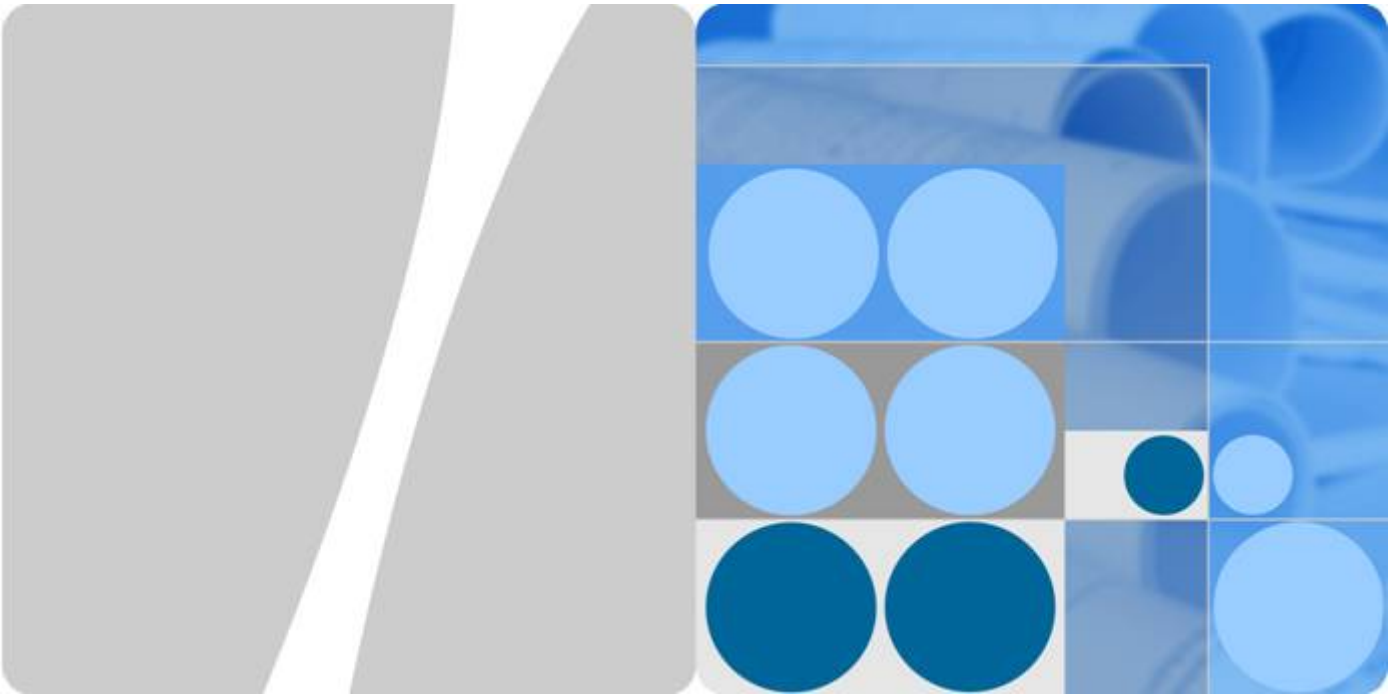


Enterprise Data Communication Products

Feature Description - Network Management

Issue 01

Date 2012-09-30

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Technologies Co., Ltd.

Address: Huawei Industrial Base
Bantian, Longgang
Shenzhen 518129
People's Republic of China

Website: http://enterprise.huawei.com

About This Document

Intended Audience

This document describes the definition, purpose, and implementation of features on enterprise datacom products including the campus network switch, enterprise router, data center switch, and WLAN. For features supported by the device, see *Configuration Guide*.

This document describes the Network Management feature in terms of its overview, principle, and applications.

This document together with other types of document helps intended readers get a deep understanding of the Network Management feature.

This document is intended for:

- Network planning engineers
- Commissioning engineers
- Data configuration engineers
- System maintenance engineers

Symbol Conventions

The symbols that may be found in this document are defined as follows.

Symbol	Description
DANGER	Indicates a hazard with a high level of risk, which if not avoided, will result in death or serious injury.
WARNING	Indicates a hazard with a medium or low level of risk, which if not avoided, could result in minor or moderate injury.
CAUTION	Indicates a potentially hazardous situation, which if not avoided, could result in equipment damage, data loss, performance degradation, or unexpected results.
TIP	Indicates a tip that may help you solve a problem or save time.

Symbol	Description
NOTE	Provides additional information to emphasize or supplement important points of the main text.

Command Conventions

The command conventions that may be found in this document are defined as follows.

Convention	Description
Boldface	The keywords of a command line are in **boldface**.
Italic	Command arguments are in *italics*.
[ ]	Items (keywords or arguments) in square brackets [ ] are optional.
{ x | y | ... }	Optional items are grouped in braces and separated by vertical bars. One item is selected.
[x | y | ...]	Optional items are grouped in brackets and separated by vertical bars. One item is selected or no item is selected.
{ x | y | ... }*	Optional items are grouped in braces and separated by vertical bars. A minimum of one item or a maximum of all items can be selected.
[x | y | ...]*	Optional items are grouped in brackets and separated by vertical bars. Several items or no item can be selected.
&<1-n>	The parameter before the & sign can be repeated 1 to n times.
#	A line starting with the # sign is comments.

Change History

Updates between document issues are cumulative. Therefore, the latest document issue contains all updates made in previous issues.

Changes in Issue 01 (2012-09-30)

Initial commercial release.

Contents

1 SNMP

About This Chapter

1.1 Introduction to SNMP

Definition

The Simple Network Management Protocol (SNMP) is a network management protocol widely used in the TCP/IP network. SNMP is a method of managing network elements through a network console workstation which runs network management software. SNMP has the following features:

- Simplicity: SNMP applies to small-scale networks requiring high speed and low costs because it uses a polling mechanism and provides basic functions. SNMP uses UDP packets, and is therefore supported by most devices.
- Powerfulness: SNMP ensures the transmission of management information between any two devices on the network, thereby allowing the network administrator to query information, modify parameters, and locate faults on any device.

Purpose

As networks rapidly develop and applications become more diversified, network management becomes difficult due to the following factors:

- The number of network devices is dramatically increasing, which increases the network administrator's workload. In addition, networks' coverage areas are constantly being expanded, making real-time monitoring and fault location of network devices difficult.
- Various devices exist on the network, and management interfaces provided by different vendors differ from each other. This makes the network management complex.

To address this problem, SNMP was developed. SNMP supports efficient batch management on network devices and filters differences between products. SNMP allows unified management regardless of the device type and vendor.

