中的日志缓冲区可存储的信息条数为 512。

#### 2. 配置步骤

# 进入 XML 视图。

```
<Sysname> xml
```

# 进行能力交换。

请将以下报文拷贝、粘贴到客户端：

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
  </capabilities>
</hello>
```

# 把 Syslog 模块 LogBuffer 表中的 BufferSize 列，改成 512。

请将以下报文拷贝、粘贴到客户端：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:web="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <config>
      <top xmlns="http://www.h3c.com/netconf/config:1.0" web:operation="merge">
        <Syslog>
          <LogBuffer>
            <BufferSize>512</BufferSize>
          </LogBuffer>
        </Syslog>
      </top>
    </config>
  </edit-config>
</rpc>
```

#### 3. 验证配置

# 如果客户端收到类似如下的报文，则表示操作成功：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

1.9 保存配置

1.9.1 功能简介

用户可以执行保存配置操作将当前配置保存到配置文件中，同时该文件将自动作为下次启动配置文件。

1.9.2 配置限制和指导

保存配置操作消耗系统资源较多，因此，请避免在系统资源占用较大的情况下执行这些操作。

1.9.3 配置步骤

请将以下报文拷贝、粘贴到客户端：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save OverWrite="false">
    <file>配置文件的名称</file>
  </save>
</rpc>
```

属性	说明
file	文本配置文件的名称，该名称必须以存储介质的名称开头，后缀为.cfg。当报文中存在file属性时，必须输入文本配置文件的名称，不能为空 设备会将当前配置同时保存到文本配置文件（后缀为“.cfg”）和对应的二进制配置文件（后缀为“.mdb”，仅软件能解析） <ul style="list-style-type: none"><li>如果 file 属性指定的文件不存在，则自动创建该文本配置文件和对应的二进制配置文件</li><li>未指定 file 属性时，设备自动将当前配置保存到下次启动配置文件（包括文本配置文件和对应的二进制配置文件）</li></ul>
OverWrite	指定的配置文件名与原有配置文件名相同时，是否覆盖原配置文件。取值为： <ul style="list-style-type: none"><li>true: 当指定的配置文件名与原有配置文件名相同时，覆盖原配置文件</li><li>false: 当指定的配置文件名与原有配置文件名相同时，当前配置保存失败，同时返回错误提示信息</li></ul> 缺省情况下，OverWrite属性取值为true

设备收到配置保存请求后会回应客户端，当客户端收到如下报文时，表示保存成功：

```
<?xml version="1.0" encoding="UTF-8"?>
  <rpc-reply message-id="100"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <ok/>
  </rpc-reply>
```

1.9.4 保存配置举例

1. 组网需求

将设备的当前配置保存到配置文件 my\_config.cfg。

## 2. 配置步骤

# 进入 XML 视图。

```
<Sysname> xml
```

# 进行能力交换。

请将以下报文拷贝、粘贴到客户端：

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
</hello>
```

# 将设备的当前配置保存到配置文件 config.cfg。

请将以下报文拷贝、粘贴到客户端：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save>
    <file>config.cfg</file>
  </save>
</rpc>
```

## 3. 验证配置

# 如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8"?>
  <rpc-reply message-id="100"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <ok/>
  </rpc-reply>
```

## 1.10 加载配置文件

### 1.10.1 功能简介

用户可以执行<load>操作，将指定配置文件中的配置会被合并到设备的当前配置中。

设备将配置文件中的配置中的配置与当前配置进行比较：

- 对于配置文件中，有，但当前配置中没有的配置，则直接运行。
- 对于配置文件与当前配置中不一致的配置，则用配置文件中的配置替换当前配置中的对应配置。

### 1.10.2 配置限制和指导

加载配置文件操作消耗系统资源较多，因此，请避免在系统资源占用较大的情况下执行这些操作。

如果只有删除设备中现有的配置才能使配置文件中的某些配置生效，则需要先删除当前配置后再加载配置文件，否则这些配置无法生效。

### 1.10.3 配置步骤

请将以下报文拷贝、粘贴到客户端，用户即可将配置文件中的配置追加到当前配置中：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <load>
    <file>配置文件的名称</file>
  </load>
</rpc>
```

其中，“配置文件的名称”必须以存储介质的名称开头，后缀为.cfg。

设备收到配置加载请求后会回应客户端，当客户端收到如下报文时，表示配置加载成功：

```
<?xml version="1.0" encoding="UTF-8"?>
  <rpc-reply message-id="100"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <ok/>
  </rpc-reply>
```

## 1.11 回滚配置

### 1.11.1 配置限制和指导

回滚配置操作消耗系统资源较多，因此，请避免在系统资源占用较大的情况下执行这些操作。

缺省情况下，通过 NETCONF 回滚配置时，不能执行<edit-config>操作。如果需要在回滚配置的同时执行<edit-config>操作，请先执行<action>操作，将 **DisableEditConfigWhenRollback** 的属性修改为 **false**。

### 1.11.2 回滚配置文件

请将以下报文拷贝、粘贴到客户端，用户即可将设备的当前配置恢复到指定配置文件中的配置：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <rollback>
    <file>配置文件的名称</file>
  </rollback>
</rpc>
```

设备收到配置回滚请求后会回应客户端，当客户端收到如下报文时，表示配置回滚成功：

```
<?xml version="1.0" encoding="UTF-8"?>
  <rpc-reply message-id="100"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <ok/>
  </rpc-reply>
```

### 1.11.3 回滚运行配置

#### 1. 功能简介

设备支持对当前正在运行的配置进行回滚，配置回滚点后，设备可以在如下场景下进行配置回滚：

- NETCONF 客户端主动下发配置回滚指令。
- NETCONF 客户端在指定的时间内无任何指令，即 NETCONF 会话空闲时间超过配置的回滚空闲超时时间。
- NETCONF 客户端和设备间的连接异常断开。

### 2. 配置限制和指导

如果存在多个 NETCONF 会话同时配置设备，建议回滚配置前先用<lock>锁定系统，否则可能导致设备运行配置与回滚前配置不一致。

### 3. 配置步骤

- (1) 对当前系统加锁，具体请参见“[1.7 配置加锁和解锁](#)”。
- (2) 启动配置回滚功能，下发<save-point>/<begin>操作配置起始回滚点。

请将以下报文拷贝、粘贴到客户端：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <begin>
      <confirm-timeout>100</confirm-timeout>
    </begin>
  </save-point>
</rpc>
```

属性	说明
<b>confirm-timeout</b>	回滚空闲超时时间，取值范围为1～65535，单位为秒，缺省值为600。该属性为可选

当客户端收到如下报文时，表示配置起始回滚点成功：

```
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <save-point>
      <commit>
        <commit-id>1</commit-id>
      </commit>
    </save-point>
  </data>
</rpc-reply>
```

- (3) 下发<edit-config>操作，对设备各功能进行配置。
- (4) 下发<save-point>/<commit>操作配置回滚点。

配置<save-point>操作后，系统支持对最近 50 个回滚点的配置回滚。超过 50 个回滚点后，进行<commit>操作时需指定 **force** 属性强制覆盖最早的回滚点。

请将以下报文拷贝、粘贴到客户端：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <commit>
      <label>SUPPORT VLAN</label>
      <comment>vlan 1 to 100 and interfaces. Each vlan used for different custom as follows: .....</comment>
```

```

</commit>
  </save-point>
</rpc>

```

<label/>和<comment/>为可选。

当客户端收到如下报文时，表示确认成功：

```

<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <save-point>
      <commit>
        <commit-id>2</commit-id>
      </commit>
    </save-point>
  </data>
</rpc-reply>

```

- (5) 下发<save-point>/<get-commits>操作获取<commit>的操作记录。

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <get-commits>
      <commit-id/>
      <commit-index/>
      <commit-label/>
    </get-commits>
  </save-point>
</rpc>

```

<commit-id>，<commit-index>，<commit-label>任选一个，也可以都不选，显示所有的commit记录。

请将以下报文拷贝、粘贴到客户端：

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <get-commits>
      <commit-label>SUPPORT VLAN</commit-label>
    </get-commits>
  </save-point>
</rpc>

```

当客户端收到如下报文时，返回对应的commit操作记录：

```

<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <save-point>
      <commit-information>
        <CommitID>2</CommitID>
        <TimeStamp>Sun Jan 1 11:30:28 2017</TimeStamp>
        <UserName>test</UserName>
        <Label>SUPPORT VLAN</Label>
      </commit-information>
    </save-point>
  </data>
</rpc-reply>

```

- (6) 下发<save-point>/<get-commit-information>操作获取对应的commit操作时的配置。



```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <get-commit-information>
      <commit-information>
        <commit-id/>
        <commit-index/>
        <commit-label/>
      </commit-information>
      <compare-information>
        <commit-id/>
        <commit-index/>
        <commit-label/>
      </compare-information>
    </get-commit-information>
  </save-point>
</rpc>

```

<commit-id/>, <commit-index/>, <commit-label/>任选一个。<compare-information/>可选。

属性	说明
<b>commit-id</b>	系统唯一标识的commit编号
<b>commit-index</b>	最近50次的commit, 0表示最近一次下发commit, 49表示最远一次commit
<b>commit-label</b>	回滚点的标签, 不同回滚点的标签不能相同
<b>get-commit-information</b>	显示最近的使用的commit配置

请将以下报文拷贝、粘贴到客户端:

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <get-commit-information>
      <commit-information>
        <commit-label>SUPPORT VLAN</commit-label>
      </commit-information>
    </get-commit-information>
  </save-point>
</rpc>

```

当客户端收到如下报文时, 显示对应的配置信息:

```

<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <save-point>
      <commit-information>
        <content>
          ...
interface vlan 1
...
        </content>
      </commit-information>
    </save-point>
  </data>
</rpc-reply>

```

```
</data>
</rpc-reply>
```

- (7) 下发<save-point>/<rollback>进行回滚配置操作。用户可选择对应的回滚点进行配置回滚，或者等待 NETCONF 会话空闲时间超过配置的回滚空闲超时时间而自动进行配置回滚到最近回滚点的配置。

请将以下报文拷贝、粘贴到客户端：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <rollback>
      <commit-id/>
      <commit-index/>
      <commit-label/>
    </rollback>
  </save-point>
</rpc>
```

<commit-id/>、<commit-index/>、<commit-label/>任选一个，或者不选时回滚此前最近 commit 的配置。

属性	说明
<b>commit-id</b>	系统唯一标识的commit编号
<b>commit-index</b>	最近50次的commit，0表示最近一次下发commit，49表示最远一次commit
<b>commit-label</b>	回滚点的标签，不同回滚点的标签不能相同

当客户端收到如下报文时，表示主动下发配置回滚成功：

```
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok></ok>
</rpc-reply>
```

- (8) 下发<save-point>/<end>结束配置回滚或取消回滚配置。

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <end/>
  </save-point>
</rpc>
```

请将以下报文拷贝、粘贴到客户端：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <end/>
  </save-point>
</rpc>
```

当客户端收到如下报文时，返回对应的 end 结果成功。

```
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

- (9) 对当前系统解锁，具体请参见“[1.7 配置加锁和解锁](#)”。

## 1.12 下发命令行

### 1.12.1 功能简介

通过 NETCONF 功能，用户可以将命令行封装在 XML 报文中对设备进行操作。

### 1.12.2 配置限制和指导

通过 NETCONF 下发命令行操作消耗系统资源较多，因此，请避免一次下发多个命令行、或多个用户同时进行 NETCONF 下发命令行操作。

### 1.12.3 命令行操作

当需要给设备发送命令时，请使用格式如下的 NETCONF 报文：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <CLI>
    <Execution>
      命令行
    </Execution>
  </CLI>
</rpc>
```

一对<Execution>子标签中可以包含多个命令行，一条命令输入完毕，换行，再输入下一条命令即可。

设备收到命令行指令后会回应客户端，当客户端收到如下报文时，表示命令行执行成功（注意命令响应被 CDATA 节点包含）：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <CLI>
    <Execution>
      <![CDATA[对应命令行响应]]>
    </Execution>
  </CLI>
</rpc-reply>
```

### 1.12.4 命令行操作配置举例

#### 1. 配置需求

向设备发送显示 VLAN 相关信息的命令。

#### 2. 配置步骤

# 进入 XML 视图。

```
<Sysname> xml
```

# 进行能力交换。

请将以下报文拷贝、粘贴到客户端：

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
```

```

    <capabilities>
      <capability>
        urn:ietf:params:netconf:base:1.0
      </capability>
    </capabilities>
  </hello>
# 向设备发送显示 VLAN 相关信息的命令。
请将以下报文拷贝、粘贴到客户端：
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <CLI>
    <Execution>
      display vlan
    </Execution>
  </CLI>
</rpc>

```

### 3. 验证配置

# 如果客户端收到类似如下的报文，则表示操作成功。

```

<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <CLI>
    <Execution><![CDATA[
<Sysname>display vlan
Total VLANs: 1
The VLANs include:
1(default)
]]>
    </Execution>
  </CLI>
</rpc-reply>

```

## 1.13 订阅事件

### 1.13.1 功能简介

用户向设备订阅事件后，设备上发生用户订阅的事件时，设备会自动向订阅的客户端发送事件的日志。

### 1.13.2 配置限制和指导

订阅只对当前连接生效。如果连接断开，订阅会自动取消。

如果不配置订阅的事件流<stream></stream>，则缺省订阅 Syslog 事件。

### 1.13.3 订阅Syslog事件

事件订阅报文格式如下：

```

<?xml version="1.0" encoding="UTF-8"?>

```

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <create-subscription xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
    <stream>NETCONF</stream>
    <filter>
      <event xmlns="http://www.h3c.com/netconf/event:1.0">
        <Code>code</Code>
        <Group>group</Group>
        <Severity>severity</Severity>
      </event>
    </filter>
    <startTime>start-time</startTime>
    <stopTime>stop-time</stopTime>
  </create-subscription>
</rpc>
```

属性	说明
stream	订阅的事件流，Syslog事件流的名称为NETCONF
event	订阅的事件，此处可订阅的事件为系统支持的日志。具体可以订阅哪些模块的事件，请参见《日志信息参考手册》
code	日志信息中的助记符
group	日志信息中的模块名
severity	日志信息中的安全级别
start-time	订阅的开始时间
stop-time	订阅的结束时间

设备收到订阅报文后会回应客户端，当客户端收到如下报文时，表示订阅成功：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply >
```

当订阅出错时设备会返回错误信息，例如：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <rpc-error>
    <error-type>error-type</error-type>
    <error-tag>error-tag</error-tag>
    <error-severity>error-severity</error-severity>
    <error-message xml:lang="en">error-message</error-message>
  </rpc-error>
</rpc-reply>
```

错误报文的详细定义请参见 RFC 4741。

1.13.4 订阅监控事件

配置该功能后，NETCONF 每隔指定时间获取一次所订阅事件，并将符合订阅条件的信息发送给用户。

事件订阅报文格式如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<create-subscription xmlns='urn:ietf:params:xml:ns:netconf:notification:1.0'>
  <stream>NETCONF_MONITOR_EXTENSION</stream>
  <filter>
    <NetconfMonitor xmlns='http://www.h3c.com/netconf/monitor:1.0'>
      <XPath>XPath</XPath>
      <Interval>interval</Interval>
      <ColumnConditions>
        <ColumnCondition>
          <ColumnName>ColumnName</ColumnName>
          <ColumnValue>ColumnValue</ColumnValue>
          <ColumnCondition>ColumnCondition</ColumnCondition>
        </ColumnCondition>
      </ColumnConditions>
      <MustIncludeResultColumns>
        <ColumnName>columnName</ColumnName>
      </MustIncludeResultColumns>
    </NetconfMonitor>
  </filter>
  <startTime>start-time</startTime>
  <stopTime>stop-time</stopTime>
</create-subscription>
</rpc>
```

属性	说明
stream	订阅的事件流，监控事件流的名称为NETCONF_MONITOR_EXTENSION
NetconfMonitor	监控事件的过滤信息
XPath	监控事件的路径，格式为：模块名[/子模块名]/表名
interval	监控的时间间隔，取值范围为1~4294967，缺省值为300，单位为秒，即每隔300秒获取一次符合订阅条件的信息
ColumnName	监控列的名称，格式为：[组名称.]列名称
ColumnValue	监控列的过滤值

属性	说明
<b>ColumnCondition</b>	监控列的过滤条件： <ul style="list-style-type: none"> <li>• <b>more</b>: 大于</li> <li>• <b>less</b>: 小于</li> <li>• <b>notLess</b>: 不小于</li> <li>• <b>notMore</b>: 不大于</li> <li>• <b>equal</b>: 等于</li> <li>• <b>notEqual</b>: 不等于</li> <li>• <b>include</b>: 包含</li> <li>• <b>exclude</b>: 不包含</li> <li>• <b>startWith</b>: 开始于</li> <li>• <b>endWith</b>: 结束于</li> </ul> 请根据监控列的过滤值的类型填写过滤条件
<b>start-time</b>	订阅的开始时间
<b>stop-time</b>	订阅的结束时间

设备收到订阅报文后会回应客户端，当客户端收到如下报文时，表示订阅成功：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
    <ok/>
</rpc-reply>
```

### 1.13.5 订阅模块上报事件

配置该功能后，所订阅模块发生用户所订阅事件时，将主动上报 NETCONF，NETCONF 将这样信息发送给用户。

事件订阅报文格式如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:xs="http://www.h3c.com/netconf/base:1.0">
<create-subscription xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
<stream>XXX_STREAM</stream>
    <filter type="subtree">
<event xmlns="http://www.h3c.com/netconf/event:1.0/xxx-features-list-name:1.0">
    <ColumnName xs:condition="Condition">value</ColumnName>
</event>
</filter>
<startTime>start-time</startTime>
<stopTime>stop-time</stopTime>
</create-subscription>
</rpc>
```

属性	说明
<b>stream</b>	订阅的事件流，请根据设备实际支持情况填写模块上报事件流的名称

属性	说明
<b>event</b>	订阅的事件的名称，一个事件流中包括多个事件，请根据事件流下的事件填写，命名空间为事件流的命名空间
<b>ColumnName</b>	当前事件下需要过滤的列的名称
<b>Condition</b>	<p>事件列的过滤条件：</p> <ul style="list-style-type: none"> <li>• <b>more</b>: 大于</li> <li>• <b>less</b>: 小于</li> <li>• <b>notLess</b>: 不小于</li> <li>• <b>notMore</b>: 不大于</li> <li>• <b>equal</b>: 等于</li> <li>• <b>notEqual</b>: 不等于</li> <li>• <b>include</b>: 包含</li> <li>• <b>exclude</b>: 不包含</li> <li>• <b>startWith</b>: 开始于</li> <li>• <b>endWith</b>: 结束于</li> </ul> <p>请根据事件列的过滤值的类型填写过滤条件</p>
<b>value</b>	当前事件下需要过滤的列的值
<b>start-time</b>	订阅的开始时间
<b>stop-time</b>	订阅的结束时间

设备收到订阅报文后会回应客户端，当客户端收到如下报文时，表示订阅成功：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

## 1.13.6 订阅事件配置举例

### 1. 组网需求

客户端订阅没有时间限制的全部事件。订阅后在断开连接之前，设备发生的所有事件都会发送给客户端。

### 2. 配置步骤

# 进入 XML 视图。

```
<Sysname> xml
```

# 进行能力交换。

请将以下报文拷贝、粘贴到客户端：

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
```



```
</capabilities>
</hello>
```

# 订阅全部事件，不限制订阅时间。

请将以下报文拷贝、粘贴到客户端：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <create-subscription xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
    <stream>NETCONF</stream>
  </create-subscription>
</rpc>
```

### 3. 验证配置

# 如果客户端收到如下报文，则表示订阅成功：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="100">
  <ok/>
</rpc-reply>
```

当用户（IP 地址为 192.168.100.130）登录设备时，设备会发送如下报文来通知订阅客户端：

```
<?xml version="1.0" encoding="UTF-8"?>
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2011-01-04T12:30:52</eventTime>
  <event xmlns="http://www.h3c.com/netconf/event:1.0">
    <Group>SHELL</Group>
    <Code>SHELL_LOGIN</Code>
    <Slot>1</Slot>
    <Severity>Notification</Severity>
    <context>VTY logged in from 192.168.100.130.</context>
  </event>
</notification>
```

## 1.14 关闭NETCONF会话

### 1.14.1 功能简介

NETCONF 支持用户关闭除自己外的 NETCONF 会话，被关闭会话的用户退回到用户视图。

### 1.14.2 Kill-session操作

请将以下报文拷贝、粘贴到客户端，用户即能关闭指定的会话：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <kill-session>
    <session-id>
      指定 session-ID
    </session-id>
  </kill-session>
</rpc>
```

设备收到会话关闭请求后会回应客户端，当客户端收到如下报文时，表示指定会话已经被关闭：

```
<?xml version="1.0" encoding="UTF-8"?>
```

```

    <rpc-reply message-id="100"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
        <ok/>
    </rpc-reply>

```

### 1.14.3 Kill-session操作配置举例

#### 1. 配置需求

当前有两个 NETCONF 登录用户，session ID 分别是 1 和 2，session ID 为 1 的用户要关闭 session ID 为 2 用户的会话。

#### 2. 配置步骤

# 进入 XML 视图。

```
<Sysname> xml
```

# 进行能力交换。

请将以下报文拷贝、粘贴到客户端：

```

<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <capabilities>
        <capability>
            urn:ietf:params:netconf:base:1.0
        </capability>
    </capabilities>
</hello>

```

# 关闭 session ID 为 2 的用户会话。

请将以下报文拷贝、粘贴到客户端：

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <kill-session>
        <session-id>2</session-id>
    </kill-session>
</rpc>

```

#### 3. 验证配置

如果客户端收到如下报文，则表明 session ID 为 2 的 NETCONF 会话已经被关闭。建立该会话的用户会从 XML 视图退回到用户视图。

```

<?xml version="1.0" encoding="UTF-8"?>
    <rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
        <ok/>
    </rpc-reply>

```

## 1.15 退出XML视图

#### 1. 配置限制和指导

进入 XML 视图后，必须先完成能力交换，再执行退出 XML 视图操作。

#### 2. 配置步骤

处于 XML 视图的用户退回到用户视图时，需要将以下报文拷贝、粘贴到客户端：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
```

```
<close-session/>
```

```
</rpc>
```

设备收到以上请求后会返回如下报文并退回到用户视图：

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<rpc-reply message-id="100"
```

```
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
```

```
  <ok/>
```

```
</rpc-reply>
```

## 2 Comware V7 支持的NETCONF操作

Comware V7 平台对 NETCONF 标准协议做了一些修订，删除了不常用的操作，增加部分新的操作

### 2.1.1 action

#### 【使用指导】

下发可配置的功能的动作，例如 **reset** 操作。

#### 【XML 格式样例】

清除全部接口的统计信息，XML 请求如下：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <action>
    <top xmlns="http://www.h3c.com/netconf/action:1.0">
      <Ifmgr>
        <ClearAllIfStatistics>
          <Clear>
          </Clear>
        </ClearAllIfStatistics>
      </Ifmgr>
    </top>
  </action>
</rpc>
```

### 2.1.2 CLI

#### 【使用指导】

执行命令行的命令。

请求消息将命令行语句封装在<CLI>标签中，命令行输出信息被封装在<CLI>标签中返回。

CLI 支持 Execution 和 Configuration 方式执行命令行：

- **Execution**：在用户视图下执行命令行。
- **Configuration**：在系统视图下执行命令行。使用该方式时需要指定 **exec-use-channel** 的属性：
  - **false**：不使用 **channel** 方式执行命令行。
  - **true**：使用临时 **channel** 执行命令行，执行完成后，自动关闭该 **channel**。
  - **persist**：使用保留 **channel** 执行命令行。使用该方式时，则需要执行 **Open-channel** 操作打开当前对话的 **channel**；使用完成后，执行 **Close-channel** 关闭 **Channel**。
- 对于其他视图下命令，则需要在 **Configuration** 下先指定进入子视图的命令，再指定配置命令。

一个 NETCONF 会话只能打开一个保留 **channel**，可以打开多个临时 **channel**。

如果没有执行 **Open-channel** 操作，则系统将自动打开 **channel**。

如果没有执行关闭保留 **channel** 操作，再次使用保留 **channel** 时，系统在上次最后执行的命令行所在视图再次执行命令行。

不支持执行交互式命令。

使用 **channel** 方式执行命令行时，不支持执行 **quit** 命令退出用户视图。

### 【XML 格式样例】

在系统视图下不使用 **channel** 方式执行 **display this** 命令：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <CLI>
    <Configuration exec-use-channel="false">display this</Configuration>
  </CLI>
</rpc>
```

## 2.1.3 close-session

### 【使用指导】

关闭当前 **NETCONF** 会话，并释放锁和这个 **session** 用到的内部资源（如内存等），退出 **XML** 视图。

### 【XML 格式样例】

关闭当前 **NETCONF** 会话，并释放锁和这个 **session** 用到的内部资源（如内存等），退出 **XML** 视图

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <close-session/>
</rpc>
```

## 2.1.4 edit-config: create

### 【使用指导】

创建配置。

**create** 操作必须指定配置对象。

- 如果当前配置表支持创建对象，且当前对象不存在，则先创建配置对象，再创建指定的配置。
- 如果配置对象下对应的配置项已经存在，则返回 **data-exist** 错误。

### 【XML 格式样例】

将 **BufferSize** 设置为 120 的 **xml** 请求如下：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <config>
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
        <Syslog xmlns="http://www.h3c.com/netconf/config:1.0" xc:operation="create">
          <LogBuffer>
            <BufferSize>120</BufferSize>
          </LogBuffer>
        </Syslog>
      </top>
    </config>
  </edit-config>
</rpc>
```

```
</top>
</config>
</edit-config>
</rpc>
```

### 2.1.5 edit-config: delete

#### 【使用指导】

删除配置。

- 当指定的删除对象中只有表索引时，则删除此配置指定对象的所有配置，同时删除指定对象。
- 当指定的删除对象中不仅仅有表索引还存在配置项时，则删除此对象下面的指定配置。
- 如果系统中指定对象不存在，则直接返回不存在的错误消息。

#### 【XML 格式样例】

同 edit-config: create，把 create 修改为 delete 即可。

### 2.1.6 edit-config: merge

#### 【使用指导】

在当前运行配置的基础上直接运行指定配置。

merge 操作的 XML 数据格式和 create 类似，只是 operation 属性需要指定为 “merge”。

merge 操作必须指定具体的操作对象（行）：

- 如果指定的对象存在，则直接配置该对象。
- 如果指定的对象不存在，但允许创建，则先创建再配置该对象。
- 如果指定的对象不存在且不允许创建，则返回 merge 失败。

#### 【XML 格式样例】

同 edit-config: create，把 create 修改为 merge 即可。

### 2.1.7 edit-config: remove

#### 【使用指导】

删除配置。

- 当指定的删除对象中只有表索引时，则删除此配置指定对象的所有配置，同时删除指定对象。
- 当指定的删除对象中不仅仅有表索引还存在配置项时，则删除此对象下面的指定配置。
- 如果系统中指定对象不存在，或者 XML 消息未指定对象，则直接返回成功。

#### 【XML 格式样例】

同 edit-config: create，把 create 修改为 remove 即可。

### 2.1.8 edit-config: replace

#### 【使用指导】

替换配置。

- 如果指定的对象存在，则替换指定对象的配置为当前配置。

- 如果指定的对象不存在，但允许创建，则先创建再配置该对象为当前配置。
- 如果指定对象不存在且不允许创建，则不进行 **replace** 操作，返回 **invalid-value** 错误，提示用户配置对象不存在。

#### 【XML 格式样例】

同 **edit-config: create**，把 **create** 修改为 **replace** 即可。

### 2.1.9 edit-config测试处理选项

#### 【使用指导】

执行 **edit-config** 操作时，可指定一个测试选项，使用 **<test-option>** 节点来决定当前配置是否下发。

该节点的缺省值为 **test-then-set**，全部取值为：

- **test-then-set**：如果没有错误则将配置设置到系统。
- **set**：将配置设置到系统。
- **test-only**：只测试，并不下发配置到系统。语法检查通过，就返回 **ok** 成功，否则失败。

#### 【XML 格式样例】

下发一个接口的配置，仅测试，XML 请求如下：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <test-option>test-only</test-option>
    <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
        <Ifmgr xc:operation="merge">
          <Interfaces>
            <Interface>
              <IfIndex>262</IfIndex>
              <Description>222</Description>
              <ConfigSpeed>2</ConfigSpeed>
              <ConfigDuplex>1</ConfigDuplex>
            </Interface>
          </Interfaces>
        </Ifmgr>
      </top>
    </config>
  </edit-config>
</rpc>
```

### 2.1.10 edit-config缺省操作选项

#### 【使用指导】

**edit-config** 操作用于修改当前系统配置。

NETCONF 定制了 merge、create、delete 和 replace 等修改配置的方式。当 XML 消息中未指定修改配置方式时，则使用缺省操作作为当前指令的操作方式，不会修改缺省操作的缺省值。

缺省操作的缺省值为 merge，可以在 XML 消息中通过<default-operation>节点设置，取值为：

- merge：当配置方式和缺省操作方式均未指定时，使用该方式。
- replace：当配置方式未指定，缺省操作指定为 replace 时，edit-config 操作缺省为 replace 操作。
- none：当配置方式未指定，缺省操作指定为 none 的时候，edit-config 操作缺省为 none 操作。none 操作主要用来检查，下发为 none 操作的配置仅仅做 Schema 校验，不进行真正的配置下发。语法检查通过，返回 ok 成功，否则失败。

### 【XML 格式样例】

下发一个空的操作，该操作仅仅验证格式，并不真正下发给系统，xml 请求如下：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <default-operation>none</default-operation>
    <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
        <Ifmgr>
          <Interfaces>
            <Interface>
              <IfIndex>262</IfIndex>
              <Description>222222</Description>
            </Interface>
          </Interfaces>
        </Ifmgr>
      </top>
    </config>
  </edit-config>
</rpc>
```

## 2.1.11 edit-config缺省错误处理选项

### 【使用指导】

edit-config 将指定的配置设置到系统上，完成配置设置的操作。

在执行 edit-config 的过程中，如果遇到一个实例配置出错，缺省情况下会直接返回错误，并提供错误选项，通过错误选项取值的不同，在发生错误的时候进行不同的处理操作。

<error-option>节点用于设置一个实例配置出错后，后续实例配置的处理方式，缺省值为 stop-on-error，全部取值为：

- stop-on-error：停止处理，返回错误。此选项为缺省选项。
- continue-on-error：继续处理，但是报告错误。
- rollback-on-error：停止并回滚配置。



### 【XML 格式样例】

下发两个接口的配置，当下发第一个接口的配置发生错误时，继续进行下一个接口配置的下发，XML 请求如下：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <error-option>continue-on-error</error-option>
    <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
        <Ifmgr xc:operation="merge">
          <Interfaces>
            <Interface>
              <IfIndex>262</IfIndex>
              <Description>222</Description>
              <ConfigSpeed>1024</ConfigSpeed>
              <ConfigDuplex>1</ConfigDuplex>
            </Interface>
            <Interface>
              <IfIndex>263</IfIndex>
              <Description>333</Description>
              <ConfigSpeed>1024</ConfigSpeed>
              <ConfigDuplex>1</ConfigDuplex>
            </Interface>
          </Interfaces>
        </Ifmgr>
      </top>
    </config>
  </edit-config>
</rpc>
```

#### 2.1.12 edit-config增量下发

### 【使用指导】

增量下发选项 **incremental** 放置在列上，对于类似 **vlan permitlist** 列表集合性质的列，可能支持增量下发，用户请求 XML 中有增量下发选项时，最终执行结果不影响本列原有的数据。

增量下发只支持 **edit-config**，但不支持 **edit-config** 中的 **replace**。

关于支持增量下发的模块，请参见模块对应的 **NETCONF XML API**。

### 【XML 格式样例】

下发一个接口的 **VLAN** 配置，使用增量下发，262 接口原有 **untagvlanlist** 为 12~15，下发后为 1~10，12~15。XML 请求如下：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:h3c="http://www.h3c.com/netconf/base:1.0">
  <edit-config>
    <target>
      <running/>
```

```

</target>
<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <top xmlns="http://www.h3c.com/netconf/config:1.0">
    <VLAN xc:operation="merge">
      <HybridInterfaces>
        <Interface>
          <IfIndex>262</IfIndex>
          <UntaggedVlanList h3c: incremental="true">1-10</UntaggedVlanList>
        </Interface>
      </HybridInterfaces>
    </VLAN>
  </top>
</config>
</edit-config>
</rpc>

```

### 2.1.13 get

#### 【使用指导】

获取设备运行状态和运行配置。

#### 【XML 格式样例】

获取 Syslog 模块的全部数据的 XML 请求如下：

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:xc="http://www.h3c.com/netconf/base:1.0">
  <get>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        <Syslog>
        </Syslog>
      </top>
    </filter>
  </get>
</rpc>

```

### 2.1.14 get-bulk

#### 【使用指导】

从指定索引的下一条开始获取 N 条运行状态和运行配置（索引行数据不返回）。

用户通过 **index** 属性指定索引，通过 **count** 属性指定 N。如未指定索引，则以第一条为索引；如未指定 N，或者数据表中符合条件的数据记录不足 N 条，则返回表中所有剩下的数据条目。

**get** 操作会返回所有符合条件的数据，在某些情况下会导致效率问题。**get-bulk** 允许用户从固定数据项开始，向后获取指定条目的数据记录。

#### 【XML 格式样例】

获取全部接口的数据的 xml 请求如下：

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-bulk>

```

```

<filter type="subtree">
  <top xmlns="http://www.h3c.com/netconf/data:1.0">
    <Ifmgr>
      <Interfaces xc:count="5" xmlns:xc="http://www.h3c.com/netconf/base:1.0">
        <Interface/>
      </Interfaces>
    </Ifmgr>
  </top>
</filter>
</get-bulk>
</rpc>

```

## 2.1.15 get-bulk-config

### 【使用指导】

从指定索引的下一条批量获取可配置的功能的变量值。

配置限制与 get-bulk 操作相同。

### 【XML 格式样例】

获取全部接口配置信息的 xml 请求如下：

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-bulk-config>
    <source>
      <running/>
    </source>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
        <Ifmgr>
          </Ifmgr>
        </top>
      </filter>
    </get-bulk-config>
  </rpc>

```

## 2.1.16 get-config

### 【使用指导】

获取可配置的功能的变量值。如果对应功能未配置，则返回一个空的<data>。

### 【XML 格式样例】

获取接口表内所有配置的 XML 请求如下：

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:xc="http://www.h3c.com/netconf/base:1.0">
  <get-config>
    <source>
      <running/>
    </source>
    <filter type="subtree">

```

```

<top xmlns="http://www.h3c.com/netconf/config:1.0">
  <Ifmgr>
    <Interfaces>
      <Interface/>
    </Interfaces>
  </Ifmgr>
</top>
</filter>
</get-config>
</rpc>

```

### 2.1.17 get-sessions

#### 【使用指导】

获取当前系统中所有 NETCONF 会话的信息（不能指定 sessions-ID）。

#### 【XML 格式样例】

获取当前系统中所有 NETCONF 会话的信息：

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-sessions/>
</rpc>

```

### 2.1.18 kill-session

#### 【使用指导】

关闭其他 NETCONF 会话。

该操作不支持关闭用户自己的 NETCONF 会话。

#### 【XML 格式样例】

关闭 session-id 为 1 的 NETCONF 会话：

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <kill-session>
    <session-id>1</session-id>
  </kill-session>
</rpc>

```

### 2.1.19 load

#### 【使用指导】

加载配置。

执行该操作后，指定配置文件中的配置被合并到当前运行配置中。

#### 【XML 格式样例】

将文件 a1.cfg 中的配置合并到设备的当前配置中：

```

<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <load>
    <file>a1.cfg</file>
  </load>

```

```
</rpc>
```

## 2.1.20 lock

### 【使用指导】

NETCONF 加锁操作将锁定配置数据，即<edit-config>操作中指定配置数据，其他操作（例如<get>操作）不受锁的限制。

用户通过 NETCONF 加锁操作锁定 NETCONF 连接后，其他用户既不能使用 NETCONF 执行下发配置操作，也不能通过 CLI 或 SNMP 等其他配置方式对设备进行配置。

### 【XML 格式样例】

禁止 NETCONF 会话修改设备的当前配置，XML 请求如下：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <lock>
    <target>
      <running/>
    </target>
  </lock>
</rpc>
```

## 2.1.21 rollback

### 【使用指导】

回滚配置。

该操作必须使用子元素<file>指定需要回滚的配置文件的名称。

执行该操作后，当前运行配置回滚为指定配置文件中的配置。

### 【XML 格式样例】

将设备当前配置回退到文件 1A.cfg 中配置的状态：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <rollback>
    <file>1A.cfg</file>
  </rollback>
</rpc>
```

## 2.1.22 save

### 【使用指导】

保存运行配置。

该操作可以使用子元素<file>来指定保存的配置文件名称。当 save 操作中不存在子元素<file>列时，则设备会自动将当前运行配置保存到主用下次启动配置文件中。

OverWrite 属性用来判断当指定的配置文件名存在时，当前配置是否覆盖原配置文件并保存成功。

### 【XML 格式样例】

将设备当前配置保存到文件 test.cfg 中：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
```

```
<save OverWrite="false">
  <file>test.cfg</file>
</save>
</rpc>
```

