HUAWEI NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X 系列 V800R023C00SPC500

配置指南

文档版本 01

发布日期 2023-09-30

华为技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为技术有限公司

地址： 深圳市龙岗区坂田华为总部办公楼 邮编：518129

网址： https://www.huawei.com

客户服务邮箱： support@huawei.com

客户服务电话： 4008302118

目 录

插图目录

表格目录

1 配置

1.1 基础配置

1.1.1 首次登录设备配置

1.1.1.1 首次登录设备简介

定义

首次登录设备指的是通过本地登录的方式，对新出厂的设备进行基本系统参数配置的操作，是用户使用设备前置条件。

目的

要对一台新出厂的设备进行业务配置时，通常需要本地登录设备。设备支持通过Console口或通过管理网口首次登录设备。

用户终端的串行口可以与设备Console口直接连接，实现对设备的本地配置。本地登录以后，完成设备名称、管理IP地址和系统时间等基本配置，并配置STelnet协议实现远程登录，为后续配置提供基础环境。

1.1.1.2 通过 Console 口首次登录设备

前置任务

在配置通过Console口登录设备之前，需要完成以下任务：

- 设备正常上电。
- 准备好Console通信电缆。
- 准备好终端仿真软件。

  不同终端仿真软件的使用方法请参照具体软件的使用指导或联机帮助。此处使用第三方软件PuTTY为例进行介绍。

缺省配置

表 1-1 设备 Console 口缺省配置

参数	缺省值
传输速率	9600bit/s
流控方式	不进行流控
校验方式	不进行校验
停止位	1
数据位	8

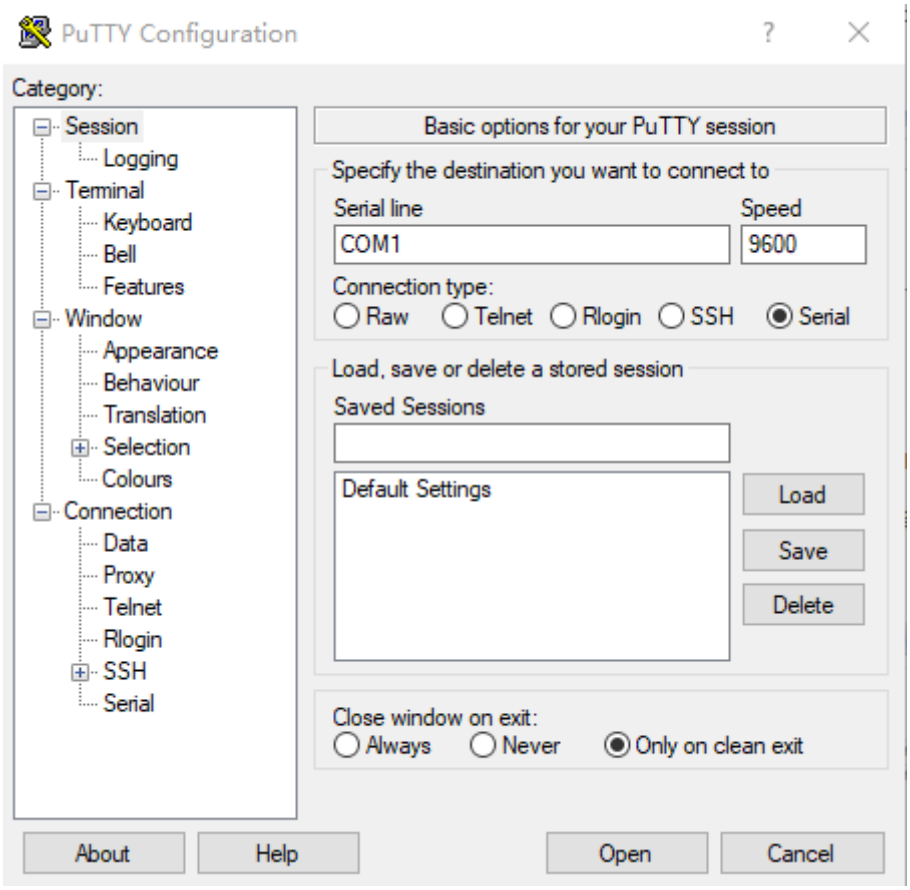
操作步骤

步骤1 将Console通信电缆的DB9（孔）插头插入PC机的串口（COM）中，再将RJ-45插头端插入设备的Console口中。

步骤2 在PC上打开终端仿真软件，新建连接，设置连接的接口以及通信参数。（此处使用第三方软件PuTTY为例进行介绍）

1. 如图1-1所示，单击“Session”，新建连接。

图 1-1 新建连接



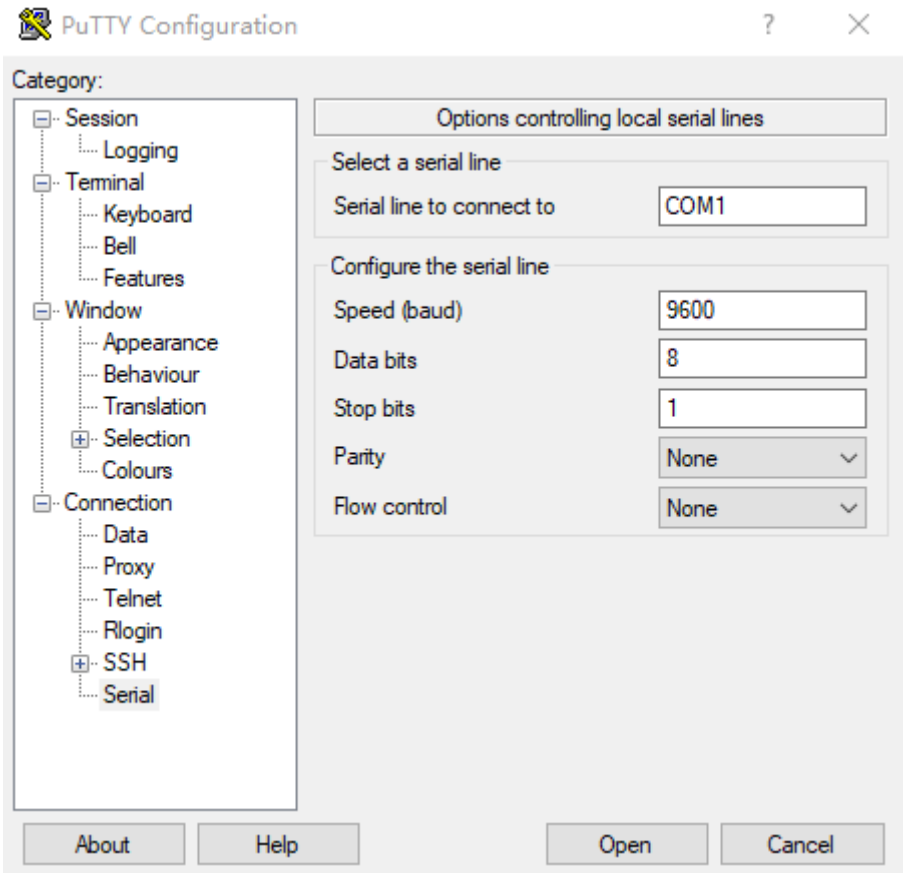
2. 如图1-2所示，单击“Serial”，设置连接的接口以及通信参数。
   a. 连接的接口请根据实际情况进行选择。例如，在Windows系统中，可以通过在“设备管理器”中查看端口信息，选择连接的接口。
   b. 设置终端软件的通信参数需与设备的缺省值保持一致，分别为：传输速率为9600bit/s、8位数据位、1位停止位、无校验和无流控。
   c. 单击“Open”。

   说明

   因为PC端可能会存在多个连接接口，这里需要选择的是连接Console线缆的那个接口。一般情况下，选择的接口是COM1。

   若修改了设备的串口通信参数值，需要在PC端更换通信参数值与设备的串口通信参数值一致后，重新连接。

图 1-2 设置连接的接口以及通信参数



步骤3 Enter键，直到系统出现如下显示，提示用户先设置登录密码，然后再登录。（以下显示信息仅为示意）

```
User interface con0 is available

Please Press ENTER.

Please configure the login password (8-16)
Enter Password:
Confirm Password:       //设置Console口登录密码
Info: Save the password now. Please wait for a moment.
Info: The max number of VTY users is 21, the number of current VTY users online is 1，and total number of terminal users online is 2.
```

```
    The current login time is 2020-06-30 18:15:10+08:00
<HUAWEI>
```

说明

- 第一次通过Console口登录设备时，须设置登录密码。登录成功后，串口的权限为默认的管理员权限。
- 密码为字符串形式，区分大小写，长度范围是8～16。输入的密码至少包含两种类型字符，包括大写字母、小写字母、数字及特殊字符。特殊字符不包括“？”和空格。
- 采用交互方式输入的密码不会在终端屏幕上显示出来。
- 为充分保证设备安全，请用户定期修改密码。

此时用户可以键入命令，对设备进行配置，如果需要帮助可以随时键入“?”。

----结束

1.1.1.3 通过管理网口首次登录设备

背景信息

通过终端连接到网络上，如果网络安全性不高，SSH（Secure Shell）可提供安全的信息保障和强大认证功能，保护设备系统不受IP欺骗等攻击。缺省情况下，用户可以通过管理网口直接登录设备。

客户端已安装OpenSSH软件。

说明

- 设备上电后会自动将管理网口（GigabitEthernet0/0/0）绑定到保留VPN（保留VPN为__LOCAL_OAM_VPN__），并为其配置固定IP地址192.168.0.1/24。
- 用户可以为终端配置192.168.0.0/24网段的其他IP，通过SSH方式登录设备，实现对设备的现场维护。
- 在设备上进行业务配置后，需要及时修改初始用户名和密码，并定期更新密码。管理网口的IP可以修改和删除，并且根据需要关闭该接口。

操作步骤

步骤1 使用OpenSSH软件登录设备。

```
C:\Users\User1>ssh xxx@192.168.0.1
User Authentication
Enter password:
```

说明

此处的用户名和密码需输入设备的缺省账号及密码，请参见《安全加固指南》中的“设备定制及缺省帐号与密码清单”。

步骤2 设备空配置启动时，如果.defcfg文件中没有预置账号，STelnet登录时会进入first-login流程，此时会提示新创建用户并设置密码。

```
Welcome to the first login settings
User name<1, 253>:admin123
Password<8, 128>:
Confirm password:
Info: Create new user successfully, Please use the new password to connect again.
```

创建成功后连接会被关闭，用户需要重新用新创建的用户登录。

----结束

1.1.1.4 首次登录设备后的基本配置

1.1.1.4.1 配置用户登录提示信息

背景信息

当用户使用设备时，若需要向用户提示当前设备的一些信息或警告等，可以在用户登录设备时，将这些提示信息以标题形式出现。

终端连接被激活但用户尚未成功登录时，系统发送登录标题login的信息内容到终端。如果用户成功登录，则系统显示shell标题。

操作步骤

步骤1 进入系统视图。

```
system-view
```

步骤2 设置登录时的提示信息。

```
header login { information text | file file-name }
```

步骤3 设置登录成功后的提示信息。

```
header shell { information text | file file-name }
```

步骤4 提交配置。

```
commit
```

----结束

1.1.1.4.2 配置设备的时区、日期和时间

背景信息

设备出厂时的系统时间是随意设置的，并非是实际时间。当配置设备准备入网时，都需要配置设备的系统时间为当前实际时间，以保证设备发出的日志、告警信息中的时间信息准确。

操作步骤

步骤1 配置设备所在地区及其对应的时区。

```
clock timezone time-zone-name { add | minus } offset
```

缺省情况下，设备采用UTC（Universal Time Coordinated）时区，时区名称默认值“DefaultZoneName”。

add将在UTC标准时间的基础上增加指定的时区偏移量。在系统默认的UTC时区的基础上，加上*offset*，就可以得到*time-zone-name*所标识的时区时间。

minus将在UTC标准时间的基础上减去指定的时区偏移量。在系统默认的UTC时区的基础上，减去*offset*，就可以得到*time-zone-name*所标识的时区时间。

设置时区后，设备本地日志的时间格式为：原系统时间 ± *offset*。例如Apr 27 2020 22:36:09+08:00。

步骤2 设置当前时间和日期。

```
clock datetime [ utc ] time date
```


无缺省值。其中，*time*以“HH:MM:SS”的格式配置设备的当前小时、分钟、秒，*date*以“YYYY-MM-DD”的格式配置的设备当前的年、月、日。

步骤3 提交配置。

```
commit
```

----结束

1.1.1.4.3 配置设备名称

背景信息

为了方便区分网络中的各台设备，可为每一台设备设置不同的设备名称。

操作步骤

步骤1 进入系统视图。

```
system-view
```

步骤2 设置设备名称。

```
sysname host-name
```

缺省情况下，设备主机名为HUAWEI。

可以执行命令**undo sysname**恢复默认的设备主机名。

步骤3 提交配置。

```
commit
```

----结束

1.1.1.4.4 配置设备的管理地址和路由

背景信息

一个网络中，每台设备须有一个全局唯一的管理地址，方便运维人员进行识别和登录。

操作步骤

- 使用管理网口配置管理IP地址。
  a. 进入系统视图。
     ```
     system-view
     ```
  b. 进入管理口视图。
     ```
     interface gigabitethernet0/0/0
     ```
  c. 配置管理口的IP地址和掩码。
     ```
     ip address ip-address { mask | mask-length }
     ```
  d. 返回系统视图。
     ```
     quit
     ```
  e. 配置设备的路由。
     ```
     ip route-static ip-address { mask | mask-length } nexthop-address
     ```
  f. 提交配置。
     ```
     commit
     ```


- 使用普通网口配置管理IP地址。
  a. 进入系统视图。
     system-view
  b. 创建管理网络用的VLAN。
     vlan *vlan-id*
  c. 返回系统视图。
     quit
  d. 进入VLANIF接口视图。
     interface vlanif *vlan-id*
  e. 配置VLANIF的接口IP地址和掩码。
     ip address *ip-address* { *mask* | *mask-length* }
  f. 返回系统视图。
     quit
  g. 进入管理用业务口视图。
     interface *interface-type interface-number*
  h. 配置接口从三层模式切换到二层模式。
     portswitch

     请用户根据实际接口类型自行选择是否要执行此步骤。
  i. 配置接口的链路类型为Access。
     port link-type access
  j. 配置接口的缺省VLAN并同时加入这个VLAN。
     port default vlan *vlan-id*
  k. 返回系统视图。
     quit
  l. 配置设备的路由。
     ip route-static *ip-address* { *mask* | *mask-length* } *nexthop-address*
  m. 提交配置。
     commit

----结束

1.1.1.4.5 配置 STelnet 协议实现登录设备

背景信息

如果管理员想要通过STelnet方式远程登录到目标设备（管理员终端IP与远程设备管理IP三层可达），则须在设备上创建用户，并配置STelnet相关协议。

说明

缺省情况下，新创建的用户在第一次登录设备时，需要修改密码；当管理员重新设置该用户密码后，该用户第一次登录设备时，也需要修改密码。

本节介绍了使用Password认证方式实现通过STelnet协议登录设备。

操作步骤

步骤1 配置VTY用户认证方式为AAA、VTY用户界面所支持的协议为SSH。

```
system-view
user-interface vty first-ui-number [ last-ui-number ]
```

```
authentication-mode aaa    //配置VTY用户认证方式为AAA认证
protocol inbound ssh    //配置VTY用户界面支持的协议为SSH协议
quit
commit
```

步骤2 配置AAA本地用户和登录密码。

```
aaa
local-user user-name password irreversible-cipher irreversible-cipher-password    //创建与SSH用户同名的本地用户和对应的登录密码
local-user user-name service-type ssh terminal    //配置本地用户的服务方式为SSH
local-user user-name level level    //配置本地用户级别
quit
commit
```

步骤3 创建SSH用户，并配置认证方式、服务方式。

```
ssh user user-name    //创建SSH用户
ssh user user-name authentication-type password    //配置SSH用户的认证方式为password
ssh user user-name service-type stelnet    //配置SSH用户的服务方式为STelnet
stelnet server enable    //使能设备的STelnet服务器端功能
ssh server-source -i interface-type interface-number    //配置SSH服务器的源接口。如果登录地址是IPv6类型，则可执行命令ssh ipv6 server-source -a ipv6-address来配置SSH服务器的源IP地址
commit
```

说明

确保SSH用户名称与本地用户名称相同。

创建的用户在第一次登录设备时，需要修改密码。

----结束

1.1.1.5 检查配置结果

操作步骤

- 执行命令**display clock**，查看系统当前日期和时钟。
- 执行命令**display sysname**，查看设备当前的主机名。
- 执行命令**display current-configuration**，查看设备当前配置。
- 执行命令**display language character-set [test]**，查看当前系统以及终端登录软件支持的字符集编码。

----结束

1.1.1.6 举例：首次登录设备并进行基本配置

组网需求

通过Console口首次登录设备后，对设备进行基本配置并配置通过STelnet远程登录的0～4号用户的级别为3级，认证方式为AAA认证。PC2与设备之间路由可达。

图 1-3 通过 Console 口首次登录设备后进行基本配置组网图

说明

本例中Interface1代表接口管理口**GigabitEthernet0/0/0**。

配置思路

1. 通过Console口登录设备。
2. 对设备进行基本配置。

操作步骤

步骤1 PC1通过设备Console口登录设备，具体操作请参见**通过Console口首次登录设备**。

步骤2 对设备进行基本配置。

设置系统的日期、时间和时区。

```
<HUAWEI> clock timezone BJ add 08:00:00
<HUAWEI> clock datetime 20:20:00 2018-08-08
```

说明

在配置设备的当前时间和日期前，需要执行**clock timezone**命令配置时区。如果不配置时区，执行**clock datetime**命令配置的是UTC时间。

设置设备名称和管理网口的IP地址。

```
<HUAWEI> system-view
[~HUAWEI] sysname Device
[*HUAWEI] commit
[~Device] interface gigabitethernet 0/0/0
[~Device-GigabitEthernet0/0/0] ip address 10.137.217.203 24
[*Device-GigabitEthernet0/0/0] quit
[*Device] commit
```

假如设备的网关是10.137.217.1，配置设备的缺省路由。

```
[~Device] ip route-static 0.0.0.0 0 10.137.217.1
[*Device] commit
```

配置SSH客户端加密算法、HMAC认证算法、密钥交换算法列表、公钥算法。

```
[~Device] ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
[~Device] ssh server hmac sha2_256 sha2_512
[~Device] ssh server key-exchange dh_group_exchange_sha256
[~Device] ssh server publickey rsa_sha2_256 rsa_sha2_512
[~Device] ssh server dh-exchange min-len 3072
[*Device] commit
```

配置SSH用户与本地用户相关参数，实现通过SSH协议登录设备。

```
[~Device] user-interface vty 0 4
[~Device-ui-vty0-4] authentication-mode aaa
[~Device-ui-vty0-4] protocol inbound ssh
[~Device-ui-vty0-4] quit
[*Device] commit
[~Device] aaa
[~Device-aaa] local-user admin123 password
Please configure the password (8-128)
Enter Password:
Confirm Password:
[~Device-aaa] local-user admin123 service-type ssh
[~Device-aaa] local-user admin123 level 3
[*Device-aaa] quit
[*Device] commit
[~Device] ssh user admin123
[*Device] ssh user admin123 authentication-type password
[*Device] ssh user admin123 service-type stelnet
[*Device] ssh server-source all-interface
[*Device] stelnet server enable
[*Device] commit
```

----结束

检查配置结果

完成以上配置后，终端PC2通过STelnet登录设备，此处以使用第三方软件OpenSSH和Windows命令行提示符为例进行配置。

- OpenSSH软件的安装请参考该软件的安装说明。
- 使用OpenSSH软件从终端访问设备时，需要使用OpenSSH的命令，命令的使用可以参见该软件的帮助文档。
- 只有安装了OpenSSH软件后，Windows命令行提示符才能识别OpenSSH相关命令。

进入Windows的命令行提示符，执行OpenSSH命令，登录设备。(以下显示信息仅为示意)

```
C:\Users\User1>ssh admin123@10.137.217.203
admin123@10.137.217.203's password:

Info: The max number of VTY users is 21, the number of current VTY users online is 5, and total number of terminal users online is 5.
      The current login time is 2020-12-15 14:23:00.
<Device>
```

配置脚本

Device的配置文件

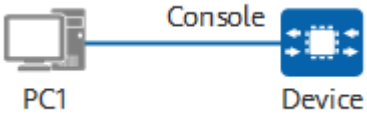
```
#
sysname Device
#
stelnet server enable
#
clock timezone BJ add 08:00:00
#
aaa
 local-user admin123 password irreversible-cipher $1d$+,JS+))\\2$KVNj(.
3`_5x0FCKGv}H&.kUTI`Ff&H*eBqO.ua>)$
 local-user admin123 service-type ssh
 local-user admin123 level 3
#
interface GigabitEthernet0/0/0
 ip address 10.137.217.203 255.255.255.0
#
ip route-static 0.0.0.0 0.0.0.0 10.137.217.1
#
stelnet server enable
ssh user admin123
ssh user admin123 authentication-type password
ssh user admin123 service-type stelnet
ssh server-source all-interface
#
ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
ssh server hmac sha2_256 sha2_512
ssh server key-exchange dh_group_exchange_sha256
ssh server publickey rsa_sha2_256 rsa_sha2_512
ssh server dh-exchange min-len 3072
#
user-interface vty 0 4
 authentication-mode aaa
 user privilege level 3
 protocol inbound ssh
#
return
```


1.1.1.7 举例：配置用户通过 Console 口首次登录设备

组网需求

当设备第一次上电，需要对此设备进行配置和管理时，可以通过Console口登录。如图 配置Console口首次登录设备组网图，设备的Console口与用户侧主机PC1相连，要求用户通过Console口登录设备。

图 1-4 配置 Console 口首次登录设备组网图



配置思路

1. 通过Console口登录设备。
2. 对设备进行基本配置。

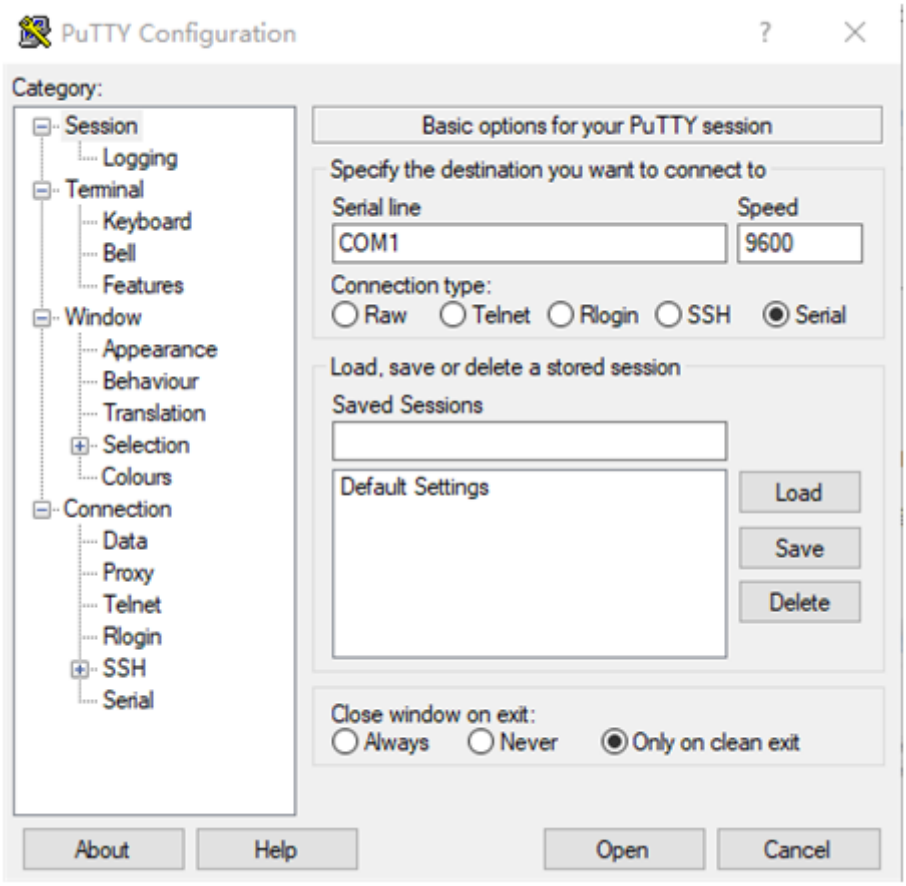
操作步骤

步骤1 将Console通信电缆的DB9（孔）插头插入PC机的串口（COM）中，再将RJ-45插头端插入设备的Console口中。

步骤2 在PC上打开终端仿真软件，新建连接，设置连接的接口以及通信参数。（此处使用第三方软件PuTTY为例进行介绍）

1. 如图 新建连接所示，单击“Session”，新建连接。

图 1-5 新建连接



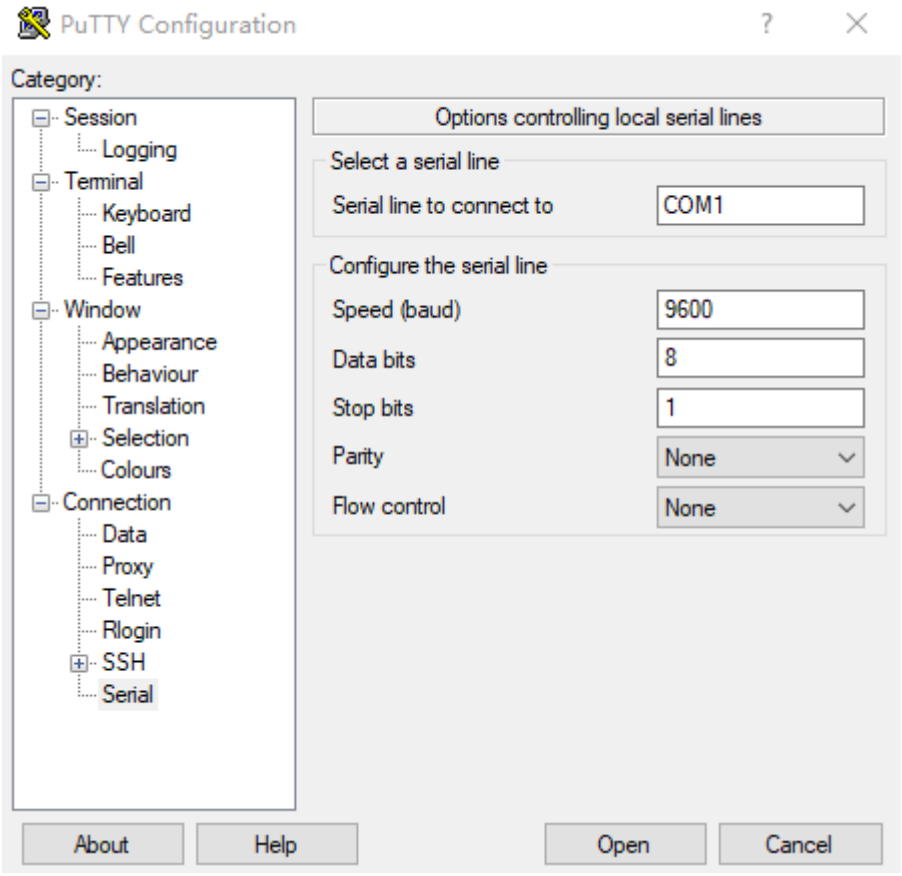
2. 如**图 设置连接的接口以及通信参数**所示，单击“Serial”，设置连接的接口以及通信参数。
   a. 连接的接口请根据实际情况进行选择。例如，在Windows系统中，可以通过在“设备管理器”中查看端口信息，选择连接的接口。
   b. 设置终端软件的通信参数需与设备的缺省值保持一致，分别为：传输速率为9600bit/s、8位数据位、1位停止位、无校验和无流控。
   c. 单击“Open”。

   说明

   因为PC端可能会存在多个连接接口，这里需要选择的是连接Console线缆的那个接口。一般情况下，选择的接口是COM1。

   若修改了设备的串口通信参数值，需要在PC端更换通信参数值与设备的串口通信参数值一致后，重新连接。

图 1-6 设置连接的接口以及通信参数



步骤3 Enter键，直到系统出现如下显示，提示用户先设置登录密码，然后再登录。（以下显示信息仅为示意）

```
User interface con0 is available

Please Press ENTER.

Please configure the login password (8-16)
Enter Password:
Confirm Password:       //设置Console口登录密码
Info: Save the password now. Please wait for a moment.
Info: The max number of VTY users is 21, the number of current VTY users online is 1，and total number of terminal users online is 2.
    The current login time is 2020-06-30 18:15:10+08:00
<HUAWEI>
```

说明

- 第一次通过Console口登录设备时，须设置登录密码。登录成功后，串口的权限为默认的管理员权限。
- 密码为字符串形式，区分大小写，长度范围是8～16。输入的密码至少包含两种类型字符，包括大写字母、小写字母、数字及特殊字符。特殊字符不包括“？”和空格。
- 采用交互方式输入的密码不会在终端屏幕上显示出来。
- 为充分保证设备安全，请用户定期修改密码。

----结束

检查配置结果

完成以上配置后，用户可以键入命令，对设备进行配置，如果需要帮助可以随时键入“?”。

1.1.2 熟悉命令行配置

1.1.2.1 如何使用命令行

1.1.2.1.1 进入命令行视图

设备提供丰富的功能，相应地也提供了多样的配置和查询命令。为便于用户使用这些命令，华为设备按功能分类将命令分别注册在不同的命令行视图下。配置某一功能时，需首先进入对应的命令行视图，然后执行相应的命令进行配置。

设备提供的命令视图有很多，下面提到的视图是最常用的视图。其他视图的进入方式在具体的命令中都有说明，请参见《命令参考》。

常用的命令行视图

- 用户视图

  在用户视图下，用户可以完成查看运行状态和统计信息等功能。

  进入视图：用户从终端成功登录至设备即进入用户视图，在屏幕上显示：

  ```
  <HUAWEI>
  ```

- 系统视图

  在系统视图下，用户可以配置系统参数以及通过该视图进入其他的功能配置视图。

  进入视图：在用户视图下，输入命令**system-view**后回车，进入系统视图。

  ```
  <HUAWEI> system-view
  Enter system view, return user view with return command.
  [~HUAWEI]
  ```

- 接口视图

  配置接口参数的视图称为接口视图。在该视图下可以配置接口相关的物理属性、链路层特性及IP地址等重要参数。

  进入视图：使用**interface**命令并指定接口类型及接口编号可以进入相应的接口视图。以GE接口为例：

  ```
  [~HUAWEI] interface gigabitethernet X/Y/Z
  [~HUAWEI-GEX/Y/Z]
  ```

  *X/Y/Z*为需要配置的接口的编号，分别对应“槽位号/子卡号/接口序号”。

- 路由协议视图

  路由协议的大部分参数是在相应的路由协议视图下进行配置的。例如IS-IS协议视图、OSPF协议视图、RIP协议视图。

  进入视图：在系统视图下，使用路由协议进程运行命令可以进入到相应的路由协议视图。

  ```
  [~HUAWEI] isis
  [~HUAWEI-isis-1]
  ```

命令行提示符“HUAWEI”是缺省的主机名（sysname）。通过提示符可以判断当前所处的视图，例如：“<>”表示用户视图，“[]”表示除用户视图以外的其他视图。

用户可以在任意视图中，执行!或#加字符串，此时的用户输入将全部（包括!和#在内）作为系统的注释行内容，不会产生对应的配置信息。

说明

- 有些在系统视图下执行的命令，在其他视图下也可以执行，但实现的功能与命令视图密切相关。
- 在系统视图下，可以执行命令**diagnose**进入诊断视图。诊断命令行主要用于设备的故障诊断，在此视图下执行某些命令可能导致设备异常或业务中断。如果您需要使用此类命令行，请联系技术支持人员，在技术支持人员指导下谨慎使用。

退出命令行视图

执行**quit**命令，即可从当前视图退出至上一层视图。

例如，执行**quit**命令从AAA视图退回到系统视图，再执行**quit**命令退回到用户视图。

```
[~HUAWEI-aaa] quit
[~HUAWEI] quit
<HUAWEI>
```

如果需要从AAA视图直接退回到用户视图，则可以在键盘上键入快捷键<Ctrl+Z>或者执行**return**命令。

使用快捷键<Ctrl+Z>直接退回到用户视图。

```
[~HUAWEI-aaa]          //键入<Ctrl+Z>
<HUAWEI>
```

执行**return**命令直接退回到用户视图。

```
[~HUAWEI-aaa] return
<HUAWEI>
```

1.1.2.1.2 选择命令行配置生效模式（立即生效、两阶段生效）

为了保障用户配置的可靠性，系统支持两种配置生效模式，即立即生效模式和两阶段生效模式。缺省情况下，系统的配置生效模式为两阶段生效模式。

用户在进行配置前必须先进入系统视图。进入系统视图后，系统根据用户选择的配置模式启动相应的配置事务。

- 立即生效模式是传统的生效模式。

  在立即生效模式下，用户在输入命令行并输入回车键后，系统执行语法检查，如果语法检查通过则配置立即生效。
- 两阶段生效模式将系统配置分为两个阶段。
  - 第一阶段用户输入配置命令，系统在候选数据集执行命令语法和语义检查，对于有错误的配置语句，系统通过命令行终端提醒用户配置错误及错误原因。
  - 用户完成系列配置命令的输入后，需要提交配置，系统进入第二阶段，即配置的提交阶段。此时系统将候选数据集上的配置下发到业务，如果业务生效则将候选数据集的配置合并到当前系统的运行数据集。在配置的提交阶段，系统会进行检查，发现配置相同时会产生提示信息。

表 1-2 立即生效模式和两阶段生效模式的优缺点

生效模式	优点	缺点
立即生效模式	配置产生的业务影响能够立即反映在系统上。	由于配置是立即生效的，用户在配置错误时会直接对运行业务产生影响，且不能将业务作为一个整体直接丢弃，用户需要逐条撤销配置。
两阶段生效模式	• 对业务的配置能够整体生效。 • 可以预览候选数据集里面的配置。 • 在预览配置后发现业务配置产生错误或配置不符合用户预期时，能够立即清除未生效的配置。 • 可以将配置过程对现有业务的影响降到最低。	需要输入**commit**命令配置才可以生效。

在两阶段生效模式下，一般都需要执行**commit**提交配置，但是以下几种情况，用户不需要执行**commit**提交配置，即可生效。

- 查询类命令不需要**commit**，例如**display interface**。
- 维护类命令不需要**commit**，例如: **reset saved-configuration**、**reset keepalive packets count**。
- 进入物理设备上存在的视图，例如物理接口视图，不需要**commit**。例如：**interface GigabitEthernet1/0/1**
- 用户执行的命令没有对已有配置产生变更时不需要**commit**。

立即生效模式

进入立即生效模式。

```
system-view immediately
```

立即生效模式下，用户的提示符如下：

```
<HUAWEI> system-view immediately
Enter system view, return user view with return command.
[HUAWEI]
```

两阶段生效模式

1. 进入两阶段生效模式。

   ```
   system-view
   ```

2. （可选）查看用户未提交的配置。

   ```
   display configuration candidate [ merge | changes ]
   ```


不指定**merge**和**changes**参数仅查看当前用户未提交的配置。指定**merge**参数，可以查看用户未提交的配置和系统已有的配置。指定**changes**参数用来查看当前会话未提交的配置和设备正在运行配置之间的差异。

在提交配置之前，用户可以继续对未提交的配置进行编辑。

说明

用户执行命令**display configuration candidate changes**查看未提交配置和当前运行配置之间的差异时，如果提示当前运行配置有变更，这种情况下需先执行**refresh configuration candidate**解决冲突，才能查看配置差异。

3. （可选）清除未提交的所有配置。
   clear configuration candidate
4. （可选）配置试运行功能。
   commit trial [*time*] [**persist** *persistId*]

   配置试运行可以在不中断现网业务的情况下，对新的功能和业务进行在网试运行，提升网络可靠性。当试运行时间超过用户的设定时间后，试运行的配置将自动回退。当不指定参数**persist**时，试运行只在当前会话有效；指定参数**persist**进入持续性配置试运行，试运行不会因为会话结束而终止。

   试运行过程中，用户可以通过**abort trial**命令取消配置试运行，系统配置回退至试运行前的配置状态。用户可以通过命令**display configuration trial status**查看系统试运行的状态。
5. 提交配置。
   commit [**persist** *persistId*]

   当持续性试运行没有超时回退之前，可以在任意会话中指定参数**persist**提交试运行的配置。

在两阶段生效模式中，如果用户进行了配置修改但未提交时，系统提示符中的“~”将变为“*”，提示用户有未提交的配置。当用户通过**commit**完成配置提交后，“*”重新变为“~”。

举例如下：

```
<HUAWEI> system-view
Enter system view, return user view with return command.
[~HUAWEI] sysname HUAWEIA
[*HUAWEI] commit
[~HUAWEI]
```

说明

- 不管是立即生效模式还是两阶段生效模式，在配置过程中，为了保护某些业务，在业务进程刚开始启动时，会进行配置锁定，所以在此期间用户执行命令或提交配置可能会出现失败，但可以执行查询操作。如果出现失败，建议等待30秒后再执行命令或提交配置，如果再次失败，说明配置被某个用户锁定。
- 用户锁定配置的命令是**configuration exclusive**，如果配置被其他用户锁定，则首先需要联系该用户解除配置锁定。
- 两阶段生效模式下，当多个用户进行相同配置时，对于后提交的用户，系统会提示配置冲突，提交不成功。

1.1.2.1.3 关闭二次确认功能

设备上有些**undo**命令，如果用户误操作会关联删除相关特性的配置，导致业务中断，造成用户网络中断。缺省情况下，为了防止用户误操作，执行这些undo命令时，需要用户进行二次交互确认，命令范围包含：**undo mpls**、**undo mpls te**、**undo mpls rsvp**、**undo mpls ldp**、**undo mpls l2vpn**、**undo multicast ipv6 routing-**

enable、**undo multicast routing-enable**、**undo pim**、**undo igmp**、**undo bfd**、**undo stp enable**。

如果用户想关闭二次确认功能，可以在系统视图下执行命令**configuration prevent-misoperation disable**。

说明

为防止误操作导致某些业务不可用，建议使能二次交互确认功能。

1.1.2.1.4 命令行智能回退功能

每条命令行都有支持的视图，比如**vlan**命令支持的视图是系统视图。在业务部署过程中，用户可能需要在不同视图下配置命令行，这样操作步骤较多，业务部署效率低。

设备支持命令行智能回退功能，即如果命令行在当前视图下无法匹配成功，系统会自动回退到系统视图下，如果可以匹配该命令行，则可以直接下发配置，从而减少了命令行操作步骤。

例如，**interface**命令支持的视图是系统视图，但是若用户正在VLAN视图下执行操作，可以直接在VLAN视图下执行**interface**命令行进入接口视图。

```
<HUAWEI> system-view
[~HUAWEI] vlan 2
[*HUAWEI-vlan2] interface gigabitethernet 1/0/1
[*HUAWEI-GigabitEthernet1/0/1]
```

如果用户想关闭命令行智能回退功能，可以在用户视图下执行命令**undo terminal command forward matched upper-view**，关闭命令行智能回退功能。缺省情况下，命令行智能回退功能使能。

说明

- 根据业务部署情况，有的情况下系统会关闭智能回退功能，以免影响业务部署。比如某命令支持系统视图和接口视图，但是作用的范围不同，如果接口视图下有业务与之互斥，不能执行，也不会在系统视图下匹配成功。
- 智能回退需要输入完整命令行，不支持联想。

1.1.2.1.5 设置命令级别

背景信息

系统将命令进行分级管理，各个视图下的每条命令都有指定的级别。设备管理员可以根据用户需要重新设置命令的级别，以实现低级别用户可以使用部分高级别命令的需求，或者将命令的级别提高，增加设备的安全性。

- 为了限制不同用户对设备的访问权限，系统对用户也进行了分级管理。用户的级别与命令级别对应，不同级别的用户登录后，只能使用等于或低于自己级别的命令。缺省情况下，命令级别按0～3级进行注册，用户级别按0～3级进行注册，未配置**command-privilege level rearrange**时用户级别和命令级别对应关系如**表1-3**所示。

表 1-3 用户级别和命令级别对应关系

用户级别	命令级别	说明
0	参观级（0）	网络诊断工具命令（ping、tracert）、从本设备出发访问外部设备的命令（Telnet客户端）、部分display命令等。
1	参观级（0）、监控级（1）	用于系统维护，包括display等命令。 **说明** 并不是所有display命令都是监控级，比如**display current-configuration**命令和**display saved-configuration configuration**命令是3级管理级。各命令的级别请参见《命令参考》手册。
2	参观级（0）、监控级（1）、配置级（2）	业务配置命令，包括路由、各个网络层次的命令，向用户提供直接网络服务。
3	参观级（0）、监控级（1）、配置级（2）、管理级（3）	用于系统基本运行的命令，对业务提供支撑作用，包括文件系统、FTP、TFTP下载、命令级别设置命令以及用于业务故障诊断的debugging命令等。

- 如果用户需要实现权限的精细管理，可以使用命令**command-privilege level rearrange**将命令级别批量提升。

须知

建议用户不要修改缺省的命令级别，以免造成操作和维护上的不便甚至给设备带来安全隐患。

操作步骤

步骤1 进入系统视图。

```
system-view
```

步骤2 设置指定视图内命令的级别。

```
command-privilege level level view view-name command-key
```

步骤3 （可选）批量提升命令的级别。

```
command-privilege level rearrange
```

在执行此命令之前，需要用户确保自己的级别为管理级别（3级或15级），否则无法执行该命令。用户管理级别可以是默认的3级，也可以是权限提升后的15级。

如果用户没有对某条命令单独调整过命令级别（执行**command-privilege level**命令修改过命令级别的命令，维持原来级别不作调整），命令级别批量提升后，原注册的所有命令行按以下原则自动调整：

- 0级和1级命令保持级别不变。
- 2级命令提升到10级，3级命令提升到15级。
- 命令级别批量提升之后，2～9级和11～14级的命令级别中没有命令行。用户可以单独调整需要的命令行到这些级别中，以实现用户权限的精细化管理。

步骤4 提交配置。

```
commit
```

----结束

1.1.2.1.6 编辑命令行

命令行编辑功能

设备的命令行接口提供基本的命令行编辑功能。设备支持多行编辑，每条命令最大长度为3100个字符，命令关键字不区分大小写，命令参数是否区分大小写则由各命令定义的参数决定。

一些常用的编辑功能如表1-4所示。

表 1-4 编辑功能表

功能键	功能
普通按键	若编辑缓冲区未满，则插入到当前光标位置，并向右移动光标，否则，响铃告警。
退格键Backspace	删除光标位置的前一个字符，光标左移，若已经到达命令首，则响铃告警。
左光标键←或<Ctrl+B>	光标向左移动一个字符位置，若已经到达命令首，则响铃告警。
右光标键→或<Ctrl+F>	光标向右移动一个字符位置，若已经到达命令尾，则响铃告警。

编辑命令行时的操作技巧

不完整关键字输入

设备支持不完整关键字输入，即在当前视图下，当输入的字符能够匹配唯一的关键字时，可以不必输入完整的关键字。该功能提供了一种快捷的输入方式，有助于提高操作效率。

如果当前输入匹配到的命令不唯一，即可以匹配到多个命令，则需要继续输入，直至当前命令唯一匹配才能下发成功。

比如**display current-configuration**命令，可以输入**d cu**、**di cu**或**dis cu**等都可以执行此命令，但不能输入**d c**或**dis c**等，因为以**d c**、**dis c**开头的命令不唯一。

须知

系统可正确执行的命令长度最大为3100个字符，包括使用不完整格式的情况。如果使用不完整格式进行配置，由于命令保存到配置文件中时使用的是完整格式，可能导致配置文件中存在长度超过3100个字符的命令。系统重启时，这类命令将无法恢复。因此，在使用不完整格式的命令进行配置时，也需要注意命令的总长度。

Tab键的使用

输入不完整的关键字后按下Tab键，系统自动补全关键字：

- 如果与之匹配的关键字唯一，则系统用此完整的关键字替代原输入并换行显示，光标距词尾空一格。例如：
  a. 输入不完整的关键字。
     ```
     [~HUAWEI] info-
     ```
  b. 按下Tab键。
     则系统用此完整的关键字替代原输入并换行显示，光标距词尾空一格。
     ```
     [~HUAWEI] info-center
     ```
- 如果与之匹配的关键字不唯一，反复按<Tab>键可循环显示所有以输入字符串开头的关键字，此时光标距词尾不空格。例如：
  a. 输入不完整的关键字。
     ```
     [~HUAWEI] info-center log
     ```
  b. 按下Tab键。
     系统首先显示所有匹配的关键字的前缀，本例中前缀是“log”。
     ```
     [~HUAWEI] info-center log-severity
     ```
     继续按Tab键循环翻词，此时光标距词尾不空格。
     ```
     [~HUAWEI] info-center logbuffer
     [~HUAWEI] info-center logfile
     [~HUAWEI] info-center loghost
     ```
     找到所需要的关键字后，停止按Tab键。
- 如果没有与之匹配的关键字，按Tab键后，换行显示，输入的关键字不变。例如：
  a. 输入错误的关键字。
     ```
     [~HUAWEI] info-center loglog
     ```
  b. 按下Tab键。
     ```
     [~HUAWEI] info-center loglog
     ```
     系统换行显示，但输入的关键字loglog不变，而且光标距词尾不空格，说明无此关键字。

1.1.2.1.7 使用命令行在线帮助

用户在使用命令行时，可以使用在线帮助以获取实时帮助，从而无需记忆大量的复杂的命令。

在线帮助通过键入“?”来获取，在命令行输入过程中，用户可以随时键入“?”以获得在线帮助。命令行在线帮助可分为完全帮助和部分帮助。

完全帮助

当用户输入命令时，可以使用命令行的完全帮助获取全部关键字和参数的提示。下面给出几种完全帮助的实例供参考：

- 在任一命令视图下，键入“?”获取该命令视图下所有的命令及其简单描述。举例如下：
```
<HUAWEI> ?
Current view commands:
  activate        Activate locked user
  cd              Change current directory
  clear           Clear operation
  clock           Clock status and configuration information
  copy            Copy from one file to another
...
```
- 键入一条命令的部分关键字，后接以空格分隔的“?”，如果该位置为关键字，则列出全部关键字及其简单描述。举例如下：
```
<HUAWEI> system-view
[~HUAWEI] user-interface vty 0 4
[~HUAWEI-ui-vty0-4] authentication-mode ?
  aaa       AAA authentication
  password  Authentication through the password of a user terminal interface
```
其中“aaa”和“password”是关键字，“AAA authentication”和“Authentication through the password of a user terminal interface”是对关键字的描述。
- 键入一条命令的部分关键字，后接以空格分隔的“?”，如果该位置为参数，则列出有关的参数名和参数描述。举例如下：
```
<HUAWEI> system-view
[~HUAWEI] ssh server timeout ?
  INTEGER<1-35791>  Set the authentication timeout, the default value is 60 seconds
[~HUAWEI] ssh server timeout 35 ?
  <cr>
[~HUAWEI] ssh server timeout 35
```
其中，“INTEGER<1-35791>”是参数取值的说明，“Set the authentication timeout, the default value is 60”是对参数作用的简单描述。“<cr>”表示该位置没有关键字或参数，直接键入回车即可执行。

部分帮助

当用户输入命令时，如果只记得此命令关键字的开头一个或几个字符，可以使用命令行的部分帮助获取以该字符串开头的所有关键字的提示。下面给出几种部分帮助的实例供参考：

- 键入一字符串，其后紧接“?”，列出以该字符串开头的所有关键字。举例如下：
```
<HUAWEI> d?
  debugging                       delete
  dir                             display
<HUAWEI> d
```
- 键入一条命令，后接一字符串紧接“?”，列出命令以该字符串开头的所有关键字。举例如下：

```
<HUAWEI> display s?
  sysname                         system
```

- 输入命令的某个关键字的前几个字母，按下**<tab>**键，可以显示出完整的关键字，前提是这几个字母可以唯一标示出该关键字，否则，连续按下<tab>键，可出现不同的关键字，用户可以从中选择所需要的关键字。

说明

以上获取的在线帮助的显示信息仅为示意，请以设备实际显示为准。

1.1.2.1.8 解读命令行的错误信息

用户键入的命令，如果通过语法检查，则正确执行，否则系统将会向用户报告错误信息。

常见错误信息参见**表1-5**。

表 1-5 命令行常见错误信息表

英文错误信息	错误原因
Error: Unrecognized command found at '^' position.	没有查找到命令
	没有查找到关键字
Error: Wrong parameter found at '^' position.	参数类型错
	参数值越界
Error: Incomplete command found at '^' position.	输入命令不完整
Error: Too many parameters found at '^' position.	输入参数太多
Error: Ambiguous command found at '^' position.	输入命令不明确

1.1.2.1.9 使用 undo 命令行

在命令前加**undo**关键字，即为**undo**命令行。**undo**命令行一般用来恢复缺省情况、禁用某个功能或者删除某项配置。几乎每条配置命令都有对应的**undo**命令行。

下面给出使用**undo**命令行的示例供参考：

- 使用**undo**命令行恢复缺省情况

  undo sysname命令是用来恢复设备的主机名到缺省情况。举例如下：

  ```
  <HUAWEI> system-view
  [~HUAWEI] sysname Server
  [*HUAWEI] commit
  [~Server] undo sysname
  [*Server] commit
  [*HUAWEI]
  ```

- 使用**undo**命令禁用某个功能

  undo sftp server enable命令是用来关闭设备的SFTP服务器功能。举例如下：

```
<HUAWEI> system-view
[~HUAWEI] sftp server enable
Info: Succeeded in starting the SFTP server.
[*HUAWEI] commit
[~HUAWEI] undo sftp server
Warning: The operation will stop the SFTP server. Continue? [Y/N]:Y
Info: Succeeded in closing the SFTP server.
[*HUAWEI] commit
```

- 使用**undo**命令删除某项设置

  undo header命令是用来取消用户登录设备时终端上显示的标题信息。举例如下：

  ```
  <HUAWEI> system-view
  [~HUAWEI] header login information "Hello,Welcome to Huawei!"
  [*HUAWEI] commit
  ```

  退出设备后重新登录，在验证用户前，会出现“Hello,Welcome to Huawei!”，然后执行相应的**undo header login**命令：

  ```
  Hello,Welcome to Huawei!
  Password:
  Info: The max number of VTY users is 21, and the number of current VTY users on
  line is 2.
      The current login time is 2019-11-06 16:31:24.
  <HUAWEI> system-view
  [~HUAWEI] undo header login
  [*HUAWEI] commit
  ```

  再次退出设备后重新登录，在验证用户前，则不会出现任何标题信息：

  ```
  Password:
  Info: The max number of VTY users is 21, and the number of current VTY users on
  line is 2.
      The current login time is 2019-11-06 16:45:06.
  <HUAWEI>
  ```

说明

以上示例中设备的显示信息仅为示意，请以设备实际显示为准。

1.1.2.1.10 查看历史命令

设备能够自动保存用户键入的历史命令。当用户需要输入之前已经执行过的命令时，可以调用设备保存的历史命令。

缺省情况下，为每个登录用户保存10条历史命令。可以通过**history-command max-size** *size-value*命令在相应的用户界面视图下重新设置保存历史命令的条数，最大设置为256。

说明

不推荐用户将此值设置过大，因为可能会花费较长时间才查看到所需要的历史命令，反而影响配置效率。

对历史命令的操作如**表1-6**所示。

表 1-6 访问历史命令

操作	命令或功能键	结果
显示历史命令	**display history-command** [**all-users**]	• 不指定**all-users**，显示当前用户键入的历史命令。 • 指定**all-users**，显示的是所有登录用户键入的历史命令。（3级及3级以上的用户才能执行此参数）
访问上一条历史命令	上光标键或者<Ctrl+P>	如果还有更早的历史命令，则取出上一条历史命令，否则响铃警告。
访问下一条历史命令	下光标键或者<Ctrl+N>	如果还有更新的历史命令，则取出下一条历史命令，否则显示为空，响铃警告。

说明

对于Windows 9X的超级终端，↑光标键无效，这是由于Windows 9X的超级终端对这个键作了不同解释，这时可以用快捷键<Ctrl+P>代替↑光标键达到同样目的。

在使用历史命令功能时，需要注意：

- 保存的历史命令与用户输入的命令格式相同，如果用户使用了命令的不完整形式，保存的历史命令也是不完整形式。
- 如果用户多次执行同一条命令，则历史命令中只保留最近的一次。但如果执行时输入的形式不同，将作为不同的命令对待。

  例如：多次执行**display current-configuration**命令，历史命令中只保存一条。如果执行**display current-configuration**和**dis curr**，将保存为两条历史命令。
- 当前用户的历史命令可以在用户视图下通过**reset history-command**命令进行清除，所有用户的历史命令可以在用户视图下通过**reset history-command all-users**命令进行清除，清除后则无法显示和访问之前执行过的历史命令。

1.1.2.1.11 使用命令行的快捷键

用户可以使用设备中的快捷键，完成对命令的快速输入，从而简化操作。

系统中的快捷键分成两类，自定义快捷键和系统快捷键：

- 自定义快捷键：共有4个，包括<Ctrl+G>、<Ctrl+L>、<Ctrl+O>和<Ctrl+U>。用户可以根据自己的需要将这4个快捷键与任意命令进行关联，当使用快捷键时，系统自动执行它所对应的命令。
- 系统快捷键：是系统中固定的。这种快捷键不由用户定义，代表固定功能。常用的系统快捷键如表1-7所示。

说明

快捷键的功能可能受用户所用的终端影响，例如用户终端本身自定义的快捷键与设备系统中的快捷键功能发生冲突，此时如果用户键入快捷键将会被终端程序截获而不能执行它所对应的命令行。

自定义快捷键

如果用户经常性地使用某一个或某几个命令时，可以将这些命令定义成快捷键，方便用户操作，提升效率。只有管理级用户有定义快捷键的权限。配置方法如下：

1. 进入系统视图。
   system-view
2. 配置快捷键对应的命令。
   hotkey { **CTRL_G** | **CTRL_L** | **CTRL_O** | **CTRL_U** } *command-text*
   系统支持用户自定义四个快捷键，快捷键的缺省值如下：
   - <Ctrl+G>：对应命令**display current-configuration**
   - <Ctrl+L>：对应命令**display ip routing-table**
   - <Ctrl+O>：对应命令**undo debugging all**
   - <Ctrl+U>：默认值为空。
3. 提交配置。
   commit

说明

- 定义快捷键时，对于由多个命令字组成的命令，即命令中间有空格，需要使用双引号标识，例如：**hotkey ctrl_l "display tcp status"**。对于单个命令字的命令，即命令中没有空格，不需要使用双引号。
- 可通过**display hotkey**命令查看系统快捷键的使用情况，包括已定义、未定义以及系统快捷键。
- 可通过**undo hotkey**命令恢复系统的快捷键缺省值。
- 快捷键的执行与命令一样，也会将命令原形记录在命令缓冲区和日志中以备问题定位和查询。
- 定义的快捷键对所有的登录用户均有效，但是如果用户没有快捷键所定义命令的使用权限，则执行此快捷键后会提示错误。

系统快捷键

表 1-7 系统快捷键

功能键	功能
<Ctrl+A>	将光标移动到当前行的开头。
<Ctrl+B>	将光标向左移动一个字符。
<Ctrl+C>	停止当前正在执行的功能。
<Ctrl+D>	删除当前光标所在位置的字符。
<Ctrl+E>	将光标移动到当前行的末尾。
<Ctrl+F>	将光标向右移动一个字符。

功能键	功能
<Ctrl+H>	删除光标左侧的一个字符。
<Ctrl+K>	在连接建立阶段终止呼出的连接。
<Ctrl+N>	显示历史命令缓冲区中的后一条命令。
<Ctrl+P>	显示历史命令缓冲区中的前一条命令。
<Ctrl+R>	重新显示当前行信息。
<Ctrl+T>	终止呼出的连接。
<Ctrl+V>	粘贴剪贴板的内容。
<Ctrl+W>	删除光标左侧的一个字符串（字）。
<Ctrl+X>	删除光标左侧所有的字符。
<Ctrl+Y>	删除光标所在位置及其右侧所有的字符。
<Ctrl+Z>	返回到用户视图。
<Ctrl+]>	终止呼入的连接或重定向连接。
<Esc+B>	将光标向左移动一个字符串（字）。
<Esc+D>	删除光标右侧的一个字符串（字）。
<Esc+F>	将光标向右移动一个字符串（字）。
<Esc+N>	将光标向下移动一行。
<Esc+P>	将光标向上移动一行。
<Esc+<>	将光标所在位置指定为剪贴板的开始位置。
<Esc+>>	将光标所在位置指定为剪贴板的结束位置。

1.1.2.1.12 设置命令行别名

背景信息

命令行别名功能可以将设备中的命令行设置为用户自定义的字符串，方便用户使用。

别名命令主要有以下应用场景：

- 别名命令可以将命令或命令关键字配置为其他字符串作为原命令行的别名。用户需要输入该命令或命令关键字时，直接输入别名字符串即可，方便用户使用习惯。例如，用户可以将命令关键字**display**的别名定义为**show**。在使用时，可以直接使用别名**show**代替命令关键字**display**。
- 别名命令在为命令或命令关键字配置别名的同时，还可以改变用户输入参数的顺序。例如，用户配置带参数的别名**alias showif parameter $ifnum $iftype command "display interface $iftype $ifnum"** 用户可以输入字符串"**showif 1 Eth-Trunk**"，别名替换后的原命令字符串为**display interface Eth-Trunk 1** 。

用户可以通过命令**terminal command alias**打开当前会话的别名特性开关，也可以通过命令**undo terminal command alias**关闭当前会话的别名特性开关。关闭当前会话

的别名特性，仅影响当前会话的别名配置功能，并不清除系统中存在的别名配置信息。当继续执行打开当前会话的别名特性开关后，配置文件中的别名配置信息继续生效。用户可以通过命令**display terminal command alias**查看命令别名特性的开关状态。

操作步骤

步骤1 进入系统视图。

```
system-view
```

步骤2 进入命令别名视图。

```
command alias
```

步骤3 进行命令别名配置。

```
alias alias-string [ parameter parameter &<1-32> ] command command
```

步骤4 提交配置。

```
commit
```

----结束

检查配置结果

用户完成命令别名配置后，可以执行命令**display command alias**查看别名配置信息。

```
<HUAWEI> display command alias
  show = display
  showif $ifnum $iftype = display interface $iftype $ifnum
```

1.1.2.1.13 字符替换

背景信息

如果设备上的某个字符串或者某类字符串不满足用户需求，可以按如下步骤执行此功能对这类字符进行批量替换。该功能仅替换当前视图下的字符串。

此功能只能在两阶段生效模式下生效。

操作步骤

步骤1 进入系统视图。

```
system-view
```

步骤2 替换字符。

```
replace configuration pattern src-string with target-string
```

步骤3 提交配置。

```
commit
```

----结束

1.1.2.1.14 在系统视图下执行用户视图命令

背景信息

对于某些命令只能在用户视图下执行，当用户需要执行该类命令时，必须退出到用户视图才能成功执行。为了便于用户执行用户视图命令，在不用切换视图的情况下，通过本配置可实现在系统视图下执行用户视图命令。

操作步骤

步骤1 进入系统视图。

```
system-view
```

步骤2 在系统视图下执行用户视图命令。

```
run command-line
```

----结束

1.1.2.1.15 配置命令行时间戳

背景信息

当用户需要记录命令行的执行时间时，可配置显示系统时间戳或当前会话的命令时间戳。

操作步骤

- 配置显示系统时间戳。

  ```
  system-view
  timestamp enable
  commit
  ```

  说明

  该功能仅对display查询命令生效。

- 配置显示当前会话的命令行时间戳。

  ```
  terminal command timestamp
  ```

  说明

  - 使能该功能后，用户输入任意命令回车后都会显示执行时间。
  - 该功能仅对当前会话生效，用户退出系统重新登录后，该功能失效，需重新配置。
  - 如果用户执行命令**undo terminal command timestamp**关闭当前会话时间戳功能，但是已执行**timestamp enable**命令使能系统的时间戳功能，则用户执行display查询命令时依然会显示时间戳。

----结束

1.1.2.1.16 启用二次认证功能

背景信息

设备上有些命令，如果用户误操作会关联删除相关特性的配置，导致业务中断，造成用户网络中断。为了防止误操作，用户可通过执行命令行**configuration re-authentication enable**启用二次认证功能。

当二次认证功能启用后，执行这些命令时，需要先输入登录密码进行二次认证后该命令才能生效，命令范围：**reboot**、**reset saved-configuration**。

说明

- 为防止误操作导致某些业务不可用，建议使能二次认证功能。
- 缺省情况下，关闭执行危险命令时的二次认证功能。

操作步骤

步骤1 进入系统。

```
system-view
```

步骤2 启用二次认证功能。

```
configuration re-authentication enable
```

步骤3 提交配置。

```
commit
```

----结束

1.1.2.2 查看命令行显示信息

1.1.2.2.1 查询命令行的配置信息

在完成一系列配置后，可以执行相应的**display**命令查看设备的配置信息和运行信息。

例如，在完成SFTP服务器的各项配置后，可以执行命令**display ssh server-info**，查看当前设备作为SSH客户端情况下，与其相连的SSH服务器或者曾经连接过的SSH服务器与RSA公钥、ECC公钥等的绑定信息。**display**命令的用法和功能可详见各配置指南手册中对应特性的“检查配置结果”。

同时，系统支持查看当前生效的配置信息和当前视图下的配置信息，命令如下：

- 查看当前生效的配置信息：

  display current-configuration

  对于某些正在生效的配置参数，如果与缺省参数相同，则不显示。
- 查看当前视图下生效的配置信息：

  display this

  对于某些正在生效的配置参数，如果与缺省参数相同，则不显示。

说明

- 可以执行**timestamp enable**命令使能系统的时间戳功能，该功能使能后，用户执行**display**查询命令时，系统会在显示信息前加上当前执行的查询时间。
- 如果回显中某项参数的输出信息过长，则该项参数的输出信息只显示前面的一部分字符。

1.1.2.2.2 查询诊断信息

在系统出现故障或日常维护时，用户可以通过查看诊断信息收集系统当前各个模块的运行信息，用于故障定位。

```
display diagnostic-information [ level level_value ] [ module-name ] &<1-8> [ slot slot-id ] [ key key-string ] [ force ]
display diagnostic-information [ level level_value ] [ module-name ] &<1-8> [ slot slot-id ] [ key key-
```

string] [**force**] *file-name*
display diagnostic-information [**level** *level_value*] [*module-name*] &<1-8> *file-name* [**slot** *slot-id*] [**key** *key-string*] [**force**]

本命令集合了多条常用display命令的输出信息，包括**display clock**、**display version**、**display current-configuration**等等，可以看作是对系统常用display命令的批量执行。

1.1.2.2.3 控制命令行显示方式

所有的命令行有共同的显示特征，并且可以根据用户的需求，灵活控制显示方式。

- 当终端屏幕上显示的信息过多时，可以使用<PgUp>和<PgDn>显示上一页信息和下一页信息。
- 当执行某一命令后，如果显示的信息超过一屏时，系统会自动暂停，以方便用户查看。此时用户可以通过功能键控制命令行的显示方式，如表1-8所示。

  说明

  screen-length *screen-length* **temporary**命令可以用来设置当前终端屏幕的临时显示行数，如果*screen-length*取值为0则关闭分屏功能，即当显示的信息超过一屏时，系统不会自动暂停。

表 1-8 控制命令行显示方式

功能键	功能
键入<Ctrl+C>或<Ctrl+Z>	停止显示或命令执行。 **说明** 也可以键入除空格键、回车键等的其他键（可以是数字键或字母键）停止显示和命令执行。
键入空格键	继续显示下一屏信息。
键入回车键	继续显示下一行信息。

1.1.2.2.4 过滤命令行显示信息

显示特性

过滤命令行显示信息可以帮助用户迅速查找到所需要的信息，例如在执行**display**命令查看显示信息时，可以使用正则表达式（即指定显示规则）来过滤显示信息。

在一次显示信息超过一屏时，提供暂停功能，在暂停显示时用户可以有三种选择，如表1-9所示。

表 1-9 显示功能表

功能键	功能
键入空格键	继续显示下一屏信息。
键入回车键	继续显示下一行信息。

功能键	功能
键入+ *regular-expression*	功能等同于管道符| **include** *regular-expression*。
键入- *regular-expression*	功能等同于管道符| **exclude** *regular-expression*。
键入/ *regular-expression*	功能等同于管道符| **begin** *regular-expression*。
键入<Ctrl_C>和其他非以上键	停止显示和命令执行。

正则表达式

背景信息

正则表达式描述了一种字符串匹配的模式，由普通字符（例如字符a到z）和特殊字符（或称“元字符”）组成。正则表达式作为一个模板，将某个字符模式与所搜索的字符串进行匹配。

正则表达式一般具有以下功能：

- 检查字符串中符合某个规则的子字符串，并可以获取该子字符串。
- 根据匹配规则对字符串进行替换操作。

正则表达式由普通字符和特殊字符组成。

- 普通字符

  普通字符匹配的对象是普通字符本身。包括所有的大写和小写字母、数字、标点符号以及一些特殊符号。例如：a匹配abc中的a，10匹配10.113.25.155中的10，@匹配xxx@xxx.com中的@。
- 特殊字符

  特殊字符配合普通字符匹配复杂或特殊的字符串组合。**表1-10**是对特殊字符及其语法意义的使用描述。

  表 1-10 特殊字符及其语法意义描述

特殊字符	功能	举例
\	转义字符。将下一个字符（特殊字符或者普通字符）标记为普通字符。	*匹配*
^	匹配行首的位置。	^10匹配10.10.10.1，不匹配172.16.1.1
$	匹配行尾的位置。	1$匹配10.10.10.1，不匹配10.10.10.2

特殊字符	功能	举例
*	匹配前面的子正则表达式零次或多次。	10*可以匹配1、10、100、1000、…… (10)*可以匹配空、10、1010、101010、……
+	匹配前面的子正则表达式一次或多次。	10+可以匹配10、100、1000、…… (10)+可以匹配10、1010、101010、……
?	匹配前面的子正则表达式零次或一次。 **说明** 当前，在华为公司数据通信设备上通过命令行运用正则表达式输入?时，系统显示为命令行帮助功能。但是如果是分屏显示时，键入过滤显示规则加?，此时?的功能为正则表达式。	10?可以匹配1或者10 (10)?可以匹配空或者10
.	匹配任意单个字符。	a.b匹配任何一个以a开头，以b结尾含有三个字符的字符串 0.0可以匹配0x0、020、…… .oo.可以匹配book、look、tool、……
()	一对圆括号内的正则表达式作为一个子正则表达式，匹配子表达式并获取这一匹配。 如果圆括号中内容为空，则等价于空串。 如果模式串只有()，则可匹配任意字符串。 如果模式串中的右括号没有匹配的左括号，则右括号就作为普通字符。 如果模式串中的左括号没有匹配的右括号，则为非法模式串。	100(200)+可以匹配100200、100200200、…… (ab)匹配abcab ()匹配任意字符串 a()b匹配12ab12 a)b匹配za)bc a(b为非法模式串
_	匹配一个符号，包括逗号、左大括号、右大括号、左括号、右括号和空格，在表达式的开头或结尾时还可作起始符、结束符（同^，$）。	_65001_可以匹配20 65001 30、20 65001、65001 30、65001、……
x|y	匹配x或y。	100|200匹配100或者200 1(2|3)4匹配124或者134，而不匹配1234、14、1224、1334

特殊字符	功能	举例
[xyz]	匹配正则表达式中的任意一个字符。不可同时匹配多个字符，也不可匹配同一个字符多次。	[123]匹配255中的2 [abc]匹配字符“a”、“b”、“c”
[^xyz]	匹配字符串中非“x”、“y”、“z”的字符。只要字符串中有非“x”、“y”、“z”的字符，就能匹配到。	[^123]匹配除123之外的任何字符 [^abc]匹配除“a”、“b”、“c”之外的任何字符
[a-z]	匹配正则表达式指定范围内的任意一个字符。不可同时匹配多个字符，也不可匹配同一个字符多次。	[0-9]匹配指定范围内的任意数字 [a-z]匹配指定范围内的任意字母 [z-a]为非法模式串
[^a-d]	匹配字符串中除“a”、“b”、“c”、“d”以外的其他字符。只要字符串中有a～d范围外的字符，就能匹配到。	[^0-9]匹配所有非数字字符 [^a-z]匹配除字母以外的其他任意字符 [^z-a]为非法模式串

说明

除非特别说明，上表中涉及到的字符指的是可以打印的字符，包括字母、数字、空格及特殊符号等。

字符的使用

某些特殊字符如果处在如下的正则表达式的特殊位置时，会引起退化，成为普通字符。

- 特殊字符处在转义符号‘\’之后，则发生转义，变为匹配该字符本身。
- 特殊字符“*”、“+”，处于正则表达式的第一个字符位置。例如：+45匹配+45，abc(*def)匹配abc*def。
- 特殊字符“^”，不在正则表达式的第一个字符位置。例如：abc^匹配abc^。
- 特殊字符“$”，不在正则表达式的最后一个字符位置。例如：12$2匹配12$2。
- 右括号“)”或者“]”没有对应的左括号“(”或“[”。例：abc)匹配abc)，0-9]匹配0-9]。

说明

除非特别说明，以上正则表达式包括括号“（ ）”内包含的子正则表达式。

- 普通字符与特殊字符的组合使用

  实际应用中，往往不是一个普通字符加上一个特殊字符配合使用，而是由多个普通字符和特殊字符组合，匹配某些特征的字符串。

正则表达式的过滤方式

说明

按过滤条件进行查询时，显示内容的第一行信息中，以包含该字符串的整条信息作为起始，而非以过滤字符串作为起始。

系统支持使用**| count**显示使用过滤条件后输出的结果的行数，支持使用**| section**显示使用过滤条件后输出的结果段信息，支持使用**| ignore-case**表示匹配字符串时不区分大小写，还支持使用**| no-more**表示过滤条件后输出的结果不分屏。**| count**和**| section**方式既可以与下面三种过滤方式配合使用，也可以单独使用。但是**| ignore-case**和**| no-more**必须与下面三种过滤方式配合使用，不能单独使用。

在支持正则表达式的命令中，有三种过滤方式可供选择：

- **| begin** *regular-expression*：输出以匹配指定正则表达式的行开始的所有行。

  即过滤掉所有待输出字符串，直到出现指定的字符串（此字符串区分大小写）为止，其后的所有字符串都会显示到界面上。
- **| exclude** *regular-expression*：输出不匹配指定正则表达式的所有行。

  即待输出的字符串中没有包含指定的字符串（此字符串区分大小写），则会显示到界面上；否则过滤不显示。
- **| include** *regular-expression*：只输出匹配指定正则表达式的所有行。

  即待输出的字符串中如果包含指定的字符串（此字符串区分大小写），则会显示到界面上；否则过滤不显示。

说明

*regular-expression*为字符串形式，长度范围是1～255。

支持多级管道符对命令行进行筛选过滤。按照用户输入的先后顺序，上一级管道处理的输出作为下一级管道处理的输入，最多支持32级过滤。

| section仅用于显示结果中具有段信息的命令。如**display current-configuration**、**display this**等命令。

下面举例来说明在命令中指定过滤方式的用法。

例1：执行命令**display pm brief**，显示不匹配正则表达式“Directory|Files”的所有行，“Directory|Files”表示匹配“Directory”或“Files”。

```
<HUAWEI> display pm brief | exclude Directory|Files
Statistics Status                 : disable
Statistics Start Time              : -
Current Statistics Cycles          : -
Number of Statistics Tasks         : 0
Number of Statistics Objects       : 0
Number of Configured Pm Servers    : 0
```

例2：执行命令**display current-configuration**，只显示匹配正则表达式“vlan”的所有行。

```
<HUAWEI> display current-configuration | include vlan
vlan batch 7 10 18 to 19 30 60 66 70 77 100 105
vlan batch 200 1024
 port default vlan 77
 port default vlan 19
 port hybrid pvid vlan 10
 port hybrid untagged vlan 10
 port hybrid pvid vlan 60
 undo port hybrid vlan 1
 port hybrid tagged vlan 60
 port trunk allow-pass vlan 60
```



```
port hybrid pvid vlan 10
port hybrid tagged vlan 7
port hybrid untagged vlan 10
```

例3：执行命令**display current-configuration**，显示所有匹配正则表达式“vlan”的个数。

```
<HUAWEI> display current-configuration | include vlan | count
Total lines: 14.
```

说明

以上举例中的显示信息仅为示意。

系统支持使用| refresh周期刷新查询结果。使用**| refresh**设备每隔一段时间显示一次查询结果，默认查询间隔是5秒。

说明

- 命令刷新周期间隔过短，会引起CPU使用率上升，请尽量大的配置查询周期间隔。
- 如果设备剩余的VTY通道数小于3，不支持**| refresh**周期查询命令。
- 只有以**display**开头的查询命令支持使用**| refresh**周期查询功能。

设备还支持将**display**命令显示的结果重定向到指定的文件。有两种重定向方式可供选择：

- **>** *filename*

  将**display**命令显示的结果输出到指定的文件。如果目标文件已经存在，则覆盖该文件的原有内容。

- **>>** *filename*

  将**display**命令显示的结果追加到指定文件的末尾，原文件的内容仍保留。

1.1.2.3 配置会话日志功能开关

背景信息

全局会话日志功能默认关闭，用户需要将输入、设备屏幕输出以及设备执行用户输入命令的时间保存到根目录下sessionlog文件中时，可打开全局会话日志功能。

记录单个连接的会话日志功能默认打开，用户使能全局会话日志功能后，会将设备上所有连接的输入、设备屏幕输出等信息记录到日志文件中，如果某些连接不需要生成会话日志，可执行相应命令关闭指定连接的会话日志功能。

用户可以通过命令**display info-center session log status**查看设备上全局会话日志功能及所有在线连接的会话日志功能状态。

操作步骤

- 打开全局会话日志功能。

  ```
  system-view
  undo info-center session log disable
  commit
  ```

