H3C 交换机 NETCONF API 二次开发指南

前 言

本文档主用来指导用户使用 NETCONF 客户端配置和管理设备。

前言部分包含如下内容：

读者对象

本手册主要适用于如下工程师：

- 具有一定 XML 和 NETCONF 技术基础的网络规划人员
- 负责网络配置和维护，且具有一定 XML 和 NETCONF 技术基础的网络管理员

本书约定

1. 命令行格式约定

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

2. 各类标志

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

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

窍门	配置、操作、或使用设备的技巧、小窍门。

3. 图标约定

本书使用的图标及其含义如下：

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

4. 示例约定

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

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

资料意见反馈

如果您在使用过程中发现产品资料的任何问题，可以通过以下方式反馈：

E-mail：info@h3c.com

感谢您的反馈，让我们做得更好！

目 录

1 NETCONF 协议介绍

NETCONF（Network Configuration Protocol，网络配置协议）是一种基于 XML 的网络管理协议，它提供了一种可编程的、对网络设备进行配置和管理的方法。用户可以通过该协议设置属性、获取属性值、获取统计信息等。这使得它在第三方软件的开发上非常便利，很容易开发出在混合不同厂商、不同设备的环境下的特殊定制的网管软件。

1.1 NETCONF协议结构

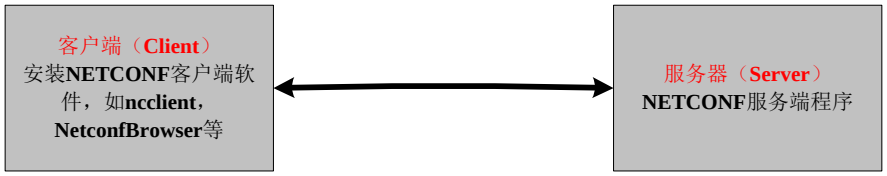
NETCONF 协议采用分层结构，分为内容层（Content）、操作层（Operations）、RPC（Remote Procedure Call，远程调用）层和通信协议层（Transport Protocol）等。

表1-1 XML 分层与 NETCONF 分层模型对应关系

NETCONF 分层	XML 分层	说明
内容层	配置数据、状态数据、统计信息等	被管理对象的集合，可以是配置数据、状态数据、统计信息等 NETCONF协议具体可读写的数据请参见《NETCONF XML API 手册》
操作层	<get>,<get-config>,<edit-config>…	在RPC中应用的基本的原语操作集，这些操作组成NETCONF的基本能力 NETCONF全面地定义了对被管理设备的各种基础操作，设备支持的操作请参见“1.6 Comware支持的协议操作”
RPC层	<rpc>,<rpc-reply>	为RPC模块的编码提供了简单的、传输协议无关的机制。通过使用<rpc>和<rpc-reply>元素分别对NETCONF请求和响应数据（即操作层和内容层的内容）进行封装
通信协议层	非FIPS模式下： Console/Telnet/SSH/HTTP/HTTPS/TLS FIPS模式下： Console/SSH/HTTPS/TLS	为NETCONF提供面向连接的、可靠的、顺序的数据链路。 非FIPS模式下： • NETCONF 支持 Telnet、SSH 和 Console 等 CLI 登录方式/协议，即 NETCONF over SSH、NETCONF over Telnet 和 NETCONF over Console • NETCONF 支持 HTTP 和 HTTPS 协议，即 NETCONF over HTTP 和 NETCONF over HTTPS • NETCONF 支持封装成 SOAP（Simple Object Access Protocol，简单对象访问协议）报文后通过 HTTP 或 HTTPS 协议传输，即 NETCONF over SOAP over HTTP 和 NETCONF over SOAP over HTTPS FIPS模式下： • NETCONF 支持 SSH 和 Console 等 CLI 方式/协议，即 NETCONF over SSH 和 NETCONF over Console • NETCONF 支持 HTTPS 登录协议，即 NETCONF over HTTPS • NETCONF 支持封装成 SOAP 报文后通过 HTTPS 协议传输，即 NETCONF over SOAP over HTTPS

1.2 NETCONF基本网络架构

图1-1 NETCONF 基本网络架构



NETCONF 网络主要由 NETCONF 客户端和 NETCONF 服务器组成：

- 客户端上需要安装 NETCONF 客户端软件（如 ncclient、NetconfBrowser），或运行基于 SOAP 请求的脚本或程序。客户端的主要作用为：
  - 通过 NETCONF 协议对网络设备进行配置和管理。
  - 主动查询网络设备的状态。
  - 接收网络设备主动发送的告警和事件，以获知网络设备的当前状态。
- 服务器（即待管理网络设备）上需要运行 NETCONF 服务端程序，即支持 NETCONF 功能。服务器主要用于响应客户端的请求，并在发生故障或事件时，主动通告给客户端。

1.3 NETCONF报文格式

1.3.1 NETCONF

NETCONF 命令必须符合 XML 语言的基本格式，格式遵循 RFC 4741。

NETCONF 报文分为请求和应答报文两种。NETCONF 请求和应答报文的格式有所不同。

执行 NETCONF 请求前，需要先校验 NETCONF 报文的数据合法性。如果校验失败，则会向客户端报错。其中，数据合法性校验通过 XML Schema 的方式完成。

1. 请求格式（能力集 Base 1.0）

Comware NETCONF 请求遵循 NETCONF 协议，其格式如下：

```
<rpc message-id ="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <operation>
</rpc>
```

其中*<operation>*为“1.6 Comware 支持的协议操作”中列举的支持的操作类型：

- 对于 get 系列操作，filter 元素下为 H3C 自定义部分，以 top 元素为起点。以 get-config 操作为例，报文格式为：

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>
    <source>
      <running/>
    </source>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
        模块信息
      </top>
```

```
    </filter>
  </get-config>
</rpc>
```

- 对于 edit-config 操作，config 元素下为 H3C 自定义部分，以 top 元素为起点。以 edit-config 操作为例，报文格式为：

```
<rpc message-id =”101” xmlns=”urn:ietf:params:xml:ns:netconf:base:1.0”>
  <edit-config>
    <target>
      <running/>
    </target>
    <config xmlns:xc=”urn:ietf:params:xml:ns:netconf:base:1.0”>
      <top xmlns=”http://www.h3c.com/netconf/config:1.0”>
        模块信息
      </top>
    </config>
  </edit-config>
</rpc>
```

说明

NETCONF 可操作的数据项，即 H3C 自定义部分的语法结构，请参见《H3C Comware 7 NETCONF XML API Reference》文档。

如下为一个 NETCONF 请求报文示例，用于获取设备上所有接口的所有参数：

```
<?xml version="1.0" encoding="utf-8"?>
<rpc message-id ="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-bulk>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        <Ifmgr>
          <Interfaces>
             <Interface/>
          </Interfaces>
        </Ifmgr>
      </top>
    </filter>
  </get-bulk>
</rpc>
]]>]]>
```

提示

在 XML 视图下进行 NETCONF 配置时，XML 报文最后需要添加“]]>]]>”，否则设备无法识别。为方便识别 XML 格式，本手册中，除上述举例外，均未添加此结束符。实际操作中，请自行添加。

2. 应答格式（能力集 Base 1.0）

统一为 RFC 4741 定义的<rpc-reply>，如：

```
<rpc-reply xmlns=”urn:ietf:params:xml:ns:netconf:base:1.0” message-id=”101”>
  <ok/>
</rpc-reply>
```

1.3.2 NETCONF over SOAP

通过 NETCONF over SOAP 访问设备，NETCONF 报文会封装在 SOAP 报文的 BODY 元素里，这些报文除了需要遵循纯 NETCONF 报文的规则外，还需要遵循以下规则：

- SOAP 消息必须用 XML 来编码。
- SOAP 消息必须使用 SOAP Envelope 命名空间和 SOAP Encoding 命名空间。
- SOAP 消息不能包含 DTD（Document Type Definition，文件类型定义）引用。
- SOAP 消息不能包含 XML 处理指令。

NETCONF over SOAP 请求和应答报文的格式有所不同。

1. 请求格式（SOAP）

如下为一个 NETCONF over SOAP 请求报文示例，用于获取设备上所有接口的所有参数：

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Header>
    <auth:Authentication env:mustUnderstand="1"
xmlns:auth="http://www.h3c.com/netconf/base:1.0">
      <auth:AuthInfo>100002ea7bd8f3bbb727c5eff9b246d85be6</auth:AuthInfo>
    </auth:Authentication>
  </env:Header>
  <env:Body>
    <rpc message-id ="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <get>
        <filter type="subtree">
          <top xmlns="http://www.h3c.com/netconf/data:1.0">
            <Ifmgr>
              <Interfaces>
                <Interface/>
              </Interfaces>
            </Ifmgr>
          </top>
        </filter>
      </get>
    </rpc>
  </env:Body>
</env:Envelope>
```

2. 应答格式（SOAP）

```
<env:Envelope xmlns:env=”http://schemas.xmlsoap.org/soap/envelope/”>
  <env:Header>
    <auth:Authentication env:mustUnderstand=”true”
xmlns:auth=”http://www.h3c.com/netconf/base:1.0”>
```

```
      <auth:AuthInfo>10027c2abebdb482633f847102fbc890d22a</auth:AuthInfo>
    </auth:Authentication>
  </env:Header>
  <env:Body>
    <rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <data>
        <top xmlns="http://www.h3c.com/netconf/config:1.0">
          <Syslog>
            <LogBuffer>
              <BufferSize>527</BufferSize>
            </LogBuffer>
            <LogHosts>
              <Host>
                <Address>1.1.1.1</Address>
                <VRF/>
                <Port>123</Port>
                <Facility>152</Facility>
              </Host>
              <Host>
                <Address>1.1.1.1</Address>
                <VRF>a</VRF>
                <Port>123</Port>
                <Facility>152</Facility>
              </Host>
              <Host>
                <Address>1.1.1.1</Address>
                <VRF>b</VRF>
                <Port>123</Port>
                <Facility>152</Facility>
              </Host>
              <Host>
                <Address>1.1.1.1</Address>
                <VRF>c</VRF>
                <Port>123</Port>
                <Facility>152</Facility>
              </Host>
            </LogHosts>
          </Syslog>
        </top>
      </data>
    </rpc-reply>
  </env:Body>
</env:Envelope>
```

1.3.3 NETCONF 报文的其他格式要求

1. CDATA 节点

Comware NETCONF 中，CLI 操作返回的数据因为是命令行的原始输出，故统一使用 CDATA 标记来封装，以防止客户端程序解析 XML 出现错误。使用时，客户端需要从 CDATA 中抽取返回结果。

2. 注释

Comware NETCONF 中，XML 请求对注释没有要求，可以随意添加合法的注释。XML 应答中不包含 XML 注释。

3. 空格

请求中，如果要保持两端的空格字符，必须对最开始和最后一个空格进行转义，转义遵循 W3C XML 规范。如果不进行转义，则两端连续的空格会被忽略。

应答元素的值中，两端最外侧的空格会被 Comware NETCONF 转义。

4. 使用转义

如果需要使用一个当前编码不能支持的字符时，可以使用标准的 W3C XML 规范来进行转义，比如	代表一个 Tab 键。

5. XML 编码

Comware NETCONF 支持使用 gb2312、GB18030、utf-8、utf-16、utf-16BE、utf-16LE、utf-32、utf-32BE、utf-32LE 进行传输。默认情况下，如果请求中不指定 XML 传输格式，将使用 utf-8 作为默认编码，转换编码过程中，如果出现类似半个汉字之类无法转换的情况，将使用?来替代无法转换的字符。

1.4 Comware NETCONF遵循的协议

NETCONF 相关的协议规范如表 1-2 所示。

表1-2 NETCONF 相关协议规范

协议号	协议名称	支持情况
RFC 4741	NETCONF Configuration Protocol	支持
RFC 4742	Using the NETCONF Configuration Protocol over Secure Shell (SSH)	支持
RFC 4743	Using NETCONF over the Simple Object Access Protocol (SOAP)	支持
RFC 4744	Using the NETCONF Protocol over the Blocks Extensible Exchange Protocol (BEEP)	不支持
RFC 5277	NETCONF Event Notifications	支持
RFC 5381	Experience of Implementing NETCONF over SOAP	支持
RFC 5539	NETCONF over Transport Layer Security (TLS)	不支持
RFC 5717	Partial Lock Remote Procedure Call (RPC) for NETCONF	不支持
RFC 6020	YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)	支持
RFC 6021	Common YANG Data Types	支持

协议号	协议名称	支持情况
RFC 6022	YANG Module for NETCONF Monitoring	支持
RFC 6087	Guidelines for Authors and Reviewers of YANG Data Model Documents	支持
RFC 6241	Network Configuration Protocol	支持
RFC 6242	Using the NETCONF Protocol over Secure Shell (SSH)	支持
RFC 6243	With-defaults Capability for NETCONF	不支持
RFC 7950	The YANG 1.1 Data Modeling Language	支持

1.5 Comware NETCONF支持的能力集

Comware NETCONF 支持的能力集如表 1-3 所示。除 RFC 中定义的标准能力集外，Comware NETCONF 还定义了一些私有能力集。私有能力集没有对应的 RFC，因此，私有能力集的 RFC 列取值为 “-”。

表1-3 Comware NETCONF 的能力集支持情况

RFC	能力集	支持情况
RFC 4741	Base 1.0 urn:ietf:params:netconf:base:1.0	支持
RFC 4741	Writable-Running urn:ietf:params:netconf:capability:writable-running:1.0	支持
RFC 4741	Candidate Configuration urn:ietf:params:netconf:capability:candidate:1.0	不支持
RFC 4741	Confirmed Commit 1.0 urn:ietf:params:netconf:capability:confirmed-commit:1.0	不支持
RFC 4741	Rollback on Error urn:ietf:params:netconf:capability:rollback-on-error:1.0	支持
RFC 4741	Validate 1.0 urn:ietf:params:netconf:capability:validate:1.0	支持
RFC 4741	Distinct Startup urn:ietf:params:netconf:capability:startup:1.0	不支持
RFC 4741	URL urn:ietf:params:netconf:capability:url:1.0?scheme={name,...}	不支持
RFC 4741	XPATH urn:ietf:params:netconf:capability:xpath:1.0	不支持
RFC 5277	Interleave urn:ietf:params:netconf:capability:interleave:1.0	支持
RFC 6022	Yang urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring	支持
RFC 6241	Validate 1.1	支持

RFC	能力集	支持情况
	urn:ietf:params:netconf:capability:validate:1.1	
RFC 6241	Confirmed Commit 1.1 urn:ietf:params:netconf:capability:confirmed-commit:1.1	不支持
RFC 6242	Base 1.1 urn:ietf:params:netconf:base:1.1	不支持
-	h3c-netconf-ext:1.0 urn:h3c:params:netconf:capability:h3c-netconf-ext:1.0	支持
-	h3c-save-point:1.0 urn:h3c:params:netconf:capability:h3c-save-point:1.0	支持
-	h3c-name2index:1.1 urn:h3c:params:netconf:capability:h3c-name2index:1.1	支持
-	not-need-top:1.0 urn:h3c:params:netconf:capability:not-need-top:1.0	支持
-	module-specified-namespace:1.0 urn:h3c:params:netconf:capability:module-specified-namespace:1.0	支持
-	h3c-lightrollback:1.0 urn:h3c:params:netconf:capability:h3c-lightrollback:1.0	支持
-	full-replace:1.0 urn:h3c:params:netconf:capability:full-replace:1.0	支持

1.6 Comware支持的协议操作

Comware NETCONF 支持的协议操作如表 1-4 所示。

表1-4 Comware NETCONF 对协议操作的支持情况

RPC 操作名称	说明	支持情况
action	修改、获取运行统计信息之类的动作	支持
CLI	作为NETCONF的补充，执行NETCONF尚未提供的命令	支持
commit	把候选库的配置提交到生效库	不支持
cancel-commit	取消已提交到运行数据库中的候选数据库配置	不支持
close-session	关闭当前会话	支持
copy-config	配置库拷贝	不支持
create-subscription	事件订阅	支持
delete-config	删除配置	不支持
discard-changes	放弃变更	不支持
edit-config	配置修改	支持

RPC 操作名称	说明	支持情况
get	获取设备数据	支持
get/filter/netconf	获取系统支持的事件流	支持
get/filter/netconf-state	获取系统NETCONF状态信息	支持
get-bulk	批量获取设备数据	支持
get-bulk-config	批量获取设备配置数据	支持
get-config	获取设备配置数据	支持
get-sessions	获取NETCONF会话信息	支持
kill-session	关闭非当前会话	支持
load	配置加载	支持
lock	加锁NETCONF	支持
rollback	配置回滚	支持
save	配置保存	支持
unlock	解锁NETCONF	支持
validate	NETCONF请求的XML语法检查	支持
get-schema	获取YANG/Schema模型文件	有限支持 目前，设备只为部分模块提供了YANG文件
cancel-subscription	取消订阅	支持

说明

action、CLI、get-bulk、get-bulk-config、get-sessions、load、save、rollback 和 cancel-subscription 为 Comware 扩展的私有操作。Comware 还为 get 和 get-config 操作增加了 count 属性。各操作的详细介绍，请参见“4 执行 Comware NETCONF 请求”。

2 使用 NETCONF 配置和管理设备

2.1 使用NETCONF配置和管理设备的通用流程

管理员通过 NETCONF 客户端配置和管理设备时，通常需要按照如下流程来操作：

(1) 选择 NETCONF 客户端并确定客户端对设备的访问方式

Comware 支持如下 NETCONF 访问方式：

- NETCONF over Telnet
- NETCONF over Console
- NETCONF over SSH
- NETCONF over SOAP over HTTP
- NETCONF over SOAP over HTTPS

需要根据 NETCONF 客户端的实际组网环境和需求，选择合适的访问方式。

(2) 在设备上为 NETCONF 用户设置权限

不同访问方式需要的用户权限不同：

- NETCONF over SOAP over HTTP 访问方式，需要为用户设置 HTTP 访问权限。
- NETCONF over SOAP over HTTPS 访问方式，需要为用户设置 HTTPS 访问权限。
- NETCONF over SSH 访问方式，需要为用户设置 SSH 访问权限。

(3) 在设备上进行访问方式相关配置

依据需要使用的访问方式，打开对应的功能。

(4) 准备 NETCONF 请求使用的 XML 消息

根据需访问模块的 API 文档，构造 NETCONF XML 请求。

(5) 准备 NETCONF 客户端

按照 RFC 4741 的描述准备客户端软件、基于 SOAP 请求的脚本或程序，使其能发送 hello 和 close-session。具体方法请参见客户端软件的操作指导等。

(6) 使用 NETCONF 客户端连接到设备

依据需要使用的访问方式，使用准备好的 NETCONF 客户端连接到设备。

(7) 测试 NETCONF 客户端和设备的连通性

测试建立连接（hello）和断开连接（close-session）的能力，使后续工作能够正常进行。

(8) 验证准备的 NETCONF 请求 XML 报文的正确性

使用工具或命令行 XML 方式来验证 XML 报文的正确性。

(9) 将请求集成到客户端脚本或程序

把测试好的 XML 报文集成到客户端程序中。

(10) （可选）最佳实践建议

通过 NETCONF 客户端配置和管理设备时，在某些特殊情况下需要进行特殊处理。例如，容错处理、大数据量处理、并发处理等。

(11) （可选）常见问题及处理方法

完成上述操作后，即可使用 NETCONF 客户端程序执行 NETCONF 操作管理设备。Comware 支持操作的详细介绍，请参见“4 执行 Comware NETCONF 请求”。

2.2 选择NETCONF客户端并确定客户端对设备的访问方式

2.2.1 NETCONF 配置方式简介

用户可通过以下方式来使用 NETCONF 配置和管理设备：

- 通过 Telnet、SSH 或 Console 登录到设备并进入 XML 视图，将合法的 NETCONF 报文（报文格式请参见“1.3.1 NETCONF”）直接拷贝、粘贴到命令行中执行，即可实现对设备的配置和管理。该方式一般用于研发和测试环境。
- 使用配置工具与设备建立连接并向设备下发 NETCONF 指令来实现对设备的配置和管理。配置工具支持以下两种传输方式：
  - NETCONF over SSH 传输方式：配置工具采用该方式与设备建立 SSH 连接后，可下发 NETCONF 格式报文（报文格式请参见“1.3.1 NETCONF”）配置和管理设备。
  - NETCONF over SOAP 传输方式：该方式通过 HTTP/HTTPS 与设备建立连接，并将 NETCONF 指令用 SOAP 封装成通用格式的报文后发送给设备，报文格式请参见“1.3.2 NETCONF over SOAP”。使用该方式前设备上必须开启 NETCONF over SOAP 功能。

客户可以根据需要选择 NETCONF 传输方式，开发和定制自己专用的配置工具软件，也可以使用第三方开发的配置工具。第三方的配置工具有：

- 采用 NETCONF over SSH 传输方式配置工具：NETCONF Browser 等。
- 采用 NETCONF over SOAP 传输方式配置工具：SOAP UI 等。

关于各配置工具的使用方法，请参见配置工具的使用指导文档。

2.2.2 访问方式介绍

H3C 设备目前支持如下访问方式：

非 FIPS 模式下：

- NETCONF over SSH
- NETCONF over Telnet
- NETCONF over Console
- NETCONF over SOAP over HTTP
- NETCONF over SOAP over HTTPS

FIPS 模式下：

- NETCONF over SSH
- NETCONF over Console
- NETCONF over SOAP over HTTPS

使用不同配置方式时，需要使用的访问方式不同：

- 使用 Telnet 或 SSH 登录到设备，进入 XML 视图执行 NETCONF 配置时，需要使用 NETCONF over Telnet 访问方式。

- 使用 Console 登录到设备，进入 XML 视图执行 NETCONF 配置时，需要使用 NETCONF over Console 访问方式。
- 使用 SSH 配置工具（例如 NETCONF Browser）执行 NETCONF 配置时，需要使用 NETCONF over SSH 访问方式。
- 使用 SOAP 配置工具下发 NETCONF 指令配置设备时，需要使用 NETCONF over SOAP over HTTP 或 NETCONF over SOAP over HTTPS 方式。

2.3 在设备上为NETCONF用户设置权限

使用配置工具与设备建立 NETCONF 连接前，需要确保 NETCONF 用户具有对应的操作权限。

2.3.1 确定 NETCONF 用户需要的权限

Comware 通过 RBAC（Role Based Access Control，基于角色的访问控制）实现基于角色的用户访问权限控制。RBAC 对用户权限的控制包括用户角色规则和资源控制策略两个方面：

- 用户角色规则用来控制用户对系统功能的操作权限。例如，定义用户角色规则允许用户配置 A 功能，或禁止用户配置 B 功能。
- 资源控制策略用来控制用户对系统资源的操作权限。例如，定义资源控制策略允许用户操作 VLAN 10。

在用户角色规则方面，NETCONF 用户角色需要具有如下权限：

- 执行 NETCONF 操作需要具有执行 XML 元素 NETCONF RPC 节点的权限，不同 NETCONF 操作需要的权限不同，具体请参见表 2-1。
- 执行 NETCONF 操作内容的权限，即用户执行 XML 元素（具体模块或模块中表的 Xpath）的权限。例如，配置用户具有执行接口模块 XML 元素的权限时，需要执行 **rule** *number* **permit read write execute xml-element ifmgr/**命令。

在资源策略方面，可以根据实际需要配置用户角色操作接口、VLAN、VPN、安全域的权限。缺省情况下，用户具有操作所有资源的权限。

不同用户角色的操作权限有所不同：

- 缺省用户角色 network-admin、mdc-admin、context-admin 具有操作对应设备/MDC/Context 的所有功能和资源（除安全日志文件管理相关命令 **display security-logfile summary**、**info-center security-logfile directory**、**security-logfile save** 之外）的权限。如果用户所属角色是这几种，则只需要指定用户角色为 network-admin、mdc-admin 或 context-admin 即可。
- 缺省用户角色 network-operator、context-operator 和 mdc-operator 默认只有读权限和部分可执行权限，必须事先配置后才有对应的操作权限。
- 其他用户角色必须事先配置后才有对应的操作权限。

NETCONF 操作需要权限的详细介绍请见表 2-1。

表2-1 执行 NETCONF 操作需要配置的权限

执行的 NETCONF 操作	需要配置的权限节点	需要配置的权限
建立NETCONF会话	不涉及	执行**xml**命令的权限 对于NETCONF over Telnet、

执行的 NETCONF 操作	需要配置的权限节点	需要配置的权限
		NETCONF over Console访问方式，用户必须具有执行**xml**命令行的权限，否则无法进入XML视图
向设备订阅事件	rpc/create-subscription	execute
给当前配置加锁	rpc/lock	execute
给当前配置解锁	rpc/unlock	execute
使用<get>获取信息	rpc/get	read
使用<get>获取NETCON状态信息	rpc/get/filter/netconf-state	read
使用<get-bulk>获取信息	rpc/get-bulk	read
使用<get-config>获取配置信息	rpc/get-config	read
使用<get-bulk-config>获取配置信息	rpc/get-bulk-config	read
使用<get-schema>获取yang文件信息	rpc/get-schema	read
<edit-config>编辑指定模块数据	rpc/edit-config	write
执行一个<action>操作	rpc/action	execute
配置保存	rpc/save	write
配置回滚	rpc/rollback	write
配置加载	rpc/load	write
命令行操作	rpc/CLI	write
获取会话信息	rpc/get-sessions	read
关闭另一个会话	rpc/kill-session	execute
执行语法验证	rpc/validate	read
配置回滚点	rpc/save-point	write
候选库配置提交	rpc/commit	execute
取消已提交的配置	rpc/cancel-commit	execute
复制数据库	rpc/copy-config	execute
删除数据库	rpc/delete-config	execute
删除候选库配置	rpc/discard-changes	execute
预配置	rpc/config-provisioned	execute
退出XML视图	不涉及	执行**xml**命令的权限
取消当前订阅事件	rpc/cancel-subscription	execute

2.3.2 定义 NETCONF 用户角色规则

(1) 进入系统视图。

system-view

(2) 创建用户角色，并进入用户角色视图。

role name *role-name*

(3) 请根据需要配置用户操作权限的规则。

- 配置用户具有执行 XML 元素 NETCONF RPC 节点的权限。

  rule *number* **permit** { **execute** | **read** | **write** } * **xml-element rpc/**

  通过 **rule** *number* **permit** { **execute** | **read** | **write** } * **xml-element rpc/?**可以查看具体操作的列表。不指定具体操作时，表示所有操作。

- 配置用户执行 XML 元素具体模块的权限。

  rule *number* { **deny** | **permit** } { **execute** | **read** | **write** } * **xml-element** [*module-xpath*]

  通过 **rule** *number* { **deny** | **permit** } { **execute** | **read** | **write** } * **xml-element ?** 可以查看具体模块列表，通过 **rule** *number* { **deny** | **permit** } { **execute** | **read** | **write** } * **xml-element** *module-name/?* 可以查看模块中具体表的列表。不指定具体模块和表时表示所有模块/所有表。