### 2.1.23 unlock

#### 【使用指导】

解锁。

当会话结束时锁也会被自动释放。

#### 【XML 格式样例】

取消锁保护，允许 NETCONF 会话修改设备的当前配置：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <unlock>
    <target>
      <running/>
    </target>
  </unlock>
</rpc>
```

# 目 录

1 CWMP .....	1-1
1.1 CWMP简介 .....	1-1
1.1.1 CWMP网络框架 .....	1-1
1.1.2 CWMP基本功能 .....	1-1
1.1.3 CWMP工作机制 .....	1-3
1.2 CWMP配置限制和指导 .....	1-5
1.3 CWMP配置任务简介 .....	1-5
1.4 通过命令行方式开启CWMP功能 .....	1-6
1.5 配置ACS属性 .....	1-6
1.5.1 ACS属性配置介绍 .....	1-6
1.5.2 配置首选的ACS属性 .....	1-6
1.5.3 通过命令行配置缺省的ACS属性 .....	1-7
1.6 配置CPE属性 .....	1-8
1.6.1 CPE属性配置介绍 .....	1-8
1.6.2 绑定SSL客户端策略 .....	1-8
1.6.3 配置CPE的用户名和密码 .....	1-9
1.6.4 配置CPE的业务代码信息 .....	1-9
1.6.5 配置CWMP连接接口 .....	1-9
1.6.6 配置自动连接参数 .....	1-10
1.6.7 配置CPE无数据传输超时的时间 .....	1-11
1.6.8 配置CPE的NAT穿越功能 .....	1-11
1.7 CWMP显示和维护 .....	1-12
1.8 CWMP典型配置举例 .....	1-12
1.8.1 CWMP基本组网配置举例 .....	1-12

# 1 CWMP

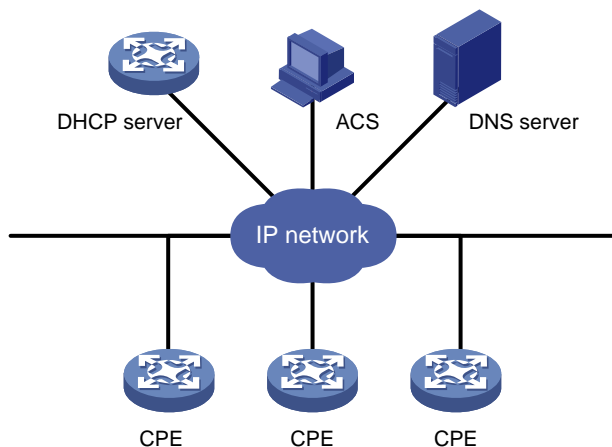
## 1.1 CWMP简介

CWMP（CPE WAN Management Protocol，CPE 广域网管理协议），又称为 TR-069 协议，是由 DSL（Digital Subscriber Line，数字用户线路）论坛发起开发的远程网络设备管理协议。CWMP 协议可通过服务器对数量众多、分布较广的设备进行远程自动部署、配置和管理，可用于包括以太网在内的不同网络中。

### 1.1.1 CWMP网络框架

CWMP网络的基本框架如 [图 1-1](#) 所示：

图1-1 CWMP 网络基本框架示意图



CWMP 网络元素主要有：

- ACS：自动配置服务器，网络中的管理设备。
  - CPE：用户端设备，网络中的被管理设备。
  - DNS server：域名服务器。CWMP 协议规定 ACS 和 CPE 使用 URL（Uniform Resource Locator，统一资源定位符）地址来互相识别和访问，DNS 用于帮助解析 URL 参数。
  - DHCP server：动态主机配置协议服务器。用来为 CPE 分配 IP 地址，并下发 ACS 相关信息。
- 该网络元素可以根据实际网络需要选择部署，一般用于设备第一次上电接入 CWMP 环境。

在 CWMP 网络框架中，设备作为 CPE 设备。

### 1.1.2 CWMP基本功能

#### 1. 通过ACS向CPE自动下发配置文件

为了便于对提供相同业务功能的新设备进行快速配置，网络管理员可以在 ACS 上创建针对该类设备的配置文件。当 CPE 设备与 ACS 建立连接后，ACS 可以判断 CPE 设备所属类别，并将对应该



类设备的配置文件下发给 CPE 设备，从而可以使相同类型的大量 CPE 设备获得相同的业务配置。目前 ACS 可以通过产品型号和序列号等将 CPE 划分为不同的类别。

ACS 向 CPE 下发配置文件时，可以通过以下两种方式进行：

- 部署为启动配置：ACS 向 CPE 发送配置文件，并设置为下次启动配置文件，当 CPE 重启之后，便可以使用新的配置运行。
- 部署为运行配置：ACS 将配置内容直接发给 CPE，并写入到 CPE 的当前配置中，配置内容即时生效，但是需要再执行保存配置的操作，以保证重启后配置不会丢失。

## 2. 通过ACS管理CPE设备软件

网络管理员可以将 CPE 设备的软件保存在 ACS 上，当 ACS 发现软件版本有更新，将会通知 CPE 进行下载。CPE 收到 ACS 的下载请求后，能够根据 ACS 报文中提供的下载地址和文件名，自动到 ACS 或指定的文件服务器下载软件文件。下载完成后，对下载文件的合法性做相应的检查，并将下载结果（成功或失败）反馈给 ACS。

## 3. 数据备份

ACS 可以要求 CPE 上传配置文件或日志文件到 ACS 或文件服务器上，并保存到指定的位置。

## 4. 通过ACS监控CPE的状态和性能

ACS 可以监控与其相连的 CPE 的各种参数。由于不同的 CPE 具有不同的性能，可执行的功能也各异，因此 ACS 必须能识别不同类型 CPE 的性能，并监控到 CPE 的当前配置以及配置的变更。

ACS 能够监控的状态和性能有：

- 厂商名称（Manufacturer）
- 厂商标识 OUI（ManufacturerOUI）
- 序列号（SerialNumber）
- 硬件版本号（HardwareVersion）
- 软件版本号（SoftwareVersion）
- 设备状态（DeviceStatus）
- 启动时间（UpTime）
- 配置文件（ConfigFile），用于更新 CPE 的本地配置文件，ACS 可以使用文件形式和当前配置形式向 CPE 下发配置文件
- ACS 地址（URL），更新 CPE 记录的 ACS 地址，可用于主备 ACS 服务器之间的切换
- ACS 用户名（Username）/密码（Password），当 ACS 上连接用户名和密码发生变更时，可以自动同步到 CPE 设备，也可用于主备 ACS 服务器切换时向 CPE 通告备用 ACS 服务器的验证信息
- Inform 报文自动发送使能标志（PeriodicInformEnable），开启 CPE 设备发送 Inform 报文的功能
- Inform 报文周期发送时间间隔（PeriodicInformInterval），配置 CPE 周期性向 ACS 发送 Inform 报文建立连接，用于定期查询更新和备份信息
- Inform 报文定期发送日期（PeriodicInformTime），配置 CPE 在指定时间点向 ACS 发送 Inform 报文建立连接，用于在指定时间查询更新和备份信息
- CPE 地址

- CPE 用户名（ConnectionRequestUsername）/密码（ConnectionRequestPassword），配置 CPE 在接受 ACS 发起的连接时所需要的验证信息

### 1.1.3 CWMP工作机制

#### 1. ACS使用RPC方法管理CPE

CWMP 通过 RPC（Remote Procedure Call，远程调用）方法来完成 ACS 对 CPE 的管理和监控。RPC 方法封装于 HTTP 或 HTTPS 协议中。主要操作方法描述如下：

- **Get:** ACS 使用该方法可以获取 CPE 上参数的值。
- **Set:** ACS 使用该方法可以设置 CPE 上参数的值。
- **Inform:** 当 CPE 与 ACS 建立连接时、各应用模块具有主动通知属性的配置发生改变时和 CPE 周期性发送本地信息到 ACS 时，CPE 都要通过该方法向 ACS 发起通告信息。
- **Download:** 为了保证 CPE 端硬件的升级以及厂商配置文件的自动下载，ACS 使用该方法可以要求 CPE 到指定的 URL 下载指定的文件来更新 CPE 的本地文件。
- **Upload:** 为了方便 ACS 对 CPE 端的管理，ACS 使用该方法可以要求 CPE 将指定的文件上传到 ACS 指定的位置。
- **Reboot:** 当 CPE 故障或者需要软件升级的时候，ACS 使用该方法可以对 CPE 进行远程重启。

#### 2. ACS和CPE的自动连接

当 CPE 有事件需要向 ACS 报告时，会自动向 ACS 发起连接请求。常见有如下事件：

- CPE 连接 ACS 的 URL 值改变。CPE 会自动使用新的 URL 值向 ACS 发起连接请求。
- CPE 上电启动。CPE 启动后，会自动向 ACS 发起连接请求。
- 配置了周期性发送 Inform 报文功能。CPE 会周期性向 ACS 服务器发起连接请求。
- 配置了定时发送 Inform 报文功能。CPE 会在指定时间向 ACS 服务器发起连接请求。

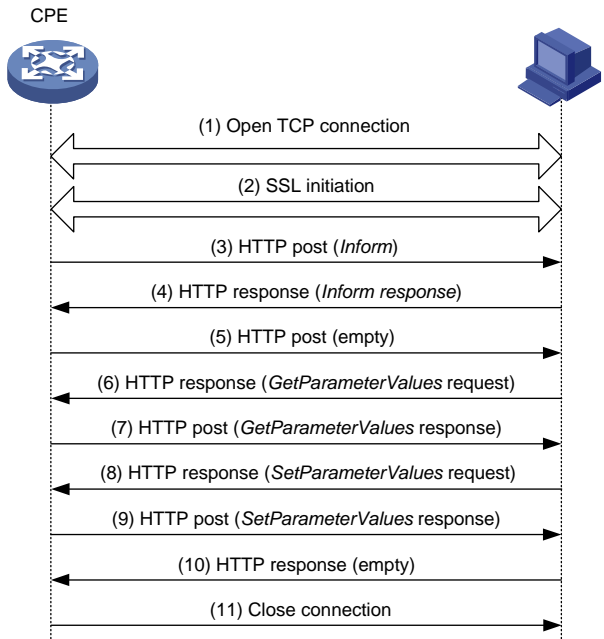
#### 3. CWMP连接建立

CWMP 连接建立的过程如下：

- (1) 建立 TCP 连接。
- (2) 若使用 HTTPS 协议，进行 SSL 初始化，建立安全机制。
- (3) CPE 发送 Inform 报文，开始建立 CWMP 连接。Inform 报文使用 Eventcode 字段描述发送 Inform 报文的原因，该举例为“6 CONNECTION REQUEST”，表示 ACS 要求建立连接。
- (4) 如果 CPE 通过 ACS 的认证，ACS 将返回 Inform 响应报文，连接建立。
- (5) 如果 CPE 没有别的请求，就会发送一个空报文，以满足 HTTP/HTTPS 报文请求/响应报文交互规则。

如 [图 1-2](#) 的步骤（1）~（5）所示。

图1-2 CWMP 消息交互举例



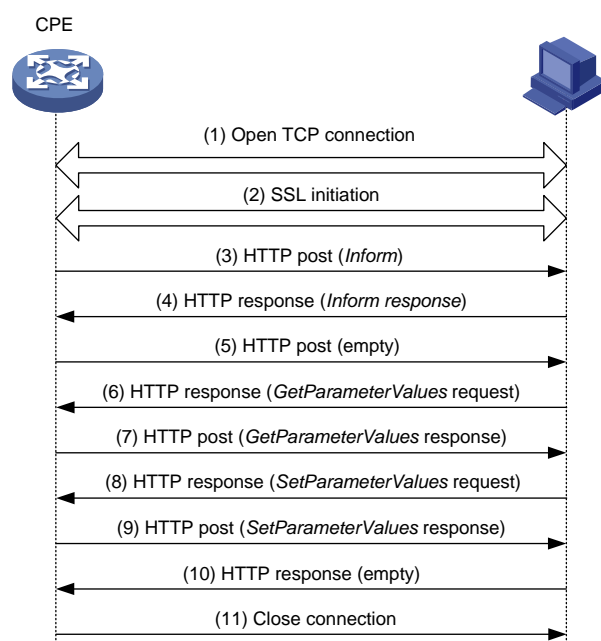
4. 主备ACS切换

为了保证可靠性，网络中通常会有主、备两台 ACS，在 CWMP 连接建立的基础上，当主 ACS 需要升级重启时，CWMP 可以完成主备切换，步骤如下：

- (1) 主用 ACS 查询 CPE 上设置的 ACS URL 的值。
- (2) CPE 把获取到的 ACS URL 的值回复给主用 ACS。
- (3) 主用 ACS 发现 CPE 的 ACS URL 是本机 URL 的值，于是发起 Set 请求，要求将 CPE 的 ACS URL 设置为备用 ACS 的 URL 的值。
- (4) 设置成功，CPE 发送响应报文。
- (5) ACS 发送空报文通知 CPE 没有别的请求。
- (6) CPE 关闭连接。

如 [图 1-3](#) 的步骤（6）～（11）所示。之后，CPE 将向备用 ACS 发起连接。

图1-3 CWMP 消息交互举例



## 1.2 CWMP配置限制和指导

ACS 和 CPE 属性参数可以通过 CPE 命令行、DHCP 服务器或者是 ACS 来进行配置。各个属性支持的配置方式不同。

当某参数支持多种配置方式时，DHCP 配置的优先级最低，ACS 配置和命令行配置的优先级相同。高优先级配置方式可以修改低优先级配置方式配置的参数，配置方式优先级相同时，以最新的配置为准。

本文档仅描述通过命令行和 DHCP 方式的配置。通过 ACS 进行 ACS 和 CPE 属性配置的信息，请参见 ACS 的手册。

## 1.3 CWMP配置任务简介

CWMP 配置任务如下：

- (1) [通过命令行方式开启CWMP功能](#)  
通过 DHCP 服务器也可以开启设备上的 CWMP 功能。
- (2) [配置ACS属性](#)
  - a. [配置首选的ACS属性](#)
  - b. （可选）[通过命令行配置缺省的ACS属性](#)
- (3) [配置CPE属性](#)
  - a. [绑定SSL客户端策略](#)  
使用 HTTPS 时必选。
  - b. （可选）[配置CPE的用户名和密码](#)

- c. (可选) [配置CPE的业务代码信息](#)
- d. (可选) [配置CWMP连接接口](#)
- e. (可选) [配置自动连接参数](#)
- f. (可选) [配置CPE无数据传输超时的时间](#)
- g. (可选) [配置CPE的NAT穿越功能](#)

## 1.4 通过命令行方式开启CWMP功能

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 CWMP 视图。

```
cwmp
```

- (3) 开启 CWMP 功能。

```
cwmp enable
```

缺省情况下，CWMP 功能处于关闭状态。

## 1.5 配置ACS属性

### 1.5.1 ACS属性配置介绍

CPE 上的 ACS 属性包括首选和缺省两种类型。CPE 优先使用首选 ACS 属性参数与 ACS 建立连接，只有当没有首选 ACS 属性参数时，才会通过缺省 ACS 属性参数与 ACS 建立连接。

首选和缺省的 ACS 属性支持的配置方式如下：

- 首选的 ACS 属性可以通过三种方式配置：命令行配置、DHCP 服务器或 ACS 下发。
- 缺省的 ACS 属性只能通过命令行配置。

### 1.5.2 配置首选的ACS属性

#### 1. 通过DHCP服务器下发首选的ACS属性

在 CWMP 网络中，DHCP 服务器主要用于向 CPE 通告 ACS 的位置和验证信息，因此 DHCP 服务器上的配置主要包含以下内容：

- 配置地址池，为 CPE 设备分配 IP 地址
- 指定 DNS 服务器
- 配置 option 43 选项，向 CPE 通告 ACS 信息

ACS 属性可以通过在 DHCP server 上配置 option 43 参数来实现。当 CPE 访问 DHCP server 时，DHCP server 会将 ACS 参数发送给 CPE。这里主要介绍 option 43 选项的配置方法，关于地址池和 DNS 服务器的配置请参见“三层技术-IP 业务配置指导”中的“DHCP”和“域名解析”。

当使用 H3C 设备作为 DHCP server 时，可以使用命令行配置 ACS 参数，命令格式为：**option 43 hex 01***length URL username password*。

- *length*: 表示关键字 **option 43 hex 01** 后面参数的总长度，用十六进制数表示。
- *URL*: ACS 的地址。

- `username`: ACS 的用户名。
- `password`: ACS 的密码。

ACS的URL、用户名和密码参数的格式必须为字符对应的ASCII码的十六进制值格式。假设将ACS地址配置为`http://169.254.76.31:7547`、用户名配置为 `1234`、密码配置为 `5678`，（`URL+1` 个空格+`username+1` 个空格+`password`）一共为 35 个字符，如 [表 1-1](#) 所示：

表1-1 ACS 参数的十六进制值格式

参数	参数值	十六进制值
字符长度	35字符	23
ACS的URL	<code>http://169.254.76.31:7547</code>	687474703A2F2F3136392E3235342E37362E33313A3735343720 空格对应的ASCII码的十六进制值为20
ACS的用户名	<code>1234</code>	3132333420 空格对应的ASCII码的十六进制值为20
ACS的密码	<code>5678</code>	35363738

可使用以下配置步骤：

```
<Sysname> system-view
[Sysname] dhcp server ip-pool 0
[Sysname-dhcp-pool-0] option 43 hex
0123687474703A2F2F3136392E3235342E37362E33313A3735343720313233342035363738
```

有关 DHCP、option 43 参数以及 `option` 命令的详细介绍，请参见“三层技术-IP 业务配置指导”中的“DHCP”。

## 2. 通过命令行配置首选的ACS属性

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 CWMP 视图。

```
cwmp
```

- (3) 配置 CPE 连接 ACS 的 URL 值。

```
cwmp acs url url
```

缺省情况下，没有配置 CPE 连接 ACS 的 URL。

- (4) 配置 CPE 连接 ACS 的用户名。

```
cwmp acs username username
```

缺省情况下，没有配置 CPE 连接 ACS 的用户名。

- (5) （可选）配置 CPE 连接 ACS 的密码。

```
cwmp acs password { cipher | simple } string
```

缺省情况下，没有配置 CPE 连接 ACS 的密码。

## 1.5.3 通过命令行配置缺省的ACS属性

- (1) 进入系统视图。

**system-view**

- (2) 进入 CWMP 视图。

**cwmp**

- (3) 配置 CPE 连接 ACS 的缺省 URL 值。

**cwmp acs default url url**

缺省情况下，未配置 CPE 连接 ACS 的缺省 URL。

- (4) 配置 CPE 连接 ACS 的缺省用户名。

**cwmp acs default username username**

缺省情况下，未配置 CPE 连接 ACS 的缺省用户名。

- (5) （可选）配置 CPE 连接 ACS 的缺省密码。

**cwmp acs default password { cipher | simple } string**

缺省情况下，未配置 CPE 连接 ACS 的缺省密码。

## 1.6 配置CPE属性

### 1.6.1 CPE属性配置介绍

除了下列 CPE 属性仅能通过命令行配置外，其他所有的 CPE 属性均可通过命令行或 ACS 进行配置：

- CWMP 连接接口
- NAT 穿越
- CPE 自动重新连接的次数
- 用于 HTTPS 的 SSL 客户端策略

### 1.6.2 绑定SSL客户端策略

#### 1. 功能简介

CWMP 使用 HTTPS 协议时，CPE 作为 HTTPS 客户端，需要在 CPE 上执行本配置。在使用 HTTPS 协议时，ACS 作为 HTTPS 服务器端，CPE 作为 HTTPS 客户端。

#### 2. 配置准备

执行本配置前，需要先创建 SSL 客户端策略。关于 SSL 客户端策略的详细介绍和配置请参见“安全配置指导”中的“SSL”。

#### 3. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 进入 CWMP 视图。

**cwmp**

- (3) 绑定 SSL 客户端策略。

**ssl client-policy policy-name**

缺省情况下，CWMP 未绑定 SSL 客户端策略。

### 1.6.3 配置CPE的用户名和密码

#### 1. 功能简介

CPE 的用户名和密码用于 CPE 对 ACS 的合法性进行验证。当连接由 ACS 发起时，会话请求报文里会携带 CPE 用户名和密码。设备收到该报文后，会与本地设置的 CPE 用户名和密码比较，如果相同则通过认证，进入连接建立的下一阶段，否则，认证失败，退出连接建立过程。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 CWMP 视图。

```
cwmp
```

- (3) 配置 ACS 连接 CPE 的认证用户名。

```
cwmp cpe username username
```

缺省情况下，未配置 ACS 连接 CPE 的认证用户名。

- (4) （可选）配置 ACS 连接 CPE 的认证密码。

```
cwmp cpe password { cipher | simple } string
```

缺省情况下，未配置 ACS 连接 CPE 的认证密码。

可以只使用用户名进行认证，不使用密码认证。

### 1.6.4 配置CPE的业务代码信息

#### 1. 功能简介

当 CPE 与 ACS 之间建立连接时，CPE 需要在 Inform 报文中携带 provision-code 信息，ACS 根据此信息可以识别设备定制的业务以及相应的参数，以便更好地管理 CPE 设备。关于 ACS 对 provision-code 的支持情况，请参见 ACS 手册。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 CWMP 视图。

```
cwmp
```

- (3) 配置 CPE 的业务代码。

```
cwmp cpe provision-code provision-code
```

缺省情况下，CPE 的业务代码为 PROVISIONINGCODE。

### 1.6.5 配置CWMP连接接口

#### 1. 功能简介

CWMP 连接接口指的是 CPE 上用于连接 ACS 的接口。CPE 会在 Inform 报文中携带 CWMP 连接接口的 IP 地址，要求 ACS 通过此 IP 地址和自己建立连接；相应的，ACS 会向该 IP 地址回复 Inform 响应报文。



缺省情况下，系统会采用一定的机制去获取一个 CWMP 连接接口，但如果获取的 CWMP 连接接口不是 CPE 和 ACS 相连的接口时，就会导致 CWMP 连接建立失败。因此，在这种情况下需要手工指定 CWMP 连接接口。

## 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 CWMP 视图。

```
cwmp
```

- (3) 设置 CPE 上用于连接 ACS 的接口。

```
cwmp cpe connect interface interface-type interface-number
```

缺省情况下，未设置 CPE 上用于连接 ACS 的接口。

## 1.6.6 配置自动连接参数

### 1. 功能简介

用户可配置 CPE 周期性地发送 Inform 报文，向 ACS 自动发起连接，或者是在特定的时间点发起连接。为避免系统资源浪费，请对 CPE 连接失败时的重连次数进行限制。

当 CPE 向 ACS 请求建立连接失败，或者在会话过程中连接异常中止（CPE 没有收到表示会话正常结束的报文）时，设备可以自动重新发起连接。

### 2. 配置周期性发送 Inform 报文

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 CWMP 视图。

```
cwmp
```

- (3) 开启 CPE 周期发送 Inform 报文功能。

```
cwmp cpe inform interval enable
```

缺省情况下，CPE 周期发送 Inform 报文功能处于关闭状态。

- (4) 配置 CPE 发送 Inform 报文的周期。

```
cwmp cpe inform interval interval
```

缺省情况下，CPE 每隔 600 秒发送一次 Inform 报文。

### 3. 配置定时发送 Inform 报文

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 CWMP 视图。

```
cwmp
```

- (3) 配置 CPE 在指定时刻发送一次 Inform 报文。

```
cwmp cpe inform time time
```

缺省情况下，未配置 CPE 定时发送 Inform 报文的时间。

#### 4. 配置CPE自动重新连接的次数

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 CWMP 视图。

```
cwmp
```

- (3) 配置当创建连接失败时自动重新连接的次数。

```
cwmp cpe connect retry retries
```

缺省情况下，自动重新连接的次数为无限次，即设备会一直按照一定周期给 ACS 发送连接请求。

### 1.6.7 配置CPE无数据传输超时的时间

#### 1. 功能简介

无数据传输超时时间主要用于以下两种情况：

- 在连接建立过程中，CPE 向 ACS 发送连接请求，但是经过无数据传输超时时间还没有收到响应报文，CPE 将认为连接失败。
- 连接建立后，如果 CPE 与 ACS 在无数据传输超时时间内没有报文交互，CPE 将认为连接失效，并断开连接。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 CWMP 视图。

```
cwmp
```

- (3) 配置 CPE 无数据传输超时的时间。

```
cwmp cpe wait timeout seconds
```

缺省情况下，无数据传输超时的时间为 30 秒。

### 1.6.8 配置CPE的NAT穿越功能

#### 1. 功能简介

无论 CPE 与 ACS 之间是否存在 NAT 网关，CPE 的主动连接请求都能到达 ACS。而当 CPE 与 ACS 之间存在 NAT 网关时，ACS 主动发起的连接请求不能到达 CPE。此时，可以在设备上开启 NAT 穿越功能，使 ACS 的请求可以穿越网关。本特性的实现遵循 RFC 3489 定义的 STUN (Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)，NAT 的 UDP 简单穿越)。有关 NAT 的详细描述，请参见“三层技术-IP 业务配置指导”中的“NAT”。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 CWMP 视图。

```
cwmp
```

(3) 开启 CPE 的 NAT 穿越功能。

```
cwmp cpe stun enable
```

缺省情况下，CPE 的 NAT 穿越功能处于关闭状态。

1.7 CWMP显示和维护

在完成上述配置后，在任意视图下执行 **display** 命令可以显示配置后 CWMP 的运行情况，通过查看显示信息验证配置的效果。

表1-2 cwmp 显示和维护

操作	命令
显示CWMP的当前配置信息	<b>display cwmp configuration</b>
显示CWMP的当前状态信息	<b>display cwmp status</b>

1.8 CWMP典型配置举例



说明

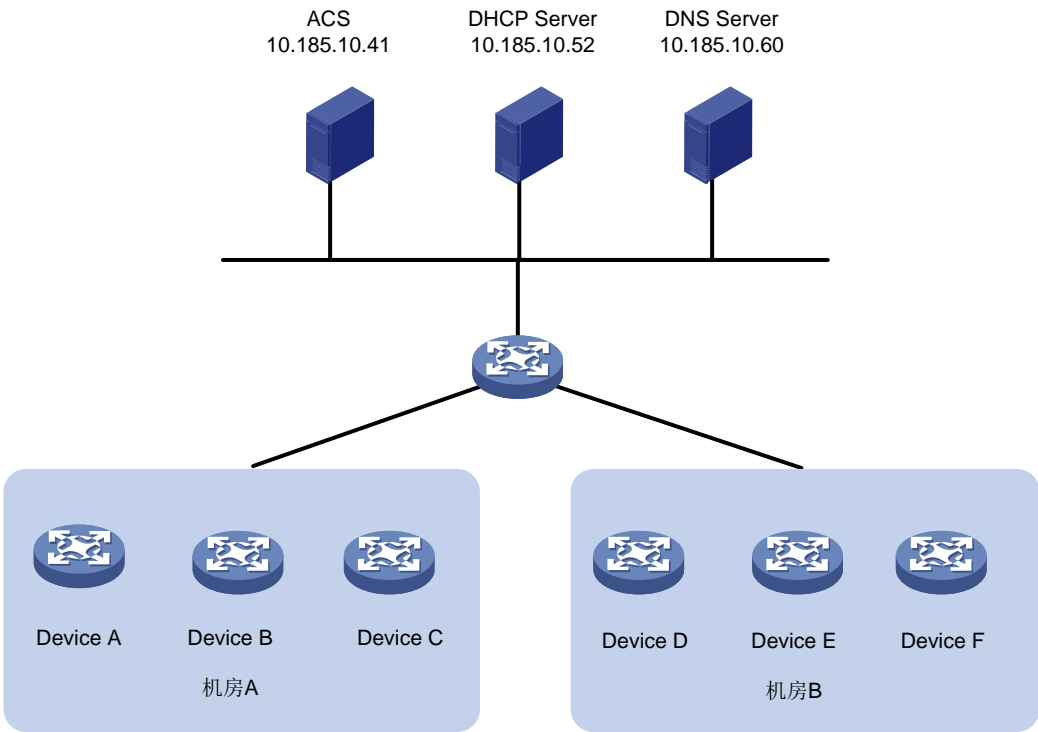
本举例使用安装了 H3C iMC BIMS 软件的服务器作为 ACS。随着软件版本的更新，BIMS 的功能和界面可能会有变化，如您所使用的软件界面与本例中不同，请参阅对应您使用版本的软件用户手册进行配置。

1.8.1 CWMP基本组网配置举例

1. 组网需求

某数据中心有两个机房（A 和 B）需要部署大量的设备，目前网络中已存在 ACS 服务器/DHCP 服务器/DNS 服务器，为提高部署效率，要求利用 CWMP 功能为两个机房中的 CPE 设备分别自动下发不同的配置文件。下面以每个机房内的三台设备为例介绍 CWMP 功能的配置方法。

图1-4 CWMP 典型配置案例组网图



其中部署到两个机房的设备及序列号如 [表 1-3](#) 所示。

表1-3 部署到机房的设备列表

机房	设备	序列号
A	DeviceA	210231A95YH10C0000045
	DeviceB	210235AOLNH12000010
	DeviceC	210235AOLNH12000015
B	DeviceD	210235AOLNH12000017
	DeviceE	210235AOLNH12000020
	DeviceF	210235AOLNH12000022

网络管理员已经为机房 A 和机房 B 的设备分别创建了配置文件 `configure1.cfg` 和 `configure2.cfg`，ACS 服务器的访问用户名和密码分别为“admin”和“12345”，URL 地址为 `http://10.185.10.41:9090`。

## 2. 配置ACS服务器

### (1) 登录 ACS 服务器（iMC）。

在 ACS 服务器上直接运行 Web 浏览器，在地址栏中输入 `http://10.185.10.41:8080/imc`（即 ACS 服务器的 IP 地址和端口号），并输入用户名和密码成功登录 iMC 界面。

### (2) 在 ACS 服务器上增加 CPE 分组。

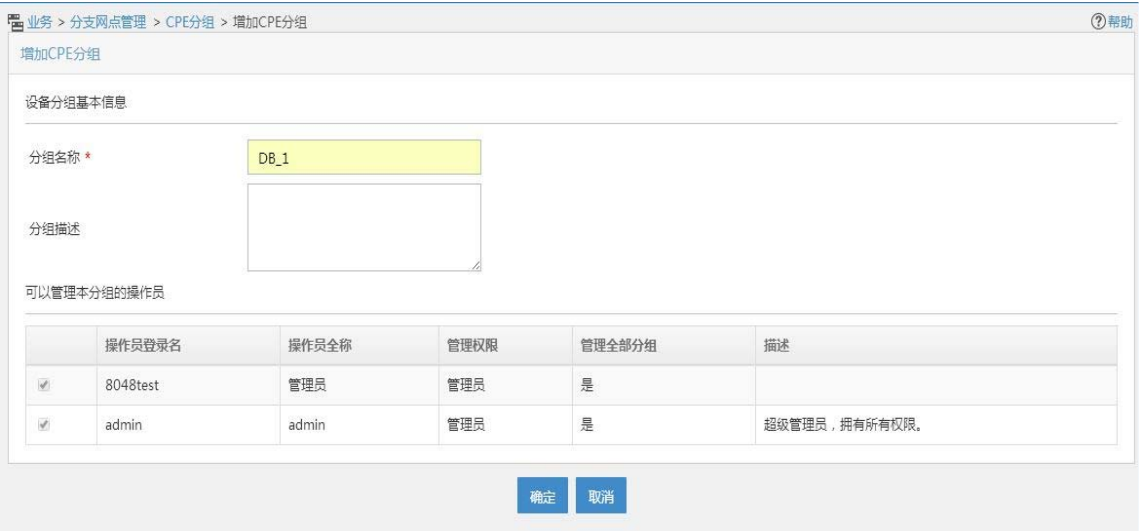
# 在导航栏中选择“业务>分支网点管理>CPE分组”，进入 [图 1-5](#) 所示页面。

图1-5 CPE 分组页面



# 单击<增加>按钮，进入 [图 1-6](#) 所示页面，以增加 “DB\_1” 组为例。

图1-6 增加 CPE 分组页面



# 设置分组名称后，单击<确定>按钮完成增加操作。

(3) 在 CPE 分组中增加 CPE，以增加 “DeviceA” 为例。

# 选择认证方式为 “ACS 用户名”，配置 CPE 名称为 DeviceA，配置 ACS 用户名为 admin；选择 ACS 认证密码生成方式为 “手工输入”，将密码和确认密码都配置为 12345；选择 CPE 设备的型号；选择 CPE 分组为 DB\_1。单击<确定>按钮完成增加操作。

# 在导航栏中选择 “业务>分支网点管理>资源管理>增加CPE”，进入 [图 1-7](#) 所示页面。

图1-7 增加 CPE 页面

业务 > 分支网点管理 > 增加CPE

增加CPE

基本信息

认证方式

ACS用户名

CPE名称 \*

DeviceA

ACS URL地址

ACS用户名 \*

admin

ACS 认证密码生成方式

手工输入

ACS密码 \*

.....

ACS 密码确认 \*

.....

CPE型号

H3C WB2320X-AGE

CPE分组

└ DB\_1

确定

取消

快速增加CPE

图1-8 增加 CPE 成功页面

业务 > 分支网点管理 > 所有CPE

★ 加入收藏 ② 帮助

查询CPE

CPE名称

型号

WB2320X-AGE

厂商

H3C

软件版本

CPE分组

序列号

CPE状态

所有

IP地址

接入IP

查询

重置

删除

同步

增加到分组

IP Ping测试

远程重启

恢复出厂设置

同步系统名称

定制列

刷新

<input type="checkbox"/>	状态	CPE名称	NAT CPE	序列号	型号	IP地址	操作
<input type="checkbox"/>	未知	DeviceA	否		WB2320X-AGE		

共有1条记录，当前第1 - 1，第 1/1 页。

«

<

1

>

»

50

重复以上步骤，将 DeviceB 和 DeviceC 设备的信息输入，完成机房 A 中的设备添加任务。

(4) 在 ACS 上配置模板库和 CPE 软件库类型。

# 在导航栏中选择“业务>分支网点管理>配置管理>配置模板库”，进入 图 1-9 所示页面。

图1-9 配置模板库页面

业务 > 分支网点管理 > 配置管理 > 配置模板库

★ 加入收藏 ② 帮助

查询条件

模板名称

模板类型

所有

目标文件夹

根目录

查询

重置

增加

导入

增加文件夹

删除

刷新

生成U盘启动文件

<input type="checkbox"/>	模板名称	类型	创建时间	说明	操作
<input type="checkbox"/>	Default Folder	文件夹	2018-02-06 09:43:36	预定义文件夹用于存放系统预定义配置片段。	
<input type="checkbox"/>	zd.cfg	配置片段	2018-02-07 15:03:31		...

共有2条记录，当前第1 - 2，第 1/1 页。

«

<

1

>

»

50

# 单击<导入>按钮，进入 图 1-10 所示页面。



注意

若导入的配置内容中第一条命令为 **system-view**，则该命令前面不能有“#”号或其他内容。

图1-10 导入配置模板页面

业务 > 分支网点管理 > 配置管理 > 配置模板库 > 导入配置模板

导入配置模板

选择文件 \*

temp.cfg

选择文件

目标文件 \*

temp.cfg

?

模板类型

配置片段

片段类型

命令行

目标文件夹

根目录

适用CPE

H3C WB2320X-AGE

选取型号

删除型号

说明

确定

取消

# 选择文件和模板类型后，显示导入配置模板成功页面。

图1-11 导入配置模板库成功页面

业务 > 分支网点管理 > 配置管理 > 配置模板库

★加入收藏 ? 帮助

查询条件

模板名称

模板类型

所有

目标文件夹

根目录

查询

重置

+ 增加

📁 导入

📁 增加文件夹

🗑 删除

🔄 刷新

💾 生成U盘启动文件

<input type="checkbox"/>	模板名称	类型	创建时间	说明	操作
<input type="checkbox"/>	📁 Default Folder	文件夹	2018-02-06 09:43:36	预定义文件夹用于存放系统预定义配置片段。	
<input type="checkbox"/>	📄 temp.cfg	配置片段	2018-05-04 11:12:17		...
<input type="checkbox"/>	📄 zd.cfg	配置片段	2018-02-07 15:03:31		...

共有3条记录，当前第1 - 3，第 1/1 页。

«

<

1

>

»

50

# 在导航栏中选择“业务>分支网点管理>配置管理>CPE软件库”，进入 图 1-12 所示页面。

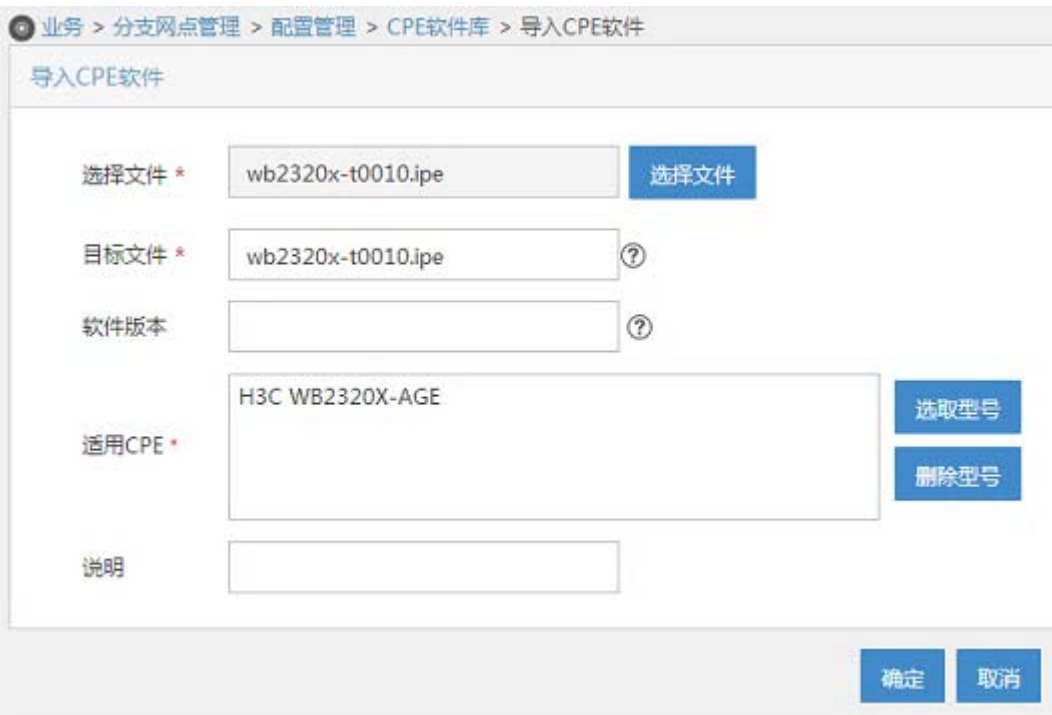


图1-12 CPE 软件库配置页面



# 单击<导入>按钮，进入 [图 1-13](#) 所示页面，选择CPE对应的软件版本。

图1-13 导入 CPE 软件库页面



(5) 在 ACS 上将配置文件与不同组别的 CPE 设备绑定，实现自动部署。

# 在导航栏中选择“业务>分支网点管理>配置管理>部署向导”，进入 [图 1-14](#) 所示页面。

图1-14 部署向导页面



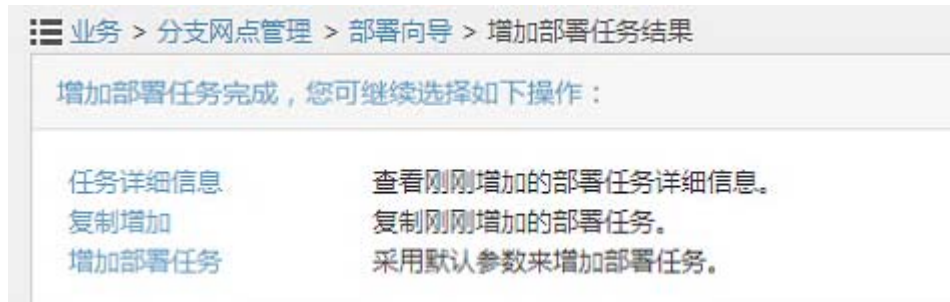
# 在导航栏中选择“业务>分支网点管理>配置管理>部署向导>自动部署CPE配置>按CPE”，并将其选择部署为“启动配置”，进入 [图 1-15](#) 所示页面。

图1-15 自动部署设备配置页面



- # 单击[选取 CPE]按钮，进入选取 CPE 型号页面，选取之前创建的 WB2320X-AGE 型号(当 CPE 数量太多时，可以根据 CPE 分组进行查询)，单击<确定>按钮完成选择操作。
- # 完成选择页面后将返回自动部署设备配置页面，单击<确定>按钮完成部署任务的创建。

图1-16 部署任务创建成功页面



对于机房 B 中的三台设备，配置步骤基本类似。

### 3. 配置DHCP服务器

- (1) 配置地址池，为 CPE 设备分配 IP 地址和 DNS 服务器，此处以分配 10.185.10.0/24 网段的地址为例。

# 开启 DHCP 服务。