Version Evolution

In May 1990, RFC 1157 was developed to define the first SNMP version: SNMPv1. RFC 1157 provides a systematic method for monitoring and managing the network. SNMPv1 cannot ensure the security of the network because it is based on community-name authentication, and only a few error codes are returned.

Later, Internet Engineering Task Force (IETF) released SNMPv2c. SNMPv2c introduced GetBulk and Inform and supported more standard error codes and data types (including the Counter64 and Counter32)

Because SNMPv2c did not provide a high level of security, the IETF released SNMPv3. SNMPv3 provides user security module-based (USM-based) encrypted authentication and view-based access control model (VACM).

The SNMP versions including SNMPv1, SNMPv2c, and SNMPv3 are widely used.

Benefits

- Improves the work efficiency of the network administrator. The network administrator can use SNMP to query information, modify information, and locate faults on any device.

- Reduces management costs. SNMP provides basic functions for managing devices with different management tasks, physical attributes, and network types.
- Reduces the impact of feature operations on the device. SNMP is simple in terms of hardware/software installation, packet type, and packet format.

1.2 Principles

1.2.1 SNMP Management Model

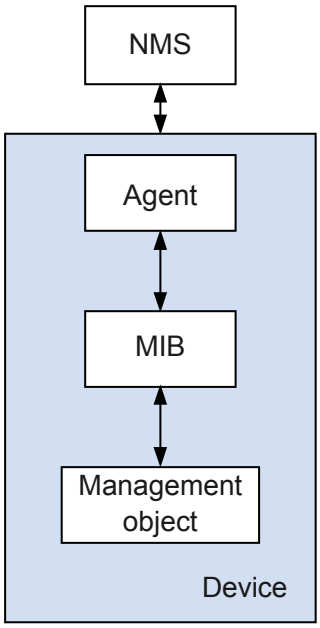
The SNMP system is composed of the NMS, agent, management object, and MIB.

The NMS is the network management center of the network and manages devices on the network.

Each managed device has the agent process, MIB, and multiple managed objects. The NMS interacts with the agent on the managed device. The agent performs operations on the MIB to perform the NMS request.

Figure 1-1 shows an SNMP management model.

Figure 1-1 SNMP management model



Elements in the network management system are as follows:

- **NMS**

  A manager on the network, or a system using SNMP to manage and monitor network devices. The NMS runs on NMS servers.

  - An NMS can send requests to an agent on a device to query or modify the value of one or multiple parameters.
  - An NMS can receive traps sent from the agent on a device to learn the current status of the device.

- **Agent**

  Agent is a process on the managed device. The agent maintains data on the managed device, receives and processes the request packets from the NMS, and then sends the response packets to the NMS.

  - Upon receiving requests of the NMS, the agent performs the required operation over the MIB and sends the operation result to the NMS.
  - When a fault or an event occurs on the device, the agent running on the device sends notifications to the NMS, reporting the current status of the device.

- **Management object**

  Object to be managed. A device may have multiple management objects, including a hardware component (such as an interface board), software, and parameters (such as a route selection protocol) configured for the hardware or software.

- **MIB**

  MIB is a database specifying variables that are maintained by the managed device and can be queried or set by the agent. MIB defines attributes of the managed device, including the name, status, access rights, and data type of objects.

  An agent can use the MIB to:

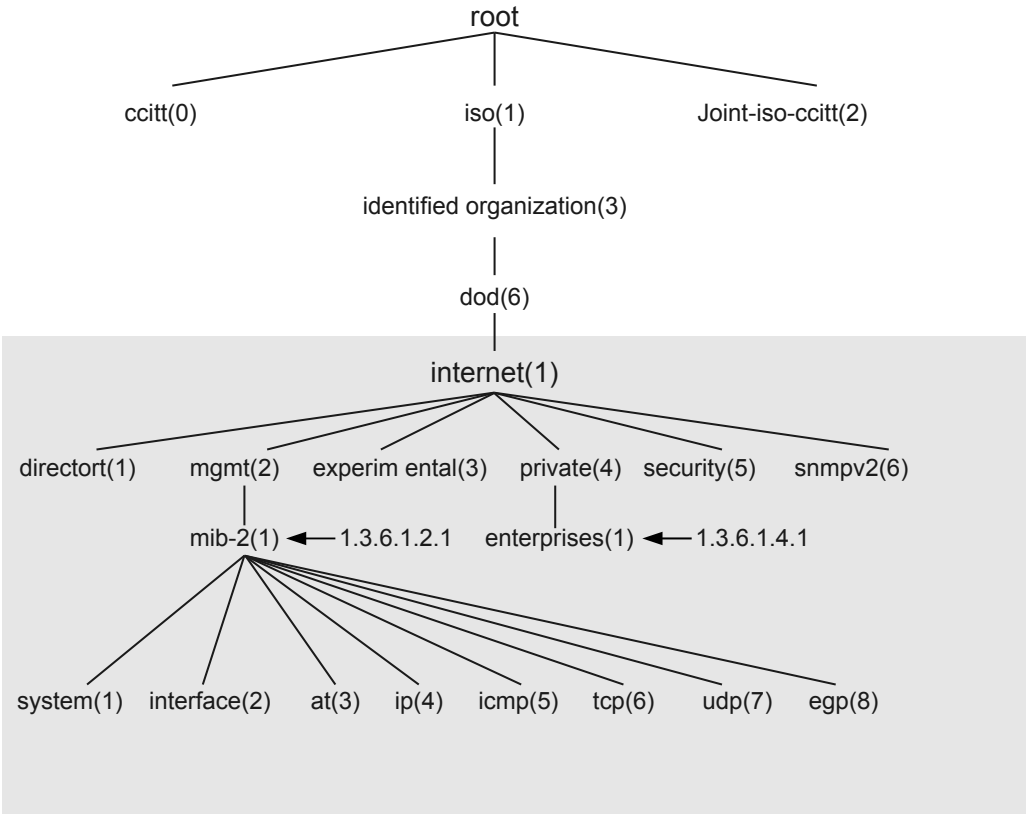
  - Learn the current status of the device.
  - Set the status parameter of the device.

  The SNMP MIB adopts a tree structure like the Domain Name System (DNS) with its root on the top without a name. **Figure 1-2** shows a part of the MIB, called object naming tree. Each object identifier (OID) maps a managed object, for example, the system OID is 1.3.6.1.2.1.1, and the interface OID is 1.3.6.1.2.1.2.

  The OID tree facilitates information management and improves management efficiency. With the OID tree, the network administrator can query information in batches.

  When configuring the agent, the user can configure the MIB object access control for the NMS based on the MIB view. A MIB view is a subset of a MIB.