2.3.3 配置 NETCONF 用户

1. 功能简介

本配置用来为 NETCONF 用户指定如下属性：

- 根据 NETCONF 访问方式，指定正确的服务类型：使用 NETCONF over SSH 访问方式时，用户的服务类型为 SSH；使用 NETCONF over SOAP over HTTP 访问方式时，用户的服务类型为 HTTP；使用 NETCONF over SOAP over HTTPS 访问方式时，用户的服务类型为 HTTPS。
- 为用户指定用户角色，使其具有所需权限。

本文以本地认证为例，介绍配置过程。使用远程认证的配置方式请参见产品对应版本的 AAA 配置指导。

2. 配置步骤

(1) 进入系统视图。

system-view

(2) 添加设备管理类本地用户，并进入设备管理类本地用户视图。

local-user *user-name* **class manage**

(3) 设置本地用户的密码。

（非 FIPS 模式）

password [{ **hash** | **simple** } *string*]

（FIPS 模式）

password

在非 FIPS 模式下，可以不为本地用户设置密码；在 FIPS 模式下，必须且只能通过交互式方式设置明文密码，否则用户的本地认证不能成功。

(4) 设置本地用户可以使用的服务类型。请根据访问方式选择其中一项进行配置

- 使用 NETCONF over SSH 访问方式：

service-type ssh

- 使用 NETCONF over SOAP over HTTP 或 NETCONF over SOAP over HTTPS 访问方式：

  （非 FIPS 模式）

  service-type { **http** | **https** } *

  （FIPS 模式）

  service-type https

缺省情况下，本地用户不能使用任何服务类型。

(5) 设置本地用户的角色。

authorization-attribute user-role *role-name*

2.4 在设备上进行访问方式相关配置

2.4.1 配置 NETCONF over SSH 访问方式

1. 配置 SSH 登录

以下配置步骤只介绍采用 **password** 方式认证 SSH 客户端的配置方法，**publickey** 方式的配置方法及 SSH 的详细介绍，请参见“安全配置指导”中的“SSH”。

(1) 进入系统视图。

system-view

(2) 生成本地密钥对。

（非 FIPS 模式）

public-key local create { **dsa** | **ecdsa** [**secp192r1** | **secp256r1** | **secp384r1** | **secp521r1**] | **rsa** } [**name** *key-name*]

（FIPS 模式）

public-key local create { **dsa** | **ecdsa** [**secp256r1** | **secp384r1** | **secp521r1**] | **rsa** } [**name** *key-name*]

(3) （可选）创建 SSH 用户，并指定 SSH 用户的服务类型为 NETCONF，认证方式为 password。

ssh user *username* **service-type netconf authentication-type password**

(4) 进入 VTY 用户线或 VTY 用户线类视图。

- 进入 VTY 用户线视图。

  line vty *first-number* [*last-number*]

- 进入 VTY 用户线类视图。

  line class vty

(5) 配置 VTY 用户线的认证方式为 scheme 方式。

（非 FIPS 模式）

authentication-mode scheme

缺省情况下，VTY 用户线的认证方式为 **password** 方式。

（FIPS 模式）

authentication-mode scheme

缺省情况下，VTY 用户线的认证方式为 **scheme** 方式。

2. 配置 NETCONF over SSH

(1) 进入系统视图。

system-view

(2) 开启 NETCONF over SSH。

netconf ssh server enable

缺省情况下，NETCONF over SSH 处于关闭状态。

(3) 配置 NETCONF over SSH 的监听端口。

netconf ssh server port *port-number*

缺省情况下，NETCONF over SSH 的监听端口为 830。

(4) （可选）开启 NETCONF 日志功能。

netconf log source { **all** | { **agent** | **soap** | **web** } * } { **protocol-operation** { **all** | { **action** | **config** | **get** | **set** | **session** | **syntax** | **others** } * } | **row-operation** | **verbose** }

缺省情况下，NETCONF 日志功能处于关闭状态。

(5) 通过 SSH 配置工具与设备建立 NETCONF over SSH 会话。关于 SSH 配置工具的使用方法，具体参见 SSH 配置工具的配置指导。

2.4.2 配置 NETCONF over SOAP over HTTP（或 HTTPS）访问方式

1. 功能简介

使用 NETCONF over SOAP over HTTP 或 NETCONF over SOAP over HTTPS 访问方式时，配置工具将配置指令封装成 SOAP 报文后通过 HTTP 或 HTTPS 协议传输到设备。

本文仅介绍基本配置，有关 SOAP 报文的 DSCP 优先级、NETCONF 客户端访问控制、NETCONF 用户使用的认证域等相关内容请参见产品对应版本的 NETCONF 配置、NETCONF 命令手册。

2. 开启 NETCONF over SOAP 功能

(1) 进入系统视图。

system-view

(2) 开启 NETCONF over SOAP 功能。

（非 FIPS 模式）

netconf soap { **http** | **https** } **enable**

（FIPS 模式）

netconf soap https enable

缺省情况下，NETCONF over SOAP 处于关闭状态。

(3) （可选）开启 NETCONF 日志功能。

netconf log source { **all** | { **agent** | **soap** | **web** } * } { **protocol-operation** { **all** | { **action** | **config** | **get** | **set** | **session** | **syntax** | **others** } * } | **row-operation** | **verbose** }

缺省情况下，NETCONF 日志功能处于关闭状态。

(4) 通过配置工具与设备建立 NETCONF over SOAP 会话。关于配置工具的使用方法，具体参见配置工具的配置指导。

2.5 准备NETCONF请求使用的XML消息

2.5.1 NETCONF API 文档介绍

提示

Comware NETCONF API 文档描述的是全集，需要结合设备的 XSD 文件来查看当前设备的支持情况。设备可能不支持某些功能、某些表格或某些列。

1. 文档组织

NETCONF API 文档分属于四个命名空间：

- data 命名空间：提供系统的运行状态数据和配置数据，为只读，支持下发 get 和 get-bulk 操作。
- config 命名空间：提供系统的配置数据信息，可读写，支持下发 get-config、get-bulk-config 和 edit-config 操作。
- action 命名空间：通常提供系统非配置类的操作（如 ping、reset 操作），可执行，支持下发 action 操作。
- event 命名空间：提供系统的事件数据信息，支持通过 create-subscription 操作订阅指定类型的事件。

每一个支持 NETCONF 的模块都包括一个或者多个 NETCONF API 文档，分别描述其在 data 命名空间、config 命名空间、action 命名空间和 event 命名空间的功能。文档命名遵循如下格式，其中 XXX 代表模块名：

- data 命名空间：Comware XXX NETCONF XML API Data Reference.docx
- config 命名空间：Comware XXX NETCONF XML API Configuration Reference.docx
- action 命名空间：Comware XXX NETCONF XML API Action Reference.docx
- event 命名空间：Comware XXX NETCONF XML API Event Reference.docx

NETCONF 模块可能还存在一个名为 Comware XXX NETCONF XML API Error Message Reference.docx 的模块错误提示信息参考文档，该文档用来描述模块操作过程中的各种错误。

2. 文档内容说明

每一个 NETCONF API 文档对应一个功能模块，如 ARP、DHCP。每个模块由一个或者多个表组成。表以行和列的形式进行组织：

- 行：表示一个对象实例。
- 列：表示每个对象实例中包含的信息。

以 ARP 表为例，一条 ARP 表项为一行，ARP 表项中的 IP 地址、MAC 地址、所属 VLAN 等为列。

在 NETCONF API 文档中，每个表包含下面几个部分的信息：

- 表名称：提供了从模块开始的路径和最终的表名，如：ACL/Base

- 表的 XML 结构（XML structure）：列举了本表从模块开始的 XML 表示方式，但不包括值信息，例如：

XML structure

```
<VLAN>
  <BatchVlans>
    <CreateVlanList></CreateVlanList>
    <DestroyVlanList></DestroyVlanList>
  </BatchVlans>
</VLAN>
```

- 表描述（Table description）：提供模块名称、表名称、表类型、行名称和约束信息。其中，表类型取值包括：
  - Multi-instance table：多实例表格，表示该表可以包含多行。
  - Single-instance table：单实例表格，表示该表能包括一行。
- 列详细信息（Columns）：提供列名称、列描述、列数据类型和约束条件等信息。列数据类型取值包括：
  - Index：表示该列为表格的索引列。
  - N/A：表示该列不作为索引列。
- 应答的 XML 结构（Responsed XML structure）：列举了本表从模块开始应答消息的 XML 表示方式，但不包括值信息。客户端可以依据返回结果来判断操作执行的效果。仅 action 命名空间中可以返回操作结果的表提供此信息。例如：

Responsed XML structure

```
<VLAN>
  <BatchVlans>
    <CreateVlanList></CreateVlanList>
    <DestroyVlanList></DestroyVlanList>
  </BatchVlans>
</VLAN>
```

- 应答列详细信息（Responsed columns）：用来描述应答 XML 结构中的各列。

2.5.2 使用 NETCONF API 文档构造 NETCONF 请求报文

NETCONF API 文档介绍了 NETCONF 请求报文（参见“1.3 NETCONF 报文格式”）中 H3C 自定义部分的模块信息语法。构造某一请求时，需要先根据操作类型查找对应模块、对应空间的 API 文档来获取需要的 XML 结构，如表 2-2 所示。

表2-2 操作类型与 API 文档对应关系

操作类型	需查询的 API 文档
<get> <get-bulk>	Comware *XXX* NETCONF XML API Data Reference.docx 例如，接口管理请查询《Comware *Ifmgr* NETCONF XML API Data

操作类型	需查询的 API 文档
	Reference.docx》
<get-config> <get-bulk-config> <edit-config>	Comware *XXX* NETCONF XML API Configuration Reference.docx 例如，接口管理请查询《Comware *Ifmgr* NETCONF XML API Configuration Reference.docx》
<action>	Comware *XXX* NETCONF XML API Action Reference.docx 例如，接口管理请查询《Comware *Ifmgr* NETCONF XML API Action Reference.docx》
<create-subscription>	Comware *XXX* NETCONF XML API Event Reference.docx 例如，接口管理请查询《Comware *Ifmgr* NETCONF XML API Event Reference.docx》

获取到所需模块的 NETCONF API 文档后，应当先确认需要的功能是否已经支持。若已经支持，则可以依据真实需求，构造所需报文。一般情况下，可以从对应的表的 XML Structure 部分开始，依据如下步骤构造 NETCONF 请求：

(1) 构造 XML 请求的协议定义部分，即 top 外层的部分。

以 get 操作为例，top 以外的部分为协议定义部分，可以直接从现有请求或者本文档拷贝。

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter>
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        指定模块，子模块，表名，列名
      </top>
    </filter>
  </get>
</rpc>
```

其他操作的协议定义部分格式，请参见“4 执行 Comware NETCONF 请求”。

(2) 构造 top 元素。

top 元素的格式通常为：<top xmlns="http://www.h3c.com/netconf/data:1.0">。其中，http://www.h3c.com/netconf/data:1.0 为 H3C 自定义的命名空间，需要根据需要设置为 data、config 或 action 命名空间，即 data:1.0 也可根据需要设置为 config:1.0 或 action:1.0。

(3) 构造模块操作部分。

拷贝 NETCONF API 文档中表格 XML 结构到“指定模块，子模块，表名，列名”部分，然后依据实际情况删除不需要的列或者为已知列赋值。

2.5.3 使用 XSD 文件构造 NETCONF 请求

目前产品发布时，会同步发布该产品支持的 XSD 文件。用户也可以根据 XSD 文件构造 NETCONF 请求。构造方法与 NETCONF API 类似，本文不再赘述。

2.5.4 使用 YANG 文件构造 NETCONF 请求

Comware NETCONF 请求遵循 NETCONF 协议，可通过 get-schema 操作获取设备上某个 YANG 文件内容，通过获取 netconf-state 操作获取设备上所有 YANG 文件列表，详细介绍请参见“4.2.10 get-schema 操作”和“4.2.11 获取 netconf-state 操作”。

用户也可以根据 YANG 文件构造 NETCONF 请求。构造方法与 NETCONF API 类似，本文不再赘述。

2.6 使用NETCONF客户端连接到设备

2.6.1 选择合适的 NETCONF 访问方式

Comware 目前支持使用 NETCONF over Telnet、NETCONF over Console、NETCONF over SSH、NETCONF over SOAP over HTTP、NETCONF over SOAP over HTTPS 几种方式访问 NETCONF 功能。按其访问方式不同，NETCONF 的访问方式可以分为下面 3 类：

- 客户端使用 SOAP 方式连接到设备：即 NETCONF over SOAP over HTTP 和 NETCONF over SOAP over HTTPS 方式。
- 客户端使用 NETCONF over SSH 访问方式连接到设备：即 NETCONF over SSH 方式。
- 客户端使用 Telnet/SSH/Console 以命令行方式连接到设备：即 NETCONF over Telnet、NETCONF over Console 和 NETCONF over SSH 方式。

使用前，需要先确定 NETCONF 客户端选取的访问方式，并在设备上使能对应的功能。使能对应功能后，NETCONF 客户端即可使用对应的连接手段访问设备的 NETCONF 功能。

2.6.2 客户端使用 SOAP 方式连接到设备

配置完设备后，客户端通过 HTTP/HTTPS 协议，向设备的 80 端口（HTTP）或者 832 端口（HTTPS）的 URL 地址/soap/netconf/，发送 SOAP 包装过的 NETCONF xml 请求来进行配置。例如，设备地址为 192.168.1.1，则请求地址为：

- HTTP 方式：http://192.168.1.1/soap/netconf/
- HTTPS 方式：https://192.168.1.1:832/soap/netconf/

提示

- HTTP/HTTP URL 地址中最后的反斜杠(/)不可省略。
- 80 为 HTTP 方式的缺省端口号，832 为 HTTPS 方式的缺省端口号。可以通过命令行修改端口号。

---

1. 基于 SOAP 的报文格式

如图 2-1 所示，基于 SOAP 的报文格式，本质是 HTTP 报文，其格式可分为 3 个层次：

- 外层：HTTP 报文头
- 中间层（HTTP 内容）：SOAP 格式的报文
- SOAP 内层（Body 部分）：NETCONF RPC 报文。

图2-1 基于 SOAP 的报文格式

2. 传递登录名和密码

如图 2-1 所示，登录时，用户名和密码分别放在元素\Envelope\Header\Authentication\UserName 和元素\Envelope\Header\Authentication\Password 中。

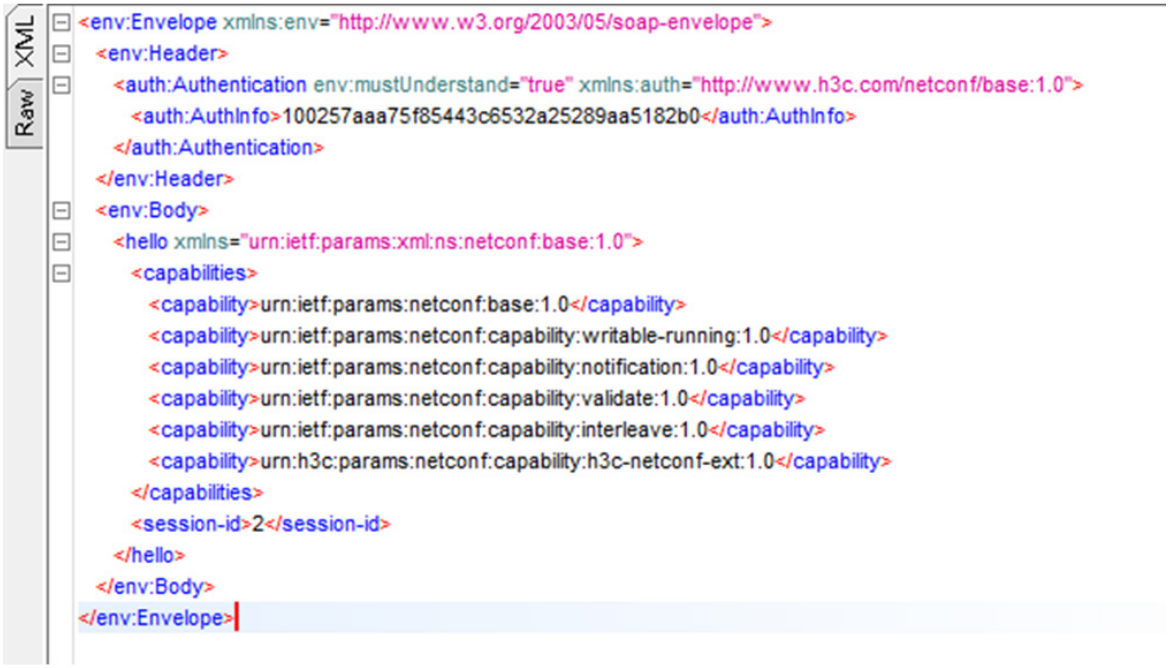
其中，元素 Authentication 必须满足原则：

- 属于命名空间 http://www.h3c.com/netconf/base:1.0。
- 具有 http://www.w3.org/2003/05/soap-envelope 命名空间下的 mustUnderstand 属性，且属性值必须为 1 或者 true。

3. 获取登录成功的凭据

如图 2-2 所示，登录成功后，返回的元素\Envelope\Header\Authentication\AuthInfo 的值为下次请求的凭据，客户端必须保留此凭据作为下次请求的凭证。

图2-2 获取登录成功的凭据



4. 登录成功后的请求使用凭据的方式

必须使用 hello 报文返回的凭据作为下次请求的依据，才能成功进行登录后的请求，凭据放在\Envelope\Header\Authentication\AuthInfo 中。

图2-3 登录成功后的请求使用凭据

5. 变更语言

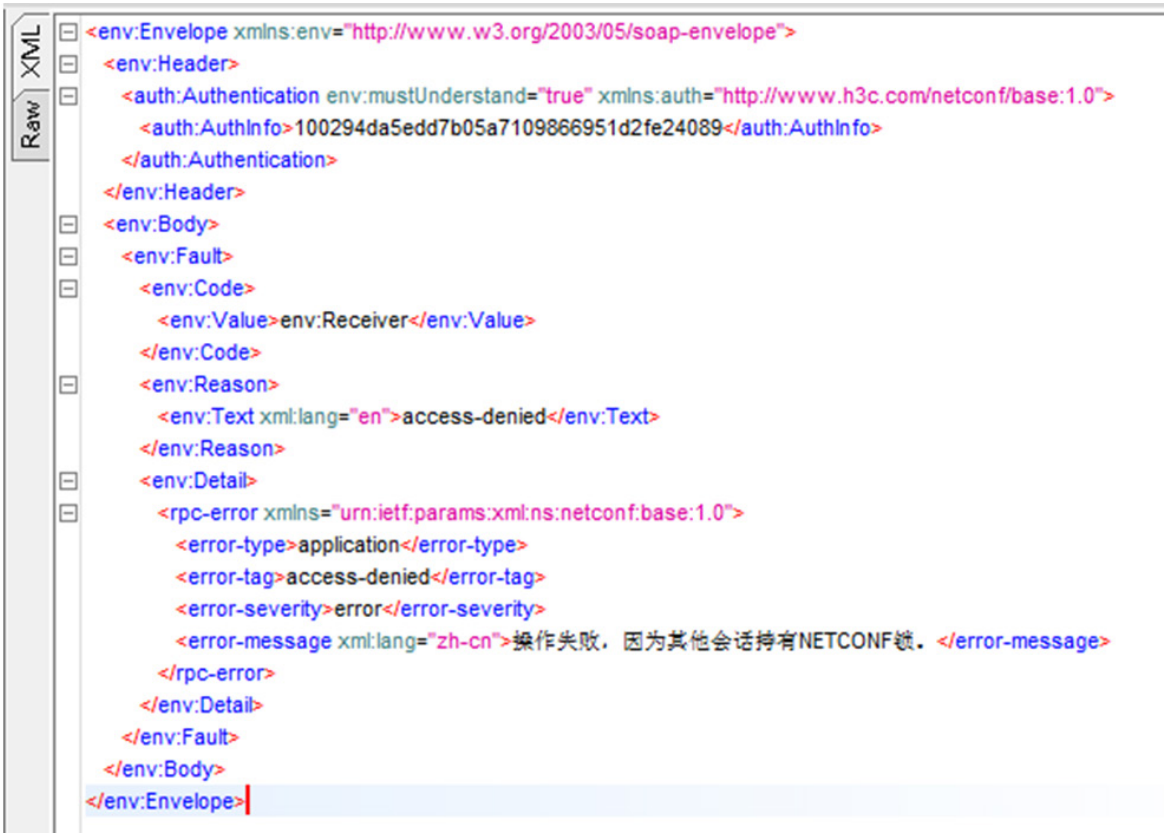
设备支持中文和英文，可以通过在请求中指定语言的方式来强迫设备在遇到错误时，返回指定语言的错误提示。如图 2-3 所示，语言放在元素\Envelope\Header\Authentication\Language 中，中文取值为 zh-cn，英文取值为 en，不提供 Language 时默认为英文。

提示

Language 必须放在元素 AuthInfo 的后面。

并不是所有的请求应答消息都支持中文。应答消息实际使用的语言，请以返回的属性 xml:lang=”xx”的指示为准。

图2-4 应答消息实际采用的语言



6. 基于 SOAP 的 NETCONF 请求和其他 NETCONF 请求的不同点

- 第一个 Hello 报文和认证是同时进行的。
- 连接为短连接，发送每个请求并接收到应答后，断开与服务器的连接。因此不支持事件订阅。
- 所有请求和正常应答都在外层封装一个 SOAP 层，只有 Body 层才封装 RPC 请求。
- 基于 SOAP 的 NETCONF 请求支持多线程并行发送请求，但不要在同一个会话中超过 4 个并发，一个会话超过 4 个并发的请求会被拒绝并返回资源不足的错误。

2.6.3 客户端使用 NETCONF over SSH 访问方式连接到设备

在设备上完成 NETCONF over SSH 连接相关配置后，客户端可通过 NETCONF over SSH 访问方式来配置和管理设备。访问过程遵循 RFC 4742。

一般情况下，客户端需要提供 IP 地址、用户名、密码、子系统、端口这几个信息。

Banner、欢迎、前置法律告警在下发到 NETCONF over SSH 之前可能会被传递到客户端，这些信息应当被客户端忽略。

2.6.4 客户端使用 Telnet/SSH/Console 以命令行方式连接到设备

客户端通过 Telnet、SSH 或 Console 方式登录设备的命令行终端后，执行 **xml** 命令进入 XML 视图，将合法的 NETCONF 报文直接拷贝、粘贴到命令行提示符处执行，即可实现对设备的配置和管理。

2.6.5 确认连接成功

使用 Telnet/SSH/Console/NETCONF over SSH 模式连接成功后，会收到服务器发送的 hello 报文，格式如下，收到后即可确认和设备连接是正常的：

```
<?xml version="1.0" encoding="UTF-8"?><hello
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"><capabilities><capability>urn:ietf:param
s:netconf:base:1.0</capability><capability>urn:ietf:params:netconf:capability:writable-r
unning:1.0</capability><capability>urn:ietf:params:netconf:capability:notification:1.0</
capability><capability>urn:ietf:params:netconf:capability:validate:1.0</capability><capa
bility>urn:ietf:params:netconf:capability:interleave:1.0</capability><capability>urn:h3c
:params:netconf:capability:h3c-netconf-ext:1.0</capability></capabilities><session-id>1<
/session-id></hello>
```

使用 SOAP 方式连接后，收到如下格式的 hello 报文，可确认 NETCONF 客户端和设备连接是正常的：

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Header>
    <auth:Authentication env:mustUnderstand="true"
xmlns:auth="http://www.h3c.com/netconf/base:1.0">
      <auth:AuthInfo>1001e17d1916e60ef0068f3c6387d4410a1b</auth:AuthInfo>
    </auth:Authentication>
  </env:Header>
  <env:Body>
    <hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <capabilities>
        <capability>urn:ietf:params:netconf:base:1.0</capability>
        <capability>urn:ietf:params:netconf:capability:writable-running:1.0</capability>
        <capability>urn:ietf:params:netconf:capability:notification:1.0</capability>
        <capability>urn:ietf:params:netconf:capability:validate:1.0</capability>
        <capability>urn:ietf:params:netconf:capability:interleave:1.0</capability>
        <capability>urn:h3c:params:netconf:capability:h3c-netconf-ext:1.0</capability>
      </capabilities>
      <session-id>1</session-id>
    </hello>
  </env:Body>
</env:Envelope>
```

2.7 测试NETCONF客户端和设备的连通性

2.7.1 交换能力集

建立 NETCONF 连接后，客户端和设备必须交换各自支持的能力集，双方收到对方的能力集后才可以进行下一步操作。

1. 设备发送给客户端的报文

设备会发送如下报文自动告知客户端支持的 NETCONF 能力集：

```
<?xml version="1.0" encoding="UTF-8"?><hello
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"><capabilities><capability>urn:ietf:param
s:netconf:base:1.0</capability><capability>urn:ietf:params:netconf:writable-running</cap
ability><capability>urn:ietf:params:netconf:capability:notification:1.0</capability><cap
ability>urn:ietf:params:netconf:capability:validate:1.0</capability><capability>urn:ietf
:params:netconf:capability:interleave:1.0</capability><capability>urn:h3c:params:netconf
:capability:h3c-netconf-ext:1.0</capability></capabilities><session-id>1</session-id></h
ello>
```

其中：

- <capabilities>和</capabilities>之间的内容表示设备支持的能力集，以设备实际返回值为准。
- <session-id>和</session-id>之间的内容表示为本次会话分配的会话 ID，用来唯一标识本次会话。

2. 客户端发送给设备的报文

客户端收到设备发送的能力集协商报文后，需要给设备发送如下格式的报文，告知设备客户端支持哪些 NETCONF 能力集。

Hello 协商报文格式如下：

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      capability-set
    </capability>
  </capabilities>
</hello>
```

其中，capability-set 表示客户端支持的能力集，由用户定义。一个<capability>和</capability>选项对中填写一个能力集，可以使用多个选项对，发送多个能力集。

3. 客户端依据设备能力调整 NETCONF 连接协议

对于 NETCONF over SSH 功能，按照协议规定，如果两者都有 Base 1.1 的能力，将自动使用 Base1.1 规定的访问方式进行通信，故客户端需要检查服务器返回的能力集，依据能力集进行通信协议调整。

2.7.2 断开 NETCONF 连接

1. 客户端发送报文

客户端通过下发以下报文断开 NETCONF 连接：

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <close-session/>
```

```
</rpc>
```