```
<DHCP_server> system-view
[DHCP_server] dhcp enable
```

# 配置 VLAN 接口 1 工作在 DHCP 服务器模式。

```
[DHCP_server] interface vlan-interface 1
[DHCP_server-Vlan-interface1] dhcp select server
[DHCP_server-Vlan-interface1] quit
```

# 配置不参与自动分配的 IP 地址（此处包括 DNS 服务器、ACS 服务器）。

```
[DHCP_server] dhcp server forbidden-ip 10.185.10.41
[DHCP_server] dhcp server forbidden-ip 10.185.10.60
```

# 配置 DHCP 地址池 0 的共有属性（网段、DNS 服务器地址）。

```
[DHCP_server] dhcp server ip-pool 0
[DHCP_server-dhcp-pool-0] network 10.185.10.0 mask 255.255.255.0
[DHCP_server-dhcp-pool-0] dns-list 10.185.10.60
```

- (2) 配置 option 43 选项。选项内容包括 ACS 的地址、用户名和密码。

# 将 ACS 的地址、用户名和密码转换成 ASCII 码。其中 URL 地址对应的 ASCII 码为 68 74 74 70 3A 2F 2F 61 63 73 2E 64 61 74 61 62 61 73 65 3A 39 30 39 30 2F 61 63 73。用户名 admin 对应的 ASCII 码为 76 69 63 6B 79，密码 12345 对应的 ASCII 码为 31 32 33 34 35。

```
[DHCP_server-dhcp-pool-0] option 43 hex
013B687474703A2F2F6163732E64617461626173653A393039302F616373207669636B79203132333435
```

### 4. 配置DNS服务器

在 DNS 服务器上需要配置域名和地址的映射，将 http://acs.database:9090 地址映射为 http://10.185.1.41:9090。具体配置方法请参见您使用的 DNS 服务器软件手册。

## 5. 将CPE接入网络

将 CPE 上电并连接网线后，CPE 会先通过 DHCP server 获取 IP 地址和连接 ACS 所需的信息，再按照 CWMP 协议的流程自动从 ACS 处获取配置文件。

## 6. 验证配置

# 在导航栏中选择“业务>分支网点管理>资源管理>CPE 交互记录”，进入设备交互记录查询页面，查看与序列号对应的设备是否已经完成部署配置的操作。

# 目 录

1 EAA.....	1-1
1.1 EAA简介.....	1-1
1.1.1 EAA框架.....	1-1
1.1.2 监控策略中的元素.....	1-2
1.1.3 EAA环境变量.....	1-4
1.2 配置用户自定义环境变量.....	1-5
1.3 配置监控策略.....	1-5
1.3.1 配置限制和指导.....	1-5
1.3.2 配置CLI监控策略.....	1-5
1.3.3 配置Tcl监控策略.....	1-7
1.4 暂停监控策略.....	1-8
1.5 EAA显示和维护.....	1-8
1.6 EAA典型配置举例.....	1-9
1.6.1 Tcl监控策略基本配置举例.....	1-9
1.6.2 CLI监控策略基本配置举例.....	1-10
1.6.3 使用环境变量的CLI监控策略配置举例.....	1-11

# 1 EAA

## 1.1 EAA简介

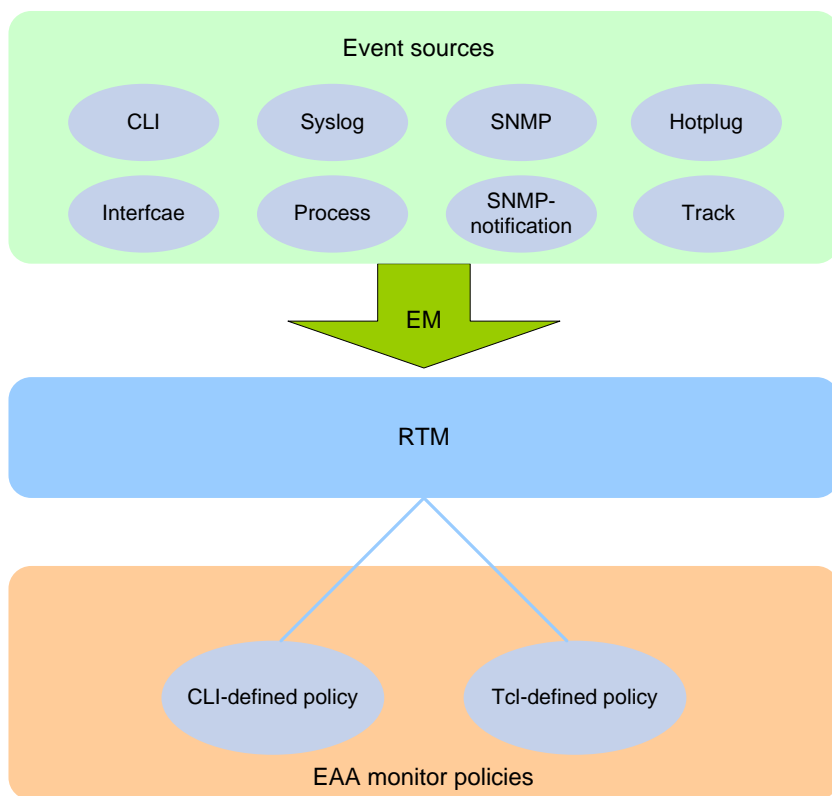
EAA（Embedded Automation Architecture，嵌入式自动化架构）是集成在系统软件中的一系列相关软件模块的总称。

使用 EAA 功能：

- 用户可以定制一系列监控策略，在策略中定义自己感兴趣的事件以及事件发生时的处理动作。监控策略被启用后，系统会实时监控设备的运行，当用户定制的事件发生时，就触发相应的监控策略并自动执行监控策略中的动作。
- 设备能够智能地监控多种事件，并做出灵活多变的响应，从而大大地提升系统的可维护性。

### 1.1.1 EAA框架

图1-1 EAA 框架示意图



EAA框架如 [图 1-1](#)所示，它包括事件源、EM（Event Monitor，事件监控）模块、RTM（Real-Time Event Manager，实时事件管理）模块和EAA监控策略。

#### 1. 事件源

事件源是系统中的软件或硬件模块，它们会触发事件。例如，CLI 事件源能触发命令行事件，Syslog 事件源能触发日志事件。

2. EM

EM 根据用户配置对事件源中发生的事件进行过滤匹配，匹配成功则通知 RTM 执行相应监控策略。当用户配置了多个监控策略，系统会创建多个 EM 模块，每个 EM 组件监控一个事件。

3. RTM

RTM 是 EAA 的核心部件，负责管理监控策略，包括监控策略的创建、状态变化和执行。

4. EAA监控策略

通过监控策略，用户可以定义自己感兴趣的事件以及事件发生时的处理动作。

监控策略有两种配置方式：一种是通过命令行来配置，一种是通过 Tcl 脚本来定义。通过命令行配置的监控策略称为基于 CLI 的监控策略，以下简称 CLI 监控策略；通过 Tcl 脚本定义的监控策略称为基于 Tcl 的监控策略，以下简称 Tcl 监控策略。

1.1.2 监控策略中的元素

每个监控策略中必须包含以下元素：事件、动作、用户角色、执行时间。

1. 事件

表示用来触发监控策略的事件。目前，EAA能够监控的事件类型如 [表 1-1](#) 所示。

表1-1 监控事件类型描述表

事件类型	描述
cli	监控命令行事件 配置该事件后，当用户输入指定的命令并对其进行特定操作（执行、帮助或者补全）就会触发策略执行
syslog	监控日志事件 配置该事件后，当系统在指定时间段内生成指定规格的日志信息时触发监控策略执行；RTM模块产生的日志不会触发策略执行
process	监控进程事件 配置该事件后，当指定进程（可以为用户命令行触发的或者系统自动触发的）发生指定状态变化（异常、关闭、启动或重启时），触发监控策略执行
hotplug	监控成员设备加入或离开IRF事件 配置该事件后，当成员设备加入或离开IRF，均会触发监控策略执行
interface	监控接口事件 接口事件中存在一个触发开关： <ol style="list-style-type: none"><li>配置该事件后，触发开关立即打开</li><li>当指定接口上的指定报文的数目达到 <b>start-op start-op start-val start-val</b> 参数指定的条件时，触发监控策略执行一次（第一次执行），并关闭触发开关，但系统会继续监控接口事件</li><li>当满足 <b>restart-op restart-op restart-val restart-val</b> 参数指定的条件时，才重新开启触发开关</li><li>如果指定接口上的指定报文的数目再次达到 <b>start-op start-op start-val start-val</b> 参数指定的条件时，则再次触发监控策略执行一次（第二次执行），并关闭触发开关，系统继续监控接口事件</li><li>如此循环</li></ol>

事件类型	描述
snmp	<p>监控SNMP节点值变化事件</p> <p>SNMP节点值变化事件中存在一个触发开关：</p> <ol style="list-style-type: none"> <li>1. 配置该事件后，触发开关立即打开</li> <li>2. 系统根据用户配置，定时轮询设备上某个节点的值，当该值达到 <b>start-op start-op start-val start-val</b> 指定的条件时，触发监控策略执行一次（第一次执行），并关闭触发开关，但系统会继续监控 SNMP 节点值变化事件</li> <li>3. 当节点值满足 <b>restart-op restart-op restart-val restart-val</b> 指定的条件时，才重新开启触发开关</li> <li>4. 当节点值再次达到 <b>start-op start-op start-val start-val</b> 指定的条件时，则再次触发监控策略执行一次（第二次执行），并关闭触发开关，系统继续监控 SNMP 节点值变化事件</li> <li>5. 如此循环</li> </ol>
snmp-notification	<p>监控SNMP告警事件</p> <p>配置该事件后，当系统生成一条告警，告警中携带的MIB对象（由oid参数指定）的值到达 <b>oid-val oid-val op op</b>指定的条件时，触发监控策略执行</p>
track	<p>监控Track事件</p> <p>配置该事件后，当关联的Track项状态由Positive变为Negative或者Negative变为Positive时，触发监控策略执行；如果关联多个Track项，则最后一个处于Positive（Negative）状态的Track项变为Negative（Positive）时，触发监控策略执行</p> <p>如果配置了抑制时间，触发策略的同时开始计时，定时器超时前，收到状态从Positive（Negative）变为Negative（Positive）的消息，直接丢弃，不会处理。直到定时器超时后，收到状态从Positive（Negative）变为Negative（Positive）的消息才处理，再一次触发策略执行</p>

## 2. 动作

表示事件发生时，监控策略将执行的动作。目前设备支持的动作有执行指定的命令行、生成一条指定内容的日志、主备倒换和重启。

## 3. 用户角色

表示执行监控策略的用户角色。用户角色中定义了允许用户操作哪些系统功能以及资源对象，设备支持的每条命令都有缺省用户角色：

- 如果监控策略中指定的用户角色权限比命令行的缺省用户角色的权限小，则不能执行该命令以及该命令后面的所有动作。
- 如果指定的用户角色不存在，则监控策略不能执行。
- 如果给某个监控策略配置了多个用户角色，则使用这些用户角色权限的并集去执行该策略。例如，给某策略配置了用户角色 A 和 B，如果策略中的动作是角色 A 或者 B 允许执行的，则策略可以执行；如果策略中存在角色 A 和 B 都不能执行的命令，则该命令以及该命令后面的所有动作都不能执行。

关于用户角色的详细描述请参见“基础配置指导”中的“RBAC”。

## 4. 运行时间

表示监控策略的运行时间，运行时间达到时即使策略没有执行完毕，也会立即停止执行策略。该元素用来限制策略的运行时间，以免策略长时间运行占用系统资源。



### 1.1.3 EAA环境变量

EAA 环境变量指的是专用于监控策略的环境变量。环境变量由<环境变量名、环境变量值>字对组成。在配置监控策略的动作时，我们可以在应该输入参数的地方输入“\$环境变量名”，表示此处需要引用环境变量值。系统在运行监控策略的时候，会自动用环境变量值去替代“\$环境变量名”。如果要修改监控参数的值，只需在系统视图下，修改环境变量值即可，而无需进入监控策略视图，修改监控策略下的具体配置。因此，定义和使用环境变量可以简化监控策略的配置，提高监控策略的灵活性和易用性。

目前，EAA 支持的环境变量包括内部环境变量和用户自定义环境变量。

#### 1. 内部环境变量

设备缺省支持的环境变量，用户不能创建、删除和修改。内部环境变量名均以“\_”开头，内部环境变量的值由系统决定。内部环境变量又包括两类：一类是公共环境变量，另一类是非公共环境变量。

- 公共环境变量可用于所有类型的事件：其中\_event\_id、\_event\_type、\_event\_type\_string 在系统启动时生成，关联的事件类型不同，其值不同，其值一旦确定不能更改；\_event\_time、\_event\_severity 在运行时产生。
- 非公共环境变量只能用于对应的事件，其值在事件触发时获得，它标示当前事件的部分信息。例如，HOTPLUG 事件对应的内部环境变量为“\_slot”，当 1 号成员设备加入或离开 IRF 的时候，环境变量“\_slot”的值为 1；当 2 号成员设备加入或离开 IRF 的时候，环境变量“\_slot”的值为 2。

目前EAA支持内部环境变量如 [表 1-2](#) 所示。

表1-2 内部环境变量描述表

事件	内部环境变量的名称	描述
CLI	_cmd	匹配上的命令
SYSLOG	_syslog_pattern	匹配的日志信息的内容
HOTPLUG	_slot	加入或离开IRF的成员设备的编号
INTERFACE	_ifname	接口的名称
SNMP	_oid	SNMP操作中携带的OID
	_oid_value	OID对应节点的值
SNMP TRAP	_oid	SNMP告警信息中携带的OID
PROCESS	_process_name	进程的名称
公共环境变量	_event_id	事件的ID
	_event_type	事件的类型
	_event_type_string	事件类型的描述
	_event_time	事件发生的时间
	_event_severity	事件的严重级别

## 2. 用户自定义环境变量

用户定义环境变量名可包含数字、字符或者“\_”，但不能以“\_”开头。用户自定义环境变量可用于所有类型的事件，其值由用户配置决定。用户定义环境变量可修改、删除。

## 1.2 配置用户自定义环境变量

### 1. 功能简介

使用本特性，用户可以自定义 EAA 环境变量的名称和值，以便定义监控策略时可以引用。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置用户自定义环境变量。

```
rtm environment var-name var-value
```

系统中缺省支持的内部环境变量，请参见 [表 1-2](#)。

## 1.3 配置监控策略

### 1.3.1 配置限制和指导

- 用户可创建多个监控策略，请确保同时启用的策略间动作不能冲突，因为当系统同时执行多个策略，且不同策略间动作有冲突时，执行结果是随机的。
- Tcl 监控策略和 CLI 监控策略的名称可以相同，相同类型的监控策略的名称不能相同。
- 同一个策略下，只能配置一个触发事件和运行时间。
- 同一监控策略下可配置多个用户角色，最多可以配置 64 个有效用户角色，超过该上限后，新配置的用户角色不会生效。

### 1.3.2 配置CLI监控策略

#### 1. 配置限制和指导

用户可为 CLI 监控策略配置多个动作，系统会按照动作的编号由小到大顺序执行这些动作。如果新配置的动作的编号和已有动作的编号相同，最后一次配置并且 **commit** 的生效。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) （可选）配置 EAA 监控的日志缓冲区的大小。

```
rtm event syslog buffer-size buffer-size
```

缺省情况下，EAA 监控的日志缓冲区的大小为 50000 条。

- (3) 创建 CLI 监控策略并进入 CLI 监控策略视图。

```
rtm cli-policy policy-name
```

- (4) 配置监控事件。

- 配置命令行事件。

```
event cli { async [ skip ] | sync } mode { execute | help | tab } pattern  
regular-exp
```

- 配置热插拔事件。

```
event hotplug [ insert | remove ] slot slot-number
```

- 配置接口事件。

```
event interface interface-type interface-number monitor-obj  
monitor-obj start-op start-op start-val start-val restart-op  
restart-op restart-val restart-val [ interval interval ]
```

- 配置进程事件。

```
event process { exception | restart | shutdown | start } [ name  
process-name [ instance instance-id ] ] [ slot slot-number ]
```

- 配置 SNMP 操作事件。

```
event snmp oid oid monitor-obj { get | next } start-op start-op  
start-val start-val restart-op restart-op restart-val restart-val  
[ interval interval ]
```

- 配置 SNMP 告警事件。

```
event snmp-notification oid oid oid-val oid-val op op [ drop ]
```

- 配置日志事件。

```
event syslog priority priority msg msg occurs times period period
```

- 配置 Track 事件。

```
event track track-list state { negative | positive } [ suppress-time  
suppress-time ]
```

缺省情况下，未配置事件。

多次执行 **event** 命令时，最后一次配置并且 **commit** 的生效。

#### (5) 配置动作。请至少选择以下一项进行配置。

- 配置事件发生时执行指定的命令行。

```
action number cli command-line
```

缺省情况下，监控策略下未配置 CLI 动作。

- 配置事件发生时执行重启操作。

```
action number reboot [ slot slot-number ]
```

缺省情况下，监控策略下未配置重启动作。

- 配置事件发生时进行主备倒换。

```
action number switchover
```

缺省情况下，监控策略下未配置主备倒换动作。

- 配置事件发生时发送指定的 LOG。

```
action number syslog priority priority facility local-number msg  
msg-body
```

缺省情况下，监控策略下未配置日志动作。

- (6) (可选) 配置执行 CLI 监控策略时使用的用户角色。

**user-role** *role-name*

缺省情况下，执行 CLI 监控策略时使用的用户角色为创建该策略的用户的角色。

安全日志管理员角色与其它用户角色互斥：配置安全日志管理员角色后，系统会自动删除已配置的其它用户角色；配置其它用户角色后，系统会自动删除已配置的安全日志管理员角色。

- (7) (可选) 配置 CLI 监控策略执行动作的持续时间。

**running-time** *time*

缺省情况下，CLI 监控策略执行动作的持续时间为 20 秒。

多次执行本命令时，最后一次配置并且 **commit** 的生效。

- (8) 启用 CLI 监控策略。

**commit**

缺省情况下，CLI 监控策略未被启用。

必须执行该命令后，CLI 监控策略下的配置才会生效。

### 1.3.3 配置Tcl监控策略

#### 1. Tcl脚本内容简介

按内容 Tcl 脚本可以分为两大部分：首行和其它行。

- (1) 首行

首行用于定义事件、用户角色和运行时间。用户创建并启用 Tcl 监控策略后，设备会立即解析 Tcl 脚本首行，并下发、生效。

Tcl 脚本首行的格式为：**::comware::rtm::event\_register** *event-type* *arg1* *arg2* *arg3* ... **user-role** *role-name1* | [ **user-role** *role-name2* | [ ... ] ]  
[ **running-time** *running-time* ]。其中：

- *event-type* 用来指定事件的类型，取值同 CLI 监控策略支持的事件类型，详情请参见 [表 1-1](#)。
- **arg** 用来指定事件参数，具体要求与相应 **event** 命令中的参数规格一致，请参考“网络管理和监控命令参考”中的“EAA”。当 **arg** 内容中包含空格时，请在 **arg** 内容的两端添加英文格式的双引号，形如“a b c”。
- **user-role** 用来指定执行脚本的用户角色，参数含义和配置要求同 CLI 监控策略。
- **running-time** 用来指定动作执行的最大时间，参数含义和配置要求同 CLI 监控策略。未指定该参数时，表示不限制 Tcl 监控策略的运行时间。

- (2) 其它行

从 Tcl 脚本的第二行开始，定义监控策略触发时将执行的动作。用户可使用多行定义多个动作，系统会按照 Tcl 脚本中配置的先后顺序执行这些动作。Tcl 监控策略支持如下三类动作：

- Tcl 语言标准命令。
- EAA 模块的 **switchover** 和 **syslog** 动作。配置该类动作时，请使用格式：**::comware::rtm::action switchover** 或 **::comware::rtm::action syslog**  
**priority** *priority* **facility** *local-number* **msg** *msg-body*。 *priority*、

*local-number*、*msg-body* 参数的详细描述请参考“网络管理和监控命令参考/EAA”中的 **action syslog** 命令。

- Comware 的其它命令行。配置该类动作时，每行为可执行的 Comware 命令行即可。

## 2. 配置限制和指导

Tcl 监控策略启用后，不允许修改 Tcl 脚本。如需修改，请先停用 Tcl 监控策略，修改后，再启用 Tcl 监控策略。否则，Tcl 监控策略将不能运行。

## 3. 配置步骤

- (1) 使用 FTP 或者 TFTP 功能将 Tcl 脚本下载到设备上，FTP 及 TFTP 具体配置请参见“基础配置指导”中的“FTP 和 TFTP”。
- (2) 创建并启用 Tcl 监控策略。

- a. 进入系统视图。

**system-view**

- b. 创建并启用 Tcl 监控策略，并将该策略与 Tcl 脚本绑定。

**rtm tcl-policy policy-name tcl-filename**

缺省情况下，不存在 Tcl 监控策略。

配置 Tcl 监控策略时，*tcl-filename* 请使用相对路径并确保所有的成员设备上都存在该脚本，以免主备倒换或脚本所在成员设备离开 IRF 后，策略无法执行。

# 1.4 暂停监控策略

## 1. 功能简介

使用本特性可以暂停运行所有的监控策略。如果配置本特性时，某个监控策略正在执行动作，则设备会等待该监控策略的所有动作全部执行完毕后，再暂停运行该监控策略。

## 2. 配置限制和指导

如果要将暂停运行的所有策略恢复运行，请使用 **undo rtm scheduler suspend** 命令。

## 3. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 暂停运行所有的监控策略。

**rtm scheduler suspend**

# 1.5 EAA显示和维护

在完成上述配置后，在任意视图下执行 **display** 命令可以显示配置后 EAA 的运行情况，通过查看显示信息验证配置的效果。

表1-3 EAA 显示和维护

操作	命令
显示CLI监控策略下生效的具体配置	<b>display current-configuration</b>

操作	命令
显示用户自定义的EAA环境变量配置	<b>display rtm environment</b> [ <i>var-name</i> ]
显示监控策略的相关信息	<b>display rtm policy</b> { <b>active</b>   <b>registered</b> [ <b>verbose</b> ] } [ <i>policy-name</i> ]
显示CLI监控策略下生效的具体配置（请在CLI监控策略视图下执行该命令）	<b>display this</b>

## 1.6 EAA典型配置举例

### 1.6.1 Tcl监控策略基本配置举例

#### 1. 组网需求

配置一个 Tcl 监控策略，当设备执行包含字符串 **display this** 的命令时，让系统自动发送日志信息 **rtm\_tcl\_test is running**。

#### 2. 组网图

图1-2 Tcl 监控策略基本配置举例组网图



#### 3. 配置步骤

# 在 PC 上使用写字板或者 word 编辑 Tcl 策略脚本 **rtm\_tcl\_test.tcl**，内容如下：

```

::comware::rtm::event_register cli sync mode execute pattern display this user-role
network-admin
::comware::rtm::action syslog priority 1 facility local4 msg rtm_tcl_test is running

```

以上脚本的含义为：当设备执行包含字符串 **display this** 的命令时，让系统自动发送日志信息 **rtm\_tcl\_test is running**。

# 通过 TFTP 将 **rtm\_tcl\_test.tcl** 下载到设备上。

```
<Sysname> tftp 1.2.1.1 get rtm_tcl_test.tcl
```

# 创建并启用 Tcl 监控策略，并将它和 Tcl 脚本 **rtm\_tcl\_test.tcl** 绑定。

```

<Sysname> system-view
[Sysname] rtm tcl-policy test rtm_tcl_test.tcl
[Sysname] quit

```

#### 4. 验证配置

# 通过 **display rtm policy registered** 命令可以看到存在策略名为 **test**，策略类型为 Tcl 的策略。

```

<Sysname> display rtm policy registered
Total number: 1
Type   Event      TimeRegistered      PolicyName
TCL    CLI          Aug 29 14:54:50 2013 test

```

# 打开允许日志输出的开关，执行 **display this** 命令，有 **rtm\_tcl\_test is running** 日志输出，同时有策略运行成功的日志输出。

```
<Sysname> terminal monitor
<Sysname> display this
#
return
<Sysname>%Jun  4 15:02:30:354 2013 Sysname RTM/1/RTM_ACTION: rtm_tcl_test is running
%Jun  4 15:02:30:382 2013 Sysname RTM/6/RTM_POLICY: TCL policy test is running successfully.
```

## 1.6.2 CLI监控策略基本配置举例

### 1. 组网需求

配置一个 CLI 监控策略，当设备执行包含数字、字母（大小写均可）的命令行的帮助操作时，让系统自动发送日志信息 **hello world**，并创建一个 VLAN。

### 2. 配置步骤

# 创建 CLI 监控策略，名称为 **test**。

```
<Sysname> system-view
[Sysname] rtm cli-policy test
```

# 配置监控事件：监控包含数字、字母（大小写均可）的命令行的帮助。

```
[Sysname-rtm-test] event cli async mode help pattern [a-zA-Z0-9]
```

# 事件发生时，发送优先级为 **4**，日志记录工具为 **local3**，信息为 **hello world** 的日志。

```
[Sysname-rtm-test] action 0 syslog priority 4 facility local3 msg "hello world"
```

# 事件发生时，进入系统视图。

```
[Sysname-rtm-test] action 2 cli system-view
```

# 事件发生时，创建 **VLAN 2**。

```
[Sysname-rtm-test] action 3 cli vlan 2
```

# 配置 CLI 监控策略执行动作的持续时间为 **2000** 秒。

```
[Sysname-rtm-test] running-time 2000
```

# 配置用户角色 **network-admin** 具有执行该策略的权限。

```
[Sysname-rtm-test] user-role network-admin
```

# 确认执行该策略。

```
[Sysname-rtm-test] commit
```

### 3. 验证配置

# 通过 **display rtm policy registered** 查看，可以看到策略名为 **test**，策略类型为 CLI 的策略。

```
[Sysname-rtm-test] display rtm policy registered
Total number: 1
```

Type	Event	TimeRegistered	PolicyName
CLI	CLI	Aug 29 14:56:50 2013	test

# 打开允许日志输出的开关，并对包含字母 **d** 的命令进行帮助，可以看到有信息为 “**hello world**” 的日志输出，同时有策略运行成功的日志输出。

```
[Sysname-rtm-test] return
<Sysname> terminal monitor
```

```

<Sysname> d?
  debugging
  delete
  diagnostic-logfile
  dir
  display

<Sysname>d%May  7 02:10:03:218 2013 Sysname RTM/4/RTM_ACTION: "hello world"
%May  7 02:10:04:176 2013 Sysname RTM/6/RTM_POLICY: CLI policy test is running successfully.

```

### 1.6.3 使用环境变量的CLI监控策略配置举例

#### 1. 组网需求

配置一个 CLI 监控策略，用于实现：当用户执行带有字符串 `loopback0` 的命令时，

- 系统会自动创建 `LoopBack0` 接口。
- 将该接口的 IP 地址配置为 `1.1.1.1/24`。
- 将匹配到的命令行通过日志打印输出。

#### 2. 配置步骤

# 创建环境变量 `loopback0IP`，将其值配置为 `1.1.1.1`。

```

<Sysname> system-view
[Sysname] rtm environment loopback0IP 1.1.1.1

```

# 创建 CLI 监控策略，名称为 `test`。

```

[Sysname] rtm cli-policy test

```

# 配置监控事件：用户执行带有字符串 `loopback0` 的命令。

```

[Sysname-rtm-test] event cli async mode execute pattern loopback0

```

# 事件发生时，系统会自动创建 `LoopBack0` 接口，将该接口的 IP 地址配置为 `1.1.1.1/24`，并将匹配到的命令行通过日志打印输出。

```

[Sysname-rtm-test] action 0 cli system-view
[Sysname-rtm-test] action 1 cli interface loopback 0
[Sysname-rtm-test] action 2 cli ip address $loopback0IP 24
[Sysname-rtm-test] action 3 syslog priority 7 facility local7 msg $_cmd
[Sysname-rtm-test] user-role network-admin
[Sysname-rtm-test] commit
[Sysname-rtm-test] return
<Sysname>

```

#### 3. 验证配置

# 打开允许日志输出的开关，当用户执行带有字符串 `loopback0` 的命令时，系统会将匹配到的命令行通过日志打印输出，同时有策略运行成功的日志输出。

```

<Sysname> terminal monitor
<Sysname> system-view
[Sysname] interface loopback0
[Sysname]

%Jan  3 09:46:10:592 2014 Sysname RTM/7/RTM_ACTION: loopback0
%Jan  3 09:46:10:613 2014 Sysname RTM/6/RTM_POLICY: CLI policy test is running successfully.

```



# 显示接口信息，发现已存在 LoopBack0 接口，IP 地址为 1.1.1.1。

```
[Sysname] display interface loopback brief
```

Brief information on interfaces in route mode:

Link: ADM - administratively down; Stby - standby

Protocol: (s) - spoofing

Interface	Link	Protocol	Primary IP	Description
Loop0	UP	UP(s)	1.1.1.1	

```
[Sysname]
```

# 目 录

1 进程监控和维护.....	1-1
1.1 进程监控和维护简介.....	1-1
1.2 进程监控和维护任务简介.....	1-1
1.3 监控和维护进程.....	1-1
1.4 监控和维护用户态进程.....	1-2
1.4.1 功能简介.....	1-2
1.4.2 配置core文件生成功能 .....	1-2
1.4.3 用户态进程显示和维护 .....	1-3
1.5 监控与维护内核线程.....	1-3
1.5.1 配置内核线程死循环检测功能 .....	1-3
1.5.2 配置内核线程饿死检测功能 .....	1-4
1.5.3 内核线程显示和维护 .....	1-5

# 1 进程监控和维护

## 1.1 进程监控和维护简介

设备的系统软件基于 Linux 内核，各个网络服务功能分别运行各自的进程，实现模块化。运行在用户空间的进程称为用户态进程，与用户态进程相对的是内核线程，内核线程运行在内核态空间。

- 系统软件的绝大部分程序是用户态进程。每个用户态进程拥有独立的进程空间，单个进程的异常不会影响系统其他进程，从而提高了系统的可靠性。通常情况下，系统会自动监控用户态进程，不需要用户干预。当单个用户态进程中包含多个独立或半独立的活动，可以将这些活动拆分成多个线程。系统软件支持多线程并发和抢占，多个线程分工合作共同实现某个功能。一个进程是否包含多个线程，由软件实现需要决定。
- 内核线程用来执行系统软件内核代码和系统调用。它拥有比用户态进程更高的安全级别，当内核线程发生异常，通常系统会完全崩溃。用户可以使用命令行来监控内核线程的运行状态。

## 1.2 进程监控和维护任务简介

进程监控和维护配置任务如下：

- 监控和维护用户态进程
  - [监控和维护进程](#)  
本节中罗列的命令对用户态进程和内核线程均适用。
  - [监控和维护用户态进程](#)  
本节中罗列的命令仅对用户态进程适用。
- 监控和维护内核线程
  - [监控和维护进程](#)  
本节中罗列的命令对用户态进程和内核线程均适用。
  - [监控与维护内核线程](#)  
本节中罗列的命令仅对内核线程适用。

## 1.3 监控和维护进程

### 1. 功能简介

本节中罗列的命令对用户态进程和内核线程均适用。使用这些命令，可以进行如下操作：

- 显示内存的整体使用情况。
- 显示系统当前运行了哪些进程，每个进程占用了多少内存和多少 CPU 资源。
- 定位异常进程。如果某个进程占用内存或者CPU资源过多，则确认该进程为异常源。如果异常源是用户态进程，可参考“[1.4 监控和维护用户态进程](#)”来进一步定位解决问题；如果异常源是内核线程，可参考“[1.5 监控与维护内核线程](#)”来进一步定位解决问题。

## 2. 配置步骤

请在任意视图下执行以下命令，来监控和维护进程。

表1-1 监控和维护进程

操作	命令
显示系统内存使用情况（本命令的详细描述请参见“基础配置命令参考中的“设备管理”）	<b>display memory</b> [ <b>summary</b> ] [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
显示进程的状态信息	<b>display process</b> [ <b>all</b>   <b>job</b> <i>job-id</i>   <b>name</b> <i>process-name</i> ] [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
显示所有进程的CPU占有率信息	<b>display process cpu</b> [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
监控进程运行状态	<b>monitor process</b> [ <b>dumbtty</b> ] [ <b>iteration</b> <i>number</i> ] [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
监控线程运行状态	<b>monitor thread</b> [ <b>dumbtty</b> ] [ <b>iteration</b> <i>number</i> ] [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]

## 1.4 监控和维护用户态进程

### 1.4.1 功能简介

当用户态进程运行异常，可使用本特性来进一步监控用户态进程、定位用户态进程故障。

### 1.4.2 配置core文件生成功能

#### 1. 功能简介

开启用户态进程的 **core** 文件生成功能，并配置能生成的 **core** 文件的最大个数后，用户态进程异常重启一次，就会产生一个 **core** 文件并记录用户态进程的异常信息。如果生成的 **core** 文件的数目达到最大值，则不再生成新的 **core** 文件。软件开发和维护人员能够根据 **core** 文件的内容来定位异常的原因和异常的位置。

#### 2. 配置限制和指导

因为生成的 **core** 文件会占用系统存储资源，如果用户对某些用户态进程的异常退出不关心，可以关闭这些用户态进程的 **core** 文件记录功能。

#### 3. 配置步骤

以下命令均在用户视图下执行。

- (1) （可选）配置 **core** 文件的保存路径。

```
exception filepath directory
```

缺省情况下，**Core** 文件的保存路径为 **flash**。

- (2) 开启/关闭用户态进程异常时生成 **core** 文件的功能，以及配置能生成的 **core** 文件的最大个数。

```
process core { maxcore value | off } { job job-id | name process-name }
```

缺省情况下，用户态进程在首次异常时会生成 **core** 文件，后续异常不再生成 **core** 文件。即 **maxcore** 的最大数值为 1。

### 1.4.3 用户态进程显示和维护

在任意视图下，通过用户态进程的显示信息，用户可以更好的了解用户态进程的实时运行状态；同时，当出现系统异常繁忙或者资源消耗异常等故障时，显示信息可以帮助用户确认出现故障的功能点，以便尽快进行功能的恢复。

在用户视图下，执行 **reset** 命令，可以清除用户态进程的指定信息。

表1-2 用户态进程显示和维护

操作	命令
显示用户态进程异常时的上下文信息	<b>display exception context</b> [ <b>count</b> <i>value</i> ] [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
显示core文件的保存路径	<b>display exception filepath</b> [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
显示所有用户态进程的日志信息	<b>display process log</b> [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
显示所有用户态进程的代码段、数据段以及堆栈等的内存使用信息	<b>display process memory</b> [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
显示用户态进程堆内存的使用情况	<b>display process memory heap job</b> <i>job-id</i> [ <b>verbose</b> ] [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
显示从指定地址开始的内存空间的内容	<b>display process memory heap job</b> <i>job-id</i> <b>address</b> <i>starting-address</i> <b>length</b> <i>memory-length</i> [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
显示指定大小已使用内存块的地址	<b>display process memory heap job</b> <i>job-id</i> <b>size</b> <i>memory-size</i> [ <b>offset</b> <i>offset-size</i> ] [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
清除用户态进程异常时记录的上下文信息	<b>reset exception context</b> [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]

## 1.5 监控与维护内核线程

### 1.5.1 配置内核线程死循环检测功能

#### 1. 功能简介

在内核态空间中，所有资源都是共享的，多个内核线程之间通过任务调度协调工作。如果某个内核线程长时间一直占用 **CPU**，就会导致其它内核线程获取不到运行机会，整个系统挂死，我们称这种现象为死循环。

开启内核线程死循环检测功能后，如果系统发现某内核线程在指定时间内一直占用 **CPU**，则判定该内核线程为死循环。系统会记录一条死循环信息供管理员查询，并自动重启整个系统来解除死循环。

#### 2. 配置限制和指导

对于内核线程死循环检测功能，建议用户使用缺省配置即可。如果确实需要修改缺省配置，请在工程师的指导下进行，以免引起系统异常。

### 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启内核线程死循环检测功能。