- 关闭当前连接的会话日志功能。

  ```
  system-view
  terminal session-log disable
  commit
  ```


说明

如果全局会话日志功能处于关闭状态，那么当前连接的会话日志功能也是关闭的，不需要执行以上操作。

----结束

1.1.3 登录设备命令行界面配置

说明

- 为充分保证设备安全，请用户定期修改密码。
- 管理员创建的帐号用户，首次登录必须强制修改初始口令。
- 系统的字符集编码固定为UTF-8，既支持中文输入又支持英文输入。使用终端登录设备时，请将终端上登录软件所使用的字符集编码设置为UTF-8，否则可能会出现中文显示为乱码的问题。
- 命令行帮助提示的字符串长度是字节数。1个英文字符1个字节，但是UTF-8编码中文字符是可变长的，导致字符串可以输入几个中文字符无法确定，输入超长时会提示错误。

1.1.3.1 登录设备命令行界面简介

设备作为服务器时，用户可以通过Console口、Telnet或STelnet方式登录本设备。设备作为客户端时，可以从本设备通过Telnet、STelnet来登录其他设备。

用户在本地或者从远端对设备进行管理与维护，需要配置用户界面、用户管理信息和终端服务来登录设备。其中用户界面提供登录入口，用户管理信息确保登录安全，终端服务则提供登录协议支持，比如远程登录Telnet协议或安全远程登录STelnet（Secure Telnet）协议。

用户可通过如表1-11所示几种方式登录服务器，对设备进行配置和管理。

表 1-11 用户登录方式

登录设备方式	优点	缺点	应用场景	说明
配置Console本地登录	使用专门的Console通信线缆连接，保证可以对设备有效控制。	不能远程登录维护设备。	• 当对设备进行第一次配置时，可以通过Console口登录设备进行配置。 • 当用户无法进行远程登录设备时，可通过Console口进行本地登录。 • 当设备无法启动时，可通过Console口进入BootLoader进行诊断或系统升级。	通过Console口进行本地登录是登录设备最基本的方式，也是其他登录方式的基础。 缺省情况下，用户可以直接通过Console口本地登录设备，命令访问级别是3。

登录设备方式	优点	缺点	应用场景	说明
配置Telnet远程登录	便于对设备进行远程管理和维护，不需要为每一台设备都连接一个终端，极大地方便了用户的操作。	传输过程采用TCP协议进行明文传输，存在安全隐患。	终端连接到网络上，使用Telnet方式登录设备，进行本地或远程的配置。应用在对安全性要求不高的网络。	缺省情况下，用户不能通过Telnet方式直接登录设备。如果需要通过Telnet方式登录设备，可以先通过Console口本地登录设备，并完成以下配置： • 确保终端和登录的设备之间路由可达（缺省情况下，设备上没有配置IP地址）。 • 配置Telnet服务器功能及参数。 • 配置Telnet用户登录的用户界面。
通过STelnet远程登录	STelnet协议实现在不安全网络上提供安全的远程登录，保证了数据的完整性和可靠性，保证了数据的安全传输。	配置较复杂。	如果网络对于安全性要求较高，可以通过STelnet方式登录设备。STelnet基于SSH（Secure Shell）协议，提供安全的信息保障和强大认证功能，保护设备不受IP欺骗等攻击。	缺省情况下，用户不能通过STelnet方式直接登录设备。如果需要通过STelnet方式登录设备，可以先通过Console口本地登录或Telnet远程登录设备，并完成以下配置： • 确保终端和登录的设备之间路由可达（缺省情况下，设备上没有配置IP地址）。 • 配置STelnet服务器功能及参数。 • 配置SSH用户登录的用户界面。 • 配置SSH用户。

Console 口概述

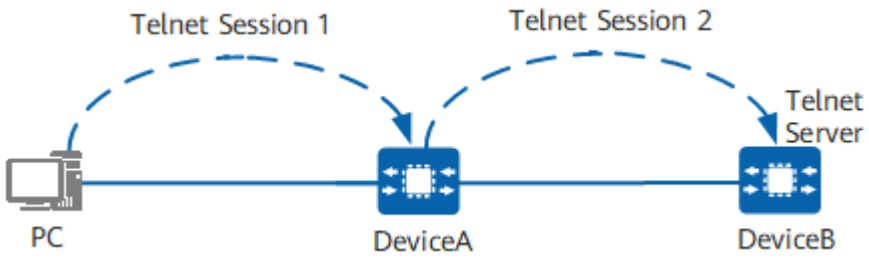
一块主控板提供一个Console口（接口类型为EIA/TIA-232 DCE）。通过将用户终端的串行接口与设备Console口直接连接，登录设备，实现对设备的本地配置。

Telnet 概述

Telnet协议在TCP/IP协议族中属于应用层协议，通过网络提供远程登录和虚拟终端功能。以服务器/客户端（Server/Client）模式工作，Telnet客户端向Telnet服务器发起请求，Telnet服务器提供Telnet服务。设备支持Telnet客户端和Telnet服务器功能。

如图1-7所示，DeviceA此时既作为Telnet服务器，也提供Telnet客户端服务。DeviceB对DeviceA提供Telnet服务器功能。

图 1-7 Telnet 连接示意图



STelnet 概述

Telnet传输过程采用TCP协议进行明文传输，缺少安全的认证方式，容易招致DoS（Denial of Service）、主机IP地址欺骗和路由欺骗等恶意攻击，存在很大的安全隐患。

相对于Telnet，STelnet基于SSH2.0协议，客户端和服务器端之间经过协商，建立安全连接，客户端可以像操作Telnet一样登录服务器端。

1.1.3.3 配置用户界面

1.1.3.3.1 了解用户界面

系统支持的用户界面有Console用户界面和VTY用户界面。

每个用户界面有对应的用户界面视图。用户界面（User-interface）视图是系统提供的一种命令行视图，用来配置和管理所有工作在异步交互方式下的物理接口和逻辑接口，从而达到统一管理各种用户界面的目的。

目前设备支持的用户界面

- Console（CON）

  控制口（Console Port）是一种通信串行口，由设备的主控板提供。

  一块主控板提供一个Console口，接口类型为EIA/TIA-232 DCE。用户终端的串行口可以与设备Console口直接连接，实现对设备的本地访问。

- VTY

  虚拟类型终端（Virtual Type Terminal）是一种虚拟线路端口。

  用户通过终端与设备建立Telnet或安全外壳SSH（Secure Shell）连接后，即建立了一条VTY，即用户可以通过VTY方式登录设备。最多支持21个用户同时通过VTY方式访问设备。

用户与用户界面的关系

用户界面与用户并没有固定的对应关系。用户界面的管理和监控对象是使用某种方式登录的用户，虽然单个用户界面某一时刻可能只有一个用户使用，但它并不针对某个用户。

用户登录时，系统会根据用户的登录方式，自动给用户分配一个当前空闲的、编号最小的某类型的用户界面，整个登录过程将受该用户界面视图下配置的约束。比如用户A使用Console口登录设备时，将受到Console用户界面视图下配置的约束，当使用VTY 1登录设备时，将受到VTY 1用户界面视图下配置的约束。同一用户登录的方式不同，分配的用户界面不同；同一用户登录的时间不同，分配的用户界面可能不同。

用户界面的编号

当用户登录设备时，系统会根据此用户的登录方式，自动分配一个当前空闲且编号最小的相应类型的用户界面给这个用户。用户界面的编号包括以下两种方式：

- 相对编号

  相对编号方式的形式是：用户界面类型＋编号。

  此种编号方式只能唯一指定某种类型的用户界面中的一个或一组，而不能跨类型操作。相对编号方式遵守的规则如下：

  - 控制口的编号：CON 0。
  - 虚拟线路的编号：第一个为VTY 0，第二个为VTY 1，依此类推。

- 绝对编号

  使用绝对编号方式，可以唯一的指定一个用户界面或一组用户界面。使用**display user-interface**（不带参数）可查看到设备当前支持的用户界面以及它们的绝对编号。

  Console口只有一个，但VTY类型的用户界面有21个，可以在系统视图下使用**user-interface maximum-vty**命令设置最大用户界面个数。

  缺省情况下，Console、VTY用户接口在系统中的绝对编号，如**表1-12**所示。

表 1-12 用户界面的相对、绝对编号说明

用户界面	说明	绝对编号	相对编号
Console用户界面	用来管理和监控通过Console口登录的用户。	0～19	0
VTY用户界面	用来管理和监控通过Telnet或STelnet方式登录的用户。	34～54	第一个为VTY 0，第二个为VTY 1，依此类推。缺省存在VTY 0～4通道。 绝对编号34～54对应相对编号VTY 0～VTY 20。

用户界面的用户验证

配置用户界面的用户验证方式后，用户登录设备时，系统对用户的身份进行验证。

对用户的验证有如下几种方式：

- Password验证：只需要口令，不需要用户名。
- AAA验证：需要用户提供用户名和口令，对Telnet用户一般采用AAA验证。

用户界面的用户级别

系统支持对登录用户进行分级管理。用户所能访问命令的级别由用户的级别决定。

- 如果对用户采用Password验证，登录到设备的用户所能访问的命令级别由登录时的用户界面级别决定。
- 如果对用户采用AAA验证，登录到设备的用户所能访问的命令级别由AAA配置信息中本地用户的级别决定。

1.1.3.3.2 配置 Console 用户界面

前提条件

当用户通过Console口登录设备实现本地维护时，可以根据使用需求或对设备安全的考虑，配置相应的Console用户界面属性。

在配置Console用户界面之前，需要完成以下任务：

- 通过终端可以登录设备。

操作步骤

表 1-13 配置 Console 用户界面的物理属性

操作步骤	命令	说明
进入系统视图	**system-view**	-
进入Console用户界面视图	**user-interface console** *interface-number*	-
设置传输速率	**speed** *speed-value*	缺省情况下，传输速率为9600bit/s。
设置流控方式	**flow-control** { **hardware** | **none** | **software** }	缺省情况下，流控方式为none。
设置校验位	**parity** { **even** | **mark** | **none** | **odd** | **space** }	缺省情况下，校验位为none。
设置停止位	**stopbits** { **1.5** | **1** | **2** }	缺省情况下，停止位为1位。
设置数据位	**databits** { **5** | **6** | **7** | **8** }	缺省情况下，数据位为8位。
退出Console用户界面视图	**quit**	-
退出系统视图到用户视图	**quit**	-
提交配置	**commit**	-

表 1-14 配置 Console 用户界面的终端属性

操作步骤	命令	说明
进入系统视图	**system-view**	-
进入Console用户界面视图	**user-interface console** *interface-number*	-
设置用户超时断连功能	**idle-timeout** *minutes* [*seconds*]	在设定的时间内，如果连接始终处于空闲状态，系统将自动断开该连接。 缺省情况下，Console用户界面断连的超时时间为5分钟。 **说明** 设置用户连接的超时时间过长或者为0会导致终端一直处于登录状态，存在安全风险，建议用户执行命令**lock**锁定当前连接。
设置终端屏幕每屏显示的行数	**screen-length** *screen-length* [**temporary**]	使用参数**temporary**可以指定终端屏幕的临时显示行数。 缺省情况下，终端屏幕显示的行数为24行。
设置历史命令缓冲区大小	**history-command max-size** *size-value*	缺省情况下，用户界面历史命令缓冲区大小为10条历史命令。
退出Console用户界面视图	**quit**	-
退出系统视图到用户视图	**quit**	-
提交配置	**commit**	-

表 1-15 配置 Console 用户界面的用户级别

操作步骤	命令	说明
进入系统视图	**system-view**	-
进入Console用户界面视图	**user-interface console** *interface-number*	-

操作步骤	命令	说明
设置用户级别	**user privilege level** *level*	用户级别和命令级别对应关系如表1-3所示。 **说明** • 缺省情况下，Console口用户界面对应的默认命令访问级别是3。 • 如果用户界面下配置的命令级别访问权限与用户名本身对应的操作权限冲突，以用户名本身对应的级别为准。
退出Console用户界面视图	**quit**	-
退出系统视图到用户视图	**quit**	-
提交配置	**commit**	-

表 1-16 配置 Console 用户界面的 AAA 验证方式

操作步骤	命令	说明
进入系统视图	**system-view**	-
进入Console用户界面视图	**user-interface console** *interface-number*	-
设置用户验证方式为AAA验证	**authentication-mode aaa**	-
退出Console用户界面视图	**quit**	-
进入AAA视图	**aaa**	-
配置本地用户名和密码	**local-user** *user-name* **password** [**cipher** | **irreversible-cipher**] *password*	为充分保证设备安全，请用户定期修改密码。
配置本地用户的接入类型为Console	**local-user** *user-name* **service-type terminal**	-
退出AAA视图	**quit**	-
退出系统视图到用户视图	**quit**	-
提交配置	**commit**	-

表 1-17 配置 Console 用户界面的 Password 验证方式

操作步骤	命令	说明
进入系统视图	**system-view**	-
进入Console用户界面视图	**user-interface console** *interface-number*	-
设置用户验证方式为密码验证	**authentication-mode password**	-
设置验证密码	**set authentication password** [**cipher** *password*]	输入的密码可以是显式或者密文，当不指定**cipher** *password*参数时，将采用交互方式输入显式密码，当指定**cipher** *password*参数时，既可以输入显式密码也可以输入密文密码，但都将以密文形式保存在配置文件中。为充分保证设备安全，请用户定期修改密码。 **说明** 开启弱密码字典维护功能后，弱密码字典中定义的密码（可以通过命令**display security weak-password-dictionary**查看）不能在该命令中配置。
退出Console用户界面视图	**quit**	-
退出系统视图到用户视图	**quit**	-
提交配置	**commit**	-

表 1-18 配置 Console 用户界面关闭

操作步骤	命令	说明
进入系统视图	**system-view**	-
进入Console用户界面视图	**user-interface console** *interface-number*	-
关闭Console用户界面视图	**shutdown**	缺省情况下，Console用户界面处于正常使能状态。
提交配置	**commit**	-

1.1.3.3.3 配置 VTY 用户界面

前提条件

当用户通过Telnet或STelnet方式登录设备实现本地或远程维护时，可以根据用户使用需求以及对设备安全的考虑，配置VTY用户界面。

在配置VTY用户界面之前，需要完成以下任务：

- 通过终端可以登录设备。

操作步骤

表 1-19 配置 VTY 用户界面的终端属性

操作步骤	命令	说明
进入系统视图	**system-view**	-
配置VTY用户界面的最大个数	**user-interface maximum-vty** *number*	缺省情况下，VTY用户界面的最大个数为21。 如果配置的VTY类型用户界面的最大个数小于当前的最大个数，不会影响当前在线用户，也不需要其他配置。 如果要配置的VTY类型用户界面的最大个数大于当前最多可以登录用户的数量，就必须为新增加的用户界面配置验证方式。 当配置VTY用户界面最大个数为0时，任何用户都无法通过VTY登录到设备。
配置可用VTY通道数的超限告警阈值	**user-interface vty available-vty-threshold** *threshold-value*	缺省情况下，可用VTY通道数的超限告警阈值为4。 当系统上可用的VTY数目小于设定的阈值时，设备上报告警；当系统上可用的VTY数目等于设定的阈值时，不产生告警，同时也不清除告警；当系统上可用的VTY数目大于设定的阈值时，告警清除。
进入VTY用户界面视图	**user-interface vty** *first-ui-number* [*last-ui-number*]	-
启用VTY终端服务	**shell**	缺省情况下，所有VTY终端服务已启动。

操作步骤	命令	说明
设置用户超时断连功能	**idle-timeout** *minutes* [*seconds*]	在设定的时间内，如果连接始终处于空闲状态，系统将自动断开该连接。 缺省情况下，VTY用户界面断连的超时时间为10分钟。 **说明** 设置用户连接的超时时间过长或者为0会导致终端一直处于登录状态，存在安全风险，建议用户执行命令**lock**锁定当前连接。
设置终端屏幕每屏显示的行数	**screen-length** *screen-length* [**temporary**]	使用参数**temporary**设置的行数只对当前活动用户界面有效，用户退出后不保存设置。 缺省情况下，终端屏幕显示的行数为24行。
设置历史命令缓冲区大小	**history-command max-size** *size-value*	缺省情况下，用户界面历史命令缓冲区大小为10条历史命令。
退出VTY用户界面视图	**quit**	-
退出系统视图到用户视图	**quit**	-
提交配置	**commit**	-

表 1-20 配置 VTY 用户界面的用户级别

操作步骤	命令	说明
进入系统视图	**system-view**	-
进入VTY用户界面视图	**user-interface vty** *first-ui-number* [*last-ui-number*]	-
设置用户级别	**user privilege level** *level*	缺省情况下，VTY用户界面的用户级别是0。 如果用户界面下配置的命令级别访问权限与用户名本身对应的操作权限冲突，以用户名本身对应的级别为准。
退出VTY用户界面视图	**quit**	-

操作步骤	命令	说明
退出系统视图到用户视图	**quit**	-
提交配置	**commit**	-

表 1-21 配置 VTY 用户界面的 AAA 验证方式

操作步骤	命令	说明
进入系统视图	**system-view**	-
进入VTY用户界面视图	**user-interface vty** *first-ui-number* [*last-ui-number*]	-
设置用户验证方式为AAA验证	**authentication-mode aaa**	-
退出VTY用户界面视图	**quit**	-
进入AAA视图	**aaa**	-
配置本地用户名和密码	**local-user** *user-name* **password** [**cipher** | **irreversible-cipher**] *password*	为充分保证设备安全，请用户定期修改密码。
配置本地用户的接入类型为Telnet或SSH	**local-user** *user-name* **service-type** { **telnet** | **ssh** }	Telnet协议本身有安全风险，建议使用SSH v2安全协议。
退出AAA视图	**quit**	-
退出系统视图到用户视图	**quit**	-
提交配置	**commit**	-

表 1-22 配置 VTY 用户界面的 Password 验证方式

操作步骤	命令	说明
进入系统视图	**system-view**	-
进入VTY用户界面视图	**user-interface vty** *first-ui-number* [*last-ui-number*]	-
设置用户验证方式为密码验证	**authentication-mode password**	-

操作步骤	命令	说明
设置验证密码	**set authentication password** [**cipher** *password*]	输入的密码可以是显式或者密文，当不指定**cipher** *password*参数时，将采用交互方式输入显式密码，当指定**cipher** *password*参数时，既可以输入显式密码也可以输入密文密码，但都将以密文形式保存在配置文件中。为充分保证设备安全，请用户定期修改密码。 **说明** 开启弱密码字典维护功能后，弱密码字典中定义的密码（可以通过命令**display security weak-password-dictionary**查看）不能在该命令中配置。
退出VTY用户界面视图	**quit**	-
退出系统视图到用户视图	**quit**	-
提交配置	**commit**	-

表 1-23 配置 VTY 用户界面的扩展功能

操作步骤	命令	说明
进入系统视图	**system-view**	-
使能VTY用户界面的安全策略	**undo user-interface vty security-policy disable**	缺省情况下，VTY用户界面的安全策略使能。
进入VTY用户界面视图	**user-interface vty** *first-ui-number* [*last-ui-number*]	-

操作步骤	命令	说明

配置VTY类型用户界面的基于ACL的登录限制	**acl** [**ipv6**] { *acl-number* | *acl-name* } { **inbound** | **outbound** }	• 当需要限制某个地址或地址段的用户登录到设备时，使用**inbound**。
		• 当需要限制已经登录的用户登录到其他设备时，使用**outbound**。
		说明
		• 用户界面支持基本访问控制列表（2000～2999）和高级访问控制列表（3000～3999）。
		• 当ACL的rule配置为**permit**时，来自其他设备的报文匹配该规则时：
		  • 如果该ACL应用在**inbound**方向，则允许其他设备访问本设备。
		  • 如果该ACL应用在**outbound**方向，则允许本设备访问其他设备。
		• 当ACL的rule配置为**deny**时，来自其他设备的报文匹配该规则时：
		  • 如果该ACL应用在**inbound**方向，则拒绝其他设备访问本设备。
		  • 如果该ACL应用在**outbound**方向，则拒绝本设备访问其他设备。
		• 当ACL配置了rule，但来自其他设备的报文没有匹配该规则时：
		  • 如果该ACL应用在**inbound**方向，则拒绝其他设备访问本设备。
		  • 如果该ACL应用在**outbound**方向，则拒绝本设备访问其他设备。
		• 当ACL未配置rule时：
		  • 如果该ACL应用在**inbound**方向，则允许任何其他设备访问本设备。
		  • 如果该ACL应用在**outbound**方向，则允许本设备访问任何其他设备。
		• 关于ACL配置的更多内容，请参见《配置指南-IP地址与服务配置》中的“ACL配置”。

退出VTY用户界面视图	**quit**	-
退出系统视图到用户视图	**quit**	-
提交配置	**commit**	-

1.1.3.3.4 检查配置结果

操作步骤

- 执行命令**display users** [**all**]，查看用户界面的使用信息。
- 执行命令**display user-interface console** *ui-number* [**summary**]，查看Console用户界面信息。
- 执行命令**display user-interface maximum-vty**，查看VTY类型用户界面的最大个数。
- 执行命令**display user-interface vty** *ui-number1* [**summary**]，查看VTY用户界面信息。
- 执行命令**display ssh server ip-block all**，查看所有认证失败的客户端IP地址。
- 执行命令**display ssh server ip-block list**，查看因认证失败而被锁定的客户端IP地址。
- 执行命令**display vty ip-block list**，查看因为认证失败而被阻止的IP地址列表。
- 执行命令**display vty ip-block all**，查看所有认证失败的IP地址。
- 执行命令**display vty mode**，查看VTY的工作模式。

----结束

1.1.3.4 配置 Console 本地登录

1.1.3.4.1 配置用户通过 Console 口登录设备

前提条件

在配置用户通过Console口登录设备之前，需完成以下任务：

- 准备好Console通信电缆。
- PC端准备好终端仿真软件。

  说明

  如果系统不带终端仿真软件，请您准备第三方终端仿真软件，使用方法请参照该软件的使用指导或联机帮助。

缺省配置

表 1-24 设备 Console 口缺省配置

参数	缺省值
传输速率	9600bit/s
流控方式	不进行流控
校验方式	不进行校验
停止位	1
数据位	8

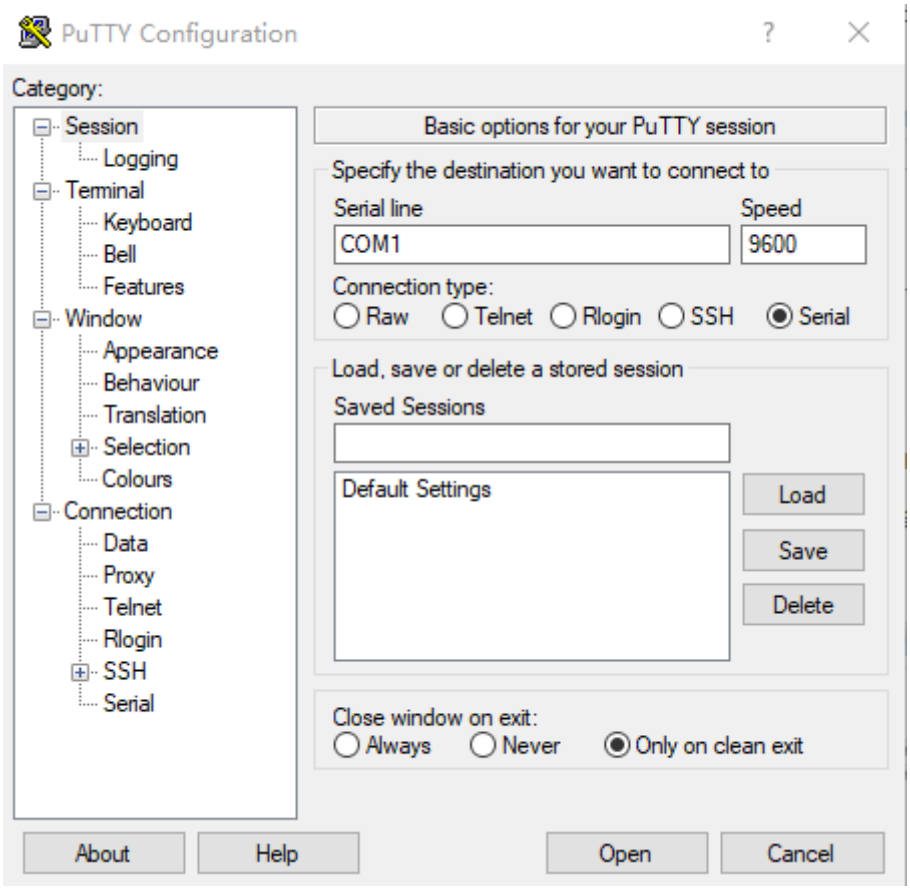
操作步骤

步骤1 将Console通信电缆的DB9（孔）插头插入PC机的串口（COM）中，再将RJ-45插头端插入设备的Console口中。

步骤2 在PC上打开终端仿真软件，新建连接，设置连接的接口以及通信参数。（此处使用第三方软件PuTTY为例进行介绍）

1. 如图1-8所示，单击“Session”，新建连接。

图 1-8 新建连接



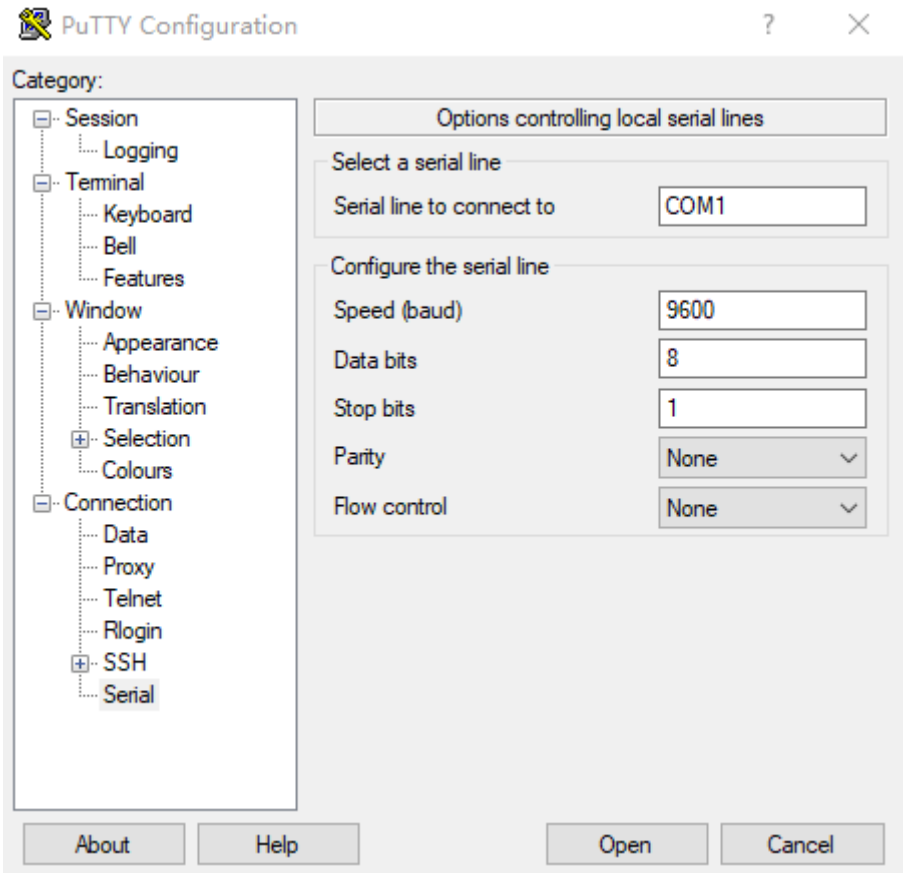
2. 如图1-9所示，单击“Serial”，设置连接的接口以及通信参数。
   a. 连接的接口请根据实际情况进行选择。例如，在Windows系统中，可以通过在“设备管理器”中查看端口信息，选择连接的接口。
   b. 设置终端软件的通信参数需与设备的缺省值保持一致，分别为：传输速率为9600bit/s、8位数据位、1位停止位、无校验和无流控。
   c. 单击“Open”。

说明

因为PC端可能会存在多个连接接口，这里需要选择的是连接Console线缆的那个接口。一般情况下，选择的接口是COM1。

若修改了设备的串口通信参数值，需要在PC端更换通信参数值与设备的串口通信参数值一致后，重新连接。

图 1-9 设置连接的接口以及通信参数



步骤3 按Enter键，直到系统出现如下显示，提示用户输入密码。（AAA认证时，提示输入用户名和密码，以下显示信息仅为示意）

```
Login authentication

Password:
```

进入设备后，用户可以键入命令，对设备进行配置，需要帮助可以随时键入“?”。

----结束

检查配置结果

- 执行**display users** [**all**]命令，查看用户界面的用户登录信息。
- 执行**display user-interface console 0**命令，查看用户界面信息。
- 执行**display local-user**命令，查看本地用户的属性信息。
- 执行**display access-user**命令，查看在线连接的用户信息。

1.1.3.4.2 举例：配置用户通过 Console 口登录设备

组网需求

当用户无法进行远程登录设备时，可通过Console口进行本地登录。使用Console口登录设备时需要密码认证，为了防止非法用户登录设备，修改Console用户界面的认证方式为AAA认证。

图 1-10 配置通过 Console 口登录设备组网图

配置思路

采用如下的思路配置通过Console口登录设备：

1. 使用终端仿真软件通过Console口登录设备。
2. 配置Console用户界面的认证方式。

说明

如果系统不带终端仿真软件，请您准备第三方终端仿真软件，使用方法请参照该软件的使用指导或联机帮助。

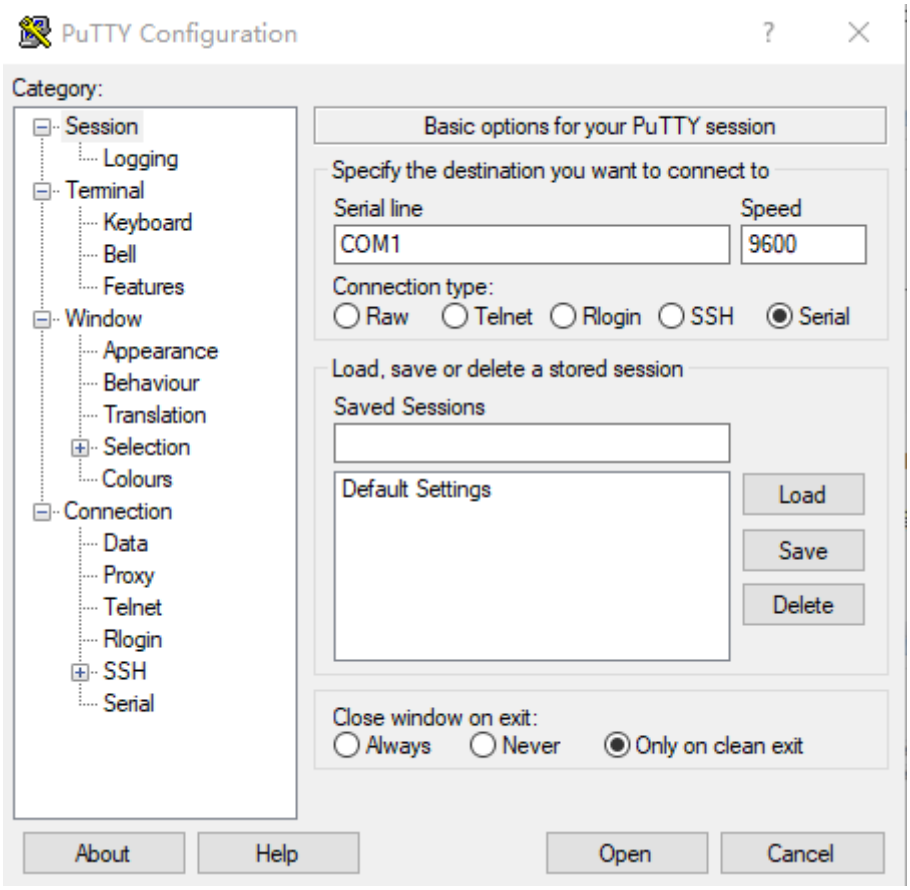
操作步骤

步骤1 将Console通信电缆的DB9（孔）插头插入PC机的串口（COM）中，再将RJ-45插头端插入设备的Console口中。

步骤2 在PC上打开终端仿真软件，新建连接，设置连接的接口以及通信参数。（此处使用第三方软件PuTTY为例进行介绍）

1. 如图1-11所示，单击“Session”，新建连接。

图 1-11 新建连接



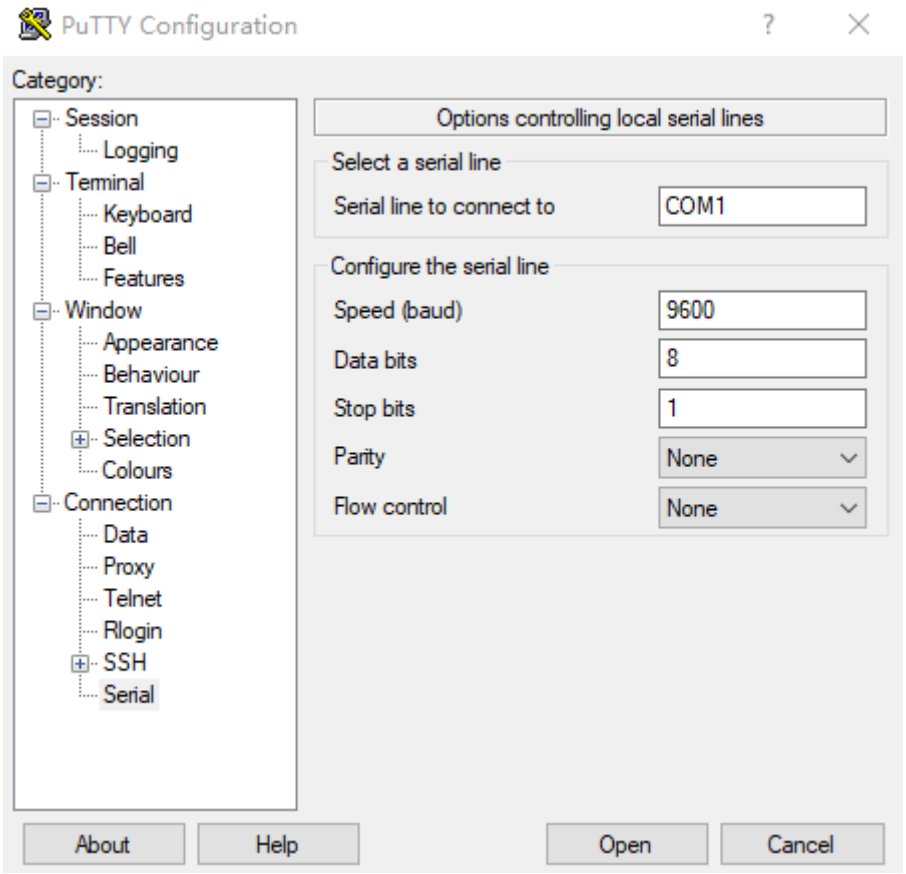
2. 如图1-12所示，单击“Serial”，设置连接的接口以及通信参数。
   a. 连接的接口请根据实际情况进行选择。例如，在Windows系统中，可以通过在“设备管理器”中查看端口信息，选择连接的接口。
   b. 设置终端软件的通信参数需与设备的缺省值保持一致，分别为：传输速率为9600bit/s、8位数据位、1位停止位、无校验和无流控。
   c. 单击“Open”。

   说明

   因为PC端可能会存在多个连接接口，这里需要选择的是连接Console线缆的那个接口。一般情况下，选择的接口是COM1。

   若修改了设备的串口通信参数值，需要在PC端更换通信参数值与设备的串口通信参数值一致后，重新连接。

图 1-12 设置连接的接口以及通信参数



步骤3 按Enter键，直到系统出现如下显示，提示用户输入密码。（AAA认证时，提示输入用户名和密码，以下显示信息仅为示意）

```
Login authentication


Password:
```

进入设备后，用户可以键入命令，对设备进行配置，需要帮助可以随时键入“?”。

步骤4 配置Console用户界面的认证方式

```
<HUAWEI> system-view
[~HUAWEI] user-interface console 0
```



```
[~HUAWEI-ui-console0] authentication-mode aaa
[*HUAWEI-ui-console0] user privilege level 3
[*HUAWEI-ui-console0] quit
[*HUAWEI] aaa
[*HUAWEI-aaa] local-user admin1234 password
Please configure the password (8-128)
Enter Password:
Confirm Password:
[~Device-aaa] local-user admin1234 level 3
[*HUAWEI-aaa] local-user admin1234 service-type terminal
[*HUAWEI-aaa] commit
```

----结束

检查配置结果

执行以上操作后，用户使用Console用户界面重新登录设备时，需要输入用户名admin1234，认证密码YsHsjx_202206才能通过身份验证，成功登录设备。

```
Username:admin1234
Password:
<HUAWEI>
```

配置脚本

```
#
aaa
 local-user admin1234 password irreversible-cipher $1d$g8wLJ`LjL!$CyE(V{3qg5DdU:PM[6=6O
$UF-.fQ,Q}>^)OBzgoU$
 local-user admin1234 service-type terminal
 local-user admin1234 level 3
#
user-interface con 0
 authentication-mode aaa
#
return
```

1.1.3.5 配置 Telnet 远程登录

1.1.3.5.1 配置用户通过 Telnet 登录设备

前提条件

在配置用户通过Telnet登录设备之前，需完成以下任务：

- 终端与设备之间路由可达。
- 已执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

背景信息

说明

使用Telnet协议存在安全风险，建议使用STelnet V2登录设备。

FIPS模式下，不支持使用Telnet协议。

在配置Telnet登录设备任务中，配置流程如**表1-25**所示。

表 1-25 用户通过 Telnet 登录设备配置流程

序号	配置任务名称	配置任务说明	配置流程说明
1	配置Telnet服务器功能及参数	包括使能Telnet服务器功能，配置Telnet服务器参数。	任务1、2、3之间没有严格的配置顺序。
2	配置Telnet登录的用户界面	包括VTY用户界面的用户优先级、用户认证方式、支持Telnet协议及其他基本属性。	
3	配置Telnet类型的本地用户	包括Telnet用户名和密码、服务类型和用户级别等。	
4	从终端通过Telnet登录设备	从终端通过Telnet客户端软件登录设备。	-

缺省配置

表 1-26 配置用户通过 Telnet 登录设备的缺省值

参数	缺省值
Telnet服务器功能	开启
Telnet服务器端口号	23
VTY用户界面的认证方式	没有配置认证方式
VTY用户界面所支持的协议	支持所有协议类型
用户级别	VTY用户界面对应的默认命令访问级别是0

操作步骤

- **配置Telnet服务器功能及参数**

  用户终端建立与设备的Telnet连接之前，需要首先确保设备的Telnet服务功能已经使能。

  表 1-27 配置 Telnet 服务器功能及参数

操作步骤	命令	说明
进入系统视图	**system-view**	-

操作步骤	命令	说明
使能Telnet服务器功能	**telnet** [**ipv6**] **server enable**	缺省情况下，在使用设备出厂缺省配置文件时，Telnet服务处于去使能状态。若设备使用的配置文件中没有配置**undo telnet** [**ipv6**] **server disable**，Telnet IPv4服务是处于使能状态，Telnet IPv6服务处于关闭状态。
（可选）配置Telnet服务器的端口号	**telnet** [**ipv6**] **server port** *port-number*	缺省情况下，端口号是23。 配置Telnet服务器的端口号，使攻击者无法获知更改后的Telnet服务器端口号，有效防止了攻击者对Telnet服务标准端口的登录。
（可选）配置访问控制列表	**telnet** [**ipv6**] **server acl** { *acl-number* | *acl-name* }	缺省情况下，没有配置访问控制列表。 配置了访问控制列表，可控制哪些客户端能以Telnet方式访问本设备。
配置Telnet服务器的源接口	• **telnet server-source -i** { *interface-type interface-number* | *interface-name* } • **telnet ipv6 server-source -a** *ipv6-address* [**-vpn-instance** *vpn-instance-name*] • **telnet** [**ipv6**] **server-source all-interface**	缺省情况下，未指定Telnet服务器端的源接口。 **说明** 如果指定的源接口是LoopBack接口，该LoopBack接口必须已经成功创建，否则会导致本配置无法成功执行。
（可选）配置Telnet协议的白名单Session-CAR	**whitelist session-car telnet-server** { **cir** *cir-value* | **cbs** *cbs-value* | **pir** *pir-value* | **pbs** *pbs-value* }*	缺省情况下，Telnet协议的白名单Session-CAR带宽的承诺信息速率为4kbit/s，承诺突发尺寸为640bytes，峰值信息速率为64kbit/s，峰值突发尺寸为10000bytes。 可以通过执行命令**whitelist session-car telnet-server disable**，去使能Telnet协议的白名单Session-CAR功能。一般情况下不建议关闭该功能。

操作步骤	命令	说明
（可选）使能Telnet服务器上的客户端IP地址锁定功能	**undo telnet server ip-block disable**	缺省情况下，Telnet服务器上的客户端IP地址锁定功能使能。 如果用户在5分钟内连续6次认证失败，则IP地址将会被锁定5分钟，可以通过执行命令**activate vty ip-block ip-address** *ip-address* [**vpnname** *vpn-name*]提前对被锁定的IP地址进行解锁。
（可选）配置在一定时间内通过Telnet登录服务器失败次数的告警上报门限和告警恢复门限	**telnet server login-failed threshold-alarm upper-limit** *report-times* **lower-limit** *resume-times* **period** *period-time*	缺省情况下，在5分钟内发生30次或30次以上次数登录失败，产生告警；在5分钟内登录失败次数小于20，取消告警。
（可选）配置单个IP地址通过Telnet登录服务器的最大连接数。	**telnet server ip-limit-session** *limit-session-num*	缺省情况下，单个IP地址通过Telnet登录服务器的最大连接数是64。
（可选）配置Telnet协议报文的DSCP优先级	**telnet server dscp** *value*	缺省情况下，Telnet协议报文的DSCP优先级值为48。
退出系统视图到用户视图	**quit**	-
提交配置	**commit**	-

- **配置Telnet登录的用户界面**

  配置VTY用户界面的用户级别及其他基本属性。

  表 1-28 配置 Telnet 登录的用户界面

操作步骤	命令	说明
进入系统视图	**system-view**	-
进入VTY用户界面视图	**user-interface vty** *first-ui-number* [*last-ui-number*]	-

操作步骤	命令	说明
配置用户界面的用户级别	**user privilege level** *level*	缺省情况下，VTY用户界面的用户级别是0。 如果用户希望操作较高级别的命令，则需要配置较高的用户级别。 如果用户界面下配置的级别访问权限与用户名本身对应的操作权限冲突，以用户名本身对应的级别为准。
配置用户验证方式	**authentication-mode** { **password** | **aaa** }	设备提供的验证方式有密码验证和AAA验证。用户可根据需要任意选择一种方式。 ● 选择密码验证后，需要执行**set authentication password**，设置本地验证的密码。 ● 选择AAA验证后，请参考**配置Telnet类型的本地用户（AAA验证方式）** 关于验证方式，请参见**1.1.3.3.3 配置VTY用户界面**，推荐使用AAA验证方式。
配置VTY用户界面支持Telnet协议	**protocol inbound** { **all** | **telnet** }	缺省情况下，用户界面支持所有协议类型。
（可选）配置用户界面的其他属性	具体配置请参见**1.1.3.3.3 配置VTY用户界面**	VTY用户界面的其他属性在设备上都有缺省值，用户一般不需要另外配置。但是可以根据用户使用需求，选择配置相关属性。
退出VTY用户界面视图	**quit**	-
退出系统视图到用户视图	**quit**	-
提交配置	**commit**	-

- **配置Telnet类型的本地用户（AAA验证方式）**

  配置管理员的用户名和密码，以保证只有管理员才能登录设备。

表 1-29 配置 Telnet 类型的本地用户（AAA 验证方式）

操作步骤	命令	说明
进入系统视图	**system-view**	-
进入AAA视图	**aaa**	-
配置本地用户名和密码	**local-user** *user-name* **password irreversible-cipher** *password* **local-user** *user-name* **password** [**cipher** | **irreversible-cipher**] *password*	为充分保证设备安全，请用户定期修改密码。
配置本地用户的服务类型	**local-user** *user-name* **service-type telnet**	-
配置本地用户的级别	**local-user** *user-name* **level** *level*	不同级别的用户登录后，只能使用等于或低于当前用户级别的命令，从而保证了设备的安全性。 如果用户界面下配置的级别访问权限与用户名本身对应的操作权限冲突，以用户名本身对应的级别为准。
提交配置	**commit**	-

- **从终端通过Telnet登录设备**

  从终端通过Telnet登录设备，可以选择使用Windows命令行提示符或第三方软件，此处以Windows的命令行提示符为例。

  请在终端上进行以下操作：

  a. 进入Windows的命令行提示符。

  b. 执行Windows命令**telnet** *ip-address port*，通过Telnet方式登录设备。

  ```
  C:\Documents and Settings\Administrator> telnet 10.137.217.177 1025
  ```

  c. 按Enter键，输入AAA验证方式配置的登录用户名和密码，验证通过后，出现用户视图的命令行提示符，至此用户成功登录设备。（以下显示信息仅为示意）

  ```
  Username:admin1234
  Password:
  Info: The max number of VTY users is 8, the number of current VTY users online is 1, and total number of terminal users online is 1.
  <Telnet Server>
  ```

----结束

检查配置结果

- 执行**display users** [**all**]命令，查看用户界面连接情况。

- 执行**display tcp status**命令，查看当前建立的所有TCP连接情况。
- 执行**display telnet server status**命令，查看Telnet服务器的当前连接信息。
- 执行**display vty ip-block list**命令，查看因为认证失败而被阻止的IP地址列表。
- 执行**display vty ip-block all**命令，查看所有认证失败的IP地址。

1.1.3.5.2 配置设备作为 Telnet 客户端登录其他设备

前提条件

在配置设备作为Telnet客户端登录其他设备之前，需完成以下任务：

- 已从终端成功登录到本设备。
- 本设备与Telnet服务器之间路由可达。
- Telnet服务器端已开启Telnet服务器功能。
- 获取Telnet服务器端配置的Telnet用户信息和端口号。
- 已执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

背景信息

说明

使用Telnet协议存在安全风险，建议使用STelnet V2登录设备。

FIPS模式下，不支持使用Telnet协议。

在配置设备作为Telnet客户端登录其他设备任务中，配置流程如**表1-30**所示。

表 1-30 设备作为 Telnet 客户端登录其他设备的配置流程

序号	配置任务名称	配置任务说明	配置流程说明
1	**（可选）配置Telnet客户端参数**	包括配置客户端源地址和客户端DSCP优先级等。	-
2	**使用Telnet命令登录其他设备**	设备通过Telnet命令登录Telnet服务器。	

操作步骤

1. **（可选）配置Telnet客户端参数**

表 1-31 （可选）配置 Telnet 客户端参数

操作步骤	命令	说明
进入系统视图	**system-view**	-

操作步骤	命令	说明
配置Telnet客户端的源地址信息	**telnet client-source** { **-a** *source-ip-address* | **-i** *interface-type interface-number* } **telnet ipv6 client-source -a** *source-ipv6-address* [**-vpn-instance** *ipv6-vpn-instance-name*]	缺省情况下，Telnet客户端的IPv4源地址为0.0.0.0，IPv6源地址为::。 配置了Telnet客户端源地址信息后，在服务器端显示的Telnet客户端的源地址信息与该步骤中配置的一致。
配置Telnet协议报文的DSCP优先级。	**telnet client dscp** *value*	缺省情况下，Telnet协议报文的DSCP优先级值为48。
退出系统视图到用户视图	**quit**	-
提交配置	**commit**	-

2. **使用Telnet命令登录其他设备**

表 1-32 使用 Telnet 命令登录其他设备

操作步骤	命令	说明
以Telnet方式通过IPv4的地址登录到服务器上	**telnet** [**-i** { *interface-type interface-number* | *interface-name* } | [**vpn-instance** *vpn-instance-name*] [**-a** *source-ip-address*]] *host-ip-address* [*port-number*]	根据网络地址类型不同，选择其中一种。 只有当服务器正在尝试连接的端口号是23时，Telnet客户端登录时可以不指定端口号，否则如果是其他端口号，Telnet客户端登录时必须指定端口号。
以Telnet方式通过IPv6的地址登录到服务器上	**telnet ipv6** [**-a** *source-ipv6-address*] [**public-net** | **vpn-instance** *ipv6-vpn-name*] *ipv6-address* [**-oi** { *interface-type interface-number* | *interface-name* }] [*port-number*]	

检查配置结果

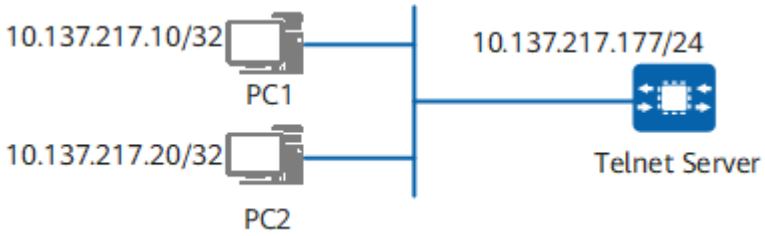
- 执行**display tcp status**命令，查看当前建立的所有TCP连接情况。

1.1.3.5.3 举例：配置用户通过 Telnet 登录设备

组网需求

如图1-13所示，用户希望简单方便的配置和管理设备。在服务器端配置Telnet用户使用AAA验证登录，并配置ACL策略，保证只有符合ACL策略的用户才能登录设备。

图 1-13 配置通过 Telnet 登录设备组网图



配置思路

采用如下的思路进行配置：

1. 配置Telnet方式登录设备，以实现远程维护网络设备。
2. 配置ACL策略，保证只有符合ACL策略的用户才能登录设备。
3. 配置管理员的用户名和密码，并配置AAA认证策略，保证只有认证通过的用户才能登录设备。

配置注意事项

当网络所处环境不足够安全时，我们建议选择安全的密码认证方式/加密认证算法/协议。安全的举例请参见：**配置用户通过STelnet登录设备**。

操作步骤

步骤1 在用户视图下执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

步骤2 配置Telnet服务器的管理网口IP地址。

```
<HUAWEI> system-view
[~HUAWEI] sysname Telnet Server
[*HUAWEI] commit
[~Telnet Server] interface gigabitethernet 0/0/0
[~Telnet Server-GigabitEthernet0/0/0] ip address 10.137.217.177 255.255.255.0
[*Telnet Server-GigabitEthernet0/0/0] quit
[*Telnet Server] commit
```

步骤3 配置服务器的端口号以及使能服务器功能。

```
[~Telnet Server] telnet server enable
[*Telnet Server] telnet server port 1025
[*Telnet Server] telnet server-source -i gigabitethernet 0/0/0
[*Telnet Server] commit
```

步骤4 配置VTY用户界面的相关参数。

配置VTY用户界面的最大个数。

```
[~Telnet Server] user-interface maximum-vty 8
[*Telnet Server] commit
```

配置允许用户登录设备的主机地址。

```
[~Telnet Server] acl 2001
[*Telnet Server-acl4-basic-2001] rule permit source 10.137.217.10 0
[*Telnet Server-acl4-basic-2001] rule deny source 10.137.217.20 0
[*Telnet Server-acl4-basic-2001] quit
[*Telnet Server] user-interface vty 0 7
[*Telnet Server-ui-vty0-7] acl 2001 inbound
[*Telnet Server-ui-vty0-7] commit
```


配置VTY用户界面的终端属性。

```
[~Telnet Server-ui-vty0-7] shell
[*Telnet Server-ui-vty0-7] idle-timeout 20
[*Telnet Server-ui-vty0-7] screen-length 30
[*Telnet Server-ui-vty0-7] history-command max-size 20
[*Telnet Server-ui-vty0-7] protocol inbound telnet
[*Telnet Server-ui-vty0-7] commit
```

配置VTY用户界面的用户验证方式。

```
[~Telnet Server-ui-vty0-7] authentication-mode aaa
[*Telnet Server-ui-vty0-7] quit
[*Telnet Server] commit
```

步骤5 配置登录用户的相关信息。

配置登录验证方式。

```
[~Telnet Server] aaa
[~Telnet Server-aaa] local-user admin1234 password
Please configure the password (8-128)
Enter Password:
Confirm Password:
[*Telnet Server-aaa] local-user admin1234 service-type telnet
[*Telnet Server-aaa] local-user admin1234 level 3
[*Telnet Server-aaa] quit
[*Telnet Server] commit
```

----结束

检查配置结果

进入PC1的Windows的命令行提示符，执行相关命令，通过Telnet方式登录设备。

```
C:\Documents and Settings\Administrator> telnet 10.137.217.177 1025
```

输入Enter键后，在登录窗口输入AAA验证方式配置的登录用户名和密码，验证通过后，出现用户视图的命令行提示符，至此用户成功登录设备。

```
Username:admin1234
Password:
Info: The max number of VTY users is 8, the number of current VTY users online is 1, and total number of terminal users online is 1.
<Telnet Server>
```

配置脚本

```
#
sysname Telnet Server
#
telnet server-source -i GigabitEthernet0/0/0
telnet server port 1025
#
acl number 2001
 rule 5 permit source 10.137.217.10 0
 rule 10 deny source 10.137.217.20 0
#
aaa
 local-user admin1234 password irreversible-cipher $1d$g8wLJ`LjL!$CyE(V{3qg5DdU:PM[6=6O
$UF-.fQ,Q}>^)OBzgoU$
 local-user admin1234 service-type telnet
 local-user admin1234  level 3
#
interface GigabitEthernet0/0/0
 ip address 10.137.217.177 255.255.255.0
#
user-interface maximum-vty 8
```



```
#
user-interface vty 0 7
 acl 2001 inbound
 authentication-mode aaa
 history-command max-size 20
 idle-timeout 20 0
 screen-length 30
#
return
```

1.1.3.5.4 举例：配置设备作为 Telnet 客户端登录其他设备

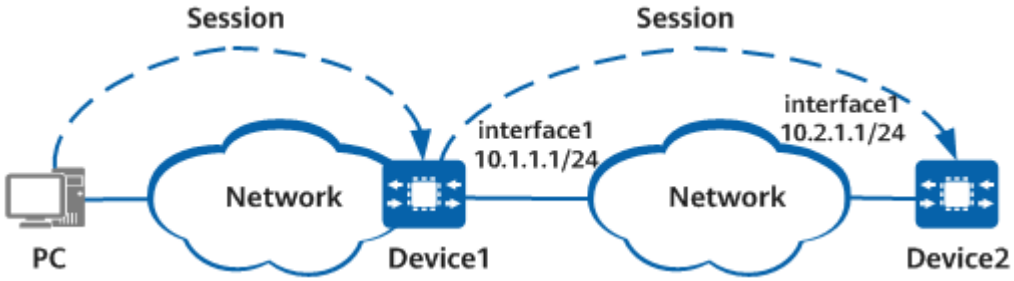
组网需求

如图1-14所示，终端PC与Device1间路由可达，Device1与Device2间路由可达。用户希望实现对远程设备Device2的管理与维护，不过终端PC与远程设备Device2间无可达路由，不能直接Telnet远程登录到Device2。用户可以通过Telnet登录到Device1，再从Device1通过Telnet登录到需要管理的设备Device2。为了防止其他非法设备通过Telnet方式登录Device2，配置ACL规则只允许Device1通过Telnet方式登录Device2。

图 1-14 配置设备作为 Telnet 客户端登录其他设备组网图

说明

本例中interface1代表GigabitEthernet1/0/1。



配置思路

采用如下的思路配置设备作为Telnet客户端登录其他设备：

1. 在Device2上配置Telnet验证方式和密码。
2. 在Device2上配置ACL规则允许Device1登录。
3. 从Device1上Telnet登录到Device2。

配置注意事项

当网络所处环境不足够安全时，我们建议选择安全的密码认证方式/加密认证算法/协议。安全的举例请参见：**配置设备作为STelnet客户端登录其他设备**。

操作步骤

步骤1 在用户视图下执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

步骤2 配置Device2的Telnet验证方式和密码。

```
<HUAWEI> system-view
[~HUAWEI] sysname Device2
```

```
[*HUAWEI] commit
[~Device2] user-interface vty 0 4
[~Device2-ui-vty0-4] authentication-mode aaa
[*Device2-ui-vty0-4] quit
[*Device2] commit
```

步骤3 配置登录用户的相关信息。

```
[~Device2] aaa
[~Device2-aaa] local-user admin1234 password
Please configure the password (8-128)
Enter Password:
Confirm Password:
[*Device2-aaa] local-user admin1234 service-type telnet
[*Device2-aaa] local-user admin1234 level 3
[*Device2-aaa] quit
[*Device2] commit
```

步骤4 在Device2上配置ACL规则允许Device1登录。

```
[~Device2] acl 2000
[*Device2-acl4-basic-2000] rule permit source 10.1.1.1 0
[*Device2-acl4-basic-2000] quit
[*Device2] user-interface vty 0 4
[*Device2-ui-vty0-4] acl 2000 inbound
[*Device2-ui-vty0-4] quit
[*Device2] commit
```

说明

采用ACL方式配置Telnet终端服务的配置为可选配置。

----结束

检查配置结果

完成以上配置后，仅可以从Device1上Telnet登录到Device2，无法从其他设备登录到Device2。

```
<HUAWEI> system-view
[~HUAWEI] sysname Device1
[*HUAWEI] commit
[~Device1] quit
<Device1> telnet 10.2.1.1
Username:admin1234
Password:
Info: The max number of VTY users is 8, the number of current VTY users online is 1, and total number of terminal users online is 1.
<Device2>
```

配置脚本

Device2的配置脚本

```
#
sysname Device2
#
acl number 2000
 rule 5 permit source 10.1.1.1 0
#
aaa
 local-user admin1234 password irreversible-cipher $1d$g8wLJ`LjL!$CyE(V{3qg5DdU:PM[6=6O
$UF-.fQ,Q}>^)OBzgoU$
 local-user admin1234 level 3
 local-user admin1234 service-type telnet
#
user-interface vty 0 4
 acl 2000 inbound
```



```
 authentication-mode aaa
#
return
```

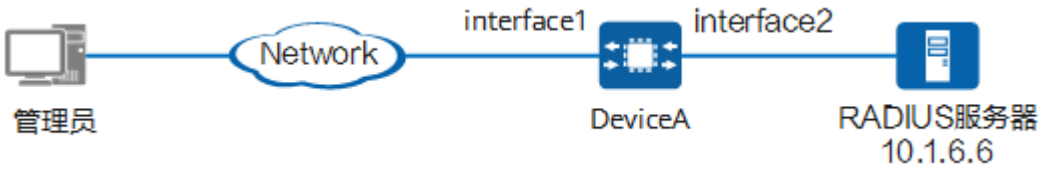
1.1.3.5.5 举例：配置基于 ACL 规则和 RADIUS 认证限制 Telnet 登录设备

组网需求

网络管理员希望能够对设备进行远程管理与维护，同时对网络的安全性要求较高，希望网络不受未授权访问的干扰。可以通过配置基于ACL规则以及RADIUS认证的Telnet登录方式满足用户的需求。

如**图 基于ACL规则和RADIUS认证限制Telnet登录设备组网图**所示，DeviceA为Telnet服务器，网络管理员和DeviceA之间、DeviceA与RADIUS服务器之间路由可达。RADIUS服务器的IP地址为10.1.6.6/24，认证端口号为1812。

图 1-15 基于 ACL 规则和 RADIUS 认证限制 Telnet 登录设备组网图



说明

本例中interface1和interface2分别代表GigabitEthernet1/0/1和GigabitEthernet1/0/2。

配置思路

采用如下的思路进行配置（基于ACL规则和RADIUS认证）：

1. 配置设备接口相关参数。
2. 配置Telnet协议，实现用户可以通过Telnet登录设备。
3. 配置ACL规则，保证只有满足该规则的用户才能登录设备。
4. 配置RADIUS协议，实现RADIUS认证。用户通过Telnet登录设备时使用RADIUS服务器上配置的用户名和密码，从而保证用户登录的安全性。
5. 配置RADIUS服务器。

配置注意事项

- 当网络所处环境不足够安全时，我们建议选择安全的密码认证方式/加密认证算法/协议。安全的举例请参见：**举例：配置通过STelnet登录设备示例（基于RADIUS认证）**
- 配置前请确保各设备之间路由可达。
- 请确保设备上配置的RADIUS服务器的地址、端口号和共享密钥配置正确，并且和RADIUS服务器上的配置保持一致。
- 域被配置成全局默认管理域之后，管理用户的用户名中携带该域名或者不携带域名时，会使用全局默认管理域下的AAA配置信息。
- 请确保已在RADIUS服务器上配置了用户。本例中假设RADIUS服务器上已配置了用户**admin123@huawei.com**（用户名@域名），其密码为**YsHsjx_202206**。

- 如果RADIUS服务器不接受包含域名的用户名，还需要配置命令**undo radius-server user-name domain-included**使设备向RADIUS服务器发送的报文中的用户名不包含域名。

操作步骤

步骤1 在用户视图下执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

步骤2 配置接口的相关参数。

配置接口IP地址。

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceA
[*DeviceA] commit
[~DeviceA] vlan batch 10 20
[*DeviceA] interface gigabitethernet 1/0/1
[*DeviceA-GigabitEthernet1/0/1] portswitch
[*DeviceA-GigabitEthernet1/0/1] port link-tpye trunk
[*DeviceA-GigabitEthernet1/0/1] port trunk allow-pass vlan 10
[*DeviceA-GigabitEthernet1/0/1] quit
[*DeviceA] interface gigabitethernet 1/0/2
[*DeviceA-GigabitEthernet1/0/2] portswitch
[*DeviceA-GigabitEthernet1/0/2] port link-tpye trunk
[*DeviceA-GigabitEthernet1/0/2] port trunk allow-pass vlan 20
[*DeviceA-GigabitEthernet1/0/2] quit
[*DeviceA] interface vlanif 10
[*DeviceA-Vlanif10] ip address 10.1.1.2 255.255.255.0
[*DeviceA-Vlanif10] quit
[*DeviceA] interface vlanif 20
[*DeviceA-Vlanif20] ip address 10.1.6.2 255.255.255.0
[*DeviceA-Vlanif20] quit
[*DeviceA] commit
```

步骤3 配置Telnet登陆。

配置服务器的端口号以及使能Telnet服务器功能。

```
[~DeviceA] telnet server enable
[*DeviceA] telnet server port 1025
[*DeviceA] telnet server-source -i vlanif 10
[*DeviceA] commit
```

配置VTY用户界面0～4的认证方式为AAA认证、支持的协议为Telnet。

```
[*DeviceA] user-interface vty 0 4
[*DeviceA-ui-vty0-4] authentication-mode aaa
[*DeviceA-ui-vty0-4] protocol inbound telnet
[*DeviceA-ui-vty0-4] user privilege level 3
[*DeviceA-ui-vty0-4] quit
[*Telnet Server] commit
```

步骤4 在DeviceA上配置ACL规则允许管理员登录。

```
[~DeviceA] acl 2000
[*DeviceA-acl4-basic-2000] rule permit source 10.137.217.10 0
[*DeviceA-acl4-basic-2000] quit
[*DeviceA] user-interface vty 0 4
[*DeviceA-ui-vty0-4] acl 2000 inbound
[*DeviceA-ui-vty0-4] quit
[*DeviceA] commit
```

说明

采用ACL方式配置Telnet终端服务的配置为可选配置。

步骤5 配置RADIUS认证。

配置RADIUS服务器模板，实现与RADIUS服务器的通信。

```
[~DeviceA] radius-server template 1
[*DeviceA-radius-1] radius-server authentication 10.1.6.6 1812
[*DeviceA-radius-1] radius-server shared-key cipher Huawei@123456789
[*DeviceA-radius-1] quit
[*DeviceA] commit
```

配置AAA认证方案，指定认证方式为RADIUS。

```
[~DeviceA] aaa
[~DeviceA-aaa] authentication-scheme auth1
[*DeviceA-aaa-authen-auth1] authentication-mode radius
[*DeviceA-aaa-authen-auth1] quit
```

创建域，并在域下引用AAA认证方案及RADIUS服务器模板。

```
[*DeviceA-aaa] domain huawei.com
[*DeviceA-aaa-domain-huawei.com] authentication-scheme auth1
[*DeviceA-aaa-domain-huawei.com] accounting-scheme acc1
[*DeviceA-aaa-domain-huawei.com] radius-server 1
[*DeviceA-aaa-domain-huawei.com] quit
[*DeviceA-aaa] quit
[*DeviceA] commit
```

配置huawei.com为全局默认管理域，这样管理员登录设备时就不需要输入域名。

```
[~DeviceA] domain huawei.com admin
[*DeviceA] commit
```

步骤6 配置RADIUS服务器。

配置步骤包括：添加设备、添加用户、配置授权用户级别为3。

----结束

检查配置结果

进入PC1的Windows的命令行提示符，执行相关命令，通过Telnet方式登录设备。

```
C:\Documents and Settings\Administrator> telnet 10.1.1.2 1025
```

在登录界面根据提示输入RADIUS服务器上配置的用户名**admin1234**和密码**YsHsjx_202206**，并按Enter键，验证通过，用户成功通过Telnet登录到DeviceA。（以下显示仅为示意）

```
Username:admin1234
Password:
Info: The max number of VTY users is 8, the number of current VTY users online is 1, and total number of terminal users online is 1.
<DeviceA>
```

配置脚本

```
#
sysname DeviceA
#
radius-server template 1
 radius-server shared-key cipher %+%##!!!!!!!!!"!!!!"!!!!*!!!!SKvr${[Fs.3t@/5k|BENhEu>W(3\~XG!!D;!!!!!2jp5!!!!!!A!!!!3"pK8qv!}9M#(4$jGWvQF/R[CNe/+:W^jk8HUe&W%+%#
 radius-server authentication 10.1.6.6 1812 weight 80
#
aaa
 authentication-scheme auth1
  authentication-mode radius
 accounting-scheme acc1
```



```
 accounting-mode radius
 domain huawei.com
  authentication-scheme auth1
  accounting-scheme acc1
  radius-server 1
#
domain huawei.com admin
#
vlan batch 10 20
#
interface Vlanif10
 ip address 10.1.1.2 255.255.255.0
#
interface Vlanif20
 ip address 10.1.6.2 255.255.255.0
#
interface GigabitEthernet1/0/1
 portswitch
 port link-type trunk
 port trunk allow-pass vlan 10
#
interface GigabitEthernet1/0/2
 portswitch
 port link-type trunk
 port trunk allow-pass vlan 20
#
telnet server-source -i Vlanif10
telnet server port 1025
#
user-interface vty 0 4
 authentication-mode aaa
 user privilege level 3
 protocol inbound telnet
#
acl number 2000
 rule 5 permit source 10.137.217.10 0
#
return
```

1.1.3.6 配置 STelnet 远程登录

1.1.3.6.1 配置用户通过 STelnet 登录设备

前提条件

在配置用户通过STelnet登录设备之前，需完成以下任务：

- 终端与设备之间路由可达。
- 终端上已安装SSH客户端软件。

说明

使用STelnet V1协议存在安全风险，建议使用STelnet V2登录设备。

缺省配置

表 1-33 配置用户通过 STelnet 登录设备的缺省值

参数	缺省值
STelnet服务器功能	关闭

操作步骤

步骤1 使能STelnet功能。

```
system-view
stelnet [ ipv4 | ipv6 ] server enable
commit
```

步骤2 配置STelnet服务器功能及参数。

详细配置请参见《配置指南-安全配置》中的“配置SSH服务器功能及参数”。

步骤3 配置SSH用户登录的用户界面。

详细配置请参见《配置指南-安全配置》中的“配置VTY界面支持SSH协议”。

步骤4 配置SSH用户。

详细配置请参见《配置指南-安全配置》中的“配置SSH用户”。

步骤5 用户通过STelnet登录设备。

从终端通过STelnet登录设备，此处以使用第三方软件OpenSSH和Windows命令行提示符为例进行配置。

- OpenSSH软件的安装请参考该软件的安装说明。
- 使用OpenSSH软件从终端访问设备时，需要使用OpenSSH的命令，命令的使用可以参见该软件的帮助文档。
- 只有安装了OpenSSH软件后，Windows命令行提示符才能识别OpenSSH相关命令。

进入Windows的命令行提示符，执行OpenSSH命令，登录设备。(以下显示信息仅为示意)

```
C:\Users\User1>ssh admin@10.136.195.11
admin@10.136.195.11's password:

Info: The max number of VTY users is 21, the number of current VTY users online is 5, and total number of terminal users online is 5.
    The current login time is 2020-12-15 14:23:00.
<HUAWEI>
```

----结束

检查配置结果

- 执行**display ssh user-information** [*username*]命令，在SSH服务器端查看SSH用户信息。如果不指定SSH用户，则可以查看SSH服务器端所有的SSH用户信息。
- 执行**display ssh server status**命令，查看SSH服务器的全局配置信息。
- 执行**display ssh server session**命令，在SSH服务器端查看与SSH客户端连接的会话信息。

1.1.3.6.2 配置设备作为 STelnet 客户端登录其他设备

前提条件

在配置设备作为STelnet客户端登录其他设备之前，需完成以下任务：

- 已从终端成功登录到本设备。

- 本设备与STelnet服务器之间路由可达。
- STelnet服务器端已开启STelnet服务器功能。
- 获取STelnet服务器端配置的SSH用户信息和端口号。

说明

使用STelnet V1协议存在安全风险，建议使用STelnet V2登录设备。

操作步骤

步骤1 配置设备首次连接SSH服务器的方式。

请参见《配置指南-安全配置》中的“配置设备首次连接SSH服务器的方式”。

步骤2 配置SSH客户端参数。

请参见《配置指南-安全配置》中的“配置SSH客户端参数”。

步骤3 使用STelnet命令登录其他设备。

表 1-34 使用 STelnet 命令登录其他设备（三层网络连接正常）

操作步骤	命令	说明
以STelnet方式通过IPv4的地址登录到SSH服务器上	**stelnet** [**-a** *source-ip-address* | **-i** *interface-type interface-number*] [**-force-receive-pubkey**] *host-ip-address* [*server-port*] [[**prefer_kex** *prefer_kex*] | [**prefer_ctos_cipher** *prefer_ctos_cipher*] | [**prefer_stoc_cipher** *prefer_stoc_cipher*] | [**prefer_ctos_hmac** *prefer_ctos_hmac*] | [**prefer_stoc_hmac** *prefer_stoc_hmac*] | [**prefer_ctos_compress zlib**] | [**prefer_stoc_compress zlib**] | [**-vpn-instance** *vpn-instance-name*] | [**-ki** *interval*] | [**-kc** *count*] | [**identity-key** *identity-key-type*] | [**user-identity-key** *user-key*]] *	根据网络地址类型不同，选择其中一种。 只有当服务器正在尝试连接的端口号是22时，STelnet客户端登录时可以不指定端口号。如果是其他端口号，STelnet客户端登录时必须指定端口号。 STelnet客户端登录SSH服务器时可以携带源IP地址、VPN实例名、选择密钥交换算法、加密算法、压缩算法和HMAC算法以及设置keepalive功能。 如果使用参数**-i** *interface-type interface-number*指定源接口，则不支持**public-net**和**-vpn-instance** *vpn-instance-name*参数。

操作步骤	命令	说明
以STelnet方式通过IPv6的地址登录到SSH服务器上	**stelnet ipv6** [**-a** *source-ipv6-address*] [**-force-receive-pubkey**] *host-ipv6-address* [[**public-net** | **-vpn-instance** *vpn-instance-name*] | [**-oi** { *interface-name* | *interface-type interface-number* }] | [*server-port*] | [**prefer_kex** *prefer_kex*] | [**prefer_ctos_cipher** *prefer_ctos_cipher*] | [**prefer_stoc_cipher** *prefer_stoc_cipher*] | [**prefer_ctos_hmac** *prefer_ctos_hmac*] | [**prefer_stoc_hmac** *prefer_stoc_hmac*] | [**prefer_ctos_compress zlib**] | [**prefer_stoc_compress zlib**] | [**-ki** *interval*] | [**-kc** *count*] | [**identity-key** *identity-key-type*] | [**user-identity-key** *user-key*]] *	

----结束

检查配置结果

- 执行**display ssh server-info**命令，在SSH客户端查看所有SSH服务器与公钥之间的对应关系。
- 执行**display ssh client session**命令，查看SSH客户端密钥重协商后在线会话的接收/发送报文数量，接收/发送报文数据量以及STelnet登录时长。

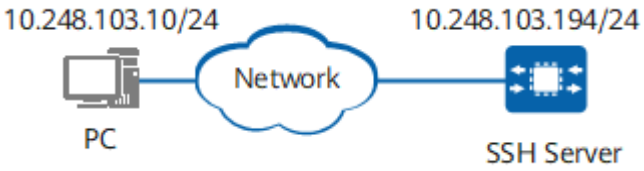
1.1.3.6.3 举例：配置用户通过 STelnet 登录设备

组网需求

如图1所示，在作为SSH服务器的设备上使能STelnet服务器功能后，SSH客户端PC可以通过不同的认证方式登录SSH服务器，这里以RSA认证方式为例介绍客户端通过STelnet登录服务器的配置过程。

为了提升系统安全性，防止非法用户登录到SSH服务器，用户可以在SSH服务器上配置ACL规则。

图 1-16 配置用户通过 STelnet 登录设备组网图



配置思路

采用如下的思路配置SSH用户通过STelnet登录设备：

1. 配置SSH服务器的管理网口IP地址。
2. 在SSH服务器端生成本地密钥对。
3. 配置SSH服务器的VTY用户界面。
4. 创建本地用户，并配置服务类型。
5. 创建SSH用户，并配置认证方式。
6. SSH客户端根据配置的SSH用户认证类型创建相应的密钥对，并将公钥拷贝至SSH服务器。
7. SSH服务器端编辑公钥，并将编辑好的公钥分配给用户。
8. 使能SSH服务器的STelnet功能，配置SSH用户的服务类型为STelnet。
9. 在SSH服务器上配置允许STelnet客户端登录的ACL规则。
10. 配置客户端登录软件的参数，STelnet至服务器。

数据准备

为完成此配置示例，需准备如下数据：

说明

为了保证更好的安全性，建议使用3072位及以上的RSA密钥对。

- SSH客户端已安装OpenSSH软件。
- SSH服务器管理网口的IP地址为10.248.103.194/24。
- 本地用户的认证方式为password，用户名为“admin123”，密码为“YsHsjx_202206”。
- SSH用户的认证方式为RSA。
- 配置基本的ACL 2000，允许10.248.103.0/24网段的客户端合法接入SSH服务器。

操作步骤

步骤1 配置SSH服务器的管理网口IP地址。