Figure 1-2 OID tree

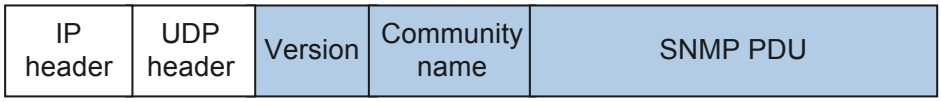


1.2.2 SNMPv1/SNMPv2c

SNMPv1/SNMPv2c Packet Format

As shown in **Figure 1-3**, an SNMPv1 packet is composed of the version, community name, and SNMP Protocol Date Unit (PDU) fields.

Figure 1-3 SNMPv1/SNMPv2c packet format



The fields in an SNMPv1/SNMPv2c packet are defined as follows:

- Version: SNMP version. The SNMPv1 packet field is 0, and the SNMPv2c packet field is 1.
- Community name: used for authenticating operations between the agent and NMS. The community name is a string of characters and can be defined by users. The community name can be a read-only or write-only community name. To authenticate the GetRequest

or GetNextRequest operations, use the read-only community name; to authenticate the Set operation, use the write-only community name.

- SNMPv1/SNMPv2c PDU: includes the PDU type, request ID, and binding variable list. The SNMPv1 PDU includes GetRequest PDU, GetNextRequest PDU, SetRequest PDU, Response PDU, and Trap PDU. The SNMPv2c PDU inherits the SNMPv1 PDU and introduces the GetBulkRequest PDU and InformRequest PDU.

  For simplification, the SNMP operations are described as the Get, GetNext, Set, Response, Trap, GetBulk, and Inform operations.

SNMPv1/SNMPv2c Operations

As shown in **Table 1-1**, SNMPv1/SNMPv2c defines seven types of operations for exchanging information between the NMS and the agent.

Table 1-1 SNMPv1/SNMPv2c Operations

Operation	Description
Get	The management process reads one or several parameter values from the MIB of the agent process.
GetNext	The management process reads the next parameter value from the MIB of the agent process.
Set	The management process sets the parameter value of one or more MIBs of the agent process.
Response	The agent process returns one or more queried values. The agent performs this operation that corresponds to the GetRequest, GetNextRequest, SetRequest, and GetBulkRequest operations. Upon receiving a Get or Set request, the agent performs the Query or Modify operation using MIB tables and then sends the responses to the NMS.
Trap	The agent process notifies the NMS of a fault or event on the managed device.
GetBulk	The NMS queries managed devices in batches.
Inform	The managed device notifies the NMS of an alarm on a managed device. After the managed device sends an inform, the NMS must send an InformResponse packet to the managed device.

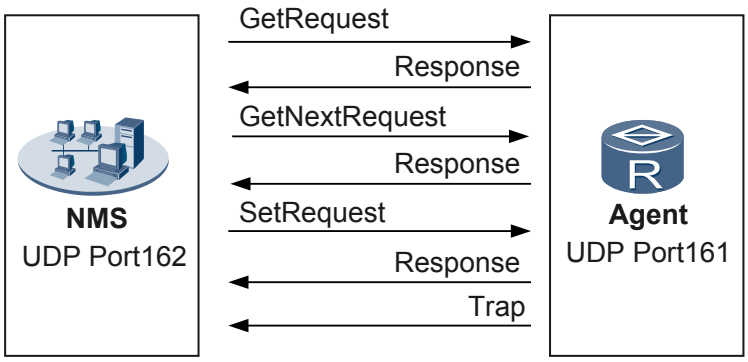
NOTE

SNMPv1 does not support the GetBulk or Inform operations.

Working Mechanisms of SNMPv1/SNMPv2c

The working mechanisms of SNMPv1 and SNMPv2c are similar, as shown in **Figure 1-4**.

Figure 1-4 Basic operations



- Get

  The following assumes that the NMS wants to use the read-only community name **public** to obtain the value of the object sysContact on the managed devices. The procedure is as follows:

  1. NMS: sends a GetRequest packet to the agent. The fields of the packet are set as follows: The version is the SNMP version in use; the community name is **public**; the PDU type is Get; the MIB object is sysContact.
  2. Agent: authenticates the version and community name of the packet. When authentication succeeds, the agent encapsulates the queried sysContact value into the PDU of the response packet. Then the agent sends the response packet to the NMS. If the agent fails to obtain the sysContact value, the agent will send an incorrect response packet to the NMS.

- GetNext

  The following assumes that the NMS wants to use the community name **public** to obtain the value of the object sysName (object next to sysContact) on the managed device. The procedure is as follows:

  1. NMS: sends a GetNext request packet to the agent. The fields of the packet are set as follows: The version is the SNMP version in use; the community name is **public**; the PDU type is GetNext; the MIB object is sysContact.
  2. Agent: authenticates the version and community name of the packet. When authentication succeeds, the agent encapsulates the queried sysName value into the PDU of the response packet. Then the agent sends the response packet to the NMS. If the agent fails to obtain the sysName value, the agent will send an incorrect response packet to the NMS.

- Set

  The following assumes that the NMS wants to use the read-only community name **private** to set the value of the object sysName on the managed device to **HUAWEI**. The procedure is as follows:

  1. NMS: sends a SetRequest packet to the agent. The fields of the packet are set as follows: The version is the SNMP version in use; the community name is **private**; the PDU type is Set; the MIB object is sysContact; the target value is **HUAWEI**.
  2. Agent: authenticates the version and community name of the packet. When authentication succeeds, the agent sets an object mapping the requested management variable. If the setting succeeds, the agent sends a response packet to the NMS. If the setting fails, the agent will send an incorrect response packet to the NMS.

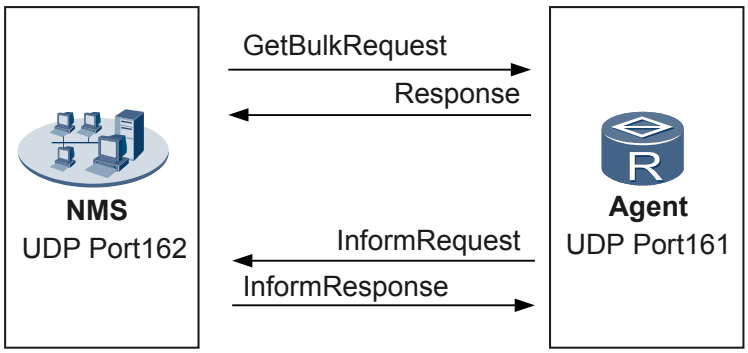
- Trap

  Trap is a spontaneous behavior of a managed device. Traps do not belong to the basic operations performed by the NMS on the managed device. If a managed device meets the triggering condition for generating a trap, the agent notifies the NMS of the exception by sending a trap. For example, when a managed device is started in hot startup mode, the agent sends a warmStart trap to the NMS.