提示

对于处于 XML 视图的用户，要退回到用户视图，使用命令行来配置设备时，需要将以上报文拷贝、粘贴到客户端，才能退出 XML 视图。

2. 结果验证

设备收到会话关闭请求后，会返回如下报文并退回到用户视图：

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

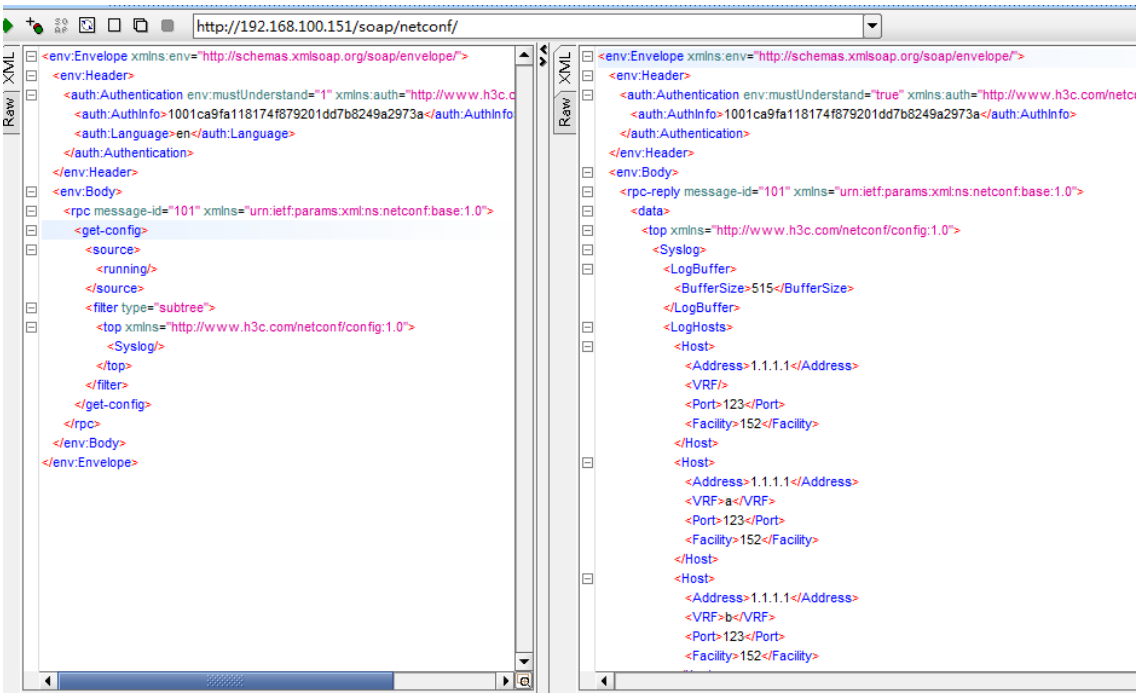
2.8 验证准备的NETCONF请求XML报文的正确性

连接到设备后，使用准备好的 XML 来查看请求是否正确。

推荐使用 SOAP UI 开源工具或者直接使用命令行 XML 方式来验证。

SOAP UI 执行的结果，如图 2-5 所示。

图2-5 SOAP UI 下发 XML 请求及返回结果举例



提示

使用命令行方式验证时，需要注意：

- 尽量避免在 Console 上使用，推荐在 Telnet 或者 SSH 下使用命令行的方式进行验证。由于串口上存在速度限制，且 XML 下无回显，在 Console 口上使用命令行方式验证很容易出现错误。
- 尽量避免手敲，尽可能拷贝和粘贴准备好的 XML 脚本。
- 与 SOAP 方式不同，命令行模式需要]]>]]>作为结束符。

2.9 将请求集成到客户端脚本或程序

当准备好的 XML 经过验证是正确的后，可以把验证好的 XML 集成到自己的客户端程序中。如果是无人值守的程序，需要考虑超时、断线等异常情况，以保证长时间情况下也能正确运行。

2.10 最佳实践建议

2.10.1 容错处理

客户端应答需要考虑在某些情况下的异常恢复，如连接断开的情况、XML 应答结果不是良好格式的情况、长时间不能得到结果的情况。

2.10.2 大数据量处理

系统中部分表的数据量可能非常大，不适合在一次请求中把数据全部取出，此时可以考虑使用 get-bulk 操作配合 count 属性。每次获取一定量的数据，取完需要下一批数据时，使用最后一条数据的索引为条件，继续使用 get-bulk，直到获取到所有的数据为止。

另外，在处理大数据量请求时，请考虑 TCP 底层超时的情况发生，需要适当放大底层 TCP timeout 参数的值，最好在客户端部署到生产环境前对最大数据情况进行模拟，以期找到一个合适的超时时间。

2.10.3 并发处理

系统支持多个用户或者会话同时下发 NETCONF 请求，但需要注意的是：

- 多个 edit-config 同时来变更系统时，如果不加锁，系统最终状态不一定是最终要求的状态。如果携带了错误后自动回滚的选项（edit-config 携带的 error-option 取值为 rollback-on-error，详细介绍请参见表 4-7），则多人情况下，可能会回退他人的修改。如果有可能，应当避免多个 NETCONF 客户端同时修改系统，或者对所有请求进行加锁后再下发。加锁方法的详细介绍，请参见“4.2.3 lock 操作”。
- 通过 lock 操作对当前 NETCONF 会话加锁后，用户仅可以通过该 NETCONF 会话配置设备。无法通过其他 NETCONF 会话、其他配置方式（如命令行或 SNMP）配置设备。
- 多个 get 系列操作或者 edit-config 请求并发时，不能保证一定得到线性的性能提升，视请求占用资源的情况，随着请求的数量增加，单个请求响应时间会加长，整体时间的缩短逐渐减小。

2.10.4 及时关闭连接

通过 SOAP 发送请求时，当不再使用会话时，建议使用 close-session 请求关闭当前会话。但需要注意的是：通过 close-session 关闭会话后，客户端有可能无法收到应答。

2.10.5 事件通知

当 debug 开关打开时，设备可能会生成大量的 debug 日志，导致正常响应滞后。因此，不建议通过 NETCONF 接收 Debug 级别的日志。

2.10.6 HTTP 和 HTTPS 混用

SOAP 同时支持 HTTP 和 HTTPS 用户，但不能串用：不能用 HTTP 登录后，使用 HTTP 的 AuthInfo 通过 HTTPS 访问；反之亦然。

2.11 常见问题及处理方法

2.11.1 命令行上 XML 请求没有反应

可能的原因是输入没有完成，可以通过手工敲入]]>]]>查看是否真的没有反应。一般情况下，输入]]>]]>后能看到一个应答，提示 XML 输入错误。为了解决该问题，需要在确认输入 XML 无误的情况下，使用拷贝粘贴的方式进行，不要手工敲击 XML 命令片段。

提示

- 由于 Console 口的速率限制问题，在 Console 口上即使拷贝粘贴，也可能出现丢失字符导致无法正确解析和应答的情况。除非对 Comware 系统非常熟悉，否则，不要在 Console 上直接敲 XML 命令。
- 如果已经在 Console 敲了 XML 命令，可以通过输入]]>]]>之后准备一个 close-session 消息，拷贝粘贴后等待退出。close-session 消息的详细介绍，请参见 “2.7.2 断开 NETCONF 连接”。

2.11.2 SOAP UI 等待客户端请求超时

因为系统中某些请求可能耗时很长，对于 SOAP 方式的请求，客户端软件可能设置了服务器的应答超时时间。当响应时间超过超时时间时，客户端软件直接返回失败。

为避免上述情况发生，建议：

- 逐渐调整客户端 socket 的超时时间，使之处于一个合适的数值。
- 通过过滤功能对需要获取的数据进行过滤，尽量避免获取整机数据。过滤功能的详细介绍，请参见 “5 Comware NETCONF 数据过滤功能”。

2.12 XML测试阶段使用的工具

设备上可使用表 2-3和表 2-4中的命令辅助调试 NETCONF 功能。

表2-3 SOAP 相关

命令	目的
display tcp	查看设备是否监听指定端口，是否存在对应服务的连接

表2-4 NETCONF over SSH 相关

命令	目的
display tcp	查看设备是否监听指定端口，是否存在对应服务的连接
display ssh server session	查看是否存在活跃的NETCONF over SSH会话
netconf log	查看NETCONF请求的日志和处理结果

3 Comware NETCONF 会话介绍

3.1 会话生命周期

3.1.1 基于 SOAP 的会话生命周期

对于基于 SOAP 的会话，从第一个 hello 报文建立连接开始建立会话，在以下情况下关闭会话：

- 发送 close-session 主动断开会话。
- 被其他用户通过 kill-session 关闭会话。
- 会话闲置时间超期。

会话期间，属于此会话的每一个请求都是一个新的 TCP 连接，应答完成后，TCP 连接关闭。后续请求依赖第一个 hello 报文返回的 AuthInfo 来标识其身份。如果最后一个请求之后会话闲置时间超过 NETCONF 会话超时时间（可通过 **`netconf idle-timeout`** 命令配置，缺省为 10 分钟），则此会话自动被关闭。如果希望保持会话，可以每隔一段时间，发送一个使用 AuthInfo 来标识的 hello 报文给服务器，服务器返回和第一个 hello 相同的应答报文。

3.1.2 基于其他访问方式的会话生命周期

对于基于 NETCONF over Console/NETCONF over Telnet/NETCONF over SSH 的会话，从连接建立开始建立会话，在以下情况下关闭会话：

- 发送 close-session 主动断开会话。
- 被其他用户通过 kill-session 关闭会话。
- 会话的底层连接断开。
- 会话闲置时间超期。缺省情况下，会话从建立到关闭期间，一直有效，且不会过期。通过 **`netconf idle-timeout`** 命令配置 NETCONF 会话超时时间后，如果会话闲置时间超过 NETCONF 会话超时时间，则该会话会被关闭。

3.2 请求并发

基于 SOAP 的会话可以通过多个线程，使用相同的 AuthInfo 来发送请求而互不干扰，但并发数量存在限制。不建议使用这个方式来进行并发。

基于其他方式连接的请求不支持同一个会话进行并发。

3.3 会话使用的RBAC权限

Comware NETCONF 中，权限分为 3 个部分，分别为协议级权限控制、模块级权限控制和实例级权限控制。其中：

- 协议级权限指的是 rpc-operation 需要的权限。
- 模块级权限指的是用户是否对模块、表、行、组、列具有访问权限。
- 实例级权限指的是具体到某个 VPN、Interface、VLAN、Secure Zone 的访问权限。

一个完整的请求如果要通过 RBAC，必须确保 3 部分都有权限才能继续。

例如，下面 edit-config 例子中：

- 协议级权限：要求用户具有 rpc/edit-config/config/top 的写权限。
- 模块级权限：要求用户具有 Ifmgr/Interfaces/Interface 表的写权限。
- 实例级权限：要求用户具有接口索引为 3 的接口的写权限。

```
<rpc message-id ="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <default-operation> merge</default-operation>
    <test-option>set</test-option>
    <error-option>continue-on-error</error-option>
    <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
        <Ifmgr xc:operation="delete">
          <Interfaces>
            <Interface>
              <IfIndex>3</IfIndex>
            </Interface>
          </Interfaces>
        </Ifmgr>
      </top>
    </config>
  </edit-config>
</rpc>
```

为了使 NETCONF 用户具有上述权限，需要为 NETCONF 用户赋予如下权限：

```
#
role name netconf
 rule 1 permit write xml-element rpc/edit-config/config/top
 rule 2 permit write xml-element ifmgr/interfaces/interface
 interface policy deny
  permit interface GigabitEthernet2/0/1
#
```

3.4 会话中使用的锁

Comware NETCONF 的锁遵循 RFC 4741 中的描述，当一个 NETCONF 客户端持有锁时，其它的 NETCONF 客户端或非 NETCONF 客户端（命令行、SNMP 等）均不可以执行系统配置操作。

3.5 最大会话个数

NETCONF over SOAP（over HTTP 和 over HTTPs）、NETCONF over Telnet、NETCONF over Console 和 NETCONF over SSH 访问方式下，每一种访问方式支持的最大 NETCONF 会话个数均依赖于 AAA 会话最大个数的配置。

4 执行 Comware NETCONF 请求

4.1 Comware支持的NETCONF操作速览

Comware NETCONF 支持的操作如表 4-1 所示。

表4-1 Comware NETCONF 支持的操作速览表

操作系列	操作名称	说明
获取业务数据	get	获取设备数据
	get-bulk	批量获取设备数据
	get-config	获取设备配置数据
	get-bulk-config	批量获取设备配置数据
配置动作	edit-config	修改系统配置
非配置动作	action	修改、获取运行统计信息之类的动作
事件相关	get/filter/netconf	获取系统支持的事件流
	create-subscription	订阅事件
	cancel-subscription	取消订阅事件
会话管理	close-session	关闭当前会话
	kill-session	关闭非当前会话
	get-sessions	获取已经建立的会话
	lock	锁定配置数据库
	unlock	解锁配置数据库
执行非交互命令行命令	CLI	作为NETCONF的补充，执行NETCONF尚未提供的命令
配置管理	rollback	配置回滚
	save	配置保存
	load	配置加载
	save-point	提供配置回滚点功能
请求验证	validate	NETCONF请求的XML语法验证
YANG操作	get/filter/netconf-state/capabilities	获取设备能力集
	get/filter/netconf-state/datastores	获取设备中的数据库
	get/filter/netconf-state/schemas	获取设备中的YANG文件名称列表
	get/filter/netconf-state/sessions	获取设备中的会话信息
	get/filter/netconf-state/statistics	获取NETCONF的统计信息

操作系列	操作名称	说明
	get-schema	获取YANG文件的内容
预配置	config-provisioned	开启预配置功能

4.2 NETCONF协议规定的操作

NETCONF 支持用户对设备进行业务操作，包括对指定信息的获取和修改。基本标签有<get>、<get-bulk>、<get-config>、<get-bulk-config>和<edit-config>，分别用来获取所有数据、获取配置数据和编辑指定模块的数据。详细规则参见各个模块的 NETCONF XML API 手册。

4.2.1 get-config 操作

<get-config>和<get-bulk-config>用来获取系统中所有可配置的变量的值，配置的方式包括 CLI、MIB、Web 等。<get-config>和<get-bulk-config>操作报文中可以包含子标签<filter>，用来对要获取的信息进行过滤。

1. 客户端发送报文

<get-config>和<get-bulk-config>的通用报文格式如下：

```
<?xml version="1.0"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>
    <source>
      <datastore/>
    </source>
    <filter>
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
          指定模块，子模块，表名，列名
      </top>
    </filter>
  </get-config>
</rpc>
```

其中，*datastore* 可以为 running：

参数	说明
running	操作的源数据库为设备的当前运行配置数据库

2. 结果验证

设备收到配置获取请求报文后会将相应配置通过如下报文反馈给客户端：

```
<?xml version="1.0"?>
<rpc-reply message-id="100"   xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    所有指定 filter 内的数据
  </data>
</rpc-reply>
```

4.2.2 edit-config 操作

1. edit-config 行为描述

默认情况下，当 NETCONF 正在执行 rollback 操作时，不允许下发 edit-config 请求。如需关闭这个限制，请使用 action 操作，详细介绍请参见“4.3.3 action 操作”。

<edit-config>支持如下选项：merge、create、replace、remove、delete、默认操作选项、默认错误处理选项、测试处理，关于这些选项的详细描述请参见表 4-7。

说明

对于有子结构的数据表，子结构下发后，系统会把所有子结构看作一个列进行处理。

2. 客户端发送报文格式

```
<?xml version="1.0"?>
<rpc message-id="100"  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <datastore/>
    </target>
    <error-option>
      失败时默认操作
    </error-option>
    <config>
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
        指定模块名，子模块名，列名，表名
      </top>
    </config>
  </edit-config>
</rpc>
```

其中，*datastore* 可以为 running：

参数	说明
running	操作的源数据库为设备的当前运行配置数据库

3. 结果验证

设备收到 edit-config 请求后会回应客户端，当客户端收到如下报文时，表示设置成功：

```
<?xml version="1.0">
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

用户还可以通过<get>操作可以查看参数的当前值是否和 edit-config 操作设置的值一致。

4. 举例——获取所有模块所有配置数据

(1) 组网需求

获取所有模块所有配置数据。

(2) 配置步骤

进入 XML 视图。

```
<Sysname> xml
```

进行能力交换。请将以下报文拷贝、粘贴到客户端。

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
</hello>
```

获取所有模块所有配置数据。请将以下报文拷贝、粘贴到客户端。

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>
    <source>
      <running/>
    </source>
  </get-config>
</rpc>
```

(3) 结果验证

如果客户端收到类似如下的报文，则表示操作成功。

```
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:web="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <data>
    <top xmlns="http://www.h3c.com/netconf/config:1.0">
      <Ifmgr>
        <Interfaces>
          <Interface>
            <IfIndex>1307</IfIndex>
            <Shutdown>1</Shutdown>
          </Interface>
          <Interface>
            <IfIndex>1308</IfIndex>
            <Shutdown>1</Shutdown>
          </Interface>
          <Interface>
            <IfIndex>1309</IfIndex>
            <Shutdown>1</Shutdown>
          </Interface>
          <Interface>
            <IfIndex>1311</IfIndex>
            <VlanType>2</VlanType>
          </Interface>
          <Interface>
            <IfIndex>1313</IfIndex>
            <VlanType>2</VlanType>
          </Interface>
```

```
        </Interfaces>
      </Ifmgr>
      <Syslog>
        <LogBuffer>
          <BufferSize>120</BufferSize>
        </LogBuffer>
      </Syslog>
      <System>
        <Device>
          <SysName>H3C</SysName>
          <TimeZone>
            <Zone>+11:44</Zone>
            <ZoneName>beijing</ZoneName>
          </TimeZone>
        </Device>
      </System>
      <Fundamentals>
        <WebUI>
          <SessionAgingTime>98</SessionAgingTime>
        </WebUI>
      </Fundamentals>
    </top>
  </data>
</rpc-reply>
```

5. 举例——获取 Syslog 模块的所有配置数据

(1) 组网需求

获取 Syslog 模块的所有配置数据。

(2) 配置步骤

进入 XML 视图。

```
<Sysname> xml
```

进行能力交换。请将以下报文拷贝、粘贴到客户端。

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
</hello>
```

获取 Syslog 模块的所有配置数据。请将以下报文拷贝、粘贴到客户端。

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>
    <source>
      <running/>
    </source>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
```

```
        <Syslog/>
      </top>
    </filter>
  </get-config>
</rpc>
```

(3) 结果验证

如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <data>
    <top xmlns="http://www.h3c.com/netconf/config:1.0">
      <Syslog>
        <LogBuffer>
          <BufferSize>120</BufferSize>
        </LogBuffer>
      </Syslog>
    </top>
  </data>
</rpc-reply>
```

6. 举例——取接口表的一条数据

(1) 组网需求

取接口表的一条数据。

(2) 配置步骤

进入 XML 视图。

```
<Sysname> xml
```

进行能力交换。请将以下报文拷贝、粘贴到客户端。

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
  </capabilities>
</hello>
```

取接口表的一条数据。请将以下报文拷贝、粘贴到客户端。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:web="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-bulk>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0"
xmlns:web="http://www.h3c.com/netconf/base:1.0">
        <Ifmgr>
          <Interfaces web:count="1">
          </Interfaces>
        </Ifmgr>
      </top>
    </filter>
  </get-bulk>
</rpc>
```

(3) 结果验证

如果客户端收到类似如下的报文，则表示操作成功。

```
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:web="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <data>
    <top xmlns="http://www.h3c.com/netconf/data:1.0">
      <Ifmgr>
        <Interfaces>
          <Interface>
            <IfIndex>3</IfIndex>
            <Name>GigabitEthernet1/0/2</Name>
            <AbbreviatedName>GE1/0/2</AbbreviatedName>
            <PortIndex>3</PortIndex>
            <ifTypeExt>22</ifTypeExt>
            <ifType>6</ifType>
            <Description>GigabitEthernet 1/0/2 Interface</Description>
            <AdminStatus>2</AdminStatus>
            <OperStatus>2</OperStatus>
            <ConfigSpeed>0</ConfigSpeed>
            <ActualSpeed>100000</ActualSpeed>
            <ConfigDuplex>3</ConfigDuplex>
            <ActualDuplex>1</ActualDuplex>
          </Interface>
        </Interfaces>
      </Ifmgr>
    </top>
  </data>
</rpc-reply>
```

7. 举例——修改参数值

(1) 组网需求

将 Syslog 模块中的日志缓冲区可存储的信息条数修改为 512。

(2) 配置步骤

进入 XML 视图。

```
<Sysname> xml
```

进行能力交换。请将以下报文拷贝、粘贴到客户端。

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
  </capabilities>
</hello>
```

把 Syslog 模块 LogBuffer 表中的 BufferSize 列改成 512。请将以下报文拷贝、粘贴到客户端。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:web="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
```

```
      <running/>
    </target>
    <config>
      <top xmlns="http://www.h3c.com/netconf/config:1.0" web:operation="merge">
        <Syslog>
          <LogBuffer>
            <BufferSize>512</BufferSize>
          </LogBuffer>
        </Syslog>
      </top>
    </config>
  </edit-config>
</rpc>
```

(3) 结果验证

如果客户端收到类似如下的报文，则表示操作成功.

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

4.2.3 lock 操作

因为设备同时最多能建立 32 个 NETCONF 连接，即最多支持 32 个用户同时使用 NETCONF 功能管理和监控设备。所以，当用户管理、维护设备或者定位网络问题时，为防止其他 NETCONF 用户修改当前配置、引入干扰，可以使用本特性为当前配置加锁。为当前配置加锁后，只有持有锁的用户可以修改设备的当前配置，其他 NETCONF 用户、CLI、WEB 等只能读取，不能修改当前配置。只有持有锁的用户可以解锁，解锁后其他用户才可以修改设备的当前配置或另外加锁。如果持有锁的用户的当前连接断开，系统会自动解锁。

1. 客户端发送报文

目前设备只支持对当前配置加锁，不能对具体的功能模块进行加锁。请将以下报文拷贝、粘贴到客户端，用户即能完成加锁操作。

```
<?xml version="1.0" encoding="UTF-8"?>
  <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <lock>
      <target>
        <datastore/>
      </target>
    </lock>
  </rpc>
```

其中，*datastore* 可以为 running：

参数	说明
running	操作的源数据库为设备的当前运行配置数据库

2. 结果验证

设备收到加锁报文后会回应客户端，当客户端收到如下报文时，表示加锁成功。

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

3. 当前配置加锁举例

(1) 组网需求

为设备加锁，以免其它用户修改设备的当前配置。

(2) 配置步骤

进入 XML 视图。

```
<Sysname> xml
```

进行能力交换。请将以下报文拷贝、粘贴到客户端。

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
</hello>
```

对当前配置加锁。请将以下报文拷贝、粘贴到客户端。

```
<?xml version="1.0" encoding="UTF-8"?>
  <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <lock>
      <target>
        <running/>
      </target>
    </lock>
  </rpc>
```

(3) 结果验证

如果客户端收到如下报文，则表示加锁成功。

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

用户加锁成功后，另一客户端发送加锁报文，设备会返回如下报文。

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<rpc-error>
  <error-type>protocol</error-type>
  <error-tag>lock-denied</error-tag>
  <error-severity>error</error-severity>
  <error-message xml:lang="en">Lock failed, lock is already held.</error-message>
  <error-info>
    <session-id>1</session-id>
```

```
      </error-info>
      </rpc-error>
    </rpc-reply>
```

以上报文表明：加锁失败，session-id 是 1 的用户已经持有锁。

4.2.4 unlock 操作

提示

只有锁的持有用户才能解锁，其它用户不能解锁。

1. 客户端发送报文

请将以下报文拷贝、粘贴到客户端，用户即能完成解锁操作。

```
<?xml version="1.0" encoding="UTF-8"?>
  <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <unlock>
      <target>
        <datastore/>
      </target>
    </unlock>
  </rpc>
```

其中，*datastore* 可以为 running：

参数	说明
running	操作的源数据库为设备的当前运行配置数据库

2. 结果验证

设备收到解锁报文后会回应客户端，当客户端收到如下报文时，表示解锁成功。

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

4.2.5 get 操作

<get>操作用来获取数据，包括运行状态数据和配置数据。

<get>操作会返回所有符合条件的数据，在某些情况下，会导致获取数据效率不高。此时可以考虑使用 Comware 扩展的<get-bulk>操作。<get-bulk>允许用户从固定数据项开始，向后获取指定条目的数据记录，具体可参见“4.3.1 get-bulk 操作”。

1. 客户端发送报文

<get>报文的通用格式如下：

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
```

```
    <filter>
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        指定模块，子模块，表名，列名
      </top>
    </filter>
  </get>
</rpc>
```

<filter>选项用于过滤信息。过滤条件和过滤结果，请参见“5 Comware NETCONF 数据过滤功能”。<filter>中可包括模块名、子模块名、表名和列名。

<get>操作报文中还可以携带 count 参数，如下为一个携带了 count 参数的<get>操作报文示例：

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc message-id ="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:xc="http://www.h3c.com/netconf/base:1.0">
  <get >
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0"
xmlns:base="http://www.h3c.com/netconf/base:1.0">
        <Syslog>
          <Logs xc:count="5">
            <Log>
              <Index></Index>
            </Log>
          </Logs>
        </Syslog>
      </top>
    </filter>
  </get >
</rpc>
```

其中，<get>操作报文中的 count 属性遵循如下约定：

- count 属性的位置可以从 top 下的节点（<Syslog>）开始，到表节点（<Logs>）为止，这几个位置都能放置 count 属性。如果在其他位置放置 count 属性，则会报告错误。
- count 放在模块节点上时，如果报文中指定的子孙节点（表）中没有 count 属性，则这些节点的 count 属性与模块节点的 count 属性一致。
- 如果 count 放在多级表节点上，只有父节点上的 count 生效；如果 count 放在子表节点上，则必须指定其祖先节点的所有索引列。
- 如果不指定 count，则获取从指定索引开始的所有数据。

2. 结果验证

设备收到配置获取请求报文后，会将相应参数的值通过如下报文反馈给客户端。

```
<?xml version="1.0"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    全部配置数据和状态数据
  </data>
</rpc-reply>
```

4.2.6 close-session 操作

close-session 操作用来关闭当前会话。

1. 客户端发送报文

请将以下报文拷贝、粘贴到客户端，用户即能关闭当前的会话。

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <close-session/>
</rpc>
```

2. 结果验证

设备收到会话关闭请求后会回应客户端，当客户端收到如下报文时，表示当前会话已经被关闭。

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

4.2.7 kill-session 操作

kill-session 操作用来关闭除自己外的另一个 NETCONF 会话，被关闭会话的用户退回到用户视图。

1. 客户端发送报文

请将以下报文拷贝、粘贴到客户端，用户即能关闭指定的会话。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <kill-session>
    <session-id>
      指定 session-ID
    </session-id>
  </kill-session>
</rpc>
```

2. 结果验证

设备收到会话关闭请求后会回应客户端，当客户端收到如下报文时，表示指定会话已经被关闭。

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

3. Kill-session 举例

(1) 配置需求

当前有两个 NETCONF 登录用户，session ID 分别是 1 和 2。session ID 为 1 的用户要关闭另一个用户的会话。

(2) 配置步骤

进入 XML 视图。

```
<Sysname> xml
```

进行能力交换。请将以下报文拷贝、粘贴到客户端。

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
```

```
      <capability>
        urn:ietf:params:netconf:base:1.0
      </capability>
    </capabilities>
  </hello>
```

关闭 session ID 为 2 的用户会话。请将以下报文拷贝、粘贴到客户端。