```
<HUAWEI> system-view
[~HUAWEI] sysname SSH Server
[*HUAWEI] commit
[~SSH Server] interface gigabitethernet 0/0/0
[~SSH Server-GigabitEthernet0/0/0] ip address 10.248.103.194 255.255.255.0
[*SSH Server-GigabitEthernet0/0/0] quit
[*SSH Server] commit
```

步骤2 在SSH服务器端生成本地密钥对。

```
[~SSH Server] rsa local-key-pair create
The key name will be:Host
The range of public key size is (2048, 4096).
NOTE: Key pair generation will take a short while.
Please input the modulus [default = 3072]:3072
[*SSH Server] commit
```

步骤3 配置SSH服务器的VTY用户界面。

```
[~SSH Server] user-interface vty 0 4
[~SSH Server-ui-vty0-4] authentication-mode aaa
[*SSH Server-ui-vty0-4] protocol inbound ssh
[*SSH Server-ui-vty0-4] quit
[*SSH Server] commit
```


说明

若配置登录协议为SSH，则设备将自动禁止Telnet功能。

步骤4 在服务器端创建本地用户，并配置用户服务方式。

```
[~SSH Server] aaa
[~SSH Server-aaa] local-user admin123 password
Please configure the password (8-128)
Enter Password:
Confirm Password:
[*SSH Server-aaa] local-user admin123 service-type ssh
[*SSH Server-aaa] local-user admin123 level 3
[*SSH Server-aaa] quit
[*SSH Server] commit
```

步骤5 在服务器端创建SSH用户，并配置认证方式。

```
[~SSH Server] ssh user admin123
[*SSH Server] ssh user admin123 authentication-type rsa
[*SSH Server] commit
```

步骤6 配置SSH服务器的公钥算法、加密算法、密钥交换算法列表、HMAC认证算法和最小密钥长度。

```
[~SSH Server] ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
[~SSH Server] ssh server hmac sha2_256 sha2_512
[~SSH Server] ssh server key-exchange dh_group_exchange_sha256 dh_group16_sha512
[~SSH Server] ssh server publickey rsa_sha2_256 rsa_sha2_512
[~SSH Server] ssh server dh-exchange min-len 3072
[*SSH Server] commit
```

步骤7 SSH客户端使用OpenSSH创建RSA密钥对，并将密钥对中的公钥拷贝至SSH服务器。

进入Windows的命令行提示符，创建RSA密钥对，并保存到本地id_rsa.pub文件中（以下内容仅为示例）。

```
C:\Users\User1>ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\User1/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\User1/.ssh/id_rsa.
Your public key has been saved in C:\Users\User1/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:c43yubJjCUjY3JqH0aVZwJFM3gWJcH4YI5+4HUDAIqo
The key's randomart image is:
+---[RSA 3072]----+
| ..o==B=.o.      |
|o .  O=*+.       |
|o. +.oB=o        |
|. . =o=o   o     |
|.  ..*. S o .    |
|E   = o  = .     |
|     . . .o      |
|        =  .     |
|       ..+.      |
+----[SHA256]-----+
```

步骤8 SSH服务器编辑SSH客户端OpenSSH生成的公钥，并将编辑后的公钥分配给SSH用户。

```
[~SSH Server] rsa peer-public-key rsa01 encoding-type openssh
[*SSH Server-rsa-public-key] public-key-code begin
[*SSH Server-rsa-public-key-rsa-key-code] ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCg5Ag490i6ilB7QuCVb35B8RJEh1DIYB88h2p1qjdh7qdMQv8rp
JaVAgQWxwzKZO0XdFuz4ReGQzTCSf7Det7Ajicddw3qi+6P8hRqZj6MPdLg/o3RN4aPCfr/
LFWCwqJ3gWGHlOC7qqjRk+6pySVoiWcSk5/elBkU7WVk/
cSWrt4qFXJV373OCesKcEVeDvAa1Tvx6L3LQroBqUO0EXzDgOthPCmOqiqvS5h3JipzqVsesdSKjeInooCQzS
Ov5eePpBcFcIvU6wFiLIZ5vnf6YtypgTVzHuje/sh4xM7Iuuon7AYXKHT8NpO9jd9zA/lKaRPXyDtei1O1Bt/
5lxnn
[*SSH Server-rsa-public-key-rsa-key-code] public-key-code end
```



```
[*SSH Server-key-code] peer-public-key end
[*SSH Server] ssh user admin123 assign rsa-key rsa01
[*SSH Server] commit
```

步骤9 使能STelnet功能，并配置用户的服务类型为STelnet。

```
[~SSH Server] stelnet server enable
[*SSH Server] ssh server-source all-interface
[*SSH Server] ssh user admin123 service-type stelnet
[*SSH Server] commit
```

步骤10 配置ACL规则。

```
[~SSH Server] acl 2000
[*SSH Server-acl4-basic-2000] rule permit source 10.248.103.0 8
[*SSH Server-acl4-basic-2000] quit
[*SSH Server] ssh server acl 2000
[*SSH Server] commit
```

----结束

检查配置结果

客户端通过OpenSSH软件登录SSH服务器。进入Windows的命令行提示符，执行OpenSSH命令，通过STelnet方式访问设备。

```
C:\Users\User1>ssh admin123@10.248.103.194
Enter passphrase for key 'C:\Users\User/.ssh/id_rsa':
Info: The max number of VTY users is 21, the number of current VTY users online is 4, and total number of terminal users online is 4.
      The current login time is 2020-12-15 15:58:03.
<SSH Server>
```

配置脚本

```
#
sysname SSH Server
#
acl number 2000
 rule 5 permit source 10.248.103.0 0.0.0.255
#
rsa peer-public-key rsa01 encoding-type openssh
 public-key-code begin
  ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCg5Ag490i6ilB7QuCVb35B8RJEh1DIYB88h2p1qjdh7qdMQv8rpJaVA
gQWxwzKZO0XdFuz4ReGQzTCSf7Det7Ajicddw3qi+6P8hRqZj6MPdLg/o3RN4aPCfr/
LFWCwqJ3gWGHlOC7qqjRk+6pySVoiWcSk5/elBkU7WVk/
cSWrt4qFXJV373OCesKcEVeDvAa1Tvx6L3LQroBqUO0EXzDgOthPCmOqiqvS5h3JipzqVsesdSKjeInooCQzSOv5e
ePpBcFcIvU6wFiLIZ5vnf6YtypgTVzHuje/sh4xM7Iuuon7AYXKHT8NpO9jd9zA/lKaRPXyDtei1O1Bt/5lxnn rsa-key
 public-key-code end
 peer-public-key end
#
aaa
 local-user admin123 password irreversible-cipher $1d$+,JS+))\\2$KVNj(.
3`_5x0FCKGv}H&.kUTI`Ff&H*eBqO.ua>)$
 local-user admin123 service-type terminal ssh
 local-user admin123 level 3
#
interface GigabitEthernet0/0/0
 ip address 10.248.103.194 255.255.255.0
#
stelnet server enable
ssh user admin123
ssh user admin123 authentication-type rsa
ssh user admin123 assign rsa-key rsa01
ssh user admin123 service-type stelnet
ssh server-source all-interface
ssh server acl 2000
#
```



```
ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
ssh server hmac sha2_256 sha2_512
ssh server key-exchange dh_group_exchange_sha256 dh_group16_sha512
ssh server publickey rsa_sha2_256 rsa_sha2_512
ssh server dh-exchange min-len 3072
#
user-interface vty 0 4
 authentication-mode aaa
 idle-timeout 120 0
 protocol inbound ssh
#
return
```

1.1.3.6.4 举例：配置设备作为 STelnet 客户端登录其他设备

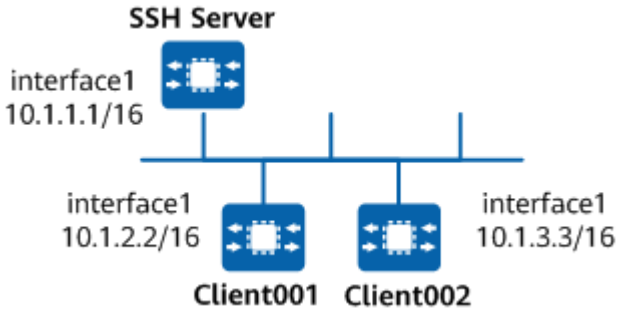
组网需求

如图1-17所示，用户希望在服务器端和客户端进行安全的数据交互，配置两个登录用户为Client001和Client002，分别使用password认证方式和RSA认证方式登录SSH服务器，并且配置新的端口号，而不使用缺省端口号。

图 1-17 配置通过 STelnet 登录其他设备组网图

说明

本例中interface1代表GigabitEthernet1/0/1。



配置思路

采用如下的思路配置通过STelnet登录其他设备：

1. 在SSH服务器端生成本地密钥对，实现在服务器端和客户端进行安全的数据交互。
2. 在SSH服务器端配置SSH用户client001和client002分别使用不同的认证方式。
3. 在SSH服务器端开启STelnet服务功能。
4. 在SSH服务器端配置SSH用户client001和client002的服务方式为STelnet。
5. 在SSH服务器端配置SSH服务器的端口号，有效防止攻击者对SSH服务标准端口的访问，确保安全性。
6. 用户client001和client002分别以STelnet方式实现登录SSH服务器。

操作步骤

步骤1 在服务器端生成本地密钥对。

```
<HUAWEI> system-view
[~HUAWEI] sysname SSH Server
```

```
[*HUAWEI] commit
[~SSH Server] rsa local-key-pair create
The key name will be:Host
The range of public key size is (2048, 4096).
NOTE: Key pair generation will take a short while.
Please input the modulus [default = 3072]:
[*SSH Server] commit
```

步骤2 在服务器端创建SSH用户。

配置VTY用户界面。

```
[~SSH Server] user-interface vty 0 4
[~SSH Server-ui-vty0-4] authentication-mode aaa
[*SSH Server-ui-vty0-4] protocol inbound ssh
[*SSH Server-ui-vty0-4] quit
[*SSH Server] commit
```

- 创建SSH用户client001。

  # 新建用户名为client001的SSH用户，且认证方式为password。

  ```
  [~SSH Server] aaa
  [~SSH Server-aaa] local-user client001 password
  Please configure the password (8-128)
  Enter Password:
  Confirm Password:
  [*SSH Server-aaa] local-user client001 level 3
  [*SSH Server-aaa] local-user client001 service-type ssh
  [*SSH Server-aaa] quit
  [*SSH Server] ssh user client001
  [*SSH Server] ssh user client001 authentication-type password
  [*SSH Server] commit
  ```

  #在客户端Client001，配置加密算法、HMAC认证算法、密钥交换算法列表、公钥算法。

  ```
  <HUAWEI> system-view
  [~HUAWEI] sysname client001
  [*HUAWEI] commit
  [*client001] ssh client cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
  [*client001] ssh client hmac sha2_256 sha2_512
  [*client001] ssh client key-exchange dh_group_exchange_sha256 dh_group16_sha512
  [*client001] ssh client publickey rsa_sha2_256 rsa_sha2_512
  [*client001] commit
  ```

- 创建SSH用户client002。

  # 新建用户名为client002的SSH用户，且认证方式为RSA。

  ```
  [~SSH Server] ssh user client002
  [*SSH Server] ssh user client002 authentication-type rsa
  [*SSH Server] ssh authorization-type default root
  [*SSH Server] commit
  ```

  # 在STelnet客户端Client002生成客户端的本地密钥对。

  ```
  <HUAWEI> system-view
  [~HUAWEI] sysname client002
  [*HUAWEI] commit
  [~client002] rsa local-key-pair create
  The key name will be: client002_Host
  The range of public key size is (2048, 4096).
  NOTE: Key pair generation will take a short while.
  Please input the modulus [default = 3072]:
  [*client002] commit
  ```

  # 配置STelnet客户端Client002的加密算法、HMAC认证算法、密钥交换算法列表、公钥算法。

  ```
  [*client002] ssh client cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
  [*client002] ssh client hmac sha2_256 sha2_512
  [*client002] ssh client key-exchange dh_group_exchange_sha256 dh_group16_sha512
  ```



```
[*client002] ssh client publickey rsa_sha2_256 rsa_sha2_512
[*client002] commit
```

查看客户端上生成的RSA密钥对的公钥部分。

```
[~client002] display rsa local-key-pair public
=====================Host key=========================
Time of key pair created : 2019-11-03 08:56:38
Key name                 : client002_Host
Key type                 : RSA encryption key
==========================================================
Key code:

3082010A
  02820101
    00A4BAB8 B964077E F7657F7F E4BE1DE8 71EE1707
    E4EE2864 2D06FBE0 BFC1CB52 F99B7A99 0132B709
    3F841CA2 3544B8B2 6EE0A9ED 04B19FE3 FB3DA86D
    BE68FFE2 2303108D BDC24B80 A1793A08 FDA0B6C1
    13C31EA5 298EC9B1 2B0BC8BD 32CFF896 29F8CA98
    8B1724AF 5DA8A390 20906ADE 6A8AD77D 6234F0C8
    DC965BA0 1771D9C0 A89ED49B 5ECF7EE2 D5997527
    FC87FE03 E51658C1 0996DFDF DC456376 2FA4B268
    4345131D 431419D2 DD5E4003 6A7D3295 145F3175
    22E80686 E6B39A05 799D6BCF A78F69B6 BC2D0836
    F5013421 77D68B89 A9EC182A 04B87BE3 500FCE14
    9C95CF78 75704359 0C70FD60 1EFC0B99 32F02142
    4CE781E4 36A60BFC 2CBD07F6 9E700CEE 4D0203
    010001

Host public key for PEM format code:
---- BEGIN SSH2 PUBLIC KEY ----
AAAAB3NzaC1yc2EAAAADAQABAAABAQCkuri5ZAd+92V/f+S+Hehx7hcH5O4oZC0G
++C/wctS+Zt6mQEytwk/hByiNUS4sm7gqe0EsZ/j+z2obb5o/+IjAxCNvcJLgKF5
Ogj9oLbBE8MepSmOybErC8i9Ms/4lin4ypiLFySvXaijkCCQat5qitd9YjTwyNyW
W6AXcdnAqJ7Um17PfuLVmXUn/If+A+UWWMEJlt/f3EVjdi+ksmhDRRMdQxQZ0t1e
QANqfTKVFF8xdSLoBobms5oFeZ1rz6ePaba8LQg29QE0IXfWi4mp7BgqBLh741AP
zhSclc94dXBDWQxw/WAe/AuZMvAhQkzngeQ2pgv8LL0H9p5wDO5N
---- END SSH2 PUBLIC KEY ----

Public key code for pasting into OpenSSH authorized_keys file:
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCkuri5ZAd+92V/f+S+Hehx7hcH5O4oZC0G++C/wctS
+Zt6mQEytwk/hByiNUS4sm7gqe0EsZ/j+z2obb5o/+IjAxCNvcJLgKF5Ogj9oLbBE8MepSmOybErC8i9Ms/
4lin4ypiLFySvXaijkCCQat5qitd9YjTwyNyWW6AXcdnAqJ7Um17PfuLVmXUn/If+A+UWWMEJlt/f3EVjdi
+ksmhDRRMdQxQZ0t1eQANqfTKVFF8xdSLoBobms5oFeZ1rz6ePaba8LQg29QE0IXfWi4mp7BgqBLh741
APzhSclc94dXBDWQxw/WAe/AuZMvAhQkzngeQ2pgv8LL0H9p5wDO5N rsa-key

Host public key for SSH1 format code:
2048 65537
20795157856672359848547361269858029949242843585831182669194523227368193104900346497
51564062838779994414811756574319056037283986651865082633457078943496774842175805981
90093729334060817838060780955449126599749626192655532498343534107533323544305478060
44311868210891515536106321547674857755678562420627679242838953538641596303196319735
54494558678562482442247018243129430270141612311783975353971113532423335500440937726
19909488601542170799462826313639069974340296484981794888174430354307491156572632525
09381070628794959223309539977269992957151749764061913059943557804219705266011480071
18555920234221614917518894262681146 9
```

```
=====================Server key=======================
Time of key pair created : 2019-11-03 08:56:39
Key name                 : client002_Server
Key type                 : RSA encryption key
==========================================================
Key code:

3081B9
  0281B1
    00B9AE42 B8419F19 35C49A7B A55DBB6F 67D931F3
    9C19ECF9 9E17961B D01ED5DD 3AE68CFA 38C57113
    C93663F2 86768B19 AD0F603E 98F2C6AB A71A6C26
    8813411D 4AA56BC4 6505EC15 94647621 AB7D03BB
```



```
  79DA9B24 09BB1FD2 3927E2F9 00F79116 466411CD
  AC3D8FF6 A051FA5A 9BCE84CE 20842134 D2D27B4A
  219CB801 9F5A90E0 518DEEFC F48F5ED4 49215B1F
  11E1AC81 5E168A97 3AA5320D 7B158556 AF5CC95C
  9B508BBC 6EEFEEF9 0E23AA13 59E1F746 D5
0203
  010001
```

将客户端上产生的RSA公钥配置到服务器端（上面**display**命令显示信息中黑体部分即为客户端产生的RSA公钥，将其拷贝粘贴至服务器端）。

```
[~SSH Server] rsa peer-public-key rsakey001
[*SSH Server-rsa-public-key] public-key-code begin
[*SSH Server-rsa-public-key-rsa-key-code] 3082010A
[*SSH Server-rsa-public-key-rsa-key-code] 2820101
[*SSH Server-rsa-public-key-rsa-key-code] 00A4BAB8 B964077E F7657F7F E4BE1DE8 71EE1707
[*SSH Server-rsa-public-key-rsa-key-code] E4EE2864 2D06FBE0 BFC1CB52 F99B7A99 0132B709
[*SSH Server-rsa-public-key-rsa-key-code] 3F841CA2 3544B8B2 6EE0A9ED 04B19FE3 FB3DA86D
[*SSH Server-rsa-public-key-rsa-key-code] BE68FFE2 2303108D BDC24B80 A1793A08 FDA0B6C1
[*SSH Server-rsa-public-key-rsa-key-code] 13C31EA5 298EC9B1 2B0BC8BD 32CFF896 29F8CA98
[*SSH Server-rsa-public-key-rsa-key-code] 8B1724AF 5DA8A390 20906ADE 6A8AD77D 6234F0C8
[*SSH Server-rsa-public-key-rsa-key-code] DC965BA0 1771D9C0 A89ED49B 5ECF7EE2 D5997527
[*SSH Server-rsa-public-key-rsa-key-code] FC87FE03 E51658C1 0996DFDF DC456376 2FA4B268
[*SSH Server-rsa-public-key-rsa-key-code] 4345131D 431419D2 DD5E4003 6A7D3295 145F3175
[*SSH Server-rsa-public-key-rsa-key-code] 22E80686 E6B39A05 799D6BCF A78F69B6 BC2D0836
[*SSH Server-rsa-public-key-rsa-key-code] F5013421 77D68B89 A9EC182A 04B87BE3 500FCE14
[*SSH Server-rsa-public-key-rsa-key-code] 9C95CF78 75704359 0C70FD60 1EFC0B99 32F02142
[*SSH Server-rsa-public-key-rsa-key-code] 4CE781E4 36A60BFC 2CBD07F6 9E700CEE 4D
[*SSH Server-rsa-public-key-rsa-key-code] 203
[*SSH Server-rsa-public-key-rsa-key-code] 10001
[*SSH Server-rsa-public-key-rsa-key-code] public-key-code end
[*SSH Server-rsa-public-key] peer-public-key end
[*SSH Server] commit
```

在SSH服务器端为SSH用户client002绑定STelnet客户端的RSA公钥。

```
[~SSH Server] ssh user client002 assign rsa-key rsakey001
```

步骤3 SSH服务器端开启STelnet服务功能，并指定SSH服务端的源接口。

开启STelnet服务功能。

```
[*SSH Server] stelnet server enable
```

指定SSH服务端的源接口。

```
[*SSH Server] ssh server-source all-interface
```

配置SSH服务器的公钥算法、加密算法、密钥交换算法列表、HMAC认证算法和最小密钥长度。

```
[~SSH Server] ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
[~SSH Server] ssh server hmac sha2_256 sha2_512
[~SSH Server] ssh server key-exchange dh_group_exchange_sha256 dh_group16_sha512
[~SSH Server] ssh server publickey rsa_sha2_256 rsa_sha2_512
[~SSH Server] ssh server dh-exchange min-len 3072
[*SSH Server] commit
```

步骤4 配置SSH用户client001、client002的服务方式为STelnet。

```
[*SSH Server] ssh user client001 service-type stelnet
[*SSH Server] ssh user client002 service-type stelnet
```

步骤5 配置SSH服务器端新的端口号。

```
[*SSH Server] ssh server port 1025
[*SSH Server] commit
```

步骤6 STelnet客户端连接SSH服务器。

第一次登录，需要使能SSH客户端首次登录功能。

使能客户端Client001首次登录功能。

```
<HUAWEI> system-view
[~HUAWEI] sysname client001
[*HUAWEI] commit
[~client001] ssh client first-time enable
[*client001] commit
[~client001] quit
```

使能客户端Client002首次登录功能。

```
[~client002] ssh client first-time enable
[*client002] commit
[~client002] quit
```

STelnet客户端Client001用password认证方式连接SSH服务器，输入配置的用户名和密码。

```
<client001> stelnet 10.1.1.1 1025
Trying 10.1.1.1 ...
Press CTRL+K to abort
Connected to 10.1.1.1 ...
The server's public key does not match the one cached before.
The server is not authenticated. Continue to access it?[Y/N]:y
The keyname:10.1.1.1 already exists. Update it? [Y/N]:n

Please input the username: client001
Please select public key type for user authentication [R for RSA/D for DSA/E for ECC] Please select [R/D/E]:r
Enter password:
```

输入密码，显示登录成功信息如下：

```
Warning: The initial password poses security risks.
The password needs to be changed. Change now? [Y/N]:n

Info: The max number of VTY users is 21, the number of current VTY users online
is 4, and total number of terminal users online is 4.
    The current login time is 2013-12-31 11:22:06.
    The last login time is 2013-12-31 10:24:13 from 10.1.2.2 through SSH.
<SSH Server>
```

STelnet客户端Client002用RSA认证方式连接SSH服务器。

```
<client002> stelnet 10.1.1.1 1025
Trying 10.1.1.1 ...
Press CTRL+K to abort
Connected to 10.1.1.1 ...
The server's public key does not match the one cached before.
The server is not authenticated. Continue to access it?[Y/N]:y
The keyname:192.168.1.182 already exists. Update it? [Y/N]: n

Please input the username: client002
Please select public key type for user authentication [R for RSA/D for DSA/E for ECC] Please select [R/D/E]:r

Info: The max number of VTY users is 21, the number of current VTY users online
is 4, and total number of terminal users online is 4.
    The current login time is 2013-12-31 11:36:06.

<SSH Server>
```

如果登录成功，用户将进入用户视图。如果登录失败，用户将收到**Session is disconnected**的信息。

----结束

检查配置结果

攻击者使用原端口号22登录SSH服务器，不能成功。

```
<client002> stelnet 10.1.1.1
Trying 10.1.1.1 ...
```



```
Press CTRL+K to abort
Error: Failed to connect to the remote host.
```

在SSH服务器端执行**display ssh server status**命令可以查看到STelnet服务已经使能。执行**display ssh user-information**命令可以查看服务器端SSH用户信息。

查看SSH状态信息。

```
[~SSH Server] display ssh server status
SSH Version                            : 2.0
SSH authentication timeout (Seconds)       : 60
SSH authentication retries (Times)         : 3
SSH server key generating interval (Hours) : 0
SSH version 1.x compatibility              : Disable
SSH server keepalive                       : Enable
SFTP server                            : Enable
STelnet server                         : Enable
SNETCONF server                            : Disable
SNETCONF server port(830)                  : Enable
SCP server                             : Disable
SSH server port                          : 1025
ACL name                               : --
ACL number                              : --
ACL6 name                              : --
ACL6 number                             : --
SSH server source address                  : 0.0.0.0
```

查看SSH用户信息。

```
[~SSH Server] display ssh user-information
------------------------------------------------------------------------------
User Name            : client001
Authentication type   : password
User public key name  : --
User public key type  : --
Sftp directory        : flashcfcard:
Service type          : stelnet

User Name            : client002
Authentication type   : rsa
User public key name  : --
User public key type  : --
Sftp directory        : flashcfcard:
Service type          : stelnet
------------------------------------------------------------------------------
Total 2, 2 printed
```

配置脚本

- SSH服务器的配置脚本

```
#
sysname SSH Server
#
rsa peer-public-key rsakey001
 public-key-code begin
 3082010A
  02820101
    00A4BAB8 B964077E F7657F7F E4BE1DE8 71EE1707 E4EE2864 2D06FBE0 BFC1CB52
    F99B7A99 0132B709 3F841CA2 3544B8B2 6EE0A9ED 04B19FE3 FB3DA86D BE68FFE2
    2303108D BDC24B80 A1793A08 FDA0B6C1 13C31EA5 298EC9B1 2B0BC8BD 32CFF896
    29F8CA98 8B1724AF 5DA8A390 20906ADE 6A8AD77D 6234F0C8 DC965BA0 1771D9C0
    A89ED49B 5ECF7EE2 D5997527 FC87FE03 E51658C1 0996DFDF DC456376 2FA4B268
    4345131D 431419D2 DD5E4003 6A7D3295 145F3175 22E80686 E6B39A05 799D6BCF
    A78F69B6 BC2D0836 F5013421 77D68B89 A9EC182A 04B87BE3 500FCE14 9C95CF78
    75704359 0C70FD60 1EFC0B99 32F02142 4CE781E4 36A60BFC 2CBD07F6 9E700CEE
    4D
  0203
    010001
```



```
 public-key-code end
 peer-public-key end
#
aaa
 local-user client001 password irreversible-cipher $1d$v!=.5/:(q-$xL=\K
+if"'S}>k7vGP5$_ox0B@ys7.'DBHL~3*aN$
 local-user client001 service-type ssh
 local-user client001 level 3
#
ssh server port 1025
stelnet server enable
ssh user client001
ssh user client001 authentication-type password
ssh user client001 service-type stelnet
ssh user client002
ssh user client002 authentication-type rsa
ssh user client002 assign rsa-key rsakey001
ssh user client002 service-type stelnet
ssh server-source all-interface
#
ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
ssh server hmac sha2_256 sha2_512
ssh server key-exchange dh_group_exchange_sha256 dh_group16_sha512
ssh server publickey rsa_sha2_256 rsa_sha2_512
ssh server dh-exchange min-len 3072
#
user-interface vty 0 4
 authentication-mode aaa
 protocol inbound ssh
#
return
```

- SSH客户端Client001的配置脚本

```
#
sysname client001
#
ssh client first-time enable
#
ssh client cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
ssh client hmac sha2_256 sha2_512
ssh client key-exchange dh_group_exchange_sha256 dh_group16_sha512
ssh client publickey rsa_sha2_256 rsa_sha2_512
#
return
```

- SSH客户端Client002的配置脚本

```
#
sysname client002
#
ssh client first-time enable
#
ssh client cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
ssh client hmac sha2_256 sha2_512
ssh client key-exchange dh_group_exchange_sha256 dh_group16_sha512
ssh client publickey rsa_sha2_256 rsa_sha2_512
#
return
```

1.1.3.6.5 举例：配置通过 STelnet 登录设备示例（基于 RADIUS 认证）

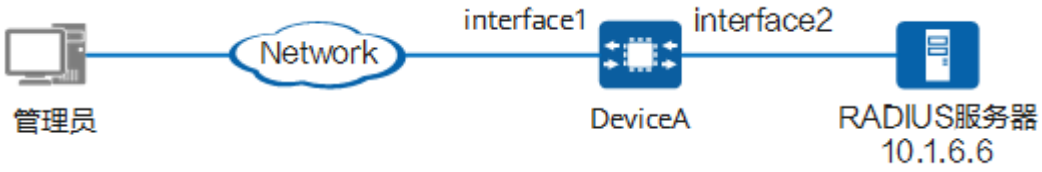
组网需求

网络管理员希望通过安全的方式远程登录设备，同时对网络的安全性要求较高，希望网络不受未授权访问的干扰。可以通过配置基于RADIUS认证的STelnet满足用户的需求。

如**图 通过STelnet登录设备（基于RADIUS认证）组网图**所示，DeviceA为SSH服务器，与RADIUS服务器之间路由可达。RADIUS服务器的IP地址为10.1.6.6/24，认证端口号为1812。

图3-4 通过STelnet登录交换机（基于RADIUS认证）组网图

图 1-18 通过 STelnet 登录设备（基于 RADIUS 认证）组网图



说明

本例中interface1和interface2分别代表GigabitEthernet和GigabitEthernet。

配置思路

采用如下思路配置通过STelnet登录设备（基于RADIUS认证）：

1. 配置设备接口相关参数。
2. 在SSH服务器端生成本地密钥对，实现在服务器端和客户端进行安全地数据交互。
3. 配置STelnet协议，实现用户可以通过STelnet登录设备。
4. 配置ACL规则，保证只有满足该规则的用户才能登录设备。
5. 配置RADIUS协议，实现RADIUS认证。用户通过STelnet登录设备时使用RADIUS服务器上配置的用户名和密码，从而保证用户登录的安全性。
6. 配置RADIUS服务器。

配置注意事项

- 配置STelnet登录设备前，用户终端应该已安装SSH服务器登录软件。
- 请确保用户终端和登录的设备及RADIUS服务器之间均路由可达。
- 域被配置成全局默认管理域之后，管理用户的用户名中携带该域名或者不携带域名时，会使用全局默认管理域下的AAA配置信息。
- 请确保已在RADIUS服务器上配置了用户。本例中假设RADIUS服务器上已配置了用户**admin123@huawei.com**（用户名@域名），其密码为**YsHsjx_202206**。
- 如果RADIUS服务器不接受包含域名的用户名，还需要配置命令**undo radius-server user-name domain-included**使设备向RADIUS服务器发送的报文中的用户名不包含域名。

操作步骤

步骤1 配置接口的相关参数。

配置接口IP地址。

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceA
```



```
[*DeviceA] commit
[~DeviceA] vlan batch 10 20
[*DeviceA] interface gigabitethernet 1/0/1
[*DeviceA-GigabitEthernet1/0/1] portswitch
[*DeviceA-GigabitEthernet1/0/1] port link-tpye trunk
[*DeviceA-GigabitEthernet1/0/1] port trunk allow-pass vlan 10
[*DeviceA-GigabitEthernet1/0/1] quit
[*DeviceA] interface gigabitethernet 1/0/2
[*DeviceA-GigabitEthernet1/0/2] portswitch
[*DeviceA-GigabitEthernet1/0/2] port link-tpye trunk
[*DeviceA-GigabitEthernet1/0/2] port trunk allow-pass vlan 20
[*DeviceA-GigabitEthernet1/0/2] quit
[*DeviceA] interface vlanif 10
[*DeviceA-Vlanif10] ip address 10.1.1.2 255.255.255.0
[*DeviceA-Vlanif10] quit
[*DeviceA] interface vlanif 20
[*DeviceA-Vlanif20] ip address 10.1.6.2 255.255.255.0
[*DeviceA-Vlanif20] quit
[*DeviceA] commit
```

步骤2 配置STelnet登陆。

在服务器端生成本地密钥对。

```
[~DeviceA] rsa local-key-pair create
The key name will be:Host
The range of public key size is (2048, 4096).
NOTE: Key pair generation will take a short while.
Please input the modulus [default = 3072]:3072
[*DeviceA] commit
```

配置SSH服务器的VTY用户界面。

```
[~DeviceA] user-interface vty 0 4
[~DeviceA-ui-vty0-4] authentication-mode aaa
[~DeviceA-ui-vty0-4] protocol inbound ssh
[DeviceA-ui-vty0-4] user privilege level 3
[~DeviceA-ui-vty0-4] quit
[*DeviceA] commit
```

配置SSH服务器的公钥算法、加密算法、密钥交换算法列表、HMAC认证算法和最小密钥长度。

```
[~DeviceA] ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
[~DeviceA] ssh server hmac sha2_256 sha2_512
[~DeviceA] ssh server key-exchange dh_group_exchange_sha256 dh_group16_sha512
[~DeviceA] ssh server publickey rsa_sha2_256 rsa_sha2_512
[*DeviceA] commit
```

在服务器端创建SSH用户，并配置认证方式为Password认证。

```
[~DeviceA] ssh user admin123
[*DeviceA] ssh user admin123 authentication-type password
[*DeviceA] commit
```

说明

如果需要配置多个SSH用户使用Password认证方式，可以执行命令**ssh authentication-type default password**将SSH用户的缺省认证方式指定为Password认证。配置完成后，不用再对每一个SSH用户配置认证方式和服务类型，从而简化配置，提高效率。

使能STelnet功能，并配置用户的服务类型为STelnet。

```
[~DeviceA] stelnet server enable
[*DeviceA] ssh server-source all-interface
[*DeviceA] ssh user admin123 service-type stelnet
[*DeviceA] commit
```

步骤3 在DeviceA上配置ACL规则允许管理员登录。


```
[DeviceA] acl 2000
[DeviceA-acl4-basic-2000] rule permit source 10.137.217.10 0
[DeviceA-acl4-basic-2000] quit
[DeviceA] ssh server acl 2000
[DeviceA] commit
```

说明

采用ACL方式配置Telnet终端服务的配置为可选配置。

步骤4 配置RADIUS认证。

配置RADIUS服务器模板，实现与RADIUS服务器的通信。

```
[~DeviceA] radius-server template 1
[*DeviceA-radius-1] radius-server authentication 10.1.6.6 1812
[*DeviceA-radius-1] radius-server shared-key cipher Huawei@123456789
[*DeviceA-radius-1] quit
[*DeviceA] commit
```

配置AAA认证方案，指定认证方式为RADIUS。

```
[~DeviceA] aaa
[~DeviceA-aaa] authentication-scheme auth1
[*DeviceA-aaa-authen-auth1] authentication-mode radius
[*DeviceA-aaa-authen-auth1] quit
```

创建域，并在域下引用AAA认证方案及RADIUS服务器模板。

```
[*DeviceA-aaa] domain huawei.com
[*DeviceA-aaa-domain-huawei.com] authentication-scheme auth1
[*DeviceA-aaa-domain-huawei.com] accounting-scheme acc1
[*DeviceA-aaa-domain-huawei.com] radius-server 1
[*DeviceA-aaa-domain-huawei.com] quit
[*DeviceA-aaa] quit
[*DeviceA] commit
```

配置huawei.com为全局默认管理域，这样管理员登录设备时就不需要输入域名。

```
[~DeviceA] domain huawei.com admin
[*DeviceA] commit
```

步骤5 配置RADIUS服务器。

配置步骤包括：添加设备、添加用户、配置授权用户级别为3。

----结束

检查配置结果

客户端通过OpenSSH软件登录SSH服务器。进入Windows的命令行提示符，执行OpenSSH命令，通过STelnet方式访问设备。在登录界面根据提示输入RADIUS服务器上配置的用户名**admin1234**和密码**YsHsjx_202206**，并按Enter键，验证通过，用户成功通过Telnet登录到DeviceA。（以下显示仅为示意）

```
C:\Documents and Settings\Administrator>ssh admin1234@10.1.1.2
Enter passphrase for key 'C:\Users\User/.ssh/id_rsa':
Enter password:

Warning: Negotiated key exchange algorithm and identity key for server authentication are not safe. It is recommended that you disable the insecure algorithm or upgrade the client.

Warning: The initial password poses security risks.
The password needs to be changed. Change now? [Y/N]:n
Info: The max number of VTY users is 21, the number of current VTY users online is 1, and total number of terminal users online is 2.
    The current login time is 2022-09-28 12:07:34.
```



```
    The last login time is 2022-09-28 06:44:35 from 172.16.0.1 through SSH.
    The last login failure time is 2022-09-28 11:59:21 from 172.16.0.1 through SSH. Consecutive login
failures since the last successful login: 3.
<DeviceA>
```

配置脚本

```
#
sysname DeviceA
#
radius-server template 1
 radius-server shared-key cipher %+%##!!!!!!!!!"!!!!"!!!!*!!!!SKvr${[Fs.3t@/5k|BENhEu>W(3\~XG!!D;!!!!!2jp5!!!!!!
A!!!!3"pK8qv!}9M#(4$jGWvQF/R[CNe/+:W^jk8HUe&W%+%#
 radius-server authentication 10.1.6.6 1812 weight 80
#
aaa
 authentication-scheme auth1
  authentication-mode radius
 accounting-scheme acc1
  accounting-mode radius
 domain huawei.com
  authentication-scheme auth1
  accounting-scheme acc1
  radius-server 1
#
domain huawei.com admin
#
vlan batch 10 20
#
interface Vlanif10
 ip address 10.1.1.2 255.255.255.0
#
interface Vlanif20
 ip address 10.1.6.2 255.255.255.0
#
interface GigabitEthernet
 portswitch
 port link-type trunk
 port trunk allow-pass vlan 10
#
interface GigabitEthernet
 portswitch
 port link-type trunk
 port trunk allow-pass vlan 20
#
stelnet server enable
ssh user admin1234
ssh user admin1234 authentication-type password
ssh user admin1234 service-type stelnet
ssh server-source all-interface
ssh server acl 2000
#
user-interface vty 0 4
 authentication-mode aaa
 user privilege level 3
 protocol inbound ssh
#
ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
ssh server hmac sha2_256 sha2_512
ssh server key-exchange dh_group_exchange_sha256 dh_group16_sha512
ssh server publickey rsa_sha2_256 rsa_sha2_512
ssh server dh-exchange min-len 3072
#
return
```


1.1.3.7 维护设备命令行界面

清除在线用户

当用户需要将某个登录用户与设备的连接断开时，可以清除指定的在线用户。

执行命令**kill user-interface** { *ui-number* | *ui-type ui-number1* }，清除在线用户。

用户可以执行命令**display users**，查看当前设备上的用户登录信息。

锁定用户配置权限

在多用户同时登录系统进行配置时，有可能会出现配置冲突的情况。为了避免业务出现异常，可以配置权限互斥功能，保证同一时间只有一个用户可以进行配置。

- 方式一：基于会话锁定用户配置权限
  a. 锁定配置权限给当前操作用户。
  configuration exclusive
  锁定用户配置权限后，可以显式地获取独享的配置权限，其他用户无法再获取到配置权限。
  说明
  - 此命令可用于所有视图。
  - 可以执行命令**display configuration-occupied user**，查看当前锁定配置集用户的信息。
  - 如果配置权限已经被锁定，则再次锁定会返回提示信息。
  b. 进入系统视图。
  system-view
  c. （可选）设置自行解锁时间间隔。
  configuration-occupied timeout *timeout-value*
  设置锁定配置集权限用户在无配置命令下发的情况下，允许锁定的最长时间间隔，超过这个时间间隔系统就自行解锁，其他用户可以正常配置。
  缺省情况下，锁定间隔为30秒。
  d. 提交配置。
  commit
- 方式二：基于用户名锁定配置
  设备允许多用户接入，对设备进行管理。这些用户可以是控制器，也可以是其他类型用户。在控制器部署的场景下，如果非控制器用户登录设备并修改配置，可能会出现控制器部署的配置和设备上的配置不一致的情况。执行此命令可以配置指定控制器用户锁定设备系统配置，避免设备与控制器的配置不一致。
  多用户共同管理同一台设备时，可以指定用户名锁定设备，仅允许使用该用户名登录的用户对设备的配置进行修改，防止其他用户修改配置。
  a. 进入系统视图。
  system-view
  b. 使用指定用户名锁定系统配置。
  configuration exclusive by-user-name *user-name*
  缺省情况下，没有锁定系统配置。

说明

- 同一时刻只能一个用户名锁定系统配置。
- 只有管理级别的用户才能执行锁定和解除锁定配置操作。
- 系统配置数据被指定用户名锁定后，系统仅允许使用该用户名登录的用户执行配置操作，其他用户执行的配置操作无法生效。如果想要其他用户的配置生效，须先执行命令**undo configuration exclusive by-user-name** *user-name*解除配置锁定，然后才能执行配置操作使配置生效。
- 执行解除配置锁定操作时，**undo configuration exclusive by-user-name** *user-name*命令中的*user-name*必须为锁定配置的用户名。
- 可以执行命令**display configuration exclusive by-user-name**查看当前系统中基于用户名加锁或解锁系统配置的锁定信息。

c. 提交配置。

```
commit
```

锁定用户界面

当用户需要暂时离开操作终端时，为防止未授权的用户操作该终端界面，可以锁定当前用户终端界面。

1. 执行命令**lock**，锁定用户终端界面。
2. 根据系统提示，输入锁定的密码，并确认密码。

```
<HUAWEI> lock
Enter Password:
Confirm Password:
Info: The terminal is locked.
```

用户输入命令**lock**后，系统提示输入两次屏保密码，如果两次输入的密码相同，则锁定当前用户界面成功。

系统锁定后，如果想再次进入系统，必须先按“Enter”键，此时提示输入登录密码，用户输入正确的登录密码才可以解除锁定进入系统。

说明

开启弱密码字典维护功能后，弱密码字典中定义的密码（可以通过命令**display security weak-password-dictionary**查看）不能在该命令中配置。

1.1.3.8 登录设备命令行界面常见配置错误

1.1.3.8.1 通过 Telnet 登录 Telnet Server 失败

故障现象

通过Telnet方式登录Telnet服务器失败。

操作步骤

步骤1 查看登录设备的用户数是否到达了上限。

从Console口登录到设备，执行命令**display users**，查看当前的VTY通道是否全部被占用。可以先执行命令**display user-interface maximum-vty**，查看当前VTY通道允许的最大用户数。

如果当前的用户数已经达到上限，可以执行命令**user-interface maximum-vty 21**，将VTY通道允许的最大用户数扩展到21个。

步骤2 查看设备上VTY类型用户界面视图下是否配置了ACL。

在Telnet服务器端上执行命令**user-interface vty**进入用户界面视图，执行命令**display this**，查看VTY用户界面是否配置了ACL限制，如果配置了ACL限制，请记录该ACL编号。

在Telnet服务器端上执行命令**display acl** *acl-number*，查看该访问控制列表中是否**deny**了Telnet客户端的地址。如果**deny**客户端的IP地址，则在ACL视图下，执行命令**undo rule** *rule-id*，删除**deny**规则，再执行命令**rule permit source** *source-ip-address source-wildcard*，修改访问控制列表**permit**客户端的IP地址访问。

步骤3 查看VTY类型用户界面视图下允许接入的协议配置是否正确。

在Telnet服务器端上执行命令**user-interface vty**进入用户界面视图，执行命令**display this**，查看VTY用户界面的protocol inbound是否为telnet或者all（在使用设备出厂缺省配置文件的情况下，用户界面支持SSH。若设备使用的配置文件中没有进行protocol inbound的相关配置，用户界面支持所有协议类型。）。如果不是，执行命令**protocol inbound** { **telnet** | **all** }修改配置，允许telnet类型用户接入设备。

步骤4 查看用户界面视图下是否设置登录认证。

- 如果使用命令**authentication-mode password**配置了VTY通道下的登录认证方式为**password**，则必须设置本地验证密码，并在登录时输入此密码。
- 如果使用命令**authentication-mode aaa**设置认证方式为**aaa**，则必须使用命令**local-user** *user-name* **password**创建AAA本地用户，并配置用户服务类型为Telnet。

----结束

1.1.3.8.2 通过 STelnet 登录 SSH Server 失败

故障现象

通过STelnet方式登录SSH服务器失败。

操作步骤

步骤1 查看SSH服务器端的SSH服务是否启动。

通过Console口或Telnet方式登录SSH服务器端，执行命令**display ssh server status**，查看SSH服务器端配置信息。

如果STelnet没有使能，执行如下命令**stelnet server enable**，使能SSH服务器端的STelnet服务。

步骤2 在SSH服务器端上查看VTY类型用户界面视图下允许接入的协议配置是否正确。

在SSH服务器端上执行命令**user-interface vty**进入用户界面视图，执行命令**display this**，查看VTY用户界面的protocol inbound是否为ssh或者all（在使用设备出厂缺省配置文件的情况下，用户界面支持SSH。若设备使用的配置文件中没有进行protocol inbound的相关配置，用户界面支持所有协议类型。）。如果不是，执行命令**protocol inbound** { **ssh** | **all** }修改配置，允许STelnet类型用户接入设备。

步骤3 查看SSH服务器端上是否配置了SSH用户。

执行命令**display ssh user-information**，查看SSH用户的配置信息。如果不存在配置信息，请在系统视图下执行命令**ssh user**、**ssh user authentication-type**和**ssh user service-type**，新建SSH用户并配置SSH用户的认证方式和SSH用户的服务方式。

步骤4 查看登录SSH服务器端的用户数是否到达了上限。

从Console口登录到设备，执行命令**display users**，查看当前的VTY通道是否全部被占用。可以先执行命令**display user-interface maximum-vty**，查看当前VTY通道允许的最大用户数。

如果当前的用户数已经达到上限，可以执行命令**user-interface maximum-vty 21**，将VTY通道允许的最大用户数扩展到21个。

步骤5 查看SSH服务器端上VTY类型用户界面下是否绑定了ACL。

在SSH服务器端上执行命令**user-interface vty**进入SSH用户会使用的界面视图，执行命令**display this**，查看VTY用户界面是否配置了ACL限制，如果配置了ACL限制，请记录该ACL编号。

在SSH服务器端上执行命令**display acl** *acl-number*，查看该访问控制列表中是否**deny**了STelnet客户端的地址。如果**deny**客户端的IP地址，则在ACL视图下，执行命令**undo rule** *rule-id*，删除**deny**规则，再执行命令**rule permit source** *source-ip-address soucer-wildcard*，修改访问控制列表**permit**客户端的IP地址访问。

步骤6 查看SSH客户端和服务器上SSH版本信息。

在SSH服务器上执行命令**display ssh server status**，查看SSH版本信息。

步骤7 查看SSH客户端是否使能了首次登录功能。

在SSH客户端的系统视图下执行命令**display this**，查看SSH客户端是否使能SSH客户端首次登录功能。

如果没有使能SSH客户端首次登录功能，则STelnet客户端第一次登录SSH服务器时，由于对SSH服务器的RSA公钥有效性检查失败，而导致登录服务器失败。执行命令**ssh client first-time enable**使能SSH客户端首次登录功能。

----结束

1.1.4 文件系统管理配置

说明

- 在对设备进行版本文件下载等文件操作过程中，请保持设备的正常供电。否则可能会引起文件损坏或文件系统损坏，从而造成设备存储介质损坏或设备不能正常启动等问题。

1.1.4.1 文件系统简介

文件系统

文件系统是指对存储器中文件、目录的管理，包括创建、删除、修改文件和目录，以及显示文件的内容等。

存储器

设备支持的存储器为Flash。

文件的命名规则

字符串形式，不支持空格，长度范围是1～255，区分大小写。文件名有两种表示方式：文件名、路径+文件名。

- 文件名

  如果直接使用文件名，则表示当前工作路径下的文件。

- 路径+文件名

  格式为**drive/path/filename**，使用这种命名方式可以唯一的标识指定路径下的文件。

  drive是设备中的存储器。

  path是指存储器中目录以及子目录，即路径。目录名使用的字符不可以是空格、“~”、“*”、“/”、“\”、“:”、“'”和“"”等字符，区分大小写。

  设备支持的路径可以是绝对路径也可以是相对路径。指定根目录（指定**drive**）的路径是绝对路径，相对路径有相对于根目录（即当前的存储器目录）的路径和相对于当前工作路径的路径，路径以“/”开头，则表示相对于根目录的路径。

  - 若路径为“cfcard:/my/test/”，这是绝对路径。
  - 若路径为“/selftest/”，表示根目录下的selftest目录，这是相对于根目录的相对路径。
  - 若路径为“selftest/”，表示当前工作路径下的selftest目录，这是相对于当前工作路径的相对路径。

  例如：**dir cfcard:/my/test/mytest.txt**，查看**cfcard:/my/test/**路径下的**mytest.txt**文件的信息，这是一种绝对路径。

  如果用相对于根目录的路径，则可以使用命令：**dir /my/test/mytest.txt**。

  如果用相对于当前工作路径的路径（若当前工作路径是**cfcard:/my/**），则使用命令**dir test/mytest.txt**。

  说明

  - 文件名在文件操作命令格式中统一用**filename**表示。
  - 目录在文件操作命令中统一用**directory**表示，目录的格式即为**drive + path**。

文件列表信息

执行命令**dir**可以查看系统的文件列表信息。

```
<HUAWEI> dir
Directory of cfcard:/

 Idx  Attr     Size(Byte)  Date        Time       FileName
   0  dr-x             -  Nov 11 2019 20:16:35   $_checkpoint
   1  dr-x             -  Nov 06 2019 15:51:57   $_install_mod
   2  dr-x             -  Oct 12 2019 18:12:15   $_license
   3  dr-x             -  Oct 12 2019 18:12:26   $_security_info
   4  dr-x             -  Nov 11 2019 20:16:31   $_startup
   5  dr-x             -  Nov 11 2019 20:15:06   $_system
   6  -rw-        14,940  Nov 11 2019 17:56:29   SPH001.PAT
   7  -rw-   572,847,476  Oct 21 2019 15:21:23   software.cc
   8  -rw-        34,505  Nov 11 2019 20:01:07   device.sys
   9  drwx             -  Nov 11 2019 21:01:39   logfile

2,994,228 KB total (801,664 KB free)
```

命令**dir**查看的部分文件列表信息详见**表1-35**。

表 1-35 文件信息描述

项目	描述
$_checkpoint	用于保存配置回退点信息的文件夹。
$_install_mod	存放MOD（动态模块包）的目录。
$_license	用户激活过的license文件备份的目录。
$_security_info	保存AAA用户历史信息的目录。
$_startup	存放下次启动配置文件的目录。
$_system	Linux系统自带的目录，存放系统使用的脚本。
.pat/.PAT	补丁文件。
**.cc	软件版本文件。
device.sys	系统硬件配置文件。
logfile	日志信息文件。

1.1.4.3 设备支持的文件系统管理方式

设备在进行文件管理的过程中，可以分别充当服务器和客户端的角色：

- 设备作为服务器：可以从终端访问设备，实现对本设备文件的管理，以及与终端间的文件传输操作。
- 设备作为客户端访问其他设备（服务器）：可以实现管理其他设备上的文件，以及与其他设备间进行文件传输操作。

对于TFTP方式，设备只支持客户端功能；对于FTP、SFTP、SCP方式，设备均支持服务器与客户端功能。

各种文件系统管理方式的应用场景，优缺点如表1-36所示，用户可以根据需求选择其中一种方式。

表 1-36 文件系统管理方式

文件系统管理方式	应用场景	优点	缺点
本地进行文件操作	通过Console口、Telnet或STelnet方式登录设备，对存储器、目录和文件进行管理。	对目录和文件的管理直接通过登录设备完成，方便快捷。	只是对本设备进行文件操作，无法进行文件的传输。

文件系统管理方式	应用场景	优点	缺点
FTP（File Transfer Protocol）	适用于对网络安全性要求不是很高的文件传输场景中，广泛用于版本升级等业务中。	• 配置较简单，支持文件传输以及文件目录的操作。 • FTP可以在两个不同文件系统主机之间传输文件。 • 具有授权和认证功能。	明文传输数据，存在安全隐患。
TFTP（Trivial File Transfer Protocol）	在网络条件良好的实验室局域网中，可以使用TFTP进行版本的在线加载和升级。适用于客户端和服务器之间，不需要复杂交互的环境。	TFTP所占的内存要比FTP小。	• 设备只支持TFTP客户端功能。 • TFTP只支持文件传输，不支持交互。 • TFTP没有授权和认证，且是明文传输数据，存在安全隐患，易于网络病毒传输以及被黑客攻击。
SFTP（Secure File Transfer Protocol）	适用于网络安全性要求高的场景，目前被广泛用于日志下载、配置文件备份等业务中。	• 数据进行了严格加密和完整性保护，安全性高。 • 支持文件传输及文件目录的操作。	配置较复杂。
SCP（Secure Copy Protocol）	适用于网络安全性要求高，且文件上传下载效率高的场景。	• 数据进行了严格加密和完整性保护，安全性高。 • 客户端与服务器连接的同时完成文件的上传下载操作（即连接和拷贝操作使用一条命令完成），效率较高。	配置较复杂（与SFTP方式的配置非常类似），且不支持交互。

直接登录系统、FTP、TFTP方式，理解和配置都比较简单，下面主要介绍SFTP和SCP方式。

SFTP 方式

SFTP是SSH协议的一部分，利用SSH协议提供的安全通道，使得远程用户可以安全地登录设备进行文件管理和文件传输等操作，为数据传输提供了更高的安全保障。同时设备支持客户端的功能，用户可以从本地设备安全登录到远程SSH服务器上，进行文件的安全传输。

SCP 方式

SCP也是SSH协议的一部分，是基于SSH协议的远程文件拷贝技术，实现文件的拷贝，包括上传和下载。SCP文件拷贝命令简单易用，提高了网络维护的效率。

1.1.4.4 本地文件操作

1.1.4.4.1 本地文件操作

前提条件

在配置本地文件操作之前，需完成以下任务：

- 终端与设备之间路由可达。
- 已从终端登录到设备。

操作步骤

- **对目录进行操作**

  表 1-37 目录操作

操作项目	命令	说明
查看当前所处的目录	**pwd**	-
改变当前所处的目录	**cd** [*directory*]	-
显示目录中的文件和子目录的列表	**dir** [**/all**] [*filename* | **/all-filesystems**]	-
显示备区目录中所有文件	**display backup-file**	-
创建目录	**mkdir** *directory*	-
删除目录	**rmdir** *directory*	• 被删除的目录必须为空目录。 • 目录被删除后，无法从回收站中恢复，原目录下被删除的文件也彻底从回收站中删除。

- **对文件进行操作**

表 1-38 文件操作

操作项目	命令	说明
显示文件的内容	**more** *file-name* [*offset*]	用户还可以执行命令**tail** *file-name* [*line*]查看文件中最后指定行的内容。
拷贝文件	**copy** *source-filename destination-filename* [**all**]	• 在拷贝文件前，确保存储器有足够的空间。 • 若目标文件名与已经存在的文件名重名，将提示是否覆盖。
将备区文件拷贝至指定路径	**copy backup-file file-name** *scrFile* **path** *desFile* **copy backup-file all path** *desFile*	-
移动文件	**move** *source-filename destination-filename*	若目标文件名与已经存在的文件名重名，将提示是否覆盖。
重新命名文件	**rename** *old-name new-name*	-
压缩文件或目录	**zip** *source-filename destination-filename*	-
解压缩文件	**unzip** *source-filename destination-filename*	-
删除文件	**delete** [**/unreserved**] [**/quiet**] *filename* [**all**]	此命令不能删除目录。 **须知** 如果使用参数**/unreserved**，则删除后的文件不可恢复。
恢复删除的文件	**undelete** *filename*	执行**delete**命令（不带**/unreserved**参数）后，文件将被放入回收站中。可以执行此命令恢复回收站中被删除的文件。
彻底删除回收站中的文件	**reset recycle-bin** [**/f** | *filename*]	需要永久删除回收站中的文件时，可进行此操作。
运行批处理文件或VSL（VRP Shell Languages）脚本文件	**execute** *filename* [*parameter* &<1-8>]	一次进行多项处理时，可在系统视图中进行此操作。编辑好的批处理文件要预先保存在设备的存储器中。

----结束

1.1.4.4.2 举例：本地文件操作

组网需求

用户通过Console口、Telnet或STelnet方式登录设备，需要对设备上的文件进行以下操作：

- 查看当前目录下的文件及子目录。
- 创建目录**test**，将文件**vrpcfg.zip**复制至**test**目录下，并命名为**backup.zip**。
- 查看**test**目录下的文件。

操作步骤

步骤1 查看当前目录下的文件及子目录。

```
<HUAWEI> system-view
[~HUAWEI] sysname Device
[*HUAWEI] commit
[~Device] quit
<Device> dir
Directory of cfcard:/

  Idx  Attr     Size(Byte)  Date        Time       FileName
    0  -rw-           889  Mar 01 2019 14:41:56   private-data.txt
    1  -rw-         6,311  Feb 17 2019 14:05:04   backup.cfg
    2  -rw-         2,393  Mar 06 2019 17:20:10   vrpcfg.zip
    3  -rw-           812  Nov 12 2019 15:43:10   hostkey
    4  drw-             -  Mar 01 2019 14:41:46   compatible
    5  -rw-           540  Nov 12 2019 15:43:12   serverkey
...
670,092 KB total (569,904 KB free)
```

步骤2 创建目录**test**，将文件**vrpcfg.zip**复制至**test**目录下，并命名为**backup.zip**。

创建目录**test**。

```
<Device> mkdir test
Info: Create directory cfcard:/test/.....Done.
```

复制**vrpcfg.zip**至**test**目录下，并命名为**backup.zip**。

```
<Device> copy vrpcfg.zip cflash:/test/backup.zip
Info: Are you sure to copy cflash:/vrpcfg.zip to cflash:/test/backup.zip?[Y/N]:y
100%  complete
Info: Copied file cflash:/vrpcfg.zip to cflash:/test/backup.zip...Done.
```

说明

如果不指定目标文件名，则目标文件名默认为源文件名，即目标文件和源文件同名。

----结束

检查配置结果

进入**test**目录。

```
<Device> cd test
```

查看当前的工作路径。

```
<Device> pwd

cfcard:/test/
```


查看**test**目录下的文件。

```
<Device> dir
Directory of cfcard:/test/

 Idx  Attr     Size(Byte)  Date        Time       FileName
   0  -rw-          2,399  Mar 12 2012 11:16:44   backup.zip

670,092 KB total (569,900 KB free)
```

配置脚本

```
#
sysname Device
#
return
```

1.1.4.5 通过 FTP 进行文件操作

1.1.4.5.1 配置设备作为 FTP 服务器

前提条件

用户可以使用FTP协议在本地与远程终端之间进行文件操作，在版本升级等文件业务操作中此协议广泛应用。

在通过FTP进行文件操作之前，需完成以下任务：

- 终端与设备之间路由可达。
- 终端支持FTP客户端软件。
- 已执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

背景信息

须知

使用FTP协议存在安全风险，建议使用SFTP V2或SCP方式进行文件操作。

FIPS模式下，不支持使用FTP协议进行文件操作。

通过FTP进行文件操作的配置流程如**表1-39**所示。序号1、2、3、4之间没有严格的配置顺序。

表 1-39 通过 FTP 进行文件操作的配置流程

序号	配置任务名称	配置任务说明
1	**配置FTP服务器功能及参数**	包括FTP服务器的使能及参数配置：端口号、源地址、超时断连时间。

序号	配置任务名称	配置任务说明
2	配置FTP本地用户	包括配置本地用户的服务类型、用户级别及FTP用户的授权目录等。
3	（可选）配置FTP访问控制	包括配置ACL规则及FTP基本访问控制列表，提高FTP访问的安全性。
4	（可选）配置IP地址锁定功能	包括IP地址锁定功能的使能及参数配置：配置连续认证失败次数上限、失败时间周期等。
5	用户通过FTP访问设备	从终端通过FTP访问设备。

缺省配置

表 1-40 缺省配置

参数	缺省值
FTP服务器功能	关闭
端口号	21
FTP用户	没有创建本地用户

操作步骤

- **配置FTP服务器功能及参数。**

  表 1-41 配置 FTP 服务器功能及参数

操作步骤	命令	说明
进入系统视图	**system-view**	-
（可选）指定FTP服务器端口号	**ftp** [**ipv6**] **server port** *port-number*	缺省情况下，FTP服务器端口号是21。 如果配置了新的端口号，FTP服务器端先断开当前已经建立的所有FTP连接，然后使用新的端口号开始尝试连接。这样可以有效防止攻击者对FTP服务标准端口的访问。

操作步骤	命令	说明
（可选）指定FTP服务器的最大连接数	**ftp server max-sessions** *max-session-count*	缺省情况下，FTP服务器的最大连接数是15。 该命令对IPv4和IPv6连接均生效。 如果设置的最大连接数小于等于当前已建立的连接数，那么设备将拒绝新的连接请求，但当前连接不会断开。
使能FTP服务器	**ftp** [**ipv6**] **server enable**	缺省情况下，设备的FTP服务器功能是关闭的。
指定FTP服务器的源接口或源地址	• **ftp server-source** { **-a** *ip-address* | **-i** { *interface-type interface-number* | interface-name } } • **ftp ipv6 server-source -a** *ipv6-address* [**-vpn-instance** *vpn-instance-name*] • **ftp** [**ipv6**] **server source all-interface** • **ftp server-source physic-isolate -i** { *interface-type interface-number* | *interface-name* } **-a** *ip-address* • **ftp ipv6 server-source physic-isolate -i** { *interface-type interface-number* | *interface-name* } **-a** *ipv6-address*	指定FTP服务器的源接口或源地址，实现对设备进出报文的过滤，保证安全性。 配置了服务器的源地址后，登录服务器时，所输入的服务器地址必须与该命令中配置的一致，否则无法成功登录。 成功设置接口隔离属性后，只能通过配置的物理口连接设备，即报文只能从配置的物理口上送，通过其他接口上送的报文会被丢弃。
（可选）配置FTP连接空闲时间	**ftp** [**ipv6**] **timeout** *minutes*	缺省情况下，连接空闲时间为10分钟。 在设定的时间内，如果FTP连接始终处于空闲状态时，系统将自动断开FTP连接。
（可选）配置在一定时间内通过FTP登录服务器失败次数的告警上报门限和告警恢复门限	**ftp server login-failed threshold-alarm upper-limit** *report-times* **lower-limit** *resume-times* **period** *period-time*	缺省情况下，在5分钟内发生30次或以上次数的登录失败，即产生告警，在5分钟内登录失败次数小于20，即告警恢复。

操作步骤	命令	说明
（可选）配置单个IP地址通过FTP连接到服务器的最大连接数	**ftp server ip-max-sessions** *ip-max-sessions-num*	缺省情况下，单个IP地址通过FTP方式连接到服务器的最大连接数是15。
提交配置	**commit**	-

说明

- 如果变更端口号前FTP服务已经启动，则不能变更成功。需执行**undo ftp** [**ipv6**] **server**命令关闭FTP服务，再进行端口号变更。
- 当客户端与设备之间的文件操作结束后，请执行**undo ftp** [**ipv6**] **server**命令，及时关闭FTP服务器功能，从而保证设备的安全。

- **配置FTP本地用户**。

  当用户通过FTP进行文件操作时，需要在作为FTP服务器的设备上配置本地用户名及口令、指定用户的服务类型以及可以访问的目录，否则用户将无法通过FTP访问设备。

  表 1-42 配置 FTP 本地用户

操作步骤	命令	说明
进入系统视图	**system-view**	-
进入AAA视图	**aaa**	-
配置本地用户名和密码	**local-user** *user-name* **password** [**cipher** | **irreversible-cipher**] *password*	为充分保证设备安全，请用户定期修改密码。
配置本地用户级别	**local-user** *user-name* **level** *level*	必须将用户级别配置在管理级别，否则FTP连接将无法成功。
配置本地用户的服务类型为FTP	**local-user** *user-name* **service-type ftp**	缺省情况下，本地用户可以使用所有的接入类型。

操作步骤	命令	说明
配置FTP用户的授权目录	**local-user** *user-name* **ftp-directory** *directory* [**access-permission** { **read-only** | **read-write** }]	缺省情况下，本地用户的FTP目录为空。 当有多个FTP用户且有相同的授权目录时，可以执行**ftp server default-directory** *directory*命令，为FTP用户配置缺省工作目录。此时，不需要通过**local-user** *user-name* **ftp-directory** *directory*命令为每个用户配置授权目录。 根据认证方式的不同，FTP服务器用户分为AAA本地认证用户和远程认证（RADIUS、HWTACACS）用户。 ● 对于本地认证用户，必须使用**local-user** *user-name* **ftp-directory** *directory*命令指定本地用户的FTP工作目录。如果没有指定，则登录失败。若不配置**access-permission**时，本地用户的FTP目录权限默认为read-write。 ● 对于远程认证用户，可以通过HWTACACS和RADIUS服务器为远程用户指定FTP工作目录。如果没有指定，则可使用**set default ftp-directory** *ftp-directory-name*命令指定FTP用户的缺省工作目录。即**set default ftp-directory** *ftp-directory-name*命令只对远程认证（RADIUS、HWTACACS）用户生效。
提交配置	**commit**	-

- **（可选）配置FTP协议的白名单Session-CAR**

  当FTP Server报文发生流量攻击时，FTP Server会话间报文会发生互相抢占带宽的情况。FTP协议的白名单Session-CAR用于对白名单报文通道进行隔离，实现区分会话限速，避免FTP Server会话间报文互相抢占带宽资源。当默认的白名单Session-CAR的带宽参数不满足业务要求时，可以对带宽参数进行调整，灵活适应业务要求。

表 1-43 （可选）配置 FTP 协议的白名单 Session-CAR

操作步骤	命令	说明
进入系统视图	**system-view**	-
配置FTP协议的白名单Session-CAR的带宽参数	**whitelist session-car ftp-server** { **cir** *cir-value* | **cbs** *cbs-value* | **pir** *pir-value* | **pbs** *pbs-value* }*	缺省情况下，FTP协议的白名单Session-CAR带宽的承诺信息速率为100kbit/s，承诺突发尺寸为3000000bytes，峰值信息速率为4000kbit/s，峰值突发尺寸为9000000bytes。
（可选）去使能FTP协议的白名单Session-CAR功能	**whitelist session-car ftp-server disable**	一般情况下不建议关闭该功能。
提交配置	**commit**	-

- **（可选）配置FTP访问控制**。

  ACL是一系列有顺序的规则组的集合，这些规则根据数据包的源地址、目的地址、端口号等来描述。ACL通过规则对数据包进行分类，这些规则应用到路由设备，路由设备根据这些规则判断哪些数据包可以接收，哪些数据包需要拒绝。

  用户可以配置FTP访问控制列表，实现只允许指定的客户端登录到设备，以提高安全性。

  ACL规则：

  - 当ACL的rule配置为**permit**时，则允许匹配该rule规则的其他设备与本设备建立FTP连接。
  - 当ACL的rule配置为**deny**时，则拒绝匹配该rule规则的其他设备与本设备建立FTP连接。
  - 当ACL配置了rule，但来自其他设备的报文没有匹配该rule规则时，则拒绝其他设备与本设备建立FTP连接。
  - 当ACL未配置rule时，则允许任何其他设备与本设备建立FTP连接。

表 1-44 （可选）配置 FTP 访问控制

操作步骤	命令	说明
进入系统视图	**system-view**	-
进入ACL视图	**acl** { [**number**] *basic-acl-number* | **name** *basic-acl-name* }	-

操作步骤	命令	说明
配置ACL规则	**rule** [*rule-id*] [**name** *rule-name*] { **permit** | **deny** } [**fragment-type fragment** | **source** { *source-ip-address* { *source-wildcard* | **0** | *src-netmask* } | **any** } | **time-range** *time-name* | **vpn-instance** *vpn-instance-name* | **logging**] *	-
退回到系统视图	**quit**	-
配置FTP基本访问控制列表	**ftp** [**ipv6**] **acl** { *acl-number* | *name* }	-
提交配置	**commit**	-

- **（可选）配置IP地址锁定功能**。

  用户FTP登录失败后，根据IP地址记录FTP登录失败的次数，当一定时间内失败次数达到阈值后，将IP地址锁定，所有通过该IP地址登录的用户均不能正常连接。

  表 1-45 （可选）配置 IP 地址锁定功能

操作步骤	命令	说明
进入系统视图	**system-view**	-
使能FTP服务器上的客户端IP地址锁定功能	**undo ftp server ip-block disable**	缺省情况下，FTP服务器上的客户端IP地址锁定功能处于使能状态。
配置连续认证失败次数和连续失败时间周期	**ftp server ip-block failed-times** *failed-times* **period** *period*	缺省情况下，锁定用户前的连续认证失败次数为6，连续失败时间周期为5分钟。
配置被锁定用户的自动解锁时间	**ftp server ip-block reactive** *reactive-period*	缺省情况下，被锁定用户的自动解锁时间为5分钟。
退回到系统视图	**quit**	-
解除已锁定的IP地址	**activate ftp server ip-block ip-address** *ip-address* [**vpn-instance** *vpn-name*]	-
提交配置	**commit**	-

- **用户通过FTP访问设备**。

从终端通过FTP访问设备，可以选择使用Windows命令行提示符或第三方软件。此处以Windows命令行提示符为例进行配置。

- 执行Windows命令**ftp** *ip-address*，通过FTP方式访问设备。

  此处输入的IP地址为设备上配置的IP地址，且与用户终端IP地址路由可达。
- 根据提示输入用户名和口令，按Enter键，当出现FTP客户端视图的命令行提示符，如ftp>，此时用户进入了FTP服务器的工作目录。（以下显示信息仅为示意）

```
C:\Windows\System32> ftp 192.168.150.208
连接到 192.168.150.208。
220 FTP service ready.
用户(192.168.150.208:(none)):admin123
331 Password required for admin123.
密码:
230 User logged in.
ftp>
```

- **通过FTP命令进行文件操作**。

  用户访问FTP服务器后，可以通过FTP命令进行文件操作，包括目录操作、文件操作、配置文件传输方式、查看FTP命令在线帮助等。

  说明

  用户的操作权限受限于服务器上对该用户的权限设置。

  以下各操作没有严格顺序，可根据需求选择一种或多种操作项目。

表 1-46 通过 FTP 命令进行文件操作

操作项目	命令	说明
改变服务器上的工作路径	**cd** *pathname*	-
改变服务器的工作路径到上一级目录	**cdup**	-
显示服务器工作路径	**pwd**	-
显示或者改变客户端的工作路径	**lcd** [*directory*]	与**pwd**不同的是，**lcd**命令执行后显示的是客户端的本地工作路径，而**pwd**显示的则是远端服务器的工作路径。
在服务器上创建目录	**mkdir** *remote-directory*	创建的目录可以为字母和数字等的组合，但不可以为<、>、？ 、\、： 等特殊字符。
在服务器上删除目录	**rmdir** *remote-directory*	-

操作项目	命令	说明
显示服务器上指定目录或文件的信息	**dir** [*remote-directory* [*local-filename*]] 或 **ls** [*remote-directory* [*local-filename*]]	• **ls**命令只能显示出目录/文件的名称，而**dir**命令可以查看目录/文件的详细信息，如大小，创建日期等。 • 如果指定远程文件时没有指定路径名称，那么系统将在用户的授权目录下搜索指定的文件。
删除服务器上指定文件	**delete** *remote-filename*	-
上传单个或多个文件	**put** *local-filename* [*remote-filename*] 或 **mput** *local-filenames*	• **put**命令是上传单个文件。 • **mput**命令是上传多个文件。
下载单个或多个文件	**get** *remote-filename* [*local-filename*] 或 **mget** *remote-filenames*	• **get**命令是下载单个文件。 • **mget**命令是下载多个文件。
配置传输文件的数据类型为ASCII模式	**ascii**	二选一 • 缺省情况下，文件传输方式为ASCII模式。 • 传输文本文件使用ASCII方式，传输程序、系统软件、数据库文件等使用二进制模式。
配置传输文件的数据类型为二进制模式	**binary**	
配置文件传输方式为被动方式	**passive**	二选一 缺省情况下，数据传输方式是主动方式。
配置文件传输方式为主动方式	**undo passive**	
查看FTP命令的在线帮助	**remotehelp** [*command*]	-
使能系统的提示功能	**prompt**	缺省情况下，不使能信息提示。
打开verbose开关	**verbose**	如果打开verbose开关，将显示所有FTP响应，包括FTP协议信息，以及FTP服务器返回的详细信息。

- **（可选）更改登录用户。**

  设备可以在不退出FTP客户端视图的情况下，以其他的用户名登录到FTP服务器。所建立的FTP连接，与执行**ftp**命令建立的FTP连接完全相同。

表 1-47 更改登录用户

操作步骤	命令	说明
FTP客户端视图下，更改当前的登录用户	**user** *username*	更改当前的登录用户后，原用户与服务器的连接将断开。

- **断开与FTP服务器的连接**。

  用户可以在FTP客户端视图中选择不同的命令断开与FTP服务器的连接。

  表 1-48 断开与 FTP 服务器连接

操作步骤	命令	说明
终止与服务器的连接，并退回到用户视图	**bye** 或 **quit**	二选一。
终止与服务器的连接，并留在FTP客户端视图	**close** 或 **disconnect**	

----结束

检查配置结果

- 使用**display ftp-server**命令，查看FTP服务器的配置和状态信息。
- 使用**display ftp-users**命令，查看登录的FTP用户信息。
- 使用**display ftp server ip auth-fail information**命令，查看认证失败的客户端IP地址的详细信息，包括首次认证失败时间及认证失败次数等。
- 使用**display ftp server ip-block list**命令，查看因认证失败而被锁定的客户端IP地址及剩余锁定时间。

1.1.4.5.2 配置设备作为 FTP 客户端

前提条件

当前设备作为FTP客户端登录远程FTP服务器，实现文件传输以及对服务器上文件及目录的管理操作。

在配置通过FTP访问其他设备的文件之前，需完成以下任务：

- 当前设备和FTP服务器路由可达。
- 已获取FTP服务器的IP地址、FTP用户名及密码。
- 如果服务器不是使用标准的端口号，则还需获取服务器端设置的端口号。
- 已执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