  The agent sends the trap to the management process only when a module on the device meets the triggering condition for generating a trap. This method reduces exchange traffic by sending traps only when major events occur.

Figure 1-5 shows the operations that are added in SNMPv2c.

Figure 1-5 New operations in SNMPv2c



- GetBulk

  The GetBulk operation is equal to consecutively performed GetNext operations. You can set the number of times that the GetNext operations are performed during one GetBulk operation.

- Inform

  A managed device notifies the NMS of an inform. After the managed device sends an inform, the NMS must send an InformResponse packet to the managed device. If the managed device does not receive the response packet, the managed device performs the following operations:

  1. Save the alarm in the inform buffer.
  2. Repeatedly send the alarm until the NMS returns the response packet or the number of times that the managed device sends alarms exceeds the allowed range.
  3. An alarm log is generated on the managed device.

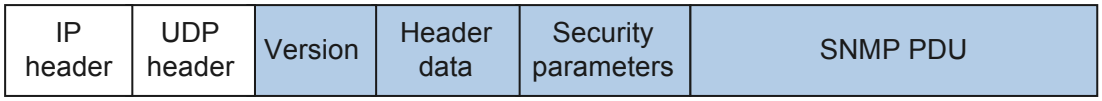
  Therefore, the informs may occupy many system resources.

1.2.3 SNMPv3

SNMPv3 Packet Format

SNMPv3 defines a new packet format shown in **Figure 1-6**.

Figure 1-6 SNMPv3 packet format



The following describes the composition of an SNMPv3 packet:

- Version: SNMP version. The SNMPv3 packet field is 2.
- Header: information such as the maximum message size supported by the transmitter, and security mode of messages.
- Security parameters: security information including the entity engine information, user name, authentication parameter, and encryption information.
- SNMPv3 PDU: includes the following information:
  - Context EgineID: SNMP ID. Together with the PDU type, it determines which application messages are to be sent.
  - Context Name: determines the Context EgineID MIB view of the managed device.
  - PDU data: includes the PDU type, request ID, and binding variable list. The SNMPv3 PDU includes GetRequest PDU, GetNextRequest PDU, SetRequest PDU, Response PDU, Trap PDU, GetBulkRequest PDU, and InformRequest PDU.

SNMPv3 Architecture

SNMPv3 uses the SNMPv3 entity for the communication between different SNMP-enabled NMSs. An SNMPv3 entity consists of SNMPv3 engines and applications, and each SNMPv3 engine or application has multiple modules.

The modular architecture of the SNMP entity has the following advantages:

- Strong adaptability: This architecture is adaptable for both simple and complex networks.
- Easy management: This architecture consists of multiple independent sub-systems and applications. When a fault occurs in the system. it is easy to locate the sub-system to which the fault belongs based on the fault type.
- Excellent expandability: An SNMP system can be extended by increasing the number of modules on the SNMP entity. For example, a module can be added in the security sub-system for the application of a new security protocol.

SNMPv3 improves security by adopting the user security model (USM) and view-based access control model (VACM).

- USM: authenticates user identity and encrypts data. These two functions require that the NMS and the agent use a shared key.
  - Identify authentication: a process in which the agent (or the NMS) confirms whether the received message is from an authorized NMS (or agent) and whether the message is changed during transmission. RFC 2104 defines Keyed-Hashing for Message Authentication Code (HMAC), an effective tool that uses the security hash function and key to generate the message authentication code. This tool is widely used in the Internet. HMAC used in SNMP contains HWAC-MD5-96 and HWAC-SHA-96. The hash function of HWAC-MD5-96 is MD5 that uses 128-bit authKey to generate the key. The hash function of HWAC-SHA-96 is SHA-1 that uses 160-bit authKey to generate the key.
  - Data encryption: uses the cipher block chaining (CBC) code of the data encryption standard (DES) and uses 128-bit privKey to generate the key. The network management station uses the key to calculate the CBC code and then adds the CBC code to the message while the agent fetches the authentication code through the same key and then obtains the actual information. Like identity authentication, data encryption also

requires the network management station and the agent to use a shared key for encryption or decryption.

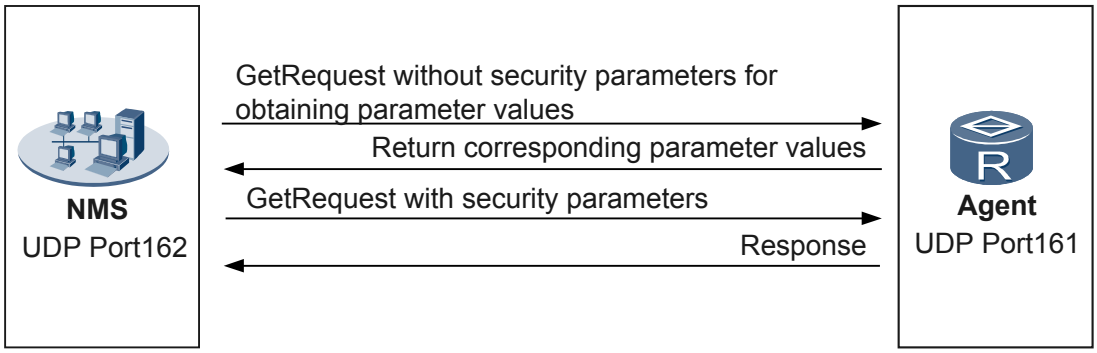
- VACM: controls access of user groups or community names based on the view. You must pre-configure a view and specify its authority. Then, when you configure a user, user group, or community, load this view to implement read/write restriction or trap function.

SNMPv3 Mechanism

The mechanism of SNMPv3 is similar to those of SNMPv1 and SNMPv2, but SNMPv3 supports identity authentication and encryption. The following describes the SNMPv3 mechanism by using the Get operation as an example.

The following assumes that the NMS wants to obtain the value of the object sysContact on the managed device in authentication and encryption mode, as shown in **Figure 1-7**.