```
<rpc message-id="101"    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <kill-session>
    <session-id>2</session-id>
  </kill-session>
</rpc>
```

(3) 结果验证

如果客户端收到如下报文，则表明 session ID 为 2 的 NETCONF 会话已经被关闭。建立该会话的用户会从 XML 视图退回到用户视图。

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="101"  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

4.2.8 create-subscription 操作

NETCONF 支持 Syslog 事件、监控事件及模块上报事件三种类型的事件订阅。基于 NETCONF over Telnet/NETCONF over SSH 模式的连接支持事件订阅。

1. 订阅 Syslog 事件

用户向设备订阅事件后，设备上发生用户订阅的事件时，设备会自动向订阅的客户端发送事件的相关信息，信息内容包括事件的 code、group、severity 以及发生时间和描述信息。此处可订阅的事件是指系统支持的日志信息。具体可以订阅哪些模块的事件，请参见《日志信息参考手册》。

需要注意的是：

- 订阅只对当前连接生效。如果连接断开，订阅会自动取消。
- 通过在订阅报文中包括多个事件过滤选项，可以订阅多个事件。
- Comware 目前不支持事件回放。
- 在当前订阅存在的情况下，不支持继续订阅或者变更当前订阅。

事件订阅报文格式如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <create-subscription  xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
    <stream>NETCONF</stream>
    <filter>
      <event xmlns="http://www.h3c.com/netconf/event:1.0">
        <Code>code</Code>
        <Group>group</Group>
        <Severity>severity</Severity>
      </event>
    </filter>
    <startTime>start-time</startTime>
```

```
    <stopTime>stop-time</stopTime>
  </create-subscription>
</rpc>
```

其中：

- stream 表示支持的事件流，目前只支持 NETCONF。
- event 表示订阅的事件。
- code 表示日志信息中的助记符。
- group 表示日志信息中的模块名。
- severity 表示日志信息中的安全级别。
- start-time 表示订阅的开始时间。
- stop-time 表示结束订阅的时间。

设备收到订阅报文后会回应客户端，当客户端收到如下报文时，表示订阅成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply >
```

当订阅出错时设备会返回错误信息，例如：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <rpc-error>
    <error-type>error-type</error-type>
    <error-tag>error-tag</error-tag>
    <error-severity>error-severity</error-severity>
    <error-message xml:lang="en">error-message</error-message>
  </rpc-error>
</rpc-reply>
```

错误报文的详细定义请参见 RFC 4741。

2. 订阅监控事件

订阅监控事件后，NETCONF 每隔指定时间获取一次所订阅事件，并将符合订阅条件的信息发送给用户。

事件订阅报文格式如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <create-subscription xmlns='urn:ietf:params:xml:ns:netconf:notification:1.0'>
    <stream>NETCONF_MONITOR_EXTENSION</stream>
    <filter>
      <NetconfMonitor xmlns='http://www.h3c.com/netconf/monitor:1.0'>
        <XPath>XPath</XPath>
        <Interval>interval</Interval>
        <ColumnConditions>
          <ColumnCondition>
            <ColumnName>ColumnName</ColumnName>
            <ColumnValue>ColumnValue</ColumnValue>
            <ColumnCondition>ColumnCondition</ColumnCondition>
```

```
          </ColumnCondition>
        </ColumnConditions>
        <MustIncludeResultColumns>
          <ColumnName>columnName</ColumnName>
        </MustIncludeResultColumns>
      </NetconfMonitor>
    </filter>
    <startTime>start-time</startTime>
    <stopTime>stop-time</stopTime>
  </create-subscription>
</rpc>
```

表4-2 订阅监控事件属性说明

属性	说明
Stream	订阅的事件流，监控事件流的名称为NETCONF_MONITOR_EXTENSION
NetconfMonitor	监控事件的过滤信息
XPath	监控事件的路径，格式为：模块名[/子模块名]/表名
Interval	监控的时间间隔，取值范围为1～4294967，缺省值为300，单位为秒，即每隔300秒获取一次符合订阅条件的信息
ColumnName	监控列的名称，格式为：[组名称.]列名称
ColumnValue	监控列的过滤值
ColumnCondition	监控列的过滤条件： • more：大于 • less：小于 • notLess：不小于 • notMore：不大于 • equal：等于 • notEqual：不等于 • include：包含 • exclude：不包含 • startWith：开始于 • endWith：结束于 请根据监控列的过滤值的类型填写过滤条件
start-time	订阅的开始时间
stop-time	订阅的结束时间

说明

请求报文中必须按元素顺序组织请求报文。

设备收到订阅报文后会回应客户端，当客户端收到如下报文时，表示订阅成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

3. 订阅模块上报事件

订阅模块上报事件后，所订阅模块发生用户所订阅事件时，将主动上报 NETCONF，NETCONF 将事件消息发送给用户。

事件订阅报文格式如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:xs="http://www.h3c.com/netconf/base:1.0">
  <create-subscription xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
    <stream>XXX_STREAM</stream>
    <filter type="subtree">
      <event xmlns="http://www.h3c.com/netconf/event:1.0/xxx-features-list-name:1.0">
        <ColumnName xs:condition="Condition">value</ColumnName>
      </event>
    </filter>
    <startTime>start-time</startTime>
    <stopTime>stop-time</stopTime>
  </create-subscription>
</rpc>
```

表4-3 订阅模块上报事件属性说明

属性	说明
Stream	订阅的事件流，请根据设备实际支持情况填写模块上报事件流的名称
Event	订阅的事件的名称，一个事件流中包括多个事件，请根据事件流下的事件填写，命名空间为事件流的命名空间
ColumnName	当前事件下需要过滤的列的名称
Condition	事件列的过滤条件： • more：大于 • less：小于 • notLess：不小于 • notMore：不大于 • equal：等于 • notEqual：不等于 • include：包含 • exclude：不包含 • startWith：开始于 • endWith：结束于 请根据事件列的过滤值的类型填写过滤条件
Value	当前事件下需要过滤的列的值
start-time	订阅的开始时间

属性	说明
stop-time	订阅的结束时间

设备收到订阅报文后会回应客户端，当客户端收到如下报文时，表示订阅成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

4. 订阅事件举例

(1) 组网需求

客户端订阅没有时间限制的全部事件。订阅后在断开连接之前，设备发生的所有事件都会发送给客户端。

(2) 配置步骤

进入 XML 视图。

```
<Sysname> xml
```

进行能力交换。请将以下报文拷贝、粘贴到客户端。

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
</hello>
```

- 订阅 Syslog 事件举例

订阅全部 Syslog 事件，不限制订阅时间。请将以下报文拷贝、粘贴到客户端。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <create-subscription xmlns ="urn:ietf:params:xml:ns:netconf:notification:1.0">
    <stream>NETCONF</stream>
  </create-subscription>
</rpc>
```

如果客户端收到如下报文，则表示订阅成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <ok/>
</rpc-reply>
```

当设备上的风扇 1 有问题时，设备会发送如下报文来通知订阅客户端。

```
<?xml version="1.0" encoding="UTF-8"?>
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2011-01-04T12:30:46</eventTime>
  <event xmlns="http://www.h3c.com/netconf/event:1.0">
    <Group>DEV</Group>
    <Code>FAN_DIRECTION_NOT_PREFERRED</Code>
    <Slot>6</Slot>
    <Severity>Alert</Severity>
```

```
    <context>Fan 1 airflow direction is not preferred on slot 6, please check
it.</context>
  </event>
</notification>
```

当用户（IP 地址为 192.168.100.130）登录设备时，设备会发送如下报文来通知订阅客户端。

```
<?xml version="1.0" encoding="UTF-8"?>
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2011-01-04T12:30:52</eventTime>
  <event xmlns="http://www.h3c.com/netconf/event:1.0">
    <Group>SHELL</Group>
    <Code>SHELL_LOGIN</Code>
    <Slot>6</Slot>
    <Severity>Notification</Severity>
    <context>VTY logged in from 192.168.100.130.</context>
  </event>
</notification>
```

- 订阅监控事件举例

以 1s 为周期订阅 Device/Base 表。请将以下报文拷贝、粘贴到客户端。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<create-subscription xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <stream>NETCONF_MONITOR_EXTENSION</stream>
  <filter>
    <NetconfMonitor xmlns="http://www.h3c.com/netconf/monitor:1.0">
      <XPath>Device/Base</XPath>
      <Interval>1</Interval>
      <ColumnConditions>
        <ColumnCondition>
          <ColumnName>HostName</ColumnName>
          <ColumnValue>H3C</ColumnValue>
          <ColumnCondition>include</ColumnCondition>
        </ColumnCondition>
      </ColumnConditions>
    </NetconfMonitor>
  </filter>
</create-subscription>
</rpc>
```

如果客户端收到如下报文，则表示订阅成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <ok/>
</rpc-reply>
```

之后设备以 1s 周期间隔发送如下报文通知订阅客户端。

```
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2020-02-22T09:28:14</eventTime>
  <NetconfMonitor xmlns="http://www.h3c.com/netconf/monitor:1.0">
    <Device xmlns="http://www.h3c.com/netconf/data:1.0">
```

```
      <Base>
        <HostName>H3C</HostName>
      </Base>
    </Device>
  </NetconfMonitor>
</notification>
```

- 订阅模块上报事件举例

  # 订阅 lfmgr 模块的所有事件。请将以下报文拷贝、粘贴到客户端。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <create-subscription xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0"
xmlns:xs="http://www.h3c.com/netconf/base:1.0">
    <stream>Ifmgr</stream>
  </create-subscription>
</rpc>
```

  # 如果客户端收到如下报文，则表示订阅成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <ok/>
</rpc-reply>
```

  当接口 deactive 时，设备会发送如下报文通知订阅客户端。

```
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2020-02-22T09:32:10</eventTime>
  <InterfaceEvent xmlns="Ifmgr:1.0">
    <Interface>
      <Name>GigabitEthernet2/0/2</Name>
      <Status>IF_DEACTIVE</Status>
      <IfIndex>286</IfIndex>
      <AdminStatus>ADMIN_UP</AdminStatus>
      <OperStatus>OPER_DOWN</OperStatus>
      <Description>The Interface GigabitEthernet2/0/2 occurred IF_DEACTIVE
event,the administration status is ADMIN_UP,operation status is
OPER_DOWN.</Description>
    </Interface>
  </InterfaceEvent>
</notification>
```

  当接口 active 时，设备会发送如下报文通知订阅客户端：

```
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2020-02-22T09:32:10</eventTime>
  <InterfaceEvent xmlns="Ifmgr:1.0">
    <Interface>
      <Name>GigabitEthernet2/0/2</Name>
      <Status>IF_ACTIVE</Status>
      <IfIndex>286</IfIndex>
      <AdminStatus>ADMIN_UP</AdminStatus>
      <OperStatus>OPER_DOWN</OperStatus>
      <Description>The Interface GigabitEthernet2/0/2 occurred IF_ACTIVE event,the
administration status is ADMIN_UP,operation status is OPER_DOWN.</Description>
    </Interface>
```

```
    </InterfaceEvent>
  </notification>
```

4.2.9 validate 操作

validate 操作用来验证指定的 XML 请求格式和其他约束是否正确。Comware 系统中，只进行基本的格式检查，不下发到后台模块进行进一步的约束检查。故即使 validate 检查通过，不代表此请求一定能成功。因为其他操作也会进行格式检查，所以客户端可以直接使用其他操作进行下发，不必使用 validate 来保证格式的正确性。

4.2.10 get-schema 操作

<get-schema>操作用来获取设备上指定 NETCONF 模型文件内容，目前仅支持 YANG 文件的获取。

1. 客户端发送报文

<get-schema>报文的通用格式如下：

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-schema xmlns='urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring'>
    <identifier>syslog-data</identifier>
    <version>2015-05-07</version>
    <format>yang</format>
  </get-schema>
</rpc>
```

其中，Identifier、version、format 三列请参见“4.2.11 获取 netconf-state 操作”。

2. 结果验证

设备收到请求报文后，会将指定 YANG 文件的内容通过如下报文反馈给客户端。

```
<?xml version="1.0"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    指定 yang 文件内容数据
  </data>
</rpc-reply>
```

4.2.11 获取 netconf-state 操作

获取 netconf-state 可返回下面五种类型的数据：

- <netconf-state><capabilities/></netconf-state>用来获取设备支持的能力集列表。
- <netconf-state><datastores/></netconf-state>用来获取设备上的 NETCONF 配置库信息列表。
- <netconf-state><schemas/></netconf-state>用来获取设备支持的 schemas 文件信息列表。
- <netconf-state><sessions/></netconf-state>用来获取设备上的 NETCONF 会话信息列表。
- <netconf-state><statistics/></netconf-state>用来获取设备 NETCONF 全局统计信息。

获取 netconf-state 报文的通用格式如下：

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
```

```
    <filter type='subtree'>
      <netconf-state xmlns='urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring'>
        <getType/>
      </netconf-state>
    </filter>
  </get>
</rpc>
```

其中：

- *getType* 可以为 capabilities、datastores、schemas、sessions 或者 statistics。如果不指定 getType，则表示获取<netconf-state>下全部类型的 neconf-state 数据。
- 此操作目前不支持过滤。

1. 获取能力集列表举例

(1) 客户端发送报文

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter type='subtree'>
      <netconf-state xmlns='urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring'>
        <capabilities/>
      </netconf-state>
    </filter>
  </get>
</rpc>
```

(2) 结果验证

如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <netconf-state xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
      <capabilities>
        <capability>urn:ietf:params:netconf:base:1.0</capability>

<capability>urn:ietf:params:netconf:capability:writable-running:1.0</capability>

<capability>urn:ietf:params:netconf:capability:notification:1.0</capability>
     </capabilities>
    </netconf-state>
  <data>
</rpc-reply>
```

其中，能力集列表以设备实际返回值为准。

2. 获取配置库列表举例

(1) 客户端发送报文

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
```

```
    <filter type='subtree'>
      <netconf-state xmlns='urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring'>
      <datastores/>
      </netconf-state>
    </filter>
  </get>
</rpc>
```

(2) 结果验证

如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <netconf-state xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
      <datastores>
        <datastore>
          <name>running</name>
          <locks>
            <global-lock>
            <locked-by-session>1</locked-by-session>
            <locked-time>2015-07-30T12:54:45</locked-time>
            </global-lock>
          </locks>
       </datastore>
      </datastores>
    </netconf-state>
  <data>
</rpc-reply>
```

其中：

- 设备目前仅支持 running 库。
- locks 表示用户是否持有锁及锁信息。

3. 获取 schemas 文件列表举例

(1) 客户端发送报文

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter type='subtree'>
      <netconf-state xmlns='urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring'>
        <schemas/>
      </netconf-state>
    </filter>
  </get>
</rpc>
```

(2) 结果验证

如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
```

```
  <data>
    <netconf-state xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
<schemas>
       <schema>
         <identifier>syslog-data</identifier>
         <version>2015-05-07</version>
         <format>yang</format>
         <namespace>http://www.h3c.com/netconf/data:1.0</namespace>
         <location>NETCONF</location>
       </schema>
     </schemas>
    </netconf-state>
  <data>
</rpc-reply>
```

其中：

- identifier、version、format 作为<get-schema>操作的输入参数可获取相应文件的具体内容。
- format 列目前仅支持 YANG 格式。
- namespace 表示本文件的命名空间。
- location 表示文件的存放位置，目前仅支持 NETCONF，即存放于设备磁盘上。

4. 获取会话信息列表举例

(1) 客户端发送报文

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter type='subtree'>
      <netconf-state xmlns='urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring'>
        <sessions/>
      </netconf-state>
    </filter>
  </get>
</rpc>
```

(2) 结果验证

如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <netconf-state xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
      <sessions>
        <session>
          <session-id>1</session-id>
          <transport>netconf-soap-over-http</transport>
          <username>test</username>
          <source-host>192.168.100.68</source-host>
          <login-time>2015-07-30T09:44:03</login-time>
          <in-rpcs>0</in-rpcs>
```

```
          <in-bad-rpcs>0</in-bad-rpcs>
          <out-rpc-errors>0</out-rpc-errors>
          <out-notifications>0</out-notifications>
        </session>
      </sessions>
    </netconf-state>
  <data>
</rpc-reply>
```

其中：

- session-id 表示用户会话 ID 信息。
- transport 表示本会话使用的传输协议。
- username 表示用户登录使用的用户名。
- source-host 表示本会话请求的源地址。
- login-time 表示本会话登录时间。
- in-rpcs 表示本会话收到的合法 RPC 个数。
- in-bad-rpcs 表示本会话收到的不合法 RPC 个数。
- out-rpc-errors 表示本会话输出的 rpc-error 个数。
- out-notifications 表示本会话输出的通知事件个数。

5. 获取全局统计信息举例

(1) 客户端发送报文

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter type='subtree'>
      <netconf-state xmlns='urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring'>
        <statistics/>
      </netconf-state>
    </filter>
  </get>
</rpc>
```

(2) 结果验证

如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <netconf-state xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
      <statistics>
        <netconf-start-time>2015-07-30T09:44:01</netconf-start-time>
        <in-bad-hellos>0</in-bad-hellos>
        <in-sessions>2</in-sessions>
        <dropped-sessions>0</dropped-sessions>
        <in-rpcs>1</in-rpcs>
        <in-bad-rpcs>0</in-bad-rpcs>
        <out-rpc-errors>0</out-rpc-errors>
```

```
        <out-notifications>0</out-notifications>
      </statistics>
    </netconf-state>
  <data>
</rpc-reply>
```

其中：

- netconf-start-time 表示 NETCONF 管理系统启动时间。
- in-bad-hellos 表示本系统收到的不合法 hello 报文总数。
- in-sessions 表示本系统曾建立的会话个数。
- dropped-sessions 表示本系统曾因超时丢弃的会话个数。
- in-rpcs 表示本系统收到的合法 RPC 请求总个数。
- in-bad-rpcs 表示本系统收到的不合法 RPC 请求总个数。
- out-rpc-errors 表示本系统输出的 rpc-error 总个数。
- out-notifications 表示本系统输出的通知事件总个数。

4.3 H3C扩展的协议操作

4.3.1 get-bulk 操作

<get-bulk>操作用来从指定索引的下一条开始批量获取后续 N 条数据（索引行数据不返回），包括运行状态数据和配置数据。用户通过 index 属性指定索引，通过 count 属性指定 N。如未指定索引，则以第一条为索引；如未指定 N，或者数据表中符合条件的数据记录不足 N 条，则返回表中所有剩余的数据条目。

<get>操作会返回所有符合条件的数据，在某些情况下，会导致获取数据效率不高。<get-bulk>允许用户从固定数据项开始，向后获取指定条目的数据记录。

1. 客户端发送报文

和<get>的报文类似，<get-bulk>报文的通用格式如下：

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-bulk>
    <filter>
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        指定模块，子模块，表名，列名
      </top>
    </filter>
  </get-bulk>
</rpc>
```

<filter>选项用于过滤信息。过滤条件和过滤结果，请参见“5 Comware NETCONF 数据过滤功能”。<filter>中可包括模块名、子模块名、表名和列名。

<get-bulk>操作报文中还可以携带 count 和索引参数，如下为一个携带了 count 和索引参数的<get-bulk>操作报文示例：

```
<?xml version="1.0" encoding="UTF-8" ?>
```

```
<rpc message-id ="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:xc="http://www.h3c.com/netconf/base:1.0">
  <get-bulk>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0"
xmlns:base="http://www.h3c.com/netconf/base:1.0">
        <Syslog>
          <Logs xc:count="5">
            <Log>
              <Index>10</Index>
            </Log>
          </Logs>
        </Syslog>
      </top>
    </filter>
  </get-bulk>
</rpc>
```

其中，<get-bulk>操作报文中的 count 属性遵循如下约定：

- count 属性的位置可以从 top 下的节点（<Syslog>）开始，到表节点（<Logs>）为止，这几个位置都能放置 count 属性。如果在其他位置放置 count 属性，则会报告错误。
- count 放在模块节点上时，如果报文中指定的子孙节点（表）中没有 count 属性，则这些节点的 count 属性与模块节点的 count 属性一致。
- 如果 count 放在多级表节点上，只有父节点上的 count 生效；如果 count 放在子表节点上，则必须指定其祖先节点的所有索引列。
- 如果不指定 count，则获取从指定索引开始的所有数据。

2. 结果验证

设备收到配置获取请求报文后，会将相应参数的值通过如下报文反馈给客户端。

```
<?xml version="1.0"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    全部配置数据和状态数据
  </data>
</rpc-reply>
```

4.3.2 get-bulk-config 操作

<get-bulk-config>用来获取系统中所有可配置的变量的值，请求的标签格式和<get-config>一致。其差别在于：get-bulk-config 操作在指定完整索引后，返回从指定索引的下一条开始的数据（指定的索引行本身不返回）。

get-bulk-config 同样支持 count 操作属性，指定 count 后，每个表返回的数据最多不超过 count 行。

1. 客户端发送报文

<get-config>和<get-bulk-config>的通用报文格式如下：

```
<?xml version="1.0"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>
```

```
    <source>
      <datastore/>
    </source>
    <filter>
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
          指定模块，子模块，表名，列名
      </top>
    </filter>
  </get-config>
</rpc>
```

其中，*datastore* 可以为 running：

参数	说明
running	操作的源数据库为设备的当前运行配置数据库

2. 结果验证

设备收到配置获取请求报文后，会将相应配置通过如下报文反馈给客户端。

```
<?xml version="1.0"?>
<rpc-reply message-id="100"   xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    所有指定 filter 内的数据
  </data>
</rpc-reply>
```

4.3.3 action 操作

action 提供了一种非配置操作的手段，例如，执行 ping 操作、清除指定接口的统计数据等。

action 的返回结果包括成功、成功并返回结果、失败并返回提示信息 3 种。

1. 客户端发送报文

<action>的通用报文格式如下：

```
<?xml version="1.0"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <action>
    <top xmlns="http://www.h3c.com/netconf/action:1.0">
      指定模块，子模块，表名，列名
    </top>
  </action>
</rpc>
```

2. 结果验证

设备收到配置获取请求报文后，会将相应配置通过如下报文反馈给客户端。

成功不返回数据的格式：

```
<?xml version="1.0"?>
<rpc-reply message-id="100"   xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

成功并返回结果的格式：

```
<?xml version="1.0"?>
<rpc-reply message-id="100"   xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <action-result>
    Action 处理结果的数据
  </action-result>
</rpc-reply>
```

3. 举例——设置当前会话允许在 rollback 时下发 edit-config 请求

(1) 组网需求

设置当前会话允许在 rollback 时下发 edit-config 请求。

(2) 配置步骤。

进入 XML 视图

```
<Sysname> xml
```

进行能力交换。请将以下报文拷贝、粘贴到客户端。

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
  </capabilities>
</hello>
```

设置允许在 rollback 时下发 edit-config 请求。请将以下报文拷贝、粘贴到客户端。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <action>
    <error-option>stop-on-error</error-option>
    <top xmlns="http://www.h3c.com/netconf/action:1.0">
      <Fundamentals>
        <CurrentSessionOpt>
          <DisableEditConfigWhenRollback>false</DisableEditConfigWhenRollback>
        </CurrentSessionOpt>
      </Fundamentals>
    </top>
  </action>
</rpc>
```

(3) 结果验证

如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>。
```

如果需要开启会话内的限制，将上述请求报文中 false 改为 true 即可。

4.3.4 load 操作

<load>操作执行后，指定文件中的配置会被合并到设备的当前配置中。设备会将指定文件中的配置和当前配置进行比较：对于指定文件中有，但当前配置中没有的配置，直接运行；对于指定文件中和当前配置中不一致的配置，则用指定文件中的配置替换当前配置中的对应配置。

提示

加载文件相当于在设备上运行此段配置，对于需要先回滚才能成功的配置，用 load 操作不能成功。

1. 客户端发送报文

请将以下报文拷贝、粘贴到客户端，用户即可将指定文件中的配置追加到当前配置中。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <load>
    <file>指定文件的名称</file>
  </load>
</rpc>
```

其中，“指定文件的名称”必须以存储介质的名称开头，后缀为.cfg。如果不指定文件名，则缺省保存到主用下次启动配置文件中。

2. 结果验证

设备收到配置加载请求后会回应客户端，当客户端收到如下报文时，表示配置加载成功。

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

4.3.5 save 操作

save 操作用来将设备的当前配置保存到指定名称的文件中。

1. 客户端发送报文

请将以下报文拷贝、粘贴到客户端，用户即可将设备的当前配置保存到指定名称的文件中。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save OverWrite=”false” VsysName=”vsys-name”>
    <file>指定文件的名称</file>
  </save>
</rpc>
```

其中：

- “指定文件的名称”必须以存储介质的名称开头，后缀为.cfg。当报文中存在<file>列时，必须输入指定文件的名称，不能为空；如果不存在该列，则设备会自动将当前配置保存到缺省的主用下次启动配置文件中。
- OverWrite 属性的默认取值为 true。缺省情况下，当指定的配置文件名与原有配置文件名相同时，原文件将被覆盖，当前配置保存成功。如果指定 OverWrite 的值为 false，则当指定的配置文件名与原有配置文件名相同时，当前配置保存失败，同时返回错误提示信息。
- VsysName 属性仅在支持 vSystem 的系统有效。用户 vSystem 下不允许指定 VsysName，且不允许指定 save 的文件名称。管理 vSystem 下根据该属性指定的 vsys-name，将指定

vSystem 的配置保存到指定的文件中。指定 VsysName 属性的同时支持 OverWrite，不支持 Binary-only 属性。

2. 结果验证

设备收到配置保存请求后会回应客户端，当客户端收到如下报文时，表示保存成功。

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

3. 配置保存举例

(1) 组网需求

将设备的当前配置保存到配置文件 my_config.cfg。

(2) 配置步骤

进入 XML 视图。

```
<Sysname> xml
```

进行能力交换。请将以下报文拷贝、粘贴到客户端。

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
</hello>
```

将设备的当前配置保存到配置文件 my_config.cfg。请将以下报文拷贝、粘贴到客户端。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save>
    <file> my_config.cfg</file>
  </save>
</rpc>
```

(3) 结果验证

如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

4.3.6 rollback 操作

rollback 操作用来将设备的当前配置恢复到指定配置文件中的配置。

1. 客户端发送报文

请将以下报文拷贝、粘贴到客户端，用户即可将设备的当前配置恢复到指定配置文件中的配置。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <rollback>
```