```
monitor kernel deadlock enable [ slot slot-number [ cpu cpu-number [ core  
core-number&<1-64> ] ] ]
```

缺省情况下，内核线程死循环检测功能处于开启状态。

- (3) （可选）配置判定内核线程是否死循环的时长。

```
monitor kernel deadlock time time [ slot slot-number [ cpu cpu-number ] ]
```

缺省情况下，当某内核线程连续运行超过 20 秒钟，则判定为死循环。

- (4) （可选）配置不检测指定内核线程是否发生了死循环。

```
monitor kernel deadlock exclude-thread tid [ slot slot-number [ cpu  
cpu-number ] ]
```

缺省情况下，开启内核线程死循环检测功能后，会监控所有内核线程是否发生了死循环。

- (5) （可选）配置内核线程死循环后系统执行的操作。

```
monitor kernel deadlock action { reboot | record-only } [ slot  
slot-number [ cpu cpu-number ] ]
```

缺省情况下，系统检测到内核线程死循环后，执行的操作为 **reboot**。

## 1.5.2 配置内核线程饿死检测功能

### 1. 功能简介

如果内核线程本身的触发条件没有达到，会导致该内核线程在一段时间内一直得不到调度，我们称这种现象为饿死。

开启内核线程饿死检测功能后，当系统检测到某内核线程饿死时，会记录一条饿死信息供管理员查询。

内核线程饿死并不会影响整个系统的运行，当触发条件达到，处于饿死状态的内核线程会自动执行。

### 2. 配置限制和指导

建议用户不要随意配置内核线程饿死检测功能。如果确实需要配置，请在工程师的指导下进行，以免引起系统异常。

### 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启内核线程饿死检测功能。

```
monitor kernel starvation enable [ slot slot-number [ cpu cpu-number ] ]
```

缺省情况下，内核线程饿死检测功能处于关闭状态。

- (3) （可选）配置判定内核线程是否饿死的时长。

```
monitor kernel starvation time time [ slot slot-number [ cpu  
cpu-number ] ]
```

缺省情况下，当某内核线程在 120 秒内一直没有运行，则认为该内核线程被饿死。

- (4) （可选）配置不检测指定内核线程是否发生了饿死。

```
monitor kernel starvation exclude-thread tid [ slot slot-number [ cpu cpu-number ] ]
```

缺省情况下，开启内核线程饿死检测功能后，会监控所有内核线程是否发生了饿死。

### 1.5.3 内核线程显示和维护

在任意视图下，通过内核线程的显示信息，用户可以更好的了解内核线程的实时运行状态；同时，当出现系统异常繁忙或者资源消耗异常等故障时，显示信息可以帮助用户确认出现故障的功能点，以便尽快进行功能的恢复。

在用户视图下，执行 **reset** 命令，可以清除内核线程的统计信息。

表1-3 内核线程显示和维护

操作	命令
显示内核线程死循环监控参数配置	<b>display kernel deadloop configuration</b> [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
显示内核线程死循环信息	<b>display kernel deadloop show-number</b> [ <i>offset</i> ] [ <b>verbose</b> ] [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
显示内核线程的异常信息	<b>display kernel exception show-number</b> [ <i>offset</i> ] [ <b>verbose</b> ] [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
显示内核线程的重启信息	<b>display kernel reboot show-number</b> [ <i>offset</i> ] [ <b>verbose</b> ] [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
显示内核线程饿死监控参数配置	<b>display kernel starvation configuration</b> [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
显示内核线程饿死信息	<b>display kernel starvation show-number</b> [ <i>offset</i> ] [ <b>verbose</b> ] [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
清除内核线程死循环信息	<b>reset kernel deadloop</b> [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
清除内核线程的异常信息	<b>reset kernel exception</b> [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
清除内核线程重启信息	<b>reset kernel reboot</b> [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]
清除内核线程饿死信息	<b>reset kernel starvation</b> [ <b>slot</b> <i>slot-number</i> [ <b>cpu</b> <i>cpu-number</i> ] ]

# 目 录

1 端口镜像.....	1-1
1.1 端口镜像简介.....	1-1
1.1.1 基本概念.....	1-1
1.1.2 端口镜像的分类.....	1-2
1.1.3 本地端口镜像.....	1-2
1.1.4 二层远程端口镜像.....	1-2
1.2 镜像配置限制和指导.....	1-4
1.3 配置本地端口镜像.....	1-4
1.3.1 配置限制和指导.....	1-4
1.3.2 配置任务简介.....	1-4
1.3.3 创建本地镜像组.....	1-4
1.3.4 配置镜像源.....	1-5
1.3.5 配置镜像目的.....	1-5
1.4 利用远程镜像VLAN实现本地镜像支持多目的端口.....	1-6
1.5 配置二层远程端口镜像.....	1-7
1.5.1 配置限制和指导.....	1-7
1.5.2 反射端口方式二层远程端口镜像配置任务简介.....	1-8
1.5.3 出端口方式二层远程端口镜像配置任务简介.....	1-8
1.5.4 创建远程目的镜像组.....	1-8
1.5.5 配置目的端口.....	1-9
1.5.6 配置远程镜像VLAN.....	1-9
1.5.7 将目的端口加入远程镜像VLAN.....	1-10
1.5.8 创建远程源镜像组.....	1-10
1.5.9 配置镜像源.....	1-10
1.5.10 配置反射端口.....	1-11
1.5.11 配置出端口.....	1-12
1.6 端口镜像显示和维护.....	1-13
1.7 端口镜像典型配置举例.....	1-13
1.7.1 本地端口镜像配置举例.....	1-13
1.7.2 二层远程端口镜像配置举例（反射端口方式）.....	1-14
1.7.3 二层远程端口镜像配置举例（出端口方式）.....	1-16



2 流镜像..... 2-1

2.1 流镜像简介..... 2-1

2.2 流镜像配置限制和指导..... 2-1

2.3 流镜像配置任务简介..... 2-1

2.4 配置流分类..... 2-1

2.5 配置流行为..... 2-2

2.6 配置QoS策略..... 2-2

2.7 应用QoS策略..... 2-3

2.7.1 基于接口应用..... 2-3

2.7.2 基于VLAN应用..... 2-3

2.7.3 基于全局应用..... 2-3

2.8 流镜像典型配置举例..... 2-4

2.8.1 流镜像基本组网配置举例..... 2-4

# 1 端口镜像

## 1.1 端口镜像简介

端口镜像通过将指定端口的报文复制到与数据监测设备相连的端口，使用户可以利用数据监测设备分析这些复制过来的报文，以进行网络监控和故障排除。

### 1.1.1 基本概念

#### 1. 镜像源

镜像源是指被监控的对象，配置为监控对象的端口为源端口。经镜像源收发的报文会被复制一份到与数据监测设备相连的端口，用户就可以对这些报文（称为镜像报文）进行监控和分析了。

#### 2. 源设备

镜像源所在的设备称为源设备。

#### 3. 镜像目的

镜像目的是指镜像报文所要到达的目的地，即与数据监测设备相连的端口，该端口称为目的端口。目的端口会将镜像报文转发给与之相连的数据监测设备。

由于一个目的端口可以同时监控多个镜像源，因此在某些组网环境下，目的端口可能收到对同一报文的多份拷贝。例如，目的端口 **Port A** 同时监控同一台设备上的源端口 **Port B** 和 **Port C** 收发的报文，如果某报文从 **Port B** 进入该设备后又从 **Port C** 发送出去，那么该报文将被复制两次给 **Port A**。

#### 4. 目的设备

目的端口所在的设备称为目的设备。

#### 5. 镜像方向

镜像方向是指在镜像源上可复制哪些方向的报文：

- 入方向：是指仅复制镜像源收到的报文。
- 出方向：是指仅复制镜像源发出的报文。
- 双向：是指对镜像源收到和发出的报文都进行复制。

#### 6. 镜像组

镜像组是一个逻辑上的概念，镜像源和镜像目的都要属于某一个镜像组。根据具体的实现方式不同，镜像组可分为本地镜像组、远程源镜像组和远程目的镜像组。

#### 7. 反射端口、出端口和远程镜像VLAN

反射端口、出端口和远程镜像 **VLAN** 都是在二层远程端口镜像的实现过程中用到的概念。远程镜像 **VLAN** 是将镜像报文从源设备传送至目的设备的专用 **VLAN**；反射端口和出端口都位于源设备上，都用来将镜像报文发送到远程镜像 **VLAN** 中。

在配置端口镜像的设备上，除源端口、目的端口、反射端口、出端口外的其他端口统称为普通端口。

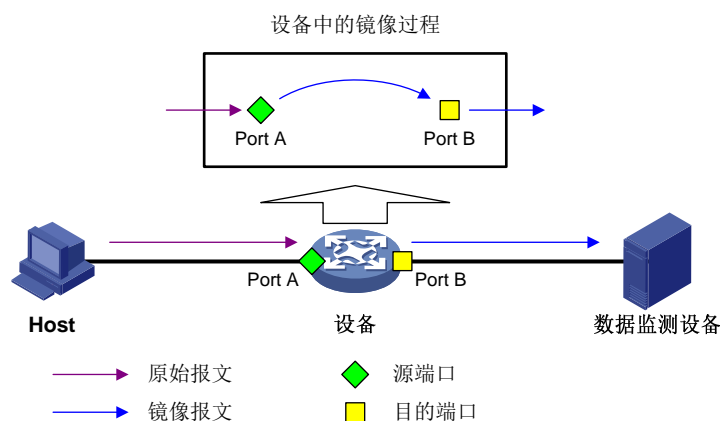
### 1.1.2 端口镜像的分类

根据镜像源与镜像目的是否位于同一台设备上，可以将端口镜像分为：

- 本地端口镜像：当源设备与数据监测设备直接相连时，源设备同时作为目的的设备，即由本设备将镜像报文转发至数据检测设备，该端口镜像称为本地端口镜像。
- 远程端口镜像：当源设备与数据监测设备不直接相连时，与数据监测设备直接相连的设备作为目的的设备，源设备需要将镜像报文复制一份至目的设备，然后由目的设备将镜像报文转发至数据监测设备，该端口镜像称为远程端口镜像。因源设备与目的设备之间通过二层网络进行连接，因此远程端口镜像又称为二层远程端口镜像

### 1.1.3 本地端口镜像

图1-1 本地端口镜像示意图



如 [图 1-1](#) 所示，现在需要设备将进入端口Port A的报文复制一份，从端口Port B将报文转发给数据监测设备。为满足该需求，可以配置本地镜像组，其中源端口为Port A，镜像方向为入方向，目的端口为Port B。

### 1.1.4 二层远程端口镜像

对于二层远程端口镜像，镜像源和镜像目的分属于不同设备上的不同镜像组：

- 远程源镜像组：镜像源所在的镜像组。
- 远程目的镜像组：镜像目的所在的镜像组。
- 中间设备：位于源设备与目的设备之间的设备。

二层远程端口镜像的实现包括反射端口方式和出端口方式。

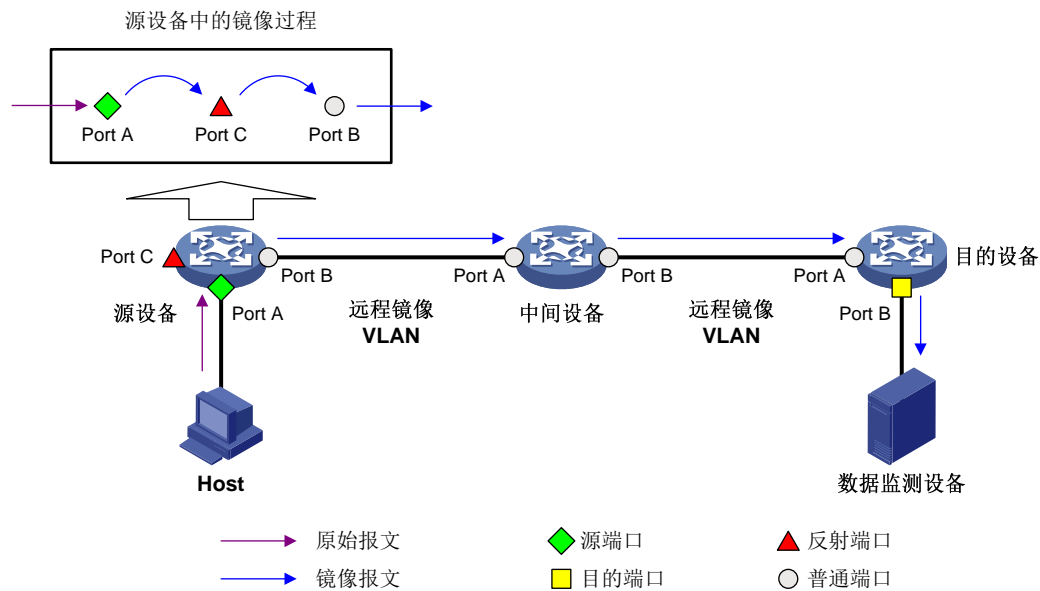
#### 1. 反射端口方式二层远程端口镜像

反射端口方式二层远程端口镜像报文的转发过程如 [图 1-2](#) 所示。

- (1) 源设备将进入镜像源的报文复制一份给反射端口。
- (2) 反射端口将镜像报文在远程镜像 VLAN 中广播。
- (3) 镜像报文经由中间设备转发至目的设备。

- (4) 目的设备收到该报文后判别其 VLAN ID，若与远程镜像 VLAN 的 VLAN ID 相同，则将镜像报文通过目的端口转发给数据监测设备。

图1-2 反射端口方式二层远程端口镜像示意图

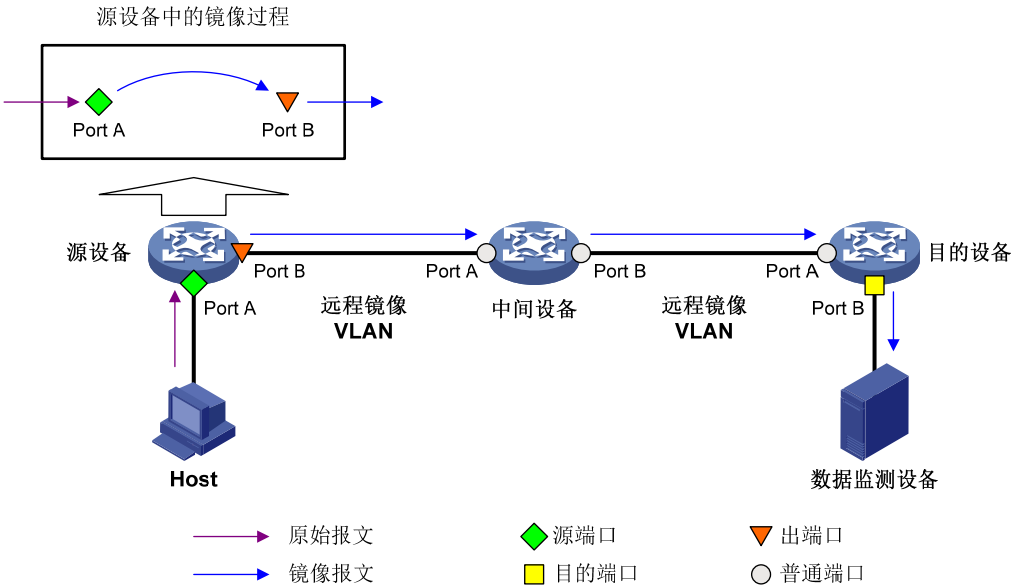


## 2. 出端口方式

出端口方式二层远程端口镜像报文的转发过程如 [图 1-3](#) 所示。

- (1) 源设备将进入镜像源的报文复制一份给出端口。
- (2) 出端口将镜像报文转发给中间设备。
- (3) 中间设备在远程镜像 VLAN 中广播，最终到达目的设备。
- (4) 目的设备收到该报文后判别其 VLAN ID，若与远程镜像 VLAN 的 VLAN ID 相同，则将镜像报文通过目的端口转发给数据监测设备。

图1-3 出端口方式二层远程端口镜像示意图



## 1.2 镜像配置限制和指导

对于二层远程端口镜像反射端口方式，源设备的反射端口将镜像报文在远程镜像 VLAN 中广播。因此，可以利用远程镜像 VLAN 的原理，在本地设备上创建远程源镜像组，并指定远程镜像 VLAN，同时将本设备上连接数据检测设备的多个端口加入该 VLAN，镜像报文在远程镜像 VLAN 中广播时便可以从这些端口中发送出去，实现将镜像报文输出到本地多个端口的需求。而对于出端口方式则无此实现。

对端口入方向的报文进行镜像，则镜像后的报文携带 VLAN Tag 的情况与原始报文保持一致；对端口出方向的报文进行镜像，则镜像后的报文始终携带报文在从源端口发送前所属 VLAN 的 VLAN Tag。

## 1.3 配置本地端口镜像

### 1.3.1 配置限制和指导

在完成镜像源和镜像目的配置之后，本地镜像组才能生效。

### 1.3.2 配置任务简介

本地端口镜像配置任务如下：

- (1) [创建本地镜像组](#)
- (2) [配置镜像源](#)
- (3) [配置镜像目的](#)

### 1.3.3 创建本地镜像组

- (1) 进入系统视图。

**system-view**

- (2) 创建本地镜像组。

**mirroring-group group-id local**

### 1.3.4 配置镜像源

#### 1. 配置限制和指导

配置源端口时，需要注意的是：

- 一个镜像组内可以配置多个源端口。
- 一个源端口无论作为单向源端口还是双向源端口，都只能加入一个镜像组。
- 源端口不能用作反射端口、出端口或目的端口。

#### 2. 配置源端口

- 在系统视图下配置源端口。

- a. 进入系统视图。

**system-view**

- b. 为本地镜像组配置源端口。

**mirroring-group group-id mirroring-port interface-list { both | inbound | outbound }**

缺省情况下，未为本地镜像组配置源端口。

- 在接口视图下配置源端口。

- a. 进入系统视图。

**system-view**

- b. 进入接口视图。

**interface interface-type interface-number**

- c. 配置当前端口为本地镜像组的源端口。

**mirroring-group group-id mirroring-port { both | inbound | outbound }**

缺省情况下，未配置当前端口为本地镜像组的源端口。

### 1.3.5 配置镜像目的

#### 1. 配置限制和指导

不能在目的端口上开启生成树协议，否则会影响镜像功能的正常使用。

从目的端口发出的报文包括镜像报文和其他端口正常转发来的报文。为了保证数据监测设备只对镜像报文进行分析，请将目的端口只用于端口镜像，不作其他用途。

一个镜像组内只能配置一个目的端口。

#### 2. 配置步骤

- 在系统视图下配置目的端口。

- a. 进入系统视图。

**system-view**

- b. 为本地镜像组配置目的端口。

```
mirroring-group group-id monitor-port interface-type  
interface-number
```

缺省情况下，未为本地镜像组配置目的端口。

- 在接口视图下配置目的端口。

- a. 进入系统视图。

```
system-view
```

- b. 进入接口视图。

```
interface interface-type interface-number
```

- c. 配置本端口为本地镜像组的目的端口。

```
mirroring-group group-id monitor-port
```

缺省情况下，未配置当前端口为本地镜像组的目的端口。

## 1.4 利用远程镜像VLAN实现本地镜像支持多目的端口

### 1. 功能简介

传统本地镜像配置方式仅支持在一个镜像组中配置一个目的端口，利用远程镜像 VLAN 的原理可以将一份镜像报文同时发送到多个目的端口。

在二层远程端口镜像中，镜像报文在远程镜像 VLAN 中以广播的方式发送，利用该原理，在本地设备上创建远程源镜像组和远程镜像 VLAN，并将本设备上连接数据检测设备的多个端口加入该 VLAN，镜像报文在远程镜像 VLAN 中广播时即可从这些端口中发送出去，实现将镜像报文发送到多个目的端口。

### 2. 配置限制和指导

请选择设备上未被使用的端口作为反射端口，不能在该端口上连接网线，否则会影响镜像功能的正常使用。

端口配置为反射口时，该端口上已存在的所有配置都将被清除；在被配置为反射口后，该端口上不能再配置其他业务功能。

不能将源端口加入到远程镜像 VLAN 中，否则会影响镜像功能的正常使用。

一个 VLAN 只能作为一个远程源镜像组的远程镜像 VLAN，且建议该 VLAN 只用于端口镜像，请不要在该 VLAN 上配置其他业务功能或创建对应的 VLAN 接口。

远程镜像 VLAN 必须为静态 VLAN，且在被配置成远程镜像 VLAN 后，该 VLAN 不能直接删除，必须先删除远程镜像 VLAN 的配置才能够删除该 VLAN。

### 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 创建远程源镜像组。

```
mirroring-group group-id remote-source
```

- (3) 为远程源镜像组配置源端口。请选择其中一项进行配置。

- 在系统视图下配置。

```
mirroring-group group-id mirroring-port mirroring-port-list { both | inbound | outbound }
```

- 请依次执行以下命令在接口视图下配置。

```
interface interface-type interface-number
```

```
mirroring-group group-id mirroring-port { both | inbound | outbound }  
quit
```

- (4) 为远程源镜像组配置反射端口。

```
mirroring-group group-id reflector-port reflector-port
```

缺省情况下，镜像组没有反射端口。

- (5) 创建远程镜像 VLAN 并进入 VLAN 视图。

```
vlan vlan-id
```

缺省情况下，镜像组没有远程镜像 VLAN。

- (6) 将镜像目的端口加入远程镜像 VLAN。

```
port interface-list
```

缺省情况下，新建 VLAN 中不包含任何端口。

- (7) 退出至系统视图。

```
quit
```

- (8) 为远程源镜像组配置远程镜像 VLAN。

```
mirroring-group group-id remote-probe vlan vlan-id
```

缺省情况下，镜像组没有远程镜像 VLAN。

## 1.5 配置二层远程端口镜像

### 1.5.1 配置限制和指导

二层远程端口镜像的配置需要分别在源设备和目的设备上进行；如果存在中间设备，则需要在中间设备上允许远程镜像 VLAN 通过，以确保源设备与目的设备之间的二层网络畅通。

请先配置目的设备，再配中间设备，最后配源设备，以保证镜像流量的正常转发。

在镜像报文从源设备到达目的设备的过程中，VLAN ID 不被修改或删除，否则二层远程镜像功能将失效。

在配置二层远程端口镜像时建议关闭 MVRP（Multiple VLAN Registration Protocol，多 VLAN 注册协议）功能，否则 MVRP 可能将远程镜像 VLAN 注册到不需要监控的端口上，导致目的端口收到不必要的报文。有关 MVRP 的详细介绍，请参见“二层技术-以太网交换配置指导”中的“MVRP”。

在一个镜像组中对同一个端口收发的报文进行双向镜像时，需要在源设备、中间设备和目的设备上关闭远程镜像 VLAN 的 MAC 地址学习功能，以保证镜像功能的正常进行。关于 MAC 地址学习功能的详细介绍，请参见“二层技术-以太网交换配置指导”中的“MAC 地址表”。

设备不支持将二层聚合接口的成员端口配置为目的端口。



## 1.5.2 反射端口方式二层远程端口镜像配置任务简介

### 1. 配置目的设备

目的设备配置任务如下：

- (1) [创建远程目的镜像组](#)
- (2) [配置目的端口](#)
- (3) [配置远程镜像VLAN](#)
- (4) [将目的端口加入远程镜像VLAN](#)

### 2. 配置源设备

源设备配置任务如下：

- (1) [创建远程源镜像组](#)
- (2) [配置镜像源](#)
- (3) [配置反射端口](#)
- (4) [配置远程镜像VLAN](#)

## 1.5.3 出端口方式二层远程端口镜像配置任务简介

### 1. 配置目的设备

目的设备配置任务如下：

- (1) [创建远程目的镜像组](#)
- (2) [配置目的端口](#)
- (3) [配置远程镜像VLAN](#)
- (4) [将目的端口加入远程镜像VLAN](#)

### 2. 配置源设备

源设备配置任务如下：

- (1) [创建远程源镜像组](#)
- (2) [配置镜像源](#)
- (3) [配置出端口](#)
- (4) [配置远程镜像VLAN](#)

## 1.5.4 创建远程目的镜像组

### 1. 配置限制和指导

仅目的设备需要进行本配置。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 创建远程目的镜像组。

```
mirroring-group group-id remote-destination
```

## 1.5.5 配置目的端口

### 1. 配置限制和指导

仅目的设备需要进行本配置。

不能在目的端口上开启生成树协议，否则会影响镜像功能的正常使用。

从目的端口发出的报文包括镜像报文和其他端口正常转发来的报文。为了保证数据监测设备只对镜像报文进行分析，请将目的端口只用于端口镜像，不作其他用途。

一个目的端口只能加入一个镜像组。

设备不支持将二层聚合接口配置为二层端口镜像的目的端口。

### 2. 在系统视图下配置目的端口

- (1) 进入系统视图。

```
system-view
```

- (2) 为远程目的镜像组配置目的端口。

```
mirroring-group group-id monitor-port interface-type interface-number
```

缺省情况下，未为远程镜像组配置目的端口。

### 3. 在接口视图下配置目的端口

- (1) 进入系统视图。

```
system-view
```

- (2) 进入接口视图。

```
interface interface-type interface-number
```

- (3) 配置本端口为远程目的镜像组的目的端口。

```
mirroring-group group-id monitor-port
```

缺省情况下，未配置当前端口为远程镜像组的目的端口。

## 1.5.6 配置远程镜像VLAN

### 1. 配置限制和指导

源设备和目的设备都需要进行本配置。

源设备和目的设备上的远程镜像组必须使用相同的远程镜像 VLAN。

远程镜像 VLAN 必须是已创建的静态 VLAN。

当 VLAN 被指定为远程镜像 VLAN 后，该 VLAN 不能再作其他用途。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 为远程目的镜像组配置远程镜像 VLAN。

```
mirroring-group group-id remote-probe vlan vlan-id
```

缺省情况下，未为远程镜像组配置远程镜像 VLAN。

## 1.5.7 将目的端口加入远程镜像VLAN

### 1. 配置限制和指导

仅目的设备需要进行本配置。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入目的接口视图。

```
interface interface-type interface-number
```

- (3) 将目的端口加入远程镜像 VLAN。

- 将 Access 类型的目的端口加入远程镜像 VLAN。

```
port access vlan vlan-id
```

- 将 Trunk 类型的目的端口加入远程镜像 VLAN。

```
port trunk permit vlan vlan-id
```

- 将 Hybrid 类型的目的端口加入远程镜像 VLAN。

```
port hybrid vlan vlan-id { tagged | untagged }
```

有关 **port access vlan**、**port trunk permit vlan** 和 **port hybrid vlan** 命令的详细介绍，请参见“二层技术-以太网交换命令参考”中的“VLAN”。

## 1.5.8 创建远程源镜像组

### 1. 配置限制和指导

仅源设备需要进行本配置。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 创建远程源镜像组。

```
mirroring-group group-id remote-source
```

## 1.5.9 配置镜像源

### 1. 配置限制和指导

仅源设备需要配置镜像源。

配置源端口时，需要注意的是：

- 不能将源端口加入到远程镜像 VLAN 中，否则会影响镜像功能的正常使用。
- 一个镜像组内可以配置多个源端口。
- 一个端口作为单向源端口最多可以加入两个镜像组（一个镜像组镜像源端口的入方向报文，另一个镜像组镜像源端口的出方向报文），作为双向源端口最多可以加入一个镜像组。
- 设备仅支持配置一个对源端口的出方向、或双向进行镜像的镜像组。
- 源端口不能用作反射端口、出端口或目的端口。

- 设备不支持将二层聚合接口配置为镜像源端口。

## 2. 配置源端口

- 在系统视图下配置源端口。

- a. 进入系统视图。

```
system-view
```

- b. 为远程源镜像组配置源端口。

```
mirroring-group group-id mirroring-port interface-list { both | inbound | outbound }
```

缺省情况下，未为远程镜像组配置源端口。

- 在接口视图下配置源端口。

- a. 进入系统视图。

```
system-view
```

- b. 进入接口视图。

```
interface interface-type interface-number
```

- c. 配置本端口为远程源镜像组的源端口。

```
mirroring-group group-id mirroring-port { both | inbound | outbound }
```

缺省情况下，未配置当前端口为远程镜像组的源端口。

## 1.5.10 配置反射端口

### 1. 配置限制和指导

仅源设备需要进行本配置。

请选择设备上未被使用的端口作为反射端口，并不要在该端口上连接网线，否则会影响镜像功能的正常使用。

在将端口配置为反射端口时，该端口将恢复为缺省配置，该端口上不能再配置其他业务。

当 IRF 端口只绑定了一个物理端口时，请勿将该物理端口配置为反射端口，以免 IRF 分裂。

一个镜像组内只能配置一个反射端口。

### 2. 在系统视图下配置反射端口

- (1) 进入系统视图。

```
system-view
```

- (2) 为远程源镜像组配置反射端口。

```
mirroring-group group-id reflector-port interface-type  
interface-number
```

缺省情况下，未为远程源镜像组配置反射端口。

### 3. 在接口视图下配置反射端口

- (1) 进入系统视图。

```
system-view
```

- (2) 进入接口视图。

**interface** *interface-type interface-number*

- (3) 配置本端口为远程源镜像组的反射端口。

**mirroring-group** *group-id reflector-port*

缺省情况下，未配置本端口为远程源镜像组的反射端口。

## 1.5.11 配置出端口

### 1. 配置限制和指导

仅源设备需要进行本配置。

不能在出端口上配置生成树协议、802.1X、IGMP Snooping、静态 ARP 和 MAC 地址学习，否则会影响镜像功能的正常使用。

出端口不能是现有镜像组的成员端口。

一个镜像组内只能配置一个出端口。

### 2. 在系统视图下配置出端口

- (1) 进入系统视图。

**system-view**

- (2) 为远程源镜像组配置出端口。

**mirroring-group** *group-id monitor-egress interface-type interface-number*

缺省情况下，未为远程源镜像组配置出端口。

- (3) 进入出端口接口视图。

**interface** *interface-type interface-number*

- (4) 将出端口加入远程镜像 VLAN。

- 将 Trunk 类型的出端口加入远程镜像 VLAN。

**port trunk permit vlan** *vlan-id*

- 将 Hybrid 类型的出端口加入远程镜像 VLAN。

**port hybrid vlan** *vlan-id { tagged | untagged }*

有关 **port trunk permit vlan** 和 **port hybrid vlan** 命令的详细介绍，请参见“二层技术-以太网交换命令参考”中的“VLAN”。

### 3. 在接口视图下配置出端口

- (1) 进入系统视图。

**system-view**

- (2) 进入接口视图。

**interface** *interface-type interface-number*

- (3) 配置本端口为远程源镜像组的出端口。

**mirroring-group** *group-id monitor-egress*

缺省情况下，未配置本端口为远程源镜像组的出端口。

## 1.6 端口镜像显示和维护

在完成上述配置后，在任意视图下执行 **display** 命令可以显示配置后镜像组的运行情况，通过查看显示信息验证配置的效果。

表1-1 端口镜像显示和维护

操作	命令
显示镜像组的配置信息	<b>display mirroring-group</b> { <i>group-id</i>   <b>all</b>   <b>local</b>   <b>remote-destination</b>   <b>remote-source</b> }

## 1.7 端口镜像典型配置举例

### 1.7.1 本地端口镜像配置举例

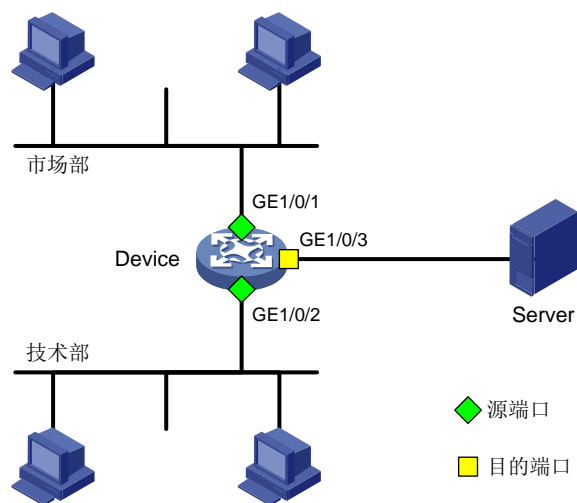
#### 1. 组网需求

Device 通过端口 GigabitEthernet1/0/1 和 GigabitEthernet1/0/2 分别连接市场部和技术部，并通过端口 GigabitEthernet1/0/3 连接 Server。

通过配置源端口方式的本地端口镜像，使 Server 可以监控所有进、出市场部和技术部的报文。

#### 2. 组网图

图1-4 本地端口镜像配置组网图



#### 3. 配置步骤

# 创建本地镜像组 1。

```
<Device> system-view
[Device] mirroring-group 1 local
```

# 配置本地镜像组 1 的源端口为 GigabitEthernet1/0/1 和 GigabitEthernet1/0/2，对源端口收发的报文都进行镜像，目的端口为 GigabitEthernet1/0/3。

```
[Device] mirroring-group 1 mirroring-port gigabitethernet 1/0/1 gigabitethernet 1/0/2 both
[Device] mirroring-group 1 monitor-port gigabitethernet 1/0/3
```

# 在目的端口 **GigabitEthernet1/0/3** 上关闭生成树协议。

```
[Device] interface gigabitethernet 1/0/3
[Device-GigabitEthernet1/0/3] undo stp enable
[Device-GigabitEthernet1/0/3] quit
```

#### 4. 验证配置

# 显示所有镜像组的配置信息。

```
[Device] display mirroring-group all
Mirroring group 1:
  Type: Local
  Status: Active
  Mirroring port:
    GigabitEthernet1/0/1 Both
    GigabitEthernet1/0/2 Both
  Monitor port: GigabitEthernet1/0/3
```

配置完成后，用户可以通过 **Server** 监控所有进、出市场部和技术部的报文。

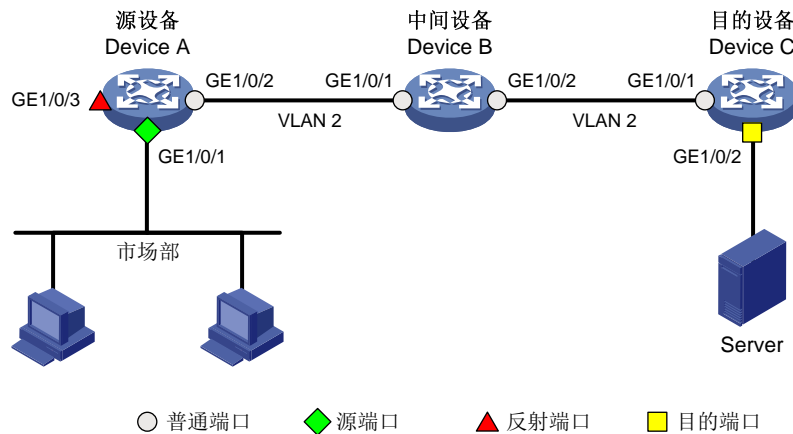
### 1.7.2 二层远程端口镜像配置举例（反射端口方式）

#### 1. 组网需求

在一个二层网络中，**Device A**、**Device B**、**Device C** 及 **Server** 如下图所示连接。其中，**Device A** 通过端口 **GigabitEthernet1/0/1** 连接市场部。

通过配置二层远程端口镜像，使 **Server** 可以监控所有进、出市场部的报文。

#### 2. 组网图



#### 3. 配置步骤

##### (1) 配置 Device C

# 配置端口 **GigabitEthernet1/0/1** 为 Trunk 口，并允许 VLAN 2 的报文通过。

```
<DeviceC> system-view
[DeviceC] interface gigabitethernet 1/0/1
[DeviceC-GigabitEthernet1/0/1] port link-type trunk
[DeviceC-GigabitEthernet1/0/1] port trunk permit vlan 2
[DeviceC-GigabitEthernet1/0/1] quit
```

# 创建远程目的镜像组 2。

```
[DeviceC] mirroring-group 2 remote-destination
```

# 创建 VLAN 2 作为远程镜像 VLAN。

```
[DeviceC] vlan 2
```

# 关闭 VLAN 2 的 MAC 地址学习功能。

```
[DeviceC-vlan2] undo mac-address mac-learning enable
```

```
[DeviceC-vlan2] quit
```

# 配置远程目的镜像组 2 的远程镜像 VLAN 为 VLAN 2，目的端口为 GigabitEthernet1/0/2，在该端口上关闭生成树协议并将其加入 VLAN 2。

```
[DeviceC] mirroring-group 2 remote-probe vlan 2
```

```
[DeviceC] interface gigabitethernet 1/0/2
```

```
[DeviceC-GigabitEthernet1/0/2] mirroring-group 2 monitor-port
```

```
[DeviceC-GigabitEthernet1/0/2] undo stp enable
```

```
[DeviceC-GigabitEthernet1/0/2] port access vlan 2
```

```
[DeviceC-GigabitEthernet1/0/2] quit
```

## (2) 配置 Device B

# 创建 VLAN 2 作为远程镜像 VLAN。

```
<DeviceB> system-view
```

```
[DeviceB] vlan 2
```

# 关闭 VLAN 2 的 MAC 地址学习功能。

```
[DeviceB-vlan2] undo mac-address mac-learning enable
```

```
[DeviceB-vlan2] quit
```

# 配置端口 GigabitEthernet1/0/1 为 Trunk 口，并允许 VLAN 2 的报文通过。