背景信息

须知

使用FTP协议存在安全风险，建议使用SFTP V2或SCP方式进行文件操作。

FIPS模式下，不支持使用FTP协议。

通过FTP访问其他设备文件的配置流程如**表1-49**所示。

表 1-49 配置设备作为 FTP 客户端访问其他设备的文件配置流程

序号	配置任务名称	配置任务说明
1	（可选）配置FTP客户端源接口或源地址	客户端源地址可以配置为源接口或源IP，达到安全校验的目的。
2	使用FTP连接其他设备进行文件操作（一键式）	二选一。 • 一键式：在连接同时可进行文件的上传和下载。 • 交互式：先建立连接，然后在FTP服务器上进行目录操作、文件操作、配置文件传输方式、查看FTP命令在线帮助等。
	使用FTP连接其他设备进行文件操作（交互式）	
3	（可选）更改登录用户	-
4	断开与FTP服务器的连接	-

操作步骤

- **（可选）配置FTP客户端源接口或源地址。**

  配置源地址需要选择设备上状态稳定的接口，如LoopBack接口。该配置简化了ACL规则的配置，只要将ACL规则的源地址或目的地址指定为该地址，实现对设备进出报文的过滤。

  表 1-50 配置 FTP 客户端源接口或源地址

操作步骤	命令	说明
进入系统视图	**system-view**	-
配置FTP客户端的IPv4源地址或源接口	**ftp client-source** { **-a** *ip-address* | **-i** *interface-type interface-number* }	建议使用Loopback接口的地址。 当配置为LoopBack接口时，一定要为此接口配置IP地址，否则会导致FTP连接建立失败。

操作步骤	命令	说明
配置FTP客户端的IPv6源地址	**ftp ipv6 client-source -a** *ipv6-address* [**-vpn-instance** *ipv6-vpn-instance-name*]	当指定的源地址不存在时可以配置成功，但功能不生效。
提交配置	**commit**	-

- **使用FTP连接其他设备进行文件操作（一键式）。**

  当只需与FTP服务器间进行文件上传或下载时，可以直接在用户视图下执行命令完成文件的传输，但此命令不能执行其他FTP操作。

  表 1-51 一键式命令进行文件操作

操作项目	命令	说明
通过IPv4地址连接FTP服务器	**ftp client-transfile** { **put** | **get** } [**-a** *source-ip4* | **-i** { *interface-type interface-number* | interface-name }] **host-ip** *ip4-address* [**port** *portnumber*] [**vpn-instance** *vpn-instancename* | **public-net**] **username** *user-name* **sourcefile** *localfilename* [**destination** *remotefilename*]	通过IPv4方式连接FTP服务器，并从服务器上下载文件至FTP客户端或者从FTP客户端上传文件至服务器。
通过IPv6地址连接FTP服务器	**ftp client-transfile** { **put** | **get** } **ipv6** [**-a** *source-ip6* | **-i** { *interface-type interface-number* | interface-name }] **host-ip** *ip6-address* [**port** *portnumber*] [**vpn-instance** *ipv6-vpn-name* | **public-net**] **username** *user-name* **sourcefile** *localfilename* [**destination** *remotefilename*]	通过IPv6方式连接FTP服务器，并从服务器上下载文件至FTP客户端或者从FTP客户端上传文件至服务器。

- **使用FTP连接其他设备进行文件操作（交互式）。**
  a. **使用FTP命令连接其他设备**。

在用户视图和FTP客户端视图下，用户均可以使用相应命令访问FTP服务器。

根据服务器端IP地址类型不同，进行如下操作。

表 1-52 使用 FTP 命令连接其他设备（服务器端 IPv4 地址类型）

操作步骤	命令	说明
用户视图下直接建立与IPv4 FTP服务器的连接	**ftp** [**-a** *source-ip-address* | **-i** { *interface-type interface-number* | *interface-name* }] *host-ip* [*port-number*] [**vpn-instance** *vpn-instance-name* | **public-net**]	二选一 FTP客户端视图下建立与FTP服务器的连接时需要先使用**ftp**命令进入FTP客户端视图。
FTP客户端视图下建立与IPv4 FTP服务器的连接	**ftp**	
	open [**-a** *source-ip* | **-i** { *interface-type interface-number* | *interface-name* }] *host-ip-address* [*port-number*] [**vpn-instance** *vpn-instance-name* | **public-net**]	

说明

在访问FTP服务器之前，可以执行命令**set net-manager vpn-instance**，设置默认的VPN实例。执行该命令后，进行FTP操作时所使用的VPN实例即用户配置的默认VPN实例。

ftp命令中指定的源地址优先级高于**ftp client-source**命令中指定源地址的优先级。如果执行命令**ftp client-source**指定了源地址后，又在**ftp**命令中指定了源地址，则采用**ftp**命令中指定的源地址进行通信。**ftp client-source**命令指定的源地址对所有的FTP连接都有效，**ftp**命令指定的源地址只对当前的FTP连接有效。

表 1-53 使用 FTP 命令连接其他设备（服务器端 IPv6 地址类型）

操作步骤	命令	说明
用户视图下直接建立与IPv6 FTP服务器的连接	**ftp ipv6** [**-a** *source-ip6*] *host-ipv6-address* [[**vpn-instance** *ipv6-vpn-instance-name*] | **public-net**] [**-oi** { *interface-type interface-number* | *interface-name* }] [*port-number*]	二选一 FTP客户端视图下建立与FTP服务器的连接时需要先使用**ftp**命令进入FTP客户端视图。
FTP客户端视图下建立与IPv6 FTP服务器的连接	**ftp**	

操作步骤	命令	说明
	open ipv6 [**-a** *source-ip6*] *host-ipv6-address* [**-oi** { *interface-type interface-number* | *interface-name* }] [*port-number*] [**vpn-instance** *vpn-instance* | **public-net**]	

用户访问服务器时，需要经过验证，输入正确的用户名和密码后，方可访问服务器。

b. **通过FTP命令进行文件操作**。

用户访问FTP服务器后，可以通过FTP命令进行文件操作，包括目录操作、文件操作、配置文件传输方式、查看FTP命令在线帮助等。

说明

用户的操作权限受限于服务器上对该用户的权限设置。

以下各操作没有严格顺序，可根据需求选择一种或多种操作项目。

表 1-54 通过 FTP 命令进行文件操作

操作项目	命令	说明
改变服务器上的工作路径	**cd** *pathname*	-
改变服务器的工作路径到上一级目录	**cdup**	-
显示服务器工作路径	**pwd**	-
显示或者改变客户端的工作路径	**lcd** [*directory*]	与**pwd**不同的是，**lcd**命令执行后显示的是客户端的本地工作路径，而**pwd**显示的则是远端服务器的工作路径。
在服务器上创建目录	**mkdir** *remote-directory*	创建的目录可以为字母和数字等的组合，但不可以为<、>、？ 、\、： 等特殊字符。
在服务器上删除目录	**rmdir** *remote-directory*	-

操作项目	命令	说明
显示服务器上指定目录或文件的信息	**dir** [*remote-directory* [*local-filename*]] 或 **ls** [*remote-directory* [*local-filename*]]	• **ls**命令只能显示出目录/文件的名称，而**dir**命令可以查看目录/文件的详细信息，如大小，创建日期等。 • 如果指定远程文件时没有指定路径名称，那么系统将在用户的授权目录下搜索指定的文件。
删除服务器上指定文件	**delete** *remote-filename*	-
上传单个或多个文件	**put** *local-filename* [*remote-filename*] 或 **mput** *local-filenames*	• **put**命令是上传单个文件。 • **mput**命令是上传多个文件。
下载单个或多个文件	**get** *remote-filename* [*local-filename*] 或 **mget** *remote-filenames*	• **get**命令是下载单个文件。 • **mget**命令是下载多个文件。
配置传输文件的数据类型为ASCII模式	**ascii**	二选一 • 缺省情况下，文件传输方式为ASCII模式。 • 传输文本文件使用ASCII方式，传输程序、系统软件、数据库文件等使用二进制模式。
配置传输文件的数据类型为二进制模式	**binary**	
配置文件传输方式为被动方式	**passive**	二选一 缺省情况下，数据传输方式是主动方式。
配置文件传输方式为主动方式	**undo passive**	
查看FTP命令的在线帮助	**remotehelp** [*command*]	-
使能系统的提示功能	**prompt**	缺省情况下，不使能信息提示。
打开verbose开关	**verbose**	如果打开verbose开关，将显示所有FTP响应，包括FTP协议信息，以及FTP服务器返回的详细信息。

操作项目	命令	说明
使能FTP客户端断点续传服务	**ftp client resumable-transfer enable**	缺省情况下，FTP客户端断点续传服务没有使能。该命令在系统视图下生效。

- **（可选）更改登录用户。**

  设备可以在不退出FTP客户端视图的情况下，以其他的用户名登录到FTP服务器。所建立的FTP连接，与执行**ftp**命令建立的FTP连接完全相同。

  表 1-55 更改登录用户

操作步骤	命令	说明
FTP客户端视图下，更改当前的登录用户	**user** *username*	更改当前的登录用户后，原用户与服务器的连接将断开。

- **断开与FTP服务器的连接。**

  用户可以在FTP客户端视图中选择不同的命令断开与FTP服务器的连接。

  表 1-56 断开与 FTP 服务器的连接

操作步骤	命令	说明
终止与服务器的连接，并退回到用户视图	**bye** 或 **quit**	二选一。
终止与服务器的连接，并留在FTP客户端视图	**close** 或 **disconnect**	

----结束

检查配置结果

- 使用**display ftp client**命令，查看设备作为FTP客户端时的源参数。

1.1.4.5.3 举例：配置设备作为 FTP 服务器

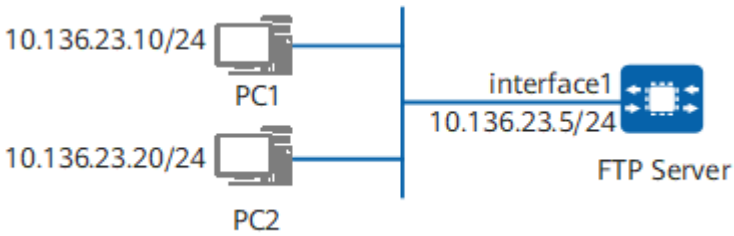
组网需求

如图1-19所示，PC1与设备相连，10.136.23.5是设备的IP地址，设备需要进行升级操作，要求将设备作为FTP服务器，从终端PC1将系统软件上传至设备，且保存当前设备的配置文件到终端进行备份。配置ACL策略，保证只有PC1才能访问FTP服务器。

图 1-19 设备作为 FTP 服务器组网图

说明

本例中interface1代表GigabitEthernet1/0/1。



配置思路

采用如下的思路配置通过FTP进行文件操作：

1. 配置设备的FTP功能及FTP用户信息（包括源地址、用户名及密码、用户级别、服务类型、授权目录）。
2. 配置FTP服务器访问权限。
3. 保存设备当前配置文件。
4. 从终端PC1通过FTP连接设备。
5. 将系统软件上传至设备以及配置文件备份至PC1。

配置注意事项

当网络所处环境不足够安全时，我们建议选择安全的密码认证方式/加密认证算法/协议。安全的举例请参见：**配置设备作为SFTP服务器**。

操作步骤

步骤1 在用户视图下执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

步骤2 配置FTP服务器的IP地址。

```
<HUAWEI> system-view
[~HUAWEI] sysname FTP_Server
[*HUAWEI] commit
[~FTP_Server] interface gigabitethernet 1/0/1
[~FTP_Server-GigabitEthernet1/0/1] undo portswitch
[*FTP_Server-GigabitEthernet1/0/1] ip address 10.136.23.5 255.255.255.0
[*FTP_Server-GigabitEthernet1/0/1] quit
[*FTP_Server] commit
```

步骤3 配置设备的FTP功能及FTP用户信息。

```
[~FTP_Server] ftp server enable
[~FTP_Server] ftp server source -i gigabitethernet 1/0/1
[*FTP_Server] aaa
[*FTP_Server-aaa] local-user admin1234 password
Please configure the password (8-128)
Enter Password:
Confirm Password:
[*FTP_Server-aaa] local-user admin1234 level 3
[*FTP_Server-aaa] local-user admin1234 service-type ftp
[*FTP_Server-aaa] local-user admin1234 ftp-directory cfcard:/
[*FTP_Server-aaa] quit
[*FTP_Server] commit
```

步骤4 配置FTP服务器访问权限。

```
[~FTP_Server] acl number 2001
[*FTP_Server-acl4-basic-2001] rule permit source 10.136.23.10 0
[*FTP_Server-acl4-basic-2001] rule deny source 10.136.23.20 0
```



```
[*FTP_Server-acl4-basic-2001] quit
[*FTP_Server] ftp server acl 2001
[*FTP_Server] commit
[~FTP_Server] quit
```

步骤5 保存设备当前配置文件。

```
<FTP_Server> save
```

步骤6 从终端PC1通过FTP连接设备，输入用户名**admin1234**和密码**YsHsjx_202206**，并采用binary模式进行文件传输。

终端以Windows操作系统为例说明。

```
C:\Documents and Settings\Administrator> ftp 10.136.23.5
连接到 10.136.23.5。
220 FTP service ready.
用户 (10.136.23.5:(none)): admin1234
331 Password required for admin1234.
密码:
230 User logged in.
ftp> binary
200 Type set to I.
ftp>
```

步骤7 将系统软件上传至设备以及配置文件备份至终端。

上传系统软件至设备。

```
ftp> put devicesoft.cc
200 Port command okay.
150 Opening BINARY mode data connection for /devicesoft.cc
226 Transfer complete.
ftp: 发送 107973953 字节，用时 151.05Seconds 560.79Kbytes/sec.
```

备份配置文件。

```
ftp> get vrpcfg.zip
200 Port command okay.
150 Opening BINARY mode data connection for /vrpcfg.zip.
226 Transfer complete.
ftp: 收到 1257 字节，用时 0.03Seconds 40.55Kbytes/sec.
```

说明

用户在进行上传和下载操作时，需要明确客户端FTP的工作路径，例如Windows操作系统默认的FTP路径是C:\Windows\System32。待上传的系统软件需要预先保存至此路径下，以及备份的配置文件也将保存在此路径下。

----结束

检查配置结果

在设备中执行**dir**命令，查看系统软件是否上传至设备。

```
<FTP_Server> dir
Directory of cfcard:/

 Idx  Attr     Size(Byte)  Date        Time       FileName
   0  -rw-             14  Mar 13 2019 14:13:38   back_time_a
   1  drw-              -  Mar 11 2019 00:58:54   logfile
   2  -rw-              4  Nov 17 2019 09:33:58   snmpnotilog.txt
   3  -rw-         11,238  Mar 12 2019 21:15:56   private-data.txt
   4  -rw-          1,257  Mar 12 2019 21:15:54   vrpcfg.zip
   5  -rw-             14  Mar 13 2019 14:13:38   back_time_b
   6  -rw-    107,973,953  Mar 13 2019 14:24:24   devicesoft.cc
   7  drw-              -  Oct 31 2019 10:20:28   sysdrv
   8  drw-              -  Feb 21 2019 17:16:36   compatible
```



```
  9  drw-           -  Feb 09 2019 14:20:10   selftest
 10  -rw-      19,174  Feb 20 2019 18:55:32   backup.cfg
 11  -rw-      23,496  Oct 15 2019 20:59:36   20191015.zip
 12  -rw-         588  Nov 04 2019 13:54:04   servercert.der
 13  -rw-         320  Nov 04 2019 13:54:26   serverkey.der
 14  drw-           -  Nov 04 2019 13:58:36   security
...
670,092 KB total (569,904 KB free)
```

在终端FTP用户的工作路径下，可以看到**vrpcfg.zip**文件已保存至此路径。

配置脚本

```
#
sysname FTP_Server
#
ftp server enable
ftp server source -i GigabitEthernet1/0/1
ftp server acl 2001
#
acl number 2001
 rule 5 permit source 10.136.23.10 0
 rule 10 deny source 10.136.23.20 0
#
aaa
 local-user admin1234 password irreversible-cipher $1d$g8wLJ`LjL!$CyE(V{3qg5DdU:PM[6=6O
$UF-.fQ,Q}>^)OBzgoU$
 local-user admin1234 level 3
 local-user admin1234 ftp-directory cfcard:
 local-user admin1234 service-type ftp
#
interface GigabitEthernet1/0/1
 ip address 10.136.23.5 255.255.255.0
#
return
```

1.1.4.5.4 举例：配置设备作为 FTP 客户端

组网需求

如图1-20所示，远端服务器提供FTP Server功能，IP地址为10.1.1.1/24。设备作为FTP客户端，IP地址为10.2.1.1/24，与服务器之间的路由可达。

设备需要进行升级操作，要求：从FTP服务器上下载系统软件至设备，且备份当前设备的配置文件到FTP服务器。

图 1-20 配置通过 FTP 访问其他设备文件组网图

说明

本例中interface1代表GigabitEthernet1/0/1。

配置思路

采用如下的思路配置FTP访问其他设备文件功能：

1. 在FTP服务器端运行FTP软件，并设置FTP用户的相关信息。
2. 通过FTP与FTP服务器建立连接。
3. 在设备上使用FTP命令下载和上传文件。

配置注意事项

当网络所处环境不足够安全时，我们建议选择安全的密码认证方式/加密认证算法/协议。安全的举例请参见：**配置设备作为SFTP客户端**。

操作步骤

步骤1 在用户视图下执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

步骤2 在FTP服务器端运行FTP软件，并设置FTP用户的相关信息。（具体操作见第三方软件帮助文档）

步骤3 通过FTP与FTP服务器建立连接。

```
<HUAWEI> ftp 10.1.1.1
Trying 10.1.1.1 ...
Press CTRL + K to abort
Connected to 10.1.1.1.
220 FTP service ready.
User(10.1.1.1:(none)):admin
331 Password required for admin.
Enter password:
230 User logged in.
[ftp]
```

步骤4 在设备上使用FTP命令下载和上传文件。

```
[ftp] binary
200 Type is Image (Binary)
[ftp] get devicesoft.cc
500 Unidentified command SIZE test123.cfg
200 PORT command okay
150 "D:\FTP\test123.cfg" file ready to send (3544 bytes) in IMAGE / Binary mode
..
226 Transfer finished successfully.
FTP: 107973953 byte(s) received in 151.05 second(s) 560.79Kbyte(s)/sec.
[ftp] put vrpcfg.zip
200 PORT command okay
150 "D:\FTP\vrpcfg.zip" file ready to receive in IMAGE / Binary mode
/     100% [**********]
226 Transfer finished successfully.
FTP: 1257 byte(s) send in 0.03 second(s) 40.55Kbyte(s)/sec.
[ftp] quit
```

----结束

检查配置结果

在设备中执行**dir**命令，查看系统软件是否下载至设备。

```
<HUAWEI> dir
Directory of cfcard:/

 Idx  Attr     Size(Byte)  Date        Time       FileName
   0  -rw-             14  Mar 13 2019 14:13:38   back_time_a
   1  drw-              -  Mar 11 2019 00:58:54   logfile
   2  -rw-              4  Nov 17 2019 09:33:58   snmpnotilog.txt
   3  -rw-         11,238  Mar 12 2019 21:15:56   private-data.txt
   4  -rw-          1,257  Mar 12 2019 21:15:54   vrpcfg.zip
```



```
  5  -rw-           14  Mar 13 2019 14:13:38   back_time_b
  6  -rw-   107,973,953  Mar 13 2019 14:24:24   devicesoft.cc
  7  drw-            -  Oct 31 2019 10:20:28   sysdrv
  8  drw-            -  Feb 21 2019 17:16:36   compatible
  9  drw-            -  Feb 09 2019 14:20:10   selftest
 10  -rw-       19,174  Feb 20 2019 18:55:32   backup.cfg
 11  -rw-       23,496  Oct 15 2019 20:59:36   20191015.zip
 12  -rw-          588  Nov 04 2019 13:54:04   servercert.der
 13  -rw-          320  Nov 04 2019 13:54:26   serverkey.der
 14  drw-            -  Nov 04 2019 13:58:36   security
...
670,092 KB total (569,904 KB free)
```

在FTP服务器的工作路径下，可以看到**vrpcfg.zip**文件已保存至此路径。

配置脚本

无

1.1.4.6 通过 SFTP 进行文件操作

1.1.4.6.1 配置设备作为 SFTP 服务器

前提条件

SFTP使得用户终端可以在SSH协议的基础上与远端设备进行安全连接，增加数据传输的安全性。

在配置通过SFTP进行文件操作之前，需完成以下任务：

- 终端与设备之间有可达路由。
- 终端上已安装SSH客户端软件。

背景信息

说明

- 使用SFTP V1协议存在安全风险，建议使用SFTP V2方式进行文件操作。

通过SFTP进行文件操作的配置流程如**表1-57**所示。

表 1-57 通过 SFTP 进行文件操作的配置流程

序号	配置任务名称	配置任务说明	配置流程说明
1	**配置SFTP服务器功能及参数**	包括服务器本地密钥对生成、SFTP服务器功能的使能及服务器参数的配置：端口号、密钥对更新时间、SSH认证超时时间、SSH验证重试次数等。	序号1、2之间没有严格的配置顺序。用户访问设备后（序号3），可执行序号4的操作，最后断开连接（序号5）。
2	**配置SSH用户**	包括SSH用户的创建、认证方式、服务方式、SFTP服务授权目录等。	

序号	配置任务名称	配置任务说明	配置流程说明
3	用户通过SFTP协议访问设备	从终端通过SSH客户端软件访问设备。	
4	通过SFTP命令进行文件操作	从终端通过SSH客户端软件实现对设备的文件和目录管理。	
5	断开与SFTP服务器的连接	-	

缺省配置

表 1-58 缺省配置

参数	缺省值
SFTP服务器功能	关闭
SSH用户的SFTP服务授权目录	空

操作步骤

- **配置SFTP服务器功能及参数**。

  服务器本地密钥对生成以及服务器参数的配置：端口号、密钥对更新时间、SSH认证超时时间或SSH认证重试次数等配置，请参见《配置指南-安全配置》中的“配置SSH服务器功能及参数”。关于SFTP功能的相关配置见表3。

  表 1-59 配置 SFTP 服务器功能及参数

操作步骤	命令	说明
进入系统视图	**system-view**	-
使能SFTP服务器功能	**sftp** [**ipv4** | **ipv6**] **server enable**	缺省情况下，SFTP服务为关闭状态。

操作步骤	命令	说明
配置SFTP服务器的默认授权目录	**sftp server default-directory** *sftpdir*	缺省情况下，SFTP服务器的默认授权目录为空。 用户可以使用以下三种方式配置SFTP服务器的默认授权目录： **ssh user** *username* **sftp-directory** *directoryname* 配置指定用户的访问目录，优先级最高。 其次是AAA视图下**local-user** *user-name* **ftp-directory** *directory*命令指定的FTP访问目录。 **sftp server default-directory** *sftpdir*的优先级最低，且对所有SSH用户生效。
（可选）配置SFTP服务器最大客户端数	**sftp max-sessions** *max-session-count*	缺省情况下，SSH服务器最大客户端数是5。 如果设置的最大连接数小于当前登录设备的用户数，那么系统将拒绝新的连接请求，当前连接不会断开。
（可选）配置SFTP服务器和客户端之间断开连接的超时时间	**sftp idle-timeout** *minutes* [*seconds*]	缺省情况下，服务器和客户端之间断开连接的超时时间是10分钟。 执行命令**sftp idle-timeout 0 0**将关闭服务器和客户端之间超时断开连接功能。
提交配置	**commit**	-

- **配置SSH用户**。

  详细配置请参见《配置指南-安全配置》中的“配置SSH用户”。

  说明

  配置AAA用户时，必须将用户级别配置为3级及3级以上，否则连接不成功。

- **用户通过SFTP协议访问设备。**

  从终端通过SFTP访问设备，需要在终端上安装SSH客户端软件。此处以使用第三方软件OpenSSH和Windows命令行提示符为例进行配置。

  - OpenSSH软件的安装请参考该软件的安装说明。
  - 使用OpenSSH软件从终端访问设备时，需要使用OpenSSH的命令，命令的使用可以参见该软件的帮助文档。

– 只有安装了OpenSSH软件后，Windows命令行提示符才能识别OpenSSH相关命令。

进入Windows的命令行提示符，执行OpenSSH命令，通过SFTP方式访问设备。

当出现SFTP客户端视图的命令行提示符，如sftp>，此时用户进入了SFTP服务器的工作目录。（以下显示信息仅为示意）

```
C:/Documents and Settings/Administrator> sftp client001@10.136.23.4
Connecting to 10.136.23.4...
The authenticity of host "10.136.23.4 (10.136.23.4)" can't be established.
DSA key fingerprint is 0d:48:82:fd:2f:52:1c:f0:c4:22:70:80:8f:7b:fd:78.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added "10.136.23.4" (DSA) to the list of known hosts.
client001@10.136.23.4's password:
sftp>
```

- ***通过SFTP命令进行文件操作。***

  *当SFTP客户端登录到SSH服务器之后，用户可以在SFTP客户端进行如**表1-60**所示的操作。以下各操作没有严格顺序，可根据需求选择一种或多种操作项目。*

表 1-60 *通过 SFTP 文件操作命令进行文件操作*

操作项目	*命令*	*说明*
改变用户的当前工作目录	***cd*** *[path]*	-
改变用户的工作目录为当前工作目录的上一级目录	***cdup***	-
显示用户的当前工作目录	**pwd**	-
显示指定目录下的文件列表	**dir** [*remote-directory* [*local-filename*]] 或 **ls** [*remote-directory* [*local-filename*]]	**dir**与**ls**执行的效果是一样的。
删除服务器上目录	**rmdir** *directory-name*	一次最多可以删除十个目录。 使用该命令删除目录时，目录中不能有文件，否则会删除失败。
在服务器上创建新目录	**mkdir** *remote-directory*	-
改变服务器上指定的文件的名字	**rename** *old-name new-name*	-
下载远程服务器上的文件	**get** *remote-filename* [*local-filename*]	-

操作项目	*命令*	*说明*
上传本地文件到远程服务器	**put** *local-filename* [*remote-filename*]	-
删除服务器上文件	**remove** *path* 或 **delete** *path*	一次最多可以删除十个文件。 **remove**与**delete**执行的效果是一样的。
SFTP客户端命令帮助	**help** [*command-name*]	-

- **断开与SFTP服务器的连接**。

  表 1-61 断开与 SFTP 服务器的连接

操作步骤	命令	说明
断开与SFTP服务器的连接	**quit**	也可执行**bye**或**exit**命令断开连接。

----结束

检查配置结果

- 使用**display ssh user-information** [*username*]命令，在SSH服务器端查看SSH用户信息。
- 使用**display ssh server status**命令，查看SSH服务器的全局配置信息。
- 使用**display ssh server session**命令，在SSH服务器端查看SSH客户端连接会话信息。

1.1.4.6.2 配置设备作为 SFTP 客户端

前提条件

配置设备作为SFTP客户端，远程服务器通过对客户端的认证及双向的数据加密，为文件传输及对服务器上文件和目录的管理提供了安全的服务。

在配置通过SFTP访问其他设备的文件之前，需完成以下任务：

- 当前设备和SSH服务器路由可达。
- 已获取SSH服务器的IP地址以及SSH用户信息，且SSH用户的用户级别为最高级别。
- 如果服务器不是使用标准的端口号，则还需获取服务器端设置的端口号。

背景信息

通过SFTP访问其他设备文件的配置流程如表1-62所示。

表 1-62 配置设备作为 SFTP 客户端访问其他设备文件的配置流程

序号	配置任务名称	配置任务说明	配置流程说明
1	（可选）配置SFTP客户端源接口或源地址	客户端源地址可以配置为源接口或源地址，达到安全校验的目的。	序号1、2、3无序操作。建立SFTP连接并进行文件操作（序号4），最后断开连接（序号5）。
2	配置设备首次连接SSH服务器的方式	有两种配置方式：使能SSH客户端首次登录功能方式和SSH客户端为SSH服务器分配公钥方式，用户可选择其一进行配置。	
3	配置SFTP客户端参数	配置SFTP客户端参数包括配置SFTP客户端发送keepalive报文的时间间隔和最大数目等。	
4	连接其他设备进行文件操作（一键式） 连接其他设备进行文件操作（交互式）	二选一。 • 一键式：在连接的同时可进行文件的上传和下载。 • 交互式：先连接SSH服务器，然后对SSH服务器上的目录和文件进行操作，以及查看SFTP客户端命令帮助。	
5	断开与SFTP服务器的连接	-	

操作步骤

- **（可选）配置SFTP客户端源接口或源地址。**

  配置源地址需要选择设备上状态稳定的接口，如LoopBack接口。该配置简化了ACL规则的配置，只要将ACL规则的源地址或目的地址指定为该地址，就可以屏蔽接口IP地址的差异以及接口状态的影响，实现对设备进出报文的过滤。

  表 1-63 配置 SFTP 客户端源接口或源地址

操作步骤	命令	说明
进入系统视图	**system-view**	-

操作步骤	命令	说明
配置SFTP客户端的源接口或源地址	**sftp client-source** { **-a** *source-ip-address* [**public-net** | **-vpn-instance** *vpn-instance-name*] | **-i** { *interface-type interface-number* | *interface-name* } } 或 **sftp ipv6 client-source -a** *source-ipv6-address* [**-vpn-instance** *ipv6-vpn-instance-name*]	缺省情况下，源地址为0.0.0.0。
提交配置	**commit**	-

- **配置设备首次连接SSH服务器的方式**。

  请参见《配置指南-安全配置》中的“配置设备首次连接SSH服务器的方式”。
- **配置SFTP客户端参数。**

  请参见《配置指南-安全配置》中的“配置SSH客户端参数”。
- **连接其他设备进行文件操作（一键式）。**

  用户可以在系统视图下执行如下命令在连接其他设备的同时，下载服务器上的文件或者上传本地文件到远程服务器中。

  表 1-64 一键式命令进行文件操作

操作项目	命令	说明
通过IPv4地址连接SFTP服务器	**sftp client-transfile** { **get** | **put** } [**-a** *source-address* | **-i** *interface-type interface-number*] **host-ip** *host-ipv4* [*port*] [**public-net** | **-vpn-instance** *vpn-instance-name* | **prefer_kex** *prefer_kex* | **identity-key** { **rsa** | **dsa** | **ecc** } | **prefer_ctos_cipher** *prefer_ctos_cipher* | **prefer_stoc_cipher** *prefer_stoc_cipher* | **prefer_ctos_hmac** *prefer_ctos_hmac* | **prefer_stoc_hmac** *prefer_stoc_hmac* | **-ki** *interval* | **-kc** *count*] * **username** *user-name* **password** *password* **sourcefile** *source-file* [**destination** *destination*]	通过IPv4方式连接SFTP服务器，并从服务器上下载文件至SFTP客户端或者从SFTP客户端上传文件至服务器。

操作项目	命令	说明
通过IPv6地址连接SFTP服务器	**sftp client-transfile** { **get** | **put** } **ipv6** [**-a** *source-ipv6-address*] **host-ip** *host-ipv6* [**-oi** *interface-type interface-number*] [*port*] [**public-net** | **-vpn-instance** *vpn-instance-name* | **prefer_kex** *prefer_kex* | **identity-key** { **rsa** | **dsa** | **ecc** } | **prefer_ctos_cipher** *prefer_ctos_cipher* | **prefer_stoc_cipher** *prefer_stoc_cipher* | **prefer_ctos_hmac** *prefer_ctos_hmac* | **prefer_stoc_hmac** *prefer_stoc_hmac* | **-ki** *interval* | **-kc** *count*] * **username** *user-name* **password** *password* **sourcefile** *source-file* [**destination** *destination*]	通过IPv6方式连接SFTP服务器，并从服务器上下载文件至SFTP客户端或者从SFTP客户端上传文件至服务器。

以下内容仅为示例，具体以设备为准。

```
<HUAWEI> system-view
[~HUAWEI] sftp client-transfile get host-ip 10.10.1.1 username client password YsHsjx_202206 sourcefile sourcefile.txt
Trying 10.10.1.1 ...
Press CTRL+K to abort
Connected to 10.10.1.1 ...
Remote file: /sourcefile.txt --->  Local file: 1#cfcard:/sourcefile.txt
Downloading the file. Please wait..
Downloading file successfully ended.
File download is completed in 375 seconds.
```

- **连接其他设备进行文件操作（交互式）。**
  a. **使用SFTP命令连接其他设备。**

表 1-65 使用 SFTP 命令连接其他设备

操作步骤	命令	说明
系统视图下通过IPv4地址连接SFTP服务器	**sftp** [**-a** *source-ip-address* | **-i** *interface-type interface-number*] [**-force-receive-pubkey**] *host-ip-address* [*port-number*] [[**prefer_kex** *prefer_kex*] | [**prefer_ctos_cipher** *prefer_ctos_cipher*] | [**prefer_stoc_cipher** *prefer_stoc_cipher*] | [**prefer_ctos_hmac** *prefer_ctos_hmac*] | [**prefer_stoc_hmac** *prefer_stoc_hmac*] | [**prefer_ctos_compress zlib**] | [**prefer_stoc_compress zlib**] | [**public-net** | **-vpn-instance** *vpn-instance-name*] | [**-ki** *interval*] | [**-kc** *count*] | [**identity-key** *identity-key-type*] | [**user-identity-key** *user-key*]] *	根据地址类型选其一。 大多数情况下，该命令可以只指定IP地址，而不需要指定其他可选项。 如果使用参数**-i** *interface-type interface-number*指定源接口，则不支持**public-net**和**-vpn-instance** *vpn-instance-name*参数。
系统视图下通过IPv6地址连接SFTP服务器	**sftp ipv6** [**-force-receive-pubkey**] [**-a** *source-ipv6-address*] *host-ipv6-address* [[[**-vpn-instance** *vpn-instance-name*] | **public-net**] | [**-oi** { *interface-name* | *interface-type interface-number* }] [*port-number*] | [**prefer_kex** { *prefer_kex* }] | [**prefer_ctos_cipher** *prefer_ctos_cipher*] | [**prefer_stoc_cipher** *prefer_stoc_cipher*] | [**prefer_ctos_hmac** *prefer_ctos_hmac*] | [**prefer_stoc_hmac** *prefer_stoc_hmac*] | [**prefer_ctos_compress zlib**] | [**prefer_stoc_compress zlib**] | [**-ki** *interval*] | [**-kc** *count*] | [**identity-key** *identity-key-type*] | [**user-identity-key** *user-key*]] *	

例如：

```
<HUAWEI> system-view
[~HUAWEI] sftp 10.137.217.201
```

连接成功后，屏幕会显示**sftp-client>**，此时已经进入了SFTP客户端视图。

b. **通过SFTP命令进行文件操作。**

当SFTP客户端登录到SSH服务器之后，用户可以在SFTP客户端进行如**表1-66**所示的操作。

以下各操作没有严格顺序，可根据需求选择一种或多种操作项目。

表 1-66 通过 SFTP 文件操作命令进行文件操作

操作项目	命令	说明
改变用户的当前工作目录	**cd** [*path*]	-
改变用户的工作目录为当前工作目录的上一级目录	**cdup**	-
显示用户的当前工作目录	**pwd**	-
显示指定目录下的文件列表	**dir** [*remote-directory* [*local-filename*]] 或 **ls** [*remote-directory* [*local-filename*]]	**dir**与**ls**执行的效果是一样的。
删除服务器上目录	**rmdir** *directory-name*	一次最多可以删除十个目录。 使用该命令删除目录时，目录中不能有文件，否则会删除失败。
在服务器上创建新目录	**mkdir** *remote-directory*	-
改变服务器上指定的文件的名字	**rename** *old-name new-name*	-
下载远程服务器上的文件	**get** *remote-filename* [*local-filename*]	-
上传本地文件到远程服务器	**put** *local-filename* [*remote-filename*]	-
删除服务器上文件	**remove** *path* 或 **delete** *path*	一次最多可以删除十个文件。 **remove**与**delete**执行的效果是一样的。
SFTP客户端命令帮助	**help** [*command-name*]	-

- **断开与SFTP服务器的连接**。

  表 1-67 断开与 SFTP 服务器的连接

操作步骤	命令	说明
断开与SFTP服务器的连接	**quit**	也可执行**bye**或**exit**命令断开连接。

----结束

检查配置结果

- 使用**display sftp-client**命令，查看SFTP客户端的配置信息
- 使用**display ssh server-info**命令，查看客户端所有的SSH服务器与公钥之间的对应关系。

1.1.4.6.3 举例：配置设备作为 SFTP 服务器

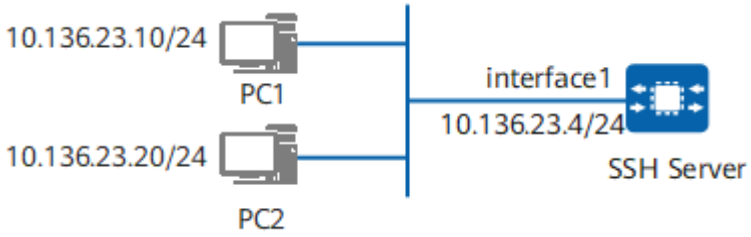
组网需求

如图1-21所示，终端PC1与设备连接，10.136.23.4是设备的IP地址。用户希望在终端与设备之间进行安全的文件传输操作。将设备配置为SSH服务器，提供SFTP服务，服务器通过对客户端的认证和双向的数据加密，实现用户对安全文件传输操作的要求。配置ACL策略，保证只有PC1才能访问SSH服务器。

图 1-21 配置通过 SFTP 进行文件操作组网图

说明

本例中interface1代表GigabitEthernet1/0/1。



配置思路

采用如下的思路配置用户通过SFTP进行文件操作：

1. 在SSH服务器端生成本地密钥对及使能SFTP服务器功能，实现在服务器端和客户端进行安全地数据交互。
2. 配置SSH用户，包括认证方式、服务类型、授权目录以及用户名和密码等。
3. 配置SSH服务器的访问权限，实现对SSH用户的限制。
4. 从终端通过第三方软件OpenSSH实现访问SSH服务器。

操作步骤

步骤1 配置SSH服务器的IP地址。

```
<HUAWEI> system-view
[~HUAWEI] sysname SSH Server
[*HUAWEI] commit
[~SSH Server] interface gigabitethernet 1/0/1
[~SSH Server-GigabitEthernet1/0/1] undo portswitch
[*SSH Server-GigabitEthernet1/0/1] ip address 10.136.23.4 255.255.255.0
[*SSH Server-GigabitEthernet1/0/1] quit
[*SSH Server] commit
```

步骤2 在服务器端生成本地密钥对，并使能SFTP服务器功能。

```
[~SSH Server] rsa local-key-pair create
The key name will be:HUAWEI_Host
The range of public key size is (2048, 4096).
NOTE: Key pair generation will take a short while.
Please input the modulus [default = 3072]:3072
[*SSH Server] sftp server enable
[*SSH Server] ssh server-source all-interface
[*SSH Server] commit
```

步骤3 配置SSH服务器的公钥算法、加密算法、密钥交换算法列表、HMAC认证算法和最小密钥长度。

```
[~SSH Server] ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
[~SSH Server] ssh server hmac sha2_256 sha2_512
[~SSH Server] ssh server key-exchange dh_group_exchange_sha256 dh_group16_sha512
[~SSH Server] ssh server publickey rsa_sha2_256 rsa_sha2_512
[~SSH Server] ssh server dh-exchange min-len 3072
[*SSH Server] commit
```

步骤4 配置SSH用户，包括认证方式、服务类型、授权目录以及用户名和密码等。

```
[~SSH Server] ssh user client001 authentication-type password
Info: Succeeded in adding a new SSH user.
[*SSH Server] ssh user client001 service-type sftp
[*SSH Server] ssh user client001 sftp-directory cfcard:/
[*SSH Server] aaa
[*SSH Server-aaa] local-user client001 password
Please configure the password (8-128)
Enter Password:
Confirm Password:
[*SSH Server-aaa] local-user client001 level 3
[*SSH Server-aaa] local-user client001 service-type terminal ssh
[*SSH Server-aaa] quit
[*SSH Server] commit
```

步骤5 配置SSH服务器的访问权限。

```
[~SSH Server] acl 2001
[*SSH Server-acl4-basic-2001] rule permit source 10.136.23.10 0
[*SSH Server-acl4-basic-2001] rule deny source 10.136.23.20 0
[*SSH Server-acl4-basic-2001] quit
[*SSH Server] ssh server acl 2001
[*SSH Server] commit
```

----结束

检查配置结果

从终端通过OpenSSH软件实现访问SSH服务器。

只有在用户终端安装了OpenSSH软件后，Windows命令行提示符才能识别OpenSSH相关命令。

```
C:/Documents and Settings/Administrator> sftp client001@10.136.23.4
Connecting to 10.136.23.4...
```



```
The authenticity of host "10.136.23.4 (10.136.23.4)" can't be established.
DSA key fingerprint is 0d:48:82:fd:2f:52:1c:f0:c4:22:70:80:8f:7b:fd:78.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added "10.136.23.4" (DSA) to the list of known hosts.
client001@10.136.23.4's password:
sftp>
```

通过第三方软件连接设备后，进入SFTP视图，此时可以执行一系列文件操作。

配置脚本

```
#
sysname SSH Server
#
acl number 2001
 rule 5 permit source 10.136.23.10 0
 rule 10 deny source 10.136.23.20 0
#
aaa
 local-user client001 password irreversible-cipher $1d$v!=.5/:(q-$xL=\K
+if"'S}>k7vGP5$_ox0B@ys7.'DBHL~3*aN$
 local-user client001 service-type terminal ssh

 local-user client001 level 3
#
interface GigabitEthernet1/0/1
 ip address 10.136.23.4 255.255.255.0
#
sftp server enable
ssh server-source all-interface
ssh server acl 2001
ssh user client001
ssh user client001 authentication-type password
ssh user client001 service-type sftp
ssh user client001 sftp-directory cfcard:
#
ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
ssh server hmac sha2_256 sha2_512
ssh server key-exchange dh_group_exchange_sha256 dh_group16_sha512
ssh server publickey rsa_sha2_256 rsa_sha2_512
ssh server dh-exchange min-len 3072
#
return
```

1.1.4.6.4 举例：配置设备作为 SFTP 客户端

组网需求

SSH提供了在一个传统不安全的网络环境中，服务器通过对客户端的认证及双向的数据加密，为网络终端访问提供了安全的服务。通过SFTP方式，客户端可以安全地连接到SSH服务器，进行文件的安全传输。

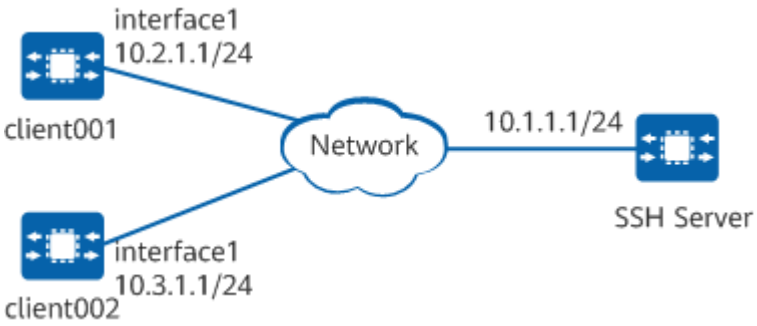
如图1-22所示，SSH服务器与客户端client001、client002路由可达，此例中用华为设备作为SSH服务器。

要求：两个客户端分别使用password方式和RSA方式与SSH服务器连接，实现安全访问服务器上的文件。

图 1-22 通过 SFTP 访问其他设备文件组网图

说明

本例中interface1代表GigabitEthernet1/0/1。



配置思路

采用如下思路配置通过SFTP访问其他设备文件功能：

1. 在服务器端生成本地密钥对及使能SFTP服务器功能，实现在服务器端和客户端进行安全地数据交互。
2. 在SSH服务器上配置用户client001和client002，分别使用password和RSA的认证方式登录SSH服务器。
3. 在客户端client002生成本地密钥对，并将客户端生成的RSA公钥配置到SSH服务器上，实现客户端登录服务器端时，对客户端进行验证。
4. 用户client001和client002分别以SFTP方式登录SSH服务器，实现访问服务器上的文件。

操作步骤

步骤1 在服务器端生成本地密钥对及使能SFTP服务器功能。

```
<HUAWEI> system-view
[~HUAWEI] sysname SSH Server
[*HUAWEI] commit
[~SSH Server] rsa local-key-pair create
The key name will be:HUAWEI_Host
The range of public key size is (2048, 4096).
NOTE: Key pair generation will take a short while.
Please input the modulus [default = 3072]:3072
[*SSH Server] sftp server enable
[*SSH Server] ssh server-source all-interface
[*SSH Server] commit
```

步骤2 配置SSH服务器的公钥算法、加密算法、密钥交换算法列表、HMAC认证算法和最小密钥长度。

```
[~SSH Server] ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
[~SSH Server] ssh server hmac sha2_256 sha2_512
[~SSH Server] ssh server key-exchange dh_group_exchange_sha256 dh_group16_sha512
[~SSH Server] ssh server publickey rsa_sha2_256 rsa_sha2_512
[~SSH Server] ssh server dh-exchange min-len 3072
[*SSH Server] commit
```

步骤3 在服务器端创建SSH用户。

新建用户名为client001的SSH用户，且认证方式为password。

```
[~SSH Server] ssh user client001
[*SSH Server] ssh user client001 authentication-type password
[*SSH Server] ssh user client001 service-type sftp
[*SSH Server] ssh user client001 sftp-directory cfcard:/
[*SSH Server] aaa
[*SSH Server-aaa] local-user client001 password
Please configure the password (8-128)
```



```
Enter Password:
Confirm Password:
[*SSH Server-aaa] local-user client001 service-type terminal ssh
[*SSH Server-aaa] local-user client001 level 3
[*SSH Server-aaa] quit
[*SSH Server] commit
```

新建用户名为client002的SSH用户，且认证方式为RSA。

```
[~SSH Server] ssh user client002
[*SSH Server] ssh user client002 authentication-type rsa
[*SSH Server] ssh authorization-type default root
[*SSH Server] ssh user client002 service-type sftp
[*SSH Server] ssh user client002 sftp-directory cfcard:/
[*SSH Server] commit
```

步骤4 在客户端client001，配置加密算法、HMAC认证算法、密钥交换算法列表、公钥算法。

```
<HUAWEI> system-view
[~HUAWEI] sysname client001
[*HUAWEI] commit
[*client001] ssh client cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
[*client001] ssh client hmac sha2_256 sha2_512
[*client001] ssh client key-exchange dh_group_exchange_sha256 dh_group16_sha512
[*client001] ssh client publickey rsa_sha2_256 rsa_sha2_512
[*client001] commit
```

步骤5 在客户端client002生成本地密钥对，并将客户端生成的RSA公钥配置到SSH服务器上。

客户端生成客户端的本地密钥对。

```
<HUAWEI> system-view
[~HUAWEI] sysname client002
[*HUAWEI] commit
[~client002] rsa local-key-pair create
The key name will be:HUAWEI_Host
The range of public key size is (2048, 4096).
NOTE: Key pair generation will take a short while.
Please input the modulus [default = 3072]:3072
[*client002] commit
```

配置客户端client002的加密算法、HMAC认证算法、密钥交换算法列表、公钥算法。

```
[*client002] ssh client cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
[*client0022] ssh client hmac sha2_256 sha2_512
[*client002] ssh client key-exchange dh_group_exchange_sha256 dh_group16_sha512
[*client002] ssh client publickey rsa_sha2_256 rsa_sha2_512
[*client002] commit
```

查看客户端上生成RSA公钥。

```
[~client002] display rsa local-key-pair public
=========================================================
Time of key pair created : 2019-11-05 12:10:40
Key name                 : Host_RSA
Key modulus              : 3072
Key type                 : RSA encryption key
=========================================================
Key code:
3082010A
  02820101
    00BBB7A0 4924AF13 04F2662D 2ED43B9D 589967EB
    D8A4F785 5AD1F662 13845081 0C65F6B3 88A9C415
    D81C34BD 41A4B580 70DC7460 E4A5407B 9B95630F
    E211F4B3 1115772D FB95D3DC 915A1858 D0DE49F7
    F39DD7A7 7795F2B9 C9562E8B 598CB50F 6D39240D
    B5C6F1D3 33A218D0 98C30104 F8F3A8CA 7172C95B
    03AEC0A0 8A7E99F6 6C1939AA 52CC2E31 B6703278
```



```
  AEE1BCD8 DC21FCA2 041C9A4C 1856A935 6894998D
  FBFA88FF 1708C3A6 7E092368 ACE983D7 C8DDCDF5
  26F5D4E5 16A15C5C D6D0018E 4EAFE055 B93FCB87
  2BB46EFB 02C04C3B F167A417 380CD0B0 0BC59493
  646CBE96 BCAF3DB7 AD0AFA0A 5D14155E D7F97DC1
  32693DE5 4B103442 8E0F4DAD 2598BE5E 19
0203
  010001

Host public key for PEM format code:
---- BEGIN SSH2 PUBLIC KEY ----
AAAAB3NzaC1yc2EAAAADAQABAAABAQC7t6BJJK8TBPJmLS7UO51YmWfr2KT3hVrR
9mIThFCBDGX2s4ipxBXYHDS9QaS1gHDcdGDkpUB7m5VjD+IR9LMRFXct+5XT3JFa
GFjQ3kn3853Xp3eV8rnJVi6LWYy1D205JA21xvHTM6IY0JjDAQT486jKcXLJWwOu
wKCKfpn2bBk5qlLMLjG2cDJ4ruG82Nwh/KIEHJpMGFapNWiUmY37+oj/FwjDpn4J
I2is6YPXyN3N9Sb11OUWoVxc1tABjk6v4FW5P8uHK7Ru+wLATDvxZ6QXOAzQsAvF
lJNkbL6WvK89t60K+gpdFBVe1/l9wTJpPeVLEDRCjg9NrSWYvl4Z
---- END SSH2 PUBLIC KEY ----

Public key code for pasting into OpenSSH authorized_keys file:
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC7t6BJJK8TBPJmLS7UO51YmWfr2KT3hVrR9mIThFCB
DGX2s4ipxBXYHDS9QaS1gHDcdGDkpUB7m5VjD+IR9LMRFXct+5XT3JFaGFjQ3kn3853Xp3eV8rnJVi6L
WYy1D205JA21xvHTM6IY0JjDAQT486jKcXLJWwOuwKCKfpn2bBk5qlLMLjG2cDJ4ruG82Nwh/KIEHJpM
GFapNWiUmY37+oj/FwjDpn4JI2is6YPXyN3N9Sb11OUWoVxc1tABjk6v4FW5P8uHK7Ru+wLATDvxZ6QX
OAzQsAvFlJNkbL6WvK89t60K+gpdFBVe1/l9wTJpPeVLEDRCjg9NrSWYvl4Z== rsa-key
```

将客户端上产生的RSA公钥配置到服务器端(上面**display**命令显示信息中黑体部分即为客户端产生的RSA公钥，将其拷贝粘贴至服务器端)。

```
[~SSH Server] rsa peer-public-key rsakey001 encoding-type der
[*SSH Server-rsa-public-key] public-key-code begin
[*SSH Server-rsa-public-key-rsa-key-code] 3082010A
[*SSH Server-rsa-public-key-rsa-key-code] 02820101
[*SSH Server-rsa-public-key-rsa-key-code] 00BBB7A0 4924AF13 04F2662D 2ED43B9D 589967EB
[*SSH Server-rsa-public-key-rsa-key-code] D8A4F785 5AD1F662 13845081 0C65F6B3 88A9C415
[*SSH Server-rsa-public-key-rsa-key-code] D81C34BD 41A4B580 70DC7460 E4A5407B 9B95630F
[*SSH Server-rsa-public-key-rsa-key-code] E211F4B3 1115772D FB95D3DC 915A1858 D0DE49F7
[*SSH Server-rsa-public-key-rsa-key-code] F39DD7A7 7795F2B9 C9562E8B 598CB50F 6D39240D
[*SSH Server-rsa-public-key-rsa-key-code] B5C6F1D3 33A218D0 98C30104 F8F3A8CA 7172C95B
[*SSH Server-rsa-public-key-rsa-key-code] 03AEC0A0 8A7E99F6 6C1939AA 52CC2E31 B6703278
[*SSH Server-rsa-public-key-rsa-key-code] AEE1BCD8 DC21FCA2 041C9A4C 1856A935 6894998D
[*SSH Server-rsa-public-key-rsa-key-code] FBFA88FF 1708C3A6 7E092368 ACE983D7 C8DDCDF5
[*SSH Server-rsa-public-key-rsa-key-code] 26F5D4E5 16A15C5C D6D0018E 4EAFE055 B93FCB87
[*SSH Server-rsa-public-key-rsa-key-code] 2BB46EFB 02C04C3B F167A417 380CD0B0 0BC59493
[*SSH Server-rsa-public-key-rsa-key-code] 646CBE96 BCAF3DB7 AD0AFA0A 5D14155E D7F97DC1
[*SSH Server-rsa-public-key-rsa-key-code] 32693DE5 4B103442 8E0F4DAD 2598BE5E 19
[*SSH Server-rsa-public-key-rsa-key-code] 0203
[*SSH Server-rsa-public-key-v-key-code] 010001
[*SSH Server-rsa-public-key-rsa-key-code] public-key-code end
[*SSH Server-rsa-public-key] peer-public-key end
```

为SSH用户client002绑定SSH客户端的RSA公钥。

```
[*SSH Server] ssh user client002 assign rsa-key rsakey001
[*SSH Server] commit
```

步骤6 SFTP客户端连接SSH服务器。

第一次登录，使能SSH客户端首次登录功能。

使能客户端client001首次登录功能。

```
<HUAWEI> system-view
[~HUAWEI] sysname client001
[*HUAWEI] commit
[~client001] ssh client first-time enable
[*client001] commit
```

使能客户端client002首次登录功能。

```
[~client002] ssh client first-time enable
[*client002] commit
```

SFTP客户端client001用password认证方式连接SSH服务器。

```
[~client001] sftp 10.1.1.1
Trying 10.1.1.1 ...
Press CTRL+K to abort
Connected to 10.1.1.1 ...
The server's public key does not match the one cached before.
The server is not authenticated. Continue to access it? [Y/N]:y
The keyname:10.1.1.1 already exists. Update it? [Y/N]:n

Please input the username: client001
Enter password:
sftp-client>
```

SFTP客户端client002用RSA认证方式连接SSH服务器。

```
[~client002] sftp 10.1.1.1
Trying 10.1.1.1 ...
Press CTRL+K to abort
Connected to 10.1.1.1 ...
The server's public key does not match the one cached before.
The server is not authenticated. Continue to access it? [Y/N]:y
The keyname:10.1.1.1 already exists. Update it? [Y/N]:n

Please input the username: client002
sftp-client>
```

----结束

检查配置结果

配置完成后，在SSH服务器端执行**display ssh server status**命令可以查看到SFTP服务已经使能。执行**display ssh user-information**命令可以查看服务器端SSH用户信息。

查看SSH状态信息。

```
[~SSH Server] display ssh server status
SSH Version                            : 2.0
SSH authentication timeout (Seconds)       : 60
SSH authentication retries (Times)         : 3
SSH server key generating interval (Hours) : 0
SSH version 1.x compatibility              : Disable
SSH server keepalive                     : Enable
SFTP IPv4 server                        : Enable
SFTP IPv6 server                        : Enable
STELNET IPv4 server                       : Enable
STELNET IPv6 server                       : Enable
SNETCONF IPv4 server                      : Disable
SNETCONF IPv6 server                      : Disable
SNETCONF IPv4 server port(830)              : Disable
SNETCONF IPv6 server port(830)              : Disable
SCP IPv4 server                         : Enable
SCP IPv6 server                         : Enable
SSH IPv4 server port                      : 22
SSH IPv6 server port                      : 22
ACL name                               : --
ACL number                              : --
ACL6 name                              : --
ACL6 number                             : --
SSH server ip-block                       : Enable
```

查看SSH用户信息。

```
[~SSH Server] display ssh user-information
------------------------------------------------------------------------------
```



```
User Name             : client001
Authentication type   : password
User public key name  : --
User public key type  : --
Sftp directory        : cfcard:
Service type          : sftp

User Name             : client002
Authentication type   : rsa
User public key name  : --
User public key type  : --
Sftp directory        : cfcard:
Service type          : sftp
------------------------------------------------------------------------------
Total 2, 2 printed
```

配置脚本

- SSH服务器上的配置脚本

```
#
sysname SSH Server
#
rsa peer-public-key rsakey001 encoding-type der
 public-key-code begin
 3082010A
  02820101
    00BBB7A0 4924AF13 04F2662D 2ED43B9D 589967EB D8A4F785 5AD1F662 13845081
    0C65F6B3 88A9C415 D81C34BD 41A4B580 70DC7460 E4A5407B 9B95630F E211F4B3
    1115772D FB95D3DC 915A1858 D0DE49F7 F39DD7A7 7795F2B9 C9562E8B 598CB50F
    6D39240D B5C6F1D3 33A218D0 98C30104 F8F3A8CA 7172C95B 03AEC0A0 8A7E99F6
    6C1939AA 52CC2E31 B6703278 AEE1BCD8 DC21FCA2 041C9A4C 1856A935 6894998D
    FBFA88FF 1708C3A6 7E092368 ACE983D7 C8DDCDF5 26F5D4E5 16A15C5C D6D0018E
    4EAFE055 B93FCB87 2BB46EFB 02C04C3B F167A417 380CD0B0 0BC59493 646CBE96
    BCAF3DB7 AD0AFA0A 5D14155E D7F97DC1 32693DE5 4B103442 8E0F4DAD 2598BE5E
    19
  0203
    010001
 public-key-code end
 peer-public-key end
#
aaa
 local-user client001 password irreversible-cipher $1d$g8wLJ`LjL!$CyE(V{3qg5DdU:PM[6=6O
$UF-.fQ,Q}>^)OBzgoU$
 local-user client001 service-type terminal ssh
 local-user client001 level 3
#
sftp server enable
ssh user client001
ssh user client001 authentication-type password
ssh user client001 service-type sftp
ssh user client001 sftp-directory cfcard:/
ssh user client002
ssh user client002 authentication-type rsa
ssh user client002 assign rsa-key rsakey001
ssh user client002 service-type sftp
ssh user client002 sftp-directory cfcard:/
#
ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
ssh server hmac sha2_256 sha2_512
ssh server key-exchange dh_group_exchange_sha256 dh_group16_sha512
ssh server publickey rsa_sha2_256 rsa_sha2_512
ssh server dh-exchange min-len 3072
#
return
```

- SSH客户端client001的配置脚本

```
#
sysname client001
```

```
#
ssh client first-time enable
#
ssh client cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
ssh client hmac sha2_256 sha2_512
ssh client key-exchange dh_group_exchange_sha256 dh_group16_sha512
ssh client publickey rsa_sha2_256 rsa_sha2_512
#
return
```

- SSH客户端client002的配置脚本

```
#
sysname client002
#
ssh client first-time enable
#
ssh client cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
ssh client hmac sha2_256 sha2_512
ssh client key-exchange dh_group_exchange_sha256 dh_group16_sha512
ssh client publickey rsa_sha2_256 rsa_sha2_512
#
return
```

1.1.4.7 通过 SCP 进行文件操作

1.1.4.7.1 配置设备作为 SCP 服务器

前提条件

SCP使得用户终端可以在SSH协议的基础上，与远端设备建立连接的同时完成文件上传和下载的操作。

配置通过SCP进行文件操作之前，需完成以下任务：

- 终端与设备之间路由可达。
- 终端上已安装支持SCP的SSH客户端软件。

背景信息

通过SCP进行文件操作的配置流程如**表1-68**所示。

表 1-68 通过 SCP 进行文件操作的配置流程

序号	配置任务名称	配置任务说明	配置流程说明
1	**配置SCP服务器功能及参数**	包括服务器本地密钥对生成、SCP服务器功能的使能及服务器参数的配置：端口号、密钥对更新时间、SSH认证超时时间、SSH验证重试次数等。	序号1、2之间没有严格的配置顺序。
2	**配置SSH用户**	包括SSH用户的创建、认证方式、服务方式等。	
3	**用户通过SCP进行文件操作**	从终端通过SCP客户端软件实现上传或下载文件的操作。	-

缺省配置

表 1-69 缺省配置

参数	缺省值
SCP服务器功能	关闭

操作步骤

- **配置SCP服务器功能及参数。**

  服务器本地密钥对生成以及服务器参数的配置：端口号、密钥对更新时间、SSH认证超时时间或SSH认证重试次数等配置，请参见《配置指南-安全配置》中的“配置SSH服务器功能及参数”。关于SCP功能的相关配置见**表3**。

  表 1-70 配置 SCP 服务器功能及参数

操作步骤	命令	说明
进入系统视图	**system-view**	-
使能SCP服务器功能	**scp** [**ipv4** | **ipv6**] **server enable**	缺省情况下，SCP服务为关闭状态。
（可选）配置SCP服务器允许同时接入SCP客户端的个数	**scp max-sessions** *max-session-count*	缺省情况下，SCP服务器允许同时接入SCP客户端的个数为2。
提交配置	**commit**	-

- **配置SSH用户。**

  详细配置请参见《配置指南-安全配置》中的“配置SSH用户”。

- **用户通过SCP进行文件操作。**

  从终端通过SCP方式上传或下载文件，需要在终端上安装支持SCP的SSH客户端软件。此处以使用第三方软件OpenSSH和Windows命令行提示符为例进行配置。

  - OpenSSH软件的安装请参考该软件的安装说明。
  - 使用OpenSSH软件从终端访问设备时，需要使用OpenSSH的命令，命令的使用可以参见该软件的帮助文档。
  - 只有安装了OpenSSH软件后，Windows命令行提示符才能识别OpenSSH相关命令。

  进入Windows的命令行提示符，执行OpenSSH命令，通过SCP方式进行文件操作。(以下显示信息仅为示意)

  ```
  C:\Documents and Settings\Administrator> scp scpuser@10.136.23.5:cfcard:/vrpcfg.zip vrpcfg-backup.zip
  ```



```
The authenticity of host '10.136.23.5 (10.136.23.5)' can't be established.
DSA key fingerprint is 46:b2:8a:52:88:42:41:d4:af:8f:4a:41:d9:b8:4f:ee.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.136.23.5' (DSA) to the list of known hosts.
scpuser@10.136.23.5's password:
vrpcfg.zip                                    100% 1257     1.2KB/s   00:00
Read from remote host 10.136.23.5: Connection reset by peer

C:\Documents and Settings\Administrator>
```

可以看到，用户终端通过SCP方式，在与远端设备建立连接的同时完成了文件上传或下载的操作，最后又回到了用户本地路径。

----结束

检查配置结果

- 使用**display ssh user-information** [*username*]命令，在SSH服务器端查看SSH用户信息。
- 使用**display ssh server status**命令，查看SSH服务器的全局配置信息。
- 使用**display ssh server session**命令，在SSH服务器端查看SSH客户端连接会话信息。

1.1.4.7.2 配置设备作为 SCP 客户端

前提条件

SCP是SSH协议的一部分，是基于SSH协议的远程文件拷贝技术，配置设备作为SCP客户端，客户端和服务器之间经过协商，建立安全连接的同时进行文件的上传和下载操作。

在配置通过SCP访问其他设备的文件之前，需完成以下任务：

- 当前设备和SSH服务器路由可达。
- 已获取SSH服务器的主机名或IP地址以及SSH用户信息。
- 如果服务器不是使用标准的端口号，则还需获取服务器端设置的端口号。

背景信息

通过SCP访问其他设备文件的配置流程如**表1-71**所示。

表 1-71 配置设备作为 SCP 客户端访问其他设备文件的配置流程

序号	配置任务名称	配置任务说明	配置流程说明
1	**（可选）配置SCP客户端源接口或源地址**	客户端源地址可以配置为源接口或源IP，达到安全校验的目的。	序号1、2、3之间没有严格的配置顺序。

序号	配置任务名称	配置任务说明	配置流程说明
2	配置设备首次连接SSH服务器的方式	有两种配置方式：使能SSH客户端首次登录功能方式和SSH客户端为SSH服务器分配公钥方式，用户可选择其一进行配置。	
3	配置SCP客户端参数	配置SCP客户端参数包括配置SCP客户端发送keepalive报文的时间间隔和最大数目等。	
4	使用SCP命令连接其他设备	-	

操作步骤

- **（可选）配置SCP客户端源接口或源地址。**

  表 1-72 （可选）配置 SCP 客户端源接口或源地址

操作步骤	命令	说明
进入系统视图	**system-view**	-
配置SCP客户端的源接口或源地址	**scp client-source** { **-a** *source-ip-address* [**public-net** | **-vpn-instance** *vpn-instance-name*] | **-i** { *interface-type interface-number* | *interface-name* } } **scp ipv6 client-source -a** *source-ipv6-address* [**-vpn-instance** *ipv6-vpn-instance-name*]	缺省情况下，SCP客户端的源IPv4地址为0.0.0.0，源IPv6地址为0::0。 当使用**-i**指定源接口为逻辑接口时，必须保证已经成功创建该逻辑接口。
提交配置	**commit**	-

- **配置设备首次连接SSH服务器的方式。**

  详细配置请参见《配置指南-安全配置》中的“配置设备首次连接SSH服务器的方式”。

- **配置SCP客户端参数。**

  详细配置请参见《配置指南-安全配置》中的“配置SSH客户端参数”。

- **使用SCP命令连接其他设备。**

  SCP与SFTP方式不同，当SCP命令执行后，与服务器建立安全连接，客户端可以直接上传文件至服务器或从服务器下载文件至本地。

表 1-73 使用 SCP 命令连接其他设备

操作步骤	命令	说明
进入系统视图	**system-view**	-
通过IPv4地址连接SCP服务器	**scp** [**-a** *source-ip-address* | **-i** *interface-type interface-number*] [**-force-receive-pubkey**] [[**-port** *server-port*] | [**public-net** | **vpn-instance** *vpn-instance-name*] | [**identity-key** *identity-key-type*] | [**user-identity-key** *user-key*] | **-r** | **-c** | [**-cipher** *cipher*] | [**-prefer-kex** *prefer-kex*]] * *source-filename destination-filename*	根据地址类型选其一。 如果使用参数**-i** *interface-type interface-number*指定源接口，则不支持**public-net**和**vpn-instance** *vpn-instance-name*参数。
通过IPv6地址连接SCP服务器	**scp ipv6** [[**vpn-instance** *vpn-instance-name*] | **public-net**] [**-force-receive-pubkey**] [[**-port** *server-port*] | [**identity-key** *identity-key-type*] | [**user-identity-key** *user-key*] | [[**-a** *source-ipv6-address*] | [**-oi** { *interface-name* | *interface-type interface-number* }]] | **-r** | **-c** | [**-cipher** *cipher*] | [**-prefer-kex** *prefer-kex*]] * *source-filename destination-filename*	

----结束

检查配置结果

- 使用**display scp-client**命令，查看SCP客户端的配置信息。
- 使用**display ssh server-info**命令，查看客户端所有的SSH服务器与公钥之间的对应关系。

1.1.4.7.3 举例：配置设备作为 SCP 客户端

组网需求

与使用SFTP协议传输文件相比，SCP协议可以简化用户传输文件的操作，将用户身份认证、文件传输等步骤合并，提高配置效率。

如图1-23所示，作为SCP客户端的设备和服务器路由可达，并从SSH服务器中下载文件至客户端。

图 1-23 配置通过 SCP 访问其他设备文件配置示例组网图

说明

本例中interface1代表GigabitEthernet1/0/1。

配置思路

采用如下的思路配置通过SCP访问其他设备文件：

1. 在SSH服务器端生成本地密钥对。
2. 在SSH服务器端创建SSH用户。
3. 在SSH服务器端使能SCP功能。
4. 从SSH服务器下载文件至本地。

操作步骤

步骤1 在服务器端生成本地密钥对。

```
<HUAWEI> system-view
[~HUAWEI] sysname SSH Server
[*HUAWEI] commit
[~SSH Server] rsa local-key-pair create
The key name will be:Host
The range of public key size is (2048, 4096).
NOTE: Key pair generation will take a short while.
Please input the modulus [default = 3072]:
[*SSH Server] commit
```

步骤2 在服务器端创建SSH用户。

配置VTY用户界面。

```
[~SSH Server] user-interface vty 0 4
[*SSH Server-ui-vty0-4] authentication-mode aaa
[*SSH Server-ui-vty0-4] protocol inbound ssh
[*SSH Server-ui-vty0-4] quit
```

新建用户名为Client的SSH用户，且认证方式为password，服务方式为all。

```
[*SSH Server] ssh user Client
[*SSH Server] ssh user Client authentication-type password
[*SSH Server] ssh user Client service-type all
```

为SSH用户Client配置密码。

```
[*SSH Server] aaa
[*SSH Server-aaa] local-user Client password
Please configure the password (8-128)
Enter Password:
Confirm Password:
[*SSH Server-aaa] local-user Client service-type terminal ssh
[*SSH Server-aaa] local-user Client level 3
[*SSH Server-aaa] quit
```

步骤3 在服务器端使能SCP服务。

```
[*SSH Server] scp server enable
[*SSH Server] ssh server-source all-interface
[*SSH Server] commit
```

步骤4 在SSH服务器端配置公钥算法、加密算法、密钥交换算法列表、HMAC认证算法和最小密钥长度。


```
[~SSH Server] ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
[~SSH Server] ssh server hmac sha2_256 sha2_512
[~SSH Server] ssh server key-exchange dh_group_exchange_sha256 dh_group16_sha512
[~SSH Server] ssh server publickey rsa_sha2_256 rsa_sha2_512
[~SSH Server] ssh server dh-exchange min-len 3072
[*SSH Server] commit
```

步骤5 在客户端配置加密算法、HMAC认证算法、密钥交换算法列表、公钥算法。

```
<HUAWEI> system-view
[~HUAWEI] sysname SCP Client
[~SCP Client] ssh client cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
[~SCP Client] ssh client hmac sha2_256 sha2_512
[~SCP Client] ssh client key-exchange dh_group_exchange_sha256 dh_group16_sha512
[~SCP Client] ssh client publickey rsa_sha2_256 rsa_sha2_512
[*SCP Client] commit
```

----结束

检查配置结果

从SCP客户端下载服务器上的文件。

第一次登录，使能SSH客户端首次登录功能。

```
<HUAWEI> system-view
[~HUAWEI] sysname SCP Client
[*HUAWEI] commit
[~SCP Client] ssh client first-time enable
[*SCP Client] commit
```

使用aes256_ctr加密算法将文件backup.cfg从IP地址为10.1.1.1的远端SSH服务器下载至本地用户目录下。

```
[~SCP Client] scp -cipher aes256_ctr Client1@10.1.1.1:backup.cfg backup.cfg
Trying 10.1.1.1 ...
Press CTRL+K to abort
Connected to 10.1.1.1 ...
 Continue to access it? [Y/N]:y
 [Y/N]:y
The server's public key will be saved with the name 10.1.1.1. Please wait...

Enter password:
backup.cfg                  100%       19174Bytes          7Kb/s
```

配置脚本

- SSH服务器上的配置脚本

```
#
sysname SSH Server
#
aaa
 local-user Client password irreversible-cipher $#z$!9S<a#>H7{7dI>%0S{AcKGC=t:zjv14LlQqHO\
\P.*=<x1]u;y*P`'GR3[m}$
 local-user Client service-type terminal ssh
 local-user Client level 3
#
scp server enable
ssh user Client
ssh user Client authentication-type password
ssh user Client service-type all
ssh server-source all-interface
#
user-interface vty 0 4
 authentication-mode aaa
 protocol inbound ssh
#
```

```
ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
ssh server hmac sha2_256 sha2_512
ssh server key-exchange dh_group_exchange_sha256 dh_group16_sha512
ssh server publickey rsa_sha2_256 rsa_sha2_512
ssh server dh-exchange min-len 3072
#
return
```

- SCP客户端上的配置脚本

```
#
sysname SCP Client
#
ssh client first-time enable
#
ssh client cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
ssh client hmac sha2_256 sha2_512
ssh client key-exchange dh_group_exchange_sha256 dh_group16_sha512
ssh client publickey rsa_sha2_256 rsa_sha2_512
#
return
```

1.1.4.8 通过 TFTP 进行文件操作

1.1.4.8.1 配置设备作为 TFTP 客户端

前提条件

当前设备可以作为TFTP客户端登录远端TFTP服务器，实现文件的上传和下载。

在配置通过TFTP访问其他设备的文件之前，需完成以下任务：

- 当前设备和TFTP服务器路由可达。
- 已获取TFTP服务器的IP地址以及下载或上传文件所在的目录。
- 已执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

背景信息

说明

使用TFTP协议存在安全风险，建议使用SFTP V2或SCP方式进行文件操作。

FIPS模式下，不支持使用TFTP协议。

通过TFTP访问其他设备文件的配置流程如**表1-74**所示。

表 1-74 配置设备作为 TFTP 客户端访问其他设备的文件配置流程

序号	配置任务名称	配置任务说明	配置流程说明
1	（可选）配置TFTP客户端源地址	客户端源地址可以配置为源接口或源IP，达到安全校验的目的。	在执行任务3前，任务1、2之间没有严格的配置顺序。
2	（可选）配置TFTP访问限制	配置TFTP的访问控制，提高TFTP访问的安全性。	

序号	配置任务名称	配置任务说明	配置流程说明
3	使用TFTP命令向其他设备上传或下载文件	包括文件的上传和下载操作。	

操作步骤

- **（可选）配置TFTP客户端源接口或源地址。**

  配置源地址需要选择设备上状态稳定的接口，如LoopBack接口。该配置简化了ACL规则的配置，只要将ACL规则的源地址或目的地址指定为该地址，就可以屏蔽接口IP地址的差异以及接口状态的影响，实现对设备进出报文的过滤。

  表 1-75 （可选）配置 TFTP 客户端源接口或源地址

操作步骤	命令	说明
进入系统视图	**system-view**	-
配置TFTP客户端的源接口或源地址	**tftp client-source** { **-a** *ip-address* | **-i** *interface-type interface-number* } **tftp ipv6 client-source -a** *ipv6-address* [**-vpn-instance** *ipv6-vpn-instance-name*]	缺省情况下，TFTP客户端的源IPv4地址为0.0.0.0，源IPv6地址为0::0。如果是源接口，必须要为该接口配置IP地址，否则会导致TFTP连接建立失败。
提交配置	**commit**	-