```
    <file>指定文件的名称</file>
  </rollback>
</rpc>
```

2. 结果验证

设备收到配置回滚请求后会回应客户端，当客户端收到如下报文时，表示配置回滚成功。

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

4.3.7 CLI 操作

通过 NETCONF 功能，用户可以将命令行封装在 XML 报文中对设备进行操作。

CLI 包括 Open-channel、Close-channel、Execution 和 Configuration 四种子操作：

- Open-channel：打开当前会话的保留 channel。
- Close-channel：关闭当前会话的保留 channel。
- Execution：只在用户视图下进行操作，不能通过 **system** 命令切换进入系统视图。
- Configuration：默认在系统视图下进行操作，可以通过 **system** 和 **quit** 命令进行视图切换。使用 channel 方式下发 CLI 时，不可以通过 **quit** 命令退回到用户视图，且除首次下发命令是在系统视图，其余所在视图为上一条命令执行完成时所在的视图。

表4-4 CLI 标签的介绍

操作	说明	举例

<Open-channel>	打开保留channel，并将当前channel所对应的视图置为系统视图。如果创建保留channel时，保留channel已存在，则返回失败	打开保留channel，且当前视图为系统视图：
		`<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
		`  <CLI>`
		`  <Open-channel/>`
		`  </CLI>`
		`</rpc>`

<Close-channel>	关闭保留channel，如果关闭失败，则返回失败	关闭保留channel：
		`<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
		`  <CLI>`
		`  <Close-channel/>`
		`  </CLI>`
		`</rpc>`

<Configuration>	当前标签增加了exec-use-channel属性，用户下发命令时必须携带该属性，该属性取值包括：
	• false：不使用 channel 下发执行命令，同原有的下发方式
	• true：使用临时 channel 下发执行命令
	• persist：使用保留 channel 下发执行命令
		在系统视图/保留channel当前视图下执行**display this**命令：
		`<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
		`  <CLI>`
		`  <Configuration exec-use-channel=”persist”>`
		`  display this`
		`  </Configuration>`
		`  </CLI>`
		`</rpc>`

<Execution>	与原有实现保持一致，只能执行用户	在用户视图下执行**display this**命令：

操作	说明	举例
	视图下的命令	<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <CLI> <Execution> display this </Execution> </CLI> </rpc>

Configuration 支持 exec-use-channel 属性，取值包括 false、true 和 persist，具体的使用方法请如表 4-5 所示。

表4-5 Configuration 支持 exec-use-channel 属性的介绍

属性值	说明	举例
false	使用原有的exec comsh方式执行命令。由于需要启动comsh执行速度稍慢	在系统视图下执行**display this**命令： <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <CLI> <Configuration exec-use-channel=”false”> display this </Configuration> </CLI> </rpc>
true	使用一个临时channel下发执行命令。此时执行命令对比原有exec comsh的方式稍快。但对于命令行本身就会使用exec comsh方式的命令执行效率无明显提升	在系统视图下执行**display this**命令： <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <CLI> <Configuration exec-use-channel=”true”> display this </Configuration> </CLI> </rpc>
persist	使用保留channel下发执行命令 如果保留channel不存在，则先自行创建保留channel，并直接从系统视图开始执行命令 如果保留channel存在，则直接在保留channel当前所处的视图执行本次下发的命令	在系统视图下执行**display this**命令： <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <CLI> <Configuration exec-use-channel=”persist”> display this </Configuration> </CLI> </rpc>

2. 客户端发送报文

当需要给设备发送命令时，请使用格式如下的 NETCONF 报文：

- 不使用 channel 下发执行命令

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <CLI>
    <Execution>
      命令行
    </Execution>
  </CLI>
```

```
</rpc>
```

或

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <CLI>
    <Configuration exec-use-channel="false">
      命令行
    </ Configuration>
  </CLI>
</rpc>
```

- 使用临时 channel 方式下发执行命令

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <CLI>
    <Configuration exec-use-channel="true">
      命令行
    </ Configuration>
  </CLI>
</rpc>
```

- 使用保留 channel 方式下发执行命令

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <CLI>
    <Configuration exec-use-channel="persist">
      命令行
    </ Configuration>
  </CLI>
</rpc>
```

- 打开保留 channel 方式

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <CLI>
    <Open-channel/>
  </CLI>
</rpc>
```

- 关闭保留 channel

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <CLI>
    <Close-channel/>
  </CLI>
</rpc>
```

一对<Execution>或者<Configuration>子标签中可以包含多个命令行，一条命令输入完毕后，换行，再输入下一条命令即可。

3. 结果验证

提示

- 所有交互式命令，即需要等待用户输入的命令，都不能保证在 CLI 模式下可以正确执行。
- Channel 方式下发的命令行回显不能超过 4K 否则会导致 channel 挂死，且 channel 不能通过 **quit** 等命令直接退出。

设备收到命令行指令后会回应客户端，当客户端收到如下报文时，表示命令行执行成功（注意命令响应被 CDATA 节点包含）。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <CLI>
    <Execution>
      <![CDATA[对应命令行响应]]>
    </Execution>
  </CLI>
</rpc-reply>
```

对于 Open-channel、Close-channel 操作成功返回如下报文：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101235" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

对于 Open-channel、Close-channel、Configuration 的 channel 下发方式操作失败返回如下报文（XXX 为失败原因）：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <error-type>application</error-type>
  <error-tag>data-missing</error-tag>
  <error-severity>error</error-severity>
  <error-message xml:lang="en">失败错误提示</error-message>
</rpc-error>
```

4. 命令行操作举例

(1) 配置需求

向设备发送显示当前配置命令。

(2) 配置步骤

进入 XML 视图。

```
<Sysname> xml
```

进行能力交换。请将以下报文拷贝、粘贴到客户端。

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
```

```
  </capabilities>
</hello>
```

向设备发送显示当前配置命令。请将以下报文拷贝、粘贴到客户端。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <CLI>
    <Execution>
      display current-configuration
    </Execution>
  </CLI>
</rpc>
```

(3) 结果验证

如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <CLI>
    <Execution><![CDATA[
#
 version 7.1.052, Demo 2501005
#
 sysname ab
#
 ftp server enable
 ftp update fast
 ftp timeout 2000
#
 irf mac-address persistent timer
 irf auto-update enable
 undo irf link-delay
#
 domain default enable system
#
 telnet server enable
#
vlan 1
#
vlan 1000
#
radius scheme system
 primary authentication 127.0.0.1 1645
   ]]>
   </Execution>
  </CLI>
</rpc-reply>
```

4.3.8 get-sessions 操作

使用 get-sessions 操作，用户可以获取当前设备的所有 NETCONF 会话信息。

1. 客户端发送报文

请将以下报文拷贝、粘贴到客户端：

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-sessions/>
</rpc>
```

2. 结果验证

设备收到命令行指令后会回应客户端，当客户端收到如下报文时，表示命令行执行成功。

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-sessions>
    <Session>
      <SessionID>用户会话 ID 信息 </SessionID>
      <Line> line 信息</Line>
      <UserName>用户登陆名称</UserName>
      <Since>用户登录时间</Since>
      <LockHeld>用户是否持有锁</LockHeld>
    </Session>
  </get-sessions>
</rpc-reply>
```

其中：

- SessionID 表示用户会话 ID 信息，返回值为一个从 1 开始的数字。
- Line 表示用户登录的用户线信息，通过 SOAP 方式登录的用户的用户线为空。
- UserName 表示登录用户的用户名。
- Since 表示用户登录时间，返回值为一个标准的 Schema 的 DateTime 类型的时间。
- LockHeld 表示用户是否持有锁，返回值为 true、false。

3. 获取会话信息举例

(1) 配置需求

获取会话信息。

(2) 配置步骤

进入 XML 视图。

```
<Sysname> xml
```

进行能力交换。请将以下报文拷贝、粘贴到客户端。

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
</hello>
```

获取设备上当前存在的 NETCONF 会话的信息。请将以下报文拷贝、粘贴到客户端。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
```

```
  <get-sessions/>
</rpc>
```

(3) 结果验证

如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <get-sessions>
    <Session>
      <SessionID>1</SessionID>
      <Line>vty0</Line>
      <UserName></UserName>
      <Since>2011-01-05T00:24:57</Since>
      <LockHeld>false</LockHeld>
    </Session>
  </get-sessions>
</rpc-reply>
```

以上信息表明：目前有一个 NETCONF 连接，SessionID 是 1，登录用户类型 vty0，登录时间是 2011-01-05T00:24:57，此用户不持有锁。

4.3.9 save-point 操作

设备支持配置自动回滚，配置回滚点后，设备可以在如下场景下进行配置回滚：

- NETCONF 客户端主动下发配置回滚指令。
- NETCONF 客户端在指定的时间内无任何指令，即 NETCONF 会话空闲时间超过配置的回滚空闲超时时间。
- NETCONF 客户端和设备间的连接异常断开。

配置回滚步骤如下：

(1) 对当前系统加锁。

(2) 下发<save-point>/<begin>操作进行配置回滚点标记。

(3) 下发<edit-config>操作，对设备进行所需配置。

(4) 下发<save-point>/<commit>进行配置确认，可分多次修改配置并确认。

(5) 下发<save-point>/<rollback>进行配置回滚。可选择对应的一次 commit 进行配置回滚，或者等待 NETCONF 会话空闲时间超过配置回滚空闲超时时间时，自动将配置回滚到最近使用的 commit 的配置。

(6) 下发<save-point>/<end>结束配置回滚功能。

(7) 对当前系统解锁。

注意

如果存在多个 NETCONF 会话同时配置设备，建议配置回滚前先用<lock>锁定系统，否则可能导致设备运行配置与回滚前配置不一致。

2. 启动配置回滚功能，标识起始回滚点

(1) 客户端发送报文

请将以下报文拷贝、粘贴到客户端：

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <begin>
      <confirm-timeout>100</confirm-timeout>
    </begin>
  </save-point>
</rpc>
```

回滚空闲超时时间<confirm-timeout>为可选，取值范围为 1～65535，单位为秒，缺省为 600 秒。

(2) 结果验证

当客户端收到如下报文时，表示配置回滚点成功。

```
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <save-point>
      <commit>
        <commit-id>1</commit-id>
      </commit>
    </save-point>
  </data>
</rpc-reply>
```

3. 主动下发配置回滚

(1) 客户端发送报文

请将以下报文拷贝、粘贴到客户端：

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <rollback>
      <commit-id/>
      <commit-index/>
      <commit-label/>
    </rollback>
  </save-point>
</rpc>
```

<commit-id/>，<commit-index/>，<commit-label/>任选一个，或者不选时，回滚此前最近使用的 commit 的配置。其中：

- commit-id 为系统唯一标识的 commit 编号。
- commit-index 表示最近 50 次的 commit，0 表示最近一次下发 commit，49 表示最远一次 commit。
- commit-label 为标签，不同的 commit 标签不能一样，可以没有标签。

(2) 结果验证

当客户端收到如下报文时，表示主动下发配置回滚成功。

```
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok></ok>
</rpc-reply>
```

4. 主动下发配置确认

下发<save-point>/<commit>操作接受当前配置。使能 save-point 后，系统可支持最近 50 个 commit 回滚点的配置回滚。超过 50 次，commit 需指定 force 属性强制覆盖最早的记录。

(1) 客户端发送报文

请将以下报文拷贝、粘贴到客户端：

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <commit>
      <label>SUPPORT VLAN<label>
      <comment>vlan 1 to 100 and interfaces. Each vlan used for different custom as
follows: ……</comment>
    </commit>
  </save-point>
</rpc>
```

其中，<label/>和<comment/>可选。

(2) 结果验证

当客户端收到如下报文时，表示主动下发配置确认成功。

```
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <save-point>
      <commit>
        <commit-id>2</commit-id>
      </commit>
    </save-point>
  </data>
</rpc-reply>
```

5. 取消配置回滚

下发<save-point>/<end>操作可以取消配置回滚。

(1) 客户端发送报文

请将以下报文拷贝、粘贴到客户端：

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <end/>
  </save-point>
</rpc>
```

(2) 结果验证

当客户端收到如下报文时，返回对应的 end 结果成功。

```
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

6. 获取系统的 commit 回滚点的记录

下发<save-point>/<get-commits>操作可以获取 commit 的操作记录。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <get-commits>
      <commit-id/>
      <commit-index/>
      <commit-label/>
    </get-commits>
  </save-point>
</rpc>
```

其中，<commit-id>、<commit-index>、<commit-label>任选一个，也可以都不选。都不选时，表示显示所有的 commit 记录。

(1) 客户端发送报文

请将以下报文拷贝、粘贴到客户端。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <get-commits>
      <commit-label>SUPPORT VLAN</commit-label>
    </get-commits>
  </save-point>
</rpc>
```

(2) 结果验证

当客户端收到如下报文时，返回对应的 commit 操作记录。

```
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <save-point>
      <commit-information>
        <CommitID>2</CommitID>
        <TimeStamp>Thu Oct 30 11:30:28 1980</TimeStamp>
        <UserName>test</UserName>
        <Label>SUPPORT VLAN</Label>
      </commit-information>
      <commit-information>
        <CommitID>1</CommitID>
        <TimeStamp>Wed Nov  8 18:26:28 1972</TimeStamp>
        <UserName>test</UserName>
      </commit-information>
    </save-point>
  </data>
</rpc-reply>
```

7. 获取系统 commit 回滚点的配置

下发<save-point>/<get-commits>操作可以获取对应的 commit 操作时的配置。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
```

```
    <get-commit-information>
      <commit-information>
        <commit-id/>
        <commit-index/>
        <commit-label/>
      </commit-information>
      <compare-information>
        <commit-id/>
        <commit-index/>
        <commit-label/>
      </compare-information
    </get-commit-information>
  </save-point>
</rpc>
```

其中，<commit-id/>、<commit-index/>、<commit-label/>任选一个。<compare-information/>可选。<commit-information>不输入参数时，显示最近使用的 commit 配置。

(1) 客户端发送报文

请将以下报文拷贝、粘贴到客户端。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <save-point>
    <get-commit-information>
      <commit-label>SUPPORT VLAN</commit-label>
    </get-commit-information>
  </save-point>
</rpc>
```

(2) 结果验证

当客户端收到如下报文时，显示对应的配置信息。

```
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <save-point>
      <commit-information>
        <content>
          …
          interface vlan 1
          …
        </content>
      </commit-information>
    </save-point>
  </data>
</rpc-reply>
```

4.3.10 索引替换功能

1. 功能简介

Comware 提供索引替换功能，在某些表中，执行 get、get-config、edit-config、action 操作时可以使用名称来替换索引。比如，假定接口 Ethernet1/0/1 的索引是 1，则可以用 Ethernet0/0/0 来替换接口索引 1。具体哪些表具有上述能力，请参考对应的 NETCONF API 文档。

例如：假定接口 GigabitEthernet1/0/1 是 Trunk 接口，要允许 VLAN 2、3 通过，可以下发下面的 XML：

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
xmlns:base=”http://www.h3c.com/netconf/base:1.0”
        <VLAN>
          <TrunkInterfaces>
            <Interface>
              <IfIndex>GigabitEthernet1/0/1</IfIndex>
              <PermitVlanList>2-3</PermitVlanList>
            </Interface>
          </TrunkInterfaces>
        </VLAN>
      </top>
    </config>
  </edit-config>
</rpc>
```

为了区分数字是名称还是索引，从 h3c-name2index1.1 版本后（版本信息从能力集中获取，能力集的详细介绍请参见“2.7 测试 NETCONF 客户端和设备的连通性”），可以使用 valuetype 属性指定该值的类型。valuetype 取值为：

- Name：值为名称类型。
- index：值为索引类型。
- auto：设备先按名称类型匹配，如果没有匹配到任何信息，再按照索引类型匹配。如果不指定 valuetype 属性，缺省为 auto。

例如，假定 IfIndex 元素为 index 类型，值为 1，可以下发下面的 XML：

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <getoperation>
    <filter>
      <top xmlns="http://www.h3c.com/netconf/config:1.0"
xmlns:base=”http://www.h3c.com/netconf/base:1.0”>
        <VLAN>
          <TrunkInterfaces>
            <Interface>
              <IfIndex base:valuetype="index">1</IfIndex>
            </Interface>
```

```
          </TrunkInterfaces>
        </VLAN>
      </top>
    </filter >
  </getoperation>
</rpc>
```

2. 限制

- 对于 get-bulk、get-bulk-config 操作，当替换的索引是表的索引，并且当前位置起定位作用时，不能使用一对多转换的功能。
- 对于 match 过滤，不支持名称到索引的转换功能。
- 对于非索引列，edit-config 时不能进行一对多转换。
- 对于不同的类型，支持的转换语法不一定完全相同。比如，接口管理支持 GigabitEthernet1/0/1 to GigabitEthernet1/0/24 这样的表示大量名称的写法，但 VPN 只支持一次加入一个 VPN 名称。

4.3.11 edit-config 的增量下发功能

某些表的列本身是一个列表，比如 VLAN TrunkInterface，其 PermitVlanList 的表示形式形如 2,3,5-8,9,10-100。在这个情况下，可能需要在不修改原有取值的情况下，增量下发新的取值。比如，不知道原来有哪些 VLAN 被 permit 了，需要在原来的基础上增加一个 VLAN 105，此时就需要下发增量的操作。增量下发的操作方式为：在支持增量下发的节点上指定 incremental=”true”属性。

要进行增量下发，必须在要增量下发的列上显式指定增量下发属性。例如，当前 VLAN 100 的 permit List 为 10，增量下发 VLAN 2 和 3，使用如下 XML：

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
           xmlns:base=”http://www.h3c.com/netconf/base:1.0”
        <VLAN xc:operation="operXXX">
          <TrunkInterfaces>
            <Interface>
              <IfIndex>100</IfIndex>
              <PermitVlanList base:incremental=”true”>2-3</PermitVlanList>
            </Interface>
          </TrunkInterfaces>
        </VLAN>
      </top>
    </config>
  </edit-config>
</rpc>
```

对于不同操作类型，增量下发 VLAN 后的操作结果如表 4-6 所示。

表4-6 增量下发 VLAN 后对于不同操作类型的操作结果

操作(*OperXXX*)	有增量属性时的操作结果	说明
create	PermitVlanList变为2,3,10	如果要增加的部分（2-3）中的一个或者多个已经存在，则报告已经存在，否则返回成功
delete	操作失败，返回配置不存在，因为2不存在	delete有增量属性时，会下发增量删除的值；delete有增量属性、无增量属性时，只要指定的要删除的任意部分不存在，均会返回配置不存在
merge	PermitVlanList变为2,3,10	-
replace	不支持relace操作	-
remove	操作成功，因为2-3不存在，PermitVlanList不变	remove有增量属性时，会下发增量删除的值；没有增量属性时，会删除当前列的全部配置

4.3.12 cancel-subscription 操作

cancel-subscription 操作用来取消订阅。在用户向设备订阅事件后，用户可以使用此操作取消已订阅的事件。

1. 客户端发送报文

取消事件订阅的请求报文格式如下：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <cancel-subscription xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
    <stream>XXX_STREAM</stream>
  </cancel-subscription>
</rpc>
```

属性	说明
stream	订阅的事件流的名称

2. 结果验证

设备收到请求报文后会回应客户端，当客户端收到如下报文时，表示取消订阅成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="100">
  <ok/>
</rpc-reply>
```

如果要取消的已订阅事件流不存在，设备会返回如下错误信息。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <rpc-error>
    <error-type>protocol</error-type>
    <error-tag>operation-failed</error-tag>
    <error-severity>error</error-severity>
```

```
    <error-message xml:lang="en">The subscription stream to be canceled doesn't exist:
Stream name=XXX_STREAM.</error-message>
  </rpc-error>
</rpc-reply>
```

4.4 获取和设置数据操作详细列表

Comware 支持的所有用来获取和设置数据的操作如表 4-7 所示。其中，包括 get、get-config、get-bulk、get-bulk-config、edit-config、action 操作。

表4-7 Comware 支持的获取和设置数据操作

操作	说明	XML 格式样例

get	获取数据，包括运行状态数据和配置数据	获取Syslog模块的全部数据的XML请求如下：
		`<rpc message-id ="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:xc="http://www.h3c.com/netconf/base:1.0">`
		`  <get>`
		`  <filter type="subtree">`
		`  <top xmlns="http://www.h3c.com/netconf/data:1.0" >`
		`  <Syslog>`
		`  </Syslog>`
		`  </top>`
		`  </filter>`
		`  </get>`
		`</rpc>`

get-config	获取配置数据，和get不同，它只返回非缺省的配置数据。如果没有配置数据，则返回一个空的<data>	获取接口表内所有配置的XML请求如下：
		`<rpc message-id ="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:xc="http://www.h3c.com/netconf/base:1.0">`
		`  <get-config>`
		`  <source>`
		`  <running/>`
		`  </source>`
		`  <filter type="subtree">`
		`  <top xmlns="http://www.h3c.com/netconf/config:1.0">`
		`  <Ifmgr>`
		`  <Interfaces>`
		`  <Interface/>`
		`  </Interfaces>`
		`  </Ifmgr>`
		`  </top>`
		`  </filter>`
		`  </get-config>`
		`</rpc>`

get-bulk	从指定索引的下一条开始批量获取后续N条数据（索引行数据不返回），包括运行状态数据和配置数据。用户通过index属性指定索引，通过count属性指定N。如未指定索引，则以第一条为索引；如未指定N，或者数据表中符合条件的数据记录不	获取全部接口数据的XML请求如下：
		`<rpc message-id ="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
		`  <get-bulk>`
		`  <filter type="subtree">`
		`  <top`

操作	说明	XML 格式样例
	足N条，则返回表中所有剩余的数据条目 get操作会返回所有符合条件的数据，这在某些情况下会导致效率问题。get-bulk允许用户从固定数据项开始，向后获取指定条目的数据记录	xmlns="http://www.h3c.com/netconf/data:1.0"> <Ifmgr> <Interfaces xc:count=”5” xmlns:xc=” http://www.h3c.com/netconf/base:1.0”> <Interface/> </Interfaces> </Ifmgr> </top> </filter> </get-bulk> </rpc>
get-bulk-config	从指定索引的下一条批量获取配置数据。和get-config类似，只返回非默认配置；其它约束类似get-bulk	获取全部接口配置信息的XML请求如下： <rpc message-id ="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1 .0" > <get-bulk-config> <source> <running/> </source> <filter type="subtree"> <top xmlns="http://www.h3c.com/netconf/config:1.0 "> <Ifmgr> </Ifmgr> </top> </filter> </get-bulk-config> </rpc>
edit-config: merge	在当前运行配置的基础上直接运行指定配置 merge操作必须指定具体的操作对象（行）： • 如果指定的对象存在，则直接配置该对象 • 如果指定的对象不存在，但允许创建，则先创建再配置该对象 • 如果指定的对象不存在且不允许创建，则返回 merge 失败	将BufferSize设置为120的XML请求如下： <rpc message-id ="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1 .0" xmlns:xc="urn:ietf:params:xml:ns:netconf:bas e:1.0"> <edit-config> <target> <running/> </target> <config> <top xmlns="http://www.h3c.com/netconf/config:1.0 "> <Syslog xmlns="http://www.h3c.com/netconf/config:1.0 " xc:operation="merge"> <LogBuffer> <BufferSize>120</BufferSize> </LogBuffer> </Syslog> </top> </config> </edit-config> </rpc>
edit-config: create	创建指定对象。create操作必须指定配置对象。create操作的XML数据格式和merge类似，只是operation属性需要指定为“create” • 如果当前配置表支持创建对象，且当前对象不存在，则先创建配置对象，再创建指定的配置 • 如果配置对象下对应的配置项	同上，把merge修改为create即可

操作	说明	XML 格式样例
	已经存在，则返回 data-exist 错误	
edit-config: replace	• 如果指定的对象存在，则替换指定对象的配置为当前配置 • 如果指定的对象不存在，但允许创建，则先创建再配置该对象为当前配置 • 如果指定对象不存在且不允许创建，则不进行 replace 操作，返回 invalid-value 错误，提示用户配置对象不存在	同上，把merge修改为replace即可
edit-config: remove	删除指定配置 • 当指定的删除对象中只有表索引时，则删除此配置指定对象的所有配置，同时删除指定对象 • 当指定的删除对象中不仅仅有表索引还存在配置项时，则删除此对象下面的指定配置 • 如果系统中指定对象不存在，或者 XML 消息未指定对象，则直接返回成功	同上，把merge修改为remove即可
edit-config: delete	删除指定配置 • 当指定的删除对象中只有表索引时，则删除此配置指定对象的所有配置，同时删除指定对象 • 当指定的删除对象中不仅仅有表索引还存在配置项时，则删除此对象下面的指定配置 • 如果系统中指定对象不存在，则直接返回不存在的错误消息	同上，把merge修改为delete即可
edit-config *默认操作选项*	edit-config操作用于修改当前系统配置。NETCONF定制了五种修改配置的方式：merge、create、delete、remove和replace。当XML消息中未指定修改配置方式的时候，则使用默认操作做为当前指令的操作方式，不会修改默认操作的缺省值 默认操作的缺省值是merge，在XML消息中可以通过<default-operation>节点来设置默认操作，取值为： • merge：当配置方式和默认操作方式均未指定时，使用该方式 • replace：当配置方式未指定，默认操作指定为 replace 的时候，edit-config 操作默认	下发一个空的操作，该操作仅仅验证格式，并不真正下发给系统，XML请求如下： <rpc message-id ="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <edit-config> <target> <running/> </target> <default-operation>none</default-operation> <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0"> <top xmlns="http://www.h3c.com/netconf/config:1.0"> <Ifmgr> <Interfaces> <Interface> <IfIndex>262</IfIndex>

操作	说明	XML 格式样例
	为 replace 操作 • none：当配置方式未指定，默认操作指定为 none 的时候，edit-config 操作默认为 none 操作。none 操作主要用来检查，下发为 none 操作的配置仅仅做 Schema 校验，不进行真正的配置下发。语法检查通过，就返回 ok 成功，否则失败	<Description>222222</Description> </Interface> </Interfaces> </Ifmgr> </top> </config> </edit-config> </rpc>
edit-config *默认错误处理选项*	edit-config将指定的配置设置到系统上，完成配置设置的操作。在执行edit-config的过程中，如果遇到一个实例配置出错，默认情况下会直接返回错误，但是为了使应用更加灵活，edit-config提供了错误选项，通过错误选项取值的不同，在发生错误的时候进行不同的处理操作 <error-option>节点用于设置一个实例配置出错后，后续实例配置的处理方式，缺省值为stop-on-error，全部取值为： • stop-on-error：停止处理，返回错误。此选项为默认选项 • continue-on-error：继续处理，但是报告错误 • rollback-on-error：停止并回滚配置	下发两个接口的配置，当下发第一个接口的配置发生错误时，继续进行下一个接口配置的下发，XML请求如下： <rpc message-id ="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1 .0"> <edit-config> <target> <running/> </target> <error-option>continue-on-error</error-optio n> <config xmlns:xc="urn:ietf:params:xml:ns:netconf:bas e:1.0"> <top xmlns="http://www.h3c.com/netconf/config:1.0 "> <Ifmgr xc:operation="merge"> <Interfaces> <Interface> <IfIndex>262</IfIndex> <Description>222</Description> <ConfigSpeed>1000000</ConfigSpeed> <ConfigDuplex>1</ConfigDuplex> </Interface> <Interface> <IfIndex>263</IfIndex> <Description>333</Description> <ConfigSpeed>1000000</ConfigSpeed> <ConfigDuplex>1</ConfigDuplex> </Interface> </Interfaces> </Ifmgr> </top> </config> </edit-config> </rpc>
edit-config *测试处理选项*	在真正执行edit-config操作时，可指定一个测试选项，使用<test-option>节点来决定当前配置是否真正下发。该节点的缺省值为test-then-set，全部取值为： • test-then-set：如果没有错误则将配置设置到系统 • set：将配置设置到系统 • test-only：只测试，并不下发	下发一个接口的配置，仅测试，XML请求如下： <rpc message-id ="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1 .0"> <edit-config> <target> <running/> </target> <test-option>test-only</test-option> <config xmlns:xc="urn:ietf:params:xml:ns:netconf:bas e:1.0">