Figure 1-7 Get operation of SNMPv3



1. NMS: sends a GetRequest packet without security parameters to the agent and requests the values of Context EgineID, Context Name, and security parameter.
2. Agent: responds to the request from the NMS by providing the requested parameters.
3. NMS: sends a GetRequest packet to the agent. The packet fields are set as follows:
   - Version: SNMPv3.
   - Header: specifies authentication and encryption.
   - Security parameters: The NMS calculates the authentication and encryption parameters using the configured algorithm. These parameters and security parameters are filled in the corresponding fields.
   - PDU: Set corresponding fields using obtained Context EgineID and Context Name. The PDU type is set to Get, the MIB object is sysContact, and the configured encryption algorithm is used to encrypt the PDU.
4. Agent: authenticates the messages. When authentication succeeds, the agent decrypts the PDU. When encryption succeeds, the agent obtains the value of sysContact and encapsulates it to the PDU in the response packet. The agent encrypts the PDU and sends the response packet to the NMS. If the query, authentication, or encryption fails, the agent will send an incorrect response packet to the NMS.

1.2.4 Comparison Among SNMP Versions

Table 1-2 Comparison in the security of SNMP of different versions

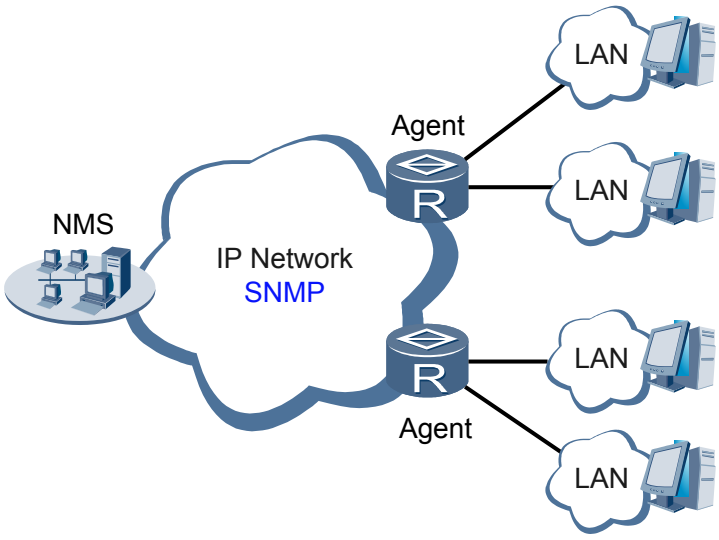
Protocol version	Security Level	Authentication Mode	Encryption Mode
SNMPv1	No authentication and no encryption	Community name	None
SNMPv2c	No authentication and no encryption	Community name	None
SNMPv3	No authentication and no encryption	User name	None
	Authentication and no encryption	MD5 or SHA	None
	Authentication and encryption	MD5 or SHA	AES128 or DES56

1.3 Applications

SNMP Application

The network administrator needs to configure and manage all devices on the network. On the network with sparsely-located devices as shown in **Figure 1-8**, it is impossible for the network administrator to configure and manage each device on site. Various devices exist on the network, and management interfaces provided by different vendors differ from each other. This makes the network management complex. To reduce the operation cost and improve the work efficiency, the network administrator can use SNMP to manage, configure, and monitor network devices remotely.

Figure 1-8 SNMP application



To configure SNMP for the network, configure the NMS on the management end and agent on the managed device.

With SNMP:

- The NMS can learn the device status by sending requests to the agent and control devices remotely.
- The agent can report the status and faults of the device to the NMS in real time.

1.4 Reference Standards and Protocols

The following table lists the references of this document.

Document	Description	Remarks
RFC 1155	Structure and identification of management information for TCP/IP-based Internets	-
RFC 1157	Simple Network Management Protocol (SNMP)	-
RFC 1212	Concise MIB definitions	-
RFC 1215	A Convention for Defining Traps for use with SNMP	-
RFC 1448	Version 2 of the Protocol Operations for SNMP	-
RFC 1901	Introduction to Community-based SNMPv2	-
RFC 1907	Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2)	-
RFC 2271	An Architecture for Describing SNMP Management Frameworks	-
RFC 2570	Introduction to Version 3 of the Internet-standard Network Management Framework	-
RFC 2578	Structure of Management Information Version 2 (SMIv2)	-
RFC 2579	Textual Conventions for SMIv2	-
RFC 2580	Conformance Statements for SMIv2	-
RFC 3410	Introduction and Applicability Statements for Internet-Standard Management Framework	-
RFC 3411	An Architecture for Describing SNMP Management Frameworks	-
RFC 3413	SNMP Applications	-
RFC 3416	Version 2 of the Protocol Operations for SNMP	-
RFC 3417	Transport Mappings for SNMP	-
RFC 3418	Management Information Base (MIB) for SNMP	-

Document	Description	Remarks
RFC 3512	Configuring Networks and Devices with SNMP	-

2 NETCONF

About This Chapter

2.1 Introduction to NETCONF

Definition

The Network Configuration Protocol (NETCONF) provides a mechanism to install, maintain, and delete configurations of network devices. You can use NETCONF to obtain configurations and status of the network devices. NETCONF enables a network device to provide standard application programming interfaces (APIs) through which applications can send configuration data to and obtain configuration data from the network device.

Purpose

As network scale and complexity increase, the traditional Simple Network Management Protocol (SNMP) has lagged behind demands for easy management (especially configuration management) of complex networks. The Extensible Markup Language (XML) based NETCONF has been introduced to provide the required configuration management.

NETCONF can be implemented by invoking the existing functions of a device. This reduces implementation costs and allows easy access to new features. NETCONF allows a client to discover extended functions supported by a server and to adjust its behavior to use the functions provided by the device.

Table 2-1 compares NETCONF and SNMP.

Table 2-1 Comparison between NETCONF and SNMP

Feature	SNMP	NETCONF
Configuration management	SNMP does not provide a protection lock mechanism to prevent multiple users from performing the same configuration.	NETCONF provides a protection lock mechanism to prevent configuration conflicts.
Query	Querying one or more records in a table requires multiple interaction processes.	NETCONF can directly query any configuration data on the system and filter the configuration data to query.
Extensibility	SNMP provides low scalability.	NETCONF offers good scalability. ● NETCONF users a multi-layer model, in which each layer is independent of other layers. The extension of one layer has little effect on other layers. ● NETCONF uses the XML encoding format to expand the protocol's management capability and system compatibility.

Feature	SNMP	NETCONF
Security	The latest version SNMPv3 uses self-defined security parameters, which prevent future expansion.	NETCONF uses existing security protocols to guarantee security and is not bound to any specific security protocol. NETCONF is more flexible than SNMP in ensuring security. **NOTE** NETCONF prefers the Secure Shell (SSH) as the transport layer protocol to transmit XML messages.