```
[DeviceB] interface gigabitethernet 1/0/1
```

```
[DeviceB-GigabitEthernet1/0/1] port link-type trunk
```

```
[DeviceB-GigabitEthernet1/0/1] port trunk permit vlan 2
```

```
[DeviceB-GigabitEthernet1/0/1] quit
```

# 配置端口 GigabitEthernet1/0/2 为 Trunk 口，并允许 VLAN 2 的报文通过。

```
[DeviceB] interface gigabitethernet 1/0/2
```

```
[DeviceB-GigabitEthernet1/0/2] port link-type trunk
```

```
[DeviceB-GigabitEthernet1/0/2] port trunk permit vlan 2
```

```
[DeviceB-GigabitEthernet1/0/2] quit
```

## (3) 配置 Device A

# 创建远程源镜像组 1。

```
<DeviceA> system-view
```

```
[DeviceA] mirroring-group 1 remote-source
```

# 创建 VLAN 2 作为远程镜像 VLAN。

```
[DeviceA] vlan 2
```

# 关闭 VLAN 2 的 MAC 地址学习功能。

```
[DeviceA-vlan2] undo mac-address mac-learning enable
```

```
[DeviceA-vlan2] quit
```

# 配置远程源镜像组 1 的远程镜像 VLAN 为 VLAN 2，源端口为 GigabitEthernet1/0/1，反射端口为 GigabitEthernet1/0/3。

```
[DeviceA] mirroring-group 1 remote-probe vlan 2
```



```
[DeviceA] mirroring-group 1 mirroring-port gigabitethernet 1/0/1 both
[DeviceA] mirroring-group 1 reflector-port gigabitethernet 1/0/3
This operation may delete all settings made on the interface. Continue? [Y/N]: y
# 配置端口 GigabitEthernet1/0/2 为 Trunk 口，并允许 VLAN 2 的报文通过。
[DeviceA] interface gigabitethernet 1/0/2
[DeviceA-GigabitEthernet1/0/2] port link-type trunk
[DeviceA-GigabitEthernet1/0/2] port trunk permit vlan 2
[DeviceA-GigabitEthernet1/0/2] quit
```

#### 4. 验证配置

# 显示 Device C 上所有镜像组的配置信息。

```
[DeviceC] display mirroring-group all
Mirroring group 2:
  Type: Remote destination
  Status: Active
  Monitor port: GigabitEthernet1/0/2
  Remote probe VLAN: 2
```

# 显示 Device A 上所有镜像组的配置信息。

```
[DeviceA] display mirroring-group all
Mirroring group 1:
  Type: Remote source
  Status: Active
  Mirroring port:
    GigabitEthernet1/0/1 Both
  Reflector port: GigabitEthernet1/0/3
  Remote probe VLAN: 2
```

配置完成后，用户可以通过 **Server** 监控所有进、出市场部的报文。

### 1.7.3 二层远程端口镜像配置举例（出端口方式）

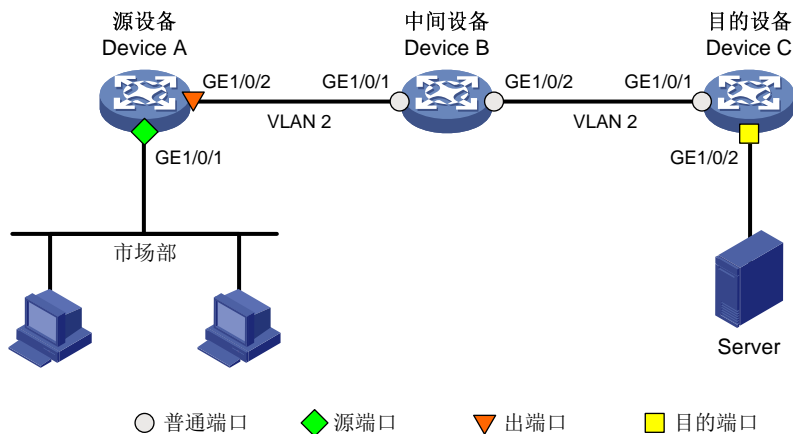
#### 1. 组网需求

在一个二层网络中，Device A、Device B、Device C 及 Server 如下图所示连接。其中，Device A 通过端口 **GigabitEthernet1/0/1** 连接市场部。

通过配置二层远程端口镜像，使 **Server** 可以监控所有进、出市场部的报文。

## 2. 组网图

图1-5 二层远程端口镜像配置组网图



## 3. 配置步骤

### (1) 配置 Device C

# 配置端口 GigabitEthernet1/0/1 为 Trunk 口，并允许 VLAN 2 的报文通过。

```
<DeviceC> system-view
[DeviceC] interface gigabitethernet 1/0/1
[DeviceC-GigabitEthernet1/0/1] port link-type trunk
[DeviceC-GigabitEthernet1/0/1] port trunk permit vlan 2
[DeviceC-GigabitEthernet1/0/1] quit
```

# 创建远程目的镜像组 2。

```
[DeviceC] mirroring-group 2 remote-destination
```

# 创建 VLAN 2 作为远程镜像 VLAN。

```
[DeviceC] vlan 2
```

# 关闭 VLAN 2 的 MAC 地址学习功能。

```
[DeviceC-vlan2] undo mac-address mac-learning enable
[DeviceC-vlan2] quit
```

# 配置远程目的镜像组 2 的远程镜像 VLAN 为 VLAN 2，目的端口为 GigabitEthernet1/0/2，在该端口上关闭生成树协议并将其加入 VLAN 2。

```
[DeviceC] mirroring-group 2 remote-probe vlan 2
[DeviceC] interface gigabitethernet 1/0/2
[DeviceC-GigabitEthernet1/0/2] mirroring-group 2 monitor-port
[DeviceC-GigabitEthernet1/0/2] undo stp enable
[DeviceC-GigabitEthernet1/0/2] port access vlan 2
[DeviceC-GigabitEthernet1/0/2] quit
```

### (2) 配置 Device B

# 创建 VLAN 2 作为远程镜像 VLAN。

```
<DeviceB> system-view
[DeviceB] vlan 2
```

# 关闭 VLAN 2 的 MAC 地址学习功能。

```
[DeviceB-vlan2] undo mac-address mac-learning enable
```

```
[DeviceB-vlan2] quit
# 配置端口 GigabitEthernet1/0/1 为 Trunk 口，并允许 VLAN 2 的报文通过。
[DeviceB] interface gigabitethernet 1/0/1
[DeviceB-GigabitEthernet1/0/1] port link-type trunk
[DeviceB-GigabitEthernet1/0/1] port trunk permit vlan 2
[DeviceB-GigabitEthernet1/0/1] quit
# 配置端口 GigabitEthernet1/0/2 为 Trunk 口，并允许 VLAN 2 的报文通过。
[DeviceB] interface gigabitethernet 1/0/2
[DeviceB-GigabitEthernet1/0/2] port link-type trunk
[DeviceB-GigabitEthernet1/0/2] port trunk permit vlan 2
[DeviceB-GigabitEthernet1/0/2] quit
```

### (3) 配置 Device A

```
# 创建远程源镜像组 1。
<DeviceA> system-view
[DeviceA] mirroring-group 1 remote-source
# 创建 VLAN 2 作为远程镜像 VLAN。
[DeviceA] vlan 2
# 关闭 VLAN 2 的 MAC 地址学习功能。
[DeviceA-vlan2] undo mac-address mac-learning enable
[DeviceA-vlan2] quit
# 配置远程源镜像组 1 的远程镜像 VLAN 为 VLAN 2，源端口为 GigabitEthernet1/0/1，出端口为 GigabitEthernet1/0/2。
[DeviceA] mirroring-group 1 remote-probe vlan 2
[DeviceA] mirroring-group 1 mirroring-port gigabitethernet 1/0/1 both
[DeviceA] mirroring-group 1 monitor-egress gigabitethernet 1/0/2
# 配置端口 GigabitEthernet1/0/2 为 Trunk 口，允许 VLAN 2 的报文通过，并在该端口上关闭生成树协议。
[DeviceA] interface gigabitethernet 1/0/2
[DeviceA-GigabitEthernet1/0/2] port link-type trunk
[DeviceA-GigabitEthernet1/0/2] port trunk permit vlan 2
[DeviceA-GigabitEthernet1/0/2] undo stp enable
[DeviceA-GigabitEthernet1/0/2] quit
```

## 4. 验证配置

# 显示 Device C 上所有镜像组的配置信息。

```
[DeviceC] display mirroring-group all
Mirroring group 2:
    Type: Remote destination
    Status: Active
    Monitor port: GigabitEthernet1/0/2
    Remote probe VLAN: 2
```

# 显示 Device A 上所有镜像组的配置信息。

```
[DeviceA] display mirroring-group all
Mirroring group 1:
    Type: Remote source
    Status: Active
```

Mirroring port:

GigabitEthernet1/0/1 Both

Monitor egress port: GigabitEthernet1/0/2

Remote probe VLAN: 2

配置完成后，用户可以通过 **Server** 监控所有进、出市场部的报文。

## 2 流镜像

### 2.1 流镜像简介

流镜像是指将指定报文复制到指定目的地，以便于对报文进行分析和监控。

流镜像通过 QoS 实现，设备先通过流分类匹配待镜像的报文，再通过流行为将符合条件的报文镜像至指定目的地。该方式可以灵活配置报文的匹配条件，从而对报文进行精细区分，并将区分后的报文镜像到目的地。有关 QoS 的详细介绍，请参见“ACL 和 QoS 配置指导”中的“QoS”。

根据报文镜像的目的地不同，流行为可分为以下类型：

- 流镜像到接口：将符合条件的报文复制一份到指定接口，利用数据检测设备分析接口收到的报文。
- 流镜像到 CPU：将符合条件的报文复制一份到 CPU（这里的 CPU 是指报文进入的成员设备上的 CPU），通过 CPU 分析报文的内容，或者将特定的协议报文上送。

### 2.2 流镜像配置限制和指导

流镜像配置中，除 **mirror-to** 命令外的其他配置命令及相关显示命令的详细介绍，请参见“ACL 和 QoS 命令参考”中的“QoS 策略”。

### 2.3 流镜像配置任务简介

流镜像配置任务如下：

- (1) [配置流分类](#)  
该配置用来匹配待镜像的报文。
- (2) [配置流行为](#)  
该配置用来指定镜像报文的目的地。
- (3) [配置QoS策略](#)  
该配置为流分类指定流行为，即指定哪些报文需要镜像到哪里。
- (4) [应用QoS策略](#)  
请选择以下一项任务进行配置：
  - [基于接口应用](#)
  - [基于VLAN应用](#)
  - [基于全局应用](#)

### 2.4 配置流分类

- (1) 进入系统视图。  
**system-view**
- (2) 定义流分类，并进入流分类视图。

**traffic classifier** *classifier-name* [ **operator** { **and** | **or** } ]

- (3) 配置报文匹配规则。

**if-match** *match-criteria*

- (4) （可选）显示用户定义流分类的配置信息。

**display traffic classifier**

该命令可在任意视图下执行。

## 2.5 配置流行为

- (1) 进入系统视图。

**system-view**

- (2) 定义流行为，并进入流行为视图。

**traffic behavior** *behavior-name*

- (3) 配置镜像报文的目的地。请选择其中一项进行配置。

- 配置流镜像到接口。

**mirror-to interface** *interface-type interface-number*

缺省情况下，未配置流镜像到接口。

同一流行为中只能配置一个目的接口，多次执行该命令，仅最后一次执行的命令生效。

- 配置流镜像到 CPU。

**mirror-to cpu**

缺省情况下，未配置流镜像到 CPU。

- (4) （可选）显示用户定义流行为的配置信息。

**display traffic behavior**

该命令可在任意视图下执行。

## 2.6 配置QoS策略

- (1) 进入系统视图。

**system-view**

- (2) 定义 QoS 策略，并进入 QoS 策略视图。

**qos policy** *policy-name*

- (3) 为流分类指定采用的流行为。

**classifier** *classifier-name* **behavior** *behavior-name*

缺省情况下，未为流分类指定流行为。

- (4) （可选）显示用户定义策略的配置信息。

**display qos policy**

该命令可在任意视图下执行。

## 2.7 应用QoS策略

### 2.7.1 基于接口应用

#### 1. 配置限制和指导

将 QoS 策略应用到接口后，可以对该接口的流量进行镜像。

一个 QoS 策略可以应用于多个接口。

设备不支持在接口出方向应用 QoS 策略配置流镜像，且在入方向上只能应用一个 QoS 策略。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入接口视图。

```
interface interface-type interface-number
```

- (3) 应用 QoS 策略到接口。

```
qos apply policy policy-name inbound
```

- (4) （可选）显示接口上 QoS 策略的配置信息和运行情况。

```
display qos policy interface
```

该命令可在任意视图下执行。

### 2.7.2 基于VLAN应用

#### 1. 配置限制和指导

将 QoS 策略应用到 VLAN 后，可以对该 VLAN 内各端口的流量进行镜像。

设备不支持在 VLAN 出方向应用 QoS 策略配置流镜像，且在入方向上只能应用一个 QoS 策略。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 应用 QoS 策略到指定 VLAN。

```
qos vlan-policy policy-name vlan vlan-id-list inbound
```

- (3) （可选）显示基于 VLAN 应用 QoS 策略的信息。

```
display qos vlan-policy
```

该命令可在任意视图下执行。

### 2.7.3 基于全局应用

#### 1. 配置限制和指导

将 QoS 策略应用到全局后，可以对设备所有端口的流量进行镜像。

设备不支持在全局出方向应用 QoS 策略配置流镜像，且在入方向上只能应用一个 QoS 策略。

## 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 应用 QoS 策略到全局。

```
qos apply policy policy-name global inbound
```

- (3) （可选）显示基于全局应用 QoS 策略的信息。

```
display qos policy global
```

该命令可在任意视图下执行。

## 2.8 流镜像典型配置举例

### 2.8.1 流镜像基本组网配置举例

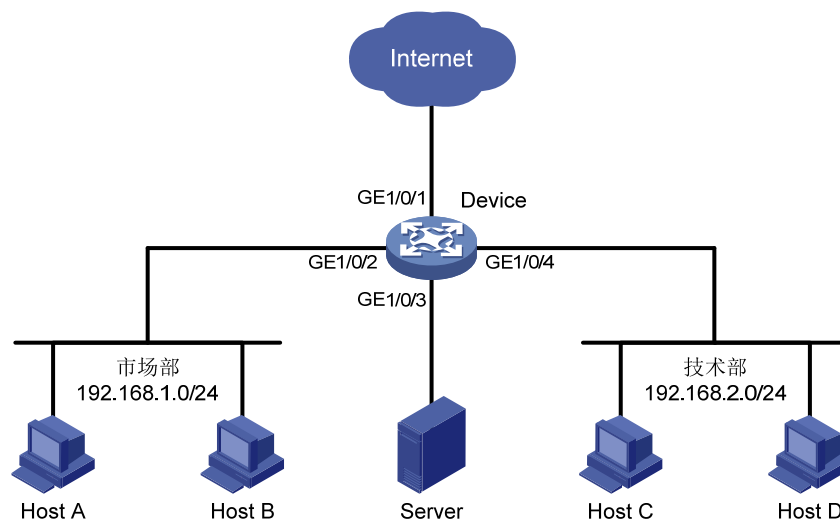
#### 1. 组网需求

某公司内的各部门之间使用不同网段的 IP 地址，其中市场部和技术部分别使用 192.168.1.0/24 和 192.168.2.0/24 网段，该公司的工作时间为每周工作日的 8 点到 18 点。

通过配置流镜像，使 Server 可以监控技术部访问互联网的 WWW 流量，以及技术部在工作时间发往市场部的 IP 流量。

#### 2. 组网图

图2-1 流镜像典型配置组网图



## 3. 配置步骤

# 定义工作时间：创建名为 **work** 的时间段，其时间范围为每周工作日的 8 点到 18 点。

```
<Device> system-view
```

```
[Device] time-range work 8:00 to 18:00 working-day
```

# 创建一个编号为 3000 的 IPv4 高级 ACL，并定义如下规则：匹配技术部访问 WWW 的报文，以及在工作时间由技术部发往市场部的 IP 报文。

```
[Device] acl advanced 3000
```



```
[Device-acl-ipv4-adv-3000] rule permit tcp source 192.168.2.0 0.0.0.255 destination-port eq  
www
```

```
[Device-acl-ipv4-adv-3000] rule permit ip source 192.168.2.0 0.0.0.255 destination  
192.168.1.0 0.0.0.255 time-range work
```

```
[Device-acl-ipv4-adv-3000] quit
```

**# 创建流分类 tech\_c，并配置报文匹配规则为 ACL 3000。**

```
[Device] traffic classifier tech_c
```

```
[Device-classifier-tech_c] if-match acl 3000
```

```
[Device-classifier-tech_c] quit
```

**# 创建流行为 tech\_b，并配置流镜像到接口 GigabitEthernet1/0/3。**

```
[Device] traffic behavior tech_b
```

```
[Device-behavior-tech_b] mirror-to interface gigabitethernet 1/0/3
```

```
[Device-behavior-tech_b] quit
```

**# 创建 QoS 策略 tech\_p，在策略中为流分类 tech\_c 指定采用流行为 tech\_b。**

```
[Device] qos policy tech_p
```

```
[Device-qospolicy-tech_p] classifier tech_c behavior tech_b
```

```
[Device-qospolicy-tech_p] quit
```

**# 将 QoS 策略 tech\_p 应用到接口 GigabitEthernet1/0/4 的入方向上。**

```
[Device] interface gigabitethernet 1/0/4
```

```
[Device-GigabitEthernet1/0/4] qos apply policy tech_p inbound
```

```
[Device-GigabitEthernet1/0/4] quit
```

#### **4. 验证配置**

配置完成后，用户可以通过 **Server** 监控技术部访问互联网的 WWW 流量，以及技术部在工作时间发往市场部的 IP 流量。

# 目 录

1 sFlow .....	1-1
1.1 sFlow简介 .....	1-1
1.1.1 sFlow的工作机制 .....	1-1
1.1.2 协议规范 .....	1-1
1.2 配置sFlow基本信息 .....	1-1
1.3 配置Flow采样 .....	1-2
1.4 配置Counter采样 .....	1-3
1.5 sFlow显示和维护 .....	1-3
1.6 sFlow典型配置举例 .....	1-3
1.6.1 sFlow基本组网配置举例 .....	1-3
1.7 sFlow常见故障处理 .....	1-5
1.7.1 远端的sFlow Collector无法收到sFlow报文 .....	1-5

# 1 sFlow

## 1.1 sFlow简介

sFlow 是一种基于报文采样的网络流量监控技术，主要用于对网络流量进行统计分析。

### 1.1.1 sFlow的工作机制

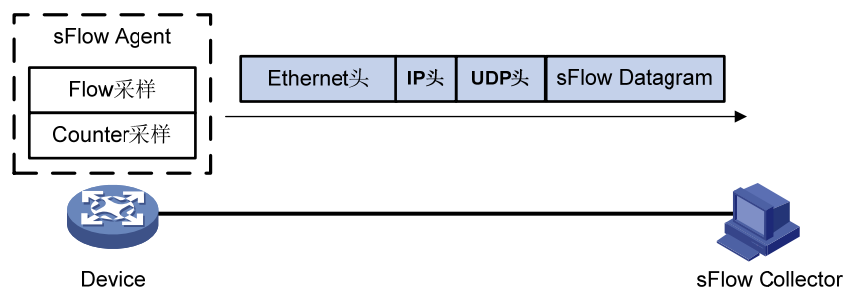
如 图 1-1 所示，sFlow系统包含嵌入在设备中的sFlow Agent和远端的sFlow Collector。其中，sFlow Agent通过采样机制获取接口的统计信息和数据包信息，将信息封装成sFlow报文，当存放sFlow报文的缓冲区满或是在sFlow报文发送定时器（定时器时间间隔固定为 1 秒）超时后，会将sFlow报文封装在UDP报文里发送到指定的sFlow Collector。sFlow Collector会对sFlow报文进行分析，并显示分析结果。一个sFlow Collector可以监控多个sFlow Agent。

sFlow 使用以下两种采样机制：

- Flow 采样：基于数据包的流采样，用于获取数据包内容的相关信息。
- Counter 采样：基于时间的接口统计信息采样，用于获取接口的统计信息。

sFlow 可以同时采用上述两种采样机制。

图1-1 sFlow 工作机制



### 1.1.2 协议规范

与 sFlow 相关的协议规范有：

- RFC3176: InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks
- sFlow.org: sFlow Version 5

## 1.2 配置sFlow基本信息

### 1. 配置限制和指导

由于设备会定期检查是否配置了 sFlow Agent 的 IP 地址，如果未配置，设备会自动查找一个 IPv4 地址作为 sFlow Agent 的 IP 地址。自动查找的 IP 地址信息不会保存在配置文件中。所以建议用户手工配置 sFlow Agent 的 IP 地址。

在设备上只能配置一个 sFlow Agent 的 IP 地址，新配置的 IP 地址会覆盖已有的配置。

## 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置 sFlow Agent 的 IP 地址。

```
sflow agent { ip ipv4-address | ipv6 ipv6-address }
```

缺省情况下，未配置 sFlow Agent 的 IP 地址。

- (3) 创建并配置 sFlow Collector。

```
sflow collector collector-id { ip ipv4-address | ipv6 ipv6-address }  
[ port port-number | datagram-size size | time-out seconds | description  
string ] *
```

缺省情况下，未配置 sFlow Collector 的相关信息。

- (4) 配置 sFlow 报文的源 IP 地址。

```
sflow source { ip ipv4-address | ipv6 ipv6-address } *
```

缺省情况下，设备使用路由决定的源 IP 地址作为 sFlow 报文的源 IP 地址。

## 1.3 配置Flow采样

### 1. 功能简介

在接口上配置 Flow 采样后，sFlow Agent 会根据配置的参数对该接口上的报文进行采样，然后将采样报文封装为 sFlow 报文。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入二层以太网接口视图。

```
interface interface-type interface-number
```

- (3) 配置 Flow 采样的采样模式。

```
sflow sampling-mode random
```

- (4) 开启 Flow 采样功能并配置 Flow 采样的报文采样率，即在 *rate* 个报文中抽取一个报文进行采样。

```
sflow sampling-rate rate
```

缺省情况下，Flow 采样功能处于关闭状态。

建议用户配置报文采样率为 2 的 N 次方且大于等于 8192，例如 32768。

- (5) （可选）配置在进行报文内容拷贝时，从原始报文的头部开始，允许拷贝的最大字节数。

```
sflow flow max-header length
```

缺省情况下，从原始报文的头部开始，允许拷贝的最大字节数为 128 字节。

建议用户使用缺省配置。

- (6) 配置经过 Flow 采样后，sFlow Agent 输出 sFlow 报文的目的 sFlow Collector 编号。

```
sflow flow collector collector-id
```

缺省情况下，Flow 采样和 sFlow Collector 没有绑定关系，即没有指定目的 sFlow Collector 编号。

## 1.4 配置Counter采样

### 1. 功能简介

在接口上配置 Counter 采样后，sFlow Agent 会周期性提取该接口上的统计信息，然后将统计信息封装为 sFlow 报文。

### 2. 配置步骤

(1) 进入系统视图。

**system-view**

(2) 进入二层以太网接口视图。

**interface interface-type interface-number**

(3) 开启 Counter 采样功能并配置 Counter 采样的时间间隔。

**sflow counter interval interval**

缺省情况下，Counter 采样功能处于关闭状态。

(4) 配置经过 Counter 采样后，sFlow Agent 输出 sFlow 报文的目的地 sFlow Collector 编号。

**sflow counter collector collector-id**

缺省情况下，Counter 采样和 sFlow Collector 没有绑定关系，即没有指定目的 sFlow Collector 编号。

## 1.5 sFlow显示和维护

在完成上述配置后，在任意视图下执行 **display** 命令可以显示配置后 sFlow 的运行情况，通过查看显示信息验证配置的效果。

表1-1 sFlow 显示和维护

操作	命令
显示sFlow配置信息	<b>display sflow</b>

## 1.6 sFlow典型配置举例

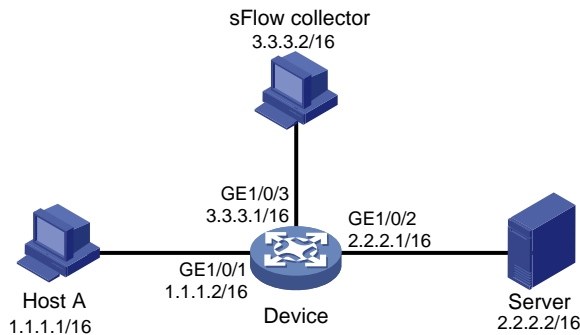
### 1.6.1 sFlow基本组网配置举例

#### 1. 组网需求

如 [图 1-2](#) 所示，在Device上运行sFlow Agent，并在接口GigabitEthernet1/0/1 上开启sFlow功能，包括Flow采样（选择随机模式）和Counter采样，从而对该接口的网络流量进行监控。最后Device将采样结果封装为sFlow报文，通过接口GigabitEthernet1/0/3 发送给 sFlow Collector，sFlow Collector对sFlow报文进行分析并显示分析结果。

## 2. 组网图

图1-2 配置 sFlow 组网图



## 3. 配置步骤

### (1) 配置 IP 地址

请按照 [图 1-2](#) 配置各接口的IP地址和子网掩码，具体配置过程略。

### (2) 配置 sFlow Agent 和 sFlow Collector 信息

# 配置 sFlow Agent 的 IP 地址。

```
<Device> system-view
[Device] sflow agent ip 3.3.3.1
```

# 配置 sFlow Collector 信息：sFlow Collector 编号为 1，IP 地址为 3.3.3.2，端口号保持缺省  
值 6343，描述信息为 netserver。

```
[Device] sflow collector 1 ip 3.3.3.2 description netserver
```

### (3) 配置 Counter 采样

# 在 GigabitEthernet1/0/1 接口上开启 Counter 采样功能并配置 Counter 采样的时间间隔为  
120 秒。

```
[Device] interface gigabitethernet 1/0/1
[Device-GigabitEthernet1/0/1] sflow counter interval 120
```

# 在 GigabitEthernet1/0/1 接口上经过 Counter 采样后，sFlow Agent 输出 sFlow 报文的目的  
sFlow Collector 编号为 1。

```
[Device-GigabitEthernet1/0/1] sflow counter collector 1
```

### (4) 配置 Flow 采样

# 在 GigabitEthernet1/0/1 上开启 Flow 采样功能并配置 Flow 采样的采样模式为随机采样，  
Flow 采样的报文采样率为 32768。

```
[Device-GigabitEthernet1/0/1] sflow sampling-mode random
[Device-GigabitEthernet1/0/1] sflow sampling-rate 32768
```

# 在 GigabitEthernet1/0/1 接口上经过 Flow 采样后，sFlow Agent 输出 sFlow 报文的目的  
sFlow Collector 编号为 1。

```
[Device-GigabitEthernet1/0/1] sflow flow collector 1
```

## 4. 验证配置

# 显示 sFlow 的配置和运行信息。

```
[Device-GigabitEthernet1/0/1] display sflow
```

```
sFlow datagram version: 5
Global information:
Agent IP: 3.3.3.1(CLI)
Source address:
Collector information:
ID      IP          Port  Aging      Size VPN-instance Description
1       3.3.3.2        6343  N/A        1400
Port information:
Interface  CID  Interval(s) FID  MaxHLen Rate      Mode      Status
GE1/0/1    1    120         1   128    32768   Random   Active
```

从上面的显示信息中可以看到开启 sFlow 功能的 GigabitEthernet1/0/1 接口处于 “Active” 状态，Counter 采样的时间间隔为 120 秒，Flow 采样的报文采样率为 32768，即在 32768 个报文中抽取一个报文进行采样，表示 sFlow 功能正在正常运行。

## 1.7 sFlow常见故障处理

### 1.7.1 远端的sFlow Collector无法收到sFlow报文

#### 1. 故障现象

远端的 sFlow Collector 无法收到 sFlow 报文。

#### 2. 故障分析

可能有以下几个原因：

- 未配置 sFlow Collector 的 IP 地址；
- 接口未配置 sFlow 采样，导致没有接口提供采样数据；
- 配置的 sFlow Collector 的 IP 地址和远端的 sFlow Collector 的 IP 地址不同，导致远端的 sFlow Collector 无法收到 sFlow 报文；
- 设备和 sFlow Collector 之间的物理连接中断；
- 配置的 sFlow 报文的长度小于 sFlow 报文头长度与配置的采样样本长度之和。

#### 3. 处理过程

- (1) 执行 **display sflow** 命令，查看当前的 sFlow 配置信息，检查是否为 sFlow 特性配置错误；
- (2) 检查设备是否已经配置可以和 sFlow Collector 通信的 IP 地址；
- (3) 检查设备和 sFlow Collector 之间的物理连接是否正常；
- (4) 检查配置的 sFlow 报文的长度是否大于 sFlow 报文头长度与配置的采样样本长度（建议使用缺省配置）之和。

# 目 录

1 信息中心.....	1-1
1.1 信息中心简介.....	1-1
1.1.1 日志信息的分类.....	1-1
1.1.2 日志信息的等级.....	1-1
1.1.3 日志信息的输出方向.....	1-2
1.1.4 日志信息的缺省输出规则.....	1-2
1.1.5 诊断日志信息的缺省输出规则.....	1-2
1.1.6 安全日志信息的缺省输出规则.....	1-2
1.1.7 隐藏日志信息的缺省输出规则.....	1-3
1.1.8 调试跟踪日志信息的缺省输出规则.....	1-3
1.1.9 日志信息的格式.....	1-3
1.2 FIPS相关说明.....	1-6
1.3 信息中心配置任务简介.....	1-6
1.3.1 普通日志配置任务简介.....	1-6
1.3.2 隐藏日志配置任务简介.....	1-6
1.3.3 安全日志配置任务简介.....	1-7
1.3.4 诊断日志配置任务简介.....	1-7
1.3.5 调试跟踪日志配置任务简介.....	1-7
1.4 开启信息中心功能.....	1-8
1.5 将日志信息输出到指定方向.....	1-8
1.5.1 配置日志信息发送到控制台.....	1-8
1.5.2 配置日志信息发送到监视终端.....	1-9
1.5.3 配置日志信息发送到日志主机.....	1-9
1.5.4 配置日志信息发送到日志缓冲区.....	1-10
1.5.5 配置日志信息保存到日志文件.....	1-11
1.6 配置日志信息的最短保存时间.....	1-12
1.7 配置命令行输入回显功能.....	1-12
1.8 抑制日志信息输出.....	1-13
1.8.1 配置重复日志抑制功能.....	1-13
1.8.2 禁止指定模块日志的输出.....	1-13
1.8.3 禁止接口生成Link up/Link down日志信息.....	1-14
1.9 将系统日志封装成告警信息发送.....	1-14
1.10 配置安全日志功能.....	1-15



1.10.1 保存安全日志 .....	1-15
1.10.2 管理安全日志文件 .....	1-16
1.11 配置诊断日志信息保存到诊断日志文件 .....	1-16
1.12 配置调试跟踪日志文件的大小 .....	1-17
1.13 信息中心显示和维护 .....	1-17
1.14 信息中心典型配置举例 .....	1-18
1.14.1 日志发送到控制台的配置举例 .....	1-18
1.14.2 日志发送到Unix日志主机的配置举例 .....	1-19
1.14.3 日志发送到Linux日志主机的配置举例 .....	1-20

# 1 信息中心

## 1.1 信息中心简介

信息中心是设备的信息枢纽，它接收各模块生成的日志信息，能够按模块和等级将收到的日志信息输出到控制台、监视终端、日志主机等方向，为管理员监控设备运行情况和诊断网络故障提供了有力的支持。

### 1.1.1 日志信息的分类

系统产生的日志信息共分为：

- 普通日志：用于记录日常信息。除特殊说明外，下文中的日志均指普通日志。
- 诊断日志：用于记录调试信息。
- 安全日志：用于记录与认证、授权等安全相关的信息。
- 隐藏日志：用于记录需要以日志的方式记录下来但不需要在终端上显示的信息（如用户通过命令行输入命令的记录信息等）。
- 调试跟踪日志：用于记录系统跟踪调试信息，调试跟踪日志信息，必须加载 **devkit** 包后才可以查看，普通用户无需关注，主要提供给服务工程师定位问题。

### 1.1.2 日志信息的等级

日志信息按严重性可划分为如 [表 1-1](#) 所示的八个等级，各等级的严重性依照数值从 0~7 依次降低。在系统输出信息时，所有信息等级高于或等于配置等级的信息都会被输出。例如，输出规则中指定允许等级为 6（informational）的信息输出，则等级 0~6 的信息均会被输出。

表1-1 日志信息等级列表

数值	信息等级	描述
0	emergency	表示设备不可用的信息，如系统授权已到期
1	alert	表示设备出现重大故障，需要立刻做出反应的信息，如流量超出接口上限
2	critical	表示严重信息，如设备温度已经超过预警值，设备电源、风扇出现故障等
3	error	表示错误信息，如接口链路状态变化等
4	warning	表示警告信息，如接口连接断开，内存耗尽告警等
5	notification	表示正常出现但是重要的信息，如通过终端登录设备，设备重启等
6	informational	表示需要记录的通知信息，如通过命令行输入命令的记录信息，执行ping命令的日志信息等
7	debugging	表示调试过程产生的信息

1.1.3 日志信息的输出方向

系统可以向以下方向发送日志信息：控制台（console）、监视终端（monitor）、日志缓冲区（logbuffer）、日志主机（loghost）和日志文件（logfile）。日志信息的各个输出方向相互独立，可在开启信息中心后分别进行配置，同一日志可同时输出到多个方向。

1.1.4 日志信息的缺省输出规则

日志信息的输出规则规定了各个输出方向可以输出的日志信息模块和输出的日志信息等级，日志信息的输出方向包括控制台、监视终端、日志主机、日志缓冲区和日志文件。各个输出方向的缺省情况如 [表 1-2](#) 所示：

表1-2 输出方向的缺省输出规则

输出方向	日志信息来源	开关	等级
控制台	所有支持的模块	开	debugging
监视终端	所有支持的模块	关	debugging
日志主机	所有支持的模块	开	informational
日志缓冲区	所有支持的模块	开	informational
日志文件	所有支持的模块	开	informational

1.1.5 诊断日志信息的缺省输出规则

诊断日志信息的输出方向只有诊断日志文件。诊断日志文件不能进行输出模块和输出级别的过滤配置，输出方向的缺省情况如 [表 1-3](#) 所示：

表1-3 输出方向的缺省输出规则

输出方向	日志信息来源	开关	等级
诊断日志文件	所有支持的模块	开	debugging

1.1.6 安全日志信息的缺省输出规则

安全日志信息的输出方向只有安全日志文件。安全日志文件不能进行输出模块和输出级别的过滤配置，输出方向的缺省情况如 [表 1-4](#) 所示：

表1-4 输出方向的缺省输出规则

输出方向	日志信息来源	开关	等级
安全日志文件	所有支持的模块	关	debugging

### 1.1.7 隐藏日志信息的缺省输出规则

隐藏日志信息的输出方向包括日志主机、日志缓冲区和日志文件。各个输出方向的缺省情况如 [表 1-5](#) 所示：

表1-5 输出方向的缺省输出规则

输出方向	日志信息来源	开关	等级
日志主机	所有支持的模块	开	informational
日志缓冲区	所有支持的模块	开	informational
日志文件	所有支持的模块	开	informational

### 1.1.8 调试跟踪日志信息的缺省输出规则

调试跟踪日志信息的输出方向只有调试跟踪日志文件。调试跟踪日志文件不能进行输出模块和输出级别的过滤配置，输出方向的缺省情况如 [表 1-6](#) 所示：

表1-6 输出方向的缺省输出规则

输出方向	日志信息来源	开关	等级
调试跟踪日志文件	所有支持的模块	开	debugging

### 1.1.9 日志信息的格式

#### 1. 格式

根据输出方向不同，日志信息的输出格式如下：

表1-7 日志信息格式表

输出方向	格式		举例
控制台、监视终端、日志缓冲区或日志文件	Prefix Timestamp Sysname Module/Level/Mnemonic: Content		%Nov 24 14:21:43:502 2016 Sysname SHELL/5/SHELL_LOGIN: VTY logged in from 192.168.1.26
日志主机	标准格式	<PRI>Timestamp Sysname %%vvModule/Level/Mnemonic: Source; Content	<190>Nov 24 16:22:21 2016 Sysname %%10 SHELL/5/SHELL_LOGIN: -DevIP=1.1.1.1; VTY logged in from 192.168.1.26
	unicom格式	<PRI>Timestamp Hostip vvModule/Level/Seria l_number: Content	<189>Oct 13 16:48:08 2016 10.1.1.1 10SHELL/5/210231a64jx073000020: VTY logged in from 192.168.1.21
	cmcc格式	<PRI>Timestamp Sysname %%vvModule /Level/Mnemonic: Source Content	<189>Oct 9 14:59:04 2016 Sysname %10SHELL/5/SHELL_LOGIN: VTY logged in from 192.168.1.21



## 说明

上表中介绍的格式是设备向各个输出方向发送的原始信息的格式，可能与用户最终看到的信息格式有差异，最终显示格式与用户使用的日志解析工具有关，请以实际情况为准。

## 2. 字段说明

- Prefix（信息类型）

对于输出方向为控制台、监视终端、日志缓冲区或日志文件的日志信息，时间戳前面会有一个信息类型标识符：

- 百分号（%）：表示该日志信息为 **informational** 级别及以上级别的 **log** 日志。
- 星号（\*）：表示该日志信息为 **debugging** 级别的 **log** 日志。
- 指数符号（^）：表示该日志信息为诊断日志（不区分级别）。

- PRI（优先级）

对于输出方向为日志主机的日志信息，时间戳前面会有一个优先级标识符。优先级的计算公式为：**facility\*8+level**。

- facility** 表示工具名称，由 **info-center loghost** 命令配置，主要用于在日志主机端标志不同的日志来源，查找、过滤对应日志源的日志。其中，**local0~local7** 分别对应取值 **16~23**。
- level** 表示日志信息的等级，具体含义请参见 [表 1-1](#)。

- Timestamp（时间戳）

时间戳记录了日志信息产生的时间，方便用户查看和定位系统事件。发送到日志主机和发送到其它方向的日志信息的时间戳精度不同：

- 发送到日志主机的日志信息的时间戳精确到秒。
- 发送到其它方向的日志信息的时间戳精确到毫秒。

发送到日志主机和发送到其它方向的日志信息的时间戳的配置命令也不同：

- 发送到日志主机的日志信息的时间戳格式由 **info-center timestamp loghost** 命令配置。
- 发送到其它方向的日志信息的时间戳格式由 **info-center timestamp** 命令配置。

各时间戳格式的详细描述如 [表 1-8](#) 所示：

表1-8 时间戳参数描述表

时间戳参数	说明	举例
<b>boot</b>	系统启动后经历的时间（即设备本次运行的持续时间），格式为： <b>xxx.yyy</b> ，其中 <b>xxx</b> 是系统启动后经历时间的毫秒数高32位， <b>yyy</b> 是低32位  除日志主机方向外，发往其它方向的日志信息均支持该参数	<b>%0.109391473 Sysname</b> <b>FTPD/5/FTPD_LOGIN: User ftp</b> <b>(192.168.1.23) has logged in successfully.</b>  其中 <b>0.109391473</b> 即为 <b>boot</b> 格式的时间戳

时间戳参数	说明	举例
<b>date</b>	系统当前的日期和时间，格式为： <ul style="list-style-type: none"> <li>日志主机：“mmm dd hh:mm:ss yyyy”</li> <li>其他方向：“MMM DD hh:mm:ss:xxx YYYY”</li> </ul> 发往所有方向的日志信息均支持该参数	%May 30 05:36:29:579 2003 Sysname FTPD/5/FTPD_LOGIN: User ftp (192.168.1.23) has logged in successfully. 其中May 30 05:36:29:579 2003即为 <b>date</b> 格式的时间戳
<b>iso</b>	ISO 8601中规定的时间戳格式 只有发往日志主机方向的日志信息支持该 参数	<189>2003-05-30T06:42:44 Sysname %%10FTPD/5/FTPD_LOGIN(l): User ftp (192.168.1.23) has logged in successfully. 其中2003-05-30T06:42:44即为 <b>iso</b> 格式 的时间戳
<b>none</b>	不带时间信息 发往所有方向的日志信息均支持该参数	% Sysname FTPD/5/FTPD_LOGIN: User ftp (192.168.1.23) has logged in successfully. 其中没有包含时间戳
<b>no-year-date</b>	系统当前日期和时间，但不包含年份信息， 格式为“MMM DD hh:mm:ss:xxx” 只有发往日志主机方向的日志信息支持该 参数	<189>May 30 06:44:22 Sysname %%10FTPD/5/FTPD_LOGIN(l): User ftp (192.168.1.23) has logged in successfully. 其中May 30 06:44:22即为 <b>no-year-date</b> 格式的时间戳

- Hostip（出接口 IP 地址）**  
 本字段表示发送的日志信息的源 IP 地址。只有配置 **info-center loghost source** 后，此字段才显示为出接口的 IP 地址，未配置时，使用 Sysname 显示。本字段只有在使用 unicom 格式发往日志主机时才存在。
- Serial\_number(设备序列号)**  
 本字段为当前系统的设备实体序列号，只有在使用 unicom 格式发往日志主机时才存在。
- Sysname（主机名或主机 IP 地址）**  
 本字段为生成该日志信息的设备的名称或 IP 地址。用户可使用 **sysname** 命令修改设备的名称。
- %%（厂家标志）**  
 本字段表示本日志信息由 H3C 设备生成。  
 本字段只有在日志信息发往日志主机时才会存在。
- vv（版本信息）**  
 本字段为日志信息的版本标识，取值为 10。  
 本字段只有在日志信息发往日志主机时才会存在。
- Module（模块名）**  
 本字段为生成该日志信息的功能模块的名称。模块列表可以通过在系统视图下输入命令 **info-center source ?** 进行查看。
- Level（信息等级）**

本字段为日志信息的等级，具体说明请参见 [表 1-1](#)。

- **Mnemonic**（助记符）

本字段为该日志信息的概述，是一个不超过 32 个字符的字符串。

- **Source**（定位信息）

本字段为该日志信息的产生者，是可选字段。本字段的取值为：IRF 的成员设备编号和日志发送者的源 IP。

- **Content**（信息文本）

本字段为该日志信息的具体内容。

## 1.2 FIPS相关说明

设备运行于 FIPS 模式时，本特性部分配置相对于非 FIPS 模式有所变化，具体差异请见本文相关描述。有关 FIPS 模式的详细介绍请参见“安全配置指导”中的“FIPS”。

## 1.3 信息中心配置任务简介

### 1.3.1 普通日志配置任务简介

普通日志配置任务如下：

- (1) [开启信息中心功能](#)

- (2) [将日志信息输出到指定方向](#)

请至少选择其中一项进行配置：

- [配置日志信息发送到控制台](#)
- [配置日志信息发送到监视终端](#)
- [配置日志信息发送到日志主机](#)
- [配置日志信息发送到日志缓冲区](#)
- [配置日志信息保存到日志文件](#)

- (3) （可选）[配置日志信息的最短保存时间](#)

- (4) （可选）[配置命令行输入回显功能](#)

- (5) （可选）[抑制日志信息输出](#)

请至少选择其中一项进行配置：

- [配置重复日志抑制功能](#)
- [禁止指定模块日志的输出](#)
- [禁止接口生成Link up/Link down日志信息](#)

- (6) （可选）[将系统日志封装成告警信息发送](#)

### 1.3.2 隐藏日志配置任务简介

隐藏日志配置任务如下：

- (1) [开启信息中心功能](#)

- (2) [将日志信息输出到指定方向](#)

请至少选择其中一项进行配置：

- [配置日志信息发送到日志主机](#)
- [配置日志信息发送到日志缓冲区](#)
- [配置日志信息保存到日志文件](#)
- (3) (可选) [配置日志信息的最短保存时间](#)
- (4) (可选) [抑制日志信息输出](#)

请至少选择其中一项进行配置：

- [配置重复日志抑制功能](#)
- [禁止指定模块日志的输出](#)

### 1.3.3 安全日志配置任务简介

安全日志配置任务如下：

- (1) [开启信息中心功能](#)
- (2) (可选) [抑制日志信息输出](#)

请至少选择其中一项进行配置：

- [配置重复日志抑制功能](#)
- [禁止指定模块日志的输出](#)
- (3) [配置安全日志功能](#)
  - [保存安全日志](#)
  - [管理安全日志文件](#)

### 1.3.4 诊断日志配置任务简介

诊断日志配置任务如下：

- (1) [开启信息中心功能](#)
- (2) (可选) [抑制日志信息输出](#)

请至少选择其中一项进行配置：

- [配置重复日志抑制功能](#)
- [禁止指定模块日志的输出](#)
- (3) [配置诊断日志信息保存到诊断日志文件](#)

### 1.3.5 调试跟踪日志配置任务简介

调试跟踪日志配置任务如下：

- (1) [开启信息中心功能](#)
- (2) (可选) [抑制日志信息输出](#)

请至少选择其中一项进行配置：

- [配置重复日志抑制功能](#)
- [禁止指定模块日志的输出](#)
- (3) [配置调试跟踪日志文件的大小](#)



## 1.4 开启信息中心功能

### 1. 功能简介

开启信息中心功能后，信息中心模块才能处理各业务模块生成的日志，用户才能看到设备生成的日志。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启信息中心功能。