操作	说明	XML 格式样例
	配置到系统。语法检查通过，就返回 ok 成功，否则失败	<top xmlns="http://www.h3c.com/netconf/config:1.0 "> <Ifmgr xc:operation="merge"> <Interfaces> <Interface> <IfIndex>262</IfIndex> <Description>222</Description> <ConfigSpeed>1000000</ConfigSpeed> <ConfigDuplex>1</ConfigDuplex> </Interface> </Interfaces> </Ifmgr> </top> </config> </edit-config> </rpc>
action	下发非配置数据的设置动作，比如，reset操作	清除全部接口的统计信息，XML请求如下： <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1 .0"> <action> <top xmlns="http://www.h3c.com/netconf/action:1.0 "> <Ifmgr> <ClearAllIfStatistics> <Clear> </Clear> </ClearAllIfStatistics> </Ifmgr> </top> </action> </rpc>

4.5 数据获取请求返回值说明

获取数据系列操作包括 get、get-bulk、get-config 和 get-bulk-config。本节介绍这些 get 系列操作的返回值。

4.5.1 get 系列请求返回值包括表、行和列的情况汇总

<get>和<get-bulk>报文的通用格式如下：

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <getoperation>
    <filter>
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        指定模块，子模块，表名，列名
      </top>
    </filter>
  </getoperation>
</rpc>
```

<get-config>和<get-bulk-config>报文的通用格式如下：

```
<?xml version="1.0" encoding="UTF-8" ?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <getoperation>
    <source>
      <running/>
    </source>
    <filter>
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        指定模块，子模块，表名，列名
      </top>
    </filter>
  </getoperation>
</rpc>
```

其中，*getoperation* 可以为 get、get-bulk、get-config 或者 get-bulk-config。<filter>选项用于过滤信息，<filter>中可包括模块名、子模块名、表名和列名。指定不同的<filter>，返回值可以是表、行和列：

- 返回值为表：
  - 如果<filter>中不指定模块（子模块），则表示全部模块（子模块）。一旦指定模块（子模块），则返回数据只包含指定模块（子模块）。
  - 如果模块下不指定表，则表示全部表。一旦指定表，则返回数据只包含指定表。
- 返回值为行：
  - 如果表下不指定列名，则表示该表的全部行。
  - 如果<filter>中只指定索引列，则返回与该索引列匹配的行的全部列。如果同时指定了索引列之外的其他列，则返回与索引列匹配的行的索引列和指定的列。
- 返回值为列：
  - 如果<filter>中只指定索引列，则返回的数据包括全部的列。如果同时指定了索引列之外的其他列，则返回的数据仅包含索引列和指定的列。
  - 对于多于一个索引的情况，索引列缺失，则返回的数据仅仅包括全部的索引
  - 对于 get-bulk 操作，前面的连续索引的值作为定位依据，后续不连续索引的值作为过滤条件存在。比如，一个多实例表存在 A、B、C、D、E 5 个索引，如果提供 A、B、D、E 4 个索引，则 A、B 的值作为定位条件，取满足 A、B 条件的第一个 C、D、E 的数据，同时，D、E 的值作为过滤条件存在。
  - 对于 get 操作，索引不全时，所有的索引列都作为过滤条件存在。

4.5.2 get 系列返回空数据的情况说明

下列情况可能导致返回空数据：

- 配置模块未启动。
- 无访问对应对象的 RBAC 权限。
- 过滤条件过滤掉全部数据。
- 发生其他非格式约束类的错误。
- 对于 get-config 系列操作，指定表中只有默认配置。

- 在当前设备上，不支持指定的表。

4.5.3 RBAC 配置对 get 系列返回结果的影响

如果表没有授予 read 权限，get 返回结果不包括此表数据。

如果列没有授予 read 权限，那么请求中一定不包括此列。如果请求不明确包括此列，则能返回其他列数据；如果显式包括此列，则此表没有数据可以返回。

对于表中行的实例，如果索引代表的对象没有权限，则对于 get 和 get-config 操作，获取不到此数据；对于 get-bulk 和 get-bulk-config 操作，会跳过此行。

4.5.4 模块自定义过滤对 get 系列返回结果的影响

某些模块可能提供自定义的过滤，用来提供更复杂的查询条件，比如 Route 模块的 Ipv4Routes 表，其过滤条件出现在行（多实例表），或者表（单实例表）上。其表现形式如下：

```
  <Route>
    <Ipv4Routes>
      <RouteEntry h3c:filter=' ip 1.1.1.1'
xmlns:h3c='http://www.h3c.com/netconf/base:1.0' >
      </RouteEntry>
    </Ipv4Routes>
  </Route>
```

如果同时出现模块自定义的过滤条件，则此过滤条件会和其他 get 系列的过滤条件共同作用，即模块提供的过滤条件会使返回的过滤结果进一步缩小。

具体哪些模块会提供自定义的过滤条件，需要查看对应的模块的 XML API 编程手册。

5 Comware NETCONF 数据过滤功能

5.1 数据过滤方式

当用户执行<get>、<get-bulk>、<get-config>或者<get-bulk-config>操作时，在 XML 语言中增加过滤条件，可以使用户只看到自己关心的数据。数据过滤包括严格匹配、正则表达式匹配和条件匹配三种。

同时指定多种过滤条件时，只有一种过滤条件可以生效。过滤条件按照优先级从高到底的顺序，依次为：严格匹配、正则表达式匹配和条件匹配。

5.2 严格匹配

严格匹配包括两种匹配方式：元素值方式和属性名方式。当两者都指定时，优先使用元素值方式。

用户在 XML 语言中直接指定对应的元素值，设备将对这些值进行严格匹配。如果指定了多个元素的值，则返回同时符合这几个条件的数据。只有完全匹配条件才返回成功。

例如，如下为一个 NETCONF 报文示例，用于获取所有状态为 UP 的接口的配置信息。

```
<rpc message-id ="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        <Ifmgr>
          <Interfaces>
            <Interface>
              <AdminStatus>2</AdminStatus>
            </Interface>
          </Interfaces>
        </Ifmgr>
      </top>
    </filter>
  </get>
</rpc>
```

当用户在行的位置上放置的某个属性的名称和当前表的某个列的名称一致，则这个属性的值将与用户下发配置中同名称的列的值进行严格匹配。例如，上面例子用属性名方式的等价 XML 请求为：

```
<rpc message-id ="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0"
xmlns:data="http://www.h3c.com/netconf/data:1.0">
        <Ifmgr>
          <Interfaces>
            <Interface data:AdminStatus="2" />
          </Interfaces>
        </Ifmgr>
```

```
      </top>
    </filter>
  </get>
</rpc>
```

以上两个 NETCONF 报文示例说明：通过两种严格匹配方式均可以获取所有状态为 UP 的接口配置信息。

5.3 正则表达式匹配

5.3.1 正则表达式匹配简介

当过滤条件比较复杂时，可以在指定元素上设置 regExp 属性为一个正则表达式，以达到过滤的目的。支持正则表达式匹配的数据类型包括：各种整数、各种日期和时间、各种字符串、IPv4 地址、IPv4 掩码、IPv6 地址、MAC 地址、OID、时区。

提示

Comware 系统中，正则表达式依赖 ucLibC、gLibC 等库实现，不同库的实现可能存在某些差异。为防止出现设备无法解析正则表达式的情况，XML 请求中应当避免出现特别复杂的正则表达式。

如下为一个 NETCONF 报文示例，用于获取接口的描述信息，并要求这些描述信息全部为大写字母，不能有其它字符。

```
<rpc message-id="1-0" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:h3c="http://www.h3c.com/netconf/base:1.0" >
  <get-config>
    <source>
      <running/>
    </source>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
        <Ifmgr>
          <Interfaces>
            <Interface>
              <Description h3c:regExp="^[A-Z]*$" />
            </Interface>
          </Interfaces>
        </Ifmgr>
      </top>
    </filter>
  </get-config>
</rpc>
```

正则表达式中常用的特殊字符匹配规则如表 5-1 所示。

表5-1 正则表达式中的特殊字符描述表

特殊字符	含义	举例
^	匹配以指定字符开始的行	^u只能匹配以u开始的行，不能匹配以Au开始的行

特殊字符	含义	举例
$	匹配以指定字符结束的行	u$只能匹配以u结尾的行，不能匹配以uA结尾的行
.	通配符，可代表任何一个字符	.s可以匹配as和bs等
*	匹配星号前面的字符或字符串零次或多次	• zo*可以匹配 z 以及 zoo • (zo)*可以匹配 zo 以及 zozo
+	匹配+前面的字符或字符串一次或多次	zo+可以匹配zo以及zoo，但不能匹配z
|	匹配|左边或右边的整个字符串	def|int只能匹配包含def或者int的字符串所在的行
( )	表示字符串，一般与“+”或“*”等符号一起使用	(123A)表示字符串123A；408(12)+可以匹配40812或408121212等字符串，但不能匹配408
\index	表示重复一次指定字符串，字符串是指\前用()括起来的字符串，index对应\前字符串的顺序号按从左至右的顺序从1开始编号：如果\前面只有一个字符串，则index只能为1；如果\前面有n个字符串，则index可以为1到n中的任意整数	(string)\1表示把string重复一次，匹配的字符串必须包含stringstring；(string1)(string2)\2表示把string2重复一次，匹配的字符串必须包含string1string2string2；(string1)(string2)\1\2表示先把string1重复一次，再重复一次string2，匹配的字符串必须包含string1string2string1string2
[ ]	表示字符选择范围，将以选择范围内的单个字符为条件进行匹配，只要字符串里包含该范围的某个字符就能匹配到	• [16A]表示可以匹配到的字符串只需要包含 1、6 或 A 中任意一个 • [1-36A] 表示可以匹配到的字符串只需要包含 1、2、3、6 或 A 中任意一个（-为连接符） 如果]需要作为普通字符出现在[ ]内时，必须把]写在[ ]中字符的最前面，形如[]string]，才能匹配到]。[没有这样的限制
[^]	表示选择范围外的字符，将以单个字符为条件进行匹配，只要字符串里包含该范围外的某个字符就能匹配到	[^16A]表示可匹配的字符串只需要包含1、6和A之外的任意字符，该字符串也可以包含字符1、6或A，但不能只包含这三个字符。例如[^16A]可以匹配abc、m16，不能匹配1、16、16A
{n}	n是一个非负整数，匹配n次	o{2}不能匹配Bob，但是能匹配food
{n,}	n是一个非负整数，至少匹配n次	o{2,}不能匹配Bob，但能匹配fooooood
{n,m}	m和n均为非负整数，其中n小于等于m。只要字符串里包含n到m个某字符就能匹配到	o{1,3}能匹配fod、food、foood、foooood，但不能匹配fd
\<	匹配包含指定字符串的字符串，字符串前面如果有字符则不能是数字、字母和下划线	\<do匹配单词domain，还可以匹配字符串doa
\>	匹配包含指定字符串的字符串，字符串后面如果有字符则不能是数字、字母和下划线	do\>匹配单词undo，还可以匹配字符串cdo
\b	匹配一个单词边界，也就是指单词和空格间的位置	er\b可以匹配never，但不能匹配verb \ber可以匹配erase，但不能匹配verb
\B	匹配非单词边界	er\B能匹配verb，但不能匹配never
\w	\w等效于[A-Za-z0-9_]，是数字、字母或下划线	v\w能匹配vlan，v\w还能匹配service

特殊字符	含义	举例
\W	\W等效于[^A-Za-z0-9_]，是除了数字、字母和下划线之外的任意字符	\Wa可以匹配-a，但是不能匹配2a和ba等
\	转义操作符，\后紧跟本表中罗列的单个特殊字符时，将去除特殊字符的特定含义	• \\可以匹配包含\的字符串 • \^可以匹配包含^的字符串 • \\b 可以匹配包含\b 的字符串

5.3.2 正则表达式操作举例

1. 组网需求

获取 Ifmgr 模块 Interfaces 表下 Description 列包含冒号的所有数据。

2. 配置步骤

进入 XML 视图。

```
<Sysname> xml
```

进行能力交换。请将以下报文拷贝、粘贴到客户端。

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
</hello>
```

获取 Ifmgr 模块 Interfaces 表下 Description 列包含冒号的所有数据。请将以下报文拷贝、粘贴到客户端。

```
<?xml version="1.0"?>
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:reg="http://www.h3c.com/netconf/base:1.0">
  <get>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        <Ifmgr>
          <Interfaces>
            <Interface>
              <Description  reg:regExp=":" />
            </Interface>
          </Interfaces>
        </Ifmgr>
      </top>
    </filter>
  </get>
</rpc>
```

3. 结果验证

如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:reg="http://www.h3c.com/netconf/base:1.0" message-id="100">
  <data>
    <top xmlns="http://www.h3c.com/netconf/data:1.0">
      <Ifmgr>
        <Interfaces>
          <Interface>
            <IfIndex>2681</IfIndex>
            <Description>Ten-GigabitEthernet1/0/49:1 Interface</Description>
          </Interface>
          <Interface>
            <IfIndex>2682</IfIndex>
            <Description>Ten-GigabitEthernet1/0/49:2 Interface</Description>
          </Interface>
          <Interface>
            <IfIndex>2683</IfIndex>
            <Description>Ten-GigabitEthernet1/0/49:3 Interface</Description>
          </Interface>
          <Interface>
            <IfIndex>2684</IfIndex>
            <Description>Ten-GigabitEthernet1/0/49:4 Interface</Description>
          </Interface>
          <Interface>
            <IfIndex>2685</IfIndex>
            <Description>Ten-GigabitEthernet1/0/50:1 Interface</Description>
          </Interface>
          <Interface>
            <IfIndex>2686</IfIndex>
            <Description>Ten-GigabitEthernet1/0/50:2 Interface</Description>
          </Interface>
          <Interface>
            <IfIndex>2687</IfIndex>
            <Description>Ten-GigabitEthernet1/0/50:3 Interface</Description>
          </Interface>
          <Interface>
            <IfIndex>2688</IfIndex>
            <Description>Ten-GigabitEthernet1/0/50:4 Interface</Description>
          </Interface>
          <Interface>
            <IfIndex>2689</IfIndex>
            <Description>Ten-GigabitEthernet1/0/51:1 Interface</Description>
          </Interface>
          <Interface>
            <IfIndex>2690</IfIndex>
            <Description>Ten-GigabitEthernet1/0/51:2 Interface</Description>
          </Interface>
          <Interface>
            <IfIndex>2691</IfIndex>
```

```
            <Description>Ten-GigabitEthernet1/0/51:3 Interface</Description>
          </Interface>
          <Interface>
            <IfIndex>2692</IfIndex>
            <Description>Ten-GigabitEthernet1/0/51:4 Interface</Description>
          </Interface>
        </Interfaces>
      </Ifmgr>
    </top>
  </data>
</rpc-reply>
```

5.4 条件匹配

5.4.1 条件匹配简介

由于正则表达仅能够完成字符匹配，对于数值逻辑的判断过滤实现起来比较麻烦。此时，可使用条件匹配过滤功能。

条件匹配通过在元素中增加 match 属性完成，属性的值（即过滤条件）可以为数字、字符串。

表5-2 条件匹配命令

操作	命令	说明
大于	match=”more:*value*”	值大于*value*，支持的数据类型为：各种日期和时间、各种数字、各种字符串、IPv4地址、IPv4掩码、IPv6地址、MAC地址、时区
小于	match=”less:*value*”	值小于*value*，支持的数据类型为：各种日期和时间、各种数字、各种字符串、IPv4地址、IPv4掩码、IPv6地址、MAC地址、时区
不小于	match=”notLess:*value*”	值不小于*value*，支持的数据类型为：各种日期和时间、各种数字、字符串、IPv4地址、IPv4掩码、IPv6地址、MAC地址、时区
不大于	match=”notMore:*value*”	值不大于*value*，支持的数据类型为：各种日期和时间、各种数字、各种字符串、IPv4地址、IPv4掩码、IPv6地址、MAC地址、时区
等于	match=”equal:*value*”	值等于*value*，支持的数据类型为：所有类型
不等于	match=”notEqual:*value*”	值不等于*value*，支持的数据类型为：所有类型
包含	match=”include:*string*”	包含字符串*string*，支持的数据类型为：各种字符串、IPv4地址、IPv4掩码
不包含	match=”exclude:*string*”	不能包含字符串*string*，支持的数据类型为：各种字符串、IPv4地址、IPv4掩码、
开始于	match=”startWith:*string*”	以字符串*string*开头，支持的数据类型为：各种字符串、OID、IPv4地址、IPv4掩码、
结束于	match=”endWith:*string*”	以字符串*string*结束，支持的数据类型为：各种字符串、IPv4地址、IPv4掩码、

如下为一个 NETCONF 报文示例，用于获取实体扩展信息中 CPU 利用率大于 50%的实体。

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:h3c="http://www.h3c.com/netconf/base:1.0" >
```

```
  <get>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        <Device>
          <ExtPhysicalEntities>
            <Entity>
              <CpuUsage h3c:match="more:50"></CpuUsage>
            </Entity>
          </ExtPhysicalEntities>
        </Device>
      </top>
    </filter>
  </get>
</rpc>
```

5.4.2 条件匹配操作举例

1. 组网需求

获取 Ifmgr 模块 Interfaces 表下 IfIndex 值大于等于 5000 的 Name 列信息。

2. 配置步骤

进入 XML 视图。

```
<Sysname> xml
```

进行能力交换。请将以下报文拷贝、粘贴到客户端。

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>
      urn:ietf:params:netconf:base:1.0
    </capability>
  </capabilities>
</hello>
```

获取 Ifmgr 模块 Interfaces 表下 IfIndex 值大于等于 5000 的 Name 列信息。请将以下报文拷贝、粘贴到客户端。

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="http://www.h3c.com/netconf/base:1.0">
  <get>
    <filter type="subtree">
      <top xmlns="http://www.h3c.com/netconf/data:1.0">
        <Ifmgr>
          <Interfaces>
            <Interface>
              <IfIndex nc:match="more:5000"/>
              <Name/>
            </Interface>
          </Interfaces>
        </Ifmgr>
      </top>
    </filter>
```

```
  </get>
</rpc>
```

3. 结果验证

如果客户端收到类似如下的报文，则表示操作成功。

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="http://www.h3c.com/netconf/base:1.0" message-id="100">
  <data>
    <top xmlns="http://www.h3c.com/netconf/data:1.0">
      <Ifmgr>
        <Interfaces>
          <Interface>
            <IfIndex>7241</IfIndex>
            <Name>NULL0</Name>
          </Interface>
          <Interface>
            <IfIndex>7243</IfIndex>
            <Name>Register-Tunnel0</Name>
          </Interface>
        </Interfaces>
      </Ifmgr>
    </top>
  </data>
</rpc-reply>
```

5.5 可重复列的过滤

可重复的列在明确指定过滤条件进行数据过滤时，会只罗列出本行本列中符合过滤条件的列，本行本列中不符合过滤条件的列不会输出，如果一个记录都没有，则本行不输出。

5.6 具有子结构情况下的过滤

具有可重复的数据结构的情况中，结构分组的成员在明确指定过滤条件进行数据过滤时，会只罗列出所有成员都符合过滤条件的分组，不符合过滤条件或者只有部分符合过滤条件的分组不会输出，如果本行中一个符合输出的分组记录都没有，则本行不输出。

6 ncclient 作为 NETCONF 客户端编程示例

本节以 ncclient 为例，介绍开源工具作为 NETCONF 客户端，与设备（作为 NETCONF 服务器）建立 NETCONF 会话的方法。

6.1 ncclient客户端使用环境准备

6.1.1 ncclient 介绍

ncclient 是一个可以用作 NETCONF 客户端的 Python 库。它为使用者提供了非常直观的 API，并将 NETCONF 的 XML 编码特性映射到 Python 构造和习语，使得网络管理脚本的编写更加容易。

该库包含的主要功能有：

- 支持 RFC 6241 定义的所有 NETCONF 特性和功能。
- 流水线式请求。
- 支持异步 RPC 请求。
- 与标准 XML 方式一致。除非真正需要，否则无需变更。
- 扩展性强，可以根据需要，灵活添加新的传输映射和功能/操作。

6.1.2 环境准备

ncclient 所需的 Python 版本为 Python 2.7 或 Python 3.5+。

ncclient 的运行依赖如下库：

- setuptools 0.6+
- Paramiko 1.7+
- lxml 3.3.0+
- libxml2
- libxslt

如果操作系统为 Debian/Ubuntu，则还需要通过 aptitude 或者 apt-get 等方式安装如下的库：

- libxml2-dev
- libxslt1-dev

6.1.3 ncclient 安装

可以通过如下方式安装 ncclient：

- 解压安装包后安装 ncclient。以 ncclient-0.6.10.tar.gz 为例，具体过程为：
  a. 解压下载后的安装包。
  b. 打开 cmd，切换到解压后 setup.py 文件所在的目录下。
  c. 执行 python setup.py build 命令。
  d. 执行 python setup.py install 命令。

- pip 方式，即执行 pip install ncclient 命令安装 ncclient。该方式需要确保主机可以访问到目标 lib 仓库（开发环境可能对默认仓库或公网仓库有访问限制，需要和网络管理员确认）。

安装完成后，需要按照如下步骤检查是否安装成功：

(1) 打开 cmd，进入 python 视图。

(2) 输入 import ncclient 命令。如果没有报错，则表示已经安装成功了。

如果需要切换 ncclient 版本，需要先执行 pip uninstall ncclient 命令卸载 ncclient。

6.1.4 脚本使用

通过 python 命令执行脚本时，需要指定脚本文件所在的路径。例如，脚本文件*.py 保存在 C/desktop/examples 目录下，在 C/desktop 下执行的 python 命令形式如下：

```
[ncclient] $ python examples/*.py
```

6.2 建立NETCONF会话

本节以 NETCONF over SSH 访问方式为例，介绍如何通过运行 Python 脚本建立 NETCONF 会话。建立 NETCONF 会话的过程通常为：

(1) 配置 NETCONF 访问方式。

(2) 创建一个*.py 脚本，并 import 依赖库。

(3) 调用 connect 方法，通过 SSH 创建 NETCONF 会话。

6.2.2 配置 NETCONF 访问方式

选用 NETCONF over SSH 访问方式时，需要在服务端配置 SSH 用户，并开启 NETCONF over SSH 服务。具体配置方法可参见“2.4.1 配置 NETCONF over SSH 访问方式”。

配置完成后，设备上将具有可用的 SSH 服务（用户名和密码）用于和客户端建立连接。本例中，假设用户名为 ssh，密码为 00password00connect 。

6.2.3 运行脚本建立 NETCONF 会话

1. 前提条件

在建立NETCONF会话之前，需要先检查NETCONF客户端和NETCONF服务器之间能否ping通，确保 ncclient 和设备（Server）三层互通。

2. 操作步骤

在 ncclient 上创建到设备的 SSH 连接。该连接使用基于用户名和密码的身份认证方式，用户名为 ssh，密码为 00password00connect 。执行脚本调用 connect 函数时，客户端连接到设备，并自动建立一个 NETCONF 会话。

说明

脚本运行过程中，如果出现类似“No module named XXX”的提示，则使用 pip install xxx 命令安装所需要的模块即可。

可以采用如下两种脚本建立 NETCONF 会话：

- 在脚本执行时提供连接相关的参数，具体过程为：

a. 新建 demo1.py 文件，将以下内容复制到文件中。

```
# -*- coding:utf-8 -*-
import sys
from ncclient import manager
from ncclient import operations

def h3c_connection(host, port, user, password):
    return manager.connect(host=host,
                           port=port,
                           username=user,
                           password=password,
                           hostkey_verify = False,
                           device_params={'name':"h3c"},
                           allow_agent = False,
                           look_for_keys = False)

def test_connect(host, port, user, password):
    with h3c_connection(host,port=port,user=user,password=password) as m:

        n = m._session.id
        print("This session id is %s."%(n))

if __name__ == '__main__':
    # test_connect("192.168.56.120", 830, "ssh", "00password00connect ")
    test_connect(sys.argv[1],sys.argv[2],sys.argv[3],sys.argv[4])
```

其中，各参数的含义为：

– host：运行 NETCONF 服务器的设备的 IP 地址，推荐使用设备上管理口的 IP 地址。此例中，IP 地址为 192.168.56.120。

– port：建立 SSH 连接的端口号。默认为 830。

– username：SSH 用户的用户名。此例中，用户名为 ssh。

– password：SSH 用户的密码。此例中，密码为 00password00connect 。

b. 进行 cmd，并切换到 demo1.py 文件所在的目录，执行以下命令。

```
>python demo1.py 192.168.56.120 830 ssh 00password00connect
```

c. 脚本运行结果如图 6-1 所示。

图6-1 脚本运行结果示意图

```
D:\D_PythonProjects\B_github_projects\h3c_ncclient\demo>python demo1.py 192.168.
56.120 830 ssh 00password00connect
This session id is 1.
```

- 在脚本文件中提供连接相关的参数，具体过程为：

a. 新建 demo2.py 文件，将以下内容复制到文件中。

```
# -*- coding:utf-8 -*-
import sys
```

```
from ncclient import manager
from ncclient import operations

def h3c_connection(host, port, user, password):
    return manager.connect(host=host,
                           port=port,
                           username=user,
                           password=password,
                           hostkey_verify = False,
                           device_params={'name':"h3c"},
                           allow_agent = False,
                           look_for_keys = False)

def test_connect(host, port, user, password):
    with h3c_connection(host,port=port,user=user,password=password) as m:

        n = m._session.id
        print("This session id is %s."%(n))

if __name__ == '__main__':
    test_connect("192.168.56.120", 830, "ssh", "00password00connect ")
```

b. 进入 cmd，并切换到 demo2.py 文件所在的目录，执行以下命令。

```
>python demo2.py
```

6.3 ncclient基本操作及其示例

本节介绍如何在 NETCONF 会话上，执行配置管理、状态查询和事件订阅等基本操作。执行 ncclient 基本操作的过程可以概括为：

(1) 根据需要，构造不同的 RPC 报文。

(2) 执行该 Python 脚本，以完成配置、查询、订阅等操作。

6.3.2 配置（edit-config）

1. 基本操作

(1) 构造 RPC 报文，本例为创建新的 VLAN。RPC 请求报文的 XML 结构可参见《H3C Comware 7 NETCONF XML API Reference》文档。

```
VLAN_RPC = """<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
                <top xmlns="http://www.h3c.com/netconf/config:1.0">
                  <VLAN xc:operation="create">
                     <VLANs>
                        <VLANID>
                           <ID>3</ID>
                           <Name>3</Name>
                           <Description>2</Description>
                        </VLANID>
                     </VLANs>
                  </VLAN>
```

```
            </top>
        </config>
"""
```

(2) 根据配置的生效模式，选择不同的操作方式。

- 立即生效模式：执行 edit-config 操作，指定对服务器端<running/>配置集进行修改，并校验返回报文。

  ```
  rpc_obj = m.edit_config(target='running', config=VLAN_RPC)
  _check_response(rpc_obj,'VLAN_MERGE')
  ```

- 二阶段生效模式：执行 edit-config 操作，指定对服务器端<candidate/>配置集进行修改，并校验返回报文。