- **（可选）配置TFTP访问限制。**

  ACL是一系列有顺序的规则组的集合，这些规则根据数据包的源地址、目的地址、端口号等来描述。ACL通过规则对数据包进行分类，这些规则应用到路由设备，路由设备根据这些规则判断哪些数据包可以接收，哪些数据包需要拒绝。

  每个ACL中可以定义多个规则，根据规则的功能分为基本ACL规则、高级ACL规则和二层ACL规则等。

  说明

  TFTP只支持基本访问控制列表（编号范围为2000～2999）。

  ACL规则：

  - 当ACL的rule配置为**permit**时，则允许本设备与匹配该rule规则的其他设备建立TFTP连接。
  - 当ACL的rule配置为**deny**时，则拒绝本设备与匹配该rule规则的其他设备建立TFTP连接。
  - 当ACL配置了rule，但来自其他设备的报文没有匹配该rule规则时，则拒绝本设备与其他设备建立TFTP连接。
  - 当ACL未配置rule时，则允许本设备与任何其他设备建立TFTP连接。

表 1-76 （可选）配置 TFTP 访问限制

操作步骤	命令	说明
进入系统视图	**system-view**	-
创建一个ACL访问控制列表，并进入ACL视图	**acl** { [**number**] *acl-number* | **name** *acl-name* }	缺省情况下，未创建ACL访问控制列表。
配置ACL规则	**rule** [*rule-id*] [**name** *rule-name*] { **permit** | **deny** } [**fragment-type fragment** | **source** { *source-ip-address* { *source-wildcard* | **0** | *src-netmask* } | **any** } | **time-range** *time-name* | **vpn-instance** *vpn-instance-name* | **logging**] *	缺省情况下，基本ACL视图下没有规则。
退回到系统视图	**quit**	-
配置TFTP访问限制	**tftp-server** [**ipv6**] **acl** *acl-number*	-
提交配置	**commit**	-

- **使用TFTP命令向其他设备上传文件或从其他设备下载文件**。

表 1-77 使用 TFTP 命令连接其他设备

操作步骤	命令	说明
通过IPv4地址连接TFTP服务器	**tftp** [**-a** *source-ip-address* | **-i** *interface-type interface-number*] *host-ip-address* [**vpn-instance** *vpn-instance-name* | **public-net**] { **get** | **put** } *source-filename* [*destination-filename*]	• **get**表示从其他设备下载文件操作。 • **put**表示向其他设备上传文件操作。
通过IPv6地址连接TFTP服务器	**tftp ipv6** [**-a** *source-ipv6-address*] *tftp-server-ipv6* [**vpn-instance** *vpn-instance-name* | **public-net**] [**-oi** *interface-type interface-number*] { **get** | **put** } *source-filename* [*destination-filename*]	

此命令中指定的源地址或者接口优先级高于**tftp client-source**命令中指定的源地址或者接口。**tftp client-source**命令指定的源地址或者接口对所有的TFTP连接都有效，**tftp**和**tftp ipv6** 命令指定的源地址或者接口只对当前的TFTP连接有效。

----结束

检查配置结果

- 执行**display tftp-client**命令，查看设备作为TFTP客户端时的源地址。

1.1.4.8.2 举例：配置设备作为 TFTP 客户端

组网需求

如图1-24所示，远端服务器提供TFTP Server功能，IP地址为10.1.1.1/24。设备作为TFTP客户端，IP地址为10.2.1.1/24，与服务器之间的路由可达。

设备需要进行升级操作，要求：从TFTP服务器上下载系统软件至设备，且备份当前设备的配置文件到TFTP服务器。

图 1-24 配置通过 TFTP 访问其他设备文件组网图

说明

使用TFTP协议存在安全风险，建议使用SFTP V2或SCP方式进行文件操作。

本例中interface1代表GigabitEthernet1/0/1。

配置思路

采用如下的思路配置TFTP传输文件功能：

1. 在TFTP服务器端运行TFTP软件，并设置TFTP的工作路径。
2. 在设备上使用TFTP命令下载和上传文件。

配置注意事项

当网络所处环境不足够安全时，我们建议选择安全的密码认证方式/加密认证算法/协议。安全的举例请参见：**配置设备作为SCP客户端**。

操作步骤

步骤1 在用户视图下执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

步骤2 在TFTP服务器端运行TFTP软件，并设置TFTP的工作路径。（具体操作见第三方软件帮助文档）

步骤3 在设备上使用TFTP命令下载和上传文件。

```
<HUAWEI> tftp 10.1.1.1 get devicesoft.cc
Info: Transfer file in binary mode.
```

```
Please wait for a while...
/    107973953 bytes transferred
Info: Downloaded the file successfully.
<HUAWEI> tftp 10.1.1.1 put vrpcfg.zip
Info: Transfer file in binary mode.
Please wait for a while...
/     100% [**********]
Info: Uploaded the file successfully.
```

----结束

检查配置结果

在设备中执行**dir**命令，查看系统软件是否下载至设备。

```
<HUAWEI> dir
Directory of cfcard:/

 Idx  Attr     Size(Byte)  Date        Time       FileName
   0  -rw-             14  Mar 13 2019 14:13:38   back_time_a
   1  drw-              -  Mar 11 2019 00:58:54   logfile
   2  -rw-              4  Nov 17 2019 09:33:58   snmpnotilog.txt
   3  -rw-         11,238  Mar 12 2019 21:15:56   private-data.txt
   4  -rw-          1,257  Mar 12 2019 21:15:54   vrpcfg.zip
   5  -rw-             14  Mar 13 2019 14:13:38   back_time_b
   6  -rw-    107,973,953  Mar 13 2019 14:24:24   devicesoft.cc
   7  drw-              -  Oct 31 2019 10:20:28   sysdrv
   8  drw-              -  Feb 21 2019 17:16:36   compatible
   9  drw-              -  Feb 09 2019 14:20:10   selftest
  10  -rw-         19,174  Feb 20 2019 18:55:32   backup.cfg
  11  -rw-         23,496  Oct 15 2019 20:59:36   20191015.zip
  12  -rw-            588  Nov 04 2019 13:54:04   servercert.der
  13  -rw-            320  Nov 04 2019 13:54:26   serverkey.der
  14  drw-              -  Nov 04 2019 13:58:36   security
...
670,092 KB total (569,904 KB free)
```

在TFTP服务器的工作路径下，可以看到**vrpcfg.zip**文件已保存至此路径。

配置脚本

无

1.1.4.9 文件系统管理常见配置错误

1.1.4.9.1 FTP 登录失败

故障原因

- FTP服务器功能没有启动。
- FTP服务器指定端口号不是缺省端口号，且FTP客户端登录时没有指定端口号。
- 未配置FTP用户的验证信息、工作目录及用户级别。
- 登录FTP服务器的用户数达到上限。
- FTP服务器配置了ACL规则限制客户端登录。

操作步骤

步骤1 检查FTP服务器功能是否启动。

在任意视图下执行命令**display ftp-server**查看FTP服务器的状态。

- 如果FTP服务器没有启动，显示信息如下：

```
<HUAWEI> display ftp-server
Server state              : Disabled
IPv6 server state         : Disabled
Timeout value (mins)      : 10
IPv6 Timeout value (mins) : 10
Listen port               : 21
IPv6 listen port          : 21
ACL name                  :
IPv6 ACL name             :
ACL number                :
IPv6 ACL number           :
Current user count        : 0
Max user number           : 15
```

在系统视图下执行命令**ftp server enable**，使能FTP服务器功能。

```
<HUAWEI> system-view
[~HUAWEI] ftp server enable
Warning: FTP is not a secure protocol, and it is recommended to use SFTP.
[*HUAWEI] commit
```

- 如果FTP服务器功能启动，显示信息如下：

```
<HUAWEI> display ftp-server
Server state              : Enabled
IPv6 server state         : Disabled
Timeout value (mins)      : 10
IPv6 Timeout value (mins) : 10
Listen port               : 21
IPv6 listen port          : 21
ACL name                  :
IPv6 ACL name             :
ACL number                :
IPv6 ACL number           :
Current user count        : 0
Max user number           : 15
```

步骤2 检查FTP服务器的端口号是否是缺省端口号。

在任意视图下执行命令**display ftp-server**查看FTP服务器的端口号。

```
<HUAWEI> display ftp-server
Server state              : Enabled
IPv6 server state         : Disabled
Timeout value (mins)      : 10
IPv6 Timeout value (mins) : 10
Listen port               : 21
IPv6 listen port          : 21
ACL name                  :
IPv6 ACL name             :
ACL number                :
IPv6 ACL number           :
Current user count        : 0
Max user number           : 15
```

如果当前FTP服务器的端口号不是21，执行命令**ftp server port**，设置FTP服务器的端口号为21。

```
<HUAWEI> system-view
[~HUAWEI] undo ftp server
Info: Succeeded in closing the FTP server.
[*HUAWEI] ftp server port 21
[*HUAWEI] ftp server enable
Warning: FTP is not a secure protocol, and it is recommended to use SFTP.
[*HUAWEI] commit
```

或者在FTP连接时，在客户端指定服务端设置的端口号。

步骤3 检查是否配置FTP用户的验证信息、授权目录及用户级别。

FTP用户名、密码、授权目录和用户级别是必配置项。没有指定FTP授权目录及用户级别而登录失败是常见故障。

详细配置参见“配置设备作为FTP服务器”中的**配置FTP本地用户**。

步骤4 检查登录FTP服务器的用户数是否达到上限。

执行命令**display ftp-users**，查看FTP用户数是否达到15个。

步骤5 检查FTP服务器端是否配置了ACL。

执行命令**display ftp-server**，查看FTP服务器端是否配置了ACL。

如果配置了ACL规则，系统仅允许在ACL规则列表中指定的IP地址登录FTP服务器。

----结束

1.1.4.9.2 FTP 传输失败

故障原因

- FTP源、目的路径中含有空格等设备不支持的字符。
- FTP服务器根目录存储空间不足。

操作步骤

步骤1 FTP源、目的路径中含有空格等设备不支持的字符。

设备中目录名使用的字符不可以是空格、“~”、“*”、“/”、“\”、“:”、“'”、“"”。

如果路径中含有空格等设备不支持的字符，请修改路径。

步骤2 检查FTP服务器根目录存储空间是否不足。

在FTP服务器端执行命令**dir**，查看FTP服务器根目录下的空闲空间。

如果存储空间已满，在用户视图下执行命令**delete /unreserved**删除不需要的文件。

----结束

1.1.5 配置文件管理配置

1.1.5.1 配置文件管理简介

定义

配置文件是能够在设备上运行的命令行的集合。

目的

通过对配置文件的管理，除了可以实现查看、保存、比较、替换、备份、恢复、压缩配置文件以及清除、回退配置的功能外，还可以指定下次启动时加载的配置文件，从而确保用户配置在设备上正常运行，避免用户配置丢失，方便用户配置移植等。

1.1.5.3 管理配置文件

1.1.5.3.1 缺省配置文件

设备出厂时带有缺省配置文件default.cfg，内容如下：

- **路由器模式**

```
#
sysname HUAWEI
#
crypto weak-algorithm disable
#
lldp enable
#
undo telnet server enable
undo telnet ipv6 server enable
#
undo icmp name timestamp-reply send
#
security password
 #
 rule admin
  forbidden word changeme_123
#
undo snmp-agent protocol source all-interface
#
undo ssh server compatible-ssh1x enable
stelnet server enable
snetconf server enable
undo ssh server-source all-interface
undo ssh ipv6 server-source all-interface
ssh server cipher aes256_gcm aes128_gcm aes256_ctr aes192_ctr aes128_ctr
ssh server hmac sha2_512 sha2_256
ssh server key-exchange dh_group_exchange_sha256
#
ssh server publickey rsa_sha2_256 rsa_sha2_512
#
ssh server dh-exchange min-len 3072
ssh server rsa-key min-length 3072
#

ssh client publickey rsa_sha2_256 rsa_sha2_512
#
ssh client cipher aes256_gcm aes128_gcm aes256_ctr aes192_ctr aes128_ctr
ssh client hmac sha2_512 sha2_256
ssh client key-exchange dh_group_exchange_sha256
#
user-interface vty 0 4
 authentication-mode aaa
 idle-timeout 10 0
 protocol inbound ssh
#
return
```

- **传输模式**

```
#
 sysname HUAWEI
#
crypto weak-algorithm disable
#
undo telnet server enable
undo telnet ipv6 server enable
#
undo icmp name timestamp-reply send
#
 undo radius enable
#
dot1x-template 1
```

```
#
 dhcpv6 disable
#
security password
 #
 rule admin
  forbidden word changeme_123
#
undo snmp-agent protocol source all-interface
#
aaa
 authentication-scheme default0
 authentication-scheme default1
 authentication-scheme default
  authentication-mode local radius
 #
 authorization-scheme default
 #
 accounting-scheme default0
 accounting-scheme default1
 #
 domain default0
 domain default1
 domain default_admin
 #
#
multicastbandwidth
#
interface GigabitEthernet0/0/0
 speed auto
 duplex auto
 undo shutdown
 ip address 192.168.0.1 255.255.0.0
#
interface NULL0
#
l2tp-group default-lns
#
undo ssh server compatible-ssh1x enable
stelnet server enable
snetconf server enable
undo ssh server-source all-interface
undo ssh ipv6 server-source all-interface
ssh server cipher aes256_gcm aes128_gcm aes256_ctr aes192_ctr aes128_ctr
ssh server hmac sha2_512 sha2_256
ssh server key-exchange dh_group_exchange_sha256
#
ssh server publickey rsa_sha2_256 rsa_sha2_512
#
ssh server dh-exchange min-len 3072
ssh server rsa-key min-length 3072
#
ssh client publickey rsa_sha2_256 rsa_sha2_512
#
ssh client cipher aes256_gcm aes128_gcm aes256_ctr aes192_ctr aes128_ctr
ssh client hmac sha2_512 sha2_256
ssh client key-exchange dh_group_exchange_sha256
#
user-interface maximum-vty 15
user-interface con 0
user-interface aux 0
user-interface vty 0 14
 authentication-mode aaa
 protocol inbound ssh
 idle-timeout 10 0
#
multicast shaping
#
mpls
```



```
mpls te
mpls rsvp-te
mpls rsvp-te hello
mpls rsvp-te hello support-peer-gr
mpls te cspf
mpls oam based-itu
#
mpls l2vpn
#
mpls ldp
remote-peer pwe3
#
ssh server-source -i GigabitEthernet0/0/0
#
local-aaa-server
#
dtls policy qx_dtls_client
#
dcn security-mode enable
#
return
```

配置文件只有管理级别的用户才可以保存和修改，是在运行过程中对配置文件作为系统文件的保护机制（不允许删除和修改）。

1.1.5.3.2 空配置启动下的默认配置文件

当设备升级到当前版本并空配置启动时，需要工程师进行现场配置，设备的运维管理很不方便。如果希望设备空配置启动后实现即插即用，可以对设备预置一个默认配置文件，具体请参考**配置下次启动时加载的配置文件**，默认配置文件中携带满足即插即用的一些配置，设备下次空配置启动后就会使用预置的默认配置文件进行配置恢复。default-custom.defcfg文件是设备自带的默认配置文件，只用于设备首次上线，用户可以根据需要自行修改，具体内容如下：

- **路由器模式**

```
#
undo crypto weak-algorithm disable
#
aaa
 local-user root password irreversible-cipher $1c$]f(3Q<j7uS$!0!)8@e`\+lj]vQx\2l&y-$M(|\n_ERFU_BF$!
6X$
 local-user root service-type ssh
 local-user root user-group manage-ug
 local-user root expire 2000-01-01
 user-password password-force-change disable
#
snmp-agent protocol source all-interface
#
stelnet server enable
snetconf server enable
ssh user root
ssh user root authentication-type password
ssh user root service-type stelnet snetconf
ssh server-source all-interface
ssh ipv6 server-source all-interface
#
ssh server key-exchange dh_group_exchange_sha256 dh_group_exchange_sha1 dh_group14_sha1
ecdh_sha2_nistp256 ecdh_sha2_nistp384 ecdh_sha2_nistp521
#
ssh server publickey ecc rsa rsa_sha2_256 rsa_sha2_512
#
ssh client key-exchange dh_group_exchange_sha256 dh_group_exchange_sha1 dh_group14_sha1
ecdh_sha2_nistp256 ecdh_sha2_nistp384 ecdh_sha2_nistp521
#
ssh client publickey ecc rsa rsa_sha2_256 rsa_sha2_512
#
return
```


- **传输模式**

```
#
undo crypto weak-algorithm disable
#
aaa
 local-user root password irreversible-cipher $1c$]f(3Q<j7uS$!0!)8@e`\+lj]vQx\2l&y-$M(|\n_ERFU_BF$!
6X$
 local-user root service-type ssh mml
 local-user root user-group manage-ug
 local-user root expire 2000-01-01
 user-password password-force-change disable
#
snmp-agent protocol source all-interface
#
stelnet server enable
snetconf server enable
ssh user root
ssh user root authentication-type password
ssh user root service-type stelnet snetconf
ssh server-source all-interface
ssh ipv6 server-source all-interface
#
ssh server key-exchange dh_group_exchange_sha256 dh_group_exchange_sha1 dh_group14_sha1
ecdh_sha2_nistp256 ecdh_sha2_nistp384 ecdh_sha2_nistp521
#
ssh server publickey ecc rsa rsa_sha2_256 rsa_sha2_512
#
ssh client key-exchange dh_group_exchange_sha256 dh_group_exchange_sha1 dh_group14_sha1
ecdh_sha2_nistp256 ecdh_sha2_nistp384 ecdh_sha2_nistp521
#
ssh client publickey ecc rsa rsa_sha2_256 rsa_sha2_512
#
undo dcn security-mode enable
#
undo dtls policy qx_dtls_client
#
return
```

说明

文件中的密码支持设置为用户自定义的明文类型，为了提高安全性，设置的密码建议包含多种类型字符，包括大写字母、小写字母、数字及特殊字符。

1.1.5.3.3 了解配置文件

配置文件的格式

配置文件为文本文件，需要满足以下要求：

- 配置文件中只能包含配置命令、视图切换命令、#（功能为跳转到系统视图下）、**quit**命令，其他类型（如display查询命令、reset/save/ping等维护命令、return、升级兼容命令等）命令执行时设备会报错，并继续加载后续命令。
- 配置文件中不能有重复命令。
- 命令缩进需要正确（一级视图对应缩进一个字符）。
- 视图下的命令，进入视图的命令一定要带上。
- 配置顺序和依赖（相互依赖的配置命令及顺序）要保证正确。
- 配置文件中的交互类型命令仅支持Y/N自动交互，配置恢复时默认是Y。
- 配置文件必须以*.zip、*.cfg或者*.dat作为扩展名，而且必须存放在存储器的根目录下。
  - *.cfg为纯文本格式，可直接查看其内容。指定为配置文件后，启动时系统对里面的命令逐条进行恢复。

- *.zip是*.cfg格式的压缩，占用空间较小。指定为配置文件后，启动时先解压成.cfg格式，然后逐条恢复。
- *.dat是二进制数据格式。当设备启动的版本与保存该配置文件的版本号一致时，启动时直接加载配置数据，无需逐条执行命令进行恢复，从而实现快速启动。

配置文件的分类

设备运行过程中，有出厂缺省配置、预置配置、当前配置、离线配置和下次启动配置，区别如下表：

概念	描述	查看方式
出厂缺省配置	设备在出厂时，通常会被安装一些基本的配置，称为出厂配置。出厂配置用来保证设备在没有配置文件或者配置文件丢失、损坏的情况下，能够正常启动、运行。当设备以出厂缺省配置启动时，称为空配置启动。 设备的出厂配置并不等同于命令行的缺省情况。例如设备在出厂时可能会包含**snetconf server enable**命令，但是该命令的缺省情况是没有使能SNETCONF服务，此时设备在空配置启动或者恢复出厂配置时，SNETCONF服务是处于使能的状态。	-
预置配置	设备上电时，从默认存储路径中读取配置文件进行设备的初始化操作，因此该配置文件中的配置称为预置配置。如果默认存储路径中没有配置文件，则设备用出厂配置初始化配置。	使用**display startup**命令可以查看到设备本次启动的配置文件。
当前配置	与初始配置相对应，设备运行过程中正在生效的配置称为当前配置。	使用**display current-configuration**命令查看设备的当前配置信息。
下次启动的配置	系统启动后，用户可以根据需要设置指定的配置文件作为下次启动时的初始配置，即下次启动的配置。	使用**display startup**命令可以查看到设备下次启动的配置文件。 使用**display saved-configuration**命令可以查看设备下次启动时的配置文件信息。

概念	描述	查看方式
离线配置	当前系统支持离线配置。当需要更换单板时，如果接口上已经配置了命令，拔出单板后，此时单板配置不丢失，并且仍然能够对接口进行配置。当在单板槽位插入其他单板时，根据插入单板的类型不同，对原有配置的影响如表1-78。	离线配置信息通过前导符“*”标识。可以使用**display current-configuration inactive**或**display current-configuration all**命令查看设备的离线配置信息。

表 1-78 插入不同类型的单板对原有配置的影响

插入单板的类型	影响
同一槽位插入和之前类型相同的单板	系统会自动恢复该单板上所有接口的配置。此时用户又可以看到该单板上的接口，也能对它们进行配置。
同一槽位插入和之前接口类型不同的单板	系统会先删除原来单板的接口配置。即使用户不对现在的单板进行配置，就再拔出，然后再插入原来拔出的单板，这些配置也不会再被恢复。 例如：单板A的接口类型为P，单板A上有配置信息，进行不同的操作及对应的影响为： 1. 先把单板A拔出后，插入接口类型为E的单板C。不对单板C进行任何配置，再拔出单板C。在单板C的接口视图下通过**display this**命令，不能看到原来在单板A上的配置信息了。 2. 再插入单板A或接口类型为P的其他单板。在单板A或接口类型为P的其他单板的接口视图下通过**display this**命令，也不能看到原来在单板A上的配置信息。
插入接口数目不同但接口类型相同的单板，且插入的单板上的接口数目比之前拔出的单板上的接口数目多	对于与之前单板接口类型和编号都相同的接口，设备会直接恢复之前保留的接口配置信息；对于其余接口，接口下只保留默认配置信息。
插入接口数目不同但接口类型相同的单板，且插入的单板上的接口数目比之前拔出的单板上的接口数目少	对于与之前单板接口类型和编号都相同的接口，设备直接恢复之前保留的接口配置信息；对于拔单板多余接口的配置信息，设备会全部删除。

用户通过命令行接口可以修改设备当前配置，为了使当前配置能够作为设备下次启动时的起始配置，需要使用**save**命令保存当前配置到默认存储器中，形成配置文件。

说明

如果使用不完整格式进行配置，由于命令保存到配置文件中时使用的是完整格式，可能导致配置文件中存在长度超过510个字符的命令（系统可正确执行的命令长度最大为510个字符）。系统重启时，这类命令将无法恢复。

1.1.5.3.4 查看配置文件

操作步骤

表 1-79 查看配置文件

操作	命令	说明
查看存储器中的配置文件名	**dir**	-
查看设备本次以及下次启动的配置文件名	**display startup**	-
查看设备指定配置文件的配置信息	**display configuration** *configuration-file*	-
查看设备下次启动时的配置文件信息	**display saved-configuration**	-
查看上次保存的系统配置信息	**display saved-configuration last**	-
查看上次自动保存配置信息的时间	**display saved-configuration time**	-
查看设备当前生效的所有配置信息	**display current-configuration** [**include-default**]	如果不选择**include-default**，则只显示用户配置的信息；如果选择**include-default**，显示信息中除了用户配置的信息，还会显示设备缺省的配置信息。
在某一视图下查看当前视图下生效的配置信息	**display this** [**include-default**]	在不同的视图下使用该命令，将直接显示相应视图下的配置。 如果不选择**include-default**，则只显示该视图下用户配置的信息；如果选择**include-default**，显示信息中除了该视图下用户配置的信息，还会显示该视图下设备缺省的配置信息。

操作	命令	说明
查看设备的离线配置信息	**display current-configuration inactive** **display current-configuration all**	显示的配置前面带星号“*”，表示此配置为离线配置。

1.1.5.3.5 保存配置文件

背景信息

用户通过命令行可以修改设备的当前配置，而这些配置是暂时的，如果要使当前配置在系统下次重启时仍然有效，在重启设备前，需要将当前配置保存到配置文件中。可以通过以下两种方法保存配置文件：

- 自动保存配置。
- 手动保存配置。

设备的配置信息保存在存储器的配置文件中，启动运行时会读取配置文件恢复对此设备的配置信息，并在内存中保留一份当前运行的配置信息。

可以通过命令**display saved-configuration**查看配置文件中的配置信息，通过命令**display current-configuration**查看内存中的配置信息。

在系统设备正常运行后，配置文件和内存中的配置信息应该是一致的，当增加、修改、删除配置后，配置文件中的配置信息和内存中的配置信息不一致，此时可以通过命令**save**来保存当前内存中的配置信息到配置文件。

在系统启动阶段，如果设备还没有正常运行，即配置文件还未完全恢复到内存中，此时执行命令**save**保存，内存中不完整的配置信息将会覆盖配置文件中的配置信息，造成部分配置信息丢失，所以设备没有正常运行前不能执行**save**命令保存配置文件。

操作步骤

- 自动保存配置。
  a. 进入系统视图。
     system-view
  b. 配置系统定时保存配置功能。
     set save-configuration [**interval** *interval* | **delay** *delay-interval* | **cpu-limit** *cpu-usage*] *
     缺省情况下，系统没有使能系统定时保存配置功能。
     当出现如下情况时，系统会取消定时保存配置文件的操作：
     - 当前存在写配置文件操作。
     - 设备正在进行配置恢复。
     - CPU利用率较高。
  c. （可选）配置服务器的相关信息，包括自动保存配置文件的服务器的IP地址、用户名及其密码、配置文件自动保存的目的路径和配置文件自动保存至服务器的传输方式。

set save-configuration backup-to-server server [**ipv6**] *server-ip* [**vpn-instance** *vpn-instance-name*] **transport-type** { **tftp** | { **ftp** | **sftp** } [**port** *port-value*] **user** *user-name* **password** *password* } [**path** *folder*]

缺省情况下，系统关闭定时保存配置文件至服务器的功能。

说明

当本地存储器空间不足或损坏，或当需要将配置文件备份时，可配置此命令指定服务器的相关信息。

建议使用安全性更高的SFTP协议保存配置文件至文件服务器。

配置文件以压缩包的形式保存在服务器上，压缩包的命名规则是年-月-日.小时-分钟-秒.设备名称.zip，例如：2019-10-25.15-13-37.HUAWEI.zip。解压后，后缀名为.cfg的文件即为配置文件。

d. 提交配置。

commit

- 手动保存配置。
  - 直接保存配置文件。

    save [*configuration-file*]

    将当前配置保存到指定文件时，文件必须以“.zip”、“.dat”或“.cfg”作为扩展名。而且系统启动配置文件必须存放在存储器的根目录下。

    如果不指定可选参数*configuration-file*，第一次保存配置文件时设备将提示是否将文件名保存为“vrpcfg.zip”。“vrpcfg.zip”是系统缺省的配置文件，初始状态是空配置。若不是第一次保存，配置信息将保存至系统当前启动配置文件里，执行**display startup**命令可以查看系统当前启动配置文件的文件名。
  - 输入密码保存配置文件。

    save shareable-configuration *configuration-file* [**password**]

    设备会根据用户输入的密码参数在配置文件中生成对应的密钥信息，当下次使用该配置文件时，需要输入密码进行验证才可以使用。

    说明

    开启弱密码字典维护功能后，弱密码字典中定义的密码（可以通过命令**display security weak-password-dictionary**查看）不能在该命令中配置

----结束

1.1.5.3.6 配置下次启动时加载的配置文件

背景信息

系统重新启动后使用指定的配置文件进行配置恢复，用户可以根据需要设定此配置文件。

在进行配置前，用户可以使用**display startup**命令查看当前设备指定的下次启动时加载的配置文件。

如果没有配置下次启动时加载的配置文件，则下次启动采用缺省配置文件（default.cfg）。

说明

不建议手工构造配置文件。如果构造配置文件格式错误，可能导致配置恢复失败或者配置恢复发生错误。

系统下次启动时使用的配置文件必须存在且必须存放在存储器的根目录下。

配置该功能后，不能在用户视图下执行不带参数的**save**命令，如果执行**save**命令，系统将使用保存后的配置文件作为下次启动的配置文件，当前配置的下次启动配置文件不会生效。

操作步骤

- 配置系统下次启动时使用的配置文件。
  startup saved-configuration *configuration-file*
- 配置系统下次启动时使用携带密钥信息的配置文件。
  startup shareable-configuration *configuration-file* [**password**]

  当配置的启动文件携带密钥信息时，需要用户输入密码验证成功后才可以使用。
- 配置系统预置的默认配置文件。
  startup default-configuration *configuration-file*

  当系统没有设定下次启动使用的配置文件时，系统启动时会使用该该预置的默认配置文件。

----结束

检查配置结果

执行命令**display configuration recover-result**查看重启后配置恢复的结果以及造成失败的原因。

1.1.5.3.7 复用其他设备的配置文件

背景信息

配置文件中存在使用系统主密钥加密保存的密文配置时，当使能系统主密钥的自动更新功能后，由于系统主密钥默认是自动随机生成，设备间是不相同的，配置文件中的密文配置无法在另一个设备上解密，密文在新设备上会被作为明文生效，即原始密文配置无法在新设备上正常恢复。如果需要将配置文件中的密文配置按照原始明文在新设备上生效，可参考如下操作步骤。

操作步骤

步骤1 在设备A上导出配置文件。

1. 保存配置文件。
   save shareable-configuration *configuration-file* [**password**]
2. 导出配置文件。

   详细配置请参见**备份配置文件到SFTP服务器或客户端**。

步骤2 在设备B上复用设备A导出的配置文件。

1. 复制设备A的配置文件至设备B。

   详细配置请参见**从SFTP服务器或客户端复制配置文件到设备**。
2. 配置导出的配置文件为设备B的下次启动时加载的配置文件。

startup shareable-configuration *configuration-file* [**password**]

----结束

检查配置结果

- 执行命令**display configuration recover-result**查看重启后配置恢复的结果以及造成失败的原因。
- 执行命令**display master-key configuration**查看系统主密钥应为用户设置的主密钥。

1.1.5.3.8 替换配置文件

背景信息

当设备的配置文件需要更新时，可将本地或远端服务器上的配置文件加载到本地设备，并替换当前正在运行的配置文件。

此功能可替换当前设备上的整个配置文件，也可以只替换某个视图下的配置，这取决于待加载的配置文件。如果待加载的配置文件中包含整个设备的配置，则替换当前设备上的所有配置，如果待加载的配置文件是某个视图下保存的配置（视图下保存的配置自动携带<replace/>标签），则替换相应视图下的配置。

说明

指定的待加载配置文件必须存在，且必须满足以下条件：

- 配置文件中只能包含配置命令、视图切换命令、#，其他类型（如**display**查询命令、**reset**/**save**/**ping**等维护命令、**quit**、**commit**、**return**、升级兼容命令等）命令执行时设备会报错，并继续加载后续命令。
- 配置文件中的交互类型命令仅支持Y/N自动交互。
- 配置文件中命令的缩进要正确。系统视图下的命令需顶格，系统视图下的一级视图需顶格，一级视图下的命令需缩进一个空格，多级视图依次缩进一格。

  #号如果顶格，表示回退到系统视图；非顶格，则只是用来隔离命令块，但缩进需正确，和其下方的命令块对齐。如果使用错误，可能会导致配置丢失或者命令在非预期视图执行。
- 配置文件必须以*.zip、*.cfg、*.txt、*.dat、*.bat作为扩展名。其中，
  - *.cfg和*.txt为纯文本格式，可直接查看其内容。指定为待加载的配置文件后，替换时系统对里面的命令逐条进行恢复。
  - *.zip是*.cfg格式文件的压缩，占用空间较小。指定为待加载的配置文件后，系统先将文件解压成.cfg格式，然后逐条恢复。*.cfg文件名必须和*.zip文件名一致，否则会导致配置文件加载失败。
  - *.dat是压缩文件格式，可以是二进制格式或者文本格式文件。只能执行从华为设备导出的*.dat文件，且不能手工修改，否则会导致配置文件加载失败。
  - *.bat是批处理文件，为纯文本格式文件，可以手工修改。

操作步骤

步骤1 进入系统视图。

system-view

步骤2 根据待加载的配置文件所在位置不同，选择执行以下命令：

- 加载本地设备上的配置文件，替换当前设备上正在运行的配置。

  load configuration file *filename* **replace** [**relative**]

- 加载指定的远端管理服务器上的配置文件，替换当前设备上正在运行的配置。

  load configuration { **server** *ip-address* | **server ipv6** *ipv6-address* } [**vpn-instance** *vpn-instance-name*] **transport-type** { **ftp** | **sftp** } **username** *user-name* **password** *password* **file** *filename* **replace** [**relative**]

- 加载指定URL获取远端管理服务器上的配置文件，替换当前设备上正在运行的配置。

  load configuration server http url *url-address* [**vpn-instance** *vpn-instance-name*] [**file** *filename*] **replace** [**relative**]

----结束

检查配置结果

完成配置后，可以按以下指导来检查配置结果。

- 执行命令**display configuration candidate**，查看用户未提交的配置，确认替换后的配置是否符合预期。
- 执行命令**display configuration replace file**，查看配置文件替换后和目标配置文件的差异内容。
- 执行命令**display configuration replace failed**，查看配置文件替换失败的命令详细信息和失败原因。

1.1.5.3.9 加载配置文件

背景信息

在设备正常运行过程中，通过加载配置文件可以批量执行命令行，进行功能配置。

说明

配置下发后需提交配置，新加载的配置才能生效。

操作步骤

步骤1 进入系统视图。

system-view

步骤2 根据加载的配置文件所在位置，选择执行如下命令之一：

- 加载本地配置文件并下发配置。

  load configuration file *filename* **merge** [**relative**]

- 加载指定远端服务器上的配置文件，并下发配置到当前设备。

  load configuration { **server** *ip-address* | **server ipv6** *ipv6-address* } [**vpn-instance** *vpn-instance-name*] **transport-type** { **ftp** | **sftp** } **username** *user-name* **password** *password-value* **file** *filename* **merge** [**relative**]

- 根据指定URL获取远端服务器上的配置文件，并下发配置到当前设备。

  load configuration server http url *url-address* [**vpn-instance** *vpn-instance-name*] **merge** [**relative**]

该命令支持下发整个配置文件中的配置，如果执行命令时携带参数**relative**，则仅下发当前视图下的配置。

步骤3 提交配置。

commit

----结束

1.1.5.3.10 比较配置文件

背景信息

用户可以通过比较当前运行配置和指定配置文件，查看哪些配置项是不一致的，决定是否需要将指定配置文件设置为下次启动时加载的配置文件。

说明

所比较的配置文件必须以“.cfg”或“.zip”作为扩展名。

操作步骤

表 1-80 比较配置文件

操作	命令	说明
比较当前运行配置与指定的配置文件的内容是否一致。	**display configuration changes** [{ **running file** *file-name* | **file** *file-name* **running** } [**with-tag**]]	配置差异比较只能在当前运行配置和配置文件之间进行。在命令执行中，先指定的称为源配置，后指定的称为目标配置。如果源配置和目标配置之前存在差异，则显示结果如下： ● 目标配置相对于源配置添加的命令，使用前缀“+”进行标识。 ● 目标配置相对于源配置删除的命令，使用前缀“-”进行标识。 ● 目标配置修改了源配置的命令，则先显示一条前缀为“-”的命令，后显示一条前缀为“+”的命令。
比较当前运行配置与指定用户标签配置是否一致。	**display configuration changes** { **running label** *label* | **label** *label* **running** } [**with-tag**]	

操作	命令	说明
比较当前的配置与下次启动的配置文件或者指定的配置文件的内容是否一致。	**compare configuration** [*configuration-file*]	当用户完成一系列操作后，可通过本命令从当前配置的首行开始，比较当前配置与下次启动的配置文件或指定的配置文件内容是否一致，以便用户决定是否需要将当前的配置保存，设置为下次启动的配置文件。 该命令在比较出不同之处时，将从不同处所在的行开始，分别对当前配置和下次启动的配置文件或指定的配置文件，显示9行文件内容，如果该不同之处到文件末尾不足9行，将显示到文件尾为止。

1.1.5.3.11 通过拷贝屏幕备份配置文件

背景信息

为防止设备意外损坏，导致配置文件无法恢复，可以通过拷贝屏幕将配置文件备份到维护终端的硬盘中。

操作步骤

步骤1 直接屏幕拷贝。在命令行界面上，执行以下命令，并拷贝所有显示信息到TXT文本文件中，从而将配置文件备份到维护终端的硬盘中。

```
display current-configuration
```

说明

屏幕上显示的配置信息受终端软件的影响，可能会出现某配置过长而换行的情况。对于换行的配置，拷贝至TXT文本中时，需要删除换行，保证一条配置信息在只处在一行中。否则当使用制作的TXT文本恢复配置时，换行的配置可能无法恢复。

----结束

1.1.5.3.12 备份配置文件到存储器

背景信息

为防止设备意外损坏，导致配置文件无法恢复，可以备份配置文件到存储器。

操作步骤

步骤1 （可选）保存配置文件。

```
save configuration-file
```

步骤2 备份配置文件到存储器中。

copy *source-filename destination-filename*

----结束

1.1.5.3.13 备份配置文件到 FTP 服务器或客户端

前提条件

在备份配置文件到FTP服务器或客户端之前，需要完成以下任务：

- 如果设备作为FTP客户端，备份配置文件到FTP服务器，需要保证设备和FTP服务器之间的FTP连接已建立，具体配置请参见《配置指南-基础配置》中的“配置设备作为FTP客户端”。
- 如果设备作为FTP服务器，备份配置文件到FTP客户端，需要保证设备和FTP客户端之间的FTP连接已建立，具体配置请参见《配置指南-基础配置》中的“配置设备作为FTP服务器”。
- 已执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

背景信息

为防止设备意外损坏，导致配置文件无法恢复，可以通过FTP方式备份配置文件。

通过FTP方式备份配置文件有两种方式：

- 设备作为FTP客户端，备份配置文件到FTP服务器。
- 设备作为FTP服务器，备份配置文件到FTP客户端。

用户可以根据实际情况选择一种方式进行备份。

说明

FTP备份配置文件比较简单，但是存在安全风险。在安全要求比较高的场景中，建议使用SFTP和SCP备份配置文件。

FIPS模式下，不支持使用FTP备份配置文件。

操作步骤

- 设备作为FTP客户端，备份配置文件到FTP服务器。
  a. 通过FTP与FTP服务器建立连接。

     ftp [**ipv6**] *host-ip*

  b. 传输配置文件。

     在设备上，使用**put**命令将配置文件上传至PC指定目录中并保存。

     put *local-filename* [*remote-filename*]

- 设备作为FTP服务器，备份配置文件到FTP客户端。
  a. 从维护终端向设备发起FTP连接。

     在PC上，通过FTP客户端与设备建立FTP连接（例如设备的IP地址是10.110.24.254，设备上创建的FTP用户的用户名为**huawei**，密码为**YsHsjx_202206**）。

     ```
     C:\Documents and Setting\Administrator> ftp 10.110.24.254
     Connected to 10.110.24.254.
     ```

```
220 FTP service ready.
User (10.110.24.254:(none)): huawei
331 Password required for huawei.
Password:
230 User logged in.
```

b. 传输配置文件。

在PC上，使用**get**命令将配置文件备份至PC指定目录中并保存。

```
ftp> get remote-filename [ local-filename ]
```

----结束

检查配置结果

在终端FTP用户的工作路径下，可以看到配置文件已保存至此路径，并且设备上和FTP服务器或客户端上的配置文件大小一致。

1.1.5.3.14 备份配置文件到 TFTP 服务器

前提条件

在备份配置文件到TFTP服务器之前，需要完成以下任务：

- 在备份配置文件到TFTP服务器之前，需要保证设备和TFTP服务器之间的TFTP连接已建立，具体配置请参见《配置指南-基础配置》中的“配置设备作为TFTP客户端”。
- 已执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

背景信息

为防止设备意外损坏，导致配置文件无法恢复，可以通过TFTP方式备份配置文件。

说明

TFTP备份配置文件比较简单，但是存在安全风险。在安全要求比较高的场景中，建议使用SFTP和SCP备份配置文件。

FIPS模式下，不支持使用TFTP备份配置文件。

操作步骤

步骤1 备份配置文件到TFTP服务器。

```
tftp [ ipv6 ] hostname-ip put sourcefilename [ destination-filename ]
```

----结束

检查配置结果

在终端TFTP用户的工作路径下，可以看到配置文件已保存至此路径，并且设备上和TFTP服务器上的配置文件大小一致。

1.1.5.3.15 备份配置文件到 SFTP 服务器或客户端

前提条件

在备份配置文件到SFTP服务器或客户端之前，需要完成以下任务：

- 如果设备作为SFTP客户端，备份配置文件到SFTP服务器，需要保证设备和SFTP服务器之间的SFTP连接已建立，具体配置请参见《配置指南-基础配置》中的“配置设备作为SFTP客户端”。
- 如果设备作为SFTP服务器，备份配置文件到SFTP客户端，需要保证设备和SFTP客户端之间的SFTP连接已建立，具体配置请参见《配置指南-基础配置》中的“配置设备作为SFTP服务器”。

背景信息

为防止设备意外损坏，导致配置文件无法恢复，可以通过SFTP方式备份配置文件。

通过SFTP方式备份配置文件有两种方式：

- 设备作为SFTP客户端，备份配置文件到SFTP服务器。
- 设备作为SFTP服务器，备份配置文件到SFTP客户端。

用户可以根据实际情况选择一种方式进行备份。

说明

FTP和TFTP备份配置文件比较简单，但是存在安全风险。在安全要求比较高的场景中，建议使用SFTP和SCP备份配置文件

操作步骤

- 设备作为SFTP客户端，备份配置文件到SFTP服务器。
  a. 进入系统视图。
  ```
  system-view
  ```
  b. 通过SFTP与SFTP服务器建立连接。
  ```
  sftp [ ipv6 ] host-ip
  ```
  c. 传输配置文件。

  在设备上，使用**put**命令将配置文件上传至PC指定目录中并保存。
  ```
  put local-filename [ remote-filename ]
  ```
- 设备作为SFTP服务器，备份配置文件到SFTP客户端。
  a. 从维护终端向设备发起SFTP连接。

  在PC上，通过SFTP客户端与设备建立SFTP连接（以下显示信息仅为示意）。
  ```
  C:/Documents and Settings/Administrator> sftp client001@10.136.23.4
  Connecting to 10.136.23.4...
  The authenticity of host "10.136.23.4 (10.136.23.4)" can't be established.
  DSA key fingerprint is 0d:48:82:fd:2f:52:1c:f0:c4:22:70:80:8f:7b:fd:78.
  Are you sure you want to continue connecting (yes/no)? yes
  Warning: Permanently added "10.136.23.4" (DSA) to the list of known hosts.
  client001@10.136.23.4's password:
  sftp>
  ```
  b. 传输配置文件。

  在PC上，使用**get**命令将配置文件备份至PC指定目录中并保存。

```
sftp> get remote-filename [ local-filename ]
```

----结束

检查配置结果

在终端SFTP用户的工作路径下，可以看到配置文件已保存至此路径，并且设备上和SFTP服务器或客户端上的配置文件大小一致。

1.1.5.3.16 备份配置文件到 SCP 服务器或客户端

前提条件

在备份配置文件到SCP服务器或客户端之前，需要完成以下任务：

- 如果设备作为SCP客户端，备份配置文件到SCP服务器，需要保证设备和SCP服务器之间的连接已建立，具体配置请参见《配置指南-基础配置》中的“配置设备作为SCP客户端”。
- 如果设备作为SCP服务器，备份配置文件到SCP客户端，需要保证设备和SCP客户端之间的连接已建立，具体配置请参见《配置指南-基础配置》中的“配置设备作为SCP服务器”。

背景信息

为防止设备意外损坏，导致配置文件无法恢复，可以通过SCP方式备份配置文件。

通过SCP方式备份配置文件有两种方式：

- 设备作为SCP客户端，备份配置文件到SCP服务器。
- 设备作为SCP服务器，备份配置文件到SCP客户端。

用户可以根据实际情况选择一种方式进行备份。

操作步骤

- 设备作为SCP客户端，备份配置文件到SCP服务器。
  a. 进入系统视图。

  ```
  system-view
  ```

  b. 传输配置文件。

  在设备上，执行以下命令将配置文件上传至PC指定目录中并保存。

  ```
  scp source-filename destination-filename
  ```

  例如：将配置文件vrpcfg.cfg通过SCP方式备份到SCP服务器，服务器的IP地址为10.1.1.1。(以下显示信息仅为示意)

  ```
  <HUAWEI> system-view
  [~HUAWEI] scp vrpcfg.cfg scpuser@10.1.1.1:cfcard:/vrpcfg-backup.cfg
  Trying 10.1.1.1...
  Press CTRL+K to abort
  Connected to 10.1.1.1...
  The server is not authenticated. Continue to access it? [Y/N]:y
  Save the server's public key? [Y/N]:y
  The server's public key will be saved with the name 10.1.1.1. Please wait...

  Please select public key type for user authentication [R for RSA/D for DSA/E for ECC] Please select [R/D/E]:e
  ```

```
Enter password:
vrpcfg.cfg                    100%        261Bytes        1Kb/s
```

- 设备作为SCP服务器，备份配置文件到SCP客户端。

  在Windows的命令行提示符界面上，执行以下命令将配置文件备份至PC指定目录中并保存。

  scp *source-filename destination-filename*

  例如：将配置文件vrpcfg.cfg通过SCP方式备份到SCP客户端，设备的IP地址为10.2.2.2。(以下显示信息仅为示意)

```
C:\Documents and Settings\Administrator> scp scpuser@10.2.2.2:cfcard:/vrpcfg.cfg vrpcfg-
backup.cfg
The authenticity of host '10.2.2.2 (10.2.2.2)' can't be established.
DSA key fingerprint is 46:b2:8a:52:88:42:41:d4:af:8f:4a:41:d9:b8:4f:ee.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.2.2.2' (DSA) to the list of known hosts.
scpuser@10.2.2.2's password:
vrpcfg.cfg                                100% 1257     1.2KB/s   00:00
Read from remote host 10.2.2.2: Connection reset by peer

C:\Documents and Settings\Administrator>
```

----结束

检查配置结果

在终端SCP用户的工作路径下，可以看到配置文件已保存至此路径，并且设备上和SCP服务器或客户端上的配置文件大小一致。

1.1.5.3.17 从存储器恢复配置文件

背景信息

如果用户进行了错误的配置，导致功能异常，可以将存储在设备存储器中的备份配置文件恢复成启动配置文件。

操作步骤

步骤1 拷贝备份的配置文件并指定配置文件名。

copy *source-filename destination-filename* [**all**]

步骤2 指定重新启动使用的配置文件。

startup saved-configuration *configuration-file*

步骤3 重新启动设备，使配置文件生效。

reboot fast

----结束

1.1.5.3.18 从 FTP 服务器或客户端复制配置文件到设备

前提条件

在从FTP服务器或客户端复制配置文件到设备之前，需要完成以下任务：

- 如果设备作为FTP客户端，从FTP服务器复制配置文件到设备，需要保证设备和FTP服务器之间的FTP连接已建立，具体配置请参见《 配置指南-基础配置 》中的“配置设备作为FTP客户端”。
- 如果设备作为FTP服务器，从FTP客户端复制配置文件到设备，需要保证设备和FTP客户端之间的FTP连接已建立，具体配置请参见《 配置指南-基础配置 》中的“配置设备作为FTP服务器”。
- 已执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

背景信息

如果用户进行了错误的配置，导致功能异常，可以通过FTP的方式从FTP服务器或客户端复制配置文件到设备，从而恢复配置文件。

通过FTP方式复制配置文件到设备有两种方式：

- 设备作为FTP客户端，从FTP服务器复制配置文件到设备。
- 设备作为FTP服务器，从FTP客户端复制配置文件到设备。

用户可以根据实际情况选择一种方式进行备份。

说明

FTP方式恢复配置文件比较简单，但是存在安全风险。在安全要求比较高的场景中，建议使用SFTP和SCP恢复配置文件

FIPS模式下，不支持FTP方式恢复配置文件。

操作步骤

- 设备作为FTP客户端，从FTP服务器复制配置文件到设备。
  a. 通过FTP与FTP服务器建立连接。

     ftp [**ipv6**] *host-ip*

  b. 传输配置文件。

     在设备上，使用命令**get**从PC将配置文件复制到设备的指定目录中并保存。

     get *remote-filename* [*local-filename*]

- 设备作为FTP服务器，从FTP客户端复制配置文件到设备。
  a. 从维护终端向设备发起FTP连接。

     在PC上，通过FTP客户端与设备建立FTP连接（例如设备的IP地址是10.110.24.254，设备上创建的FTP用户的用户名为**huawei**，密码为**YsHsjx_202206**）。

     ```
     C:\Documents and Setting\Administrator> ftp 10.110.24.254
     Connected to 10.110.24.254.
     220 FTP service ready.
     User (10.110.24.254:(none)): huawei
     331 Password required for huawei.
     Password:
     230 User logged in.
     ```

  b. 传输配置文件。

     在PC上，使用**put**命令从PC将配置文件复制到设备的指定目录中并保存。

     ftp> **put** *local-filename* [*remote-filename*]

----结束

检查配置结果

在设备中执行**dir**命令，查看配置文件是否从FTP服务器或客户端复制到设备。

1.1.5.3.19 从 TFTP 服务器复制配置文件到设备

前提条件

在从TFTP服务器复制配置文件到设备之前，需要完成以下任务：

- 从TFTP服务端复制配置文件到设备之前，需要保证设备和TFTP服务器之间的TFTP连接已建立，具体配置请参见《 配置指南-基础配置 》中的“配置设备作为TFTP客户端”
- 已执行命令**install feature-software WEAKEA**安装弱安全协议特性包（WEAKEA）。

背景信息

如果用户进行了错误的配置，导致功能异常，可以通过TFTP的方式从TFTP服务器复制配置文件到设备，从而恢复配置文件。

说明

TFTP方式恢复配置文件比较简单，但是存在安全风险。在安全要求比较高的场景中，建议使用SFTP和SCP恢复配置文件

FIPS模式下，不支持TFTP方式恢复配置文件。

操作步骤

步骤1 从TFTP服务端复制配置文件到设备。

tftp [**ipv6**] *hostname-ip* **get** *source-filename* [*destination-filename*]

----结束

检查配置结果

在设备中执行**dir**命令，查看配置文件是否从TFTP服务器复制到设备。

1.1.5.3.20 从 SFTP 服务器或客户端复制配置文件到设备

前提条件

在从SFTP服务器或客户端复制配置文件到设备之前，需要完成以下任务：

- 如果设备作为SFTP客户端，从SFTP服务器复制配置文件到设备，需要保证设备和SFTP服务器之间的SFTP连接已建立，具体配置请参见《 配置指南-基础配置 》中的“配置设备作为SFTP客户端”。
- 如果设备作为SFTP服务器，从SFTP客户端复制配置文件到设备，需要保证设备和SFTP客户端之间的SFTP连接已建立，具体配置请参见《 配置指南-基础配置 》中的“配置设备作为SFTP服务器”。

背景信息

如果用户进行了错误的配置，导致功能异常，可以通过SFTP的方式从SFTP服务器或客户端复制配置文件到设备，从而恢复配置文件。

通过SFTP方式复制配置文件到设备有两种方式：

- 设备作为SFTP客户端，从SFTP服务器复制配置文件到设备。
- 设备作为SFTP服务器，从SFTP客户端复制配置文件到设备。

用户可以根据实际情况选择一种方式进行备份。

操作步骤

- 设备作为SFTP客户端，从SFTP服务器复制配置文件到设备。
  a. 进入系统视图。
  ```
  system-view
  ```
  b. 通过SFTP与SFTP服务器建立连接。
  ```
  sftp [ ipv6 ] host-ip
  ```
  c. 传输配置文件。

  在设备上，使用**get**命令从PC将配置文件复制到设备的指定目录中并保存。
  ```
  get remote-filename [ local-filename ]
  ```
- 设备作为SFTP服务器，从SFTP服务器复制配置文件到设备。
  a. 从维护终端向设备发起SFTP连接。

  在PC上，通过SFTP客户端与设备建立SFTP连接（以下显示信息仅为示意）。
  ```
  C:/Documents and Settings/Administrator> sftp client001@10.136.23.4
  Connecting to 10.136.23.4...
  The authenticity of host "10.136.23.4 (10.136.23.4)" can't be established.
  DSA key fingerprint is 0d:48:82:fd:2f:52:1c:f0:c4:22:70:80:8f:7b:fd:78.
  Are you sure you want to continue connecting (yes/no)? yes
  Warning: Permanently added "10.136.23.4" (DSA) to the list of known hosts.
  client001@10.136.23.4's password:
  sftp>
  ```
  b. 传输配置文件。

  在PC上，使用**put**命令从PC将配置文件复制到设备的指定目录中并保存。
  ```
  sftp> put local-filename [ remote-filename ]
  ```

----结束

检查配置结果

在设备中执行**dir**命令，查看配置文件是否从SFTP服务器或客户端复制到设备。

1.1.5.3.21 从 SCP 服务器或客户端复制配置文件到设备

前提条件

在从SCP服务器或客户端复制配置文件到设备之前，需要完成以下任务：

- 如果设备作为SCP客户端，从SCP服务器复制配置文件到设备，需要保证设备和SCP服务器之间的SCP连接已建立，具体配置请参见《 配置指南-基础配置 》中的“配置设备作为SCP客户端”。

- 如果设备作为SCP服务器，从SCP客户端复制配置文件到设备，需要保证设备和SCP客户端之间的SCP连接已建立，具体配置请参见《 配置指南-基础配置 》中的“配置设备作为SCP服务器”。

背景信息

如果用户进行了错误的配置，导致功能异常，可以通过SCP的方式从SCP服务器或客户端复制配置文件到设备，从而恢复配置文件。

通过SCP方式复制配置文件到设备有两种方式：

- 设备作为SCP客户端，从SCP服务器复制配置文件到设备。
- 设备作为SCP服务器，从SCP客户端复制配置文件到设备。

用户可以根据实际情况选择一种方式进行备份。

操作步骤

- 设备作为SCP客户端，从SCP服务器复制配置文件到设备。
  a. 进入系统视图。

     system-view

  b. 传输配置文件。

     在设备上，执行以下命令从PC将配置文件复制到设备的指定目录中并保存。

     scp *source-filename destination-filename*

     例如：从SCP服务器将配置文件vrpcfg.cfg通过SCP方式复制到设备，服务器的IP地址为10.1.1.1。(以下显示信息仅为示意)

     ```
     <HUAWEI> system-view
     [~HUAWEI] scp scpuser@10.1.1.1:cfcard:/vrpcfg.cfg vrpcfg-backup.cfg
     Trying 10.1.1.1...
     Press CTRL+K to abort
     Connected to 10.1.1.1...
     The server is not authenticated. Continue to access it? [Y/N]:y
     Save the server's public key? [Y/N]:y
     The server's public key will be saved with the name 10.1.1.1. Please wait...

     Please select public key type for user authentication [R for RSA/D for DSA/E for ECC] Please select [R/D/E]:e
     Enter password:
     vrpcfg.cfg                    100%        261Bytes          1Kb/s
     ```

- 设备作为SCP服务器，从SCP客户端复制配置文件到设备。

  在操作系统的命令行提示符界面上，执行以下命令将配置文件复制至PC指定目录中并保存。

  scp *source-filename destination-filename*

  例如：从SCP客户端将配置文件vrpcfg.cfg通过SCP方式复制到设备，设备的IP地址为10.2.2.2。(以下显示信息仅为示意)

  ```
  C:\Documents and Settings\Administrator> scp vrpcfg.cfg scpuser@10.2.2.2:cfcard:/vrpcfg-backup.cfg
  The authenticity of host '10.2.2.2 (10.2.2.2)' can't be established.
  DSA key fingerprint is 46:b2:8a:52:88:42:41:d4:af:8f:4a:41:d9:b8:4f:ee.
  Are you sure you want to continue connecting (yes/no)? yes
  Warning: Permanently added '10.2.2.2' (DSA) to the list of known hosts.
  scpuser@10.2.2.2's password:
  vrpcfg.cfg                              100% 1257     1.2KB/s   00:00
  Read from remote host 10.2.2.2: Connection reset by peer
  ```

```
C:\Documents and Settings\Administrator>
```

----结束

检查配置结果

在设备中执行**dir**命令，查看配置文件是否从SCP服务器或客户端复制到设备。

1.1.5.3.22 压缩配置文件

背景信息

如果配置文件过大，为了方便存储，可以将配置文件进行压缩。对于存储在设备中的压缩后的配置文件，也可以进行解压缩。

操作步骤

- 压缩配置文件。

  zip *source-filename destination-filename*
- 解压缩配置文件。

  unzip *source-filename destination-filename*

----结束

1.1.5.3.23 清除配置

背景信息

在以下情况下需要清除配置文件：

- 设备软件升级之后，原配置文件与当前软件不匹配。
- 配置文件遭到破坏，或加载了错误的配置文件。

须知

reset saved-configuration命令会清空设备下次启动使用的配置文件信息，请慎重执行该命令，建议在技术支持人员指导下使用。

当用户需要将设备上的某个接口用作其他用途时，原始的配置需要逐条删除。如果该接口下存在大量的配置，那么用户将耗费大量的时间进行删除动作，增大了用户的维护量。为了减少用户的维护量和降低操作的复杂度，可以一键式清除接口下的配置。

操作步骤

- 清空设备下次启动使用的配置。
  a. 取消指定系统下次启动时使用的配置文件，从而使设备配置恢复到缺省值。

     reset saved-configuration

说明

取消指定系统下次启动时使用的配置文件后，如果不使用**startup saved-configuration**命令重新指定含有正确配置信息的配置文件，或者保存配置到下次启动的配置文件，则设备下次启动时，将会以缺省配置启动。

系统在清除设备配置文件前会比较当前启动与下次启动的配置文件：

- 如果一致，执行该命令将同时清除这两个配置文件的设置。系统下次启动采用缺省配置文件。
- 如果不一致，执行该命令将清除下次启动的配置文件的设置，保留当前启动配置文件的设置。
- 如果当前启动配置文件为空，下次启动配置文件不为空，执行该命令后，则正常清除下次启动配置文件的设置。
- 如果下次启动配置文件为空，当前启动配置文件不为空，执行该命令后，系统将提示错误，并且不做任何清除操作。如果执行该命令后重启设备，则管理口的地址就没有了，只能通过串口登录设备后重新配置。

b. 重新启动设备，使配置生效。

```
reboot fast
```

- 一键式清除指定接口下配置信息或将配置恢复到缺省值。

表 1-81 一键式清除指定接口下配置信息

视图	命令	说明
系统视图	**clear configuration interface** *interface-type interface-number*	请记住需要清除的接口类型和编号，否则可能会导致其他接口配置被清除，从而导致业务中断
用户视图	**clear configuration this**	用户可直接在当前接口视图下执行该操作，简化用户操作。

说明

该操作会清除指定接口下的配置信息，请慎重执行清除命令。

- 清除预置的默认配置文件。

```
reset default-configuration
```

说明

慎重执行该命令，建议在技术支持人员指导下使用。

----结束

1.1.5.3.24 回退配置

背景信息

用户如果发现配置错误、配置产生故障或者此配置对网络产生了超出预期的结果而需要恢复原来的配置时，可以使用此功能。

用户执行命令并回车后，系统就会检测当前配置和历史配置是否产生差异。如果有差异，系统就会提交用户的配置操作。

在使用配置回退功能之前，设备上通过以下方式产生配置回退点。

- 用户执行commit命令提交配置后，会自动生成配置回退点。自动生成配置回退点功能默认使能，如需关闭此功能，请在系统视图下执行命令**configuration checkpoint auto-save disable**，执行此命令行后，再执行commit命令时将不会自动生成配置回退点
- 系统默认每天2:00定时自动生成配置回退点，如需修改自动生成回退点的时间，请在系统视图下执行命令**set save-configuration checkpoint daily time** *time* 。
- 用户执行一系列配置命令后，必须输入**commit** [**trial** [*time*] [**label** *label*]] [**persist** *persistId*] [**description** *description*]或**commit label** *label* [**description** *description*]命令，系统才会检查当前配置和历史配置的差异，并且生成配置回退点。如果用户希望能够快速找到要回退的配置点，可以使用参数**description** *description*为此次配置添加简单的描述，便于以后查看。推荐使用此模式进行配置编辑和提交。

操作步骤

步骤1 查看配置回退点的信息以及最近配置的配置变更。

具体操作请参见**表1-82**。

表 1-82 查看配置回退点的信息

操作	命令	说明
查看所有配置回退点列表。	**display configuration commit list** [**verbose**] [*number-of-commits* | **label**]	用户可以使用该命令确认配置回退点是否生成、以及各回退点详细描述。 如果希望查看最近一个或者多个配置回退点的相关信息，可以使用*number-of-commits*参数指定个数。

操作	命令	说明
查看指定配置回退点的配置变更。	**display configuration commit changes** [**at** *commit-id* | **since** *commit-id* | [**last**] *number-of-commits*]	通过查看指定配置回退点的配置变更，可以分析如果执行回退，可能变化的配置有哪些，从而确定是否执行配置回退，以及如果配置回退会对系统产生造成哪些影响： ● 如果希望查看所有配置回退点的配置变更，在命令中不指定任何参数。 ● 如果希望查看某个配置回退点的配置变更，可以使用**at** *commit-id*参数指定配置回退点。 ● 如果希望查看从某个配置回退点开始到当前状态的配置变更，可以使用**since** *commit-id*参数指定配置回退点。 ● 如果希望查看最近一次或者多次配置变更，可以使用[**last**] *number-of-commits*参数指定次数。如果指定参数**last**，查看的是变更点的最终差异，不指定的则查看的是每个变更点的配置差异。

步骤2 根据用户所需的历史配置状态进行配置回退。

- 在用户视图下进行配置回退。
  rollback configuration { **to** { **commit-id** *commit-id* | **label** *label* | **file** *file-name* } | **last** *number-of-commits* }

  配置的回退包括：创建的配置会被删除、删除的配置会被重新创建、修改的配置会被改回原值。
  - 如果希望系统回退到某个配置回退点生成时的历史配置状态，可以使用**commit-id** *commit-id*参数指定配置回退点。
  - 如果希望系统回退到指定用户标签点生成时的历史配置状态，可以使用**label** *label*参数指定配置回退用户标签。
  - 如果希望系统回退到指定配置文件生成时的历史配置状态，可以使用**file** *file-name*参数指定配置回退配置文件。
  - 如果希望系统回退到最近一个或者多个配置回退点之前的历史配置状态，可以使用**last** *number-of-commits*
- 在系统视图下进行配置回退。
  system-view
  load configuration rollback changes { **at commit-id** *commit-id* | **to commit-id** *commit-id* | **last**


```
number-of-commits | to label user-label }
quit
commit
```

步骤3 （可选）设置配置回退点用户标签。

```
set configuration commit commit-id label label-string
```

缺省情况下，配置回退点没有配置用户标签。

步骤4 （可选）删除系统中指定配置回退点的标签或者最早生成的配置回退点列表信息。

```
clear configuration commit { commit-id label | oldest number-of-commits }
```

步骤5 （可选）删除指定用户标签的配置回退点。

```
clear configuration commit label label-name
```

----结束

任务示例

用户登录设备，发现配置错误，将配置回退到备份的配置文件中的配置。

1. 查看当前设备上备份的配置文件名。

```
<HUAWEI> dir
Directory of cfcard:/

 Idx  Attr     Size(Byte)  Date        Time        FileName
   0  -rw-           889  Mar 01 2019 14:41:56    private-data.txt
   1  -rw-         6,311  Feb 17 2019 14:05:04    backup.cfg
   2  -rw-         2,393  Mar 06 2019 17:20:10    vrpcfg.zip
   3  -rw-           812  Nov 12 2019 15:43:10    hostkey
   4  drw-             -  Mar 01 2019 14:41:46    compatible
   5  -rw-           540  Nov 12 2019 15:43:12    serverkey...
670,092 KB total (569,904 KB free)
```

2. 回退配置到备份的配置文件。

```
<HUAWEI> rollback configuration to file backup.cfg
Warning: This operation will revert configuration changes to the file backup.cfg.
Continue? [Y/N]:y
```

3. 检查配置结果。

   执行命令**display current-configuration**，查看之前的错误配置是否还存在。

检查配置结果

- 执行命令**display configuration rollback result**，查看最近一次配置回退的失败信息和配置命令在回退过程中的提示信息。
- 执行命令**display configuration commit at** *commit-id*，查看指定配置回退点的全量配置信息。
- 执行命令**display configuration rollback changes load-result**，查看加载指定配置回退点或者用户标签所在配置时的具体失败命令和原因。

1.1.5.3.25 差异配置粘贴

背景信息

配置相同的多台设备，如果其中一台设备的配置发生变更，为保持配置一致，其他设备的配置需同步变更，此时用户可使用此功能查询出有差异的配置，将差异配置粘贴至其他设备，使多台设备配置保持同步。

说明

此功能只能在两阶段生效模式下执行。

操作步骤

步骤1 查看配置差异。

- 查看当前运行配置和指定配置文件中携带标签的差异信息。

  表 1-83 查看当前运行配置和指定配置文件中携带标签的差异信息

操作	命令	说明
比较当前运行配置与指定的配置文件的内容是否一致。	**display configuration changes running file** *file-name* | **file** *file-name* **running with-tag**	配置差异比较只能在当前运行配置和配置文件之间进行。在命令执行中，先指定的称为源配置，后指定的称为目标配置。如果源配置和目标配置之前存在差异，则显示结果如下： – 目标配置相对于源配置添加的命令，使用前缀“+”进行标识。 – 目标配置相对于源配置删除的命令，使用前缀“-”进行标识。 – 目标配置修改了源配置的命令，则先显示一条前缀为“-”的命令，后显示一条前缀为“+”的命令。 **说明** 参数**running**表示当前运行的配置，如果先指定**running**，那么当前运行的配置为源配置。
比较当前运行配置与指定用户标签配置是否一致。	**display configuration changes running label** *label* | **label** *label* **running with-tag**	

- 查看配置回退点记录的携带标签的配置变更信息。
  display configuration commit changes [{ **at** *commit-id* | **since** *commit-id* | **last** *number-of-commits* }] **with-tag**

步骤2 复制差异配置，粘贴至需要同步差异的目标设备，替换当前的运行配置。

1. 进入系统视图。
   system-view
2. 进入差异配置粘贴视图。
   load configuration terminal
3. 将查询的差异配置直接复制粘贴到目标设备。
4. 粘贴结束符：**end-diff**或者**abort**。

- **end-diff**：结束差异配置粘贴，并退出差异配置粘贴视图。
- **abort**：取消差异配置粘贴，并退出差异配置粘贴视图。

说明

退出差异配置粘贴视图后，可执行命令**display configuration candidate**，查看替换后的配置是否符合预期。

步骤3 提交配置。

```
commit
```

----结束

1.1.5.3.26 使用配置模板下发配置

背景信息

设备不同的视图下的配置，可能会出现大量的重复情况。此时，可以通过创建一个配置模板，将重复的配置添加到配置模板中，相应视图下应用该配置模板即可，使配置看起来更简洁。

操作步骤

步骤1 进入系统视图。

```
system-view
```

步骤2 创建配置模板，并进入配置模板视图。

```
command group group-name
```

步骤3 进入需要配置的业务视图实例中，然后配置需要下发的数据。

业务视图实例支持正则表达式，将所有满足正则表达式的业务视图实例都与当前配置模板相关联。例如执行命令**interface <Loopback.>**，将设备上所有的Loopback接口与配置模板关联。正则表达式的相关介绍请参见《配置指南-熟悉命令行配置》中的“过滤命令行显示信息”。

步骤4 退出至配置模板视图。

```
quit
```

步骤5 （可选）查看配置模板中的配置信息。

```
display this command group candidate merge
```

步骤6 （可选）查看配置模板中发生变更的配置信息。

```
display this command group candidate
```

步骤7 提交配置并退出至系统视图。

```
end-group
```

如果不需要使用当前的配置模板，可以执行命令**abort**，放弃当前正在配置的模板，并退出配置模板视图。

步骤8 创建业务视图实例。

步骤9 应用配置模板。

```
apply-command-group group { group-name } & < 1-8 >
```

根据需要在不同的视图下执行该步骤：

系统视图下执行此命令，设备会将配置模板中的配置下发至所有匹配到的业务视图实例中。

指定业务视图实例下执行此命令，配置将下发至指定业务视图实例中。

----结束

任务示例

1. 创建配置模板并进入配置模板视图。
```
<HUAWEI> system-view
[~HUAWEI] command group g1
```
2. 进入Loopback接口视图实例，并下发配置。该操作将所有的Loopback接口与当前模板关联。
```
[~HUAWEI-group-g1] interface <Loopback.>
[~HUAWEI-group-g1-<Loopback.>] ipv6  enable
[~HUAWEI-group-g1-<Loopback.>] quit
[~HUAWEI-group-g1] display this command group candidate merge
command group g1
 interface <Loopback.>
  ipv6 enable
[~HUAWEI-group-g1] end-group
Warning: The configuration in the view of the current group will be committed. Are you sure you want to continue? [Y/N]:y
```
3. 创建接口Loopback 1和Loopback 2。
```
[~HUAWEI] interface loopback 1
[*HUAWEI-LoopBack1] quit
[*HUAWEI] interface loopback 2
[*HUAWEI-LoopBack2] quit
[*HUAWEI] commit
```
4. 在接口视图下应用配置模板g1。
```
[~HUAWEI] apply-command-group g1
Warning: The template configurations will be applied to the matched view. Are you sure you want to continue? [Y/N]:y
```
5. 显示视图下用户配置模板信息，不显示模板注释信息。
```
[~HUAWEI] interface loopback 1
[~HUAWEI-LoopBack1] display this inheritance no-comment
#
interface LoopBack1
 apply-command-group g1
 ipv6 enable
#
return
[~HUAWEI-LoopBack1] interface loopback 2
[~HUAWEI-LoopBack2] display this inheritance no-comment
#
interface LoopBack2
 apply-command-group g1
 ipv6 enable
#
return
```

检查配置结果

- 执行命令**display current-configuration** [**inheritance** [**no-comment**]]，查看从模板继承的配置信息。
- 执行命令**display this inheritance** [**no-comment**]，查看系统当前视图从模板继承的配置信息。

1.1.5.3.27 举例：配置下次启动时加载的配置文件

组网需求

如图1-25所示，设备当前系统软件版本已经不能满足用户需求，用户需要部署更多的特性。此时需要远程为该设备进行系统软件升级。

图 1-25 配置下次启动时加载的配置文件组网图

配置思路

采用如下的思路配置系统启动以实现系统升级：

1. 将新的系统软件上传至设备根目录。
2. 保存系统当前配置，以使升级后配置仍生效。
3. 配置设备下次启动时加载的系统软件。
4. 配置设备下次启动时加载的配置文件。
5. 重新启动设备实现设备的升级。

操作步骤

步骤1 将新的系统软件上传至设备根目录。

1. 在进行配置前，执行**display startup**命令查看当前设备下次启动文件的配置情况。

```
<HUAWEI> display startup
MainBoard:
  Configured startup system software:       cfcard:/basicsoft.cc
  Startup system software:                  cfcard:/basicsoft.cc
  Next startup system software:             cfcard:/basicsoft.cc
  Startup saved-configuration file:         cfcard:/vrpcfg.zip
  Next startup saved-configuration file:    cfcard:/vrpcfg.zip
  Startup paf file:                         default
  Next startup paf file:                    default
  Startup patch package:                    NULL
  Next startup patch package:               NULL
```

2. 配置设备作为SFTP服务器。

   采用文件传输方式将新的系统软件文件上传至设备。文件传输方式较多，此处将设备配置为SFTP服务器，从客户端获取系统软件文件。上传文件前需确保存储器有足够的空间保存新的系统软件文件，若是空间不足，需要清理存储器。

   # 配置SFTP服务器的IP地址。

```
<HUAWEI> system-view
[~HUAWEI] sysname SSH Server
[*HUAWEI] commit
[~SSH Server] interface gigabitethernet 1/0/1
[~SSH Server-GigabitEthernet1/0/1] undo portswitch
[*SSH Server-GigabitEthernet1/0/1] ip address 10.248.103.194 255.255.255.0
[*SSH Server-GigabitEthernet1/0/1] quit
[*SSH Server] commit
```


配置SSH服务器的公钥算法、加密算法、密钥交换算法列表、HMAC认证算法和最小密钥长度。

```
[~SSH Server] ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
[~SSH Server] ssh server hmac sha2_256 sha2_512
[~SSH Server] ssh server key-exchange dh_group_exchange_sha256 dh_group16_sha512
[~SSH Server] ssh server publickey rsa_sha2_256 rsa_sha2_512
[~SSH Server] ssh server dh-exchange min-len 3072
[*SSH Server] commit
```

在服务器端生成本地密钥对，并使能SFTP服务器功能。

```
[~SSH Server] dsa local-key-pair create
The key name will be:HUAWEI_Host
The range of public key size is (2048, 4096).
NOTE: Key pair generation will take a short while.
Please input the modulus [default = 3072]:3072
[*SSH Server] sftp server enable
[*SSH Server] ssh server-source all-interface
[*SSH Server] commit
```