```
info-center enable
```

缺省情况下，信息中心处于开启状态。

## 1.5 将日志信息输出到指定方向

### 1.5.1 配置日志信息发送到控制台

#### 1. 配置限制和指导

**terminal monitor**、**terminal debugging**、**terminal logging** 命令只对当前连接有效。当终端与设备重新建立连接后，这些命令会恢复到缺省情况。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置日志信息发送到控制台时的输出规则。

```
info-center source { module-name | default } console { deny | level  
severity }
```

缺省情况下，日志信息的输出规则请参见“[1.1.4 日志信息的缺省输出规则](#)”。

- (3) （可选）配置时间戳输出格式。

```
info-center timestamp { boot | date | none }
```

缺省情况下，信息的时间戳输出格式为 **date** 格式。

- (4) 退回到用户视图。

```
quit
```

- (5) 开启控制台对日志信息的监视功能。

```
terminal monitor
```

缺省情况下，允许日志信息输出到控制台。

- (6) 开启当前终端对调试信息的显示功能。

```
terminal debugging
```

缺省情况下，当前终端对调试信息的显示功能处于关闭状态。

- (7) 配置控制台显示日志信息的等级。

```
terminal logging level severity
```

缺省情况下，控制台显示的日志信息最低等级为 6（informational）。

## 1.5.2 配置日志信息发送到监视终端

### 1. 功能简介

监视终端是指以 AUX、VTY、TTY 类型用户线登录的用户终端。

### 2. 配置限制和指导

**terminal monitor**、**terminal debugging**、**terminal logging** 命令只对当前连接有效。当终端与设备重新建立连接后，这些命令会恢复到缺省情况。

### 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置日志信息发送到监视终端时的输出规则。

```
info-center source { module-name | default } monitor { deny | level  
severity }
```

缺省情况下，日志信息的输出规则请参见“[1.1.4 日志信息的缺省输出规则](#)”。

- (3) （可选）配置时间戳输出格式。

```
info-center timestamp { boot | date | none }
```

缺省情况下，信息的时间戳输出格式为 **date** 格式。

- (4) 退回到用户视图。

```
quit
```

- (5) 开启监视终端对日志信息的监视功能。

```
terminal monitor
```

缺省情况下，不允许日志信息输出到监视终端。

- (6) 开启当前终端对调试信息的显示功能。

```
terminal debugging
```

缺省情况下，当前终端对调试信息的显示功能处于关闭状态。

- (7) 配置监视终端显示日志信息的等级。

```
terminal logging level severity
```

缺省情况下，监视终端显示的日志信息最低等级为 6（informational）。

## 1.5.3 配置日志信息发送到日志主机

### 1. 配置限制和指导

设备支持通过以下功能将某些业务模块的日志发送给日志主机，这些功能按优先级从高到低的顺序依次为：

- (1) 快速日志输出
- (2) Flow 日志
- (3) 信息中心

对于同一业务模块，如果用户配置了高优先级的输出方式，则不再采用其他方式输出。哪些业务模块的日志支持快速日志输出，以及快速日志输出详细介绍请参见“网络管理和监控配置指导”中的“快速日志输出”；哪些业务模块的日志支持通过 Flow 日志输出，以及 Flow 日志的详细介绍请参见“网络管理和监控配置指导”中的“Flow 日志”。

## 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置日志信息发送到日志主机时的输出规则。

```
info-center source { module-name | default } loghost { deny | level  
severity }
```

缺省情况下，日志信息的输出规则请参见“[1.1.4 日志信息的缺省输出规则](#)”。

- (3) （可选）配置发送日志信息时使用的源 IP 地址。

```
info-center loghost source interface-type interface-number
```

缺省情况下，使用出接口的主 IP 地址作为发送的日志信息的源 IP 地址。

- (4) （可选）配置发往日志主机的日志信息的输出格式。

```
info-center format { unicom | cmcc }
```

缺省情况下，发往日志主机的日志信息的格式为非定制格式。

- (5) （可选）配置发送的日志信息的时间戳格式。

```
info-center timestamp loghost { date | iso [ with-timezone ] |  
no-year-date | none }
```

缺省情况下，发往日志主机的日志信息的时间戳输出格式为 **date** 格式。

- (6) 配置日志主机及相关参数。

```
info-center loghost { hostname | ipv4-address | ipv6 ipv6-address } [ port  
port-number ] [ dscp dscp-value ] [ facility local-number ]
```

缺省情况下，未配置日志主机及相关参数。

*port-number* 参数的值需要和日志主机侧的配置一致，否则，日志主机接收不到日志信息。

## 1.5.4 配置日志信息发送到日志缓冲区

- (1) 进入系统视图。

```
system-view
```

- (2) 配置日志信息发送到日志缓冲区时的输出规则。

```
info-center source { module-name | default } logbuffer { deny | level  
severity }
```

缺省情况下，日志信息的输出规则请参见“[1.1.4 日志信息的缺省输出规则](#)”。

- (3) （可选）配置时间戳输出格式。

```
info-center timestamp { boot | date | none }
```

缺省情况下，日志信息的时间戳输出格式为 **date** 格式。

- (4) 允许日志信息输出到日志缓冲区。

**info-center logbuffer**

缺省情况下，允许日志输出到日志缓冲区。

- (5) （可选）配置系统向日志缓冲区输出信息以及日志缓冲区的容量。

**info-center logbuffer size buffersize**

缺省情况下，日志缓冲区可存储 512 条信息。

## 1.5.5 配置日志信息保存到日志文件

### 1. 功能简介

通过使用本特性，用户可以将系统产生的日志信息保存到设备的日志文件中以便随时查看。

日志在保存到日志文件前，先保存在日志文件缓冲区。系统会按照指定的频率将日志文件缓冲区的内容写入日志文件，频率一般配置为 24 小时一次，在设备比较空闲的时候（比如清晨）进行保存，用户也可以手工触发保存。成功保存后，保存前的日志文件缓冲区里的内容会被清空。

设备在有存储需要时，会自动生成日志文件。日志文件有容量限制，当存储介质的存储空间不足或者存储介质有存储空间但日志文件的大小达到最大值，且日志文件写满保护功能处于关闭状态，系统会使用最新日志覆盖最旧日志。

日志文件有容量限制，当日志文件的大小达到最大值，且日志文件写满保护功能处于开启状态，不再覆盖旧日志，而是停止记录日志。



#### 提示

当尚未生成日志文件，且存储介质的剩余空间不足时，设备不会将新生成的日志存储到日志文件。请定期清理存储介质的存储空间，以免影响日志文件功能。

---

### 2. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 配置日志信息保存到日志文件时的输出规则。

**info-center source { module-name | default } logfile { deny | level severity }**

缺省情况下，日志信息的输出规则请参见“[1.1.4 日志信息的缺省输出规则](#)”。

- (3) 开启日志文件功能。

**info-center logfile enable**

缺省情况下，允许日志信息输出到日志文件。

- (4) （可选）开启日志文件写满保护功能。

**info-center logfile overwrite-protection [ all-port-powerdown ]**

缺省情况下，日志文件写满保护功能处于关闭状态。

本命令仅 FIPS 模式下支持。

- (5) （可选）配置单个日志文件最大能占用的存储空间的大小。

**info-center logfile size-quota size**

缺省情况下，单个日志文件最大能占用的存储空间为 10MB。

为了保证设备的正常运行，**info-center logfile size-quota** 配置的日志文件的大小最小不能低于 1MB，最大不能超过 10MB。

- (6) （可选）配置存储日志文件的目录。

**info-center logfile directory** *dir-name*

缺省情况下，存放日志文件的目录为 **flash:/logfile**。

该命令会在 IRF 重启或主从设备倒换后失效。

- (7) 将日志文件缓冲区中的内容保存到日志文件。请选择其中一项进行配置。

- 配置自动将日志文件缓冲区中的内容保存到日志文件。

**info-center logfile frequency** *freq-sec*

缺省情况下，设备自动保存日志文件的频率为 86400 秒。

- 在任意视图下执行以下命令，手动将日志文件缓冲区中的内容保存到日志文件。

**logfile save**

## 1.6 配置日志信息的最短保存时间

### 1. 功能简介

使用本特性可以确保重要日志在指定时间内不被新的日志覆盖。

日志文件和日志缓冲区有容量限制，当存储的日志达到容量限制时，系统会直接使用最新日志覆盖最早生成的日志。配置日志信息的最短保存时间后，系统在覆盖一条日志前，会检查该日志在系统中存在的时间（根据日志生成时间和当前系统时间计算）：

- 如果日志存在时间小于最短保存时间，则不删除该日志，新日志保存失败；
- 如果日志存在时间大于最短保存时间，则使用最新的日志覆盖本条日志。

### 2. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 配置日志信息的最短保存时间。

**info-center syslog min-age** *min-age*

缺省情况下，未配置日志信息的最短保存时间。

## 1.7 配置命令行输入回显功能

### 1. 功能简介

当用户进行命令行、参数或者 Y/N 确认信息输入时，如果被大量的日志信息打断，用户可能记不清已经输入了哪些字符串，还需要输入哪些字符串。使用命令行输入回显功能，能够协助用户配置。系统会在日志信息输出完毕后回显用户已有的输入或者 Y/N 确认信息，以便用户继续执行配置。

### 2. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 开启命令行输入回显功能。

```
info-center synchronous
```

缺省情况下，命令行输入回显功能处于关闭状态。

## 1.8 抑制日志信息输出

### 1.8.1 配置重复日志抑制功能

#### 1. 功能简介

当设备持续向某个方向发送同一条日志时，大量重复的信息会浪费设备资源和网络资源，并导致有用的信息被淹没，不利于设备的维护。为了避免此问题，可开启重复日志抑制功能。

开启重复日志抑制功能后，设备每产生一条新日志，会输出该日志，并开启重复日志抑制定时器：

- 如果在抑制周期内，设备后续连续生成的日志均与该日志相同（要求日志的如下字段均完全相同：模块名、信息等级、日志助记符、定位信息和信息文本），则系统会认为后续生成的日志是该日志的重复日志，不再输出。等抑制周期到达时，系统会生成一条专用日志来记录本周期内抑制的日志的内容和条数供用户查阅。
- 如果在日志抑制周期内有其它新日志信息产生时，设备会进行以下处理：
  - 系统会结束对上一条日志的抑制，并生成一条专用日志来记录抑制的日志的内容和条数供用户查阅。
  - 输出新的日志，并重启重复日志抑制定时器。
- 如果直到抑制周期结束，系统没有生成日志，则关闭重复日志抑制定时器，且不输出任何日志。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启重复日志抑制功能。

```
info-center logging suppress duplicates
```

缺省情况下，重复日志抑制功能处于关闭状态。

### 1.8.2 禁止指定模块日志的输出

#### 1. 功能简介

当系统日志较多时，对于用户不关心的日志，可禁止指定模块日志的输出，或者禁止指定模块特定助记符的日志输出。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 禁止指定模块日志的输出。

```
info-center logging suppress module module-name mnemonic { all |  
mnemonic-value }
```

缺省情况下，未禁止指定模块日志的输出。

### 1.8.3 禁止接口生成Link up/Link down日志信息

#### 1. 功能简介

缺省情况下，设备的所有接口在接口状态改变时都会生成 Link up/Link down 的日志信息。为了方便管理，用户可以根据实际情况禁止某些接口生成接口 Link up/Link down 的日志信息：

- 用户只关心某个或某些接口的状态时，可以使用该功能禁止其它接口生成 Link up/Link down 日志信息。
- 某个接口的状态因不稳定而频繁地改变，生成大量的 Link up/Link down 日志信息时，可以使用该功能禁止该接口生成 Link up/Link down 日志信息。

使用本特性后，如果接口状态改变，将不再生成接口 Link up/Link down 的日志信息。这样可能会影响用户监控接口状态，所以在一般情况下建议采用缺省配置。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入接口视图。

```
interface interface-type interface-number
```

- (3) 禁止接口生成 Link up/Link down 日志信息。

```
undo enable log updown
```

缺省情况下，允许所有接口在状态发生改变时生成接口 Link up 和 Link down 的日志信息。

## 1.9 将系统日志封装成告警信息发送

#### 1. 功能简介

使用本特性后，设备除了根据输出规则将日志输出到各方向外，还会将日志信息封装成告警信息，发送到 SNMP 模块和日志告警缓冲区。

- 发送到 SNMP 模块后，是否输出到目的主机，以 Trap 还是 Inform 形式发送，由 SNMP 模块的配置决定，关于告警信息以及 SNMP 配置的介绍请参见“网络管理和监控配置指导”中的“SNMP”。
- 发送到日志告警缓冲区后，用户可通过“日志告警缓冲区对应的 MIB 节点”来直接读取日志信息封装成的告警信息。用户还可通过 **info-center syslog trap buffersize** 命令来配置日志告警缓冲区的大小。

#### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 将系统日志封装成告警信息发送。

```
snmp-agent trap enable syslog
```

缺省情况下，系统日志不会封装成告警信息发送。

- (3) 配置日志告警缓冲区的大小。



```
info-center syslog trap buffersize buffersize
```

缺省情况下，日志告警缓冲区的大小为 1024 条。

## 1.10 配置安全日志功能

### 1.10.1 保存安全日志

#### 1. 功能简介

查看系统日志是了解设备状态、定位和排除网络问题的一个重要方法，而在系统日志中与设备安全相关的安全日志显得尤为重要。但通常情况下，安全日志与其它日志一同输出，经常被淹没在大量的系统日志中，很难识别、不便于查看。针对这个问题，系统提供了安全日志同步保存功能。安全日志同步保存功能的配置和安全日志文件的管理相互分离，安全日志文件实行专人专管。

开启安全日志同步保存功能后，系统将安全日志进行集中处理：当生成的日志信息中有安全日志，在不影响日志信息现有输出规则的前提下，系统会将安全日志信息同步保存到专用的安全日志文件。这样既实现了安全日志的集中管理，又有利于用户随时快捷地查看安全日志，了解设备状态。

安全日志会先被输出到安全日志文件缓冲区（**security-logfile buffer**），系统会按照配置中指定的频率将安全日志文件缓冲区的内容写入安全日志文件（安全日志管理员也可以手工触发保存）。当安全日志文件缓冲区里的内容成功保存到安全日志文件后，安全日志文件缓冲区会被立即清空。系统只支持单个安全日志文件。

#### 2. 配置限制和指导

安全日志文件写满并收到新的安全日志时，新安全日志会覆盖安全日志文件中的旧安全日志。为了防止安全日志的丢失，用户可以配置安全日志文件使用率告警上限。当达到上限时，系统会输出日志信息提醒用户，此时，用户可以使用安全日志管理员身份登录设备，将安全日志文件进行备份，以防止重要历史数据丢失。

#### 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启安全日志同步保存功能。

```
info-center security-logfile enable
```

缺省情况下，安全日志同步保存功能处于关闭状态。

- (3) 配置设备自动保存安全日志文件的频率。

```
info-center security-logfile frequency freq-sec
```

缺省情况下，设备自动保存安全日志文件的频率为 86400 秒。

- (4) （可选）配置单个安全日志文件最大能占用的存储空间的大小。

```
info-center security-logfile size-quota size
```

缺省情况下，单个安全日志文件最大能占用的存储空间的大小为 10MB。

- (5) （可选）配置安全日志文件使用率的告警上限。

```
info-center security-logfile alarm-threshold usage
```

缺省情况下，安全日志文件使用率的告警门限是 80。即当安全日志文件使用率达到 80% 时，系统会发出日志提醒用户。



## 1.10.2 管理安全日志文件

### 1. 配置限制和指导

只有具有安全日志管理员角色的用户登录设备后，才能管理安全日志文件。安全日志管理员的相关配置请参见“安全配置指导”中的“AAA”。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 修改存储安全日志文件的路径。

```
info-center security-logfile directory dir-name
```

缺省情况下，存储安全日志文件路径为存储设备根目录下的 **seclog** 文件夹。

该命令会在 IRF 重启或主从设备倒换后失效。

- (3) 在任意视图下执行以下命令，手动将安全日志文件缓冲区中的内容全部保存到安全日志文件。

```
security-logfile save
```

- (4) （可选）在任意视图下执行以下命令，显示安全日志文件的概要信息。

```
display security-logfile summary
```

## 1.11 配置诊断日志信息保存到诊断日志文件

### 1. 功能简介

通过使用本特性，用户可以将系统产生的诊断日志信息保存到设备的诊断日志文件中以便随时查看。诊断日志在保存到诊断日志文件前，先保存在诊断日志文件缓冲区。系统会按照指定的频率将诊断日志文件缓冲区的内容写入诊断日志文件，频率一般配置为 24 小时一次，在设备比较空闲的时候（比如清晨）进行保存，用户也可以手工触发保存。成功保存后，保存前的诊断日志文件缓冲区里的内容会被清空。

诊断日志文件有容量限制，当诊断日志文件的大小达到最大值时，系统会使用最新日志覆盖最旧日志。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启诊断日志保存功能。

```
info-center diagnostic-logfile enable
```

缺省情况下，诊断日志同步保存功能处于开启状态。

- (3) （可选）配置单个诊断日志文件最大可占用的存储空间的大小。

```
info-center diagnostic-logfile quota size
```

缺省情况下，单个诊断日志文件最大能占用的存储空间为 10MB。

为了保证设备的正常运行，**info-center diagnostic-logfile quota** 配置的日志文件的大小最小不能低于 1MB，最大不能超过 10MB。

- (4) (可选) 配置存储诊断日志文件的目录。

**info-center diagnostic-logfile directory** *dir-name*

缺省情况下，存储诊断日志文件路径为 **flash:/diagfile**。

该命令会在 IRF 重启或 Master 和 Slave 倒换后失效。

- (5) 将诊断日志文件缓冲区中的内容保存到诊断日志文件。

- 配置自动保存的频率将诊断日志文件缓冲区中的内容保存到诊断日志文件。

**info-center diagnostic-logfile frequency** *freq-sec*

缺省情况下，设备自动保存诊断日志文件的频率为 **86400** 秒。

- 在任意视图下执行以下命令，手动将诊断日志文件缓冲区中的内容保存到诊断日志文件。

**diagnostic-logfile save**

## 1.12 配置调试跟踪日志文件的大小

### 1. 功能简介

设备在调试过程中，会产生大量的调试跟踪日志。如果调试跟踪日志文件存储空间太小，可能会导致日志被很快覆盖，不利于定位问题。使用本特性，可以用来配置调试跟踪日志文件最大能占用的存储空间的大小。系统只支持单个跟踪调试日志文件。

### 2. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 配置调试跟踪日志文件最大可占用的存储空间的大小。

**info-center trace-logfile quota** *size*

缺省情况下，调试跟踪日志文件最大能占用的存储空间的大小为 **1MB**。

## 1.13 信息中心显示和维护

在完成上述配置后，在任意视图下执行 **display** 命令可以显示配置后信息中心的运行情况，通过查看显示信息验证配置的效果。

在用户视图下执行 **reset logbuffer** 命令可以将日志缓冲区的统计信息清除。

表1-9 信息中心显示和维护

操作	命令
显示诊断日志文件的配置	<b>display diagnostic-logfile summary</b>
显示各个输出方向的信息	<b>display info-center</b>
显示日志缓冲区的状态和日志缓冲区记录的日志信息	<b>display logbuffer</b> [ <b>reverse</b> ] [ <b>level severity</b>   <b>size buffersize</b>   <b>slot slot-number</b> ] *
显示日志缓冲区的概要信息	<b>display logbuffer summary</b> [ <b>level severity</b>   <b>slot slot-number</b> ] *
显示日志文件的配置	<b>display logfile summary</b>

操作	命令
显示安全日志文件的概要信息（只有安全日志管理员才能执行该命令）	<code>display security-logfile summary</code>
清除日志缓冲区内的信息	<code>reset logbuffer</code>

## 1.14 信息中心典型配置举例

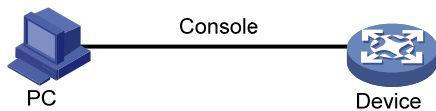
### 1.14.1 日志发送到控制台的配置举例

#### 1. 组网需求

将信息等级高于等于 **warning** 的日志信息发送到控制台上；  
允许输出日志信息的模块为 **FTP**。

#### 2. 组网图

图1-1 日志信息发送到控制台配置组网图



#### 3. 配置步骤

# 开启信息中心功能。

```
<Device> system-view
[Device] info-center enable
```

# 关闭控制台方向所有模块日志信息的输出开关。

```
[Device] info-center source default console deny
```



#### 说明

由于系统对各方向允许输出的日志信息的缺省情况不一样，所以配置前必须将所有模块指定方向（本例为 **console**）上日志信息的输出开关关闭，再根据当前的需求配置输出规则，以免输出太多不需要的信息。

# 配置输出规则：允许 **FTP** 模块的、等级高于等于 **warning** 的日志信息输出。

```
[Device] info-center source ftp console level warning
[Device] quit
```

# 开启终端显示功能（该功能缺省情况下为开启状态）。

```
<Device> terminal logging level 6
<Device> terminal monitor
```

The current terminal is enabled to display logs.

以上命令配置成功后，如果指定的模块产生了日志信息，信息中心会自动把这些日志发送到控制台，在控制台的屏幕上显示。

## 1.14.2 日志发送到Unix日志主机的配置举例

### 1. 组网需求

将系统的日志信息发送到 Unix 日志主机；

将信息等级高于等于 **informational** 的日志信息将会发送到日志主机上；

仅允许输出日志信息的模块为 **FTP**。

### 2. 组网图

图1-2 日志信息发送到 Unix 日志主机配置组网图



### 3. 配置步骤

配置前请确保 **Device** 和 **PC** 之间路由可达。（具体配置步骤略）

#### (1) Device 上的配置

# 开启信息中心功能。

```
<Device> system-view
```

```
[Device] info-center enable
```

# 配置发送日志信息到 IP 地址为 **1.2.0.1/16** 的日志主机，日志主机记录工具为 **local4**。

```
[Device] info-center loghost 1.2.0.1 facility local4
```

# 关闭 **loghost** 方向所有模块日志信息的输出开关。

```
[Device] info-center source default loghost deny
```



说明

由于系统对各方向允许输出的日志信息的缺省情况不一样，所以配置前必须将所有模块指定方向（本例为 **loghost**）上日志信息的输出开关关闭，再根据当前的需求配置输出规则，以免输出太多不需要的信息。

# 配置输出规则：允许 **FTP** 模块的、等级高于等于 **informational** 的日志信息输出到日志主机（注意：允许输出信息的模块由产品决定）。

```
[Device] info-center source ftp loghost level informational
```

#### (2) 日志主机上的配置

下面以 **Solaris** 操作系统上的配置为例介绍日志主机上的配置，在其它厂商的 **Unix** 操作系统上的配置操作基本类似。

a. 以超级用户的身份登录日志主机。

b. 在 **/var/log/** 路径下为 **Device** 创建同名日志文件夹 **Device**，在该文件夹创建文件 **info.log**，用来存储来自 **Device** 的日志。

```
# mkdir /var/log/Device
```

```
# touch /var/log/Device/info.log
```

c. 编辑 **/etc/** 路径下的文件 **syslog.conf**，添加以下内容。

```
# Device configuration messages
local4.info /var/log/Device/info.log
```

以上配置中，**local4** 表示日志主机接收日志的工具名称，**info** 表示信息等级。Unix 系统会把等级高于等于 **informational** 的日志记录到 **/var/log/Device/info.log** 文件中。



#### 说明

在编辑 **/etc/syslog.conf** 时应注意以下问题：

- 注释必须独立成行，并以字符 **#** 开头。
- 在文件名之后不得有多余的空格。
- **/etc/syslog.conf** 中指定的工具名称及信息等级与 Device 上 **info-center loghost** 和 **info-center source** 命令的相应参数的指定值要保持一致，否则日志信息可能无法正确输出到日志主机上。

- d. 查看系统守护进程 **syslogd** 的进程号，中止 **syslogd** 进程，并重新用 **-r** 选项在后台启动 **syslogd**，使修改后配置生效。

```
# ps -ae | grep syslogd
147
# kill -HUP 147
# syslogd -r &
```

进行以上操作之后，Device 的日志信息会输出到 PC，PC 会将这些日志信息存储到相应的文件中了。

### 1.14.3 日志发送到Linux日志主机的配置举例

#### 1. 组网需求

系统的日志信息发送到 Linux 日志主机上；

将信息等级高于等于 **informational** 的日志信息发送到日志主机上；

仅允许输出日志信息的模块为 **FTP**。

#### 2. 组网图

图1-3 日志信息发送到 Linux 日志主机配置组网图



#### 3. 配置步骤

配置前请确保 Device 和 PC 之间路由可达，具体配置步骤略。

##### (1) Device 上的配置

# 开启信息中心功能。

```
<Device> system-view
```

```
[Device] info-center enable
```

#配置发送日志信息到 IP 地址为 **1.2.0.1/16** 的日志主机，日志主机记录工具为 **local5**。

```
[Device] info-center loghost 1.2.0.1 facility local5
```

# 关闭 loghost 方向所有模块日志信息的输出开关。

```
[Device] info-center source default loghost deny
```

---



#### 说明

由于系统对各方向允许输出的日志信息的缺省情况不一样，所以配置前必须将所有模块的需求方向（本例为 loghost）上日志信息的输出开关关闭，再根据当前的需求配置输出规则，以免输出太多不需要的信息。

---

# 配置输出规则：允许 FTP 模块、等级高于等于 informational 的日志信息输出到日志主机。

```
[Device] info-center source ftp loghost level informational
```

#### (2) 日志主机上的配置

下面以 Solaris 操作系统上的配置为例介绍日志主机上的配置，在其它厂商的 Unix 操作系统上的配置操作基本类似。

a. 以超级用户的身份登录日志主机。

b. 在 /var/log/ 路径下为 Device 创建同名日志文件夹 Device，在该文件夹创建文件 info.log，用来存储来自 Device 的日志。

```
# mkdir /var/log/Device
```

```
# touch /var/log/Device/info.log
```

c. 编辑 /etc/ 路径下的文件 syslog.conf，添加以下内容。

```
# Device configuration messages
```

```
local5.info /var/log/Device/info.log
```

以上配置中，local5 表示日志主机接收日志的工具名称，info 表示信息等级。Linux 系统会把等级高于等于 informational 的日志记录到 /var/log/Device/info.log 文件中。

---



#### 说明

在编辑 /etc/syslog.conf 时应注意以下问题：

- 注释必须独立成行，并以字符 # 开头。
  - 在文件名之后不得有多余的空格。
  - /etc/syslog.conf 中指定的工具名称及信息等级与 Device 上 info-center loghost 和 info-center source 命令的相应参数的指定值要保持一致，否则日志信息可能无法正确输出到日志主机上。
- 

d. 第四步：查看系统守护进程 syslogd 的进程号，中止 syslogd 进程，并重新用 -r 选项在后台启动 syslogd，使修改后配置生效。对 Linux 日志主机，必须保证 syslogd 进程是以 -r 选项启动。