2.2 Principles

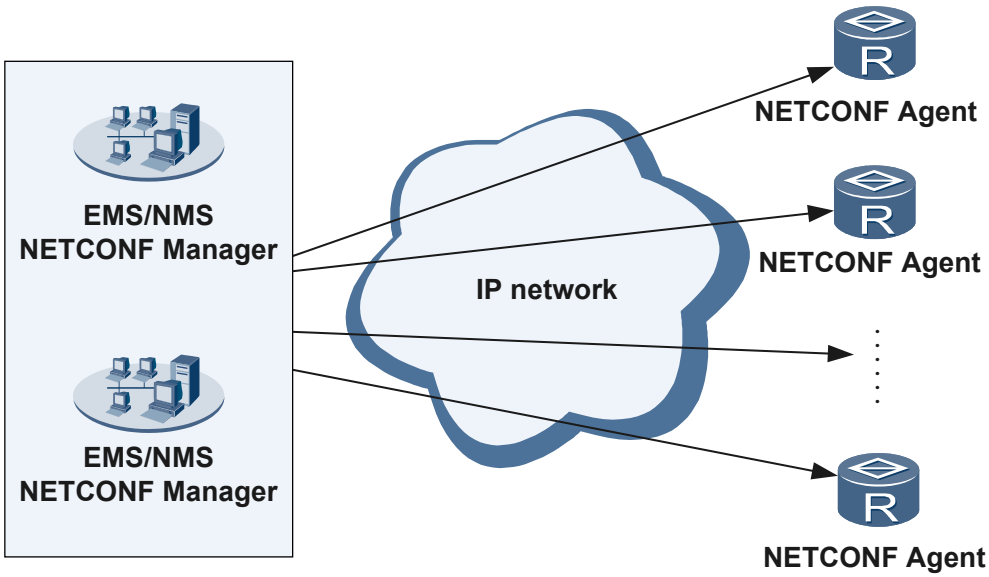
2.2.1 Basic Concepts

NETCONF Components

NETCONF provides a standard framework and a set of standard Remote Procedure Call (RPC) methods, through which network administrators and application developers can manage configurations of network devices. The device configuration data and the NETCONF protocol are encoded with the Extensible Markup Language (XML).

NETCONF uses a client/server architecture, as shown in **Figure 2-1**.

Figure 2-1 NETCONF client/server architecture



The NETCONF architecture involves two roles: NETCONF manager and NETCONF agent.

- **NETCONF manager**: functions as a client. It runs on NMS/EMS and interacts with the NETCONF agent to manage devices. The network administrator uses the NETCONF

manager (NMS/EMS) to send <rpc> requests encoded in XML format to the NETCONF agent.

- **NETCONF Agent**: functions as a server on the network. To manage device configurations, the NETCONF manager sends a request to the NETCONF agent. The NETCONF agent decodes the request and manages the configuration of the device with the help of Configuration Management Framework (CMF) and sends a response to the NETCONF manager in XML format.

NETCONF Structure

NETCONF can be conceptually partitioned into four layers, as described in **Table 2-2**.

Table 2-2 NETCONF structure

Layer	Example	Description
Layer 1: transport protocols	BEEP, SSH, SSL	The transport layer provides a communication path for interaction between the NETCONF manager and NETCONF agent. NETCONF can be layered over any transport protocol that meets the following basic requirements: ● The transport protocol is connection-oriented. The NETCONF manager and NETCONF agent must establish a persistent connection. This connection must provide reliable, sequenced data transmission. ● The transport layer provides user authentication, data integrity, and confidentiality for NETCONF. ● The transport protocol provides a mechanism to distinguish the session type (client or server) for NETCONF. **NOTE** Currently, the device only supports SSH as the transport layer protocol of NETCONF.
Layer 2: RPC	<rpc>, <rpc-reply>	The RPC layer provides a simple RPC request and reply mechanism independent of transport protocols. The client uses the <rpc> element to encapsulate RPC request information and sends the RPC request information to the server using a secure, connection-oriented session. The server uses the <rpc-reply> element to encapsulate RPC response information (content at the operation and content layers) and sends the RPC response information to the client. Normally, the <rpc-reply> element encapsulates data required by the client or a configuration success message. If the client sends an incorrect request or the server fails to process a request from the client, the server encapsulates the <rpc-error> element containing detailed error information in the <rpc-reply> element and sends the <rpc-reply> element to the client.

Layer	Example	Description
Layer 3: operations	<get-config>, <edit-config>, <notification>	The operation layer defines a series of operations used in RPC. These operations compose the basic capabilities of NETCONF.
Layer 4: content	Configuration data	The content layer describes configuration data required for network management. The content layer is the only layer that is not standardized. It has no standard modeling language or data model. Therefore, configuration data varies according to devices from different vendors.

2.2.2 NETCONF Transport Layer

The transport layer provides a reliable mechanism to send data based on the sequence number. The mechanism provides such capabilities as user authentication, data integrity, and confidentiality.

NETCONF is connection-oriented and must provide reliable and sequenced data transmission. NETCONF does not verify, check integrity of, or acknowledge transmitted data. Therefore, NETCONF requires SSH as its transport layer protocol.

After the NETCONF agent process is started, the NETCONF agent can receive connection requests sent from the NETCONF manager and process corresponding configurations. User authentication is performed during connection setup. The process is as follows:

1. The NETCONF agent process is started.
2. The NETCONF agent creates a listening port and waits for connection requests from the NETCONF manager (SSH client).
3. The SSH client sends a connection request to the NETCONF agent.
4. The NETCONF agent and SSH client start data transmission and exchange complete information and encryption keys. The NETCONF agent authenticates the client using the **ssh-userauth** service.
5. After the client is authenticated, the NETCONF agent sends the authentication information to the user management module to establish a session with the SSH client. The client starts the **ssh-connection** service.

A connection at the transport layer of NETCONF has been set up. The NETCONF manager can manage the NETCONF agent through this session.

After a NETCONF connection is established, the client and the server exchange NETCONF messages. The NETCONF agent sends a Hello message that carries its capabilities and identifiers. The client also sends a Hello message that carries its capabilities. After receiving the Hello message sent from the client, the NETCONF agent identifies the capabilities that will be used and ignores unnecessary and unidentified capabilities.

NOTE

For details about NETCONF capabilities, see **2.2.4 NETCONF Operation Layer**.

The following is a sample of a Hello message sent by the NETCONF manager.

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

```
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:writable-running:1.0</
capability>
    <capability>urn:ietf:params:netconf:capability:candidate:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:validate:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:startup:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:rollback-on-error:1.0</
capability>
    <capability>urn:huawei:netconf:capability:sync:1.0capability>
    <capability>http://www.huawei.com/netconf/capability/sync/1.0</capability>
    <capability>urn:ietf:params:netconf:capability:confirmed-commit:1.0</
capability>
    <capability>http://www.huawei.com/netconf/capability/discard-commit/1.0</
capability>
    <capability>http://www.huawei.com/netconf/capability/rollback/1.0</capability>
    <capability>http://www.huawei.com/netconf/capability/exchange/1.0</capability>
    <capability>http://www.huawei.com/netconf/capability/action/1.0</capability>
    <capability>http://www.huawei.com/netconf/capability/update/1.0</capability>
  </capabilities>
</hello>
]]>]]>
```

The following is a sample of a Hello message sent by the NETCONF agent.