  ```
  rpc_obj = m.edit_config(target=' candidate ', config=VLAN_RPC)
  _check_response(rpc_obj,'VLAN_MERGE')
  ```

  完成 edit-config 操作后，通过如下命令执行 commit 操作。

  ```
  m.commit()
  ```

- 试运行模式：执行 edit-config 操作，指定对服务器端<candidate/>配置集进行修改，并校验返回报文。

  ```
  rpc_obj = m.edit_config(target=' candidate ', config=VLAN_RPC)
  _check_response(rpc_obj,'VLAN_MERGE')
  ```

  完成 edit-config 操作后，通过如下命令将配置提交试运行，并设置运行时间。

  ```
  m.commit(confirmed=True, timeout=' 300 ')
  ```

  完成 edit-config 操作后，也可以执行以下命令丢弃<candidate/>中未提交的数据，即放弃配置。

  ```
  m.discard_changes()
  ```

上述命令中的 m 为 with h3c_connection(host,port=port,user=user,password=password) as m:

2. 示例

(1) 以下为通过 edit-config 操作创建 VLAN 的 Python 程序：

```
# -*- coding:utf-8 -*-
import sys
import logging
from ncclient import manager
from ncclient import operations

log = logging.getLogger(__name__)

# 用于创建 VLAN 的 RPC 报文。
VLAN_RPC = """<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
               <top xmlns="http://www.h3c.com/netconf/config:1.0">
                  <VLAN xc:operation="create">
                     <VLANs>
                        <VLANID>
                           <ID>3</ID>
                           <Name>3</Name>
                           <Description>2</Description>
                        </VLANID>
```

```
                    </VLANs>
                </VLAN>
            </top>
        </config>
"""

# 建立与设备的连接。
def h3c_connection(host, port, user, password):
    return manager.connect(host=host,
                           port=port,
                           username=user,
                           password=password,
                           hostkey_verify = False,
                           device_params={'name':"h3c"},
                           allow_agent = False,
                           look_for_keys = False)

# 检查 RPC 回复报文。
def _check_response(rpc_obj, snippet_name):
    print("RPC reply for %s is  %s" % (snippet_name, rpc_obj.xml))
    xml_str = rpc_obj.xml
    if "<ok/>" in xml_str:
        print("%s successful"%snippet_name)
    else:
        print("Cannot successfully execute: %s" % snippet_name)

def test_edit_config_running(host, port, user, password):
    # 创建 NETCONF 会话。
    with h3c_connection(host,port=port,user=user,password=password) as m:
        # 发送 RPC 请求并检查 RPC 回复报文。
        rpc_obj = m.edit_config(target='running', config=VLAN_RPC)
        _check_response(rpc_obj,'VLAN_MERGE')

if __name__ == '__main__':
test_edit_config_running("192.168.56.120", 830, "ssh", "00password00connect ")
```

(2)　以下为通过 edit-config 操作删除 VLAN 的 Python 程序：

```
# -*- coding:utf-8 -*-
import sys
import logging
from ncclient import manager
from ncclient import operations

log = logging.getLogger(__name__)

# 用于删除 VLAN 的 RPC 报文。
VLAN_RPC_DELETE = """<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
                <top xmlns="http://www.h3c.com/netconf/config:1.0">
                   <VLAN xc:operation="delete">
```

```
                    <VLANs>
                       <VLANID>
                          <ID>4</ID>
                       </VLANID>
                    </VLANs>
                 </VLAN>
              </top>
           </config>
"""

# 建立与设备的连接。
def h3c_connection(host, port, user, password):
    return manager.connect(host=host,
                           port=port,
                           username=user,
                           password=password,
                           hostkey_verify = False,
                           device_params={'name':"h3c"})

# 检查 RPC 回复报文。
def _check_response(rpc_obj, snippet_name):
    print("RPC reply for %s is %s" % (snippet_name, rpc_obj.xml))
    xml_str = rpc_obj.xml
    if "<ok/>" in xml_str:
        print("%s successful" % snippet_name)
    else:
        print("Cannot successfully execute: %s" % snippet_name)

def test_edit_config_running(host, port, user, password):
    # 创建 NETCONF 会话。
    with h3c_connection(host, port=port, user=user, password=password) as m:
        # 发送 RPC 请求并检查 RPC 回复报文。
        rpc_obj = m.edit_config(target='running', config=VLAN_RPC_DELETE)
        _check_response(rpc_obj, 'VLAN_DELETE')

if __name__ == '__main__':
    test_edit_config_running("192.168.56.120", 830, "ssh", "00password00connect")
```

6.3.3 查询（get/get-config）

1. 基本操作

(1) 构造过滤条件，不同操作方式，报文格式不同：

- get 操作需要指定的命名空间为 xmlns=http://www.h3c.com/netconf/data:1.0。以获取接口信息为例，RPC 报文如下：

```
IF_GET_RPC = """
                <top xmlns="http://www.h3c.com/netconf/data:1.0">
                    <Ifmgr>
                       <Interfaces>
```

```
                        <Interface></Interface>
                    </Interfaces>
                </Ifmgr>
            </top>
"""
```

- get-config 操作需要指定的命名空间为 xmlns=http://www.h3c.com/netconf/config:1.0。以获取接口信息为例，RPC 报文如下：

```
IF_GET_CONFIG_RPC = """
            <top xmlns="http://www.h3c.com/netconf/config:1.0">
                <Ifmgr>
                    <Interfaces>
                        <Interface></Interface>
                    </Interfaces>
                </Ifmgr>
            </top>
"""
```

(2) 调用对应接口，发送<get>或<get-config>报文到服务器端，查询当前运行数据和配置数据。

- 调用 get 接口，发送 get 操作 RPC 请求，并将回复报文写入到 xml 文件中。

```
get_reply = m.get(("subtree", IF_GET_RPC)).data_xml
print(get_reply)
with open("%s_get.xml" % host, 'w') as f:
f.write(get_reply)
```

- 调用 get 或 get-config 接口，发送 get-config 操作 RPC 请求，并将回复报文写入到 xml 文件中。

  通过调用 get 和 get-config 接口，均可以执行 get-config 操作。调用 get 接口时，设备通过请求报文中的命令空间判断当前请求是 get 类型还是 get-config 类型。

  调用 get 接口，执行 get-config 操作的方法为：

```
get_reply2 = m.get(("subtree", IF_GET_CONFIG_RPC)).data_xml
print(get_reply2)
with open("%s_getconfig.xml" % host, 'w') as f:
f.write(get_reply2)
```

  调用 get-config 接口，执行 get-config 操作的方法为：

```
c = m.get_config(source='running',filter=('subtree',IF_GET_CONFIG_RPC)).data_xml
with open("%s_getconfig2.xml" % host, 'w') as f:
f.write(c)
```

2. 示例

以下为通过 get 或 get-config 操作获取接口信息的 Python 程序：

```
# -*- coding:utf-8 -*-
import sys
import logging
from ncclient import manager
from ncclient import operations

log = logging.getLogger(__name__)
```

```
# 获取接口信息的 RPC 报文。
IF_GET_RPC = """
              <top xmlns="http://www.h3c.com/netconf/data:1.0">
                  <Ifmgr>
                     <Interfaces>
                        <Interface></Interface>
                     </Interfaces>
                  </Ifmgr>
               </top>
"""

IF_GET_CONFIG_RPC = """
              <top xmlns="http://www.h3c.com/netconf/config:1.0">
                  <Ifmgr>
                     <Interfaces>
                        <Interface></Interface>
                     </Interfaces>
                  </Ifmgr>
               </top>
"""

# 建立与设备的连接。
def h3c_connection(host, port, user, password):
    return manager.connect(host=host,
                           port=port,
                           username=user,
                           password=password,
                           hostkey_verify = False,
                           device_params={'name':"h3c"},
                           allow_agent = False,
                           look_for_keys = False)


def test_get(host, port, user, password):
    # 创建 NETCONF 会话。
    with h3c_connection(host,port=port,user=user,password=password) as m:
        n = m._session.id
        print("This session id is %s."%(n))

        # 发送 get 操作 RPC 请求，并将回复报文写入到 xml 文件中。
        get_reply = m.get(("subtree", IF_GET_RPC)).data_xml
        print(get_reply)
        with open("%s_get.xml" % host, 'w') as f:
            f.write(get_reply)

        # 通过 get 接口发送 get-config 操作 RPC 请求，并将回复报文写入到 xml 文件中。
        get_reply2 = m.get(("subtree", IF_GET_CONFIG_RPC)).data_xml
        print(get_reply2)
```

```
        with open("%s_getconfig.xml" % host, 'w') as f:
            f.write(get_reply2)

        # 通过 get-config 接口发送 get-config 操作 RPC 请求，并将回复报文写入到 xml 文件中。
        c = m.get_config(source='running',filter=('subtree',IF_GET_CONFIG_RPC)).data_xml
        with open("%s_getconfig2.xml" % host, 'w') as f:
            f.write(c)


if __name__ == '__main__':
    test_get("192.168.56.120", 830, "ssh", "00password00connect ")
```

6.3.4 查询（get-bulk/get-bulk-config）

get-bulk 和 get-bulk-config 操作为 H3C 自行定义的私有操作类型。在 ncclient 中 get-bulk/get-bulk-config 接口和 get 接口的使用方法类似。

1. 基本操作

(1) 构造过滤条件。

以获取接口数据为例，RPC_GET_BULK 使用 get-bulk 方法，获取接口索引为<IfIndex>1536</IfIndex>之后的所有接口的数据，RPC 报文为：

```
RPC_GET_BULK = """
              <top xmlns="http://www.h3c.com/netconf/data:1.0">
                  <Ifmgr>
                     <Interfaces>
                        <Interface>
                            <IfIndex>1536</IfIndex>
                        </Interface>
                     </Interfaces>
                  </Ifmgr>
               </top>
"""
```

使用 get-bulk-config 方法获取所有接口的配置数据，其报文格式为（RPC_GET_BULK_CONFIG 的 RPC 请求中未指定索引）：

```
RPC_GET_BULK_CONFIG = """
              <top xmlns="http://www.h3c.com/netconf/config:1.0">
                  <Ifmgr>
                  </Ifmgr>
                  <GRPC>
                  </GRPC>
               </top>
"""
```

(2) 调用对应接口，发送<get-bulk>或<get-bulk-config>报文到服务器端，批量查询当前运行数据和配置数据。

- 调用 get-bulk 接口，发送 get-bulk 操作 RPC 请求。

```
reply = m.get_bulk(filter=('subtree',RPC_GET_BULK))
print(reply)
```

- 调用 get-bulk-config 接口，发送 get-bulk-config 操作 RPC 请求。

```
reply = m.get_bulk_config(source= "running", filter=('subtree',
RPC_GET_BULK_CONFIG))
print(reply)
```

2. 示例

以下为通过 get-bulk 或 get-bulk-config 操作批量获取接口信息的 Python 程序：

```
# -*- coding:utf-8 -*-
import sys
import logging
from ncclient import manager
from ncclient import operations

log = logging.getLogger(__name__)

# 通过 get-bulk 操作获取接口信息的 RPC 报文。
RPC_GET_BULK = """
              <top xmlns="http://www.h3c.com/netconf/data:1.0">
                  <Ifmgr>
                     <Interfaces>
                        <Interface>
                            <IfIndex>1536</IfIndex>
                        </Interface>
                     </Interfaces>
                  </Ifmgr>
               </top>
"""

# 通过 get-bulk-config 操作获取接口信息的 RPC 报文。
RPC_GET_BULK_CONFIG = """
              <top xmlns="http://www.h3c.com/netconf/config:1.0">
                  <Ifmgr>
                  </Ifmgr>
                  <GRPC>
                  </GRPC>
               </top>
"""

# 建立与设备的连接。
def h3c_connection(host, port, user, password):
    return manager.connect(host=host,
                           port=port,
                           username=user,
                           password=password,
                           hostkey_verify = False,
                           device_params={'name':"h3c"},
                           allow_agent = False,
                           look_for_keys = False)
```

```
def test_get_bulk(host, port, user, password):
    # 创建 NETCONF 会话。
    with h3c_connection(host,port=port,user=user,password=password) as m:
        # 发送 get-bulk 请求。
        reply = m.get_bulk(filter=('subtree',RPC_GET_BULK))
        print(reply)
        # 发送 get-bulk-config 请求。
        reply = m.get_bulk_config(source= "running", filter=('subtree',
RPC_GET_BULK_CONFIG))
        print(reply)

if __name__ == '__main__':
    test_get_bulk("192.168.56.120", 830, "ssh", "00password00connect ")
```

6.3.5 action 操作

1. 基本操作

(1) 构造 RPC action 报文。

以清除全部接口的统计信息为例，RPC 报文如下：

```
ACTION_RPC = """
              <rpc message-id="{}" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
                <action>
                    <top xmlns="http://www.h3c.com/netconf/action:1.0">
                        <Ifmgr>
                            <ClearAllIfStatistics>
                                <Clear>
                                </Clear>
                            </ClearAllIfStatistics>
                          </Ifmgr>
                        </top>
                </action>
              </rpc>
"""
```

(2) 下发 action 报文。

```
msgid = 100
rpc = ACTION_RPC.format(msgid)
m._session.send(rpc)
time.sleep(2)
```

2. 示例

以下为通过 action 操作清除全部接口统计信息的 Python 程序。

```
# -*- coding:utf-8 -*-
import sys
import logging
import  time
from ncclient import manager
```

```
from ncclient import operations

log = logging.getLogger(__name__)

# 通过 action 操作清除全部接口统计信息的 RPC 报文。
ACTION_RPC = """
              <rpc message-id="{}" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
                <action>
                    <top xmlns="http://www.h3c.com/netconf/action:1.0">
                        <Ifmgr>
                            <ClearAllIfStatistics>
                                <Clear>
                                </Clear>
                            </ClearAllIfStatistics>
                          </Ifmgr>
                        </top>
                </action>
              </rpc>
"""

# 建立与设备的连接。
def h3c_connection(host, port, user, password):
    return manager.connect(host=host,
                           port=port,
                           username=user,
                           password=password,
                           hostkey_verify = False,
                           device_params={'name':"h3c"},
                           allow_agent = False,
                           look_for_keys = False)


def test_action(host, port, user, password):
    # 创建 NETCONF 会话。
    with h3c_connection(host,port=port,user=user,password=password) as m:
        n = m._session.id
        print("This session id is %s."%(n))

        # 发送 action 报文。
        msgid = 100
        rpc = ACTION_RPC.format(msgid)
        m._session.send(rpc)
        time.sleep(2)

if __name__ == '__main__':
    test_action("192.168.56.120", 830, "ssh", "00password00connect ")
```

6.3.6 订阅（subscription）

1. 基本操作

(1) 构造订阅报文，RPC 报文如下：

```
# 创建 netconf 订阅报文。
Rpc4Subscription = """<rpc message-id="{}"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
                <create-subscription
xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
                    <stream>NETCONF</stream>
                </create-subscription>
            </rpc>
"""

# 订阅 ifmgr 事件。
RPC4Ifmgr = """
    <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:xs="http://www.h3c.com/netconf/base:1.0">
        <create-subscription
xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
            <stream>Ifmgr</stream>
        </create-subscription>
    </rpc>
"""
```

(2) 下发事件订阅报文。

```
# 为 rpc 设置 Message ID。
msgid = 100
rpc = Rpc4Subscription.format(msgid)
# 发送订阅报文。
m._session.send(rpc)
m._session.send(RPC4Ifmgr))
```

(3) 等待事件信息上报。

```
m.take_notification(block=True,timeout=100)
```

2. 示例

以下为订阅设备上接口管理事件的 Python 程序：

```
# -*- coding:utf-8 -*-
import sys
import logging
import time
from ncclient import manager
from ncclient import operations
from ncclient.operations import CreateSubscription

log = logging.getLogger(__name__)

# 创建 netconf 订阅的 RPC 报文。
```

```
Rpc4Subscription = """<rpc message-id="{}"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
                <create-subscription
xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
                    <stream>NETCONF</stream>
                </create-subscription>
            </rpc>
"""

# 订阅 ifmgr 事件的 RPC 报文。
RPC4Ifmgr = """
    <rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:xs="http://www.h3c.com/netconf/base:1.0">
        <create-subscription xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
            <stream>Ifmgr</stream>
        </create-subscription>
    </rpc>
"""

# 建立与设备的连接。
def h3c_connection(host, port, user, password):
    return manager.connect(host=host,
                           port=port,
                           username=user,
                           password=password,
                           hostkey_verify = False,
                           device_params={'name':"h3c"},
                           allow_agent = False,
                           look_for_keys = False)


def test_notification(host, port, user, password):
    # 创建 NETCONF 会话。
    with h3c_connection(host,port=port,user=user,password=password) as m:
        n = m._session.id
        print("This session id is %s."%(n))

        # 为 RPC 设置 Message ID。
        msgid = 100
        rpc = Rpc4Subscription.format(msgid)
        # 发送订阅报文。
        m._session.send(rpc)
        m._session.send(RPC4Ifmgr)
        # 等待事件信息上报。
        m.take_notification(block=True,timeout=100)


if __name__ == '__main__':
    logging.basicConfig(level=logging.DEBUG)
```

```
test_notification("192.168.56.120", 830, "ssh", "00password00connect ")
```

7 附录

7.1 附录 A Comware系统中NETCONF操作可能返回的错误说明

7.1.1 错误处理说明

- XML 中如果存在 Schema 错误，则请求不继续处理，直接返回错误。Schema 错误检查中，有些检查可能被多种形式的约束条件来检查，此时返回的错误信息与 XSD 的约束条件有关，具体情况，请参考对应的 XSD 文件。
- 本节列举的错误是常规情况下的返回值。依据模块不同，返回的错误信息可能更加详细。
- 如果 Schema 错误中的值过长，则该值有可能被截断。比如，如下 XML 报文中时间“2014-13-18T17:03:”是被截断的，此时请使用 Xpath 确认错误元素或者属性位置。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <create-subscription  xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
    <stream>NETCONF</stream>
    <filter>
      <event xmlns="http://www.h3c.com/netconf/event:1.0">
        <Code>SHELL_LOGIN</Code>
      </event>
    </filter>
    <startTime>2014-07-13T17:00:59</startTime>
    <stopTime>2014-13-18T17:03:05</stopTime>
  </create-subscription>
</rpc>
<error-message xml:lang="en">
  The value of element '/rpc/create-su
  bscription[1]/stopTime[1]' is invalid. 2014-13-18T17:03: is an invalid value of
  datetime type value.
</error-message>
```

7.1.2 Schema 验证常见错误

Comware 根据 Schema 对用户 XML 请求进行基本的格式验证，保证在大部分情况下请求格式是无误的。

表7-1 Schema 错误说明

错误提示	场景说明	错误样例
Expecting element '*path*'.	必须的元素缺失,其Xpath路径为*path*	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"><error-type>rpc</error-type><error-tag>missing-element</error-tag><error-severity>error</error-severity><error-message xml:lang="en">Expecting element '/Envelope/Body[1]/rpc[1]/copy

错误提示	场景说明	错误样例
		-config[1]/target'.</error-message> <error-info> <bad-element>target</bad-element> </error-info> </rpc-error>
The element *'a'* should have an attribute *'b'*.	元素***a***必须的属性***b***缺失	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>rpc</error-type> <error-tag>missing-attribute</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The element '/Envelope/Body[1]/rpc[1]' should have an attribute 'message-id'.</error-message> <error-info> <bad-element>rpc</bad-element> <bad-attribute>message-id</bad-attribute> </error-info> </rpc-error>
The value of element *'a'* is invalid. *reason*	元素的***a***值无效，具体原因是：*reason*。*reason*可能取值请参见表7-2	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The value of element '/Envelope/Body[1]/rpc[1]/get[1]/filter[1]/top[1]/SNMP[1]/System[1]/AgentStatus[1]' is invalid. The value does not match the enumeration.</error-message> <error-info> <bad-element>AgentStatus</bad-element> </error-info> </rpc-error>
The value of attribute *'a'* for element *'b'* is invalid. *reason*	元素***b***的属性***a***的值无效，具体原因是：*reason*。*reason*可能取值请参见表7-2	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>rpc</error-type> <error-tag>bad-attribute</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The value of attribute 'count' for element '/Envelope/Body[1]/rpc[1]/get-bulk[1]/filter[1]/top[1]' is invalid. aa is an invalid value of unsignedInt type value.</error-message> <error-info> <bad-element>top</bad-element> <bad-attribute>count</bad-attr

错误提示	场景说明	错误样例
		ibute> </error-info> </rpc-error>
The number of occurrences of element '%s/%s' is invalid.	元素出现的次数不正确	<rpc-error xmlns="urn:ietf:params:xml:ns: netconf:base:1.0"> <error-type>rpc</error-type> <error-tag>bad-element</error- tag> <error-severity>error</error-s everity> <error-message xml:lang="en">The number of occurrences of element '/Envelope/Body[1]/rpc[1]/get- config[1]/filter[1]/top[1]/Sec urityZone[1]/Interfaces[1]/Int erface[1]/ZoneName' is invalid.</error-message> <error-info> <bad-element>ZoneName</bad-ele ment> </error-info> </rpc-error>
Unexpected element '%s'.	%s元素不应在此出现	<rpc-error xmlns="urn:ietf:params:xml:ns: netconf:base:1.0"> <error-type>application</error -type> <error-tag>unknown-element</er ror-tag> <error-severity>error</error-s everity> <error-message xml:lang="en">Unexpected element '/Envelope/Body[1]/rpc[1]/get[1]/filter[1]/top[1]/Device[1]/ SummerTime[1]'.</error-message > <error-info> <bad-element>SummerTime</bad-e lement> </error-info> </rpc-error>
Unexpected attribute '%s' of element '%s'.	元素%s中不应出现%s属性	<rpc-error xmlns="urn:ietf:params:xml:ns: netconf:base:1.0"> <error-type>application</error -type> <error-tag>unknown-attribute</ error-tag> <error-severity>error</error-s everity> <error-message xml:lang="en">Unexpected attribute 'match' of element '/Envelope/Body[1]/rpc[1]/get[1]/filter[1]/top[1]/SNMP[1]/Sy stem[1]/AgentStatus[1]'.</erro r-message> <error-info> <bad-element>AgentStatus</bad- element> <bad-attribute>match</bad-attr ibute> </error-info> </rpc-error>

错误提示	场景说明	错误样例
The child elements of '%s' are invalid.	指定位置元素下的子元素出现次数不正确	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"><error-type>rpc</error-type><error-tag>bad-element</error-tag><error-severity>error</error-severity><error-message xml:lang="en">The child elements of '/Envelope/Body[1]' are invalid.</error-message><error-info><bad-element>Body</bad-element></error-info></rpc-error>
Element '%s' can not have a textual child element.	元素%s不能包含文本	<?xml version="1.0" encoding="UTF-8"?><rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101"><rpc-error><error-type>rpc</error-type><error-tag>bad-element</error-tag><error-severity>error</error-severity><error-message xml:lang="en">Element '/rpc' can not have a textual child element.</error-message><error-info><bad-element>rpc</bad-element></error-info></rpc-error></rpc-reply>
Element '%s' can not have a child element.	元素%s不能包含子节点	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"><error-type>rpc</error-type><error-tag>bad-element</error-tag><error-severity>error</error-severity><error-message xml:lang="en">Element '/Envelope/Body[1]/rpc[1]/get[1]/filter[1]/top[1]/License[1]/Systems[1]/System[1]/Max[1]' can not have a child element.</error-message><error-info><bad-element>Max</bad-element></error-info></rpc-error>

表7-2 Schema 元素或者属性值错误的具体情况

错误原因	场景说明	错误样例
value is an invalid value of *type* type value.	元素或者属性的值不是一个有效的XSD类型的值 • *value*：XML 中元素或者属性的值，如果值过长，将被截断 • *type*：当前错误位置的数据类型，报告的类型是	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"><error-type>rpc</error-type><error-tag>bad-element</error-tag><error-severity>error</error-severity><error-message xml:lang="en">The value of element

错误原因	场景说明	错误样例
	XSD 的数据类型名称	'/Envelope/Body[1]/rpc[1]/action[1]/top[1]/License[1]/Operations[1]/Operation[1]/DeviceNode[1]/Chassis[1]' is invalid. 12345678901234565 is an invalid value of unsignedShort type value.</error-message> <error-info> <bad-element>Chassis</bad-element> </error-info> </rpc-error>
%s cannot be empty.	元素值不能为空	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>rpc</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The value of element '/Envelope/Body[1]/rpc[1]/action[1]/top[1]/License[1]/Operations[1]/Operation[1]/OpType[1]' is invalid. unsignedInt cannot be empty.</error-message> <error-info> <bad-element>OpType</bad-element> </error-info> </rpc-error>
The value does not match the regular expression.	元素值和XSD定义的正则表达式不匹配	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>rpc</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The value of element '/Envelope/Body[1]/rpc[1]/action[1]/top[1]/License[1]/Operations[1]/Operation[1]/OpTarget[1]/ActivationKey[1]' is invalid. The value does not match the regular expression.</error-message> <error-info> <bad-element>ActivationKey</bad-element> </error-info> </rpc-error>
The value does not match the enumeration.	元素值和XSD定义的enumeration枚举值不匹配	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The value of element

错误原因	场景说明	错误样例
		'/Envelope/Body[1]/rpc[1]/get[1]/filter[1]/top[1]/SNMP[1]/System[1]/AgentStatus[1]' is invalid. The value does not match the enumeration.</error-message> <error-info> <bad-element>AgentStatus</bad-element> </error-info> </rpc-error>
The length must be equal to %s.	元素长度和XSD定义的长度不相等	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>rpc</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The value of element '/Envelope/Body[1]/rpc[1]/edit-config[1]/config[1]/top[1]/Syslog[1]/LogHosts[1]/Host[1]/VRF[1]' is invalid. The length must be equal to 2.</error-message> <error-info> <bad-element>VRF</bad-element> </error-info> </rpc-error>
The total digit length must be equal to %s.	元素的数字总长度和XSD定义的长度不相等	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>rpc</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The value of element '/Envelope/Body[1]/rpc[1]/edit-config[1]/config[1]/top[1]/Syslog[1]/LogHosts[1]/Host[1]/VRF[1]' is invalid. The total digit length must be equal to 2.</error-message> <error-info> <bad-element>VRF</bad-element> </error-info> </rpc-error>
The length must be less than or equal to %s.	元素的长度大于XSD定义的maxLength最大长度	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>rpc</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The value of element '/Envelope/Body[1]/rpc[1]/action[1]/top[1]/License[1]/Operations[1]/Operation[1]/OpTarget[1]/ActivationFile[1]' is invalid. The length must be less