```
# ps -ae | grep syslogd
```

```
147
```

```
# kill -9 147
```

```
# syslogd -r &
```

进行以上操作之后，系统就可以在相应的文件中记录日志信息了。

# 目 录

1 Packet Capture .....	1-1
1.1 Packet Capture简介 .....	1-1
1.2 构建捕获过滤规则.....	1-1
1.2.1 过滤规则的构成 .....	1-1
1.2.2 捕获过滤规则关键字 .....	1-1
1.2.3 捕获过滤规则操作符 .....	1-2
1.2.4 捕获过滤规则表达式 .....	1-4
1.3 构建显示过滤规则.....	1-4
1.3.1 显示过滤规则关键字 .....	1-5
1.3.2 显示过滤规则操作符 .....	1-6
1.3.3 显示过滤规则表达式 .....	1-6
1.4 配置准备.....	1-7
1.5 配置报文捕获.....	1-7
1.5.1 配置限制和指导 .....	1-7
1.5.2 配置报文捕获并将捕获的报文保存到本地 .....	1-7
1.5.3 配置报文捕获并将捕获的报文显示到登录终端 .....	1-8
1.6 停止报文捕获.....	1-8
1.7 解析并显示报文文件.....	1-8
1.8 Packet Capture典型配置举例 .....	1-8
1.8.1 报文捕获配置举例 .....	1-8

# 1 Packet Capture

## 1.1 Packet Capture简介

Packet Capture 是一种报文捕获及分析特性。使用该特性能够捕获设备接口的入方向报文，并将这些报文信息直接在用户的登录终端上显示或者存储为 pcap 格式的文件，方便用户后续查看。使用该特性还可以解析 pcap 或 pcapng 报文文件。

## 1.2 构建捕获过滤规则

### 1.2.1 过滤规则的构成

Packet Capture 可以使用过滤表达式指定过滤规则，对需要捕获、显示的报文进行过滤，仅捕获、显示用户关心的报文。

过滤规则由关键字和操作符组合而成：

- 关键字又分为：
  - 常量关键字：该类关键字为固定的字符串。使用时，用户需完整输入该关键字。
  - 变量关键字：该类关键字形式固定，但内容可变。用户可自定义该关键字的取值。
- 操作符分为逻辑操作符、运算操作符和比较操作符。

关于捕获过滤规则的详细介绍请参见网页：<http://wiki.wireshark.org/CaptureFilters>。

关于显示过滤规则的详细介绍请参见网页：<http://wiki.wireshark.org/DisplayFilters>。

### 1.2.2 捕获过滤规则关键字

#### 1. 常量关键字

表1-1 常量关键字

常量关键字类型	描述	关键字
协议	捕获指定协议的报文 如果没有指明协议类型，则捕获Packet Capture支持的所有协议的报文	支持的协议有：arp、icmp、ip、ipv6、tcp、udp等
报文传输方向	捕获指定传输方向的报文 如果没有指定本关键字，缺省报文传输方向为源或目的方向。比如port 23等价于src or dst port 23	<ul style="list-style-type: none"><li>• src: 表示源方向</li><li>• dst: 表示目的方向</li><li>• src or dst: 表示源或目的方向</li></ul>
报文传输方向类型	捕获指定的报文传输方向类型的报文 如果没有指定本类关键字，缺省报文传输方向类型为主机。比如src 2.2.2.2等价于src host 2.2.2.2	<ul style="list-style-type: none"><li>• host: 表示主机</li><li>• net: 表示网段</li><li>• port: 表示端口号</li><li>• portrange: 表示端口号范围</li></ul>



常量关键字类型	描述	关键字
特殊关键字	-	<ul style="list-style-type: none"> <li>• <b>broadcast</b>: 表示捕获广播报文</li> <li>• <b>multicast</b>: 表示捕获组播报文、广播报文</li> <li>• <b>less</b>: 表示小于等于</li> <li>• <b>greater</b>: 表示大于等于</li> <li>• <b>len</b>: 表示报文长度</li> <li>• <b>vlan</b>: 表示捕获 VLAN 报文</li> </ul>

## 2. 变量关键字

捕获过滤规则的变量关键字不可以单独使用，其前需要使用常量关键字对其进行修饰。

协议类型常量关键字、**broadcast** 和 **multicast** 关键字不能对变量关键字进行修饰。其它的常量关键字不可单独使用，其后需要使用变量关键字。

表1-2 变量关键字

变量关键字类型	举例
整型	使用二进制、八进制、十进制或十六进制形式表示。例如： <b>port 23</b> ，表示端口号为23
整型范围	使用二进制、八进制、十进制、十六进制形式和“-”表示。例如： <b>portrange 100-200</b> ，表示端口号范围为100到200
IPv4地址	使用点分十进制格式表示。例如： <b>src 1.1.1.1</b> ，表示源主机IPv4地址是1.1.1.1
IPv6地址	使用冒号分十六进制格式表示。例如： <b>dst host 1::1</b> ，表示报文的目的地主机IPv6地址是1::1
IPv4网段	使用IPv4地址和掩码或者IPv4网络号表示。以下两种表达式等价： <ul style="list-style-type: none"> <li>• <b>src 1.1.1.1</b>，表示源主机的 IPv4 网段为 1.1.1</li> <li>• <b>src net 1.1.1.0/24</b>，表示源主机的 IPv4 网段为 1.1.1.0/24</li> </ul>
IPv6网段	使用IPv6地址和网络前缀表示。例如： <b>dst net 1::/64</b> ，表示目的IPv6网段为1::/64 需要注意的是，指定IPv6网段变量关键字时，必须指定 <b>net</b> 常量关键字

## 1.2.3 捕获过滤规则操作符

### 1. 逻辑操作符

逻辑操作符的逻辑运算顺序为从左到右，其中非操作符优先级最高，与操作符和或操作符的优先级相同。

表1-3 逻辑操作符

逻辑操作符	描述
<b>!或者not</b>	非操作符。表示对捕获过滤规则取反操作
<b>&amp;&amp;或者and</b>	与操作符。表示连接多个捕获过滤规则。当此操作符连接多个过滤规则时，报文符合此操作符连接的全部过滤规则，才会过滤成功，否则，过滤失败

逻辑操作符	描述
或者or	或操作符。表示对多个捕获过滤规则进行选择，只要满足一个过滤规则，则过滤成功，否则，过滤失败

## 2. 运算操作符

表1-4 运算操作符

运算操作符	描述
+	加法运算符，用来将其两侧的值加到一起
-	减法运算符，用来将它前面的数值减去它后面的数值
*	乘法运算符，用来将其两侧的值相乘
/	除法运算符，用来将其左边的值被右边的值
&	按位与，用来将其两侧数值逐位进行比较产生一个新值。对于每一位，只有两个操作数的对应位都为1时结果才为1
	按位或，用来将其两侧的操作数逐位进行比较产生一个新值。对于每一位，如果其中任意操作数中对应的位为1，那么结果位就为1
<<	按位左移，用来将其左侧操作数的每位向左移动，移动的位数由其右侧操作数指定
>>	按位右移，用来将其左侧操作数的每位向右移动，移动的位数由其右侧操作数指定
[ ]	取位运算符，与协议类型关键字结合使用。例如：ip[6]，表示IP报文偏移6个字节后，取得的一个字节的值

## 3. 比较操作符

表1-5 比较操作符分类

比较操作符	描述
=	相等，判断两侧操作数是否相等。例如：ip[6]=0x1c，表示捕获IPv4报文数据域偏移6字节，取得的一个字节值为0x1c的报文
!=	不等，判断两侧操作数是否不等。例如：len!=60，表示捕获报文长度不等于60字节的报文
>	大于，判断左侧操作数大于右侧操作数。例如：len>100，表示捕获报文长度大于100字节的报文
<	小于，判断左侧操作数小于右侧操作数。例如：len<100，表示捕获报文长度小于100字节的报文
>=	大于等于，判断左侧操作数大于等于右侧操作数；与常量关键字greater等价。例如：len>=100，表示捕获报文长度大于等于100字节的报文
<=	小于等于，判断左侧操作数小于等于右侧操作数；与常量关键字less等价。例如：len<=100，表示捕获报文长度小于等于100字节的报文

## 1.2.4 捕获过滤规则表达式

### 1. 逻辑操作符表达式

由关键字和逻辑运算符组合的捕获过滤表达式。例如：`not port 23 and not port 22`，表示捕获端口号既不是 23，又不是 22 的报文；`port 23 or icmp`，表示捕获端口号是 23 或 ICMP 协议的报文。

由逻辑操作符连接的多个变量关键字，可以使用同一个常量关键字进行修饰（就近原则），例如：`src 192.168.56.1 or 192.168.27`，表示捕获的源 IPv4 地址为 192.168.56.1 或者源 IPv4 网段为 192.168.27 的报文。上述表达式与“`src 192.168.56.1 or src 192.168.27`”等价。

### 2. `expr relop expr`表达式

由关键字、运算操作符和比较操作符组合的捕获过滤表达式。其中，`expr` 是算术表达式；`relop` 为比较操作符。例如：`len+100>=200`，表示捕获长度大于等于 100 字节的报文。

### 3. `proto [ expr:size ]`表达式

由协议类型关键字和运算操作符“`[ ]`”组合的捕获过滤表达式。其中，`proto` 表示协议类型，`expr` 为算术表达式，表示偏移量，`size` 为整数，表示字节个数，缺省值为 1。`proto [ expr:size ]` 的返回值为从 `proto` 协议报文数据区域起始位置，偏移 `expr` 个字节开始，取 `size` 个字节的数据。例如：`ip[0]&0xf != 5`，表示捕获第一个字节与 0x0f 按位相与得到的值不是 5 的 IP 报文。

`expr:size` 也可以使用表达式的名称表示。例如：`icmp` 表示 ICMP 报文的类型域，则表达式：`icmp [icmptype]=0x08`，表示捕获 icmp 的 `type` 字段的值为 0x08 的报文。

### 4. `vlan vlan_id`表达式

由关键字 `vlan`、逻辑操作符等组合的捕获过滤表达式。其中，`vlan_id` 为整型，表示 VLAN 编号。例如，`vlan 1 and ip4`，表示捕获 VLAN 编号为 1 的 IPv4 报文。

需要注意的是：

- 对于带 `VLAN tag` 且接口允许通过的报文，必须使用此类捕获过滤表达式且关键字 `vlan` 要在其它捕获过滤条件之前指定，否则不能正常过滤。例如：`vlan 3 and src 192.168.1.10 and dst 192.168.1.1`，表示捕获 VLAN 3 内、192.168.1.10 发往 192.168.1.1 的报文。
- 对于接口收到的不带 `VLAN tag` 的报文，设备会在报文头中添加 `VLAN tag`，为了捕获该类报文，必须在捕获过滤规则中设置过滤条件为“`vlan xx`”。对于三层报文，`xx` 为报文出接口的缺省 VLAN ID；对于二层报文，`xx` 为入接口的缺省 VLAN ID。

## 1.3 构建显示过滤规则

当进行显示过滤时，所有报文仍然保存在捕获报文文件中；显示过滤只是将符合显示过滤条件的报文显示出来，不会改变文件的内容。

## 1.3.1 显示过滤规则关键字

### 1. 常量关键字

表1-6 常量关键字

常量关键字类型	描述	关键字
协议	显示指定协议的报文 如果没有指明协议类型，则显示捕获的所有协议的报文	支持的协议有：eth、ip、ipv6、tcp、udp、icmp、http、ftp、telnet等
报文字段	指定报文的特定字段 使用点“.”表示包含关系，格式为： <i>protocol.field[.level1-subfield]...[.leveln-subfield]</i>	例如： <ul style="list-style-type: none"><li>tcp.flags.syn 表示 tcp 协议报文 flags 字段中的 syn 位</li><li>tcp.port 表示 tcp 协议的 port 字段</li></ul>

### 2. 变量关键字

报文字段的取值为变量关键字。报文的各个字段具有不同的类型，如 [表 1-7](#) 所示。

表1-7 变量关键字的类型

字段类型	举例
整型	将整型字段值用二进制、八进制、十进制、十六进制形式表示。以下几种表达方式等价： <ul style="list-style-type: none"><li>ip.len le 1500</li><li>ip.len le 02734</li><li>ip.len le 0x436</li></ul> 表示显示IP报文长度小于等于1500字节的报文信息
布尔变量	不使用其它操作符，单独使用报文字段，则默认指定字段的类型为布尔类型。例如：tcp.flags.syn，表示如果捕获到的报文存在tcp.flags.syn字段，则表达式的值为真，显示过滤成功；否则为假，显示过滤失败
MAC地址（6字节）	MAC地址使用以下三种分隔符表示：分号“:”、点“.”或者破折号“-”；分隔符可以在两个或者四个字节间使用。以下几种方式等价： <ul style="list-style-type: none"><li>eth.dst==ff:ff:ff:ff:ff:ff</li><li>eth.dst==ff-ff-ff-ff-ff-ff</li><li>eth.dst ==ffff.ffff.ffff</li></ul> 以上的显示过滤表达式表示显示目的MAC地址为ffff.ffff.ffff的报文信息
IPv4地址	IPv4地址使用点分十进制格式表示。例如： <ul style="list-style-type: none"><li>ip.addr==192.168.0.1，表示显示源或者目的 IP 地址为 192.168.0.1 的报文信息</li><li>ip.addr==129.111.0.0/16，表示显示源或者目的 IP 地址属于 129.111 网段的报文信息</li></ul>
IPv6地址	IPv6地址使用冒号分十六进制格式表示。例如： <ul style="list-style-type: none"><li>ipv6.addr==1::1 表示显示源或者目的 IPv6 地址为 1::1 的报文信息</li><li>ipv6.addr==1::/64 表示显示源或者目的 IPv6 地址属于 1::/64 网段的报文信息</li></ul>

字段类型	举例
字符串	一些报文字段类型为字符串。例如， <code>http.request version=="HTTP/1.1"</code> 表示显示http报文的request version字段为字符串HTTP/1.1的报文信息，双引号表示报文字段内容为字符串

## 1.3.2 显示过滤规则操作符

### 1. 逻辑操作符

逻辑操作符的逻辑运算顺序为从左到右。优先级从高到低依次为：括号操作符、非操作符、与操作符和或操作符，其中与操作符和或操作符的优先级相同。

表1-8 显示过滤逻辑操作符

英文	描述和举例
!或not	非操作符。表示对显示过滤规则取反操作
&&或and	与操作符。表示连接多个显示过滤规则
或or	或操作符。表示对多个显示过滤规则进行选择
[ ]	括号操作符。与协议名称组合使用，详细介绍请参见“ <a href="#">1.3.3 4. protocol...表达式</a> ”

### 2. 比较操作符

表1-9 显示过滤比较操作符

分类	描述和举例
eq或==	相等，判断两侧操作数是否相等。例如： <code>ip.src==10.0.0.5</code> ，表示显示源IP地址为10.0.0.5的报文信息
ne或!=	不等，判断两侧操作数是否不等。例如： <code>ip.src!=10.0.0.5</code> ，表示显示源IP地址不是10.0.0.5的报文信息
gt或>	大于，判断左侧操作数大于右侧操作数。例如： <code>frame.len&gt;100</code> ，表示显示捕获的帧长度大于100字节的帧信息
lt或<	小于，判断左侧操作数小于右侧操作数。例如： <code>frame.len&lt;100</code> ，表示显示捕获的帧长度小于100字节的帧信息
ge 或>=	大于等于，判断左侧操作数大于等于右侧操作数。例如： <code>frame.len ge 0x100</code> ，表示显示捕获的帧长度大于等于256字节的帧信息
le或<=	小于等于，判断左侧操作数小于等于右侧操作数。例如： <code>frame.len le 0x100</code> 表示显示捕获的帧长度小于等于256字节的帧信息

## 1.3.3 显示过滤规则表达式

### 1. 逻辑操作符表达式

由关键字和逻辑运算符组合的显示过滤表达式。例如：`ftp or icmp`，表示显示所有ftp协议和icmp协议报文信息。

## 2. 比较操作符表达式

由关键字和比较运算符组合的显示过滤表达式。例如：`ip.len<=28`，表示显示所有 IP 报文的长度字段小于等于 28 字节的 IP 报文。

## 3. 报文字段表达式

只由报文字段组成的显示过滤表达式，作用是显示存在某一具体字段的报文信息。例如：`tcp.flags.syn`，表示显示所有存在 `tcp.flags.syn` 位的报文。

## 4. `proto[...]`表达式

由协议类型和运算操作符 “[ ]” 组合的显示过滤表达式，`proto[...]` 的类型为十六进制整型，其中，`proto` 表示协议类型、字段。括号操作符内容有以下两种表达方式：

- `[n:m]`，`n` 表示偏移位置，`m` 表示指定的字节数；表示从偏移 `n` 个字节位置开始取后面 `m` 个字节数据。例如：`eth.src[0:3]==00:00:83`，表示源 MAC 地址的前三个字节分别为 0x00、0x00、0x83。
- `[n-m]`，`n` 表示偏移起始位置，`m` 表示偏移结束位置；表示从偏移 `n` 个字节位置取到第 `m` 个字节位置，共取 `m-n+1` 个字节数据。例如：`eth.src[1-2]==00:83`，表示 MAC 地址的第二个字节和第三个字节分别为 0x00、0x83。
- `[n]`，与 `[n:1]` 等价，表示取偏移 `n` 个字节位置的一个字节数据。例如：`eth.src[2]==83`，表示 MAC 地址的第三个字节为 0x83。

## 1.4 配置准备

- (1) 使用 `boot-loader` 或 `install` 命令安装 Packet Capture 特性软件包。关于 `boot-loader` 和 `install` 命令的详细介绍请参见“基础配置命令参考”中的“软件升级”。
- (2) 重新登录设备。

## 1.5 配置报文捕获

### 1.5.1 配置限制和指导

在使用本功能的过程中，会阻断当前配置终端的输入，用户不能输入其它命令对设备进行操作，只有当报文捕获/显示结束时，配置终端才能输入其它命令。在大流量背景下退出捕获报文，可能会有延时。

### 1.5.2 配置报文捕获并将捕获的报文保存到本地

请在用户视图执行以下命令，配置报文捕获并将捕获的报文保存到本地。

```
packet-capture interface interface-type interface-number [ capture-filter  
capt-expression | limit-captured-frames limit | limit-frame-size bytes |  
autostop filesize kilobytes | autostop duration seconds | autostop files  
numbers | capture-ring-buffer filesize kilobytes | capture-ring-buffer  
duration seconds | capture-ring-buffer files numbers ] * write filepath [ raw  
| { brief | verbose } ] *
```

### 1.5.3 配置报文捕获并将捕获的报文显示到登录终端

请在用户视图执行以下命令，配置报文捕获并将捕获的报文显示到登录终端。

```
packet-capture interface interface-type interface-number [ capture-filter capt-expression | display-filter disp-expression | limit-captured-frames limit | limit-frame-size bytes | autostop duration seconds ] * [ raw | { brief | verbose } ] *
```

## 1.6 停止报文捕获

### 1. 功能简介

在配置报文捕获时，用户可通过参数来实现自动停止报文捕获。在报文捕获过程中，使用本功能可手工停止报文捕获。

### 2. 配置步骤

执行快捷键<Ctrl+C>，可结束报文捕获。

## 1.7 解析并显示报文文件

### 1. 功能简介

用户将捕获的报文保存到报文文件后，可以：

- 将文件上传到 FTP/TFTP 服务器，再通过第三方软件 Wireshark 解析并显示报文文件的内容。
- 使用 **packet-capture read** 命令，在设备本地解析并显示报文文件的内容。

### 2. 配置限制和指导

使用快捷键<Ctrl+C>可停止解析/显示本地报文文件。

### 3. 配置步骤

请在用户视图执行以下命令，解析并显示本地报文文件。

```
packet-capture read filepath [ display-filter disp-expression ] [ raw | { brief | verbose } ] *
```

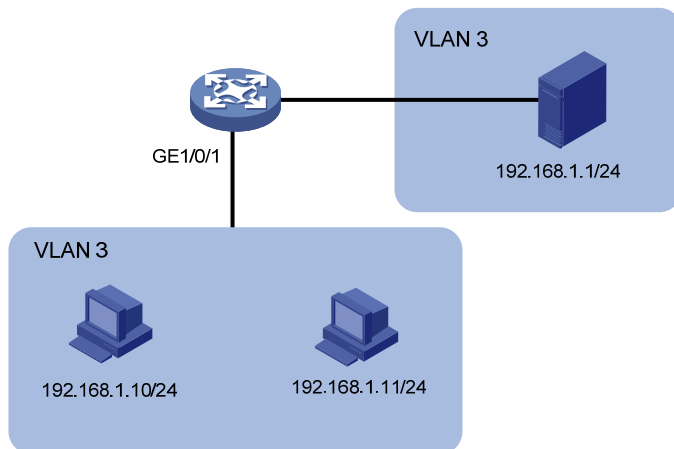
## 1.8 Packet Capture典型配置举例

### 1.8.1 报文捕获配置举例

#### 1. 组网需求

在设备的二层接口 GigabitEthernet1/0/1 上开启报文捕获功能。需要捕获接口 GigabitEthernet1/0/1 入方向上的、VLAN 3 的、192.168.1.10 到 192.168.1.1 以及 192.168.1.11 到 192.168.1.1 的所有软件转发报文和硬件转发报文。

## 2. 组网图



## 3. 配置步骤

- (1) 安装 **Packet Capture** 特性软件包，以便用户可以配置报文捕获功能

# 查看设备的版本信息，准备和设备当前运行的 **Boot** 包、**System** 包兼容的 **Packet Capture** 特性软件包。

```
<Device> display version
H3C Comware Software, Version 7.1.070, Demo 01
Copyright (c) 2004-2018 New H3C Technologies Co., Ltd. All rights reserved.
H3C XXX uptime is 0 weeks, 0 days, 5 hours, 33 minutes
Last reboot reason : Cold reboot
Boot image: flash:/boot-01.bin
Boot image version: 7.1.070, Demo 01
    Compiled Oct 20 2016 16:00:00
System image: flash:/system-01.bin
System image version: 7.1.070, Demo 01
    Compiled Oct 20 2016 16:00:00
其它显示信息略……。
```

# 从 IP 地址为 192.168.1.1 的 TFTP 服务器上下载 **Packet Capture** 特性软件包 **packet-capture-01.bin**。

```
<Device> tftp 192.168.1.1 get packet-capture-01.bin
Press CTRL+C to abort.

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload    Total   Spent    Left   Speed
100 11.3M    0 11.3M    0     0   155k      0  --:--:--  0:01:14 --:--:-- 194k
Writing file...Done.
```

# 给 IRF 中的所有成员设备（此处以 **slot 1** 和 **slot 2** 为例）都安装 **Packet Capture** 特性软件包，并让该软件包在设备重启后能够继续生效。

```
<Device> install activate feature flash:/packet-capture-01.bin slot 1
Verifying the file flash:/packet-capture-01.bin on slot 1....Done.
Identifying the upgrade methods....Done.
Upgrade summary according to following table:
```



flash:/packet-capture-01.bin

Running Version	New Version
None	Demo 01

Slot	Upgrade Way
1	Service Upgrade

Upgrading software images to compatible versions. Continue? [Y/N]:y

This operation might take several minutes, please wait.....Done.

<Device> install activate feature flash:/packet-capture-01.bin slot 2

Verifying the file flash:/packet-capture-01.bin on slot 2....Done.

Identifying the upgrade methods....Done.

Upgrade summary according to following table:

flash:/packet-capture-01.bin

Running Version	New Version
None	Demo 01

Slot	Upgrade Way
2	Service Upgrade

Upgrading software images to compatible versions. Continue? [Y/N]:y

This operation might take several minutes, please wait.....Done.

<Device> install commit

This operation will take several minutes, please wait.....Done.

# 重新登录设备，以使用户能够执行 **packet-capture interface** 和 **packet-capture read** 命令行。

- (2) 在接口 GigabitEthernet1/0/1 的入方向上应用 QoS 策略，用来限制只捕获 192.168.1.10 到 192.168.1.1、192.168.1.11 到 192.168.1.1 的硬件转发报文。（二层接口上的大部分流量为硬件转发报文，对于硬件报文，需要镜像到 CPU 才能捕获；对于软件转发报文，不需要配置 QoS 策略，直接开启报文捕获功能即可）

# 创建 IPv4 高级 ACL 3000，用来匹配 192.168.1.10 到 192.168.1.1、192.168.1.11 到 192.168.1.1 的报文。

<Device> system-view

[Device] acl advanced 3000

[Device-acl-ipv4-adv-3000] rule permit ip source 192.168.1.10 0 destination 192.168.1.1 0

[Device-acl-ipv4-adv-3000] rule permit ip source 192.168.1.11 0 destination 192.168.1.1 0

[Device-acl-ipv4-adv-3000] quit

# 定义流行为 behavior1，配置流量镜像到 CPU。

[Device] traffic behavior behavior1

[Device-behavior-behavior1] mirror-to cpu

[Device-behavior-behavior1] quit

# 定义类 classifier1，匹配 ACL3000。

[Device] traffic classifier classifier1

[Device-classifier-class1] if-match acl 3000

[Device-classifier-class1] quit

# 定义一个名为 user1 的策略，并在策略 user1 中为类 classifier1 指定采用流行为 behavior1。

```
[Device] qos policy user1
[Device-qospolicy-user1] classifier classifier1 behavior behavior1
[Device-qospolicy-user1] quit
# 将策略 user1 应用到接口 GigabitEthernet1/0/1 的入方向上。
[Device] interface gigabitethernet 1/0/1
[Device-GigabitEthernet1/0/1] qos apply policy user1 inbound
[Device-GigabitEthernet1/0/1] quit
[Device] quit
```

### (3) 开启报文捕获功能

# 开启 GigabitEthernet1/0/1 接口上的报文捕获功能，指定捕获报文个数上限为 10，指定捕获的报文存入文件 a.pcap。在 192.168.1.10 上使用 Telnet 方式登录 192.168.1.1，创造报文捕获条件。

```
<Device> packet-capture interface gigabitethernet 1/0/1 capture-filter "vlan 3 and src
192.168.1.10 or 192.168.1.11 and dst 192.168.1.1" limit-captured-frames 10 write
flash:/a.pcap
Capturing on 'GigabitEthernet1/0/1'
10
```

## 4. 验证配置

# 在设备上解析报文文件 flash:/a.pcap。

```
<Device> packet-capture read flash:/a.pcap
 1  0.000000 192.168.1.10 -> 192.168.1.1 TCP 62 6325 > telnet [SYN] Seq=0 Win=65535 Len=0
MSS=1460 SACK_PERM=1
 2  0.000061 192.168.1.10 -> 192.168.1.1 TCP 60 6325 > telnet [ACK] Seq=1 Ack=1 Win=65535
Len=0
 3  0.024370 192.168.1.10 -> 192.168.1.1 TELNET 60 Telnet Data ...
 4  0.024449 192.168.1.10 -> 192.168.1.1 TELNET 78 Telnet Data ...
 5  0.025766 192.168.1.10 -> 192.168.1.1 TELNET 65 Telnet Data ...
 6  0.035096 192.168.1.10 -> 192.168.1.1 TELNET 60 Telnet Data ...
 7  0.047317 192.168.1.10 -> 192.168.1.1 TCP 60 6325 > telnet [ACK] Seq=42 Ack=434 Win=65102
Len=0
 8  0.050994 192.168.1.10 -> 192.168.1.1 TCP 60 6325 > telnet [ACK] Seq=42 Ack=436 Win=65100
Len=0
 9  0.052401 192.168.1.10 -> 192.168.1.1 TCP 60 6325 > telnet [ACK] Seq=42 Ack=438 Win=65098
Len=0
10  0.057736 192.168.1.10 -> 192.168.1.1 TCP 60 6325 > telnet [ACK] Seq=42 Ack=440 Win=65096
Len=0
```

# 目 录

1 云平台连接.....	1-1
1.1 云平台连接简介.....	1-1
1.1.1 云平台的多连接机制 .....	1-1
1.1.2 云平台连接建立过程 .....	1-1
1.2 硬件适配关系.....	1-2
1.3 配置云平台连接.....	1-2
1.4 云平台连接显示和维护.....	1-3
1.5 云平台连接典型配置举例.....	1-3
1.5.1 云平台连接基础配置举例 .....	1-3

# 1 云平台连接

## 1.1 云平台连接简介

云平台连接是指设备与 H3C 云平台服务器(H3C Cloud server)通过 Internet 建立的远程管理通道。通过云平台连接，网络管理员可以在没有直接接入到设备所在网络的情况下，通过云平台服务器对分布在不同地域的设备进行管理和维护。

### 1.1.1 云平台的多连接机制

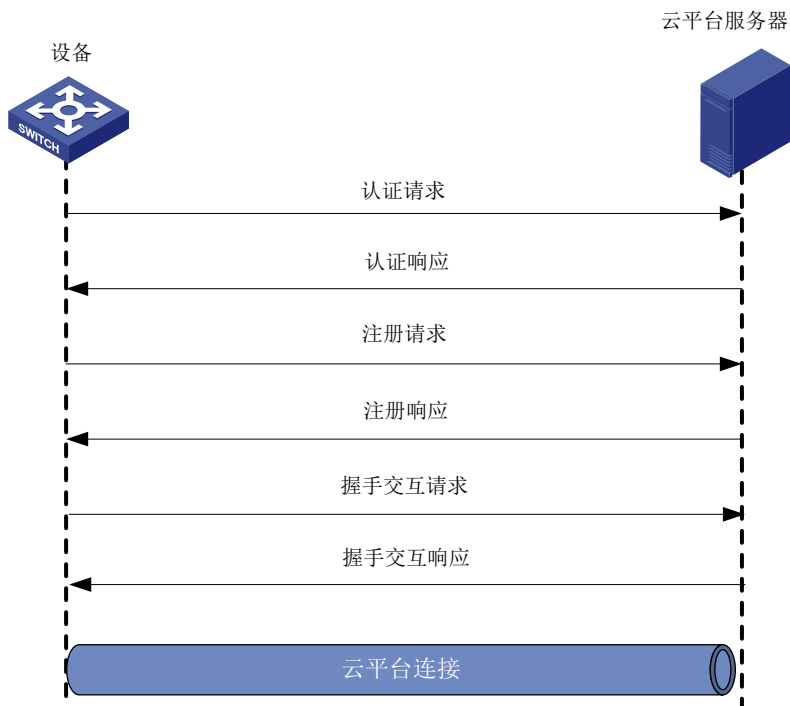
设备与云平台服务器建立主连接后，设备上不同的业务模块可以与云平台服务器上对应的微服务建立子连接，各个子连接之间互不影响。云平台的多连接机制可以为不同业务的数据提供不同的通信通道，避免多业务间的干扰。

### 1.1.2 云平台连接建立过程

如 [图 1-1](#) 所示，以设备与云平台服务器为例，建立云平台连接的过程如下：

- (1) 设备向云平台服务器发送认证请求报文。
- (2) 云平台服务器收到认证请求报文后，检查服务器上是否已添加认证请求报文中携带的设备序列号，如果已添加则回应认证成功响应报文，否则回应认证失败响应报文。
- (3) 设备收到认证成功响应报文后，向云平台服务器发送注册请求报文。
- (4) 云平台服务器向设备返回注册响应报文并携带云平台主连接的 URL。
- (5) 设备收到注册响应报文后，使用该 URL 向云平台服务器发送握手交互（从 HTTP 协议切换为 WebSocket 协议）请求。
- (6) 云平台服务器回复握手交互响应报文。
- (7) 通过以上报文交互，完成建立主连接。
- (8) 设备向云平台服务器发送子连接 URL 请求报文，云平台服务器将所有微服务及其对应 URL 发送给设备。
- (9) 业务模块向云平台连接管理模块注册后，云平台连接管理模块返回该业务模块的 URL 列表，业务模块与云平台服务器建立子链接。

图1-1 云平台连接建立过程



## 1.2 硬件适配关系

S5110V2-SI 和 S5000V3-EI 系列交换机不支持本功能。

S3100V3-SI 系列交换机仅 Release 6127 及以上版本支持本功能。

## 1.3 配置云平台连接

### 1. 功能简介

在设备上需要以域名的形式指定云平台服务器的地址，以便设备与指定服务器建立连接。

设备与云平台服务器建立连接后，会周期性地向服务器发送 **Keepalive** 报文进行保活。如果设备在连续 3 个 **Keepalive** 报文的发送周期内没有收到云平台服务器的响应，设备会重新向云平台服务器发送认证请求，与其重新建立连接。

### 2. 配置准备

配置云平台连接之前，需要在云平台服务器上添加待管理设备的序列号。在建立连接时，云平台服务器将根据设备的序列号，回复认证响应报文。

配置域名解析，以便将云平台服务器的域名解析为正确的 IP 地址。

### 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置云平台服务器域名。

```
cloud-management server domain domain-name
```

缺省情况下，未指定云平台服务器域名。

- (3) 配置向云平台服务器发送 **Keepalive** 报文的时间间隔。

**cloud-management keepalive interval**

缺省情况下，设备向云平台服务器发送 **Keepalive** 报文的时间间隔为 180 秒。

## 1.4 云平台连接显示和维护

在完成上述配置后，在任意视图下执行 **display** 命令，可以显示配置后云平台连接的运行情况，通过查看显示信息，来验证配置的效果。

在用户视图下，用户可以执行 **reset** 命令来重新建立设备与云平台的连接。

表1-1 云平台连接的显示和维护

操作	命令
显示云平台连接的状态	<b>display cloud-management state</b>
重新建立设备与云平台的连接	<b>reset cloud-management tunnel</b>

## 1.5 云平台连接典型配置举例

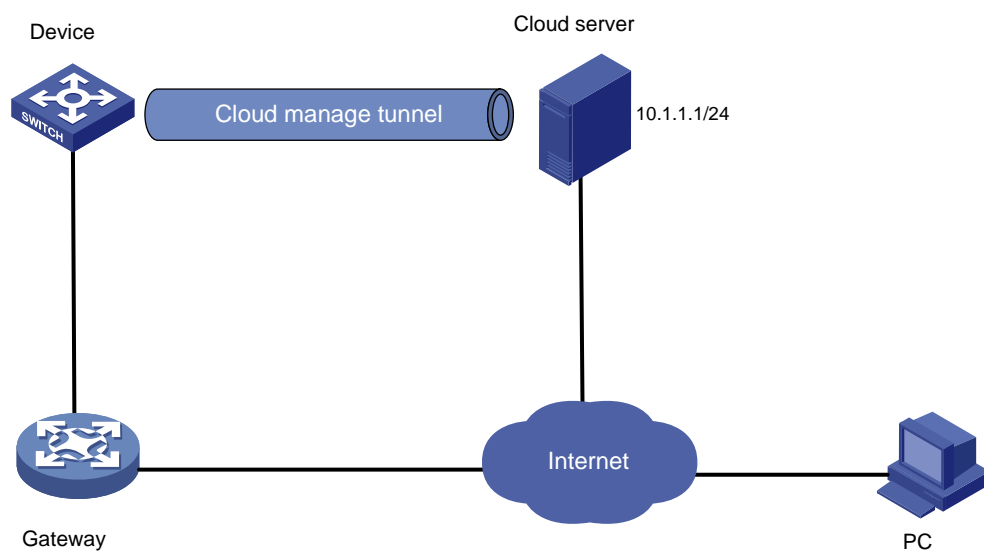
### 1.5.1 云平台连接基础配置举例

#### 1. 组网需求

设备与云平台服务器之间建立云平台连接，使网络管理员可以在远程 PC 端通过登入云平台服务器对设备进行管理。

#### 2. 组网图

图1-2 云平台连接配置组网图



### 3. 配置步骤

#### (1) 配置 IP 地址和路由

配置设备各接口 IP 地址，并配置路由协议确保各设备之间路由可达，具体配置过程略。

#### (2) 配置云平台服务器

在云平台服务器上添加待管理设备的序列号。云平台服务器的 IP 地址为 10.1.1.1/24，域名为 cloud.com。

#### (3) 配置设备

# 指定云平台服务器的域名为 cloud.com。

```
<Device> system-view
```

```
[Device] cloud-management server domain cloud.com
```

# 配置主机名 cloud.com 对应的 IP 地址为 10.1.1.1。

```
[Device] ip host cloud.com 10.1.1.1
```

### 4. 验证配置

# 查看云平台连接信息，可以看到设备与云平台服务器之间已经建立了云平台连接并进入 Established 状态。