配置SSH用户，包括认证方式、服务类型、授权目录以及用户名和密码等。

```
[~SSH Server] ssh user client authentication-type password
Info: Succeeded in adding a new SSH user.
[*SSH Server] ssh user client service-type sftp
[*SSH Server] ssh user client sftp-directory cfcard:
[*SSH Server] aaa
[*SSH Server] local-user client password
Please configure the password (8-128)
Enter Password:
Confirm Password:
[*SSH Server-aaa] local-user client level 3
[*SSH Server-aaa] local-user client service-type terminal ssh
[*SSH Server-aaa] quit
[*SSH Server] commit
```

配置SSH服务器的访问权限。

```
[~SSH Server] acl 2000
[*SSH Server-acl4-basic-2000] rule permit source 10.248.103.0 8
[*SSH Server-acl4-basic-2000] quit
[*SSH Server] ssh server acl 2000
[*SSH Server] commit
```

3. 在用户终端PC的命令行提示符中，执行**sftp 10.248.103.194**命令成功与设备建立SFTP连接后，使用**put**命令向设备上传新的系统软件文件**newbasicsoft.cc**。

   上传成功后，在SSH服务器可执行**dir**命令查看上传后的系统软件文件。

```
<SSH Server> dir
Directory of  cfcard:/

 Idx  Attr     Size(Byte)  Date        Time       FileName
   0  drw-              -  Apr 16 2012 13:19:58   logfile
   1  -rw-     85,925,409  Apr 16 2012 13:18:02   basicsoft.cc
   2  -rw-              4  Oct 27 2011 17:25:22   snmpnotilog.txt
   3  -rw-          6,033  Jul 16 2012 16:40:02   private-data.txt
   4  -rw-          3,275  Jul 14 2012 14:18:08   vrpcfg.zip
   5  drw-              -  Nov 14 2011 19:14:26   sysdrv
   6  drw-     88,239,759  Jul 16 2012 19:14:26   newbasicsoft.cc
...

670,092 KB total (569,904 KB free)
```

步骤2 保存系统当前配置。

```
<SSH Server> save
```

系统将提示当前配置将保存至设备，是否继续，输入**y**后，系统提示当前配置已成功保存在设备中。

步骤3 配置设备下次启动时加载的系统软件。

```
<SSH Server> startup system-software newbasicsoft.cc
```


步骤4 配置设备下次启动时加载的配置文件。

```
<SSH Server> startup saved-configuration vrpcfg.zip
```

说明

在步骤1中，通过**display startup**查看下次启动文件的配置情况，可以看到“Next startup saved-configuration file: cfcard:/**vrpcfg.zip**”，说明当前设备已经指定**vrpcfg.zip**作为下次启动时加载的配置文件，所以此步骤可以省略。但如果需要指定其他的配置文件作为下次启动时加载的配置文件时，必须要执行此步骤。

步骤5 检查配置结果。

配置完成之后，执行如下命令，查看设备下次启动时加载的系统软件和配置文件。

```
<SSH Server>  display startup
MainBoard:
  Configured startup system software:        cfcard:/basicsoft.cc
  Startup system software:                  cfcard:/basicsoft.cc
  Next startup system software:              cfcard:/newbasicsoft.cc
  Startup saved-configuration file:          cfcard:/vrpcfg.zip
  Next startup saved-configuration file:     cfcard:/vrpcfg.zip
  Startup paf file:                         default
  Next startup paf file:                     default
  Startup patch package:                     NULL
  Next startup patch package:                NULL
```

步骤6 重新启动设备。

由于已保存过配置文件，所以此时可以执行如下命令行进行快速重新启动。

```
<SSH Server> reboot fast
```

系统会提示即将重新启动，是否继续，输入**y**即可。

----结束

检查配置结果

等候几分钟，设备重启完成，可再次进入系统。执行命令行**display version**，可以看到设备当前的系统软件版本为新的版本，表明升级完成。

配置脚本

```
#
sysname SSH Server
#
acl number 2000
 rule 5 permit source 10.248.103.0 0.0.0.255
#
aaa
 local-user client password irreversible-cipher $1d$+,JS+))\\2$KVNj(.3`_5x0FCKGv}H&.kUTI`Ff&H*eBqO.ua>)$
 local-user client service-type terminal ssh
 local-user client level 3
#
interface GigabitEthernet1/0/1
 ip address 10.248.103.194 255.255.255.0
#
sftp server enable
ssh server-source all-interface
ssh server acl 2000
ssh user client
ssh user client authentication-type password
ssh user client service-type sftp
ssh user client sftp-directory cfcard:
#
ssh server cipher aes128_ctr aes256_ctr aes192_ctr aes128_gcm aes256_gcm
ssh server hmac sha2_256 sha2_512
```



```
ssh server key-exchange dh_group_exchange_sha256 dh_group16_sha512
ssh server publickey rsa_sha2_256 rsa_sha2_512
ssh server dh-exchange min-len 3072
#
return
```

1.1.6 ZTP 配置

1.1.6.1 ZTP 特性描述

1.1.6.1.1 ZTP 介绍

介绍零配置自动部署ZTP（Zero Touch Provisioning）的定义和目的。

定义

ZTP是指空配置设备上电启动时采用的一种自动加载版本文件（包括系统软件、配置文件、补丁文件）的功能。

目的

在部署网络设备时，设备硬件安装完成后，需要管理员到安装现场对设备进行软件调试。当设备数量较多、分布较广时，管理员需要在每一台设备上进行手工配置，既影响了部署的效率，又需要较高的人力成本。

设备运行ZTP功能，可以从文件服务器获取版本文件并自动加载，实现设备的免现场配置、部署，从而降低人力成本，提升部署效率。

受益

实现设备的免现场配置、部署，降低人力成本，提升部署效率。

1.1.6.1.2 原理描述

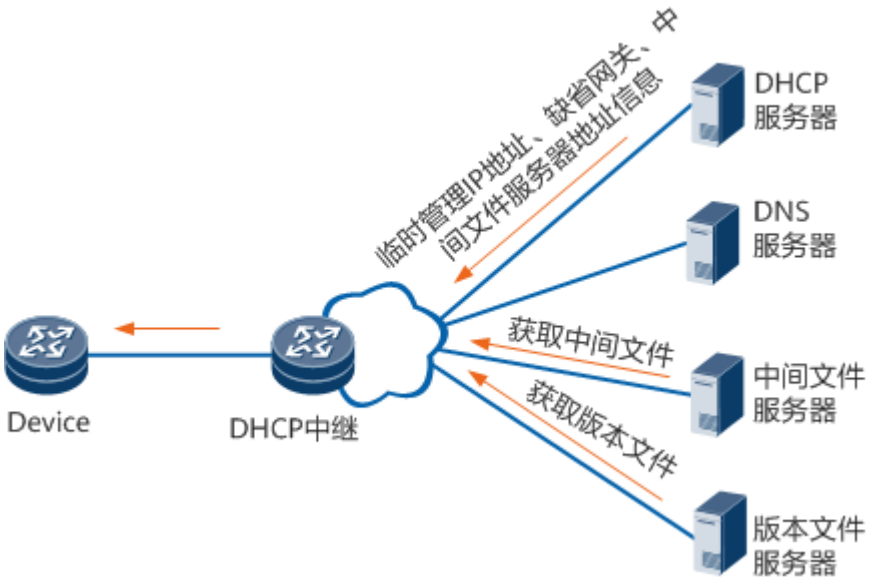
基本原理

自动部署的方式及典型组网环境

空配置的设备在上电后自动进入ZTP流程，通过DHCP方式完成自动部署。ZTP支持通过DHCP申请IPv4和IPv6两种协议地址。

通过DHCP方式实现设备自动部署的典型组网环境如**图1-26**所示。

图 1-26 自动部署的典型组网图



- **DHCP服务器**：用来为执行ZTP的设备分配临时管理IP地址、缺省网关、DNS服务器地址、中间文件服务器地址等信息。
- **DHCP中继**：当执行ZTP的设备与DHCP服务器位于不同网段时，需要通过DHCP中继转发DHCP交互报文。
- **中间文件服务器**：用来保存ZTP过程中设备需要的中间文件，中间文件的类型可以是**ini文件格式**、**cfg文件格式**或 **Python格式的中间文件**。通过解析中间文件，执行ZTP的设备可以获取版本文件服务器地址、版本文件等信息。中间文件服务器可以是TFTP、FTP或SFTP类型的服务器。
- **版本文件服务器**：用来保存设备需要的版本文件，如系统软件、配置文件、补丁文件。版本文件服务器可以和中间文件服务器部署在同一个服务器上，可以是TFTP、FTP或SFTP类型的服务器。
- **DNS服务器**：用来提供域名和IPv4/IPv6地址之间的映射关系。执行ZTP的设备可以通过DNS服务器将IPv4/IPv6文件服务器的域名解析为IPv4/IPv6地址，以便于从IPv4/IPv6文件服务器获取需要的文件。

说明

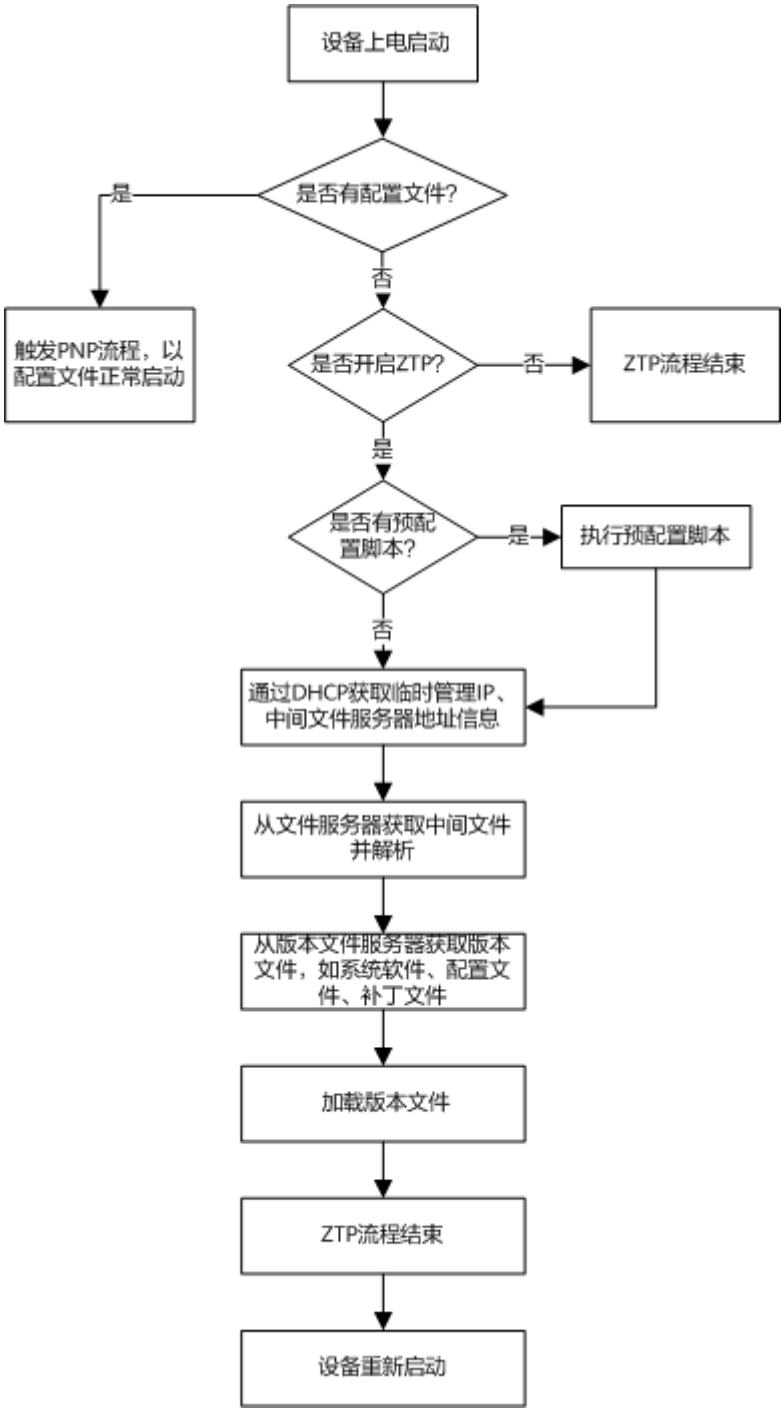
使用TFTP/FTP协议存在安全风险，建议使用SFTP进行文件传输。

如果ZTP通过DHCP申请IPv4/IPv6地址，必须选择对应支持IPv4/IPv6协议的DHCP服务器、DHCP中继、中间文件服务器、版本文件服务器和DNS服务器，且中间文件中的服务器地址填写IPv4/IPv6地址。

ZTP 详细工作流程

ZTP的流程如**图1-27**所示。

图 1-27 ZTP 流程图



ZTP的实现流程分以下几个阶段：

1. 设备上电启动阶段。

   设备上电启动后，如果设备有配置文件，则触发PNP流程，以配置文件正常启动；如果是空配置设备，需要检查设备是否有预配置，如果有，需要执行预配置脚本，否则进入ZTP流程。

2. DHCP信息获取阶段。

设备在管理网口、以太接口上广播发送DHCP请求报文。DHCP服务器向设备发送DHCP应答报文，报文中的Option选项指明设备需要从DHCP服务器获得的信息，包括DHCP服务器的IP地址、缺省网关、中间文件服务器的地址、中间文件名等信息。

说明

DHCP信息获取阶段，如果设备上管理网口和以太接口长时间无法收到DHCP应答报文，可能是因为与本设备连接的上游设备使用了LACP模式的Eth-trunk口并存在VLAN。设备将尝试创建Eth-trunk接口轮询以太接口，并接受VLAN广播报文，来学习VLAN并获取IP地址。

3. 获取中间文件和版本文件阶段。

   设备根据DHCP应答报文中获取的信息，从中间文件服务器上下载中间文件，然后再通过中间文件从版本文件服务器上下载版本文件。

   如果中间文件是 **ini格式的中间文件**或 **cfg格式的中间文件**，设备会根据文件中的版本文件服务器地址、版本文件名信息，从版本文件服务器上下载相应的版本文件；如果是 **Python格式的中间文件**，设备会自动执行该脚本，通过运行脚本从版本文件服务器上下载相应的版本文件。

4. 重新启动阶段。

   设备将下载的版本文件：系统软件、配置文件、补丁文件，设置为下次启动文件，然后重新启动，完成自动部署。其中，配置文件必须存在，否则设备重启后会再次执行ZTP流程。

预配置脚本

由于ZTP在设备空配置时启动，当设备空配置状态接入网络后可能无法满足DHCP组网要求时，预配置脚本可以提供一种在ZTP运行之前对设备进行适配的能力，保证设备和DHCP服务器交互正常。目前，预配置脚本主要在以下场景中使用：

- 用于接口使能自协商功能。
- 用于创建Eth-Trunk。

预配置脚本文件的文件名必须以“.py”作为后缀名，只能包含数字、字母和_的组合，区分大小写，不支持空格，长度范围是1～65，不能以数字开头，不能包含其他特殊字符，命名示例：preconfig.py。请使用Python3.7语法编写或修改脚本文件，详细脚本文件解释请见**预配置脚本文件解释**。

预配置脚本文件示例

说明

该脚本文件仅作为样例，用户可以根据实际应用场景进行修改。

下列文件中的SHA256校验码只是举例，配置时以实际计算的值为准。

注意

预配置脚本中，无论正常运行结束，还是异常运行结束，都要保证 _ops.cli 在结束时被close。

```
#sha256sum="68549835edaa5c5780d7b432485ce0d4fdaf6027a8af24f322a91b9f201a5101"
#!/usr/bin/env python
# coding=utf-8
#
```



```
# Copyright (C) Huawei Technologies Co., Ltd. 2008-2013. All rights reserved.
# ----------------------------------------------------------------------------------------------------------------------
# Project Code    : VRPV8
# File name       : preconfig.py
# ----------------------------------------------------------------------------------------------------------------------
# History:
# Date              Modification
# 20180415          created file.
# ----------------------------------------------------------------------------------------------------------------------

import sys
import http.client
import logging
import logging.handlers
import string
import traceback
import re
import xml.etree.ElementTree as etree
import ops

from time import sleep

# error code
OK          = 0
ERR         = 1
NOT_START_PNP = 2

# User Input: TYPE: list()
ETHTRUNK_MEMBER_LIST = [
    'GigabitEthernet1/0/1',
    'GigabitEthernet1/0/0'
]

# User Input:  TYPE: integer
VLAN = 127


ETHTRUNK_WORK_MODE = 'Static'
MAX_TIMES_CHECK_STARTUPCFG = 36
CHECK_CHECK_STARTUP_CFG_INTERVAL = 5
class OPIExecError(Exception):
    pass

class NoNeedPNP(Exception):
    pass

class OPSConnection(object):
    """Make an OPS connection instance."""

    def __init__(self, host, port=80):
        self.host = host
        self.port = port
        self.headers = {
            "Content-type": "application/xml",
            "Accept":       "application/xml"
            }

        self.conn = http.client.HTTPConnection(self.host, self.port)

    def close(self):
        """Close the connection"""
        self.conn.close()

    def create(self, uri, req_data):
        """Create a resource on the server"""
        ret = self._rest_call("POST", uri, req_data)
        return ret

    def delete(self, uri, req_data):
```

```
        """Delete a resource on the server"""
        ret = self._rest_call("DELETE", uri, req_data)
        return ret

    def get(self, uri, req_data=None):
        """Retrieve a resource from the server"""
        ret = self._rest_call("GET", uri, req_data)
        return ret

    def set(self, uri, req_data):
        """Update a resource on the server"""
        ret = self._rest_call("PUT", uri, req_data)
        return ret

    def _rest_call(self, method, uri, req_data):
        """REST call"""
        if req_data is None:
            body = ""
        else:
            body = req_data

        self.conn.request(method, uri, body, self.headers)
        response = self.conn.getresponse()
        rest_message = convert_byte_to_str(response.read())
        ret = (response.status, response.reason, rest_message)
        if response.status != http.client.OK:
            logging.info(body)
        return ret

def convert_byte_to_str(data):
    result = data
    if not isinstance(data, str):
        result = str(data, "iso-8859-1")
    return result

def get_startup_cfg_info(ops_conn):
    uri = "/cfg/startupInfos/startupInfo"
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
<startupInfo>
<position/>
<configedSysSoft/>
<curSysSoft/>
<nextSysSoft/>
<curStartupFile/>
<nextStartupFile/>
<curPatchFile/>
<nextPatchFile/>
</startupInfo>'''

    config = None
    config1 = None
    ret, _, rsp_data = ops_conn.get(uri, req_data)
    if ret != http.client.OK or rsp_data == '':
        logging.warning('Failed to get the startup information')
        return ERR, config, config1

    root_elem = etree.fromstring(rsp_data)
    namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
    mpath = 'data' + uri.replace('/', '/vrp:')  # match path
    nslen = len(namespaces['vrp'])
    elem = root_elem.find(mpath, namespaces)
    if elem is None:
        logging.error('Failed to get the startup information')
        return ERR, config, config1

    for child in elem:
        tag = child.tag[nslen + 2:]
        if tag == 'curStartupFile' and child.text != 'NULL':
            config = child.text
```



```
        if tag == 'nextStartupFile' and child.text != 'NULL':
            config1 = child.text
        else:
            continue

    return OK, config, config1

def is_need_start_pnp(ops_conn):
    ret, config, _ = get_startup_cfg_info(ops_conn)
    if ret == OK and config is not None and config != "cfcard:/vrpcfg.zip":
        logging.info("No need to run ztp pre-configuration when device starts with configuration file")
        return False
    return True

def check_nextstartup_file(ops_conn):
    cnt = 0
    check_time = MAX_TIMES_CHECK_STARTUPCFG
    while cnt < check_time:
        ret, _, config1 = get_startup_cfg_info(ops_conn)
        if ret == OK and config1 is not None and config1 == "cfcard:/vrpcfg.zip":
            logging.info("check next startup file successful")
            return OK

        sleep(CHECK_CHECK_STARTUP_CFG_INTERVAL)  # sleep to wait for system ready when no query result
        if cnt%6 == 0:
            logging.info("check next startup file...")
        cnt += 1
    return OK

def print_precfg_info(precfg_info):
    """ Print Pre Config Info """

    str_temp = string.Template(
        'Pre-config information:\n'
        '  Eth-Trunk Name:       $ethtrunk_name\n'
        '  Eth-Trunk Work Mode:  $ethtrunk_work_mode\n'
        '  Eth-Trunk MemberIfs:  $ethtrunk_member_ifs\n'
        '  Vlan:                 $vlan_pool\n'
    )

    precfg = str_temp.substitute(ethtrunk_name=precfg_info.get('ethtrunk_ifname'),
                                 ethtrunk_work_mode=precfg_info.get('ethtrunk_work_mode'),
                                 ethtrunk_member_ifs=', '.join(precfg_info.get('ethtrunk_member_ifs')),
                                 vlan_pool=precfg_info.get('vlan'))

    logging.info(precfg)

def get_device_productname(ops_conn):
    """Get system info, returns a dict"""
    logging.info("Get the system information...")
    uri = "/system/systemInfo"
    req_data =  \
'''<?xml version="1.0" encoding="UTF-8"?>
<systemInfo>
    <productName/>
</systemInfo>
'''
    ret, _, rsp_data = ops_conn.get(uri, req_data)
    if ret != http.client.OK or rsp_data == '':
        raise OPIExecError('Failed to get the system information')

    productname = ""
    root_elem = etree.fromstring(rsp_data)
    namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
    uri = uri + '/productName'
    uri = 'data' + uri.replace('/', '/vrp:')
    elem = root_elem.find(uri, namespaces)
    if elem is not None:
        productname = elem.text
```



```
    logging.info('Current product name : {0}'.format(productname))
    return productname

def active_port_license(ops_conn, if_port):
    """ active port-basic license """
    """ NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X GE 10GE """
    """ 50GE """
    productname = get_device_productname(ops_conn)
    active_flag = False

    if 'NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X' in productname:
        # NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X product, all port need active port-basic
        uri = "/devm/portResourceInfos"
        lcsDescription = ['NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X Any 4GE/FE Port RTU',
                          'NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X 4*10GE Port RTU']

        position = re.search('\d+/\\d+/\\d+', if_port)
        position = position.group() if position is not None else None
        if position is not None:
            for info in lcsDescription:
                root_elem = etree.Element('portResourceInfos')
                portResourceInfo_elem = etree.SubElement(root_elem, 'portResourceInfo')
                etree.SubElement(portResourceInfo_elem, 'lcsDescription').text = info
                lcsports_elem = etree.SubElement(portResourceInfo_elem, 'lcsPorts')
                lcsport_elem = etree.SubElement(lcsports_elem, 'lcsPort')
                etree.SubElement(lcsport_elem, 'position').text = position
                etree.SubElement(lcsport_elem, 'isAct').text = 'active'

                try:
                    req_data = etree.tostring(root_elem, 'UTF-8')
                    ret, _, _ = ops_conn.set(uri, req_data)
                    if ret == http.client.OK:
                        active_flag = True
                        break
                except OPIExecError:
                    pass
        else:
            logging.error('parse position failed, product: {0}, interface: {1}'.format(productname, if_port))

    elif if_port.startswith('50GE') and if_port.endswith('1'):
        # 2*50GE, only port 1 need active port-basic
        uri = "/lcs/lcsResUsages"

        position = re.search('\d+/\\d+/\\d+', if_port)
        position = position.group() if position is not None else None
        if position is not None:
            root_elem = etree.Element('lcsResUsages')
            lcsResUsages_elem = etree.SubElement(root_elem, 'lcsResUsage')
            etree.SubElement(lcsResUsages_elem, 'resItemName').text = "LANJ50GEE00"
            lcsPorts_elem = etree.SubElement(lcsResUsages_elem, 'lcsPorts')
            lcsport_elem = etree.SubElement(lcsPorts_elem, 'lcsPort')
            etree.SubElement(lcsport_elem, 'position').text = position
            etree.SubElement(lcsport_elem, 'isAct').text = 'active'

            try:
                req_data = etree.tostring(root_elem, 'UTF-8')
                ret, _, _ = ops_conn.set(uri, req_data)
                if ret == http.client.OK:
                    active_flag = True
            except OPIExecError:
                pass
        else:
            logging.error('parse position failed, product: {0}, interface: {1}'.format(productname, if_port))

    else:
        logging.info('The current device no need active port-basic')
        active_flag = True
```



```
    if active_flag is False:
        logging.info('{0} port-basic license active failed'.format(if_port))

def create_ethtrunk(ops_conn, ifname, work_mode, member_ifs):
    """ create interface eth-trunk """
    logging.info('Create interface {0}, Work-Mode: {1}'.format(ifname, work_mode))

    if ifname in ['', None] or work_mode in ['', None] or not member_ifs:
        logging.error('Create Eth-Trunk Parameters is invalid')
        return

    for iface in member_ifs:
        active_port_license(ops_conn, iface)

    uri = '/ifmtrunk/TrunkIfs/TrunkIf'
    str_temp = string.Template("""
<?xml version="1.0" encoding="UTF-8"?>
<TrunkIf operation="create">
    <ifName>$ifName</ifName>
    <workMode>$workmode</workMode>
    <TrunkMemberIfs>
$ifs
    </TrunkMemberIfs>
</TrunkIf>
""")
    ifs_temp = string.Template("""     <TrunkMemberIf operation="create">
        <memberIfName>$memberifname</memberIfName>
    </TrunkMemberIf>""")
    ifs = []
    for iface in member_ifs:
        ifs.append(ifs_temp.substitute(memberifname=iface))

    ifs = '\n'.join(ifs)
    req_data = str_temp.substitute(ifs=ifs, ifName=ifname, workmode=work_mode)

    ret, _, rsp_data = ops_conn.create(uri, req_data)
    if ret != http.client.OK:
        logging.error(rsp_data)
        raise OPIExecError('Failed to create Eth-Trunk interface')

    logging.info('Success to create Eth-Trunk interface')


def delete_ethtrunk(ops_conn, ifname):
    """ """
    logging.info('Delete interface {0}'.format(ifname))

    uri = '/ifmtrunk/TrunkIfs/TrunkIf'
    str_temp = string.Template("""
<?xml version="1.0" encoding="UTF-8"?>
    <TrunkIf operation="delete">
    <ifName>$ifName</ifName>
    </TrunkIf>
    """)

    req_data = str_temp.substitute(ifName=ifname)

    try:
        ret, _, rsp_data = ops_conn.delete(uri, req_data)
        if ret != http.client.OK:
            logging.error(rsp_data)
            raise OPIExecError('Failed to delete Eth-Trunk interface')
    except Exception as reason:
        logging.error('Error:', reason)
    else:
        logging.info('Successed to delete Eth-Trunk interface')


def config_vlan(ops_conn, vlan):
```

```
    """ Config Vlan Pool to Pnp """

    if vlan == 0:
        logging.info('Current vlan is 0, no need config')
        return

    logging.info('Config Vlan Pool To Pnp')
    uri = '/pnp/vlanNotify'

    str_temp = string.Template("""
<?xml version="1.0" encoding="UTF-8"?>
<vlanNotify>
  <startVlan>$startVlan</startVlan>
  <endVlan>$endVlan</endVlan>
</vlanNotify>
    """)

    req_data = str_temp.substitute(startVlan=vlan, endVlan=vlan)
    ret, _, rsp_data = ops_conn.create(uri, req_data)
    if ret != http.client.OK:
        logging.error(rsp_data)
        raise OPIExecError('Failed to config vlan to Pnp')

    logging.info('Successed to config vlan to Pnp')

def config_interface_nego_auto_and_l2mode(_ops):
    handle, err_desp = _ops.cli.open()
    if err_desp not in ['Success', 'Error: The line has been opened.']:
        raise OPIExecError('Failed to open cli')
    _ops.cli.execute(handle, "sys")
    fd, _, err_desp = _ops.cli.execute(handle,"interface GigabitEthernet0/2/4", None)
    if fd is None or err_desp != 'Success':
        _ops.cli.close(handle)
        raise OPIExecError('Failed to execute interface GigabitEthernet0/2/4')
    _ops.cli.execute(handle,"negotiation auto",None)
    _ops.cli.execute(handle,"portswitch",None)

    fd, _, err_desp = _ops.cli.execute(handle,"interface GigabitEthernet0/2/5", None)
    if fd is None or err_desp != 'Success':
        _ops.cli.close(handle)
        raise OPIExecError('Failed to execute interface GigabitEthernet0/2/5')
    _ops.cli.execute(handle,"negotiation auto", None)

    fd, _, err_desp = _ops.cli.execute(handle,"commit",None)
    if fd is None or err_desp != 'Success':
        _ops.cli.close(handle)
        raise OPIExecError('Failed to execute commit')
    ret = _ops.cli.close(handle)
    logging.info('Successed to config interface nego auto')
    return 0

def main_proc(ops_conn, precfg_info):
    ifname = precfg_info.get('ethtrunk_ifname')
    work_mode = precfg_info.get('ethtrunk_work_mode')
    member_ifs = precfg_info.get('ethtrunk_member_ifs')
    vlan = precfg_info.get('vlan')
    _ops = ops.ops()

    if is_need_start_pnp(ops_conn) is False:
        return NOT_START_PNP

    sleep(15)
    try:
        config_interface_nego_auto_and_l2mode(_ops)
    except OPIExecError as reason:
        logging.error('Error: %s' % reason)
        return ERR

    try:
```

```
        create_ethtrunk(ops_conn, ifname, work_mode, member_ifs)
    except OPIExecError as reason:
        logging.error('Error: %s' % reason)
        return ERR

    try:
        config_vlan(ops_conn, vlan)
    except OPIExecError as reason:
        logging.error('Error: %s' % reason)
        delete_ethtrunk(ops_conn, ifname)
        return ERR

    try:
        check_nextstartup_file(ops_conn)
    except OPIExecError as reason:
        logging.error('Error: %s', reason)

    return OK


def main():
    host = 'localhost'

    try:
        work_mode = ETHTRUNK_WORK_MODE
    except NameError:
        work_mode = 'Static'

    try:
        vlan = VLAN
    except NameError:
        vlan = 0

    try:
        member_list = ETHTRUNK_MEMBER_LIST
    except NameError:
        member_list = []

    precfg_info = {
        'ethtrunk_ifname': 'Eth-Trunk0',
        'ethtrunk_work_mode': work_mode,
        'ethtrunk_member_ifs': member_list,
        'vlan': vlan
    }

    print_precfg_info(precfg_info)

    try:
        ops_conn = OPSConnection(host)
        ret = main_proc(ops_conn, precfg_info)
    except Exception:
        logging.error(traceback.print_exc())
        ret = ERR
    finally:
        ops_conn.close()

    return ret

if __name__ == '__main__':
    """ """
    main()
```

预配置脚本文件解释

说明

加粗的内容表示用户可以修改，请根据实际运行环境进行配置。

⚠ 注意

预配置脚本中，无论正常运行结束，还是异常运行结束，都要保证 _ops.cli 在结束时被close。

- 指定该脚本文件的SHA256校验码。

  ```
  #sha256sum="68549835edaa5c5780d7b432485ce0d4fdaf6027a8af24f322a91b9f201a5101"
  ```

  用户可以通过该SHA256校验码对下载的脚本文件进行完整性检测。

  用户可以使用以下两种方法生成脚本文件的SHA256校验码：

  a. SHA256计算工具（如HashMyFiles）；

  b. 执行Windows自带的**certutil -hashfile** *filename* **SHA256**命令。

  说明

  SHA256校验码是由#sha256sum=”行以下内容计算而来，生成SHA256校验码时，需将示例中第一行删除，第二行提至第一行，计算完成后将新生成SHA256校验码“#sha256sum=”写入文件开头。

  文件支持使用SHA256算法进行完整性校验，该种算法安全性较高。

- 指定设备使用Eth-Trunk成员口。

  ```
  ETHTRUNK_MEMBER_LIST = [
     'GigabitEthernet1/0/1',
     'GigabitEthernet1/0/0'
  ]
  ```

  “**GigabitEthernet1/0/1**”为设备接口名称。

- 指定设备DHCP使用的VLAN值。

  ```
  VLAN = 127
  ```

  此模块不需要用户编辑。

- 指定设备Eth-Trunk工作模式。

  ```
  ETHTRUNK_WORK_MODE = 'Static'
  ```

  此模块不需要用户编辑。

- 定义设备检查启动项设置失败时的最大重试次数。

  ```
  MAX_TIMES_CHECK_STARTUPCFG = 36
  ```

- 定义检查系统软件是否设置成功的间隔。

  ```
  CHECK_CHECK_STARTUP_CFG_INTERVAL = 5
  ```

- 定义OPS连接类。

  ```
  class OPSConnection()
  ```

  此模块不需要用户编辑。

- 封装OPS连接。

  ```
  self.conn = http.client.HTTPConnection
  ```

  此模块不需要用户编辑。

- 调用平台底层接口。

  ```
  def close()
  def create()
  def delete()
  def get()
  def set()
  ```

  此模块不需要用户编辑。

- 定义请求为REST风格。

  ```
  def _rest_call()
  ```

  此模块不需要用户编辑。
- OPS进程执行异常。

  ```
  class OPIExecError()
  ```

  此模块不需要用户编辑。
- 打印预配置信息。

  ```
  print_precfg_info()
  ```

  此模块不需要用户编辑。
- 创建和配置Eth-Trunk口。

  ```
  create_ethtrunk()
  ```

  此模块不需要用户编辑。
- 激活端口License。

  ```
  active_port_license()
  ```

  此模块不需要用户编辑。
- 删除Eth-Trunk口。

  ```
  delete_ethtrunk()
  ```

  此模块不需要用户编辑。
- 配置设备VLAN值。

  ```
  config_vlan()
  ```

  此模块不需要用户编辑。
- 定义设备上电自动部署功能的总流程。

  ```
  def main_proc()
  def main()
  ```

  此模块不需要用户编辑。

  main()函数要求必须有，否则脚本无法运行。

ini 格式的中间文件

ini文件是中间文件格式中的一种，用来保存设备及其版本文件信息。

ini文件的文件名必须以“.ini”作为后缀名，格式如下:

说明

下列文件中的SHA256校验码只是举例，配置时以实际计算的值为准。

```
#sha256sum="88298f97c634cb04b1eb4fe9ad2255abffc0a246112e1960cb6402f6b799f8b6"
;BEGIN ROUTER
[GLOBAL CONFIG]
FILESERVER=sftp://username:password@hostname:port/path/

[DEVICEn DESCRIPTION]
ESN=2102351931P0C3000154
MAC=00e0-fc12-3456
DEVICETYPE=DEFAULT
BOOT-PYTHON-FILE=V800R023C00SPC500.py
SYSTEM-SOFTWARE=V800R023C00SPC500.cc
SYSTEM-CONFIG=V800R023C00SPC500.cfg
SYSTEM-PAT=V800R023C00SPC500SPH001.PAT
;END ROUTER
```


ZTP执行过程中可能存在问题，可以执行如下Python脚本来解决问题，并将BOOT-PYTHON-FILE字段配置的Python脚本放在FILESERVER文件服务器中。

```
#sha256sum="045237bbf3e623cd728b4b8fd4fa9e270838e0d540486e7dcdfdcb8086fd152b"
import logging
import ops

def main():
    try:
        logging.info("******************")
        logging.info(" run python script start")
        _ops = ops.ops()
        handle, err_desp = _ops.cli.open()
        _ops.cli.execute(handle, "return", None)
        fd, _, err_desp = _ops.cli.execute(handle, "display startup", None)
        logging.info('display startup')
        logging.info(fd)
        fd, _, err_desp = _ops.cli.execute(handle, "system-view", None)
        logging.info('system-view')
        logging.info(fd)
        fd, _, err_desp = _ops.cli.execute(handle, "display interface brief", None)
        logging.info('display interface brief')
        logging.info(fd)
        fd, _, err_desp = _ops.cli.execute(handle, "diagnose", None)
        logging.info('diagnose')
        logging.info(fd)
        fd, _, err_desp = _ops.cli.execute(handle, "disp access um blackbox state", None)
        logging.info('disp access um blackbox state')
        logging.info(fd)
        fd, _, err_desp = _ops.cli.execute(handle, "commit", None)
        logging.info('commit')
        logging.info(fd)
        logging.info(" run python script end")
        logging.info("******************")
    except Exception as reason:
        logging.error(reason)
    return True

if __name__ == "__main__":
    main()
```

表 1-84 ini 文件字段含义

字段	是否可选	描述
#sha256sum	必选	ini文件的SHA256校验码。 **说明** SHA256校验码是由“;BEGIN ROUTER”到“;END ROUTER”中的内容计算而来。 实际使用过程中，需将上述文件中的第一行删除，并将“;BEGIN ROUTER”提至第一行，然后开始计算SHA256校验码，最后将生成的SHA256校验码“#sha256sum=”写入文件开头。 文件支持使用SHA256算法进行完整性校验，该种算法安全性较高。 用户可以使用以下两种方法生成脚本文件的SHA256校验码： 1. SHA256计算工具（如HashMyFiles）； 2. 执行Windows自带的**certutil -hashfile** *filename* SHA256命令。

字段	是否可选	描述
;BEGIN ROUTER	必选	起始标志，此字段不能修改。
[GLOBAL CONFIG]	必选	全局配置起始标志，此字段不能修改。
FILESERVER	必选	此字段为版本文件服务器的地址。可以使用TFTP/FTP/SFTP方式获取文件，格式如下： ● tftp://*hostname*/*path*/ ● ftp://[*username*[:*password*]@]*hostname*/*path*/ ● sftp://[*username*[:*password*]@]*hostname*[:*port*]/*path*/ 其中*username*、*password*、*port*参数为可选项。*path*参数指定文件服务器存放版本文件的目录。*hostname*参数为服务器地址，可以是IPv4地址或域名，也可以是IPv6地址。*port*参数的取值范围为0～65535，超出范围按照默认端口22处理，仅在SFTP服务器地址为IPv4情况下支持配置端口号。
[DEVICE*n* DESCRIPTION]	必选	文件信息描述起始标志，*n*表示设备的编号。整数形式，从0开始。
ESN	可选	设备序列号。如果ESN=DEFAULT，表示不需要匹配ESN序列号，否则需要和设备匹配ESN。 缺省情况下，ESN为DEFAULT。如果该标志不存在或为空，则表示为缺省情况。 **说明** ● 设备的ESN号可以通过设备出厂时外包装上的名牌信息获取。 ● ESN号不区分大小写。 ● 建议使用设备的ESN号指定具体设备的配置信息，尽量不要使用DEFAULT进行批量配置。

字段	是否可选	描述
MAC	可选	设备MAC地址，格式为：XXXX-XXXX-XXXX，X为十六进制数。如果MAC=DEFAULT，表示不需要匹配MAC地址，否则需要和设备匹配MAC地址。 优先匹配ESN，再匹配MAC。 缺省情况下，MAC为DEFAULT。如果该标志不存在或为空，则表示为缺省情况。 **说明** • 设备的MAC地址可以通过设备出厂时外包装上的名牌信息获取。 • MAC地址不区分大小写。 • 用户需严格按照设备上显示的MAC地址格式填写中间文件，例如：设备显示为00e0-fc12-3456，则00e0fc123456是错误填写，“-”也会进行校验。 • 建议使用设备的MAC地址指定具体设备的配置信息，尽量不要使用DEFAULT进行批量配置。
DEVICETYPE	可选	匹配的设备类型。如果DEVICETYPE=DEFAULT，表示不匹配设备类型，否则需要匹配设备类型。 缺省情况下，DEVICETYPE为DEFAULT。如果该标志不存在或为空，则表示为缺省情况。 **说明** • 具体的设备类型请参见“描述》硬件描述》机框”章节。 • DEVICETYPE与实际设备不符时，将重新执行ZTP流程。
SYSTEM-SOFTWARE	可选	系统软件名称，后缀名为“.cc”，可以包含文件路径，如：“path1/path2/V800R023C00SPC500.cc”。
SYSTEM-CONFIG	必选	配置文件名称，后缀名为“.cfg”、“.zip”或“.dat”，可以包含文件路径，如：“path1/path2/V800R023C00SPC500.cfg”。 **说明** 配置文件名称不要使用设备缺省的名称vrpcfg.zip。
SYSTEM-PAT	可选	补丁文件名称，后缀名为“.PAT”，可以包含文件路径，如：“path1/path2/V800R023C00SPC500.PAT”。
;END ROUTER	必选	文件结束标志，此字段不能修改。

字段	是否可选	描述
BOOT-PYTHON-FILE	可选	ZTP执行过程中需要执行的Python脚本文件，用来解决一些ZTP执行过程中可能遇到的问题，请将BOOT-PYTHON-FILE字段配置的Python脚本放在FILESERVER文件服务器中。 **说明** BOOT-PYTHON-FILE配置的Python脚本文件名称不要与ZTP预配置文件重名，名称不可包含非法字符，如果文件中包含使ZTP异常退出的命令行，例如**set ztp disable**，**undo pnp enable**，ops视图disable等，可能会导致ZTP退出，影响ZTP主流程。

说明

设备按照[DEVICEn DESCRIPTION]配置在ini文件中从前到后的顺序进行匹配。

匹配时会优先匹配DEVICETYPE选项：

- 如果DEVICETYPE为DEFAULT，或者该字段不存在或为空时，则无需匹配DEVICETYPE字段，只需匹配ESN或MAC，其中有一个匹配成功，则认为该条DESCRIPTION配置有效，否则认为该条DESCRIPTION配置无效。
- 如果DEVICETYPE有值，且不为DEFAULT时，需先匹配DEVICETYPE字段，匹配不成功，则认为该DESCRIPTION配置无效，继续匹配下一个；匹配成功时，需继续匹配ESN或MAC，其中有一个匹配成功，则认为该条DESCRIPTION配置有效，否则认为该条DESCRIPTION配置无效。
- 如果ESN和MAC全部为DEFAULT时，则无需对这两个字段进行匹配。

Python 格式的中间文件

Python脚本文件是中间文件格式中的一种。设备通过运行Python脚本来下载版本文件。

Python脚本文件的文件名必须以“.py”作为后缀名，格式如**Python脚本文件示例**所示，请使用Python3.7语法编写或修改脚本文件。详细脚本文件解释请见**Python脚本文件解释**。