错误原因	场景说明	错误样例
		than or equal to 255.</error-message> <error-info> <bad-element>ActivationFile</bad-element> </error-info> </rpc-error>
The length must be greater than or equal to %s.	元素的长度小于XSD定义的minLength最小长度	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>rpc</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The value of element '/Envelope/Body[1]/rpc[1]/action[1]/top[1]/License[1]/Operations[1]/Operation[1]/OpTarget[1]/ActivationKey[1]' is invalid. The length must be greater than or equal to 1.</error-message> <error-info> <bad-element>ActivationKey</bad-element> </error-info> </rpc-error>
The value must be less than %s.	元素的值大于等于XSD定义的maxExclusive最大值	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>rpc</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The value of element '/Envelope/Body[1]/rpc[1]/edit-config[1]/config[1]/top[1]/Syslog[1]/LogHosts[1]/Host[1]/Port[1]' is invalid. The value must be less than 5.</error-message> <error-info> <bad-element>Port</bad-element> </error-info> </rpc-error>
The value must be greater than %s.	元素的值小于等于XSD定义的minExclusive最小值	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>rpc</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The value of element '/Envelope/Body[1]/rpc[1]/edit-config[1]/config[1]/top[1]/Syslog[1]/LogHosts[1]/Host[1]/Port[1]' is invalid. The value must be greater than 2.</error-message>

错误原因	场景说明	错误样例
		<error-info> <bad-element>Port</bad-element> </error-info> </rpc-error>
The value must be less than or equal to %s.	元素的值大于XSD定义的maxInclusive最大值	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>rpc</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The value of element '/Envelope/Body[1]/rpc[1]/get[1]/filter[1]/top[1]/ACL[1]/Groups[1]/Group[1]/GroupID[1]' is invalid. The value must be less than or equal to 5999.</error-message> <error-info> <bad-element>GroupID</bad-element> </error-info> </rpc-error>
The value must be greater than or equal to %s.	元素的值小于XSD定义的minInclusive最小值	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>rpc</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The value of element '/Envelope/Body[1]/rpc[1]/get[1]/filter[1]/top[1]/ACL[1]/Groups[1]/Group[1]/GroupID[1]' is invalid. The value must be greater than or equal to 2000.</error-message> <error-info> <bad-element>GroupID</bad-element> </error-info> </rpc-error>

7.1.3 NETCONF 处理中的错误

表7-3 NETCONF 处理中的错误说明

错误提示	场景说明	错误样例
Invalid value.	设置操作的值不符合模块要求	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>invalid-value</error-tag> <error-severity>error</error-severity> <error-path>/Envelope/Body[1]/rpc[1]/edit-config[1]/config[1]/top[1]/Syslo

错误提示	场景说明	错误样例
		g[1]/LogHosts[1]/Host[1]</error-path> <error-message xml:lang="en">Invalid value.</error-message> <error-info> <top xmlns="http://www.h3c.com/netconf/config:1.0"> <Syslog> <LogHosts> <Host> <Address>1.#.1.1</Address> <VRF>aa</VRF> </Host> </LogHosts> </Syslog> </top> </error-info> </rpc-error>
Can't issue the configuration because of configuration conflict.	设置操作的配置与系统现有配置冲突	视具体模块而定
Configuration already exists.	执行Create操作时，要创建的配置已存在（配置不是默认值）	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>data-exists</error-tag> <error-severity>error</error-severity> <error-path>/Envelope/Body[1]/rpc[1]/edit-config[1]/config[1]/top[1]/SNMP[1]/Communities[1]/Community[1]</error-path> <error-message xml:lang="en">Configuration already exists.</error-message> <error-info> <top xmlns="http://www.h3c.com/netconf/config:1.0"> <SNMP> <Communities> <Community> <Name>123</Name> <Type>1</Type> </Community> </Communities> </SNMP> </top> </error-info> </rpc-error>
Configuration target does not exist.	SET的对象不存在	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>invalid-value</error-tag> <error-severity>error</error-severity> <error-path>/Envelope/Body[1]/rpc[1]/edit-config[1]/config[1]/top[1]/SNMP[1]/Communities[1]/Community[1]</error-path> <error-message xml:lang="en">Configuration target does not exist.</error-message> <error-info> <top xmlns="http://www.h3c.com/netconf/config:1.0">

错误提示	场景说明	错误样例
		<SNMP> <Communities> <Community> <Name>12311</Name> <Type>1</Type> </Community> </Communities> </SNMP> </top> </error-info> </rpc-error>
Configuration does not exist.	执行Delete操作，要删除的配置不存在（配置为默认配置）	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>application</error-type> <error-tag>data-missing</error-tag> <error-severity>error</error-severity > <error-path>/Envelope/Body[1]/rpc[1]/ edit-config[1]/config[1]/top[1]/SNMP[1]/Communities[1]/Community[1]</error -path> <error-message xml:lang="en">Configuration does not exist.</error-message> <error-info> <top xmlns="http://www.h3c.com/netconf/con fig:1.0"> <SNMP> <Communities> <Community> <Name>1231</Name> <Type/> </Community> </Communities> </SNMP> </top> </error-info> </rpc-error>
Current session has held the lock.	在同一个会话中，对当前配置重复加锁	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>protocol</error-type> <error-tag>lock-denied</error-tag> <error-severity>error</error-severity ><error-message xml:lang="en">Current session has held the lock.</error-message> <error-info> <session-id>1</session-id> </error-info> </rpc-error>
Unlock failed because no NETCONF lock exists.	在同一个会话中，对当前配置重复解锁	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>protocol</error-type> <error-tag>operation-failed</error-ta g> <error-severity>error</error-severity ><error-message xml:lang="en">Unlock failed because no NETCONF lock exists.</error-message> </rpc-error>
Unlock Failed because the lock is not held by current session.	在非当前会话中，对已加锁的配置进行解锁	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>protocol</error-type>

错误提示	场景说明	错误样例
		<error-tag>operation-failed</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">Unlock Failed because the lock is not held by current session.</error-message> </rpc-error>
Lock failed because the NETCONF lock is held by another session.	在非当前会话中，对已加锁的配置进行加锁	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>protocol</error-type> <error-tag>lock-denied</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">Lock failed because the NETCONF lock is held by another session.</error-message> <error-info> <session-id>1</session-id> </error-info> </rpc-error>
Can not edit tables with only table names.	执行Delete操作，请求报文中只指定到表	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>invalid-value</error-tag> <error-severity>error</error-severity> <error-path>/Envelope/Body[1]/rpc[1]/edit-config[1]/config[1]/top[1]/Device[1]/Base[1]</error-path> <error-message xml:lang="en">Can not edit tables with only table names.</error-message> </rpc-error>
When the delete or remove operation is issued, data cannot be assigned to non-index columns.	执行Delete或Remove操作时，普通列不能带值	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>invalid-value</error-tag> <error-severity>error</error-severity> <error-path>/Envelope/Body[1]/rpc[1]/edit-config[1]/config[1]/top[1]/Device[1]/Base[1]/HostName[1]</error-path> <error-message xml:lang="en">When the delete or remove operation is issued, data cannot be assigned to non-index columns.</error-message> </rpc-error>
When the delete or remove operation is issued, data cannot be assigned to non-index columns.	下发删除操作时，索引列不可以不指定值	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>invalid-value</error-tag> <error-severity>error</error-severity> <error-path>/Envelope/Body/rpc/edit-config/config/openconfig-acl:acl/acl-sets/acl-set[name=''][type='ACL_IPV4']/config/description</error-path> <error-message xml:lang="en">When the delete or remove

错误提示	场景说明	错误样例
		operation is issued, data cannot be assigned to non-index columns.</error-message> </rpc-error>
When the create, merge, or replace operation is issued, empty data cannot be assigned to columns.	下发create/merge/replace等操作时，列不可以不指定值	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>invalid-value</error-tag> <error-severity>error</error-severity> <error-path>/Envelope/Body/rpc/edit-config/config/openconfig-acl:acl/acl-sets/acl-set[name=''][type='ACL_IPV4']/config/description</error-path> <error-message xml:lang="en"> When the create, merge, or replace operation is issued, empty data cannot be assigned to columns.</error-message> </rpc-error>
For a <get-bulk> or <get-bulk-config> operation request, a table can't contain both non-index columns and subtables, and it can contain only one subtable.	执行get-bulk操作或get-bulk-config操作时，请求报文中同时包含了普通列和子表	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>operation-not-supported</error-tag><error-severity>error</error-severity> <error-path>/Envelope/Body[1]/rpc[1]/get-bulk[1]/filter[1]/top[1]/GRPC[1]/Subscriptions[1]/Subscription[1]/SourceAddress[1]</error-path> <error-message xml:lang="en">For a <get-bulk> or <get-bulk-config> operation request, a table can't contain both non-index columns and subtables, and it can contain only one subtable.</error-message> </rpc-error>
When the create, merge, or replace operation is issued, empty data cannot be assigned to columns.	执行create、merge或replace操作时，普通列带空值	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>invalid-value</error-tag> <error-severity>error</error-severity> <error-path>/Envelope/Body[1]/rpc[1]/edit-config[1]/config[1]/top[1]/Fundamentals[1]/WebUI[1]</error-path> <error-message xml:lang="en">When the create, merge, or replace operation is issued, empty data cannot be assigned to columns.</error-message> </rpc-error>
The configuration not supported on the specified target.	当前对象不支持此配置 在一个只支持二层接口的表里，下发了一个三层接口来进行设置操作，也会返回此错误	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>invalid-value</error-tag> <error-severity>error</error-severity> <error-path>/Envelope/Body[1]/rpc[1]/edit-config[1]/config[1]/top[1]/Ifmgr[1]/Interfaces[1]/Interface[1]/Loopba

错误提示	场景说明	错误样例
		ck[1]</error-path> <error-message xml:lang="en">The configuration not supported on the specified target.</error-message> <error-info> <top xmlns="http://www.h3c.com/netconf/config:1.0"> <Ifmgr> <Interfaces> <Interface> <IfIndex>449</IfIndex> <Loopback>1</Loopback> </Interface> </Interfaces> </Ifmgr> </top> </error-info> </rpc-error>
The session does not have the privilege to process the request.	当前用户对当前对象没有RBAC权限	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>resource-denied</error-tag> <error-severity>error</error-severity><error-path>/rpc/edit-config[1]/config[1]/top[1]/Device[1]/Base[1]/TimeZone[1]/ZoneName[1]</error-path> <error-message xml:lang="en">The session does not have the privilege to process the request.</error-message> <error-info> <top xmlns="http://www.h3c.com/netconf/config:1.0"> <Device> <Base> <TimeZone> <ZoneName>1</ZoneName> </TimeZone> </Base> </Device> </top> </error-info> </rpc-error>
Resources are insufficient for processing the request.	系统资源不足，不能完成本次操作。比如，配置资源已经达到了系统规格上限	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>resource-denied</error-tag> <error-severity>error</error-severity><error-path>/rpc/edit-config[1]/config[1]/top[1]/Device[1]/Base[1]/TimeZone[1]/ZoneName[1]</error-path> <error-message xml:lang="en">Resources are insufficient for processing the request..</error-message> <error-info> <top xmlns="http://www.h3c.com/netconf/config:1.0"> <Device> <Base> <TimeZone> <ZoneName>1</ZoneName>

错误提示	场景说明	错误样例
		</TimeZone> </Base> </Device> </top> </error-info> </rpc-error>
The system does not support this operation.	不支持当前操作	视具体模块而定
Operation failed.	上面所述场景之外的错误	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>application</error-type> <error-tag>operation-failed</error-ta g> <error-severity>error</error-severity > <error-path>/Envelope/Body[1]/rpc[1]/ edit-config[1]/config[1]/top[1]/SNMP[1]/System[1]</error-path> <error-message xml:lang="en">Operation failed.</error-message> <error-info> <top xmlns="http://www.h3c.com/netconf/con fig:1.0"> <SNMP> <System> <AgentStatus>disable</AgentStatus> </System> </SNMP> </top> </error-info> </rpc-error>
The session cannot kill itself.	在当前会话中删除当前会话	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>application</error-type> <error-tag>invalid-value</error-tag> <error-severity>error</error-severity > <error-message xml:lang="en">The session cannot kill itself.</error-message> </rpc-error>
Hello is required before any other operation.	创建订阅时没有先发送hello报文	<?xml version="1.0" encoding="UTF-8"?> <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf :base:1.0" xmlns:xs="http://www.h3c.com/netconf/ base:1.0" message-id="100"> <rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>application</error-type> <error-tag>operation-failed</error-ta g> <error-severity>error</error-severity > <error-message xml:lang="en">Hello is required before any other operation.</error-message> </rpc-error> </rpc-reply>]]>]
Event subscription already exists.	重复创建相同事件流的订阅	<?xml version="1.0" encoding="UTF-8"?> <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf :base:1.0" message-id="100"> <rpc-error

错误提示	场景说明	错误样例
		xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>protocol</error-type> <error-tag>operation-failed</error-ta g> <error-severity>error</error-severity > <error-message xml:lang="en">Event subscription already exists.</error-message> </rpc-error> </rpc-reply>
The event stream name can't be empty.	创建订阅时，<stream>标签中没有填写事件流名称	<?xml version="1.0" encoding="UTF-8"?> <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf :base:1.0" message-id="100"> <rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>protocol</error-type> <error-tag>operation-not-supported</e rror-tag> <error-severity>error</error-severity > <error-message xml:lang="en">The event stream name can't be empty.</error-message> </rpc-error> </rpc-reply>
You don't have the privilege to perform this NETCONF operation.	当前用户没有创建订阅的RBAC权限	<?xml version="1.0" encoding="UTF-8"?> <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf :base:1.0" xmlns:xs="http://www.h3c.com/netconf/ base:1.0" message-id="100"> <rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>application</error-type> <error-tag>access-denied</error-tag> <error-severity>error</error-severity > <error-message xml:lang="en">You don't have the privilege to perform this NETCONF operation.</error-message> </rpc-error> </rpc-reply>
You don't have the privilege to subscribe the *event-name* event.	当前用户没有订阅指定事件流中具体事件的RBAC权限，创建订阅失败	<?xml version="1.0" encoding="UTF-8"?> <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf :base:1.0" xmlns:xs="http://www.h3c.com/netconf/ base:1.0" message-id="100"> <rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>application</error-type> <error-tag>access-denied</error-tag> <error-severity>error</error-severity > <error-message xml:lang="en">You don't have the privilege to subscribe the InterfaceEvent event.</error-message> </rpc-error> </rpc-reply>
The element stopTime is required.	创建订阅时指定了startTime而没有指	<?xml version="1.0" encoding="UTF-8"?> <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf

错误提示	场景说明	错误样例
	定stopTime	:base:1.0" xmlns:xs="http://www.h3c.com/netconf/ base:1.0" message-id="100"> <rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>protocol</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity > <error-message xml:lang="en">The element stopTime is required.</error-message> <error-info><bad-element>stopTime</ba d-element> </error-info> </rpc-error> </rpc-reply>
The startTime should be earlier than the current system time.	创建订阅时，startTime晚于当前时间	<?xml version="1.0" encoding="UTF-8"?> <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf : base:1.0" xmlns:xs="http://www.h3c.com/netconf/ base:1.0" message-id="100"> <rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>protocol</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity > <error-message xml:lang="en">The startTime should be earlier than the current system time.</error-message> <error-info> <bad-element>startTime</bad-element> </error-info> </rpc-error> </rpc-reply>]]>]
The startTime should be earlier than the stopTime.	创建订阅时，startTime晚于stopTime	<?xml version="1.0" encoding="UTF-8"?> <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf :base:1.0" xmlns:xs="http://www.h3c.com/netconf/ base:1.0" message-id="100"> <rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>protocol</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity > <error-message xml:lang="en">The startTime should be earlier than the stopTime.</error-message> <error-info> <bad-element>startTime</bad-element> </error-info> </rpc-error> </rpc-reply>
The stopTime should be later than the current system time.	创建订阅时，stopTime早于当前时间	<?xml version="1.0" encoding="UTF-8"?> <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf :base:1.0" xmlns:xs="http://www.h3c.com/netconf/ base:1.0" message-id="100"> <rpc-error

错误提示	场景说明	错误样例
		xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>protocol</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity > <error-message xml:lang="en">The stopTime should be later than the current system time.</ error-message> <error-info> <bad-element>stopTime</bad-element> </error-info> </rpc-error> </rpc-reply>]]>]
Maximum number of NETCONF sessions exceeded.	创建NETCONF会话的个数达到上限	<env:Envelope xmlns:env="http://www.w3.org/2003/05/ soap-envelope"> <env:Body> <env:Fault> <env:Code> <env:Value>env:Receiver</env:Value> </env:Code> <env:Reason> <env:Text xml:lang="en"> Maximum number of NETCONF sessions exceeded.</env:Text> </env:Reason> </env:Fault> </env:Body> </env:Envelope>
Operation failed. flash:/startup.cfg can't be overwritten because the OverWrite flag is false.	执行NETCONF的save操作时，OverWrite属性值为false	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>application</error-type> <error-tag>operation-failed</error-ta g> <error-severity>error</error-severity > <error-message xml:lang="en">Operation failed. flash:/startup.cfg can't be overwritten because the OverWrite flag is false.</error-message> </rpc-error>
Failed to set HTTPS server status because the specified SSL server policy doesn't exist.	设置一个不存在的ssl server-policy	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>application</error-type> <error-tag>operation-failed</error-ta g> <error-severity>error</error-severity > <error-path>/Envelope/Body[1]/rpc[1]/ edit-config[1]/config[1]/top[1]/Funda mentals[1]/WebUI[1]/HTTPS[1]/SSLPolic yName[1]</error-path> <error-message xml:lang="en">Failed to set HTTPS server status because the specified SSL server policy doesn't exist.</error-message> <error-info> <top xmlns="http://www.h3c.com/netconf/con fig:1.0"> <Fundamentals> <WebUI> <HTTPS> <SSLPolicyName>hjhkjhkj</SSLPolicyNam e> </HTTPS> </WebUI> </Fundamentals> </top>

错误提示	场景说明	错误样例
		</error-info> </rpc-error>
The replace operation does not support incremental-edit.	增量下发不支持replace操作	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>application</error-type> <error-tag>operation-not-supported</e rror-tag> <error-severity>error</error-severity > <error-path>/Envelope/Body[1]/rpc[1]/ edit-config[1]/config[1]/top[1]/VLAN[1]/TrunkInterfaces[1]/Interface[1]</e rror-path> <error-message xml:lang="en">The replace operation does not support incremental-edit.</error-message> </rpc-error>
The incremental column must have value.	增量下发的列必须有值	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>application</error-type> <error-tag>invalid-value</error-tag> <error-severity>error</error-severity > <error-path>/Envelope/Body[1]/rpc[1]/ edit-config[1]/config[1]/top[1]/VLAN[1]/TrunkInterfaces[1]/Interface[1]/Pe rmitVlanList[1]</error-path> <error-message xml:lang="en">The incremental column must have value.</error-message> </rpc-error>
The value of attribute 'count' is invalid. Ancestor node index is incomplete.	执行get或get-bulk操作时，子表节点指定count，其祖先节点索引列不全	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>application</error-type> <error-tag>invalid-value</error-tag> <error-severity>error</error-severity > <error-path>/Envelope/Body[1]/rpc[1]/ get-bulk[1]/filter[1]/top[1]/GRPC[1]/ Subscriptions[1]/Subscription[1]/Dest inationProfiles[1]</error-path> <error-message xml:lang="en">The value of attribute 'count' is invalid. Ancestor node index is incomplete.</error-message> </rpc-error>
The NETCONF_MONITOR_EXTENSION stream must include filter information.	订阅监控事件必须包含过滤信息	<?xml version="1.0" encoding="UTF-8"?> <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf :base:1.0" message-id="100"> <rpc-error xmlns="urn:ietf:params:xml:ns:netconf :base:1.0"> <error-type>application</error-type> <error-tag>operation-not-supported</e rror-tag> <error-severity>error</error-severity > <error-message xml:lang="en">The NETCONF_MONITOR_EXTENSION stream must include filter

错误提示	场景说明	错误样例
		information.</error-message> </rpc-error> </rpc-reply>
The NETCONF_MONITOR_EXTENSION stream must have NetconfMonitor information.	订阅监控事件必须包含NetconfMonitor信息	<?xml version="1.0" encoding="UTF-8"?> <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="100"> <rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>operation-not-supported</error-tag><error-severity>error</error-severity> <error-message xml:lang="en">The NETCONF_MONITOR_EXTENSION stream must have NetconfMonitor information.</error-message> </rpc-error> </rpc-reply>]]>]
Incorrect name for Stream NETCONF_MONITOR_EXTENSIO.	创建订阅时，流名称错误	<?xml version="1.0" encoding="UTF-8"?> <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="100"> <rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>operation-not-supported</error-tag><error-severity>error</error-severity><error-message xml:lang="en">Incorrect name for Stream NETCONF_MONITOR_EXTENSIO.</error-message> </rpc-error> </rpc-reply>
The subscribing to event time must be in the range of 1970 to 2035.	创建订阅时，startTime或stopTime的时间不在1970年-2035年之间	<?xml version="1.0" encoding="UTF-8"?> <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:xs="http://www.h3c.com/netconf/base:1.0" message-id="100"> <rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>protocol</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The subscribing to event time must be in the range of 1970 to 2035.</error-message> <error-info> <bad-element>stopTime</bad-element> </error-info> </rpc-error> </rpc-reply>
The value %s of attribute 'match' is invalid.	match属性指定的值不正确。	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>application</error-type> <error-tag>invalid-value</error-tag> <error-severity>error</error-severity> <error-path>/Envelope/Body/rpc/get/fi

错误提示	场景说明	错误样例
		lter/top/Device/ExtPhysicalEntities/Entity[PhysicalIndex='']/MAC</error-path> <error-message xml:lang="en">The value 50 of attribute 'match' is invalid.</error-message> </rpc-error>
The value of the ColumnValue parameter for column HostName can't be empty.	下发事件订阅报文时指定的列的值不能为空。	<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="16"> <rpc-error> <error-type>application</error-type> <error-tag>operation-not-supported</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">The value of the ColumnValue parameter for column HostName can't be empty.</error-message> </rpc-error> </rpc-reply>
Unexpected element：	报文中的元素位置错误。可能是未定义的元素，或该元素前的元素缺失等原因导致。	<rpc-error xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"> <error-type>rpc</error-type> <error-tag>unknown-element</error-tag> <error-severity>error</error-severity> <error-path>/Envelope/Body/rpc/edit-config/config/openconfig-acl:acl/acl-sets/acl-set[name='test01'][type='ACL_IPV4']/config/hello</error-path> <error-message xml:lang="en">Unexpected element 'http://openconfig.net/yang/acl':'hello' under element "/Envelope/Body/rpc/edit-config/config/openconfig-acl:acl/acl-sets/acl-set[name='test01'][type='ACL_IPV4']/config"</error-message> <error-info> <bad-element>hello</bad-element> </error-info> </rpc-error>
Incorrect column name for column %s of XPath %s.	NetconfMonitor订阅报文中ColumName值指定错误，指定XPATH下没有该列。	<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="33"> <rpc-error> <error-type>application</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">Incorrect column name for column 1 of XPath

错误提示	场景说明	错误样例
		Device/Base.</error-message> </rpc-error> </rpc-reply>
XPath %s is wrong. Valid format is ModuleName[/SubmoduleName]/TableName.	事件订阅报文中指定的XPATH错误。	<?xml version="1.0" encoding="utf-8"?> <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="34"> <rpc-error> <error-type>application</error-type> <error-tag>bad-element</error-tag> <error-severity>error</error-severity> <error-message xml:lang="en">XPath Base is wrong. Valid format is ModuleName[/SubmoduleName]/TableName. </error-message> </rpc-error> </rpc-reply>

7.2 附录 B Comware Schema内置类型说明

Comware 系统中，部分数据类型难以使用 Schema 标准方式简单描述。为了方便检查使用，这些类型的检查被作为内置实现，在外表现为 string 类型，但其逻辑检查不是执行的 string 类型。内置类型都定义在 basetype.xsd 中。Comware 支持的内置类型如表 7-4 所示。

表7-4 Schema 内置类型

名称	检查逻辑说明
OID	点分的32位整数，最多128个分节
_Ipv6Address	按照IPv6地址规范来检查
IPV6_ADDR_PREFIX	按照IPv6地址前缀规范来检查
_PortRange	用连字符（-）和逗号（,）分割的多段范围字符串，形如1,3-9,11,12。两端和中间不能有空格，不能有其他字符，数字必须从小到大排列，数字范围是0～65535
_VlanRange	连字符（-）和逗号（,）分割的多段范围字符串，形如1,3-9,11,12。两端和中间不能有空格，不能有其他字符，数字必须从小到大排列，数字范围是1～4094
_Ipv4Mask	按照IPv4的掩码逻辑来检查
_IfIndexRange	用连字符（-）和逗号（,）分割的多段范围字符串，形如1,3-9,11,12。两端和中间不能有空格，不能有其他字符，数字必须从小到大排列，数字范围是1～0xBFFFFFFF

7.3 附录 C Comware Schema的Boolean/boolean说明

Schema 中的 boolean 在不同系统中的实现有所不同。有些系统中取值为 true、false，而有些系统中除 true、false 外，还支持取值 1 和 0。在 Comware 系统中，为了保持兼容，Boolean/boolean 的有效值为 true、false、1 和 0。其中，true 和 1 都是真，false 和 0 都为假。


说明

NETCONG API 文档中，对于 boolean/Boolean 只列举了 true 和 false 取值，但其实它们也可能支持 1 和 0。

7.4 附录 D Comware NETCONF处理可能会遇到的情况建议

7.4.1 部分支持的情况

某些表中，存在某些列在部分情况下支持、部分情况下不支持的情况。此时，对于 delete/remove 操作，这些列会返回成功，以便于只下发索引时能够不返回错误。

7.4.2 IP 和掩码为索引的情况

同时存在 IP 和掩码时，为了不出现错误的结果，建议不要下发 IP 和 Mask 作用后的结果导致 IP 变化的请求。比如，不建议下发如下请求，因为 Mask 影响 IP 的结果，导致 IP 实际为 192.168.0.0。
IP：192.168.1.1　Mask：255.255.0.0

7.4.3 请求下发顺序

默认情况下，get、get-config、get-bulk、get-bulk-config、edit-config、action 中的元素可以按照任意顺序下发。但某些表存在可以重复的分组元素，此时这些表中的列必须按照顺序下发。为了保证下发请求时不出现错误，请尽量按照 NETCONF API 文档中 XML 结构里罗列的 XML 出现顺序进行下发。

举例：IRF/IRFPorts 表的 xml api 如下：

```
<IRF>
  <IRFPorts>
    <IRFPort>
      <MemberID></MemberID>
      <Port></Port>
      <Interface>
        <IfName></IfName>
        <MDCName></MDCName>
        <Mode></Mode>
      </Interface>
    </IRFPort>
  </IRFPorts>
</IRF>
```

其中，IfName、MDCName 、Mode 列均属一个同一个可重复组“Interface”，因此客户端下发报文时需要按照 xml api 规定的顺序下发：

```
<rpc message-id="100" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
      <top xmlns="http://www.h3c.com/netconf/config:1.0">
        <IRF>
          <IRFPorts xc:operation="merge">
            <IRFPort>
              <MemberID>1</MemberID>
              <Port>2</Port>
              <Interface>
                <IfName>Ten-GigabitEthernet1/0/48</IfName>
                <Mode>1</Mode>
              </Interface>
              <Interface>
                <IfName>Ten-GigabitEthernet1/0/49</IfName>
                <Mode>1</Mode>
              </Interface>
            </IRFPort>
          </IRFPorts>
        </IRF>
      </top>
    </config>
  </edit-config>
</rpc>
```