```
[Device] display cloud-management state
```

```
Cloud connection state    : Established
```

```
Device state              : Request_success
```

```
Cloud server address      : 10.1.1.1
```

```
Cloud server domain name  : cloud.com
```

```
Cloud server port        : 443
```

```
Connected at              : Wed Jan 27 14:18:40 2016
```

```
Duration                  : 00d 00h 02m 01s
```

# 目 录

<b>1 SmartMC .....</b>	<b>1-1</b>
1.1 SmartMC简介 .....	1-1
1.1.1 SmartMC网络框架 .....	1-1
1.1.2 SmartMC网络的建立 .....	1-2
1.1.3 SmartMC支持的功能 .....	1-3
1.2 SmartMC配置限制和指导 .....	1-6
1.3 SmartMC配置任务简介 .....	1-6
1.4 SmartMC配置准备 .....	1-6
1.5 开启SmartMC功能 .....	1-7
1.6 配置FTP服务器信息 .....	1-8
1.7 配置SmartMC网络的出接口 .....	1-8
1.8 开启以太网链路全自动聚合功能 .....	1-9
1.9 修改成员设备缺省用户的密码 .....	1-9
1.10 配置SmartMC组 .....	1-9
1.11 为成员设备创建VLAN .....	1-10
1.12 为成员设备批量下发配置 .....	1-10
1.13 为AP和IP电话接入端口批量下发配置 .....	1-10
1.14 备份配置文件 .....	1-11
1.15 开启资源监控功能 .....	1-12
1.16 升级成员设备的启动软件和配置文件 .....	1-12
1.16.1 功能简介 .....	1-12
1.16.2 配置限制和指导 .....	1-12
1.16.3 配置准备 .....	1-12
1.16.4 升级成员设备的启动软件和配置文件 .....	1-12
1.16.5 通过SmartMC组升级成员设备的启动软件和配置文件 .....	1-13
1.17 网络拓扑管理 .....	1-14
1.17.1 刷新网络拓扑 .....	1-14
1.17.2 保存网络拓扑 .....	1-15
1.18 替换故障成员设备 .....	1-15
1.19 SmartMC显示和维护 .....	1-15
1.20 SmartMC典型配置举例 .....	1-16
1.20.1 SmartMC基本组网配置举例 .....	1-16



# 1 SmartMC



说明

- 仅 Release 6127 及以上版本支持 SmartMC 特性。
- S5110V2-SI、S5000V3-EI 和 S5000E-X 系列交换机不支持 SmartMC 特性。
- S5130S-SI[LI]、S5120V2-SI[LI]和 S3100V3-SI 系列交换机仅支持作为 SmartMC 网络中的成员设备。

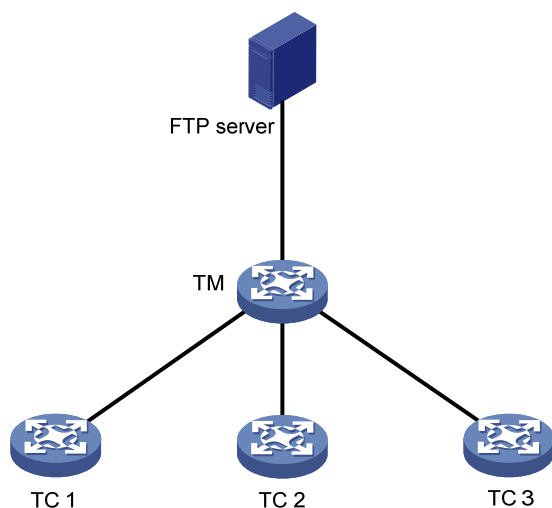
## 1.1 SmartMC简介

SmartMC（Smart Management Center，智能管理中心）功能用于集中管理和维护网络边缘大量分散的网络设备。SmartMC 网络中有且只有一台设备为管理设备，其他设备均为成员设备。通过在管理设备上简单的配置，即可实现对整个网络中的所有成员设备进行批量管理，例如对成员设备进行备份和下载配置文件、软件版本升级、批量下发配置和故障设备替换等功能。

### 1.1.1 SmartMC网络框架

SmartMC网络的基本框架如 [图 1-1](#) 所示。

图1-1 SmartMC 网络基本框架示意图



SmartMC 网络元素主要有：

- 管理设备：管理 SmartMC 网络中所有设备，也可称之为 TM（Topology master）。
- 成员设备：SmartMC 网络中被管理的设备，也可称之为 TC（Topology client）。
- FTP 服务器：用来保存设备的启动软件和配置文件等。

### 1.1.2 SmartMC网络的建立

SmartMC 网络可以自动建立，也可以手动建立。如果是自动建立，管理设备会使用 NETCONF 会话获取成员设备的信息（例如端口信息、LLDP 邻居信息、STP 信息、设备类型、软件版本等）以形成 SmartMC 网络拓扑；如果是手动建立，管理设备会使用 NETCONF 会话获取各成员设备的 LLDP 邻居信息、使用 SNMP get 请求获取成员设备的硬件信息以形成 SmartMC 网络拓扑。

#### 1. 自动建立SmartMC网络

自动建立 SmartMC 网络的过程如下：

- (1) 管理设备和成员设备开启 SmartMC 功能后，管理设备每隔 15 秒发送一次 SmartMC 广播报文（广播报文中携带自己的桥 MAC、Vlan-interface1 的 IP 地址等信息），询问网络中是否存在成员设备。
- (2) 成员设备收到广播报文后，记录管理设备的设备信息，并向管理设备发送 SmartMC 单播应答报文（应答报文中封装自己的桥 MAC、Vlan-interface1 的 IP 地址等信息）。
- (3) 管理设备收到成员设备的应答报文后，使用成员设备的缺省用户名（admin）和密码（admin）与成员设备建立 NETCONF 会话，并通过该会话获取成员设备的详细信息（例如成员设备的端口信息、LLDP 邻居信息、STP 信息、设备类型、软件版本等）。
- (4) 管理设备与成员设备建立保活连接，并将成员设备加入到 SmartMC 网络。
- (5) 管理设备通过使用 NETCONF 会话获取到的各成员设备的 LLDP 邻居信息形成 SmartMC 网络拓扑。

SmartMC 网络建立成功后，管理设备和成员设备通过 SmartMC 广播报文和应答报文感知对方的存在：

- 成员设备收到管理设备的 SmartMC 广播报文时，将会检查广播报文中的桥 MAC 与本地记录的桥 MAC 是否一致。如果一致，则向管理设备回应 SmartMC 应答报文。如果在限定时间（1～2 分钟）内没有收到管理设备的广播报文，则认为管理设备不存在，成员设备将清除该管理设备的信息。
- 管理设备收到成员设备的应答报文时，将会检查应答报文中的桥 MAC 与本地记录的桥 MAC 是否一致。如果一致，则说明该成员设备存在。如果 150 秒内没有收到成员设备的应答报文，则认为该成员设备离线，管理设备将成员设备置为离线状态。

#### 2. 手动建立SmartMC网络

通过手动建立 SmartMC 网络时，管理设备与成员设备不进行 SmartMC 报文交互，用户登录管理设备的 Web 页面，在可视化页面中点击“添加设备”，然后在添加设备对话框中输入成员设备的 IP 地址、用户名和密码，手动添加成员设备。有关手动建立 SmartMC 网络的详细介绍，请参见《Web 用户手册》。

管理设备完成如下操作后，将成员设备加入到 SmartMC 网络中：

- 验证可以通过 Telnet 登录到成员设备。
- 使用 NETCONF 获取成员设备的基本信息（如 LLDP 邻居信息）。
- 使用 SNMP get 请求获取成员设备的硬件信息。

### 1.1.3 SmartMC支持的功能

#### 1. 为成员设备批量下发配置

使用本功能，用户可以一次将多条配置批量下发给成员设备，不需要登录到成员设备逐条配置，从而可以简化配置过程，节省配置时间。功能处理流程是：

- (1) 用户在管理设备上创建命令行批处理文件，并编辑成员设备需要批量执行的命令行。
- (2) 管理设备作为 Telnet 客户端与成员设备建立 Telnet 连接，将批量命令下发给成员设备。
- (3) 成员设备批量执行管理设备下发的命令行。

#### 2. 为AP和IP电话接入端口批量下发配置

用户在管理设备上配置 AP 和 IP 电话接入 SmartMC 网络时，端口使用的命令行批处理文件。

管理设备将接入 SmartMC 网络的设备分为三类：AP、IP 电话以及其它类型的设备。

配置完成后，当管理设备通过 LLDP 感知到有 AP 或者 IP 电话接入 SmartMC 网络时：

- 若存在对应设备类型的预设模板，会先将端口下的配置恢复到缺省情况，再根据设备类型，自动向端口下发指定的命令行批处理文件中的配置。
- 若不存在对应设备类型的预设模板，则保持原有配置。

当 AP 或者 IP 电话和 SmartMC 网络断开连接后，对应端口的配置不会改变，当有设备再次通过该端口接入 SmartMC 网络时：

- 如果接入的是 AP 或者 IP 电话，但是和上次接入的设备类型不同，若存在对应设备类型的预设模板，管理设备会先将端口下的配置恢复到缺省情况，再向此端口下发配置；若不存在对应设备类型的预设模板，保持原有配置。
- 如果接入的不是 AP 和 IP 电话，则保持原有配置。

#### 3. 备份配置文件

使用本功能，用户可以在管理设备上指定一台或多台成员设备备份配置文件，不用再登录到成员设备上相应配置。功能处理流程是：

- (1) 管理设备通过 SmartMC 单播报文通知成员设备备份配置文件。
- (2) 成员设备执行保存配置文件操作，将当前运行配置保存到下次主用配置文件中，并将该配置文件备份到 FTP 服务器上。

备份配置文件分为自动备份配置文件和手动备份配置文件：

- 自动备份配置文件：配置该功能后，SmartMC 网络中的所有设备会立即备份一次自己的配置文件，以后则按照指定的时间间隔进行配置文件备份操作。
- 手动备份配置文件：用户可以手动指定成员设备或 SmartMC 组进行配置文件备份操作。

#### 4. 升级成员设备的启动软件和配置文件

使用本功能，用户可以在管理设备上指定一台或多台成员设备升级启动软件或配置文件，不用再登录到成员设备上相应配置。功能处理流程是：

- (1) 用户将成员设备的启动软件和配置文件保存在 FTP 服务器上，在管理设备上指定成员设备或 SmartMC 组需要下载的升级文件，再执行升级操作。
- (2) 成员设备收到管理设备的下载指令后，自动到 FTP 服务器下载待升级的文件。
- (3) 成员设备对启动软件和配置文件进行升级。

- 成员设备使用 ISSU 方式在后台自动完成软件升级, 升级过程中根据升级文件的兼容情况, 可能会重启设备。
- 成员设备使用待升级配置文件中的配置替换当前运行配置, 替换过程中不会重启设备。

## 5. 替换故障成员设备

使用本功能, 当 SmartMC 网络中的成员设备出现故障时, 用户可以用新设备替换故障成员设备。功能处理流程是:

- (1) 用户将与故障成员设备型号完全相同的新成员设备安装到原故障成员设备的位置。
- (2) 管理设备通过 SmartMC 单播报文通知新成员设备进行替换故障成员设备操作。
- (3) 新成员设备从 FTP 服务器下载原故障成员设备的配置文件, 并执行该文件中的配置, 完成故障成员设备替换。

替换故障成员设备分为自动替换和手动替换:

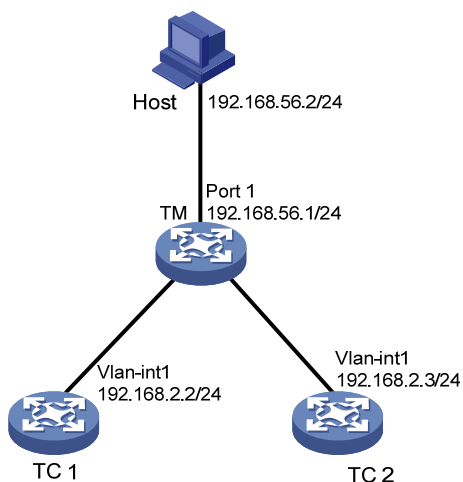
- 自动替换: 配置该功能后, 管理设备会记录所有成员设备的位置信息。用户将新成员设备安装到原来故障成员设备的位置, 管理设备发现新成员设备后, 先判断新成员设备与故障成员设备的型号和拓扑位置是否一致, 如果一致, 则进行故障替换操作, 并通知新成员设备到 FTP 服务器下载原故障成员设备的配置文件。新成员设备下载配置文件并运行配置文件中的配置。
- 手动替换: 用户在命令行中手动输入新成员设备的 ID 和故障成员设备的 ID 进行替换操作。此时, 新成员设备到 FTP 服务器下载原故障成员设备的配置文件并运行配置文件中的配置。

## 6. 配置SmartMC网络的出接口

SmartMC 网络的出接口位于管理设备上, 是 SmartMC 网络与外部网络互通使用的接口。可以配置多个 SmartMC 网络的出接口, 以满足 SmartMC 网络和外部网络互通的需求。

如 [图 1-2](#) 所示, Host 通过三层以太网接口 Port 1 连接到管理设备, 所在网段为 192.168.56.0/24, SmartMC 网络在 VLAN 1 内建立, 所在网段为 192.168.2.0/24。此时 Host 可以访问管理设备的 Web 管理页面, 而无法访问两个成员设备的 Web 管理页面。如果将 Port 1 设置为 SmartMC 网络的出接口, 用户通过 Port 1 访问管理设备的 SmartMC 管理页面之后, 进入 “可视化 > 拓扑” 页面, 从拓扑图中选中一台成员设备, 然后点击 “登录 Web 页面”, 便可以访问成员设备的 Web 管理页面。此时, 管理设备是将成员设备的地址映射为 “SmartMC 网络的出接口 IP 地址: 端口号” 的形式, 让用户使用新地址访问成员设备的 Web 管理页面, 例如将 TC 1 的地址映射为 “192.168.56.1:5002”。

图1-2 配置 SmartMC 网络的出接口组网图



## 7. 以太网链路全自动聚合功能

在管理设备上配置本功能后，管理设备会将该配置下发给所有的成员设备：

- 两台成员设备间存在互连链路时，成员设备会自动在本设备上创建聚合接口，并将物理接口加入该聚合接口，而不需要用户手动配置。这样，当两台成员设备之间存在多条互连链路时，可以增加链路带宽，同时链路之间相互动态备份，能够有效的提高链路的可靠性。
- 管理设备和成员设备之间不会自动进行链路聚合，用户可以根据需求进行手工配置。有关配置以太网链路手工聚合的详细介绍，请参见“二层技术-以太网交换配置指导”中的“以太网链路聚合”。



### 说明

新成员设备的全自动聚合功能的开关状态和其加入的 SmartMC 网络的全自动聚合功能的开关状态保持一致。

## 8. 为成员设备创建VLAN

使用本功能，用户在为成员设备创建 VLAN 时，系统会自动将成员设备中所有未连接管理设备和其他成员设备的 Access 类型的端口加入到 VLAN 中，用户不再需要手动将端口加入到 VLAN 中。

对于连接离线设备的 Access 类型端口，需要手动清除离线设备后再执行本操作。

如果成员设备创建 VLAN 成功，但是没有成功的向 VLAN 中添加所有满足条件的端口，则所有满足条件的端口的配置将恢复到创建 VLAN 前的状态。

一台成员设备创建 VLAN 失败不会影响其他成员设备的 VLAN 创建。

## 9. 资源监控

在管理设备上开启对管理设备或成员设备的资源监控后，可以在管理设备上查看各设备的资源监控信息，如 CPU 利用率、内存利用率以及温度监控信息。

## 1.2 SmartMC配置限制和指导

成员设备上只需开启 SmartMC 功能，其他 SmartMC 相关功能只能在管理设备上配置。

对于备份配置文件、替换故障成员设备、升级成员设备的启动软件配置文件以及以太网链路全自动聚合等功能，只对自动加入 SmartMC 网络的成员设备生效，对手动添加的成员设备不生效。

设备在 VLAN1 内建立 SmartMC 网络，为保证 SmartMC 功能正常运行，请不要对 VLAN1 进行安全性相关配置。

## 1.3 SmartMC配置任务简介

SmartMC 配置任务如下：

(1) [开启SmartMC功能](#)

(2) [配置FTP服务器信息](#)

仅在备份配置文件、升级成员设备的启动软件和配置文件、替换故障成员设备时必选。

(3) （可选）[配置SmartMC网络的出接口](#)

(4) （可选）[开启以太网链路全自动聚合功能](#)

(5) （可选）[修改成员设备缺省用户的密码](#)

(6) [配置SmartMC组](#)

通过 SmartMC 组升级成员设备的启动软件、配置文件和通过 SmartMC 组批量下发配置时为必选。

(7) （可选）配置下发和管理

○ [为成员设备创建VLAN](#)

○ [为成员设备批量下发配置](#)

○ [为AP和IP电话接入端口批量下发配置](#)

○ [备份配置文件](#)

(8) （可选）监控和维护

○ [开启资源监控功能](#)

○ [升级成员设备的启动软件和配置文件](#)

○ [网络拓扑管理](#)

○ [替换故障成员设备](#)

## 1.4 SmartMC配置准备

配置 SmartMC 功能前，用户需要在管理设备和成员设备上配置如下功能：

- 开启 Telnet 服务，配置 VTY 用户线的认证方式为 **scheme**。关于 Telnet 及 VTY 用户线的详细介绍，请参见“基础配置指导”中的“登录设备”。
- 配置本地用户及密码，并配置该用户的服务类型为 Telnet、HTTP 和 HTTPS，RBAC 角色为 network-admin。关于本地用户的详细介绍，请参见“安全配置指导”中的“AAA”。关于用户角色的详细介绍，具体请参见“基础配置指导”中的“RBAC”。



- 对于管理设备，所设置的本地用户的用户名和密码要和 **smartmc tm username username password { cipher | simple } string enable** 命令指定用户名、密码相同，管理设备使用此本地用户进行内部通信。
- 对于成员设备，请设置用户名为 **admin**、密码为 **admin** 的本地用户。因为管理设备使用缺省用户名 **admin**、密码 **admin** 和其建立 NETCONF 会话，以将其加入到 SmartMC 网络中。SmartMC 网络构建成功后，可以使用 **smartmc tc password** 命令修改成员设备缺省用户 **admin** 的密码。
- 开启基于 HTTP 的 NETCONF over SOAP 功能。关于 NETCONF over SOAP 功能的详细介绍，具体请参见“网络管理与监控”中的“NETCONF”。
- 全局开启 LLDP 功能。关于 LLDP 功能的详细介绍，具体请参见“二层技术-以太网交换”中的“LLDP”。
- 如果需要使用 Web 管理管理设备和成员设备，则需要开启 HTTP 或 HTTPS 服务，并配置本地用户的服务类型为 HTTP 或 HTTPS。关于 Web 登录、HTTP 和 HTTPS 的详细介绍，请参见“基础配置指导”中的“登录设备”。
- 如果需要使用手动建立 SmartMC 网络，则需要在成员设备上配置 **snmp-agent community read public** 和 **snmp-agent sys-info version v2c** 命令。关于 SNMP 的详细介绍，请参见“网络管理与监控”中的“SNMP”。

## 1.5 开启SmartMC功能

### 1. 功能简介

用户分别登录到管理设备和成员设备上，开启 SmartMC 功能。当 SmartMC 网络自动构建成功后，便可以在管理设备上配置相关功能，实现对成员设备的配置、管理。

### 2. 配置限制和指导

一个 SmartMC 网络有且仅有一台管理设备。

将管理设备切换为成员设备或关闭 SmartMC 功能时，会清空当前运行配置中与 SmartMC 功能相关的配置。

SmartMC 功能会占用一定的 ACL 资源，如果 ACL 资源不足，会导致 SmartMC 功能开启失败。请使用 **display acl** 命令查看 ACL 的配置和运行情况，并根据实际情况使用 **undo acl** 命令删除不需要的 ACL。释放 ACL 资源后，再开启 SmartMC 功能。有关 ACL 的详细介绍，请参见“ACL 和 QoS 配置指导”中的“ACL”。

开启 SmartMC 功能时，设备会检查 80 端口、443 端口是否被占用，因为 HTTP 和 HTTPS 服务需要占用这两个端口号，如果 80 端口或者 443 端口被占用，则 SmartMC 功能开启失败。

### 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启 SmartMC 功能并配置设备的角色。

```
smartmc { tc | tm username username password { cipher | simple } string }
enable
```

缺省情况下，SmartMC 功能处于关闭状态。

## 1.6 配置FTP服务器信息

### 1. 功能简介

FTP 服务器用于保存成员设备升级使用的启动软件和配置文件，以及管理设备和成员设备正常运行过程中备份的配置文件。

### 2. 配置限制和指导

用户可以配备专用的 FTP 服务器，也可以配置管理设备作为 FTP 服务器：

- 如果使用管理设备作为 FTP 服务器，请保证管理设备有足够的存储资源保存成员设备所需要的文件。关于 FTP 的详细介绍，请参见“基础配置指导”中的“FTP 和 TFTP”。
- 如果使用专用的FTP服务器，建议将FTP服务器和管理设备相连，FTP服务器将使用VLAN 1 和管理设备、成员设备通信。如果将FTP服务器和成员设备相连，为成员设备创建VLAN后，会将成员设备和FTP服务器相连的接口划入新创建的VLAN，从而导致FTP服务器和SmartMC 网络断开连接。关于为成员设备创建VLAN的详细介绍，请参见“[为成员设备创建VLAN](#)”。

### 3. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置 FTP 服务器信息。

```
smartmc ftp-server server-address username username password { cipher  
| simple } string
```

缺省情况下，未配置 FTP 服务器信息。

## 1.7 配置SmartMC网络的出接口

### 1. 配置限制和指导

SmartMC 网络在 VLAN1 内建立，不能将 Vlan-interface1 配置为 SmartMC 网络的出接口。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 VLAN 接口视图或三层以太网视图。

- 进入 VLAN 接口视图。

```
interface vlan interface-number
```

- 进入三层以太网接口视图。

```
interface interface-type interface-number
```

- (3) 将接口配置为 SmartMC 网络的出接口。

```
smartmc outbound
```

缺省情况下，未将接口配置为 SmartMC 网络的出接口，SmartMC 网络无法和外部网络互通。



## 1.8 开启以太网链路全自动聚合功能

### 1. 配置限制和指导

开启或关闭以太网链路全自动聚合功能后，会导致网络震荡，成员设备会短时间离线。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 开启以太网链路全自动聚合功能。

```
smartmc auto-link-aggregation enable
```

缺省情况下，以太网链路全自动聚合功能处于关闭状态。

## 1.9 修改成员设备缺省用户的密码

### 1. 功能简介

对于自动加入 SmartMC 网络的成员设备，管理设备会使用缺省用户名 **admin**、密码 **admin** 与其建立 NETCONF 会话，并将其加入到 SmartMC 网络中。SmartMC 网络组建完成后，建议用户修改缺省用户 **admin** 的密码，提高 SmartMC 网络的安全性。

### 2. 配置限制和指导

该功能只能修改自动加入 SmartMC 网络的成员设备。

用户不能修改手动添加到 SmartMC 网络中的成员设备的密码，否则将导致管理设备无法对成员设备进行管理。

用户可以执行 **display smartmc tc verbose** 命令查看成员设备添加到 SmartMC 网络的方式。

### 3. 修改成员设备缺省用户的密码

- (1) 进入系统视图。

```
system-view
```

- (2) 修改成员设备缺省用户的密码。

```
smartmc tc password string
```

## 1.10 配置SmartMC组

### 1. 功能简介

如果在管理设备上创建了 SmartMC 组，并将成员设备加入 SmartMC 组，对设备进行配置或升级时，指定对应的 SmartMC 组即可完成对组内所有成员设备的操作。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 创建 SmartMC 组，并进入 SmartMC 组视图。

```
smartmc group group-name
```

- (3) （可选）查看预定义的设备类型。

**match device-type ?**

如果预定义的设备类型中无用户所需的设备类型，则用户需要手动添加成员设备类型。

- (4) 配置成员设备的匹配规则。

```
match { device-type device-type | ip-address ip-address  
{ ip-mask-length | ip-mask } | mac-address mac-address mac-mask-length }
```

缺省情况下，未配置成员设备的匹配规则。

- (5) 退回系统视图。

```
quit
```

- (6) （可选）手动添加成员设备的设备类型。

```
smartmc tc sysoid sysoid device-type device-type
```

手动添加的设备类型不能与预定义的设备类型相同。

用户可以执行 **display smartmc tc verbose** 命令获取成员设备的 SYSOID。

## 1.11 为成员设备创建VLAN

### 1. 配置限制和指导

如果多次为同一成员设备创建 VLAN，则最新配置生效。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 为成员设备创建 VLAN，并将 Access 类型端口加入 VLAN。

```
smartmc vlan vlan-id { group group-name-list | tc tc-id-list }
```

## 1.12 为成员设备批量下发配置

- (1) 请在用户视图下执行本命令，创建命令行批处理文件，并编辑成员设备需要批量执行的命令行。

```
create batch-file cmd-filename
```

编辑命令时，每条命令占一行，编辑完成后输入“%”和回车结束编辑，并退回用户视图。

设备不检查命令行的正确性，因此，在编辑命令行时，请保证其正确性。

- (2) 进入系统视图。

```
system-view
```

- (3) 向成员设备或 SmartMC 组批量下发配置。

```
smartmc batch-file cmd-filename deploy { group group-name-list | tc  
tc-id-list }
```

## 1.13 为AP和IP电话接入端口批量下发配置

### 1. 配置限制和指导

端口批量配置文件中的配置必须全部为端口视图下的配置，否则可能导致配置错误。

命令行批处理文件的内容不能超过 8190 字符。

配置命令行批处理文件时，设备不检查命令行批处理文件名称的正确性，因此，配置本功能时，请保证其正确性。配置完成后，请不要删除或重命名此文件。

## 2. 配置步骤

- (1) （可选）请在用户视图下执行本命令，创建命令行批处理文件，并编辑 AP 和 IP 需要批量执行的命令行。

```
create batch-file cmd-filename
```

编辑命令时，每条命令占一行，编辑完成后输入“%”和回车结束编辑，并退回用户视图。

设备不检查命令行的正确性，因此，在编辑命令行时，请保证其正确性。

- (2) 进入系统视图。

```
system-view
```

- (3) 配置成员设备连接 AP 或 IP 电话时，端口使用的命令行批处理文件。

```
smartmc batch-file batch-file-name apply { ap | phone }
```

## 1.14 备份配置文件

### 1. 功能简介

成员设备备份配置文件时，会生成文件名为“设备桥 MAC 地址\_backup.cfg”的配置文件，上传至 FTP 服务器。

### 2. 配置限制和指导

将管理设备切换为成员设备后，请手工删除 FTP 服务器中原管理设备备份的配置文件。否则，切换后的成员设备可能下载原管理设备的配置文件并运行，导致网络冲突。

最多同时备份配置文件的成员设备的数量和 FTP 服务器性能相关。如果发现成员设备备份配置文件失败，请将最多同时备份配置文件的成员设备的数量设置为较小值。

### 3. 配置准备

配置本功能前，需先配置 FTP 服务器信息。对 FTP 服务器的配置可参见“[1.6 配置 FTP 服务器信息](#)”。

### 4. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 配置最多同时备份配置文件的成员设备的数量。

```
smartmc backup configuration max-number max-number
```

缺省情况下，同时备份配置文件的成员设备的最大个数为 5。

- (3) 配置备份配置文件功能。请选择一项进行配置。

- 开启自动备份配置文件功能，并配置自动备份配置文件的时间间隔。

```
smartmc backup configuration interval interval
```

缺省情况下，自动备份配置文件功能处于关闭状态。

- 手动备份设备的配置文件。

```
smartmc backup configuration { group group-name-list | tc  
[ tc-id-list ] }
```

## 1.15 开启资源监控功能

### 1. 配置步骤

- (1) 进入系统视图。

**system-view**

- (2) 配置管理设备获取成员设备资源监控数据的时间间隔。

**smartmc resource-monitor interval interval**

缺省情况下，管理设备获取成员设备资源监控数据的时间间隔为 1 分钟。

- (3) 配置资源监控数据的老化时间。

**smartmc resource-monitor max-age max-age**

缺省情况下，资源监控数据的老化时间为 24 小时。

- (4) 开启资源监控功能。

**smartmc resource-monitor [ cpu | memory | temperature ] \* [ group  
group-name-list | tc { tc-id-list | mac-address mac-address } | tm ]**

缺省情况下，资源监控功能处于关闭状态。

如果不指定资源类型，则开启成员设备上所有资源的监控功能。

如果不指定设备的角色，则开启管理设备和所有成员设备的资源监控功能。

## 1.16 升级成员设备的启动软件和配置文件

### 1.16.1 功能简介

使用本功能，用户可以选择立即升级、延时升级以及定时升级成员设备的启动软件和配置文件。

如果不指定延迟时间或具体升级时间点，则立即对设备进行升级。

### 1.16.2 配置限制和指导

设备同时仅能够进行一个升级操作，请完成一次升级操作后，再进行下一次升级操作。

如果选择了立即升级，则无法取消升级；如果选择了延时升级或者定时升级，在设备开始升级前，还可以通过 **undo smartmc upgrade** 命令取消升级。

### 1.16.3 配置准备

配置本功能前，需先配置FTP服务器信息。对FTP服务器的配置可参见“[1.6 配置FTP服务器信息](#)”。

### 1.16.4 升级成员设备的启动软件和配置文件

#### 1. 一步式升级成员设备的启动软件和配置文件

- (1) 进入系统视图。

**system-view**

- (2) 升级成员设备的启动软件

```
smartmc upgrade boot-loader tc { tc-id-list { boot boot-filename system
system-filename | file ipe-filename } }<1-40> [ delay delay-time | time
in-time ]
```

- (3) 升级成员设备的配置文件

```
smartmc upgrade startup-configuration tc { tc-id-list
cfg-filename }<1-40> [ delay delay-time | time in-time ]
```

## 2. 分步式升级成员设备的启动软件和配置文件

- (1) 进入系统视图。

```
system-view
```

- (2) 升级成员设备的启动软件。

- o 分别配置各成员设备升级使用的启动软件。

```
smartmc tc tc-id boot-loader { ipe-filename | boot boot-filename
system system-filename }
```

- o 升级成员设备的启动软件。

```
smartmc upgrade boot-loader tc tc-id-list
```

- (3) 升级成员设备的配置文件。

- o 分别配置各成员设备升级使用的配置文件。

```
smartmc tc tc-id startup-configuration cfg-filename
```

- o 升级成员设备的配置文件

```
smartmc upgrade startup-configuration tc tc-id-list
```

## 1.16.5 通过SmartMC组升级成员设备的启动软件和配置文件

### 1. 一步式升级SmartMC组的启动软件和配置文件

- (1) 进入系统视图。

```
system-view
```

- (2) 升级 SmartMC 组的启动软件。

```
smartmc upgrade boot-loader group { group-name-list [ boot
boot-filename system system-filename | file ipe-filename ] }<1-40>
[ delay minutes | time in-time ]
```

- (3) 升级 SmartMC 组的配置文件。

```
smartmc upgrade startup-configuration group { group-name-list
cfg-filename }<1-40> [ delay minutes | time in-time ]
```

### 2. 分步式升级SmartMC组的启动软件和配置文件

- (1) 进入系统视图。

```
system-view
```

- (2) 进入 SmartMC 组视图。

```
smartmc group group-name
```

- (3) 配置组内成员设备使用的启动软件。

```
boot-loader file { ipe-filename | boot boot-filename system
system-filename }
```

缺省情况下，未配置组内成员设备使用的启动软件。

- (4) 配置组内成员设备使用的配置文件。

```
startup-configuration cfgfile
```

缺省情况下，未配置组内成员设备使用的配置文件。

- (5) 退回系统视图。

```
quit
```

- (6) 执行成员设备升级操作。请至少选择其中一项进行配置。

- 升级 SmartMC 组的启动软件。

```
smartmc upgrade boot-loader group group-name-list [ delay minutes |
time in-time ]
```

- 升级 SmartMC 组的配置文件。

```
smartmc upgrade startup-configuration group group-name-list [ delay
minutes | time in-time ]
```

## 1.17 网络拓扑管理

### 1.17.1 刷新网络拓扑

#### 1. 功能简介

设备支持自动和手动两种拓扑刷新方式：

- 自动刷新拓扑：配置拓扑刷新时间间隔后，管理设备按照该时间间隔自动刷新网络拓扑。
- 手动刷新拓扑：用户可以通过手动执行拓扑刷新命令，手动刷新当前网络的拓扑。

#### 2. 配置限制和指导

网络内成员设备越多，拓扑刷新所用时间越长。

#### 3. 配置步骤

请选择其中一项进行配置。

- 请在任意视图下执行本命令，手动刷新拓扑。

```
smartmc topology-refresh
```

- 自动刷新网络拓扑。

- 进入系统视图。

```
system-view
```

- 配置自动刷新网络拓扑的时间间隔。

```
smartmc topology-refresh interval interval
```

缺省情况下，拓扑刷新时间间隔为 60 秒。

## 1.17.2 保存网络拓扑

### 1. 功能简介

配置该功能后，当前网络拓扑将保存到 Flash 中的 topology.db 拓扑文件中。当管理设备重启后，管理设备和成员设备根据该拓扑文件恢复原来的网络拓扑。

### 2. 配置步骤

- (1) 进入系统视图。

```
system-view
```

- (2) 保存网络拓扑。

```
smartmc topology-save
```

## 1.18 替换故障成员设备

### 1. 配置限制和指导

新加入的成员设备的邻居关系、产品型号、IRF 编号必须与故障成员设备一致。

手动替换故障成员设备时，新加入的成员设备的编号不能和 SmartMC 网络中既有的成员设备（包括离线成员设备、在线成员设备）的编号一样，否则，替换失败。

### 2. 配置准备

配置本功能前，需先配置 FTP 服务器信息。对 FTP 服务器的配置可参见“[1.6 配置 FTP 服务器信息](#)”。

配置替换故障成员设备功能前，请先将新成员设备安装到原故障成员设备的位置，并连接好线缆。

### 3. 配置替换故障成员设备功能

- (1) 进入系统视图。

```
system-view
```

- (2) 替换故障成员设备。请选择其中一项进行配置。

- 开启自动替换故障成员设备功能。

```
smartmc auto-replace enable
```

缺省情况下，自动替换故障成员设备功能处于关闭状态。

- 配置手动替换故障成员设备功能

```
smartmc replace tc tc-id1 faulty-tc tc-id2
```

## 1.19 SmartMC 显示和维护

在完成上述配置后，在管理设备的任意视图下执行 **display** 命令可以显示配置后 SmartMC 的运行情况，通过查看显示信息验证配置的效果。

表1-1 SmartMC 显示和维护

操作	命令
显示成员设备备份配置文件的状态	<b>display smartmc backup configuration status</b>
显示执行命令行批处理文件的结果	<b>display smartmc batch-file status [ ap   last number / phone ]</b>

操作	命令
显示SmartMC功能的配置信息	<code>display smartmc configuration</code>
显示SmartMC网络中设备间的连接信息	<code>display smartmc device-link</code>
显示SmartMC组的信息	<code>display smartmc group [ group-name ] [ verbose ]</code>
显示新成员设备替换故障成员设备的替换状态	<code>display smartmc replace status</code>
显示设备的资源监控信息	<code>display smartmc resource-monitor [ cpu   memory   temperature ] * [ tc tc-id   tm ]</code>
显示资源监控功能的配置信息	<code>display smartmc resource-monitor configuration</code>
显示成员设备的信息	<code>display smartmc tc [ tc-id ] [ verbose ]</code>
显示成员设备的日志缓冲区中的日志信息	<code>display smartmc tc tc-id log buffer [ module module-name [ mnemonic mnemonic-value ] ]</code>
显示成员设备重启的日志信息	<code>display smartmc tc tc-id log restart</code>
显示成员设备创建VLAN的结果	<code>display smartmc vlan</code>
显示成员设备的升级状态	<code>display smartmc upgrade status</code>

## 1.20 SmartMC典型配置举例

### 1.20.1 SmartMC基本组网配置举例

#### 1. 组网需求

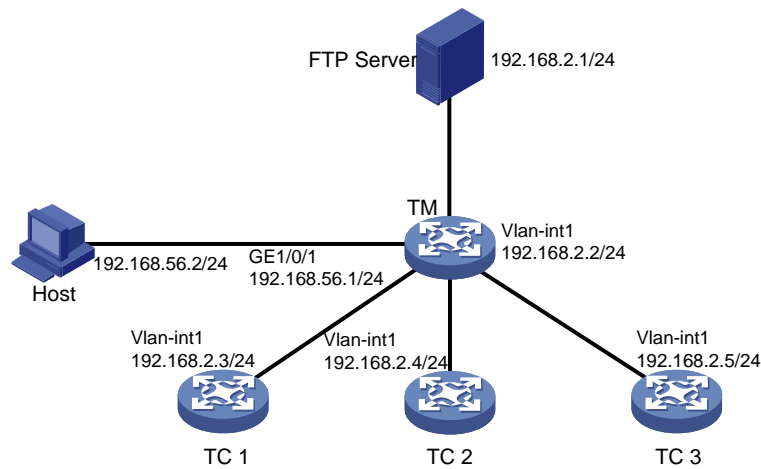
SmartMC网络的物理连接如 [图 1-3](#) 所示，TM为管理设备，TC 1～TC 3 为成员设备。通过自动方式建立SmartMC网络，在管理设备上配置接口GigabitEthernet1/0/1 为SmartMC网络的出接口，并通过SmartMC组升级所有成员设备的配置文件。

- 所有成员设备的设备类型都相同，均为 S5130S-LI 系列设备。
- Host 通过 GigabitEthernet1/0/1 接口连接到管理设备，Host 和 GigabitEthernet1/0/1 接口的 IP 地址在网段 192.168.56.0/24 内。管理设备及各成员设备的 Vlan-interface 1 接口的 IP 地址在网段 192.168.2.0/24 内。
- FTP 服务器的 IP 地址为 192.168.2.1，用户名为 admin，密码为 admin。
- 配置文件名称为 startup.cfg，存放在 FTP 服务器上。



## 2. 组网图

图1-3 SmartMC 配置举例组网图



## 3. 配置步骤

### (1) 配置 TC 1。

# 配置 VLAN1 接口。

```
<TC1> system-view
[TC1] interface vlan-interface 1
[TC1-Vlan-interface1] ip address 192.168.2.3 24
[TC1-Vlan-interface1] quit
```

# 开启 HTTP、HTTPS 服务。

```
[TC1] ip http enable
[TC1] ip https enable
```

# 开启 Telnet 服务。

```
[TC1] telnet server enable
```

# 开启基于 HTTP 的 NETCONF over SOAP 功能。

```
[TC1] netconf soap http enable
```

# 全局开启 LLDP 功能。

```
[TC1] lldp global enable
```

# 配置本地用户 admin，密码为 admin、服务类型为 Telnet、HTTP 和 HTTPS、RBAC 角色为 network-admin。

```
[TC1] local-user admin
[TC1-luser-manage-admin] password simple admin
[TC1-luser-manage-admin] service-type telnet http https
[TC1-luser-manage-admin] authorization-attribute user-role network-admin
[TC1-luser-manage-admin] quit
```

# 配置 VTY 用户线 0~63 的认证方式为 scheme。

```
[TC1] line vty 0 63
[TC1-line-vty0-63] authentication-mode scheme
[TC1-line-vty0-63] quit
```

# 开启 SmartMC 功能，并配置设备角色为成员设备。

```

[TC1] smartmc tc enable
(2) 配置 TC 2 和 TC 3。请参照配置 TC 1 的方法配置 TC 2 和 TC 3，此处不再赘述。
(3) 配置 TM。
# 配置 GigabitEthernet1/0/1 接口。
<TM> system-view
[TM] interface gigabitethernet 1/0/1
[TM-GigabitEthernet1/0/1] port link-mode route
[TM-GigabitEthernet1/0/1] ip address 192.168.52.2 24
[TM-GigabitEthernet1/0/1] quit
# 配置 VLAN1 接口。
[TM] interface vlan-interface 1
[TM-Vlan-interface1] ip address 192.168.2.2 24
[TM-Vlan-interface1] quit
# 开启 HTTP、HTTPS 服务。
[TM] ip http enable
[TM] ip https enable
# 开启 Telnet 服务。
[TM] telnet server enable
# 开启基于 HTTP 的 NETCONF over SOAP 功能。
[TM] netconf soap http enable
# 全局开启 LLDP 功能。
[TM] lldp global enable
# 配置本地用户 admin，密码为 admin、服务类型为 Telnet、HTTP 和 HTTPS、RBAC 角色
为 network-admin。
[TM] local-user admin
[TM-luser-manage-admin] password simple admin
[TM-luser-manage-admin] service-type telnet http https
[TM-luser-manage-admin] authorization-attribute user-role network-admin
[TM-luser-manage-admin] quit
# 配置 VTY 用户线 0~63 的认证方式为 scheme。
[TM] line vty 0 63
[TM-line-vty0-63] authentication-mode scheme
[TM-line-vty0-63] quit
# 开启 SmartMC 功能并配置设备的角色为管理设备，用户名为 admin，明文密码为 admin。
[TM] smartmc tm username admin password simple admin enable
# 将接口 GigabitEthernet1/0/1 配置为 SmartMC 网络的出接口。
[TM] interface gigabitethernet 1/0/1
[TM-GigabitEthernet1/0/1] smartmc outbound
[TM-GigabitEthernet1/0/1] quit
# 配置 FTP 服务器信息，指定 FTP 服务器的 IP 为 192.168.2.1，用户名为 admin，明文密码
为 admin。
[TM] smartmc ftp-server 192.168.2.1 username admin password simple admin
# 创建 SmartMC 组 S1，并进入 SmartMC 组视图。
[TM] smartmc group S1

```

# 配置 SmartMC 组的匹配规则为按照 IP 地址匹配成员设备。

```
[TM-smartmc-group-S1] match ip-address 192.168.2.0 24
```

# 配置 SmartMC 组使用的配置文件为 startup.cfg。

```
[TM-smartmc-group-S1] startup-configuration startup.cfg
```

```
[TM-smartmc-group-S1] quit
```

# 执行立即升级操作。

```
[TM] smartmc upgrade startup-configuration group S1 startup.cfg
```

#### 4. 验证配置

# SmartMC 网络建立完成后，在 TM 上可以查看到 TC 1、TC 2、TC 3 的相关信息。

```
[TM] display smartmc tc
```

TCID	DeviceType	Sysname	IpAddress	MacAddress	Status	Version
1	S5130S-LI	TC1	192.168.2.3	201c-e7c3-0300	Normal	COMWAREV700R001
2	S5130S-LI	TC2	192.168.2.4	201c-e7c3-0301	Normal	COMWAREV700R001
3	S5130S-LI	TC3	192.168.2.5	201c-e7c3-0302	Normal	COMWAREV700R001

# 显示成员设备配置文件的升级状态。

```
<TM> display smartmc upgrade status
```

ID	IpAddress	MacAddress	Status	UpdateTime	UpdateFile
1	192.168.2.3	201c-e7c3-0300	Finished	Immediately	startup.cfg
2	192.168.2.4	201c-e7c3-0301	Finished	Immediately	startup.cfg
3	192.168.2.5	201c-e7c3-0302	Finished	Immediately	startup.cfg