Python 脚本文件示例

说明

该脚本文件仅作为样例，用户可以根据实际开局场景进行修改。

下列文件中的SHA256校验码只是举例，配置时以实际计算的值为准。

```
#sha256sum="68549835edaa5c5780d7b432485ce0d4fdaf6027a8af24f322a91b9f201a5101"
#!/usr/bin/env python
# coding=utf-8
#
# Copyright (C) Huawei Technologies Co., Ltd. 2008-2013. All rights reserved.
# ------------------------------------------------------------------------------------------------------------------------
# History:
# Date            Author                Modification
# 20180122          Author                  created file.
# ------------------------------------------------------------------------------------------------------------------------
```

```
"""
Zero Touch Provisioning (ZTP) enables devices to automatically load version files including system software,
patch files, configuration files when the device starts up, the devices to be configured must be new devices
or have no configuration files.

This is a sample of Zero Touch Provisioning user script. You can customize it to meet the requirements of
your network environment.
"""

import hashlib
import http.client
import logging
import os
import re
import string
import traceback
import xml.etree.ElementTree as etree
from time import sleep
from urllib.parse import urlparse

import ops


# error code
OK = 0
ERR = 1

# File server in which stores the necessary system software, configuration and patch files:
#   1) Specify the file server which supports the following format.
#      tftp://hostname/path
#      ftp://[username[:password]@]hostname/path
#      sftp://[username[:password]@]hostname[:port]/path
#   2) Do not add a trailing slash at the end of file server path.
FILE_SERVER = 'sftp://username:password@hostname:port/path/'

# Remote file paths:
#   1) The path may include directory name and file name.
#   2) If file name is not specified, indicate the procedure can be skipped.
#   3) If you do not want image, please set it as REMOTE_PATH_IMAGE = {} or REMOTE_PATH_IMAGE =
{'DEVICETYPE': ''}
# File paths of system software on file server, filename extension is '.cc'.
REMOTE_PATH_IMAGE = {
    'NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X': 'V800R023C00SPC500.cc'
}
# File path of configuration file on file server, filename extension is '.cfg', '.zip' or '.dat'.
REMOTE_PATH_CONFIG = 'conf_%s.cfg'
# If you do not want patch, please set it as REMOTE_PATH_PATCH = {} or REMOTE_PATH_PATCH =
{'DEVICETYPE': ''}
# File path of patch file on file server, filename extension is '.pat'
REMOTE_PATH_PATCH = {
    'NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X': 'V800R023C00SPC500SPH001.pat'
}
# File path of sha256 file, contains sha256 value of image / patch / configuration, file extension is '.txt'
# If you do not want sha256-file,please set it as REMOTE_PATH_SHA256 = ''
REMOTE_PATH_SHA256 = 'sha256.txt'

# constant
# autoconfig
HTTP_OK = 200
HTTP_BAD_REQUEST = 400
HTTP_BAD_RESPONSE = -1
CONFLICT_RETRY_INTERVAL = 5

POST_METHOD = 'POST'
GET_METHOD = 'GET'
DELETE_METHOD = 'DELETE'
PUT_METHOD = 'PUT'

MAX_TIMES_GET_STARTUP = 120
```



```
GET_STARTUP_INTERVAL = 15

MAX_TIMES_CHECK_STARTUP = 205
MAX_TIMES_CHECK_STARTUP_SLAVE = 265
CHECK_STARTUP_INTERVAL = 5

FILE_DELETE_DELAY_TIME = 3

# ztplib
INSTALL_CHECK_TIMES = 3
PRE_CONFIG_RETRY_TIMES = 12
NOT_START_PNP = 2
ZTP_SLEEP_INTERVAL = 3
LAST_STATE_MAP = {'true': 'enable', 'false': 'disable'}

# DNS
DNS_STATE_MAP = {'true': 'enable', 'false': 'disable'}

# download
FILE_TRANSFER_RETRY_TIMES = 3
FILE_DOWNLOAD_INTERVAL_TIME = 5

DISK_SPACE_NOT_ENOUGH = 48
SPACE_NOT_ENOUGH_TAG_LIST = [48, 3617]

IPV4 = 'ipv4'
IPV6 = 'ipv6'

OPS_CLIENT = None


# exception
class PNPStopError(Exception):
    """Stop by pnp"""


class OPIExecError(Exception):
    """OPS Connection Exception"""


class NoNeedZTP2PNPError(Exception):
    """No need start ztp"""


class SysRebootError(Exception):
    """Device reboot error"""


class ZTPDisableError(Exception):
    """ZTP set disable error"""


# opslib
class OPSConnection:
    """Make an OPS connection instance."""
    __slots__ = ['host', 'port', 'headers', 'conn']

    def __init__(self, host, port=80):
        self.host = host
        self.port = port
        self.headers = {
            'Content-type': 'application/xml',
            'Accept': 'application/xml'
        }

        self.conn = http.client.HTTPConnection(self.host, self.port)

    def close(self):
        """Close the connection"""
```



```
        self.conn.close()

    def create(self, uri, req_data, need_retry=True):
        """Create a resource on the server"""
        ret = self._rest_call(POST_METHOD, uri, req_data)
        if ret[0] != HTTP_OK and need_retry:
            sleep(CONFLICT_RETRY_INTERVAL)
            ret = self._rest_call(POST_METHOD, uri, req_data)
        return ret

    def delete(self, uri, req_data, need_retry=True):
        """Delete a resource on the server"""
        ret = self._rest_call(DELETE_METHOD, uri, req_data)
        if ret[0] != HTTP_OK and need_retry:
            sleep(CONFLICT_RETRY_INTERVAL)
            ret = self._rest_call(DELETE_METHOD, uri, req_data)
        return ret

    def get(self, uri, req_data=None, need_retry=True):
        """Retrieve a resource from the server"""
        ret = self._rest_call(GET_METHOD, uri, req_data)
        if (ret[0] != HTTP_OK or ret[2] == '') and need_retry:
            sleep(CONFLICT_RETRY_INTERVAL)
            ret = self._rest_call(GET_METHOD, uri, req_data)
        return ret

    def set(self, uri, req_data, need_retry=True):
        """Update a resource on the server"""
        ret = self._rest_call(PUT_METHOD, uri, req_data)
        if ret[0] != HTTP_OK and need_retry:
            sleep(CONFLICT_RETRY_INTERVAL)
            ret = self._rest_call(PUT_METHOD, uri, req_data)
        return ret

    def _rest_call(self, method, uri, req_data):
        """REST call"""
        body = '' if req_data is None else req_data

        try:
            self.conn.request(method, uri, body, self.headers)
        except http.client.CannotSendRequest:
            logging.warning('An error occurred during http request, try to send request again')
            self.close()
            self.conn = http.client.HTTPConnection(self.host, self.port)
            self.conn.request(method, uri, body, self.headers)
        except http.client.InvalidURL:
            logging.warning('Failed to find url: %s in OPS whitelist', uri)
            return HTTP_BAD_REQUEST, '', ''

        try:
            response = self.conn.getresponse()
        except AttributeError:
            logging.warning('An error occurred during http response, try again')
            return HTTP_BAD_RESPONSE, '', ''

        rest_message = response.read()
        if isinstance(rest_message, bytes):
            rest_message = str(rest_message, 'iso-8859-1')
        # logging.debug('uri = %s ret = %s \n %s \n %s', uri, response.status, req_data, rest_message)

        ret = (response.status, response.reason, rest_message)
        return ret


OPS_CLIENT = OPSConnection("localhost")


# pnplib
def dhcp_stop():
```



```
    """Stop DHCP client, include dhcpv4 and dhcpv6."""
    logging.info('Stopping dhcp client')

    uri = '/pnp/stopPnp'
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
        <stopPnp/>'''
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        # ignore stop pnp err
        logging.warning('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        logging.warning('Failed to stop dhcp client')
        return

    logging.info('DHCP client has stopped')
    return


# commlib
def get_cwd():
    """Get the full filename of the current working directory"""
    logging.info("Get the current working directory...")
    namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
    uri = "/vfm/pwds/pwd"
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
                    <pwd>
                        <dictionaryName/>
                    </pwd>
                '''
    ret, _, rsp_data = OPS_CLIENT.get(uri, req_data)
    if ret != http.client.OK or rsp_data is '':
        raise OPIExecError('Failed to get the current working directory')

    root_elem = etree.fromstring(rsp_data)
    uri = 'data' + uri.replace('/', '/vrp:') + '/vrp:dictionaryName'
    elem = root_elem.find(uri, namespaces)
    if elem is None:
        raise OPIExecError('Failed to get the current working directory for no "directoryName" element')

    return elem.text


def file_exist(file_name, dir_path=None):
    """Returns True if file_path refers to an existing file, otherwise returns False"""
    uri = '/vfm/dirs/dir'
    str_temp_1 = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <dir>
            <fileName>$fileName</fileName>
        </dir>''')
    str_temp_2 = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <dir>
            <dirName>$dirName</dirName>
            <fileName>$fileName</fileName>
        </dir>''')

    if dir_path:
        req_data = str_temp_2.substitute(dirName=dir_path, fileName=file_name)
    else:
        req_data = str_temp_1.substitute(fileName=file_name)
    ret, _, rsp_data = OPS_CLIENT.get(uri, req_data)
    if ret != HTTP_OK or rsp_data == '':
        return False

    root_elem = etree.fromstring(rsp_data)
    namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
    uri = 'data' + uri.replace('/', '/vrp:') + '/vrp:fileName'
    elem = root_elem.find(uri, namespaces)
    if elem is None:
        return False
```



```
    return True


def copy_file(src_path, dest_path):
    """Copy a file"""
    logging.info('Copy file %s to %s', os.path.basename(src_path), os.path.basename(dest_path))

    if 'slave' in dest_path:
        file_name = dest_path.split(':/')[1]
        if file_exist(file_name, 'slave#cfcard:/'):
            logging.info('Detect dest file exist, delete it first')
            delete_file(dest_path)

    uri = '/vfm/copyFile'
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <copyFile>
            <srcFileName>$src</srcFileName>
            <desFileName>$dest</desFileName>
        </copyFile>''')
    req_data = str_temp.substitute(src=src_path, dest=dest_path)

    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data, False)
    if ret != HTTP_OK:
        file_name = dest_path.split(':/')[1]
        if file_exist(file_name, "slave#cfcard:/"):
            logging.info('Exists file copy fragment, delete it')
            delete_file(dest_path)
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        logging.error('Failed to copy %s to %s', os.path.basename(src_path), os.path.basename(dest_path))
        return False

    logging.info('succeed to copy')
    return True


def delete_file(file_path):
    """Delete a file permanently"""
    if file_path is None or file_path == '':
        return

    logging.info('Delete file %s permanently', os.path.basename(file_path))
    uri = '/vfm/deleteFileUnRes'
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <deleteFileUnRes>
            <fileName>$filePath</fileName>
        </deleteFileUnRes>
        ''')
    req_data = str_temp.substitute(filePath=file_path)
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        logging.error('Failed to delete the file %s permanently', os.path.basename(file_path))


def delete_file_all(file_path, slave, protect_file_list=None):
    """Delete a file permanently on all main boards"""
    if not file_path:
        return
    if protect_file_list:
        for protect_file in protect_file_list:
            if file_path == protect_file:
                return
    file_name = os.path.basename(file_path)
    file_path_t = file_path[:len(file_path) - len(file_name)]
    if file_exist(file_name, file_path_t):
        delete_file(file_path)
    if slave and file_exist(file_name, 'slave#' + file_path_t):
        delete_file('slave#' + file_path)
```



```
def has_slave_mpu():
    """Whether device has slave MPU, returns a bool value
    :raise OPIExecError
    """
    logging.info("Test whether device has slave MPU")
    uri = '/devm/phyEntitys'
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
        <phyEntitys>
            <phyEntity>
                <entClass>mpuModule</entClass>
                <entStandbyState/>
                <position/>
            </phyEntity>
        </phyEntitys>'''

    has_slave = False
    mpu_slot = {}.fromkeys(('master', 'slave'))
    ret, err_code, rsp_data = OPS_CLIENT.get(uri, req_data)
    if ret != HTTP_OK or rsp_data == '':
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to get the device slave information')

    root_elem = etree.fromstring(rsp_data)
    namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
    uri = 'data{0}/vrp:phyEntity'.format(uri.replace('/', '/vrp:'))
    for entity in root_elem.findall(uri, namespaces):
        elem = entity.find("vrp:entStandbyState", namespaces)
        if elem is not None and elem.text.lower().find('slave') >= 0:
            has_slave = True
            elem = entity.find("vrp:position", namespaces)
            if elem is not None:
                mpu_slot['slave'] = elem.text
        if elem is not None and elem.text.lower().find('master') >= 0:
            elem = entity.find("vrp:position", namespaces)
            if elem is not None:
                mpu_slot['master'] = elem.text

    logging.info('Device has slave: %s', has_slave)
    return has_slave, mpu_slot


def get_system_info():
    """Get device product esn mac
    :raise: OPIExecError
    """
    logging.info("Get the system information...")
    uri = "/system/systemInfo"
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
    <systemInfo>
        <productName/>
        <esn/>
        <mac/>
    </systemInfo>
    '''

    sys_info = {}.fromkeys(('productName', 'esn', 'mac'), '')
    ret, err_code, rsp_data = OPS_CLIENT.get(uri, req_data)
    if ret != HTTP_OK or rsp_data == '':
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to get the system information')

    root_elem = etree.fromstring(rsp_data)
    namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
    uri = 'data' + uri.replace('/', '/vrp:')
    nslen = len(namespaces['vrp'])
    elem = root_elem.find(uri, namespaces)
    if elem is not None:
        for child in elem:
```



```
            tag = child.tag[nslen + 2:]
            if tag in list(sys_info.keys()):
                sys_info[tag] = child.text

    return sys_info


def reboot_system(save_config='false'):
    """Reboot system."""
    logging.info('System will reboot to make the configuration take effect')

    if save_config not in ['true', 'false']:
        return

    logging.info("start to issue reboot command")
    sleep(10)
    uri = "/devm/reboot"
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <reboot>
            <saveConfig>$saveConfig</saveConfig>
        </reboot>''')
    req_data = str_temp.substitute(saveConfig=save_config)
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK or rsp_data == '':
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to execute the reboot system operation')


def check_file_type_valid(image, config, patch, sha256_file):
    """Test whether argument paths are valid."""
    logging.info("Test whether argument paths are valid...")
    # check image file path
    file_name = os.path.basename(image)
    if file_name is not '' and not file_name.lower().endswith('.cc'):
        logging.error('Error: Invalid filename extension of system software')
        return False

    # check config file path
    file_name = os.path.basename(config)
    file_name = file_name.lower()
    _, ext = os.path.splitext(file_name)
    if file_name is not '' and ext not in ['.cfg', '.zip', '.dat']:
        logging.error('Error: Invalid filename extension of configuration file')
        return False

    # check patch file path
    file_name = os.path.basename(patch)
    if file_name is not '' and not file_name.lower().endswith('.pat'):
        logging.error('Error: Invalid filename extension of patch file')
        return False

    # check sha256 file path
    file_name = os.path.basename(sha256_file)
    if file_name is not '' and not file_name.lower().endswith('.txt'):
        logging.error('Error: Invalid filename extension of %s file', sha256_file)
        return False

    return True

def hide_content(content: str, least_length: int) -> str:
    if not content:
        return str(content)

    content = str(content)
    if least_length > len(content):
        return content
    hide = ""
    for _ in content[:-least_length]:
        hide += "*"
```



```
    return hide + content[-least_length:]

# startuplib
class StartupInfo:
    """Startup configuration information

    image: startup system software
    config: startup saved-configuration file
    patch: startup patch package
    """

    def __init__(self, image=None, config=None, patch=None):
        self.image = image
        self.config = config
        self.patch = patch


class Startup:
    """Startup configuration information

    current: current startup configuration
    next: current next startup configuration
    """

    def __init__(self):
        self.current, self.next = self._get_startup_info()
        self.startup_info_from_ini_or_cfg = {}
        self.startup_info_before_set = StartupInfo()

    @staticmethod
    def _get_startup_info(retry=True):
        """Get device startup information
            :raise
                opslib.OPIExecError
        """
        uri = '/cfg/startupInfos/startupInfo'
        req_data = '''<?xml version="1.0" encoding="UTF-8"?>
            <startupInfo>
                <position/>
                <configedSysSoft/>
                <curSysSoft/>
                <nextSysSoft/>
                <curStartupFile/>
                <nextStartupFile/>
                <curPatchFile/>
                <nextPatchFile/>
            </startupInfo>'''

        if retry is True:
            retry_time = MAX_TIMES_GET_STARTUP
        else:
            retry_time = 1

        cnt = 0
        elem = None
        namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
        ns_len = len(namespaces['vrp'])
        path = 'data' + uri.replace('/', '/vrp:')  # match path
        while cnt < retry_time:
            ret, _, rsp_data = OPS_CLIENT.get(uri, req_data)
            if ret != HTTP_OK or rsp_data == '':
                cnt += 1
                logging.warning('Failed to get the startup information')
                # sleep to wait for system ready when no query result
                sleep(GET_STARTUP_INTERVAL)
                continue

            root_elem = etree.fromstring(rsp_data)
            elem = root_elem.find(path, namespaces)
```

```
        if elem is not None:
            break
        logging.warning('No query result while getting startup info')
        # sleep to wait for system ready when no query result
        sleep(GET_STARTUP_INTERVAL)
        cnt += 1

    if elem is None:
        raise OPIExecError('Failed to get the startup information')

    current = StartupInfo()  # current startup info
    curnext = StartupInfo()  # next startup info
    for child in elem:
        # skip the namespace, '{namespace}text'
        tag = child.tag[ns_len + 2:]
        if tag == 'curSysSoft':
            current.image = child.text
        elif tag == 'nextSysSoft':
            curnext.image = child.text
        elif tag == 'curStartupFile' and child.text != 'NULL':
            current.config = child.text
        elif tag == 'nextStartupFile' and child.text != 'NULL':
            curnext.config = child.text
        elif tag == 'curPatchFile' and child.text != 'NULL':
            current.patch = child.text
        elif tag == 'nextPatchFile' and child.text != 'NULL':
            curnext.patch = child.text
        else:
            continue

    return current, curnext

@staticmethod
def _set_startup_image_file(file_path, slave=True):
    """Set the next startup system software"""
    file_name = os.path.basename(file_path)
    logging.info('Set the next startup system software to %s, please wait a moment', file_name)
    uri = '/sum/startupbymode'
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <startupbymode>
            <softwareName>$fileName</softwareName>
            <mode>$startupMode</mode>
        </startupbymode>''')

    if slave:
        startup_mode = 'STARTUP_MODE_ALL'
    else:
        startup_mode = 'STARTUP_MODE_PRIMARY'

    req_data = str_temp.substitute(fileName=file_name, startupMode=startup_mode)
    # it is a action operation, so use create for HTTP POST
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to set startup system software')

@staticmethod
def _set_startup_config_file(file_path):
    """Set the next startup saved-configuration file"""
    file_name = os.path.basename(file_path)
    logging.info('Set the next startup saved-configuration file to %s', file_name)
    uri = '/cfg/setStartup'
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <setStartup>
            <fileName>$fileName</fileName>
        </setStartup>''')

    req_data = str_temp.substitute(fileName=file_name)
    # it is a action operation, so use create for HTTP POST
```



```
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to set startup configuration file')

@staticmethod
def _del_startup_config_file():
    """Delete startup config file"""
    logging.info('Delete the next startup config file')
    uri = '/cfg/clearStartup'
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
        <clearStartup>
        </clearStartup>'''
    # it is a action operation, so use create for HTTP POST
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to delete startup configuration file')

@staticmethod
def _set_startup_patch_file(file_path):
    """Set the next startup patch file"""
    file_name = os.path.basename(file_path)
    logging.info('Set the next startup patch file to %s', file_name)
    uri = "/patch/startup"
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <startup>
            <packageName>$fileName</packageName>
        </startup>''')
    req_data = str_temp.substitute(fileName=file_name)
    # it is a action operation, so use create for HTTP POST
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to set startup patch file')

@staticmethod
def _reset_startup_patch_file():
    """Reset patch file for system to startup"""
    logging.info('Reset the next startup patch file')
    uri = '/patch/resetpatch'
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
        <resetpatch/>'''
    # it is a action operation, so use create for HTTP POST
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to reset startup patch file')

def _check_next_startup_file(self, file_name, check_item, slave):
    """Check next startup file ready
        check_item: [image, config, patch]
    """
    if check_item not in ['image', 'config', 'patch']:
        return True

    logging.info('Check the next startup %s information', check_item)
    if slave:
        check_time = MAX_TIMES_CHECK_STARTUP_SLAVE
    else:
        check_time = MAX_TIMES_CHECK_STARTUP
    cnt = 0
    while cnt < check_time:
        _, next_startup = self._get_startup_info()
        startup_file_name = getattr(next_startup, check_item)
        if startup_file_name == file_name:
            sleep(CHECK_STARTUP_INTERVAL)
            logging.info('The next system %s check successfully', check_item)
            return True
```

```
        # sleep to wait for system ready when no query result
        sleep(CHECK_STARTUP_INTERVAL)
        if cnt % 12 == 0:
            # logging every minute
            logging.info('Checking the next startup %s, please wait a moment', check_item)
        cnt += 1
    logging.warning('The next system %s is not ready', check_item)
    return False

def set_startup_info(self, image_file, config_file, patch_file, slave):
    """Set the next startup information."""
    # backup startup_info set by user
    cur_startup, cur_next_startup = self._get_startup_info()
    self.startup_info_before_set.image = cur_next_startup.image
    self.startup_info_before_set.patch = cur_next_startup.patch
    self.startup_info_before_set.config = cur_next_startup.config
    logging.info("save startup config before ztp setting")

    logging.info('Start to set next startup information')
    # 1. Set next startup system software
    if image_file is not None:
        try:
            self._set_startup_image_file(image_file)
            if self._check_next_startup_file(image_file, 'image', slave) is False:
                raise OPIExecError('Failed to check the next startup image file')
        except OPIExecError as reason:
            logging.error(reason)
            delete_file_all(image_file, slave, [cur_startup.image, cur_next_startup.image])
            self.reset_startup_info(slave)
            raise

    # 2. Set next startup patch file
    if patch_file is not None:
        try:
            self._set_startup_patch_file(patch_file)
            if self._check_next_startup_file(patch_file, 'patch', slave) is False:
                raise OPIExecError('Failed to check the next startup patch file')
        except OPIExecError as reason:
            logging.error(reason)
            delete_file_all(patch_file, slave, [cur_startup.patch, cur_next_startup.patch])
            self.reset_startup_info(slave)
            raise

    # 3. Set next startup config file
    if config_file is not None:
        try:
            self._set_startup_config_file(config_file)
            if self._check_next_startup_file(config_file, 'config', slave) is False:
                raise OPIExecError('Failed to check the next startup config file')
        except OPIExecError as reason:
            logging.error(reason)
            delete_file_all(config_file, slave, [cur_startup.config, cur_next_startup.config])
            self.reset_startup_info(slave)
            raise

def reset_startup_info(self, slave):
    """Reset startup info and delete the downloaded files"""
    logging.info('Start to reset next startup information')
    if not self.startup_info_before_set.image:
        logging.error('image of roll back point is None')
        return
    cur_startup, next_startup = self._get_startup_info()

    # 1. Reset next startup config file and delete it
    try:
        # user configure startup info after ZTP
        if next_startup.config != self.startup_info_from_ini_or_cfg.get("SYSTEM-CONFIG"):
            logging.info("no need to reset startup config")
            if self.startup_info_from_ini_or_cfg.get("SYSTEM-CONFIG"):
```

```
            sleep(FILE_DELETE_DELAY_TIME)
            delete_file_all(self.startup_info_from_ini_or_cfg.get("SYSTEM-CONFIG"), slave,
                    [cur_startup.config, next_startup.config])
        # user do not configure startup info
        elif next_startup.config != self.startup_info_before_set.config:
            logging.info("reset startup config to the beginning")
            if self.startup_info_before_set.config is None:
                self._del_startup_config_file()
            else:
                self._set_startup_config_file(self.startup_info_before_set.config)
                if self._check_next_startup_file(self.startup_info_before_set.config, 'config', slave) is not True:
                    raise OPIExecError('Failed to check the next startup config file')
            if next_startup.config:
                sleep(FILE_DELETE_DELAY_TIME)
                delete_file_all(next_startup.config, slave,
                        [cur_startup.config, self.startup_info_before_set.config])
    except Exception as reason:
        logging.error(reason)

    # 2. Reset next startup patch file and delete it
    try:
        # user configure startup info after ZTP
        if next_startup.patch != self.startup_info_from_ini_or_cfg.get("SYSTEM-PAT"):
            logging.info("no need to reset startup patch")
            if self.startup_info_from_ini_or_cfg.get("SYSTEM-PAT"):
                sleep(FILE_DELETE_DELAY_TIME)
                delete_file_all(self.startup_info_from_ini_or_cfg.get("SYSTEM-PAT"), slave,
                        [cur_startup.patch, next_startup.patch])
        # user do not configure startup info
        elif next_startup.patch != self.startup_info_before_set.patch:
            logging.info("reset startup patch to the beginning")
            if self.startup_info_before_set.patch is None:
                self._reset_startup_patch_file()
                if STARTUP._check_next_startup_file(None, 'patch', slave) is not True:
                    raise opslib.OPIExecError('Failed to reset the next startup patch file')
            else:
                self._set_startup_patch_file(self.startup_info_before_set.patch)
                if self._check_next_startup_file(self.startup_info_before_set.patch, 'patch', slave) is not True:
                    raise OPIExecError('Failed to check the next startup patch file')
            if next_startup.patch:
                sleep(FILE_DELETE_DELAY_TIME)
                delete_file_all(next_startup.patch, slave,
                        [cur_startup.patch, self.startup_info_before_set.patch])
    except Exception as reason:
        logging.error(reason)

    # 3. Reset next startup system software and delete it
    try:
        # user configure startup info after ZTP
        if next_startup.image != self.startup_info_from_ini_or_cfg.get("SYSTEM-SOFTWARE"):
            logging.info("no need to reset startup image")
            if self.startup_info_from_ini_or_cfg.get("SYSTEM-SOFTWARE"):
                sleep(FILE_DELETE_DELAY_TIME)
                delete_file_all(self.startup_info_from_ini_or_cfg.get("SYSTEM-SOFTWARE"), slave,
                        [cur_startup.image, next_startup.image])
        # user do not configure startup info
        elif next_startup.image != self.startup_info_before_set.image:
            logging.info("reset startup config to the beginning")
            self._set_startup_image_file(self.startup_info_before_set.image)
            if self._check_next_startup_file(self.startup_info_before_set.image, 'image', slave) is not True:
                raise OPIExecError('Failed to check the next startup image file')
            if next_startup.image:
                sleep(FILE_DELETE_DELAY_TIME)
                delete_file_all(next_startup.image, slave,
                        [cur_startup.image, self.startup_info_before_set.image])
    except Exception as reason:
        logging.error(reason)

def set_startup_info_from_ini_or_cfg(self, startup_info):
```

```
        for item_key in ['SYSTEM-SOFTWARE', 'SYSTEM-CONFIG', 'SYSTEM-PAT']:
            if not startup_info[item_key]:
                self.startup_info_from_ini_or_cfg[item_key] = startup_info[item_key]
            else:
                self.startup_info_from_ini_or_cfg[item_key] = 'cfcard:/' + startup_info[item_key]


def convert_byte_to_str(data):
    result = data
    if not isinstance(data, str):
        result = str(data, "iso-8859-1")
    return result


def sha256sum(fname, need_skip_first_line=False):
    """
    Calculate sha256 num for this file.
    """

    def read_chunks(fhdl):
        '''read chunks'''
        chunk = fhdl.read(8096)
        while chunk:
            yield chunk
            chunk = fhdl.read(8096)
        else:
            fhdl.seek(0)

    sha256_obj = hashlib.sha256()
    if isinstance(fname, str):
        with open(fname, "rb") as fhdl:
            # skip the first line
            fhdl.seek(0)
            if need_skip_first_line:
                fhdl.readline()
            for chunk in read_chunks(fhdl):
                sha256_obj.update(chunk)
    elif fname.__class__.__name__ in ["StringIO", "StringO"]:
        for chunk in read_chunks(fname):
            sha256_obj.update(chunk)
    else:
        pass
    return sha256_obj.hexdigest()


def sha256_get_from_file(fname):
    """Get sha256 num form file, stored in first line"""
    with open(fname, "rb") as fhdl:
        fhdl.seek(0)
        line_first = convert_byte_to_str(fhdl.readline())
    # if not match pattern, the format of this file is not supported
    if not re.match('^#sha256sum="[\\w]{64}"[\r\n]+$', line_first):
        return 'None'

    return line_first[12:76]


def sha256_check_with_first_line(fname):
    """Validate sha256 for this file"""

    work_fname = os.path.join("ztp", fname)
    sha256_calc = sha256sum(work_fname, True)
    sha256_file = sha256_get_from_file(work_fname)

    if sha256_file.lower() != sha256_calc:
        logging.warning('SHA256 check failed, file %s', fname)
        logging.warning('SHA256 checksum of the file "%s" is %s', fname, sha256_calc)
        logging.warning('SHA256 checksum received from the file "%s" is %s', fname, sha256_file)
        return False
```



```
    return True


def parse_sha256_file(fname):
    """parse sha256 file"""

    def read_line(fhdl):
        """read a line by loop"""
        line = fhdl.readline()
        while line:
            yield line
            line = fhdl.readline()
        else:
            fhdl.seek(0)

    sha256_dic = {}
    work_fname = os.path.join("ztp", fname)
    with open(work_fname, "rb") as fhdl:
        for line in read_line(fhdl):
            line_spilt = convert_byte_to_str(line).split()
            if 2 != len(line_spilt):
                continue
            dic_tmp = {line_spilt[0]: line_spilt[1]}
            sha256_dic.update(dic_tmp)
    return sha256_dic


def verify_and_parse_sha256_file(fname):
    """
    verify data integrity of sha256 file and parse this file

    format of this file is like:
    ----------------------------------------------------------------------

    file-name               sha256
    conf_5618642831132.cfg 1254b2e49d3347c4147a90858fa5f59aa2594b7294304f34e7da328bf3cdfbae
    ----------------------------------------------------------------------
    """
    if not sha256_check_with_first_line(fname):
        return ERR, None
    return OK, parse_sha256_file(fname)


def sha256_check_with_dic(sha256_dic, fname):
    """sha256 check with dic"""
    if fname not in sha256_dic:
        logging.info('sha256_dic does not has key %s, no need to do sha256 verification', fname)
        return True

    sha256sum_result = sha256sum(fname, False)
    if sha256_dic[fname].lower() == sha256sum_result:
        logging.info('SHA256 check %s successfully', fname)
        return True

    logging.warning('SHA256 check failed, file %s', fname)
    logging.warning('SHA256 checksum of the file "%s" is %s', fname, sha256sum_result)
    logging.warning('SHA256 checksum received for the file "%s" is %s', fname, sha256_dic[fname])

    return False


def check_parameter(aset):
    seq = ['&', '>', '<', "", "'"]
    if aset:
        for c in seq:
            if c in aset:
                return True
    return False
```



```
def check_filename():
    sys_info = get_system_info()
    url_tuple = urlparse(FILE_SERVER)
    if check_parameter(url_tuple.username) or check_parameter(url_tuple.password):
        logging.error('Invalid username or password, the name should not contain: ' + '&' + ' >' + ' <' + ' "' +
" '.")
        return ERR
    file_name = os.path.basename(REMOTE_PATH_IMAGE.get(sys_info['productName'], ''))
    if file_name is not '' and check_parameter(file_name):
        logging.error(
            'Invalid filename of system software, the name should not contain: ' + '&' + ' >' + ' <' + ' "' + " '.")
        return ERR
    file_name = os.path.basename(REMOTE_PATH_CONFIG)
    if file_name is not '' and check_parameter(file_name):
        logging.error(
            'Invalid filename of configuration file, the name should not contain: ' + '&' + ' >' + ' <' + ' "' + " '.")
        return ERR
    file_name = os.path.basename(REMOTE_PATH_PATCH.get(sys_info['productName'], ''))
    if file_name is not '' and check_parameter(file_name):
        logging.error(
            'Invalid filename of patch file, the name should not contain: ' + '&' + ' >' + ' <' + ' "' + " '.")
        return ERR
    try:
        file_name = os.path.basename(REMOTE_PATH_SHA256)
    except NameError:
        file_name = ''
    if file_name is not '' and check_parameter(file_name):
        logging.error(
            'Invalid filename of sha256 file, the name should not contain: ' + '&' + ' >' + ' <' + ' "' + " '.")
        return ERR

    return OK

def check_path_valid(path):
    invalid_pattern = r"\.\."
    if re.search(invalid_pattern, path):
        return False
    else:
        return True

def download_cfg_file(startup_info, slave, ip_protocol, vpn_instance, sha256_val_dic):
    """ Download configuration file """

    if not check_path_valid(startup_info['SYSTEM-CONFIG']):
        logging.error('cfg file path %s is invalid', startup_info['SYSTEM-CONFIG'])
        return False, ""
    url = os.path.join(startup_info['FILESERVER'], startup_info['SYSTEM-CONFIG'])
    local_path_config = os.path.join('cfcard:', os.path.basename(startup_info['SYSTEM-CONFIG']))
    delete_file_all(local_path_config, slave)
    ret = download_file(url, os.path.basename(local_path_config), ip_protocol, vpn_instance)

    if ret == ERR or not file_exist(os.path.basename(url)):
        logging.error('%s download fail', os.path.basename(startup_info['SYSTEM-CONFIG']))
        return False, local_path_config

    if sha256_val_dic is not None:
        if not startup_info['SYSTEM-CONFIG']:
            return False, local_path_config
        file_name = os.path.basename(startup_info['SYSTEM-CONFIG'])
        if not sha256_check_with_dic(sha256_val_dic, file_name):
            logging.error('Error: SHA256 check failed, file "%s"' % file_name)
            return False, local_path_config

    if slave:
        ret = copy_file(local_path_config, 'slave#' + local_path_config)
        if ret is False:
            logging.error('%s failed copy to slave board', os.path.basename(startup_info['SYSTEM-CONFIG']))
```



```
            return False, local_path_config

    return True, local_path_config


def download_patch_file(startup_info, slave, ip_protocol, vpn_instance, sha256_val_dic):
    """ Download patch file """
    if not check_path_valid(startup_info['SYSTEM-PAT']):
        logging.error('patch file path %s is invalid', startup_info['SYSTEM-PAT'])
        return ERR, ""
    file_name = os.path.basename(startup_info['SYSTEM-PAT'])
    url = os.path.join(startup_info['FILESERVER'], startup_info['SYSTEM-PAT'])
    local_path_patch = os.path.join('cfcard:', file_name)
    delete_file_all(local_path_patch, slave)
    ret = download_file(url, file_name, ip_protocol, vpn_instance)
    if ret == DISK_SPACE_NOT_ENOUGH:
        logging.error('The space of disk is not enough')
        return DISK_SPACE_NOT_ENOUGH, local_path_patch
    if ret != OK or not file_exist(file_name):
        logging.error('%s download fail', file_name)
        return ERR, local_path_patch

    if not sha256_check_with_dic(sha256_val_dic, file_name):
        logging.error('Error: SHA256 check failed, file "%s"' % file_name)
        return ERR, local_path_patch

    if slave:
        ret = copy_file(local_path_patch, 'slave#' + local_path_patch)
        if ret is False:
            logging.error('%s failed copy to slave board', file_name)
            return ERR, local_path_patch

    return OK, local_path_patch


def download_image_file(startup_info, slave, ip_protocol, vpn_instance, sha256_val_dic):
    """ Download system software """
    if not check_path_valid(startup_info['SYSTEM-SOFTWARE']):
        logging.error('image file path %s is invalid', startup_info['SYSTEM-SOFTWARE'])
        return ERR, ""
    file_name = os.path.basename(startup_info['SYSTEM-SOFTWARE'])
    url = startup_info['FILESERVER'] + '/' + startup_info['SYSTEM-SOFTWARE']
    local_path_image = os.path.join('cfcard:', file_name)
    delete_file_all(local_path_image, slave)
    ret = download_file(url, file_name, ip_protocol, vpn_instance)
    if ret == DISK_SPACE_NOT_ENOUGH:
        logging.error('The space of disk is not enough')
        return DISK_SPACE_NOT_ENOUGH, local_path_image
    if ret != OK or not file_exist(file_name):
        logging.error('%s download fail', file_name)
        return ERR, local_path_image

    if not sha256_check_with_dic(sha256_val_dic, file_name):
        logging.error('Error: SHA256 check failed, file "%s"' % file_name)
        return ERR, local_path_image

    if slave:
        ret = copy_file(local_path_image, 'slave#' + local_path_image)
        if ret is False:
            logging.error('%s failed to copy to slave board', file_name)
            return ERR, local_path_image

    return OK, local_path_image


def check_not_contains(target_start_file, start_files):
    return target_start_file and os.path.basename(target_start_file) not in start_files
```



```
def check_contains(target_start_file, start_files):
    return target_start_file and os.path.basename(target_start_file) in start_files


def format_fileserver_path(fileServerPath, ip_protocol):
    """
    :param fileServerPath:  ftp://aaa:aaa@9.9.9.90/a  sftp://aaa:aaa@9.9.9.90:21/a
    :param protocol: ipv4  ipv6
    :return: fileServerPath
    """
    serverPathParts = urlparse(fileServerPath)
    if ip_protocol.lower() == IPV4:
        netlocs = serverPathParts.netloc.split("@")
        url = netlocs[1]
        idx = url.rfind(":")
        if (idx != -1):
            ipAndPort = url.split(":")
            ipEnd = ipAndPort[0][ipAndPort[0].rfind("."):]
            netloc = "*:*@*.*.*" + ipEnd + ":" + ipAndPort[1]
        else:
            ipEnd = serverPathParts.netloc[serverPathParts.netloc.rfind("."):]
            netloc = "*:*@*.*.*" + ipEnd
        return serverPathParts.scheme + "://" + netloc + serverPathParts.path
    elif ip_protocol.lower() == IPV6:
        ipEnd = serverPathParts.netloc[serverPathParts.netloc.rfind(":"):]
        netloc = "*:*@...." + ipEnd
        return serverPathParts.scheme + "://" + netloc + serverPathParts.path
    return None


def verify_sha256_file(slave, ip_protocol, vpn_instance):
    try:
        sha256_val_dic = {}
        cwd = get_cwd()
        file_path = REMOTE_PATH_SHA256
        if not file_path.startswith('/'):
            file_path = '/' + file_path
        file_name = os.path.basename(file_path)
        if file_name:
            url = FILE_SERVER + file_path
            local_path = os.path.join(cwd, "ztp", file_name)
            ret = download_file(url, local_path, ip_protocol, vpn_instance)
            if ret is ERR:
                logging.error('Error: Failed to download sha256 file "%s"' % file_name)
                return ERR, sha256_val_dic
            logging.info('Info: Download sha256 file successfully')
            ret, sha256_val_dic = verify_and_parse_sha256_file(file_name)

            # delete the file immediately
            delete_file_all(local_path, slave)
            if ret is ERR:
                logging.error('Error: sha256 check failed, file "%s"' % file_name)
                return ERR, sha256_val_dic
            return OK, sha256_val_dic
        else:
            sha256_val_dic = {}
            return OK, sha256_val_dic
    except NameError:
        logging.info('no need sha256 to check download file')
        sha256_val_dic = {}
        return OK, sha256_val_dic



def download_startup_file(startup_info, slave, ip_protocol, vpn_instance):
    """Download startup file"""
    # init here
    local_path_config = None
    local_path_patch = None
```



```
    local_path_image = None

    # current STARTUP_INFO
    cur_startup, next_startup = STARTUP._get_startup_info()
    cur_config = None if not cur_startup.config else os.path.basename(cur_startup.config)
    cur_patch = None if not cur_startup.patch else os.path.basename(cur_startup.patch)
    cur_image = None if not cur_startup.image else os.path.basename(cur_startup.image)
    next_config = None if not next_startup.config else os.path.basename(next_startup.config)
    next_patch = None if not next_startup.patch else os.path.basename(next_startup.patch)
    next_image = None if not next_startup.image else os.path.basename(next_startup.image)

if not check_path_valid(startup_info['FILESERVER']):
        fileserver_path = format_fileserver_path(startup_info['FILESERVER'], ip_protocol)
        logging.error('file server path %s is invalid', fileserver_path)
        return ERR, local_path_image, local_path_config, local_path_patch

    # download sha256 file first, used to verify data integrity of files which will be downloaded next
            ret, sha256_val_dic = verify_sha256_file(slave, ip_protocol, vpn_instance)

            if ret is ERR:
                return ERR, None, None, None
    # if user change the startup to the name in ini/cfg, ztp will not download
    # 1. Download configuration file
    if check_not_contains(startup_info['SYSTEM-CONFIG'], [cur_config, next_config]):
        ret, local_path_config = download_cfg_file(startup_info, slave, ip_protocol, vpn_instance,
sha256_val_dic)
        if ret is False:
            logging.info('delete startup file [cfg]')
            delete_startup_file(local_path_image, local_path_config, local_path_patch, slave)
            return ERR, local_path_image, local_path_config, local_path_patch
        logging.info('succeed to download config file')
    elif check_contains(startup_info['SYSTEM-CONFIG'], [cur_config, next_config]):
        logging.warning('The configured config version is the same as the current device version')

    # 2. Download patch file
    if check_not_contains(startup_info['SYSTEM-PAT'], [cur_patch, next_patch]):
        ret, local_path_patch = download_patch_file(startup_info, slave, ip_protocol, vpn_instance,
sha256_val_dic)
        if ret is ERR:
            delete_startup_file(local_path_image, local_path_config, local_path_patch, slave)
            return ERR, local_path_image, local_path_config, local_path_patch
        if ret == DISK_SPACE_NOT_ENOUGH:
            delete_startup_file(local_path_image, None, local_path_patch, slave)
            logging.info('disk space not enough, delete patch')
            return OK, None, local_path_config, None
    elif check_contains(startup_info['SYSTEM-PAT'], [cur_patch, next_patch]):
        logging.warning('The configured patch version is the same as the current device version')

    # 3. Download system software
    if check_not_contains(startup_info['SYSTEM-SOFTWARE'], [cur_image, next_image]):
        ret, local_path_image = download_image_file(startup_info, slave, ip_protocol, vpn_instance,
sha256_val_dic)
        if ret is ERR:
            delete_startup_file(local_path_image, local_path_config, local_path_patch, slave)
            return ERR, local_path_image, local_path_config, local_path_patch
        if ret == DISK_SPACE_NOT_ENOUGH:
            delete_startup_file(local_path_image, None, local_path_patch, slave)
            logging.info('disk space not enough, delete image and patch')
            return OK, None, local_path_config, None
    elif check_contains(startup_info['SYSTEM-SOFTWARE'], [cur_image, next_image]):
        logging.warning('The configured image version is the same as the current device version')

    return OK, local_path_image, local_path_config, local_path_patch

def set_startup_file(image_file, config_file, patch_file, slave):
    """Set startup file"""
    try:
        STARTUP.set_startup_info(image_file, config_file, patch_file, slave)
    except OPIExecError:
```

```
        return ERR

    logging.info('Set startup info ready %s %s %s', format_file_path(image_file),
format_file_path(config_file),
                 format_file_path(patch_file))
    return OK

def format_file_path(file_path):
    """format fiel_path,protect full path"""
    if file_path:
        return os.path.basename(file_path)
    return file_path

def delete_startup_file(image_file, config_file, patch_file, slave):
    """Delete all system file"""
    delete_file_all(image_file, slave)
    delete_file_all(config_file, slave)
    delete_file_all(patch_file, slave)


# ztplib
def set_ztp_last_status(state):
    """Set ztp last status."""
    uri = '/ztpops/ztpStatus/ztpLastStatus'
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <ztpLastStatus>$ztpLastStatus</ztpLastStatus>''')
    req_data = str_temp.substitute(ztpLastStatus=state)
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        logging.error('Failed to set ztp last status to %s', LAST_STATE_MAP[state])
        return

    logging.info('Succeed to set ztp last status to %s', LAST_STATE_MAP[state])


def get_ztp_enable_status():
    """Get ztp enable status
    :raise: OPIExecError
    """
    uri = '/ztpops/ztpStatus/ztpEnable'
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
        <ztpEnable/>'''
    ret, err_code, rsp_data = OPS_CLIENT.get(uri, req_data)
    if ret != HTTP_OK or rsp_data == '':
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to get ztp enable status')

    root_elem = etree.fromstring(rsp_data)
    namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
    uri = 'data' + uri.replace('/', '/vrp:')
    elem = root_elem.find(uri, namespaces)
    if elem is None:
        raise OPIExecError('Failed to read ztp enable status')

    return elem.text


def parse_environment(env):
    lines = re.split(r'\r\n|\r|\n', env)
    for line in lines:
        if re.match('.*ztp_exit_flag.*', line):
            item = re.split(r'[ ][ ]*', line)
            logging.info('parse environment, ztp_exit_flag: ' + item[2])
            return item[2]

    return None
```



```
def get_ztp_exit_environment():
    _ops = ops.ops()
    handle, err_desp = _ops.cli.open()
    ret = _ops.cli.execute(handle, "display ops environment")
    _ops.cli.close(handle)
    if ret[2] == 'Success' and ret[0]:
        return parse_environment(ret[0])

    return None


def check_ztp_continue():
    """Check if ztp can continue to run"""
    res = True
    try:
        enable_state = get_ztp_enable_status()
        ztp_exit_flag = get_ztp_exit_environment()
        if enable_state == 'false' or ztp_exit_flag == 'true':
            res = False
    except OPIExecError as ex:
        logging.warning(ex)

    return res


# DNS
class DNSServer:
    """Dns protocol service"""
    __slots__ = ['dns_servers', 'enable_state', 'vpn_instance']

    def __init__(self):
        self.dns_servers = []
        self.enable_state = 'false'
        self.vpn_instance = {}

    def _set_dns_enable_switch(self, switch):
        """Set DNS global switch."""
        if switch not in ['true', 'false']:
            return

        if self.enable_state == switch:
            logging.info('The current enable state of dns is %s, no need to set', DNS_STATE_MAP.get(switch))
            return

        uri = '/dns/dnsGlobalCfgs/dnsGlobalCfg'
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <dnsGlobalCfg>
                <dnsEnable>$dnsEnable</dnsEnable>
            </dnsGlobalCfg>''')
        req_data = str_temp.substitute(dnsEnable=switch)
        ret, err_code, rsp_data = OPS_CLIENT.set(uri, req_data)
        if ret != HTTP_OK:
            logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
            raise OPIExecError('Failed to %s DNS' % DNS_STATE_MAP.get(switch))

        self.enable_state = switch
        return

    def add_dns_servers_ipv4(self, dns_servers, vpn_instance):
        """Add IPv4 DNS servers configuration.
        :raise: OPIExecError
        """
        while '255.255.255.255' in dns_servers:
            dns_servers.remove('255.255.255.255')

        # only configure new dns servers
        new_dns_servers = list(set(dns_servers).difference(set(self.dns_servers)))
        if not new_dns_servers:
            return
```



```
    self._set_dns_enable_switch('true')
    logging.info('Add DNS IPv4 servers')

    uri = '/dns/dnsIpv4Servers'
    root_elem = etree.Element('dnsIpv4Servers')
    for server_addr in new_dns_servers:
        dns_server = etree.SubElement(root_elem, 'dnsIpv4Server')
        etree.SubElement(dns_server, 'ipv4Addr').text = server_addr
        etree.SubElement(dns_server, 'vrfName').text = vpn_instance
    req_data = etree.tostring(root_elem, 'UTF-8')
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to config DNS IPv4 server')

    # configure success
    self.dns_servers.extend(new_dns_servers)
    self.vpn_instance.update(dict.fromkeys(new_dns_servers, vpn_instance))

def del_dns_servers_ipv4(self):
    """Delete IPv4 DNS servers configuration.
    :raise: OPIExecError
    """
    if not self.dns_servers:
        logging.info('Current dns server is empty, no need to delete')
        return

    logging.info('Delete DNS IPv4 servers')

    uri = '/dns/dnsIpv4Servers'
    root_elem = etree.Element('dnsIpv4Servers')
    for server_addr in self.dns_servers:
        dns_server = etree.SubElement(root_elem, 'dnsIpv4Server')
        etree.SubElement(dns_server, 'ipv4Addr').text = server_addr
        etree.SubElement(dns_server, 'vrfName').text = self.vpn_instance.get(server_addr)
    req_data = etree.tostring(root_elem, 'UTF-8')

    ret, err_code, rsp_data = OPS_CLIENT.delete(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to delete DNS IPv4 server')

    # delete all dns server success
    self.vpn_instance = {}
    self.dns_servers = []

    self._set_dns_enable_switch('false')

@staticmethod
def get_addr_by_hostname(host, vpn_instance, addr_type='1'):
    """Translate a host name to IPv4 address format. The IPv4 address is returned as a string.
    :raise: OPIExecError
    """
    logging.info('Get ipv4 address by host name %s', host)
    uri = '/dns/dnsNameResolution'
    root_elem = etree.Element('dnsNameResolution')
    etree.SubElement(root_elem, 'host').text = host
    etree.SubElement(root_elem, 'addrType').text = addr_type
    etree.SubElement(root_elem, 'vrfName').text = vpn_instance
    req_data = etree.tostring(root_elem, "UTF-8")
    logging.warning(req_data)
    ret, err_code, rsp_data = OPS_CLIENT.get(uri, req_data)
    if ret != HTTP_OK or rsp_data == '':
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to get ipv4 address by host name')

    root_elem = etree.fromstring(rsp_data)
    namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
```



```
        uri = 'data' + uri.replace('/', '/vrp:') + '/vrp:'
        elem = root_elem.find(uri + 'ipv4Addr', namespaces)
        if elem is None:
            logging.error(rsp_data)
            raise OPIExecError('Failed to read IP address by host name')
        formated_ip = convert_ip_format(elem.text)
        logging.info(rsp_data.replace(elem.text, formated_ip))
        return elem.text
def convert_ip_format(ip_addr_str):
    """ip format
      :input:
        x.x.x.x
      :return
         *.*.*.x
    """

    if ip_addr_str and '.' in ip_addr_str:
        last_point = ip_addr_str.rfind('.')
        ip_addr_str = re.sub(r'\w+', '*', ip_addr_str[:last_point + 1]) + ip_addr_str[last_point + 1:]
    return ip_addr_str

# download
def download_file(url, local_path, ip_protocol, vpn_instance):
    """
    Description:
        Download file, support TFTP, FTP, SFTP.
    Args:
      url: URL of remote file
        tftp://hostname/path
        ftp://[username[:password]@]hostname/path
        sftp://[username[:password]@]hostname[:port]/path

      local_path: local path to put the file
        cfcard:/xxx

      ip_protocol: ipv4 or ipv6
      vpn_instance: vpn_instance
    Returns:
        ERR[1]: download fail
        OK[0]:  download success
    """

    url_tuple = urlparse(url)
    func_dict = {
        'tftp': {
            IPV4: TFTPv4,
            IPV6: TFTPv6,
        },
        'ftp': {
            IPV4: FTPv4,
            IPV6: FTPv6,
        },
        'sftp': {
            IPV4: SFTPv4,
            IPV6: SFTPv6,
        }
    }

    scheme = url_tuple.scheme
    if scheme not in func_dict.keys():
        logging.error('Unknown file transfer scheme %s', scheme)
        return ERR

    if ip_protocol == IPV4:
        if not re.match(r'\d+\.\d+\.\d+\.\d+', url_tuple.hostname):
            # get server ip by hostname from dns
            try:
                dns_vpn = '_public_' if vpn_instance in [None, ''] else vpn_instance
                server_ip = DNS.get_addr_by_hostname(url_tuple.hostname, dns_vpn)
```



```
            format_server_ip = convert_ip_format(server_ip)
            logging.info("server ip: " + format_server_ip)
        except OPIExecError as ex:
            logging.error(ex)
            return ERR

        url = url.replace(url_tuple.hostname, server_ip)

    vpn_instance = '' if vpn_instance in [None, '_public_'] else vpn_instance
    logging.info('Start to download file %s using %s', os.path.basename(local_path), scheme)

    ret = ERR
    cnt = 0
    while cnt < 1 + FILE_TRANSFER_RETRY_TIMES:
        if cnt:
            logging.info('Try downloading again, please wait a moment')
        try:
            ret = func_dict[scheme][ip_protocol](url, local_path, vpn_instance).start()
            logging.info("download module return code: [{}]".format(ret))
            if ret in SPACE_NOT_ENOUGH_TAG_LIST:
                ret = DISK_SPACE_NOT_ENOUGH
            if ret in [OK, DISK_SPACE_NOT_ENOUGH]:
                logging.info('download file %s using %s, ret:%d', os.path.basename(local_path), scheme, ret)
                break
            ret = ERR
            logging.error('Failed to download file %s using %s', os.path.basename(local_path), scheme)
            sleep(FILE_DOWNLOAD_INTERVAL_TIME)
        except OPIExecError as ex:
            logging.error(ex)
        except Exception as ex:
            logging.exception(ex)
        cnt += 1
    return ret


class Download:
    """File download base class"""
    def __init__(self, local_path):
        self.local_path = local_path

    def start(self):
        """Start to download file"""
        uri = self.get_uri()
        req_data = self.get_req_data()
        self.pre_download()
        ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data, False)
        if ret != HTTP_OK:
            delete_file_all(self.local_path, False)
            logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
            root = etree.fromstring(rsp_data)
            rpc_error = root.find('rpc-error')
            if rpc_error and rpc_error.find('error-app-tag') is not None:
                ret = int(rpc_error.find('error-app-tag').text)
            else:
                ret = ERR
        else:
            ret = OK
        # If the download time is too short, the key is not created when the key is being deleted.
        sleep(5)
        self.after_download()
        return ret

    def get_uri(self):
        """Return download request uri"""
        raise NotImplementedError

    def get_req_data(self):
        """Return download request xml message"""
```

```
        raise NotImplementedError

    def pre_download(self):
        """Do some actions before download file"""
        raise NotImplementedError

    def after_download(self):
        """Do some actions after download file"""
        raise NotImplementedError


class FTP(Download):
    """FTP download class"""

    def get_uri(self):
        """Return ftp download request uri"""
        return '/ftpc/ftpcTransferFiles/ftpcTransferFile'

    def get_req_data(self):
        """Implemented by subclasses"""
        raise NotImplementedError

    def pre_download(self):
        """FTP not care"""

    def after_download(self):
        """FTP not care"""


class FTPv4(FTP):
    """FTPv4 download class"""

    def __init__(self, url, local_path, vpn_instance):
        self.url = url
        super().__init__(local_path)
        self.vpn_instance = vpn_instance

    def get_req_data(self):
        """Return ftpv4 download request xml message"""
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <ftpcTransferFile>
                <serverIpv4Address>$serverIp</serverIpv4Address>
                <commandType>get</commandType>
                <userName>$username</userName>
                <password>$password</password>
                <localFileName>$localPath</localFileName>
                <remoteFileName>$remotePath</remoteFileName>
                <vpnInstanceName>$vpnInstance</vpnInstanceName>
            </ftpcTransferFile>''')
        url_tuple = urlparse(self.url)
        req_data = str_temp.substitute(serverIp=url_tuple.hostname,
                                       username=url_tuple.username,
                                       password=url_tuple.password,
                                       remotePath=url_tuple.path[1:],
                                       localPath=self.local_path,
                                       vpnInstance=self.vpn_instance)
        return req_data


class FTPv6(FTP):
    """FTPv6 download class"""

    def __init__(self, url, local_path, vpn_instance):
        self.url = url
        super().__init__(local_path)
        self.vpn_instance = vpn_instance

    def get_req_data(self):
        """Return ftpv6 download request xml message"""
```

```
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <ftpcTransferFile>
                <serverIpv6Address>$serverIp</serverIpv6Address>
                <commandType>get</commandType>
                <userName>$username</userName>
                <password>$password</password>
                <localFileName>$localPath</localFileName>
                <remoteFileName>$remotePath</remoteFileName>
                <ipv6VpnName>$vpnInstance</ipv6VpnName>
            </ftpcTransferFile>''')
        url_tuple = urlparse(self.url)
        idx = url_tuple.netloc.rfind('@')
        server_ip = url_tuple.netloc[idx + 1:]
        req_data = str_temp.substitute(serverIp=server_ip,
                                       username=url_tuple.username,
                                       password=url_tuple.password,
                                       remotePath=url_tuple.path[1:],
                                       localPath=self.local_path,
                                       vpnInstance=self.vpn_instance)
        return req_data


class TFTP(Download):
    """TFTP download class"""

    def get_uri(self):
        """Return ftp download request uri"""
        return '/tftpc/tftpcTransferFiles/tftpcTransferFile'

    def get_req_data(self):
        """Implemented by subclasses"""
        raise NotImplementedError

    def pre_download(self):
        """TFTP not case"""

    def after_download(self):
        """TFTP not case"""


class TFTPv4(TFTP):
    """TFTPv4 download class"""

    def __init__(self, url, local_path, vpn_instance):
        self.url = url
        super().__init__(local_path)
        self.vpn_instance = vpn_instance

    def get_req_data(self):
        """Return tftpv4 download request xml message"""
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <tftpcTransferFile>
                <serverIpv4Address>$serverIp</serverIpv4Address>
                <commandType>get_cmd</commandType>
                <localFileName>$localPath</localFileName>
                <remoteFileName>$remotePath</remoteFileName>
                <vpnInstanceName>$vpnInstance</vpnInstanceName>
            </tftpcTransferFile>''')
        url_tuple = urlparse(self.url)
        req_data = str_temp.substitute(serverIp=url_tuple.hostname,
                                       remotePath=url_tuple.path[1:],
                                       localPath=self.local_path,
                                       vpnInstance=self.vpn_instance)
        return req_data


class TFTPv6(TFTP):
    """TFTPv6 download class"""
```



```
    def __init__(self, url, local_path, vpn_instance):
        self.url = url
        super().__init__(local_path)
        self.vpn_instance = vpn_instance

    def get_req_data(self):
        """Return tftpv4 download request xml message"""
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <tftpcTransferFile>
                <serverIpv6Address>$serverIp</serverIpv6Address>
                <commandType>get_cmd</commandType>
                <localFileName>$localPath</localFileName>
                <remoteFileName>$remotePath</remoteFileName>
                <ipv6VpnName>$vpnInstance</ipv6VpnName>
            </tftpcTransferFile>''')
        url_tuple = urlparse(self.url)
        idx = url_tuple.netloc.rfind('@')
        server_ip = url_tuple.netloc[idx + 1:]
        req_data = str_temp.substitute(serverIp=server_ip,
                                       remotePath=url_tuple.path[1:],
                                       localPath=self.local_path,
                                       vpnInstance=self.vpn_instance)
        return req_data


class SFTP(Download):
    """SFTP download class"""

    def get_uri(self):
        """Return ftp download request uri"""
        return '/sshc/sshcConnects/sshcConnect'

    def get_req_data(self):
        """Implemented by subclasses"""
        raise NotImplementedError

    def pre_download(self, ):
        self._set_sshc_first_time('Enable')

    def after_download(self):
        self._del_sshc_rsa_key()
        self._set_sshc_first_time('Disable')

    @classmethod
    def _set_sshc_first_time(cls, switch):
        """Set SSH client attribute of authenticating user for the first time access"""
        if switch not in ['Enable', 'Disable']:
            return ERR

        logging.info('Set SSH client first-time enable switch = %s', switch)
        uri = "/sshc/sshClient"
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <sshClient>
                <firstTimeEnable>$enable</firstTimeEnable>
            </sshClient>''')
        req_data = str_temp.substitute(enable=switch)
        ret, _, _ = OPS_CLIENT.set(uri, req_data)
        if ret != HTTP_OK:
            if switch == 'Enable':
                reason = 'Failed to enable SSH client first-time'
            else:
                reason = 'Failed to disable SSH client first-time'

            raise OPIExecError(reason)

        return OK

    def _del_rsa_peer_key(self):
        """Delete RSA peer key configuration"""
```



```
        logging.info('Delete RSA peer key')
        uri = '/rsa/rsaPeerKeys/rsaPeerKey'
        root_elem = etree.Element('rsaPeerKey')
        etree.SubElement(root_elem, 'keyName').text = self.get_key_name()
        req_data = etree.tostring(root_elem, 'UTF-8')
        ret, _, _ = OPS_CLIENT.delete(uri, req_data)
        if ret != HTTP_OK:
            logging.error('Failed to delete RSA peer key')

    def _del_sshc_rsa_key(self, key_type='RSA'):
        """Delete SSH client RSA key configuration"""
        logging.info('Delete SSH client RSA key')
        uri = '/sshc/sshCliKeyCfgs/sshCliKeyCfg'
        root_elem = etree.Element('sshCliKeyCfg')
        etree.SubElement(root_elem, 'serverName').text = self.get_key_name()
        etree.SubElement(root_elem, 'pubKeyType').text = key_type
        req_data = etree.tostring(root_elem, 'UTF-8')
        ret, _, _ = OPS_CLIENT.delete(uri, req_data)
        if ret != HTTP_OK:
            logging.error('Failed to delete SSH client RSA key')

        self._del_rsa_peer_key()

    def get_key_name(self):
        """Get sftp server ip"""
        raise NotImplementedError


class SFTPv4(SFTP):
    """SFTPv4 download class"""

    def __init__(self, url, local_path, vpn_instance):
        self.url = url
        super().__init__(local_path)
        self.vpn_instance = vpn_instance

    def get_key_name(self):
        url_tuple = urlparse(self.url)
        return url_tuple.hostname

    def get_req_data(self):
        """Return sftpv4 download request xml message"""
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <sshcConnect>
                <HostAddrIPv4>$serverIp</HostAddrIPv4>
                <commandType>get</commandType>
                <userName>$username</userName>
                <password>$password</password>
                <serverPort>$port</serverPort>
                <localFileName>$localPath</localFileName>
                <remoteFileName>$remotePath</remoteFileName>
                <vpnInstanceName>$vpnInstance</vpnInstanceName>
                <transferType>SFTP</transferType>
            </sshcConnect>''')
        url_tuple = urlparse(self.url)
        try:
            if url_tuple.port is None:
                port = 22
            else:
                port = url_tuple.port
        except ValueError:
            port = 22

        logging.info('Sftp download file using port:%s', port)
        req_data = str_temp.substitute(serverIp=url_tuple.hostname,
                                       username=url_tuple.username,
                                       password=url_tuple.password,
                                       port=port,
                                       remotePath=url_tuple.path[1:],
```



```
                                localPath=self.local_path,
                                vpnInstance=self.vpn_instance)
        return req_data


class SFTPv6(SFTP):
    """SFTPv6 download class"""

    def __init__(self, url, local_path, vpn_instance):
        self.url = url
        super().__init__(local_path)
        self.vpn_instance = vpn_instance

    def get_key_name(self):
        url_tuple = urlparse(self.url)
        idx = url_tuple.netloc.find('@')
        return url_tuple.netloc[idx + 1:]

    def get_req_data(self):
        """Return sftpv4 download request xml message"""
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <sshcConnect>
                <HostAddrIPv6>$serverIp</HostAddrIPv6>
                <commandType>get</commandType>
                <userName>$username</userName>
                <password>$password</password>
                <localFileName>$localPath</localFileName>
                <remoteFileName>$remotePath</remoteFileName>
                <ipv6VpnName>$vpnInstance</ipv6VpnName>
                <transferType>SFTP</transferType>
            </sshcConnect>''')
        url_tuple = urlparse(self.url)
        server_ip = self.get_key_name()
        req_data = str_temp.substitute(serverIp=server_ip,
                                username=url_tuple.username,
                                password=url_tuple.password,
                                remotePath=url_tuple.path[1:],
                                localPath=self.local_path,
                                vpnInstance=self.vpn_instance)
        return req_data


def _is_startup_info_valid(startup_info):
    """Does startup info valid
        FILESERVER, SOFTWARE, CONFIG, PATCH, not None
    """
    return startup_info.get('SYSTEM-CONFIG', None) and startup_info.get('FILESERVER', None)


def main_proc(vpn_instance, ip_protocol):
    """
    :param vpn_instance:
    :param ip_protocol:
    :return:
    """
    global REMOTE_PATH_CONFIG

    sys_info = get_system_info()
    slave, _ = has_slave_mpu()  # Check whether slave MPU board exists or not
    logging.info('Get devicetype=%s, esn=%s, mac=%s from the current system',
                 sys_info['productName'],
                 hide_content(sys_info['esn'], 4),
                 hide_content(sys_info['mac'], 4))
    if not REMOTE_PATH_IMAGE.get(sys_info['productName']):
        logging.warning(
            "The product name of the current device [{}] not in
REMOTE_PATH_IMAGE".format(sys_info['productName']))
    if not REMOTE_PATH_PATCH.get(sys_info['productName']):
        logging.warning(
```

```
        "The product name of the current device [{}] not in
REMOTE_PATH_PATCH".format(sys_info['productName']))
    if '%s' in REMOTE_PATH_CONFIG:
        REMOTE_PATH_CONFIG = REMOTE_PATH_CONFIG % sys_info['esn']
    startup_info = {'FILESERVER': FILE_SERVER,
                    'SYSTEM-SOFTWARE': REMOTE_PATH_IMAGE.get(sys_info['productName'], ''),
                    'SYSTEM-CONFIG': REMOTE_PATH_CONFIG,
                    'SYSTEM-PAT': REMOTE_PATH_PATCH.get(sys_info['productName'], '')}
    STARTUP.set_startup_info_from_ini_or_cfg(startup_info)
    if not _is_startup_info_valid(startup_info):
        logging.warning('FILESERVER is None or SYSTEM-CONFIG is None, no need download and '
                        'set system startup file')
        return ERR

    ret = check_filename()
    if ret == ERR:
        return ERR

    # check remote file paths
    try:
        remote_path_sha256 = REMOTE_PATH_SHA256
    except NameError:
        remote_path_sha256 = ''
    if not check_file_type_valid(REMOTE_PATH_IMAGE.get(sys_info['productName'], ''),
REMOTE_PATH_CONFIG,
                                 REMOTE_PATH_PATCH.get(sys_info['productName'], ''), remote_path_sha256):
        return ERR

    ret, image_file, config_file, patch_file = download_startup_file(startup_info, slave,
                                                                     ip_protocol, vpn_instance)
    if ret == ERR:
        logging.info('failed to download file')
        return ERR

    if check_ztp_continue() is False:
        logging.info('user stop ztp before setting, ztp will reset startup')
        delete_startup_file(image_file, config_file, patch_file, slave)
        return ERR

    ret = set_startup_file(image_file, config_file, patch_file, slave)
    if ret == ERR:
        return ERR

    if not check_ztp_continue():
        logging.info('user stop ztp after setting, ztp will reset startup')
        STARTUP.reset_startup_info(slave)
        return ERR

    set_ztp_last_status('true')
    dhcp_stop()
    try:
        reboot_system()
    except OPIExecError as reason:
        logging.error("reboot failed: {}".format(reason))
        set_ztp_last_status('false')
        STARTUP.reset_startup_info(slave)
        return ERR

    return OK


def main(vpn_instance='', ip_protocol=IPV4):
    """The main function of user script. It is called by ZTP frame, so do not remove or change this function.

    Args:
    Raises:
    Returns: user script processing result
    """
    ip_protocol = ip_protocol.lower()
```



```
    try:
        ret = main_proc(vpn_instance, ip_protocol)
    except Exception as reason:
        logging.error(reason)
        trace_info = traceback.format_exc()
        logging.error(trace_info)
        ret = ERR

    finally:
        # Close the OPS connection
        OPS_CLIENT.close()

    return ret


while True:
    try:
        STARTUP = Startup()
        break
    except OPIExecError as ex:
        logging.warning(ex)
    sleep(CHECK_STARTUP_INTERVAL)

DNS = DNSServer()

if __name__ == "__main__":
    main()
```

Python 脚本文件解释

说明

- 示例中加粗的内容表示用户可以修改，请根据实际运行环境进行配置。
- 示例中没有加粗的内容请勿随意修改，否则可能导致ZTP功能不可用。
- 请勿修改脚本逻辑，否则脚本执行过程中可能出现死循环或者执行失败等问题，导致ZTP功能不可用。
- 如果以上示例无法满足要求，请联系华为工程师。

- 指定该脚本文件的SHA256校验码。

  ```
  #sha256sum="68549835edaa5c5780d7b432485ce0d4fdaf6027a8af24f322a91b9f201a5101"
  ```

  用户可以通过该SHA256校验码对下载的脚本文件进行完整性检测。

  用户可以使用以下两种方法生成脚本文件的SHA256校验码：

  a. SHA256计算工具（如HashMyFiles）；

  b. 执行Windows自带的**certutil -hashfile** *filename* SHA256命令。

  说明

  SHA256校验码是由#sha256sum=”行以下内容计算而来，生成SHA256校验码时，需将示例中第一行删除，第二行提至第一行，计算完成后将新生成SHA256校验码“#sha256sum=”写入文件开头。

  文件支持使用SHA256算法进行完整性校验，该种算法安全性较高。

- 指定文件获取方式。

  ```
  FILE_SERVER = 'sftp://username:password@hostname:port/path/'
  ```

  用户可以从TFTP/FTP/SFTP服务器获取版本文件：

  - tftp://*hostname*/*path*/
  - ftp://[*username*[:*password*]@]*hostname*/*path*/
  - sftp://[*username*[:*password*]@]*hostname*[:*port*]/*path*/

其中*username*、*password*、*port*参数为可选项。

- 指定系统软件的文件名。

```
REMOTE_PATH_IMAGE = {
    'NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X': 'V800R023C00SPC500.cc'
}
```

“**NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X**”指设备的型号。

“**V800R023C00SPC500.cc**”为该型号设备获取的系统软件的文件名，可以包含文件路径，如“path1/path2/V800R023C00SPC500.cc”。

如果不需要加载系统软件，可以设置值为空或不写该设备类型，例如：

```
REMOTE_PATH_IMAGE = {
    'NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X' : ''
}
```

或

```
REMOTE_PATH_IMAGE = {}
```

说明

此处填写的设备型号如果与实际设备不符时，将会直接跳过该检查继续执行ZTP流程，即认为不需要设置此项，只记录日志。

- 指定配置文件的文件名。

```
REMOTE_PATH_CONFIG = 'conf_%s.cfg'
```

“%s”表示设备的序列号，不可编辑。设备通过序列号信息获取对应的配置文件。该文件名可以包含文件路径，如“path1/path2/V800R023C00SPC500.cfg”。

说明

- 建议使用esn序列号指定具体设备的配置文件，尽量不要使用不包含esn序列号的配置文件进行批量配置。
- esn序列号区分大小写，必须和设备完全一致。
- 当文件服务器上没有相应的**conf_%s.cfg**时会报配置文件下载失败的问题，例如：设备的esn序列号为2102351HLD10J2000012，当文件服务器上没有**conf_2102351HLD10J2000012.cfg**时就会报错。

- 指定补丁文件的文件名。

```
REMOTE_PATH_PATCH = {
    'NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X': 'V800R023C00SPC500SPH001.PAT'
}
```

“**NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X**”指设备的型号。

“**V800R023C00SPC500SPH001.PAT**”为该型号设备获取的补丁软件的文件名，可以包含文件路径，如“path1/path2/V800R023C00SPC500.PAT”。

如果不需要加载补丁文件，可以设置值为空或不写该设备类型，例如：

```
REMOTE_PATH_PATCH = {
    'NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X' : ''
}
```

或

```
REMOTE_PATH_PATCH = {}
```

- 指定SHA256校验文件的文件名。

```
REMOTE_PATH_SHA256 = 'sha256.txt'
```

用户可以通过SHA256校验文件对设备下载的文件进行完整性检测。

SHA256校验文件格式请参见 **版本文件的完整性校验**。

如果不需要对下载的文件进行校验，可以将该值设置为空：''。

- HTTP消息状态。
  ```
  HTTP_OK = 200
  HTTP_BAD_REQUEST = 400
  HTTP_BAD_RESPONSE = -1
  ```
  此模块不需要用户编辑。
- 定义请求失败后发起二次请求的等待时间。
  ```
  CONFLICT_RETRY_INTERVAL = 5
  ```
- HTTP消息类型。
  ```
  POST_METHOD = 'POST'
  GET_METHOD = 'GET'
  DELETE_METHOD = 'DELETE'
  PUT_METHOD = 'PUT'
  ```
  此模块不需要用户编辑。
- 定义获取启动信息失败时的最大重试次数。
  ```
  MAX_TIMES_GET_STARTUP = 120
  ```
- 定义获取设备启动信息的间隔。
  ```
  GET_STARTUP_INTERVAL = 15
  ```
- 定义单主控设备检查启动项设置失败时的最大重试次数。
  ```
  MAX_TIMES_CHECK_STARTUP = 205
  ```
- 定义双主控设备检查启动项设置失败时的最大重试次数。
  ```
  MAX_TIMES_CHECK_STARTUP_SLAVE = 265
  ```
- 定义检查系统软件是否设置成功的间隔。
  ```
  CHECK_STARTUP_INTERVAL = 5
  ```
- 定义删除文件前的等待时间。
  ```
  FILE_DELETE_DELAY_TIME = 3
  ```
- ZTP状态值映射，用于日志。
  ```
  LAST_STATE_MAP = {'true': 'enable', 'false': 'disable'}
  ```
- DNS状态值映射，用于日志。
  ```
  DNS_STATE_MAP = {'true': 'enable', 'false': 'disable'}
  ```
- 定义下载失败重试次数。
  ```
  FILE_TRANSFER_RETRY_TIMES = 3
  ```
- 定义下载失败后下次开始的等待时间。
  ```
  FILE_DOWNLOAD_INTERVAL_TIME = 5
  ```
- 空间不足状态码。
  ```
  DISK_SPACE_NOT_ENOUGH = 48
  ```
  此模块不需要用户编辑。
- 空间不足状态码列表。
  ```
  SPACE_NOT_ENOUGH_TAG_LIST = [48, 3617]
  ```
  此模块不需要用户编辑。
- PNP终止异常。
  ```
  class PNPStopError()
  ```
  此模块不需要用户编辑。
- OPS进程执行异常。
  ```
  class OPIExecError()
  ```
  此模块不需要用户编辑。
- ZTP不启动异常。
  ```
  class NoNeedZTP2PNPError()
  ```


此模块不需要用户编辑。

- 设备重新启动异常。
  ```
  class SysRebootError()
  ```
  此模块不需要用户编辑。
- ZTP设置关闭异常。
  ```
  class ZTPDisableError()
  ```
  此模块不需要用户编辑。
- 定义OPS连接类。
  ```
  class OPSConnection()
  ```
  此模块不需要用户编辑。
- 封装OPS连接。
  ```
  self.conn = http.client.HTTPConnection()
  ```
  此模块不需要用户编辑。
- 调用平台底层接口。
  ```
  def close()
  def create()
  def delete()
  def get()
  def set()
  ```
  此模块不需要用户编辑。
- 定义请求为REST风格。
  ```
  def _rest_call()
  ```
  此模块不需要用户编辑。
- 关闭DHCP客户端，包括DHCPv4和DHCPv6。
  ```
  def dhcp_stop()
  ```
  此模块不需要用户编辑。
- 获取用户工作目录。
  ```
  def get_cwd()
  ```
  此模块不需要用户编辑。
- 检查需下载的文件是否存在。
  ```
  def file_exist()
  ```
  此模块不需要用户编辑。
- 复制文件。
  ```
  def copy_file()
  ```
  此模块不需要用户编辑。
- 操作失败后删除文件。
  ```
  def delete_file()
  ```
  文件加载失败后，要删去设备上加载失败的文件，使设备回归到启用ZTP前的状态，方便后续操作。

  此模块不需要用户编辑。
- 删除所有主控板上的文件。
  ```
  def delete_file_all()
  ```
  此模块不需要用户编辑。
- 判断设备是否有备用主控板。
  ```
  def has_slave_mpu()
  ```

此模块不需要用户编辑。

- 获取设备的系统信息。

  ```
  def get_system_info()
  ```

  此模块不需要用户编辑。

- 重启系统。

  ```
  def reboot_system()
  ```

  此模块不需要用户编辑。

- 检测参数路径是否有效。

  ```
  def check_file_type_valid()
  ```

  此模块不需要用户编辑。

- 字符串脱敏处理

  ```
  def hide_content()
  ```

  此模块不需要用户编辑。

- 获取设备下次启动信息。

  ```
  def _get_startup_info()
  ```

  此模块不需要用户编辑。

- 设置下次启动的系统软件。

  ```
  def _set_startup_image_file()
  ```

  此模块不需要用户编辑。

- 设置下次启动的配置文件。

  ```
  def _set_startup_config_file()
  ```

  此模块不需要用户编辑。

- 删除下次启动的配置文件。

  ```
  def _del_startup_config_file()
  ```

  此模块不需要用户编辑。

- 设置下次启动的补丁文件。

  ```
  def _set_startup_patch_file()
  ```

  此模块不需要用户编辑。

- 重置下次启动的补丁文件。

  ```
  def _reset_startup_patch_file()
  ```

  此模块不需要用户编辑。

- 检查下次启动文件是否就绪。

  ```
  def _check_next_startup_file()
  ```

  此模块不需要用户编辑。

- 设置下次启动信息。

  ```
  def set_startup_info()
  ```

  此模块不需要用户编辑。

- 重置下次启动信息并且删除已下载的文件。

  ```
  def reset_startup_info()
  ```

  此模块不需要用户编辑。

- 对文件进行SHA256校验。

  ```
  def sha256sum()
  def sha256_get_from_file()
  def sha256_check_with_first_line()
  def sha256_check_with_dic()
  def parse_sha256_file()
  def verify_and_parse_sha256_file()
  ```


此模块不需要用户编辑。

- 检查用户名、密码、文件名是否有特殊字符。
  ```
  def check_parameter()
  def check_filename()
  ```
  此模块不需要用户编辑。
- 下载配置文件。
  ```
  def download_cfg_file()
  ```
  此模块不需要用户编辑。
- 下载补丁文件。
  ```
  def download_patch_file()
  ```
  此模块不需要用户编辑。
- 下载系统软件。
  ```
  def download_image_file()
  ```
  此模块不需要用户编辑。
- 下载下次启动文件。
  ```
  def download_startup_file()
  ```
  此模块不需要用户编辑。
- 设置下次启动文件。
  ```
  def set_startup_file()
  ```
  此模块不需要用户编辑。
- 删除下次启动文件。
  ```
  def delete_startup_file()
  ```
  此模块不需要用户编辑。
- 设置ZTP执行状态。
  ```
  def set_ztp_last_status()
  ```
  此模块不需要用户编辑。
- 获取ZTP的使能状态。
  ```
  def get_ztp_enable_status()
  ```
  此模块不需要用户编辑。
- 解析ZTP执行环境。
  ```
  def parse_environment()
  def get_ztp_exit_environment()
  ```
  此模块不需要用户编辑。
- 检查ZTP流程是否能继续。
  ```
  def check_ztp_continue()
  ```
  此模块不需要用户编辑。
- 设置DNS全局开关。
  ```
  def _set_dns_enable_switch()
  ```
  此模块不需要用户编辑。
- 增加DNS IPv4服务器配置。
  ```
  def add_dns_servers_ipv4()
  ```
  此模块不需要用户编辑。
- 删除DNS IPv4服务器配置。
  ```
  def del_dns_servers_ipv4()
  ```
  此模块不需要用户编辑。
- 解析域名。
  ```
  def get_addr_by_hostname()
  ```


此模块不需要用户编辑。

- 定义文件下载参数。
  ```
  def download_file()
  ```
  此模块不需要用户编辑。
- 开始下载文件。
  ```
  def start()
  ```
  此模块不需要用户编辑。
- 返回请求下载的URI。
  ```
  def get_uri()
  ```
  此模块不需要用户编辑。
- 返回请求下载的XML消息。
  ```
  def get_req_data()
  ```
  此模块不需要用户编辑。
- 定义下载文件前的操作。
  ```
  def pre_download()
  ```
  此模块不需要用户编辑。
- 定义下载文件后的操作。
  ```
  def after_download()
  ```
  此模块不需要用户编辑。
- 设置SSH客户端首次访问认证用户属性。
  ```
  def _set_sshc_first_time()
  ```
  此模块不需要用户编辑。
- 清除RSA密钥。
  ```
  def _del_rsa_peer_key()
  ```
  此模块不需要用户编辑。
- 清除SSH的服务器地址和RSA密钥。
  ```
  def _del_sshc_rsa_key()
  ```
  此模块不需要用户编辑。
- 获取SFTP服务器地址。
  ```
  def get_key_name()
  ```
  此模块不需要用户编辑。
- 定义设备上电自动部署功能的总流程。
  ```
  def main_proc()
  def main()
  if __name__ == "__main__":
      main()
  ```
  此模块不需要用户编辑。

  main函数要求必须有，否则脚本无法运行。

cfg 格式的中间文件

cfg文件是中间文件格式中的一种，用来保存设备及其版本文件信息。

cfg文件的文件名必须以“.cfg”作为后缀名，格式如下:

说明

下列文件中的SHA256校验码只是举例，配置时以实际计算的值为准。

```
#sha256sum="fffcd63f5e31f0891a0349686969969c1ee429dedeaf7726ed304f2d08ce1bc7"
fileserver=sftp://username:password@hostname:port/path/;
mac=00e0-fc12-3456;esn=2102351931P0C3000154;devicetype=DEFAULT;system-
version=V800R023C00SPC500;boot_python_file=V800R023C00SPC500.py;system-
software=V800R023C00SPC500.cc;system-config=V800R023C00SPC500.cfg;system-
pat=V800R023C00SPC500SPH001.PAT;
```

ZTP执行过程中可能存在问题，可以执行如下Python脚本来解决问题，并将BOOT-PYTHON-FILE字段配置的Python脚本放在FILESERVER文件服务器中。

```
#sha256sum="045237bbf3e623cd728b4b8fd4fa9e270838e0d540486e7dcdfdcb8086fd152b"
import logging
import ops

def main():
    try:
        logging.info("*****************")
        logging.info(" run python script start")
        _ops = ops.ops()
        handle, err_desp = _ops.cli.open()
        _ops.cli.execute(handle, "return", None)
        fd, _, err_desp = _ops.cli.execute(handle, "display startup", None)
        logging.info('display startup')
        logging.info(fd)
        fd, _, err_desp = _ops.cli.execute(handle, "system-view", None)
        logging.info('system-view')
        logging.info(fd)
        fd, _, err_desp = _ops.cli.execute(handle, "display interface brief", None)
        logging.info('display interface brief')
        logging.info(fd)
        fd, _, err_desp = _ops.cli.execute(handle, "diagnose", None)
        logging.info('diagnose')
        logging.info(fd)
        fd, _, err_desp = _ops.cli.execute(handle, "disp access um blackbox state", None)
        logging.info('disp access um blackbox state')
        logging.info(fd)
        fd, _, err_desp = _ops.cli.execute(handle, "commit", None)
        logging.info('commit')
        logging.info(fd)
        logging.info(" run python script end")
        logging.info("*****************")
    except Exception as reason:
        logging.error(reason)
    return True

if __name__ == "__main__":
    main()
```


表 1-85 cfg 文件字段含义

字段	是否可选	描述
#sha256sum	必选	指定该脚本文件的SHA256校验码。 **说明** SHA256校验码是由#sha256sum=”行以下内容计算而来，生成SHA256校验码时，需将示例中第一行删除，第二行提至第一行，计算完成后将新生成SHA256校验码“#sha256sum=”写入文件开头。 文件支持使用SHA256算法进行完整性校验，该种算法安全性较高。 用户可以使用以下两种方法生成脚本文件的SHA256校验码： 1. SHA256计算工具（如HashMyFiles）； 2. 执行Windows自带的**certutil -hashfile** *filename* SHA256命令。
fileserver	必选	此字段为版本文件服务器的地址。可以使用TFTP/FTP/SFTP方式获取文件，格式如下： ● tftp://*hostname*/*path*/ ● ftp://[*username*[:*password*]@]*hostname*/*path*/ ● sftp://[*username*[:*password*]@]*hostname*[:*port*]/*path*/ 其中*username*、*password*、*port*参数为可选项。*path*参数指定文件服务器存放版本文件的目录。*hostname*参数为服务器地址，可以是IPv4地址或域名，也可以是IPv6地址。*port*的取值范围为0～65535，超出范围按照默认端口22处理，仅在SFTP服务器地址为IPv4情况下支持配置端口号。
esn	可选	设备序列号。如果esn=DEFAULT，表示不需要匹配esn序列号，否则需要和设备匹配esn。 缺省情况下，esn为DEFAULT。如果该标志不存在或为空，则表示为缺省情况。 **说明** ● 设备的ESN号可以通过设备出厂时外包装上的名牌信息获取。 ● ESN号不区分大小写。 ● 建议使用设备的ESN号指定具体设备的配置信息，尽量不要使用DEFAULT进行批量配置。

字段	是否可选	描述
mac	可选	设备mac地址，格式为：XXXX-XXXX-XXXX，X为十六进制数。如果mac=DEFAULT，表示不需要匹配MAC地址，否则需要和设备匹配MAC地址。 优先匹配esn，再匹配mac。 缺省情况下，mac为DEFAULT。如果该标志不存在或为空，则表示为缺省情况。 **说明** • 设备的MAC地址可以通过设备出厂时外包装上的名牌信息获取。 • MAC地址不区分大小写。 • 用户需严格按照设备上显示的MAC地址格式填写中间文件，例如：设备显示为00e0-fc12-3456，则00e0fc123456是错误填写，“-”也会进行校验。 • 建议使用设备的MAC地址指定具体设备的配置信息，尽量不要使用DEFAULT进行批量配置。
devicetype	可选	匹配的设备类型。如果devicetype=DEFAULT，表示不匹配设备类型，否则需要匹配设备类型。 缺省情况下，devicetype为DEFAULT。如果该标志不存在或为空，则表示为缺省情况。 **说明** • 具体的设备类型请参见“描述》硬件描述》机框”章节。 • devicetype与实际设备不符时，将重新执行ZTP流程。
system-version	可选	系统版本号，支持匹配到C版本，例如V800R023C00SPC500。
system-software	可选	系统软件名称，后缀名为“.cc”，可以包含文件路径，如：“path1/path2/V800R023C00SPC500.cc”。
system-config	必选	配置文件名称，后缀名为“.cfg”、“.zip”或“.dat”，可以包含文件路径，如：“path1/path2/V800R023C00SPC500.cfg”。 **说明** 配置文件名称不要使用设备缺省的名称vrpcfg.zip。
system-pat	可选	补丁文件名称，后缀名为“.PAT”，可以包含文件路径，如：“path1/path2/V800R023C00SPC500.PAT”。

字段	是否可选	描述
BOOT-PYTHON-FILE	可选	ZTP执行过程中需要执行的python脚本文件，用来解决一些ZTP执行过程中可能遇到的问题，请将BOOT-PYTHON-FILE字段配置的Python脚本放在FILESERVER文件服务器中。 **说明** BOOT-PYTHON-FILE配置的Python脚本文件名称不要与ZTP预配置文件重名，名称不可包含非法字符，如果文件中包含使ZTP异常退出的命令行，例如**set ztp disable**，**undo pnp enable**，ops视图disable等，可能会导致ZTP退出，影响ZTP主流程。

说明

- 设备按照配置行在cfg文件中从前到后的顺序进行匹配。
- 当devicetype匹配不成功，则认为改行配置无效，继续匹配下一行。
- 当devicetype不需要匹配（devicetype=DEFAULT）或匹配成功时，需要匹配esn或mac，其中有一个匹配成功，则认为该条行配置有效，否则认为该条行配置无效。如果esn和mac全部为DEFAULT时，同样认为该行配置有效。
- 如果中间文件中包含版本号信息，则必须要包含系统软件名称，并且要求系统软件的版本号与中间文件的版本号信息一致。

版本文件的完整性校验

用户可以通过SHA256校验文件对设备下载的文件进行完整性检测。被下载文件的SHA256校验码预先保存在SHA256校验文件中，当设备下载了文件后，会生成该文件的SHA256校验码并与SHA256校验文件中的进行比较。如果不一致则表示文件不完整，设备不会加载该文件。

SHA256校验文件为文本格式，必须以“.txt”作为后缀名，格式如下：

说明

用户可以使用以下两种方法生成脚本文件的SHA256校验码：

1. SHA256计算工具（如HashMyFiles）；
2. 执行Windows自带的**certutil -hashfile** *filename* SHA256命令。

下列文件中的SHA256校验码只是举例，配置时以实际计算的值为准。

文件支持使用SHA256算法进行完整性校验，该种算法安全性较高。

```
#sha256sum="29d29a2b0ef2136f0f192667d71627020e58438fbfb87323f2dae27b5cd9a797"

file-name                sha256
conf_5618642831132.cfg   319c16ebcbc987ef11f28f78cb7d6e7ea4950b8b195e1388c031f3327cc2666e
```

表 1-86 SHA256 校验文件字段含义

字段	是否可选	描述
#sha256sum	必选	该SHA256校验文件本身的校验码。

字段	是否可选	描述
file-name	必选	文件名。
sha256	必选	对应文件的SHA256校验码。

导致 ZTP 退出或失败的条件

如表1-87所示，这些条件会导致设备退出ZTP流程或ZTP功能失效。

表 1-87 导致 ZTP 退出或失败的条件列表

条件	影响性
配置了**set ztp disable**命令。	ZTP使能状态关闭，如果需要再次使用ZTP功能，需要执行**set ztp enable**命令。
设备上出现如下任意配置： • 在任意接口下配置IP地址，不包括管理网口地址（192.168.0.1）、Loopback口地址以及DCN相关子接口IP地址等。 • 全局配置VLAN。 • 全局配置BD。 • 全局配置VSI。 • 设备设置为AP角色。 • 接口下配置DHCP Client。	ZTP退出。
设备通过DCN登录成功。	ZTP退出。

自动补丁修复机制

在ZTP开局自动部署过程中，出现非环境问题（环境问题如设备初始不稳定、服务器未启动等）时，不需要工程师前往站点维修或设备返厂维修，ZTP支持自动补丁修复机制。通过人工定位问题根因，联系工程师制作修复补丁，ZTP运行过程中可以自主识别、设置修复补丁，并重启设备使修复补丁生效。设备重启后ZTP会再次运行，由于此时问题已被修复，ZTP可以顺利完成开局部署。

说明

- 修复补丁默认不会最终保留。如果ZTP中间文件中未设置补丁，则ZTP部署完成后最终留在设备上的补丁为部署前设备上的自带补丁；如果ZTP中间文件中设置了补丁，则ZTP部署完成后最终留在设备上的补丁为中间文件中的补丁。
- 如果需要最终保留修复补丁，可以将ZTP原有补丁配置项也配置为ZTP修复补丁。

ini 格式中间文件中修复补丁配置示例

```
#sha256sum="88298f97c634cb04b1eb4fe9ad2255abffc0a246112e1960cb6402f6b799f8b6"
;BEGIN ROUTER
[GLOBAL CONFIG]
```



```
FILESERVER=sftp://username:password@hostname:port/path/

[DEVICEn DESCRIPTION]
ESN=2102351931P0C3000154
MAC=00e0-fc12-3456
DEVICETYPE=DEFAULT
BOOT-PYTHON-FILE=V800R023C00SPC500.py
SYSTEM-SOFTWARE=V800R023C00SPC500.cc
SYSTEM-CONFIG=V800R023C00SPC500.cfg
SYSTEM-PAT=V800R023C00SPC500SPH001.PAT
SYSTEM-FIX-PAT=V800R023C00SPC500.PAT
;END ROUTER
```

Python 格式中间文件中修复补丁配置示例

```
#sha256sum="68549835edaa5c5780d7b432485ce0d4fdaf6027a8af24f322a91b9f201a5101"
#!/usr/bin/env python
# coding=utf-8
#
# Copyright (C) Huawei Technologies Co., Ltd. 2008-2013. All rights reserved.
# ------------------------------------------------------------------------------------------------------------------------
# History:
# Date              Author                  Modification
# 20180122          Author                  created file.
# ------------------------------------------------------------------------------------------------------------------------

"""
Zero Touch Provisioning (ZTP) enables devices to automatically load version files including system software,
patch files, configuration files when the device starts up, the devices to be configured must be new devices
or have no configuration files.

This is a sample of Zero Touch Provisioning user script. You can customize it to meet the requirements of
your network environment.
"""

import hashlib
import http.client
import logging
import os
import re
import string
import traceback
import xml.etree.ElementTree as etree
from time import sleep
from urllib.parse import urlparse

import ops


# error code
OK = 0
ERR = 1

# File server in which stores the necessary system software, configuration and patch files:
#   1) Specify the file server which supports the following format.
#      tftp://hostname/path
#      ftp://[username[:password]@]hostname/path
#      sftp://[username[:password]@]hostname[:port]/path
#   2) Do not add a trailing slash at the end of file server path.
FILE_SERVER = 'sftp://username:password@hostname:port/path/'

# Remote file paths:
#   1) The path may include directory name and file name.
#   2) If file name is not specified, indicate the procedure can be skipped.
#   3) If you do not want image, please set it as REMOTE_PATH_IMAGE = {} or REMOTE_PATH_IMAGE =
{'DEVICETYPE': ''}
# File paths of system software on file server, filename extension is '.cc'.
REMOTE_PATH_IMAGE = {
    'NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X': 'V800R023C00SPC500.cc'
```

```
}
# File path of configuration file on file server, filename extension is '.cfg', '.zip' or '.dat'.
REMOTE_PATH_CONFIG = 'conf_%s.cfg'
# If you do not want patch, please set it as REMOTE_PATH_PATCH = {} or REMOTE_PATH_PATCH =
{'DEVICETYPE': ''}
# File path of patch file on file server, filename extension is '.pat'
REMOTE_PATH_PATCH = {
    'NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X': 'V800R023C00SPC500SPH001.pat'
}
# File path of fix-patch file on file server, filename extension is '.pat'
# If you don't want fix-patch, or there are not errors in ztp process, set it
# as REMOTE_PATH_FIX_PATCH = {} or REMOTE_PATH_FIX_PATCH = {'DEVICETYPE': ''}.
REMOTE_PATH_FIX_PATCH = {
   'NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E X':  ‘V800R023C00SPC500.pat'
}

# File path of sha256 file, contains sha256 value of image / patch / configuration, file extension is '.txt'
# If you do not want sha256-file,please set it as REMOTE_PATH_SHA256 = ''
REMOTE_PATH_SHA256 = 'sha256.txt'

# constant
# autoconfig
HTTP_OK = 200
HTTP_BAD_REQUEST = 400
HTTP_BAD_RESPONSE = -1
CONFLICT_RETRY_INTERVAL = 5

POST_METHOD = 'POST'
GET_METHOD = 'GET'
DELETE_METHOD = 'DELETE'
PUT_METHOD = 'PUT'

MAX_TIMES_GET_STARTUP = 120
GET_STARTUP_INTERVAL = 15

MAX_TIMES_GET_STARTUP = 205
MAX_TIMES_CHECK_STARTUP_SLAVE = 265
CHECK_STARTUP_INTERVAL = 5

FILE_DELETE_DELAY_TIME = 3

# ztplib
INSTALL_CHECK_TIMES = 3
PRE_CONFIG_RETRY_TIMES = 12
NOT_START_PNP = 2
ZTP_SLEEP_INTERVAL = 3
LAST_STATE_MAP = {'true': 'enable', 'false': 'disable'}

# DNS
DNS_STATE_MAP = {'true': 'enable', 'false': 'disable'}

# download
FILE_TRANSFER_RETRY_TIMES = 3
FILE_DOWNLOAD_INTERVAL_TIME = 5

DISK_SPACE_NOT_ENOUGH = 48
SPACE_NOT_ENOUGH_TAG_LIST = [48, 3617]

IPV4 = 'ipv4'
IPV6 = 'ipv6'

OPS_CLIENT = None


# exception
class PNPStopError(Exception):
    """Stop by pnp"""
```



```
class OPIExecError(Exception):
    """OPS Connection Exception"""


class NoNeedZTP2PNPError(Exception):
    """No need start ztp"""


class SysRebootError(Exception):
    """Device reboot error"""


class ZTPDisableError(Exception):
    """ZTP set disable error"""


# opslib
class OPSConnection:
    """Make an OPS connection instance."""
    __slots__ = ['host', 'port', 'headers', 'conn']

    def __init__(self, host, port=80):
        self.host = host
        self.port = port
        self.headers = {
            'Content-type': 'application/xml',
            'Accept': 'application/xml'
        }

        self.conn = http.client.HTTPConnection(self.host, self.port)

    def close(self):
        """Close the connection"""
        self.conn.close()

    def create(self, uri, req_data, need_retry=True):
        """Create a resource on the server"""
        ret = self._rest_call(POST_METHOD, uri, req_data)
        if ret[0] != HTTP_OK and need_retry:
            sleep(CONFLICT_RETRY_INTERVAL)
            ret = self._rest_call(POST_METHOD, uri, req_data)
        return ret

    def delete(self, uri, req_data, need_retry=True):
        """Delete a resource on the server"""
        ret = self._rest_call(DELETE_METHOD, uri, req_data)
        if ret[0] != HTTP_OK and need_retry:
            sleep(CONFLICT_RETRY_INTERVAL)
            ret = self._rest_call(DELETE_METHOD, uri, req_data)
        return ret

    def get(self, uri, req_data=None, need_retry=True):
        """Retrieve a resource from the server"""
        ret = self._rest_call(GET_METHOD, uri, req_data)
        if (ret[0] != HTTP_OK or ret[2] == '') and need_retry:
            sleep(CONFLICT_RETRY_INTERVAL)
            ret = self._rest_call(GET_METHOD, uri, req_data)
        return ret

    def set(self, uri, req_data, need_retry=True):
        """Update a resource on the server"""
        ret = self._rest_call(PUT_METHOD, uri, req_data)
        if ret[0] != HTTP_OK and need_retry:
            sleep(CONFLICT_RETRY_INTERVAL)
            ret = self._rest_call(PUT_METHOD, uri, req_data)
        return ret

    def _rest_call(self, method, uri, req_data):
        """REST call"""
```



```
        body = '' if req_data is None else req_data

        try:
            self.conn.request(method, uri, body, self.headers)
        except http.client.CannotSendRequest:
            logging.warning('An error occurred during http request, try to send request again')
            self.close()
            self.conn = http.client.HTTPConnection(self.host, self.port)
            self.conn.request(method, uri, body, self.headers)
        except http.client.InvalidURL:
            logging.warning('Failed to find url: %s in OPS whitelist', uri)
            return HTTP_BAD_REQUEST, '', ''

        try:
            response = self.conn.getresponse()
        except AttributeError:
            logging.warning('An error occurred during http response, try again')
            return HTTP_BAD_RESPONSE, '', ''

        rest_message = response.read()
        if isinstance(rest_message, bytes):
            rest_message = str(rest_message, 'iso-8859-1')
        # logging.debug('uri = %s ret = %s \n %s \n %s', uri, response.status, req_data, rest_message)

        ret = (response.status, response.reason, rest_message)
        return ret


OPS_CLIENT = OPSConnection("localhost")


# pnplib
def dhcp_stop():
    """Stop DHCP client, include dhcpv4 and dhcpv6."""
    logging.info('Stopping dhcp client')

    uri = '/pnp/stopPnp'
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
        <stopPnp/>'''
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        # ignore stop pnp err
        logging.warning('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        logging.warning('Failed to stop dhcp client')
        return

    logging.info('DHCP client has stopped')
    return


# commlib
def get_cwd():
    """Get the full filename of the current working directory"""
    logging.info("Get the current working directory...")
    namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
    uri = "/vfm/pwds/pwd"
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
                    <pwd>
                        <dictionaryName/>
                    </pwd>
                '''
    ret, _, rsp_data = OPS_CLIENT.get(uri, req_data)
    if ret != http.client.OK or rsp_data is '':
        raise OPIExecError('Failed to get the current working directory')

    root_elem = etree.fromstring(rsp_data)
    uri = 'data' + uri.replace('/', '/vrp:') + '/vrp:dictionaryName'
    elem = root_elem.find(uri, namespaces)
    if elem is None:
```



```
        raise OPIExecError('Failed to get the current working directory for no "directoryName" element')

    return elem.text


def file_exist(file_name, dir_path=None):
    """Returns True if file_path refers to an existing file, otherwise returns False"""
    uri = '/vfm/dirs/dir'
    str_temp_1 = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <dir>
            <fileName>$fileName</fileName>
        </dir>''')
    str_temp_2 = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <dir>
            <dirName>$dirName</dirName>
            <fileName>$fileName</fileName>
        </dir>''')

    if dir_path:
        req_data = str_temp_2.substitute(dirName=dir_path, fileName=file_name)
    else:
        req_data = str_temp_1.substitute(fileName=file_name)
    ret, _, rsp_data = OPS_CLIENT.get(uri, req_data)
    if ret != HTTP_OK or rsp_data == '':
        return False

    root_elem = etree.fromstring(rsp_data)
    namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
    uri = 'data' + uri.replace('/', '/vrp:') + '/vrp:fileName'
    elem = root_elem.find(uri, namespaces)
    if elem is None:
        return False

    return True


def copy_file(src_path, dest_path):
    """Copy a file"""
    logging.info('Copy file %s to %s', os.path.basename(src_path), os.path.basename(dest_path))

    if 'slave' in dest_path:
        file_name = dest_path.split(':/')[1]
        if file_exist(file_name, 'slave#cfcard:/'):
            logging.info('Detect dest file exist, delete it first')
            delete_file(dest_path)

    uri = '/vfm/copyFile'
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <copyFile>
            <srcFileName>$src</srcFileName>
            <desFileName>$dest</desFileName>
        </copyFile>''')
    req_data = str_temp.substitute(src=src_path, dest=dest_path)

    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data, False)
    if ret != HTTP_OK:
        file_name = dest_path.split(':/')[1]
        if file_exist(file_name, "slave#cfcard:/"):
            logging.info('Exists file copy fragment, delete it')
            delete_file(dest_path)
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        logging.error('Failed to copy %s to %s', os.path.basename(src_path), os.path.basename(dest_path))
        return False

    logging.info('succeed to copy')
    return True


def delete_file(file_path):
```

```
    """Delete a file permanently"""
    if file_path is None or file_path == '':
        return

    logging.info('Delete file %s permanently', os.path.basename(file_path))
    uri = '/vfm/deleteFileUnRes'
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <deleteFileUnRes>
            <fileName>$filePath</fileName>
        </deleteFileUnRes>
        ''')
    req_data = str_temp.substitute(filePath=file_path)
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        logging.error('Failed to delete the file %s permanently', os.path.basename(file_path))


def delete_file_all(file_path, slave, protect_file_list=None):
    """Delete a file permanently on all main boards"""
    if not file_path:
        return
    if protect_file_list:
        for protect_file in protect_file_list:
            if file_path == protect_file:
                return
    file_name = os.path.basename(file_path)
    file_path_t = file_path[:len(file_path) - len(file_name)]
    if file_exist(file_name, file_path_t):
        delete_file(file_path)
    if slave and file_exist(file_name, 'slave#' + file_path_t):
        delete_file('slave#' + file_path)


def has_slave_mpu():
    """Whether device has slave MPU, returns a bool value
    :raise OPIExecError
    """
    logging.info("Test whether device has slave MPU")
    uri = '/devm/phyEntitys'
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
        <phyEntitys>
            <phyEntity>
                <entClass>mpuModule</entClass>
                <entStandbyState/>
                <position/>
            </phyEntity>
        </phyEntitys>'''

    has_slave = False
    mpu_slot = {}.fromkeys(('master', 'slave'))
    ret, err_code, rsp_data = OPS_CLIENT.get(uri, req_data)
    if ret != HTTP_OK or rsp_data == '':
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to get the device slave information')

    root_elem = etree.fromstring(rsp_data)
    namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
    uri = 'data{0}/vrp:phyEntity'.format(uri.replace('/', '/vrp:'))
    for entity in root_elem.findall(uri, namespaces):
        elem = entity.find("vrp:entStandbyState", namespaces)
        if elem is not None and elem.text.lower().find('slave') >= 0:
            has_slave = True
            elem = entity.find("vrp:position", namespaces)
            if elem is not None:
                mpu_slot['slave'] = elem.text
        if elem is not None and elem.text.lower().find('master') >= 0:
            elem = entity.find("vrp:position", namespaces)
            if elem is not None:
```



```
                mpu_slot['master'] = elem.text

    logging.info('Device has slave: %s', has_slave)
    return has_slave, mpu_slot


def get_system_info():
    """Get device product esn mac
    :raise: OPIExecError
    """
    logging.info("Get the system information...")
    uri = "/system/systemInfo"
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
    <systemInfo>
        <productName/>
        <esn/>
        <mac/>
    </systemInfo>
    '''

    sys_info = {}.fromkeys(('productName', 'esn', 'mac'), '')
    ret, err_code, rsp_data = OPS_CLIENT.get(uri, req_data)
    if ret != HTTP_OK or rsp_data == '':
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to get the system information')

    root_elem = etree.fromstring(rsp_data)
    namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
    uri = 'data' + uri.replace('/', '/vrp:')
    nslen = len(namespaces['vrp'])
    elem = root_elem.find(uri, namespaces)
    if elem is not None:
        for child in elem:
            tag = child.tag[nslen + 2:]
            if tag in list(sys_info.keys()):
                sys_info[tag] = child.text

    return sys_info


def reboot_system(save_config='false'):
    """Reboot system."""
    logging.info('System will reboot to make the configuration take effect')

    if save_config not in ['true', 'false']:
        return

    logging.info("start to issue reboot command")
    sleep(10)
    uri = "/devm/reboot"
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <reboot>
            <saveConfig>$saveConfig</saveConfig>
        </reboot>''')
    req_data = str_temp.substitute(saveConfig=save_config)
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK or rsp_data == '':
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to execute the reboot system operation')


def check_file_type_valid(image, config, patch, sha256_file):
    """Test whether argument paths are valid."""
    logging.info("Test whether argument paths are valid...")
    # check image file path
    file_name = os.path.basename(image)
    if file_name is not '' and not file_name.lower().endswith('.cc'):
        logging.error('Error: Invalid filename extension of system software')
        return False
```

```
    # check config file path
    file_name = os.path.basename(config)
    file_name = file_name.lower()
    _, ext = os.path.splitext(file_name)
    if file_name is not '' and ext not in ['.cfg', '.zip', '.dat']:
        logging.error('Error: Invalid filename extension of configuration file')
        return False

    # check patch file path
    file_name = os.path.basename(patch)
    if file_name is not '' and not file_name.lower().endswith('.pat'):
        logging.error('Error: Invalid filename extension of patch file')
        return False

    # check sha256 file path
    file_name = os.path.basename(sha256_file)
    if file_name is not '' and not file_name.lower().endswith('.txt'):
        logging.error('Error: Invalid filename extension of %s file', sha256_file)
        return False

    return True

def hide_content(content: str, least_length: int) -> str:
    if not content:
        return str(content)

    content = str(content)
    if least_length > len(content):
        return content
    hide = ""
    for _ in content[:-least_length]:
        hide += "*"
    return hide + content[-least_length:]

# startuplib
class StartupInfo:
    """Startup configuration information

    image: startup system software
    config: startup saved-configuration file
    patch: startup patch package
    """

    def __init__(self, image=None, config=None, patch=None):
        self.image = image
        self.config = config
        self.patch = patch


class Startup:
    """Startup configuration information

    current: current startup configuration
    next: current next startup configuration
    """

    def __init__(self):
        self.current, self.next = self._get_startup_info()
        self.startup_info_from_ini_or_cfg = {}
        self.startup_info_before_set = StartupInfo()

    @staticmethod
    def _get_startup_info(retry=True):
        """Get device startup information
            :raise
                opslib.OPIExecError
        """
        uri = '/cfg/startupInfos/startupInfo'
```



```
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
        <startupInfo>
            <position/>
            <configedSysSoft/>
            <curSysSoft/>
            <nextSysSoft/>
            <curStartupFile/>
            <nextStartupFile/>
            <curPatchFile/>
            <nextPatchFile/>
        </startupInfo>'''

    if retry is True:
        retry_time = MAX_TIMES_GET_STARTUP
    else:
        retry_time = 1

    cnt = 0
    elem = None
    namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
    ns_len = len(namespaces['vrp'])
    path = 'data' + uri.replace('/', '/vrp:')  # match path
    while cnt < retry_time:
        ret, _, rsp_data = OPS_CLIENT.get(uri, req_data)
        if ret != HTTP_OK or rsp_data == '':
            cnt += 1
            logging.warning('Failed to get the startup information')
            # sleep to wait for system ready when no query result
            sleep(GET_STARTUP_INTERVAL)
            continue

        root_elem = etree.fromstring(rsp_data)
        elem = root_elem.find(path, namespaces)
        if elem is not None:
            break
        logging.warning('No query result while getting startup info')
        # sleep to wait for system ready when no query result
        sleep(GET_STARTUP_INTERVAL)
        cnt += 1

    if elem is None:
        raise OPIExecError('Failed to get the startup information')

    current = StartupInfo()  # current startup info
    curnext = StartupInfo()  # next startup info
    for child in elem:
        # skip the namespace, '{namespace}text'
        tag = child.tag[ns_len + 2:]
        if tag == 'curSysSoft':
            current.image = child.text
        elif tag == 'nextSysSoft':
            curnext.image = child.text
        elif tag == 'curStartupFile' and child.text != 'NULL':
            current.config = child.text
        elif tag == 'nextStartupFile' and child.text != 'NULL':
            curnext.config = child.text
        elif tag == 'curPatchFile' and child.text != 'NULL':
            current.patch = child.text
        elif tag == 'nextPatchFile' and child.text != 'NULL':
            curnext.patch = child.text
        else:
            continue

    return current, curnext

@staticmethod
def _set_startup_image_file(file_path, slave=True):
    """Set the next startup system software"""
    file_name = os.path.basename(file_path)
```

```
    logging.info('Set the next startup system software to %s, please wait a moment', file_name)
    uri = '/sum/startupbymode'
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <startupbymode>
            <softwareName>$fileName</softwareName>
            <mode>$startupMode</mode>
        </startupbymode>''')

    if slave:
        startup_mode = 'STARTUP_MODE_ALL'
    else:
        startup_mode = 'STARTUP_MODE_PRIMARY'

    req_data = str_temp.substitute(fileName=file_name, startupMode=startup_mode)
    # it is a action operation, so use create for HTTP POST
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to set startup system software')

@staticmethod
def _set_startup_config_file(file_path):
    """Set the next startup saved-configuration file"""
    file_name = os.path.basename(file_path)
    logging.info('Set the next startup saved-configuration file to %s', file_name)
    uri = '/cfg/setStartup'
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <setStartup>
            <fileName>$fileName</fileName>
        </setStartup>''')

    req_data = str_temp.substitute(fileName=file_name)
    # it is a action operation, so use create for HTTP POST
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to set startup configuration file')

@staticmethod
def _del_startup_config_file():
    """Delete startup config file"""
    logging.info('Delete the next startup config file')
    uri = '/cfg/clearStartup'
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
        <clearStartup>
        </clearStartup>'''
    # it is a action operation, so use create for HTTP POST
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to delete startup configuration file')

@staticmethod
def _set_startup_patch_file(file_path):
    """Set the next startup patch file"""
    file_name = os.path.basename(file_path)
    logging.info('Set the next startup patch file to %s', file_name)
    uri = "/patch/startup"
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <startup>
            <packageName>$fileName</packageName>
        </startup>''')
    req_data = str_temp.substitute(fileName=file_name)
    # it is a action operation, so use create for HTTP POST
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to set startup patch file')
```



```
@staticmethod
def _reset_startup_patch_file():
    """Reset patch file for system to startup"""
    logging.info('Reset the next startup patch file')
    uri = '/patch/resetpatch'
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
        <resetpatch/>'''
    # it is a action operation, so use create for HTTP POST
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to reset startup patch file')

def _check_next_startup_file(self, file_name, check_item, slave):
    """Check next startup file ready
        check_item: [image, config, patch]
    """
    if check_item not in ['image', 'config', 'patch']:
        return True

    logging.info('Check the next startup %s information', check_item)
    if slave:
        check_time = MAX_TIMES_CHECK_STARTUP_SLAVE
    else:
        check_time = MAX_TIMES_CHECK_STARTUP
    cnt = 0
    while cnt < check_time:
        _, next_startup = self._get_startup_info()
        startup_file_name = getattr(next_startup, check_item)
        if startup_file_name == file_name:
            sleep(CHECK_STARTUP_INTERVAL)
            logging.info('The next system %s check successfully', check_item)
            return True
        # sleep to wait for system ready when no query result
        sleep(CHECK_STARTUP_INTERVAL)
        if cnt % 12 == 0:
            # logging every minute
            logging.info('Checking the next startup %s, please wait a moment', check_item)
        cnt += 1
    logging.warning('The next system %s is not ready', check_item)
    return False

def set_startup_info(self, image_file, config_file, patch_file, slave):
    """Set the next startup information."""
    # backup startup_info set by user
    cur_startup, cur_next_startup = self._get_startup_info()
    self.startup_info_before_set.image = cur_next_startup.image
    self.startup_info_before_set.patch = cur_next_startup.patch
    self.startup_info_before_set.config = cur_next_startup.config
    logging.info("save startup config before ztp setting")

    logging.info('Start to set next startup information')
    # 1. Set next startup system software
    if image_file is not None:
        try:
            self._set_startup_image_file(image_file)
            if self._check_next_startup_file(image_file, 'image', slave) is False:
                raise OPIExecError('Failed to check the next startup image file')
        except OPIExecError as reason:
            logging.error(reason)
            delete_file_all(image_file, slave, [cur_startup.image, cur_next_startup.image])
            self.reset_startup_info(slave)
            raise

    # 2. Set next startup patch file
    if patch_file is not None:
        try:
            self._set_startup_patch_file(patch_file)
            if self._check_next_startup_file(patch_file, 'patch', slave) is False:
```

```
                raise OPIExecError('Failed to check the next startup patch file')
            except OPIExecError as reason:
                logging.error(reason)
                delete_file_all(patch_file, slave, [cur_startup.patch, cur_next_startup.patch])
                self.reset_startup_info(slave)
                raise

        # 3. Set next startup config file
        if config_file is not None:
            try:
                self._set_startup_config_file(config_file)
                if self._check_next_startup_file(config_file, 'config', slave) is False:
                    raise OPIExecError('Failed to check the next startup config file')
            except OPIExecError as reason:
                logging.error(reason)
                delete_file_all(config_file, slave, [cur_startup.config, cur_next_startup.config])
                self.reset_startup_info(slave)
                raise

    def reset_startup_info(self, slave):
        """Reset startup info and delete the downloaded files"""
        logging.info('Start to reset next startup information')
        if not self.startup_info_before_set.image:
            logging.error('image of roll back point is None')
            return
        cur_startup, next_startup = self._get_startup_info()

        # 1. Reset next startup config file and delete it
        try:
            # user configure startup info after ZTP
            if next_startup.config != self.startup_info_from_ini_or_cfg.get("SYSTEM-CONFIG"):
                logging.info("no need to reset startup config")
                if self.startup_info_from_ini_or_cfg.get("SYSTEM-CONFIG"):
                    sleep(FILE_DELETE_DELAY_TIME)
                    delete_file_all(self.startup_info_from_ini_or_cfg.get("SYSTEM-CONFIG"), slave,
                                    [cur_startup.config, next_startup.config])
            # user do not configure startup info
            elif next_startup.config != self.startup_info_before_set.config:
                logging.info("reset startup config to the beginning")
                if self.startup_info_before_set.config is None:
                    self._del_startup_config_file()
                else:
                    self._set_startup_config_file(self.startup_info_before_set.config)
                    if self._check_next_startup_file(self.startup_info_before_set.config, 'config', slave) is not True:
                        raise OPIExecError('Failed to check the next startup config file')
                if next_startup.config:
                    sleep(FILE_DELETE_DELAY_TIME)
                    delete_file_all(next_startup.config, slave,
                                    [cur_startup.config, self.startup_info_before_set.config])
        except Exception as reason:
            logging.error(reason)

        # 2. Reset next startup patch file and delete it
        try:
            # user configure startup info after ZTP
            if next_startup.patch != self.startup_info_from_ini_or_cfg.get("SYSTEM-PAT"):
                logging.info("no need to reset startup patch")
                if self.startup_info_from_ini_or_cfg.get("SYSTEM-PAT"):
                    sleep(FILE_DELETE_DELAY_TIME)
                    delete_file_all(self.startup_info_from_ini_or_cfg.get("SYSTEM-PAT"), slave,
                                    [cur_startup.patch, next_startup.patch])
            # user do not configure startup info
            elif next_startup.patch != self.startup_info_before_set.patch:
                logging.info("reset startup patch to the beginning")
                if self.startup_info_before_set.patch is None:
                    self._reset_startup_patch_file()
                    if STARTUP._check_next_startup_file(None, 'patch', slave) is not True:
                        raise opslib.OPIExecError('Failed to reset the next startup patch file')
                else:
```

```
                self._set_startup_patch_file(self.startup_info_before_set.patch)
                if self._check_next_startup_file(self.startup_info_before_set.patch, 'patch', slave) is not True:
                    raise OPIExecError('Failed to check the next startup patch file')
            if next_startup.patch:
                sleep(FILE_DELETE_DELAY_TIME)
                delete_file_all(next_startup.patch, slave,
                                [cur_startup.patch, self.startup_info_before_set.patch])
        except Exception as reason:
            logging.error(reason)

        # 3. Reset next startup system software and delete it
        try:
            # user configure startup info after ZTP
            if next_startup.image != self.startup_info_from_ini_or_cfg.get("SYSTEM-SOFTWARE"):
                logging.info("no need to reset startup image")
                if self.startup_info_from_ini_or_cfg.get("SYSTEM-SOFTWARE"):
                    sleep(FILE_DELETE_DELAY_TIME)
                    delete_file_all(self.startup_info_from_ini_or_cfg.get("SYSTEM-SOFTWARE"), slave,
                                    [cur_startup.image, next_startup.image])
            # user do not configure startup info
            elif next_startup.image != self.startup_info_before_set.image:
                logging.info("reset startup config to the beginning")
                self._set_startup_image_file(self.startup_info_before_set.image)
                if self._check_next_startup_file(self.startup_info_before_set.image, 'image', slave) is not True:
                    raise OPIExecError('Failed to check the next startup image file')
                if next_startup.image:
                    sleep(FILE_DELETE_DELAY_TIME)
                    delete_file_all(next_startup.image, slave,
                                    [cur_startup.image, self.startup_info_before_set.image])
        except Exception as reason:
            logging.error(reason)

    def set_startup_info_from_ini_or_cfg(self, startup_info):
        for item_key in ['SYSTEM-SOFTWARE', 'SYSTEM-CONFIG', 'SYSTEM-PAT']:
            if not startup_info[item_key]:
                self.startup_info_from_ini_or_cfg[item_key] = startup_info[item_key]
            else:
                self.startup_info_from_ini_or_cfg[item_key] = 'cfcard:/' + startup_info[item_key]


def convert_byte_to_str(data):
    result = data
    if not isinstance(data, str):
        result = str(data, "iso-8859-1")
    return result


def sha256sum(fname, need_skip_first_line=False):
    """
    Calculate sha256 num for this file.
    """

    def read_chunks(fhdl):
        '''read chunks'''
        chunk = fhdl.read(8096)
        while chunk:
            yield chunk
            chunk = fhdl.read(8096)
        else:
            fhdl.seek(0)

    sha256_obj = hashlib.sha256()
    if isinstance(fname, str):
        with open(fname, "rb") as fhdl:
            # skip the first line
            fhdl.seek(0)
            if need_skip_first_line:
                fhdl.readline()
            for chunk in read_chunks(fhdl):
```



```
            sha256_obj.update(chunk)
    elif fname.__class__.__name__ in ["StringIO", "StringO"]:
        for chunk in read_chunks(fname):
            sha256_obj.update(chunk)
    else:
        pass
    return sha256_obj.hexdigest()


def sha256_get_from_file(fname):
    """Get sha256 num form file, stored in first line"""
    with open(fname, "rb") as fhdl:
        fhdl.seek(0)
        line_first = convert_byte_to_str(fhdl.readline())
    # if not match pattern, the format of this file is not supported
    if not re.match('^#sha256sum="[\\w]{64}"[\r\n]+$', line_first):
        return 'None'

    return line_first[12:76]


def sha256_check_with_first_line(fname):
    """Validate sha256 for this file"""

    work_fname = os.path.join("ztp", fname)
    sha256_calc = sha256sum(work_fname, True)
    sha256_file = sha256_get_from_file(work_fname)

    if sha256_file.lower() != sha256_calc:
        logging.warning('SHA256 check failed, file %s', fname)
        logging.warning('SHA256 checksum of the file "%s" is %s', fname, sha256_calc)
        logging.warning('SHA256 checksum received from the file "%s" is %s', fname, sha256_file)
        return False

    return True


def parse_sha256_file(fname):
    """parse sha256 file"""

    def read_line(fhdl):
        """read a line by loop"""
        line = fhdl.readline()
        while line:
            yield line
            line = fhdl.readline()
        else:
            fhdl.seek(0)

    sha256_dic = {}
    work_fname = os.path.join("ztp", fname)
    with open(work_fname, "rb") as fhdl:
        for line in read_line(fhdl):
            line_spilt = convert_byte_to_str(line).split()
            if 2 != len(line_spilt):
                continue
            dic_tmp = {line_spilt[0]: line_spilt[1]}
            sha256_dic.update(dic_tmp)
    return sha256_dic


def verify_and_parse_sha256_file(fname):
    """
    verify data integrity of sha256 file and parse this file

    format of this file is like:
    ----------------------------------------------------------------------

    file-name              sha256
```

```
    conf_5618642831132.cfg 1254b2e49d3347c4147a90858fa5f59aa2594b7294304f34e7da328bf3cdfbae
    ----------------------------------------------------------------------
    """
    if not sha256_check_with_first_line(fname):
        return ERR, None
    return OK, parse_sha256_file(fname)


def sha256_check_with_dic(sha256_dic, fname):
    """sha256 check with dic"""
    if fname not in sha256_dic:
        logging.info('sha256_dic does not has key %s, no need to do sha256 verification', fname)
        return True

    sha256sum_result = sha256sum(fname, False)
    if sha256_dic[fname].lower() == sha256sum_result:
        logging.info('SHA256 check %s successfully', fname)
        return True

    logging.warning('SHA256 check failed, file %s', fname)
    logging.warning('SHA256 checksum of the file "%s" is %s', fname, sha256sum_result)
    logging.warning('SHA256 checksum received for the file "%s" is %s', fname, sha256_dic[fname])

    return False


def check_parameter(aset):
    seq = ['&', '>', '<', "'", '"']
    if aset:
        for c in seq:
            if c in aset:
                return True
    return False


def check_filename():
    sys_info = get_system_info()
    url_tuple = urlparse(FILE_SERVER)
    if check_parameter(url_tuple.username) or check_parameter(url_tuple.password):
        logging.error('Invalid username or password, the name should not contain: ' + '&' + ' >' + ' <' + ' "' +
" '.")
        return ERR
    file_name = os.path.basename(REMOTE_PATH_IMAGE.get(sys_info['productName'], ''))
    if file_name is not '' and check_parameter(file_name):
        logging.error(
            'Invalid filename of system software, the name should not contain: ' + '&' + ' >' + ' <' + ' "' + " '.")
        return ERR
    file_name = os.path.basename(REMOTE_PATH_CONFIG)
    if file_name is not '' and check_parameter(file_name):
        logging.error(
            'Invalid filename of configuration file, the name should not contain: ' + '&' + ' >' + ' <' + ' "' + " '.")
        return ERR
    file_name = os.path.basename(REMOTE_PATH_PATCH.get(sys_info['productName'], ''))
    if file_name is not '' and check_parameter(file_name):
        logging.error(
            'Invalid filename of patch file, the name should not contain: ' + '&' + ' >' + ' <' + ' "' + " '.")
        return ERR
    try:
        file_name = os.path.basename(REMOTE_PATH_SHA256)
    except NameError:
        file_name = ''
    if file_name is not '' and check_parameter(file_name):
        logging.error(
            'Invalid filename of sha256 file, the name should not contain: ' + '&' + ' >' + ' <' + ' "' + " '.")
        return ERR

    return OK
```



```
def download_cfg_file(startup_info, slave, ip_protocol, vpn_instance, sha256_val_dic):
    """ Download configuration file """
    url = os.path.join(startup_info['FILESERVER'], startup_info['SYSTEM-CONFIG'])
    local_path_config = os.path.join('cfcard:', os.path.basename(startup_info['SYSTEM-CONFIG']))
    delete_file_all(local_path_config, slave)
    ret = download_file(url, os.path.basename(local_path_config), ip_protocol, vpn_instance)

    if ret == ERR or not file_exist(os.path.basename(url)):
        logging.error('%s download fail', os.path.basename(startup_info['SYSTEM-CONFIG']))
        return False, local_path_config

    if sha256_val_dic is not None:
        if not startup_info['SYSTEM-CONFIG']:
            return False, local_path_config
        file_name = os.path.basename(startup_info['SYSTEM-CONFIG'])
        if not sha256_check_with_dic(sha256_val_dic, file_name):
            logging.error('Error: SHA256 check failed, file "%s"' % file_name)
            return False, local_path_config

    if slave:
        ret = copy_file(local_path_config, 'slave#' + local_path_config)
        if ret is False:
            logging.error('%s failed copy to slave board', os.path.basename(startup_info['SYSTEM-CONFIG']))
            return False, local_path_config

    return True, local_path_config


def download_patch_file(startup_info, slave, ip_protocol, vpn_instance, sha256_val_dic):
    """ Download patch file """
    file_name = os.path.basename(startup_info['SYSTEM-PAT'])
    url = os.path.join(startup_info['FILESERVER'], startup_info['SYSTEM-PAT'])
    local_path_patch = os.path.join('cfcard:', file_name)
    delete_file_all(local_path_patch, slave)
    ret = download_file(url, file_name, ip_protocol, vpn_instance)
    if ret == DISK_SPACE_NOT_ENOUGH:
        logging.error('The space of disk is not enough')
        return DISK_SPACE_NOT_ENOUGH, local_path_patch
    if ret != OK or not file_exist(file_name):
        logging.error('%s download fail', file_name)
        return ERR, local_path_patch

    if not sha256_check_with_dic(sha256_val_dic, file_name):
        logging.error('Error: SHA256 check failed, file "%s"' % file_name)
        return ERR, local_path_patch

    if slave:
        ret = copy_file(local_path_patch, 'slave#' + local_path_patch)
        if ret is False:
            logging.error('%s failed copy to slave board', file_name)
            return ERR, local_path_patch

    return OK, local_path_patch


def download_image_file(startup_info, slave, ip_protocol, vpn_instance, sha256_val_dic):
    """ Download system software """
    file_name = os.path.basename(startup_info['SYSTEM-SOFTWARE'])
    url = startup_info['FILESERVER'] + '/' + startup_info['SYSTEM-SOFTWARE']
    local_path_image = os.path.join('cfcard:', file_name)
    delete_file_all(local_path_image, slave)
    ret = download_file(url, file_name, ip_protocol, vpn_instance)
    if ret == DISK_SPACE_NOT_ENOUGH:
        logging.error('The space of disk is not enough')
        return DISK_SPACE_NOT_ENOUGH, local_path_image
    if ret != OK or not file_exist(file_name):
        logging.error('%s download fail', file_name)
        return ERR, local_path_image
```



```
    if not sha256_check_with_dic(sha256_val_dic, file_name):
        logging.error('Error: SHA256 check failed, file "%s"' % file_name)
        return ERR, local_path_image

    if slave:
        ret = copy_file(local_path_image, 'slave#' + local_path_image)
        if ret is False:
            logging.error('%s failed to copy to slave board', file_name)
            return ERR, local_path_image

    return OK, local_path_image


def download_startup_file(startup_info, slave, ip_protocol, vpn_instance):
    """Download startup file"""
    # init here
    local_path_config = None
    local_path_patch = None
    local_path_image = None

    # current STARTUP_INFO
    cur_startup, next_startup = STARTUP._get_startup_info()
    cur_config = None if not cur_startup.config else os.path.basename(cur_startup.config)
    cur_patch = None if not cur_startup.patch else os.path.basename(cur_startup.patch)
    cur_image = None if not cur_startup.image else os.path.basename(cur_startup.image)
    next_config = None if not next_startup.config else os.path.basename(next_startup.config)
    next_patch = None if not next_startup.patch else os.path.basename(next_startup.patch)
    next_image = None if not next_startup.image else os.path.basename(next_startup.image)

    # download sha256 file first, used to verify data integrity of files which will be downloaded next
    try:
        cwd = get_cwd()
        file_path = REMOTE_PATH_SHA256
        if not file_path.startswith('/'):
            file_path = '/' + file_path
        file_name = os.path.basename(file_path)
        if file_name:
            url = FILE_SERVER + file_path
            local_path = os.path.join(cwd, "ztp", file_name)
            ret = download_file(url, local_path, ip_protocol, vpn_instance)
            if ret is ERR:
                logging.error('Error: Failed to download sha256 file "%s"' % file_name)
                return ERR, None, None, None
            logging.info('Info: Download sha256 file successfully')
            ret, sha256_val_dic = verify_and_parse_sha256_file(file_name)

            # delete the file immediately
            delete_file_all(local_path, slave)
            if ret is ERR:
                logging.error('Error: sha256 check failed, file "%s"' % file_name)
                return ERR, None, None, None
        else:
            sha256_val_dic = {}
    except NameError:
        sha256_val_dic = {}
        logging.info('no need sha256 to check download file')

    # if user change the startup to the name in ini/cfg, ztp will not download
    # 1. Download configuration file
    if startup_info['SYSTEM-CONFIG'] and startup_info['SYSTEM-CONFIG'] not in [cur_config, next_config]:
        ret, local_path_config = download_cfg_file(startup_info, slave, ip_protocol, vpn_instance,
sha256_val_dic)
        if ret is False:
            logging.info('delete startup file [cfg]')
            delete_startup_file(local_path_image, local_path_config, local_path_patch, slave)
            return ERR, local_path_image, local_path_config, local_path_patch
        logging.info('succeed to download config file')
    elif startup_info['SYSTEM-CONFIG'] and startup_info['SYSTEM-CONFIG'] in [cur_config, next_config]:
        logging.warning('The configured config version is the same as the current device version')
```



```
    # 2. Download patch file
    if startup_info['SYSTEM-PAT'] and startup_info['SYSTEM-PAT'] not in [cur_patch, next_patch]:
        ret, local_path_patch = download_patch_file(startup_info, slave, ip_protocol, vpn_instance,
sha256_val_dic)
        if ret is ERR:
            delete_startup_file(local_path_image, local_path_config, local_path_patch, slave)
            return ERR, local_path_image, local_path_config, local_path_patch
        if ret == DISK_SPACE_NOT_ENOUGH:
            delete_startup_file(local_path_image, None, local_path_patch, slave)
            logging.info('disk space not enough, delete patch')
            return OK, None, local_path_config, None
    elif startup_info['SYSTEM-PAT'] and startup_info['SYSTEM-PAT'] in [cur_patch, next_patch]:
        logging.warning('The configured patch version is the same as the current device version')

    # 3. Download system software
    if startup_info['SYSTEM-SOFTWARE'] and startup_info['SYSTEM-SOFTWARE'] not in [cur_image,
next_image]:
        ret, local_path_image = download_image_file(startup_info, slave, ip_protocol, vpn_instance,
sha256_val_dic)
        if ret is ERR:
            delete_startup_file(local_path_image, local_path_config, local_path_patch, slave)
            return ERR, local_path_image, local_path_config, local_path_patch
        if ret == DISK_SPACE_NOT_ENOUGH:
            delete_startup_file(local_path_image, None, local_path_patch, slave)
            logging.info('disk space not enough, delete image and patch')
            return OK, None, local_path_config, None
    elif startup_info['SYSTEM-SOFTWARE'] and startup_info['SYSTEM-SOFTWARE'] in [cur_image,
next_image]:
        logging.warning('The configured image version is the same as the current device version')

    return OK, local_path_image, local_path_config, local_path_patch

def set_startup_file(image_file, config_file, patch_file, slave):
    """Set startup file"""
    try:
        STARTUP.set_startup_info(image_file, config_file, patch_file, slave)
    except OPIExecError:
        return ERR

    logging.info('Set startup info ready %s %s %s', format_file_path(image_file),
format_file_path(config_file),
                 format_file_path(patch_file))
    return OK

def format_file_path(file_path):
    """format fiel_path,protect full path"""
    if file_path:
        return os.path.basename(file_path)
    return file_path

def delete_startup_file(image_file, config_file, patch_file, slave):
    """Delete all system file"""
    delete_file_all(image_file, slave)
    delete_file_all(config_file, slave)
    delete_file_all(patch_file, slave)


# ztplib
def set_ztp_last_status(state):
    """Set ztp last status."""
    uri = '/ztpops/ztpStatus/ztpLastStatus'
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <ztpLastStatus>$ztpLastStatus</ztpLastStatus>''')
    req_data = str_temp.substitute(ztpLastStatus=state)
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        logging.error('Failed to set ztp last status to %s', LAST_STATE_MAP[state])
```



```
        return

    logging.info('Succeed to set ztp last status to %s', LAST_STATE_MAP[state])


def get_ztp_enable_status():
    """Get ztp enable status
    :raise: OPIExecError
    """
    uri = '/ztpops/ztpStatus/ztpEnable'
    req_data = '''<?xml version="1.0" encoding="UTF-8"?>
        <ztpEnable/>'''
    ret, err_code, rsp_data = OPS_CLIENT.get(uri, req_data)
    if ret != HTTP_OK or rsp_data == '':
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to get ztp enable status')

    root_elem = etree.fromstring(rsp_data)
    namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
    uri = 'data' + uri.replace('/', '/vrp:')
    elem = root_elem.find(uri, namespaces)
    if elem is None:
        raise OPIExecError('Failed to read ztp enable status')

    return elem.text


def parse_environment(env):
    lines = re.split(r'\r\n|\r|\n', env)
    for line in lines:
        if re.match('.*ztp_exit_flag.*', line):
            item = re.split(r'[ ][ ]*', line)
            logging.info('parse environment, ztp_exit_flag: ' + item[2])
            return item[2]

    return None


def get_ztp_exit_environment():
    _ops = ops.ops()
    handle, err_desp = _ops.cli.open()
    ret = _ops.cli.execute(handle, "display ops environment")
    _ops.cli.close(handle)
    if ret[2] == 'Success' and ret[0]:
        return parse_environment(ret[0])

    return None


def check_ztp_continue():
    """Check if ztp can continue to run"""
    res = True
    try:
        enable_state = get_ztp_enable_status()
        ztp_exit_flag = get_ztp_exit_environment()
        if enable_state == 'false' or ztp_exit_flag == 'true':
            res = False
    except OPIExecError as ex:
        logging.warning(ex)

    return res


# DNS
class DNSServer:
    """Dns protocol service"""
    __slots__ = ['dns_servers', 'enable_state', 'vpn_instance']

    def __init__(self):
```



```
    self.dns_servers = []
    self.enable_state = 'false'
    self.vpn_instance = {}

def _set_dns_enable_switch(self, switch):
    """Set DNS global switch."""
    if switch not in ['true', 'false']:
        return

    if self.enable_state == switch:
        logging.info('The current enable state of dns is %s, no need to set', DNS_STATE_MAP.get(switch))
        return

    uri = '/dns/dnsGlobalCfgs/dnsGlobalCfg'
    str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
        <dnsGlobalCfg>
            <dnsEnable>$dnsEnable</dnsEnable>
        </dnsGlobalCfg>''')
    req_data = str_temp.substitute(dnsEnable=switch)
    ret, err_code, rsp_data = OPS_CLIENT.set(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to %s DNS' % DNS_STATE_MAP.get(switch))

    self.enable_state = switch
    return

def add_dns_servers_ipv4(self, dns_servers, vpn_instance):
    """Add IPv4 DNS servers configuration.
    :raise: OPIExecError
    """
    while '255.255.255.255' in dns_servers:
        dns_servers.remove('255.255.255.255')

    # only configure new dns servers
    new_dns_servers = list(set(dns_servers).difference(set(self.dns_servers)))
    if not new_dns_servers:
        return

    self._set_dns_enable_switch('true')
    logging.info('Add DNS IPv4 servers')

    uri = '/dns/dnsIpv4Servers'
    root_elem = etree.Element('dnsIpv4Servers')
    for server_addr in new_dns_servers:
        dns_server = etree.SubElement(root_elem, 'dnsIpv4Server')
        etree.SubElement(dns_server, 'ipv4Addr').text = server_addr
        etree.SubElement(dns_server, 'vrfName').text = vpn_instance
    req_data = etree.tostring(root_elem, 'UTF-8')
    ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data)
    if ret != HTTP_OK:
        logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
        raise OPIExecError('Failed to config DNS IPv4 server')

    # configure success
    self.dns_servers.extend(new_dns_servers)
    self.vpn_instance.update(dict.fromkeys(new_dns_servers, vpn_instance))

def del_dns_servers_ipv4(self):
    """Delete IPv4 DNS servers configuration.
    :raise: OPIExecError
    """
    if not self.dns_servers:
        logging.info('Current dns server is empty, no need to delete')
        return

    logging.info('Delete DNS IPv4 servers')

    uri = '/dns/dnsIpv4Servers'
```



```
        root_elem = etree.Element('dnsIpv4Servers')
        for server_addr in self.dns_servers:
            dns_server = etree.SubElement(root_elem, 'dnsIpv4Server')
            etree.SubElement(dns_server, 'ipv4Addr').text = server_addr
            etree.SubElement(dns_server, 'vrfName').text = self.vpn_instance.get(server_addr)
        req_data = etree.tostring(root_elem, 'UTF-8')

        ret, err_code, rsp_data = OPS_CLIENT.delete(uri, req_data)
        if ret != HTTP_OK:
            logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
            raise OPIExecError('Failed to delete DNS IPv4 server')

        # delete all dns server success
        self.vpn_instance = {}
        self.dns_servers = []

        self._set_dns_enable_switch('false')

    @staticmethod
    def get_addr_by_hostname(host, vpn_instance, addr_type='1'):
        """Translate a host name to IPv4 address format. The IPv4 address is returned as a string.
        :raise: OPIExecError
        """
        logging.info('Get ipv4 address by host name %s', host)
        uri = '/dns/dnsNameResolution'
        root_elem = etree.Element('dnsNameResolution')
        etree.SubElement(root_elem, 'host').text = host
        etree.SubElement(root_elem, 'addrType').text = addr_type
        etree.SubElement(root_elem, 'vrfName').text = vpn_instance
        req_data = etree.tostring(root_elem, "UTF-8")
        logging.warning(req_data)
        ret, err_code, rsp_data = OPS_CLIENT.get(uri, req_data)
        if ret != HTTP_OK or rsp_data == '':
            logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
            raise OPIExecError('Failed to get ipv4 address by host name')

        root_elem = etree.fromstring(rsp_data)
        namespaces = {'vrp': 'http://www.huawei.com/netconf/vrp'}
        uri = 'data' + uri.replace('/', '/vrp:') + '/vrp:'
        elem = root_elem.find(uri + 'ipv4Addr', namespaces)
        if elem is None:
            logging.error(rsp_data)
            raise OPIExecError('Failed to read IP address by host name')
        formated_ip = convert_ip_format(elem.text)
        logging.info(rsp_data.replace(elem.text, formated_ip))
        return elem.text
def convert_ip_format(ip_addr_str):
    """ip format
      :input:
        x.x.x.x
      :return
         *.*.*.x
    """

    if ip_addr_str and '.' in ip_addr_str:
        last_point = ip_addr_str.rfind('.')
        ip_addr_str = re.sub(r'\w+', '*', ip_addr_str[:last_point + 1]) + ip_addr_str[last_point + 1:]
    return ip_addr_str

# download
def download_file(url, local_path, ip_protocol, vpn_instance):
    """
    Description:
        Download file, support TFTP, FTP, SFTP.
    Args:
      url: URL of remote file
        tftp://hostname/path
        ftp://[username[:password]@]hostname/path
        sftp://[username[:password]@]hostname[:port]/path
```

```
  local_path: local path to put the file
    cfcard:/xxx

  ip_protocol: ipv4 or ipv6
  vpn_instance: vpn_instance
Returns:
    ERR[1]: download fail
    OK[0]:  download success
"""

url_tuple = urlparse(url)
func_dict = {
    'tftp': {
        IPV4: TFTPv4,
        IPV6: TFTPv6,
    },
    'ftp': {
        IPV4: FTPv4,
        IPV6: FTPv6,
    },
    'sftp': {
        IPV4: SFTPv4,
        IPV6: SFTPv6,
    }
}

scheme = url_tuple.scheme
if scheme not in func_dict.keys():
    logging.error('Unknown file transfer scheme %s', scheme)
    return ERR

if ip_protocol == IPV4:
    if not re.match(r'\d+\.\d+\.\d+\.\d+', url_tuple.hostname):
        # get server ip by hostname from dns
        try:
            dns_vpn = '_public_' if vpn_instance in [None, ''] else vpn_instance
            server_ip = DNS.get_addr_by_hostname(url_tuple.hostname, dns_vpn)

            format_server_ip = convert_ip_format(server_ip)
            logging.info("server ip: " + format_server_ip)
        except OPIExecError as ex:
            logging.error(ex)
            return ERR

        url = url.replace(url_tuple.hostname, server_ip)

vpn_instance = '' if vpn_instance in [None, '_public_'] else vpn_instance
logging.info('Start to download file %s using %s', os.path.basename(local_path), scheme)

ret = ERR
cnt = 0
while cnt < 1 + FILE_TRANSFER_RETRY_TIMES:
    if cnt:
        logging.info('Try downloading again, please wait a moment')
    try:
        ret = func_dict[scheme][ip_protocol](url, local_path, vpn_instance).start()
        logging.info("download module return code: [{}]".format(ret))
        if ret in SPACE_NOT_ENOUGH_TAG_LIST:
            ret = DISK_SPACE_NOT_ENOUGH
        if ret in [OK, DISK_SPACE_NOT_ENOUGH]:
            logging.info('download file %s using %s, ret:%d', os.path.basename(local_path), scheme, ret)
            break
        ret = ERR
        logging.error('Failed to download file %s using %s', os.path.basename(local_path), scheme)
        sleep(FILE_DOWNLOAD_INTERVAL_TIME)
    except OPIExecError as ex:
        logging.error(ex)
    except Exception as ex:
```



```
            logging.exception(ex)
        cnt += 1
    return ret


class Download:
    """File download base class"""
    def __init__(self, local_path):
        self.local_path = local_path

    def start(self):
        """Start to download file"""
        uri = self.get_uri()
        req_data = self.get_req_data()
        self.pre_download()
        ret, err_code, rsp_data = OPS_CLIENT.create(uri, req_data, False)
        if ret != HTTP_OK:
            delete_file_all(self.local_path, False)
            logging.error('HTTP response: HTTP/1.1 %s %s\n%s', ret, err_code, rsp_data)
            root = etree.fromstring(rsp_data)
            rpc_error = root.find('rpc-error')
            if rpc_error and rpc_error.find('error-app-tag') is not None:
                ret = int(rpc_error.find('error-app-tag').text)
            else:
                ret = ERR
        else:
            ret = OK
        self.after_download()
        return ret

    def get_uri(self):
        """Return download request uri"""
        raise NotImplementedError

    def get_req_data(self):
        """Return download request xml message"""
        raise NotImplementedError

    def pre_download(self):
        """Do some actions before download file"""
        raise NotImplementedError

    def after_download(self):
        """Do some actions after download file"""
        raise NotImplementedError


class FTP(Download):
    """FTP download class"""

    def get_uri(self):
        """Return ftp download request uri"""
        return '/ftpc/ftpcTransferFiles/ftpcTransferFile'

    def get_req_data(self):
        """Implemented by subclasses"""
        raise NotImplementedError

    def pre_download(self):
        """FTP not care"""

    def after_download(self):
        """FTP not care"""


class FTPv4(FTP):
    """FTPv4 download class"""

    def __init__(self, url, local_path, vpn_instance):
```

```
        self.url = url
        super().__init__(local_path)
        self.vpn_instance = vpn_instance

    def get_req_data(self):
        """Return ftpv4 download request xml message"""
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <ftpcTransferFile>
                <serverIpv4Address>$serverIp</serverIpv4Address>
                <commandType>get</commandType>
                <userName>$username</userName>
                <password>$password</password>
                <localFileName>$localPath</localFileName>
                <remoteFileName>$remotePath</remoteFileName>
                <vpnInstanceName>$vpnInstance</vpnInstanceName>
            </ftpcTransferFile>''')
        url_tuple = urlparse(self.url)
        req_data = str_temp.substitute(serverIp=url_tuple.hostname,
                                       username=url_tuple.username,
                                       password=url_tuple.password,
                                       remotePath=url_tuple.path[1:],
                                       localPath=self.local_path,
                                       vpnInstance=self.vpn_instance)
        return req_data


class FTPv6(FTP):
    """FTPv6 download class"""

    def __init__(self, url, local_path, vpn_instance):
        self.url = url
        super().__init__(local_path)
        self.vpn_instance = vpn_instance

    def get_req_data(self):
        """Return ftpv6 download request xml message"""
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <ftpcTransferFile>
                <serverIpv6Address>$serverIp</serverIpv6Address>
                <commandType>get</commandType>
                <userName>$username</userName>
                <password>$password</password>
                <localFileName>$localPath</localFileName>
                <remoteFileName>$remotePath</remoteFileName>
                <ipv6VpnName>$vpnInstance</ipv6VpnName>
            </ftpcTransferFile>''')
        url_tuple = urlparse(self.url)
        idx = url_tuple.netloc.rfind('@')
        server_ip = url_tuple.netloc[idx + 1:]
        req_data = str_temp.substitute(serverIp=server_ip,
                                       username=url_tuple.username,
                                       password=url_tuple.password,
                                       remotePath=url_tuple.path[1:],
                                       localPath=self.local_path,
                                       vpnInstance=self.vpn_instance)
        return req_data


class TFTP(Download):
    """TFTP download class"""

    def get_uri(self):
        """Return ftp download request uri"""
        return '/tftpc/tftpcTransferFiles/tftpcTransferFile'

    def get_req_data(self):
        """Implemented by subclasses"""
        raise NotImplementedError
```



```
    def pre_download(self):
        """TFTP not case"""

    def after_download(self):
        """TFTP not case"""


class TFTPv4(TFTP):
    """TFTPv4 download class"""

    def __init__(self, url, local_path, vpn_instance):
        self.url = url
        super().__init__(local_path)
        self.vpn_instance = vpn_instance

    def get_req_data(self):
        """Return tftpv4 download request xml message"""
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <tftpcTransferFile>
                <serverIpv4Address>$serverIp</serverIpv4Address>
                <commandType>get_cmd</commandType>
                <localFileName>$localPath</localFileName>
                <remoteFileName>$remotePath</remoteFileName>
                <vpnInstanceName>$vpnInstance</vpnInstanceName>
            </tftpcTransferFile>''')
        url_tuple = urlparse(self.url)
        req_data = str_temp.substitute(serverIp=url_tuple.hostname,
                                       remotePath=url_tuple.path[1:],
                                       localPath=self.local_path,
                                       vpnInstance=self.vpn_instance)
        return req_data


class TFTPv6(TFTP):
    """TFTPv6 download class"""

    def __init__(self, url, local_path, vpn_instance):
        self.url = url
        super().__init__(local_path)
        self.vpn_instance = vpn_instance

    def get_req_data(self):
        """Return tftpv4 download request xml message"""
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <tftpcTransferFile>
                <serverIpv6Address>$serverIp</serverIpv6Address>
                <commandType>get_cmd</commandType>
                <localFileName>$localPath</localFileName>
                <remoteFileName>$remotePath</remoteFileName>
                <ipv6VpnName>$vpnInstance</ipv6VpnName>
            </tftpcTransferFile>''')
        url_tuple = urlparse(self.url)
        idx = url_tuple.netloc.rfind('@')
        server_ip = url_tuple.netloc[idx + 1:]
        req_data = str_temp.substitute(serverIp=server_ip,
                                       remotePath=url_tuple.path[1:],
                                       localPath=self.local_path,
                                       vpnInstance=self.vpn_instance)
        return req_data


class SFTP(Download):
    """SFTP download class"""

    def get_uri(self):
        """Return ftp download request uri"""
        return '/sshc/sshcConnects/sshcConnect'

    def get_req_data(self):
```

```
        """Implemented by subclasses"""
        raise NotImplementedError

    def pre_download(self, ):
        self._set_sshc_first_time('Enable')

    def after_download(self):
        self._del_sshc_rsa_key()
        self._set_sshc_first_time('Disable')

    @classmethod
    def _set_sshc_first_time(cls, switch):
        """Set SSH client attribute of authenticating user for the first time access"""
        if switch not in ['Enable', 'Disable']:
            return ERR

        logging.info('Set SSH client first-time enable switch = %s', switch)
        uri = "/sshc/sshClient"
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <sshClient>
                <firstTimeEnable>$enable</firstTimeEnable>
            </sshClient>''')
        req_data = str_temp.substitute(enable=switch)
        ret, _, _ = OPS_CLIENT.set(uri, req_data)
        if ret != HTTP_OK:
            if switch == 'Enable':
                reason = 'Failed to enable SSH client first-time'
            else:
                reason = 'Failed to disable SSH client first-time'

            raise OPIExecError(reason)

        return OK

    def _del_rsa_peer_key(self):
        """Delete RSA peer key configuration"""
        logging.info('Delete RSA peer key')
        uri = '/rsa/rsaPeerKeys/rsaPeerKey'
        root_elem = etree.Element('rsaPeerKey')
        etree.SubElement(root_elem, 'keyName').text = self.get_key_name()
        req_data = etree.tostring(root_elem, 'UTF-8')
        ret, _, _ = OPS_CLIENT.delete(uri, req_data)
        if ret != HTTP_OK:
            logging.error('Failed to delete RSA peer key')

    def _del_sshc_rsa_key(self, key_type='RSA'):
        """Delete SSH client RSA key configuration"""
        logging.info('Delete SSH client RSA key')
        uri = '/sshc/sshCliKeyCfgs/sshCliKeyCfg'
        root_elem = etree.Element('sshCliKeyCfg')
        etree.SubElement(root_elem, 'serverName').text = self.get_key_name()
        etree.SubElement(root_elem, 'pubKeyType').text = key_type
        req_data = etree.tostring(root_elem, 'UTF-8')
        ret, _, _ = OPS_CLIENT.delete(uri, req_data)
        if ret != HTTP_OK:
            logging.error('Failed to delete SSH client RSA key')

        self._del_rsa_peer_key()

    def get_key_name(self):
        """Get sftp server ip"""
        raise NotImplementedError


class SFTPv4(SFTP):
    """SFTPv4 download class"""

    def __init__(self, url, local_path, vpn_instance):
        self.url = url
```

```
        super().__init__(local_path)
        self.vpn_instance = vpn_instance

    def get_key_name(self):
        url_tuple = urlparse(self.url)
        return url_tuple.hostname

    def get_req_data(self):
        """Return sftpv4 download request xml message"""
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <sshcConnect>
                <HostAddrIPv4>$serverIp</HostAddrIPv4>
                <commandType>get</commandType>
                <userName>$username</userName>
                <password>$password</password>
                <serverPort>$port</serverPort>
                <localFileName>$localPath</localFileName>
                <remoteFileName>$remotePath</remoteFileName>
                <vpnInstanceName>$vpnInstance</vpnInstanceName>
                <transferType>SFTP</transferType>
            </sshcConnect>''')
        url_tuple = urlparse(self.url)
        try:
            if url_tuple.port is None:
                port = 22
            else:
                port = url_tuple.port
        except ValueError:
            port = 22

        logging.info('Sftp download file using port:%s', port)
        req_data = str_temp.substitute(serverIp=url_tuple.hostname,
                                       username=url_tuple.username,
                                       password=url_tuple.password,
                                       port=port,
                                       remotePath=url_tuple.path[1:],
                                       localPath=self.local_path,
                                       vpnInstance=self.vpn_instance)
        return req_data


class SFTPv6(SFTP):
    """SFTPv6 download class"""

    def __init__(self, url, local_path, vpn_instance):
        self.url = url
        super().__init__(local_path)
        self.vpn_instance = vpn_instance

    def get_key_name(self):
        url_tuple = urlparse(self.url)
        idx = url_tuple.netloc.find('@')
        return url_tuple.netloc[idx + 1:]

    def get_req_data(self):
        """Return sftpv4 download request xml message"""
        str_temp = string.Template('''<?xml version="1.0" encoding="UTF-8"?>
            <sshcConnect>
                <HostAddrIPv6>$serverIp</HostAddrIPv6>
                <commandType>get</commandType>
                <userName>$username</userName>
                <password>$password</password>
                <localFileName>$localPath</localFileName>
                <remoteFileName>$remotePath</remoteFileName>
                <ipv6VpnName>$vpnInstance</ipv6VpnName>
                <transferType>SFTP</transferType>
            </sshcConnect>''')
        url_tuple = urlparse(self.url)
        server_ip = self.get_key_name()
```

```
        req_data = str_temp.substitute(serverIp=server_ip,
                                       username=url_tuple.username,
                                       password=url_tuple.password,
                                       remotePath=url_tuple.path[1:],
                                       localPath=self.local_path,
                                       vpnInstance=self.vpn_instance)
        return req_data


def _is_startup_info_valid(startup_info):
    """Does startup info valid
        FILESERVER, SOFTWARE, CONFIG, PATCH, not None
    """
    return startup_info.get('SYSTEM-CONFIG', None) and startup_info.get('FILESERVER', None)


def main_proc(vpn_instance, ip_protocol):
    """
    :param vpn_instance:
    :param ip_protocol:
    :return:
    """
    global REMOTE_PATH_CONFIG

    sys_info = get_system_info()
    slave, _ = has_slave_mpu()  # Check whether slave MPU board exists or not
    logging.info('Get devicetype=%s, esn=%s, mac=%s from the current system',
                 sys_info['productName'],
                 hide_content(sys_info['esn'], 4),
                 hide_content(sys_info['mac'], 4))
    if not REMOTE_PATH_IMAGE.get(sys_info['productName']):
        logging.warning(
            "The product name of the current device [{}] not in
REMOTE_PATH_IMAGE".format(sys_info['productName']))
    if not REMOTE_PATH_PATCH.get(sys_info['productName']):
        logging.warning(
            "The product name of the current device [{}] not in
REMOTE_PATH_PATCH".format(sys_info['productName']))
    if '%s' in REMOTE_PATH_CONFIG:
        REMOTE_PATH_CONFIG = REMOTE_PATH_CONFIG % sys_info['esn']
    startup_info = {'FILESERVER': FILE_SERVER,
                    'SYSTEM-SOFTWARE': REMOTE_PATH_IMAGE.get(sys_info['productName'], ''),
                    'SYSTEM-CONFIG': REMOTE_PATH_CONFIG,
                    'SYSTEM-PAT': REMOTE_PATH_PATCH.get(sys_info['productName'], '')}
    STARTUP.set_startup_info_from_ini_or_cfg(startup_info)
    if not _is_startup_info_valid(startup_info):
        logging.warning('FILESERVER is None or SYSTEM-CONFIG is None, no need download and '
                        'set system startup file')
        return ERR

    ret = check_filename()
    if ret == ERR:
        return ERR

    # check remote file paths
    try:
        remote_path_sha256 = REMOTE_PATH_SHA256
    except NameError:
        remote_path_sha256 = ''
    if not check_file_type_valid(REMOTE_PATH_IMAGE.get(sys_info['productName'], ''),
REMOTE_PATH_CONFIG,
                                 REMOTE_PATH_PATCH.get(sys_info['productName'], ''), remote_path_sha256):
        return ERR

    ret, image_file, config_file, patch_file = download_startup_file(startup_info, slave,
                                                                     ip_protocol, vpn_instance)
    if ret == ERR:
        logging.info('failed to download file')
        return ERR
```

```
    if check_ztp_continue() is False:
        logging.info('user stop ztp before setting, ztp will reset startup')
        delete_startup_file(image_file, config_file, patch_file, slave)
        return ERR

    ret = set_startup_file(image_file, config_file, patch_file, slave)
    if ret == ERR:
        return ERR

    if not check_ztp_continue():
        logging.info('user stop ztp after setting, ztp will reset startup')
        STARTUP.reset_startup_info(slave)
        return ERR

    set_ztp_last_status('true')
    dhcp_stop()
    try:
        reboot_system()
    except OPIExecError as reason:
        logging.error("reboot failed: {}".format(reason))
        set_ztp_last_status('false')
        STARTUP.reset_startup_info(slave)
        return ERR

    return OK


def main(vpn_instance='', ip_protocol=IPV4):
    """The main function of user script. It is called by ZTP frame, so do not remove or change this function.

    Args:
    Raises:
    Returns: user script processing result
    """
    ip_protocol = ip_protocol.lower()
    try:
        ret = main_proc(vpn_instance, ip_protocol)
    except Exception as reason:
        logging.error(reason)
        trace_info = traceback.format_exc()
        logging.error(trace_info)
        ret = ERR

    finally:
        # Close the OPS connection
        OPS_CLIENT.close()

    return ret


while True:
    try:
        STARTUP = Startup()
        break
    except OPIExecError as ex:
        logging.warning(ex)
    sleep(CHECK_STARTUP_INTERVAL)

DNS = DNSServer()

if __name__ == "__main__":
    main()
```

CFG 格式中间文件中修复补丁配置示例

```
#sha256sum="fffcd63f5e31f0891a0349686969969c1ee429dedeaf7726ed304f2d08ce1bc7"
fileserver=sftp://username:password@hostname:port/path/;
mac=00e0-fc12-3456;esn=2102351931P0C3000154;devicetype=DEFAULT;system-
```

```
version=V800R023C00SPC500;boot_python_file=V800R023C00SPC500.py;system-
software=V800R023C00SPC500.cc;system-config=V800R023C00SPC500.cfg;system-
pat=V800R023C00SPC500SPH001.PAT;system-fix-pat=V800R023C00SPC500.PAT;
```

修复补丁字段含义

修复补丁新增字段解释如**表1-88**所示，其他ini格式字段含义参考 **ini格式的中间文件**，Python格式字段含义参考**Python脚本文件解释**，cfg格式字段含义参考 **cfg格式的中间文件**。

表 1-88 修复补丁新增字段含义

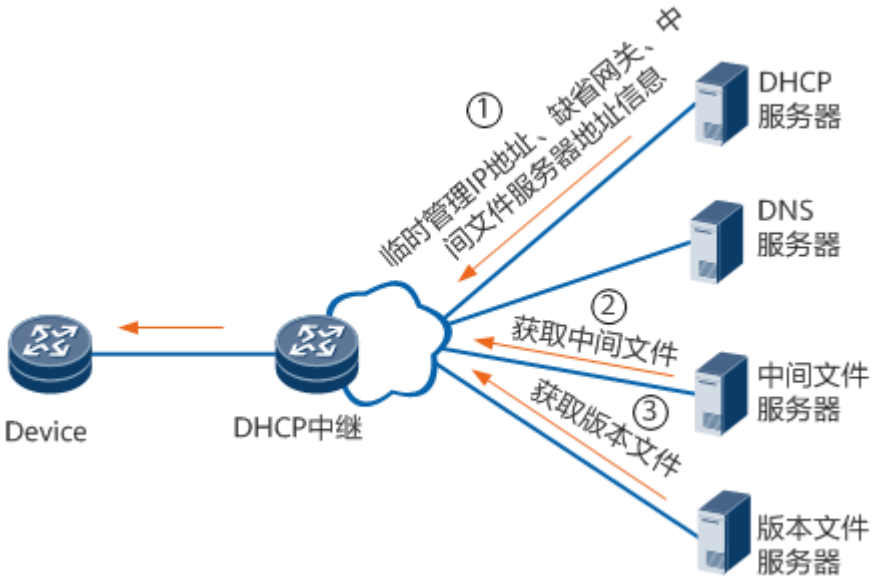
字段	是否可选	描述
SYSTEM-FIX-PAT	可选	ini格式中间文件中的修复补丁名称。如果不需要修复补丁，可以不配置此项，或配置空值，如："SYSTEM-FIX-PAT="。
REMOTE_PATH_FIX_PATCH	可选	Python格式中间文件中的修复补丁名称。如果不需要修复补丁，可以不配置此项，或配置空值，如："REMOTE_PATH_FIX_PATCH= {} "。
system-fix-pat	可选	cfg格式中间文件中的修复补丁名称。如果不需要修复补丁，可以不配置此项，或配置空值，如："system-fix-pat="。

1.1.6.1.3 应用

ZTP 实现空配置设备自动部署

如**图1-28**所示，设备空配置上电时，通过DHCP服务器获取中间文件服务器的地址，从中间文件服务器获取中间文件。通过中间文件获取版本文件服务器地址、系统软件、配置文件等信息，从服务器上下载版本文件，重启后完成自动部署。

图 1-28 ZTP 实现空配置设备自动部署



1.1.6.1.4 术语与缩略语

术语

无。

缩略语

缩略语	英文全称	中文全称
ZTP	Zero Touch Provisioning	零配置自动部署
DHCP	Dynamic Host Configuration Protocol	动态主机配置协议
TFTP	Trivial File Transfer Protocol	简单文件传输协议
FTP	File Transfer Protocol	文件传输协议
SFTP	Secure File Transfer Protocol	安全文件传输协议

1.1.6.2 ZTP 配置

设备可以通过零配置自动部署ZTP（Zero Touch Provisioning）实现空配置下的上电自动部署。

背景信息

通过配置ZTP，可以实现空配置设备上电自动部署，在大规模部署网络设备时可通过ZTP提高部署效率。

VS模式下，该特性仅在Admin-VS支持。

1.1.6.2.1 ZTP 概述

定义

ZTP是指空配置设备上电启动时采用的一种自动加载版本文件（包括系统软件、配置文件、补丁文件）的功能。

目的

在部署网络设备时，设备硬件安装完成后，需要管理员到安装现场对设备进行软件调试。当设备数量较多、分布较广时，管理员需要在每一台设备上进行手工配置，既影响了部署的效率，又需要较高的人力成本。

设备运行ZTP功能，可以从文件服务器获取版本文件并自动加载，实现设备的免现场配置、部署，从而降低人力成本，提升部署效率。

受益

实现设备的免现场配置、部署，降低人力成本，提升部署效率。

1.1.6.2.2 ZTP 特性限制

1.1.6.2.3 配置通过 DHCP 实现 ZTP 自动部署

通过DHCP完成ZTP实现自动部署以降低人力成本，提升部署效率。

应用环境

在部署网络设备时，设备硬件安装完成后，需要管理员到安装现场对设备进行软件调试。当设备数量较多、分布较广时，管理员需要在每一台设备上进行手工配置，既影响了部署的效率，又需要较高的人力成本。

设备运行ZTP功能，通过DHCP实现自动部署，实现设备的免现场配置、部署，从而降低人力成本，提升部署效率。

前置任务

在配置ZTP之前，需要完成以下任务：

- DHCP服务器、文件服务器到待配置设备的网关间路由可达。
- 确保待配置设备中没有启动配置文件。

编辑中间文件

操作步骤

步骤1 中间文件可以是ini文件、cfg文件或Python脚本文件。其中，ini文件和cfg文件的使用要求低且配置简单，Python脚本文件对用户要求高，所以推荐首次使用ZTP的用户选择ini文件或者cfg文件作为中间文件。文件格式请参见 **ini格式的中间文件**、 **cfg格式的中间文件**或 **Python格式的中间文件**。

----结束

配置 DHCPv4 Server 和 DHCPv4 Relay

背景信息

需要运行ZTP的设备在上电之前，须先部署DHCPv4服务器，以确保作为DHCP客户端的空配置设备能正常获取到IP地址、网关及中间文件服务器地址、中间文件名称等信息。

运行ZTP的设备进入DHCP discover阶段后，在发送DHCP discover消息时会携带DHCP option 60和61。DHCP option 60 (Vendor class identifier)用来携带设备厂商及型号信息；DHCP option 61 (Client-identifier)用来携带设备序列号。

DHCPv4服务器上需配置的Options字段见**表1-89**。

表 1-89 Options 字段说明

Option编号	是否可选	Option作用
1	必选	设置IP地址的子网掩码。
3	必选	设置DHCP客户端的出口网关。
6	可选	设置DNS服务器的IP地址。当用户设置中间文件服务器的主机名为域名类型时（如“www.ztp.com”），需要部署DNS服务器来将域名转换为相应的IP地址。如果设置的主机名为IP地址，则不需要再部署DNS服务器。
66	可选	设置中间文件服务器的主机名。文件服务器可以是TFTP/FTP/SFTP服务器，格式如下： ● tftp://*hostname* ● ftp://[*username*[:*password*]@]*hostname* ● sftp://[*username*[:*password*]@]*hostname*[:*port*] 其中*username*、*password*、*port*参数为可选项。*hostname*既可以是域名也可以是IP地址，如果设置的是域名地址，则需要部署DNS服务器。*port*的取值范围为0～65535，超出范围按照默认端口22处理，仅在SFTP服务器地址为IPv4情况下支持配置端口号。 **说明** 当*hostname*为IP地址时，可不配置文件传输类型，此时默认为TFTP。

Option编号	是否可选	Option作用
67	必选	设置中间文件名。中间文件的名称为*.ini、*.py或*.cfg。 ● 中间件文件名长度不要超过64个字符，超过64字符时可能会导致文件下载失败。 ● 中间件文件名中不能包含特殊字符，如：&、>、<、"或者'。 ● 中间文件名格式为：*path*/*filename*。其中*path*可以是不包括文件服务器主机名的相对路径，如“/script/ztp_script.py”，也可以是包括服务器主机名的绝对路径，如“sftp://10.1.1.1/script/ztp_script.py”。若使用相对路径，则需要设置Option 66。
150	可选	设置TFTP服务器的IP地址。

说明

ZTP通过DHCPv4申请的IPv4地址租期至少为1小时且不支持续租。

当待配置设备与DHCPv4服务器不在同一网段时，需要配置DHCP中继以转发DHCP的交互报文。

以下配置步骤以路由器为例，如果选择其他类型设备作为DHCPv4服务器或中继，请参考相应的资料配置。

操作步骤

步骤1 在作为DHCPv4服务器的设备上配置地址池，详细配置请参见“配置地址池”，地址池中**Option**编号：6、66、67、150的配置请参见“（可选）配置DHCPv4自定义选项”。

步骤2 配置地址池为接入用户分配地址，详细配置请参见“配置本地地址池为接入用户分配地址示例”。

步骤3 （可选）如果组网中存在DHCP中继，在作为DHCP中继的设备上进行配置，详细配置请参见“配置DHCP Relay”。

----结束

配置 DHCPv6 Server 和 DHCPv6 Relay

背景信息

需要运行ZTP的设备在上电之前，须先部署DHCPv6服务器，以确保作为DHCP客户端的空配置设备能正常获取到IP地址、网关及中间文件服务器地址、中间文件名称等信息。

运行ZTP的设备进入DHCPv6 Solicit阶段后，在发送DHCPv6 Solicit消息时会携带DHCPv6 option 6，DHCPv6 option 6用来携带客户端请求的选项代码。

DHCPv6服务器上需配置的Options字段见**表1-90**。

表 1-90 Options 字段说明

Option编号	是否可选	Option作用
5	必选	申请的IA地址，IPv6地址以及生存期。
59	必选	中间文件路径，中间文件的名称为*.ini、*.py或*.cfg。中间文件名格式如下： ● tftp://*hostname*/path/filename ● ftp://[*username*[:*password*]@]*hostname*/path/filename ● sftp://[*username*[:*password*]@]*hostname*/path/filename

说明

ZTP通过DHCPv6申请的IPv6地址租期至少为1小时且不支持续租。

当待配置设备与DHCPv6服务器不在同一网段时，需要配置DHCPv6中继以转发DHCPv6的交互报文。

操作步骤

步骤1 在作为DHCPv6服务器的设备上配置地址池，地址池中需要配置**Option 59**。

步骤2 配置地址池为接入用户分配地址。

步骤3 （可选）如果组网中存在DHCPv6中继，在作为DHCPv6中继的设备上进行配置，详细配置请参见“配置DHCPv6 Relay”。

----结束

配置文件服务器

背景信息

文件服务器用于存放空配置设备需要下载的文件，包括中间文件、版本文件等。用户可以将路由器配置为文件服务器，但由于文件服务器需要占用设备的存储资源，因此在使用路由器作为文件服务器时，需要考虑存储空间的问题。所以在ZTP网络中，一般需要部署第三方服务器，配置的具体方法请参见第三方服务器的操作指导。

用户可以将中间文件和版本文件部署在同一个文件服务器上。文件服务器可以是TFTP/FTP/SFTP服务器。

说明

文件服务器与空配置设备的缺省网关之间必须路由可达。

后续处理

配置完文件服务器后，将中间文件、版本文件存放至文件服务器。

说明

为充分保证文件服务器的安全，建议配置的文件服务器用户名唯一，并将其权限设置为只读，防止被非法修改。ZTP过程结束后，请关闭相应的文件服务器功能。

上电启动设备

背景信息

上述配置步骤完成后，将待配置设备上电启动。设备将自动下载版本文件并重新启动，完成自动部署。

操作步骤

步骤1 上电启动设备。

----结束

（可选）加载预配置脚本

背景信息

当设备空配置启动时，在进入ZTP流程之前，如果需要对设备进行预配置命令下发，则需要设置预配置脚本。

操作步骤

步骤1 根据文件类型和格式要求编辑预配置脚本，文件格式请参见 **预配置脚本**。

步骤2 上传预配置脚本至主控板的存储介质中。

说明

本设备支持FTP，TFTP以及SFTP上传文件，请根据实际情况选择文件上传方式上传到设备。

步骤3 执行命令**set ztp pre-configuration** *file-name*，加载预配置脚本。

若用户希望设备空配置启动时不执行ZTP预配置流程，可以执行命令**reset ztp pre-configuration**，清空预配置脚本。

步骤4 执行命令**display ztp status**，可以查看当前设备预配置脚本的配置状态。

说明

在设备软件大包从低版本升级到当前版本时，若设置的启动配置文件为vrpcfg.zip，加载的预配置脚本会被执行（若不想执行预配置脚本，可以通过执行命令**reset ztp pre-configuration**清空预配置脚本）；当设置其它配置文件时，预配置脚本不会被执行。

----结束

（可选）配置自动补丁修复

通过配置ZTP自动补丁修复，可以解决ZTP开局自动部署时遇到的非环境问题（环境问题如设备初始不稳定、服务器未启动等）。

背景信息

在ZTP开局自动部署过程中，出现非环境问题（环境问题如设备初始不稳定、服务器未启动等）时，不需要工程师前往站点维修或设备返厂维修，ZTP支持自动补丁修复机制。通过人工定位问题根因，联系工程师制作修复补丁，ZTP运行过程中可以自主识别、设置修复补丁，并重启设备使修复补丁生效。设备重启后ZTP会再次运行，由于此时问题已被修复，ZTP可以顺利完成开局部署。

操作步骤

步骤1 将修复补丁信息配置在ZTP的中间文件中，配置详情见**自动补丁修复机制**。

步骤2 将修复补丁上传到文件服务器中，并将修改后的ZTP中间文件重新上传到文件服务器中。

步骤3 空配置重启设备，ZTP重新运行，识别中间文件中的修复补丁信息，自动完成修复过程。

说明

- 修复过程中，在设置好修复补丁后，会立刻检测ZTP是否使能，如果此时ZTP去使能，则恢复下次启动补丁为ZTP开局前设备自带的原始补丁。
- 修复完成后，ZTP设置了下次启动大包、补丁和配置文件，如果此时ZTP去使能，则下次启动补丁回退到修复补丁，不会回退为开局前设备自带补丁。

----结束

开启 ZTP 功能

背景信息

为了使设备空配置启动时能够自动执行ZTP流程，需要开启设备的ZTP功能。

操作步骤

步骤1 执行命令**set ztp enable**，配置设备下次空配置启动时执行ZTP流程。

若用户希望设备空配置启动时不执行ZTP流程，也可以使用命令**set ztp disable**关闭设备的ZTP功能。

步骤2 执行命令**display ztp status**，可以查看设备下次空配置启动是否执行ZTP流程。

----结束

检查配置结果

操作步骤

步骤1 设备启动完成后，登录设备并通过命令**display startup**查看设备的启动文件是否与要求的一致。

步骤2 执行命令**display ztp status**查看设备是否是通过ZTP完成部署。

步骤3 如果设备没有完成自动配置，可以通过设备上保存的ZTP日志查看出错原因。

设备执行ZTP流程的信息会被保存在cfcard:/ztp目录下，文件名为：ztp_年月日时分秒.log。

----结束

1.1.6.2.4 配置举例

介绍通过DHCP实现自动部署的配置示例，示例中包括组网需求、配置注意事项和配置思路等。

配置通过 DHCP 实现 ZTP 自动部署示例

组网需求

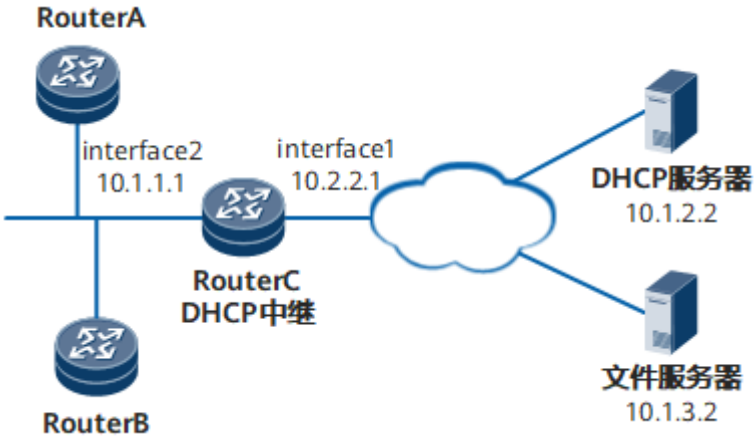
如图1-29所示，某网络中新增两台空配置设备RouterA和RouterB，连接到现网设备RouterC上。RouterC作为RouterA和RouterB的出口网关。RouterC与DHCP服务器、文件服务器之间路由可达。

用户希望空配置的RouterA和RouterB在上电启动后能够自动加载相应的系统软件和配置文件，完成开局部署，以降低现场配置的人力、时间成本。

图 1-29 配置 ZTP 组网图

说明

本例中interface1，interface2分别表示GE1/0/1，GE1/0/2。



配置思路

采用如下的思路配置：

1. 配置文件服务器，将文件服务器作为SFTP Server，存放中间文件及系统软件、配置文件。

   说明

   使用FTP协议存在安全风险，建议使用SFTP进行文件传输。

2. 编辑Python、ini或者cfg格式的中间文件，使不同的设备能够通过中间文件获取相应的系统软件和配置文件。

3. 配置DHCP服务器和中继，使空配置设备可以获得DHCP服务器发送的DHCP信息。
4. 将RouterA和RouterB上电，启动ZTP流程。

操作步骤

步骤1 配置文件服务器

- 使用设备作为文件服务器。
- 使用第三方服务器作为文件服务器，配置的具体方法请参见第三方服务器的操作指导，设置PC上SFTP的工作目录为D:\ztp。文件服务器配置完成后，将设备需要加载的系统软件和配置文件放在D:\ztp目录下。

步骤2 编辑中间文件

请按照 **编辑中间文件**中的要求编辑中间文件，这里以cfg为例，文件名称为 **ztp_script.cfg**，内容请参见 **cfg格式的中间文件**。

中间文件编辑完成后存放至文件服务器的D:\ztp目录下。

步骤3 配置DHCP服务器

配置DHCP服务器分配给客户端的IP地址池，并参照**表1-91**配置DHCP服务器Option选项的值。具体的配置方法请参见相应的产品资料对应DHCP服务器配置相关章节。

表 1-91 DHCP 服务器 Option 选项取值

Option编号	含义	取值
1	IP地址的子网掩码	255.255.225.0
3	DHCP客户端的出口网关	10.1.1.1
67	文件服务器地址及中间文件名	sftp://client001:YsHsjx_202206@10.1.3.2/ztp_script.cfg

设置DHCP服务器的IP地址及网关，要求能够与RouterA、RouterB的网关之间路由可达。

步骤4 配置DHCP中继

配置RouterC的DHCP中继功能，同时配置RouterC与RouterA、RouterB相连的接口IP地址为10.1.1.1，作为RouterA、RouterB的缺省网关。

```
<HUAWEI> system-view
[~HUAWEI] sysname RouterC
[*HUAWEI] commit
[~RouterC] interface GigabitEthernet 1/0/1
[~RouterC-GigabitEthernet1/0/1] ip address 10.2.2.1 255.255.255.0
[*RouterC-GigabitEthernet1/0/1] undo shutdown
[*RouterC-GigabitEthernet1/0/1] commit
[~RouterC-GigabitEthernet1/0/1] quit
[~RouterC] interface GigabitEthernet 1/0/2
[~RouterC-GigabitEthernet1/0/2] ip address 10.1.1.1 255.255.255.0
[*RouterC-GigabitEthernet1/0/2] dhcp select relay
[*RouterC-GigabitEthernet1/0/2] ip relay address 10.1.2.2
```



```
[*RouterC-GigabitEthernet1/0/2] undo shutdown
[*RouterC-GigabitEthernet1/0/2] commit
[~RouterC-GigabitEthernet1/0/2] quit
```

步骤5 将RouterA、RouterB上电，启动ZTP流程

步骤6 验证配置结果

设备启动完成后，可以登录到设备后执行命令**display startup**查看设备当前的系统软件、配置文件是否与预期的一致。以RouterA为例。

```
<RouterA> display startup
MainBoard:
  Configured startup system software:       cfcard:/V800R023C00SPC500B140_0424_new.cc
  Startup system software:                cfcard:/V800R023C00SPC500B140_0424_new.cc
  Next startup system software:             cfcard:/V800R023C00SPC500B140_0424_new.cc
  Startup saved-configuration file:         cfcard:/vrpcfg.cfg
  Next startup saved-configuration file:    cfcard:/vrpcfg.cfg
  Startup paf file:                       default
  Next startup paf file:                  default
  Startup patch package:                  cfcard:/NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E
XV800R023C00SPC500.PAT
  Next startup patch package:             cfcard:/NetEngine 8100 X, NetEngine 8000 X, NetEngine 8000E
XV800R023C00SPC500.PAT
```

----结束

配置文件

ztp_script.cfg文件

说明

下列文件中的SHA256校验码只是举例，配置时以实际计算的值为准。

```
#sha256sum="fffcd63f5e31f0891a0349686969969c1ee429dedeaf7726ed304f2d08ce1bc7"
fileserver=sftp://username:password@hostname:port/path/;
mac=00e0-fc12-3456;esn=2102351931P0C3000154;devicetype=DEFAULT;system-
version=V800R023C00SPC500;boot_python_file=V800R023C00SPC500.py;system-
software=V800R023C00SPC500.cc;system-config=V800R023C00SPC500.cfg;system-
pat=V800R023C00SPC500SPH001.PAT;
```

vrpcfg.cfg文件

说明

下列文件以配置接口IP和路由为例，实际使用中根据需要进行修改。

```
#
sysname HUAWEI
#
ip vpn-instance __LOCAL_OAM_VPN__
 ipv4-family
#
interface Ethernet0/0/0
 undo shutdown
 ip binding vpn-instance __LOCAL_OAM_VPN__
 ip address 192.168.130.10 255.255.255.0
#
ip route-static vpn-instance __LOCAL_OAM_VPN__ 0.0.0.0 0.0.0.0 192.168.130.20
#
```

RouterC的配置文件

```
#
sysname RouterC
#
interface GigabitEthernet1/0/1
```



```
 undo shutdown
 ip address 10.2.2.1 255.255.255.0
#
interface GigabitEthernet1/0/2
 undo shutdown
 ip address 10.1.1.1 255.255.255.0
 dhcp select relay
 ip relay address 10.1.2.2
#
return
```