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

   <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0/capability>
    <capability>urn:ietf:params:netconf:capability:writable-running:1.0</
capability>
    <capability>urn:ietf:params:netconf:capability:candidate:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:startup:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:rollback-on-error:1.0</
capability>
    <capability>http://www.huawei.com/netconf/capability/sync/1.0</capability>
    <capability>http://www.huawei.com/netconf/capability/exchange/1.0</capability>
    <capability>http://www.huawei.com/netconf/capability/active/1.0</capability>
    <capability>http://www.huawei.com/netconf/capability/action/1.0</capability>
    <capability>http://www.huawei.com/netconf/capability/update/1.0</capability>
   </capabilities>
   <session-id>1149</session-id>
</hello>
```

The following are the samples of invalid Hello messages sent by the NETCONF manager:

- Hello messages without base capabilities

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

- Incorrect Hello messages

  ```
  <?xml version="1.0">
  <capabilities>
  <capabilities>urn:ietf:params:netconf:base:1.0</capability>
  <capabilities>urn:ietf:params:netconf:capability:candidate:1.0</capability>
  </capabilities>
  </hello>
  ```

2.2.3 NETCONF RPC Layer

A program can use the Remote Procedure Call (RPC) protocol to request services from a remote computer program over a network without knowing the underlying network technologies. The RPC protocol assumes the existence of a transmission protocol such as Transmission Control

Protocol (TCP) or User Datagram Protocol (UDP) to transmit data exchanged between communicating programs. In the open systems interconnection (OSI) reference model, RPC traverses the transport layer and application layer. RPC simplifies development of applications such as a network distributed multi-program.

NETCONF uses a RPC-based communication mechanism. The NETCONF manager and NETCONF agent use <rpc> and <rpc-reply> elements to provide framing of NETCONF requests and responses independent of transport layer protocols, implementing device configuration and management. After the NETCONF manager and NETCONF agent exchange hello messages, the NETCONF manager sends <rpc> request messages to the NETCONF agent to configure and manage the NETCONF agent. The NETCONF agent sends an <rpc-reply> message in response to each request message.

<rpc> Element

The <rpc> element encapsulates a request message that the NETCONF manager sends to the NETCONF agent. The format of an RPC request message is as follows:

```
<rpc message-id="101"
     xmlns="urn:idtf:params:xml:ns:netconf:base:1.0">
    <some-method>
        <!--method parameters here  -->
    </some-method>
</rpc>
```

<rpc-reply> Element

The <rpc-reply> element encapsulates a response message for an RPC request message. The NETCONF agent must return an <rpc-reply> message containing every attribute in the <rpc> request message to the NETCONF manager. For example, the <rpc> element invokes the NETCONF <get> operation to obtain the user ID. The <rpc> request message contains the user-id attribute, as follows:

```
<rpc message-id="101"
     xmlns="urn:idtf:params:xml:ns:netconf:base:1.0">
     xmlns:ex="http://example.net/content/1.0"
     ex:user-id="fred"
    <get/>
</rpc>
```

The <rpc-reply> message contains the value of user-id, as follows:

```
<rpc-reply message-id="101"
     xmlns="urn:idtf:params:xml:ns:netconf:base:1.0">
     xmlns:ex="http://example.net/content/1.0"
     ex:user-id="fred"
    <data>
       <!--contents here  -->
    <data>
</rpc-reply>
```

If an error occurs during RPC request processing, the NETCONF agent returns an <rpc-reply> message containing the <rpc-error> element and other elements to notify the NETCONF manager of the failure cause.

2.2.4 NETCONF Operation Layer

The operation layer is the core of NETCONF, and its function is similar to SNMP primitives. The NETCONF operation layer defines an underlying operation set used to configure and obtain device information saved in a database. The defined operations allow you to obtain, configure,

copy, and delete device information in the database. The basic operations defined by NETCONF must be a minimal set of functions.

NOTE

By default, the NETCONF agent supports all basic operations defined by NETCONF.

For details about the NETCONF database, see **2.2.5 NETCONF Content Layer**.

When the NETCONF manager and NETCONF agent establish an NETCONF session, they send the base capability of NETCONF urn:ietf:params:netconf:base:1.0 to the peer.

Table 2-3 describes basic operations defined in the base capability.

Table 2-3 Basic operations defined in the base capability of NETCONF

Basic Operation	Description
<get-config>	Obtains all or specified configuration data from the <running/>, <candidate/>, and <startup/> configuration databases.
<get>	Obtains some or all running configuration data and status data from the <running/> configuration database.
<edit-config>	Creates, modifies, or deletes configuration data.
<copy-config>	Replaces the target configuration database with the source configuration database. If no target configuration database has been created, this operation creates a configuration database. If a target configuration database has been created, the source configuration database replaces the target configuration database.
<delete-config>	Deletes a configuration database. The <running/> configuration database cannot be deleted.
<lock>	Locks a configuration database. A locked configuration database cannot be modified by other users. Locking a configuration database ensures that the configuration of a database is not affected by the configuration of the NETCONF manager, SNMP, or command-line interface (CLI) script, preventing a conflict.
<unlock>	Unlocks a locked configuration database. Users can unlock only the configuration databases they have locked.
<close-session>	Closes an NETCONF session.
<kill-session>	Forcibly closes an NETCONF session. Only an administrator can perform this operation.

In addition to the basic functionality, NETCONF also allows the NETCONF manager to discover the extended protocol set supported by the NETCONF agent. The feature is called capabilities. NETCONF uses capabilities to define additional functions except the basic functionality. The capabilities enhance the NETCONF functionality and strengthen the fault tolerance and scalability. This facilitates the implementation of the NETCONF-based open network management architecture and provides an efficient method for equipment manufacturers to comply with NETCONF and expand device functionality.

Table 2-4 describes the standard capabilities defined by NETCONF and operations defined in the standard capabilities.

Table 2-4 Standard capabilities defined by NETCONF and operations defined in the standard capabilities

Standard Capability	Description	Operation Defined in the Standard Capability	Dependency
writable-running	This capability allows a device to access the <running/> configuration database. That is, the device supports <edit-config> and <copy-config> operations for running configuration data.	-	-
Candidate configuration	This capability indicates that a device supports the <candidate/> configuration database. This capability allows the device to perform operations on configuration data without affecting the configuration data in use.	Two operations are added: ● <commit>: converts all configuration data in the <candidate/> configuration database into running configuration data. ● <discard-changes>: discards uncommitted configuration data from the <candidate/> configuration database. After this operation is performed, the configuration data in the <candidate/> configuration database remains the same as the data in the <running/> configuration database.	-
Rollback on error	This capability allows a device to perform rollback when an error occurs. If an error occurs and the <rpc-error> element is generated, the server stops performing the <edit-config> operation and restores the specified configuration to the status before the <edit-config> operation is performed.	-	-

Standard Capability	Description	Operation Defined in the Standard Capability	Dependency
Distinct startup	This capability allows a device to perform a distinct startup. The NETCONF agent checks parameter availability and consistency. **NOTE** This capability indicates that the NETCONF agent supports the <startup/> configuration database and can distinguish the <running/> configuration database from the <startup/> configuration database. To permanently save configuration data in the <running/> configuration database, the device performs a <copy-config> operation to copy the configuration data from the <running/> configuration database to the <startup/> configuration database.	-	-

In addition to the base capability, Huawei defines proprietary capabilities and operations in the proprietary capabilities, as described in **Table 2-5**.

Table 2-5 Proprietary capabilities defined by Huawei and operations defined in the proprietary capabilities

Proprietary Capability	Description	Operation Defined in the Proprietary Capability	Dependency
Sync	This capability allows a device to perform data synchronization. The NETCONF manager sends a request to the NETCONF agent to update the local data set of the NETCONF manager. A file transfer protocol is used to synchronize NETCONF agent data to a destination folder.	● <sync-full>: synchronizes the device configuration file to a target server in compressed mode. ● <sync-inc>: synchronizes the device configuration file between two check points.	-

Proprietary Capability	Description	Operation Defined in the Proprietary Capability	Dependency
Active notification	This capability allows a device to notify the peer that it is active.	<active>: When the NETCONF agent is performing an operation that takes a long time, it periodically sends <active> packets to notify the NETCONF manager that it is active and performing an operation.	-
Action	This capability allows a device to run commands.	<execute-action>: requests the NETCONF agent to run a maintenance command (excluding basic configuration and query commands).	-
Update	This capability allows a device to update configuration data.	<update>: updates configuration data in the <candidate/> configuration database to the <running/> configuration database when a conflict occurs during data commitment.	-
Exchange	This capability allows a device to exchange information with a peer. If an NETCONF session has the exchange capability, it supports the <get-next> operation.	-	-

2.2.5 NETCONF Content Layer

The content layer describes configuration data required for network management. The configuration data varies according to devices from different vendors. The content layer is the only layer that is not standardized. It has no standard modeling language or data model.

NETCONF defines three standard conceptual configuration databases.

- <**running/**>

  This configuration database stores various configuration parameters that are running on a device. It is the only standard database that is mandatory if the NETCONF agent does not support the **:candidate** capability. The NETCONF agent must be allowed to edit the database.

  The <running/> database also stores all the conceptual status information currently available on the device. A <get> operation can be performed on the <running/> database

to obtain the status information, statistics, and configuration parameters. The <get-config> operation obtains only the configuration data.

- <**candidate**/>

  The NETCONF supports the **:candidate** capability to perform operations on the <candidate/> database. Any change to the <candidate/> database does not directly affect the network device. The administrator performs a <commit> operation to activate the changes made to the <candidate/> database and make them a part of <running/> database. After a successful <commit> operation, the <candidate/> database and <running/> database have the same content. A <lock> operation can be performed only on the <running/> database.

  The <candidate/> database is a global database, and the administrator can perform a <discard-changes> operation to remove the modified configurations that are not required.

- <**startup**/>

  The NETCONF supports the **:startup** capability to perform corresponding operations on the <startup/> database. The NETCONF agent allows an administrator to define a separate data set and perform a distinct startup for a device. If this capability takes effect, the NETCONF agent does not save changes of the <running/> database to the data set. Instead, the NETCONF agent performs a <copy-config> operation to overwrite content in the <startup/> database with current configurations. If this capability does not take effect, the NETCONF agent updates the data set when the running configuration is modified. In either of the preceding situations, the NETCONF agent needs to maintain the configurations of the data set and can restore the configuration after the device restarts.

2.3 References and Protocols

The following table lists the references for this document.

Document	Description	Remarks
RFC 4741	NETCONF Configuration Protocol	-
RFC 4742	NETCONF Configuration Protocol over Secure Shell (SSH)	-

3 CWMP

About This Chapter

3.1 Introduction to CWMP

Definition

The CPE WAN Management Protocol (CWMP), also called Technical Report 069 (TR-069), is a technical specification drafted by the Digital Subscriber Line (DSL) forum. CWMP defines the communication mechanism between the customer premises equipment (CPE) and the auto-configuration server (ACS).

Purpose

CWMP provides methods to manage and configure home network devices on the next generation network. Currently, CWMP is applicable to DSL networks, which are facing the following terminal management problems:

- Different vendors manage their terminals in different ways.

  Terminal vendors manage their terminals by using different protocols such as the Optical Network Terminal Management and Control Interface (OMCI) and Embedded Operations Channel (EOC) protocols. Carriers must integrate these vendors' network management systems multiple times to implement unified management.
- Various terminals lead to complex terminal management.

  With emergence of new access technologies, various terminals are developed, such as access points (APs), optical network terminals (ONTs), and shared risk groups (SRGs), which are difficult to manage.
- Troubleshooting is difficult because of a large number of terminals.

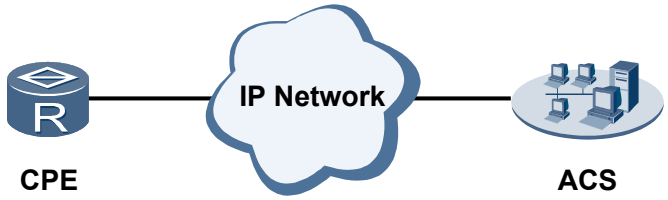
  On a network, most faults occur on the user side, where a large number of terminals are scattered; therefore, troubleshooting is difficult.

To solve the preceding problems, CWMP defines a mechanism to manage the CPE by an ACS. This facilitates CPE management, reduces maintenance and operation costs, and improves troubleshooting efficiency.

3.2 Principles

3.2.1 CWMP Network Model

Figure 3-1 CWMP network model



A CWMP network model contains:

- ACS: manages and maintains CPEs on the network.
- CPE: managed by the ACS.

NOTE

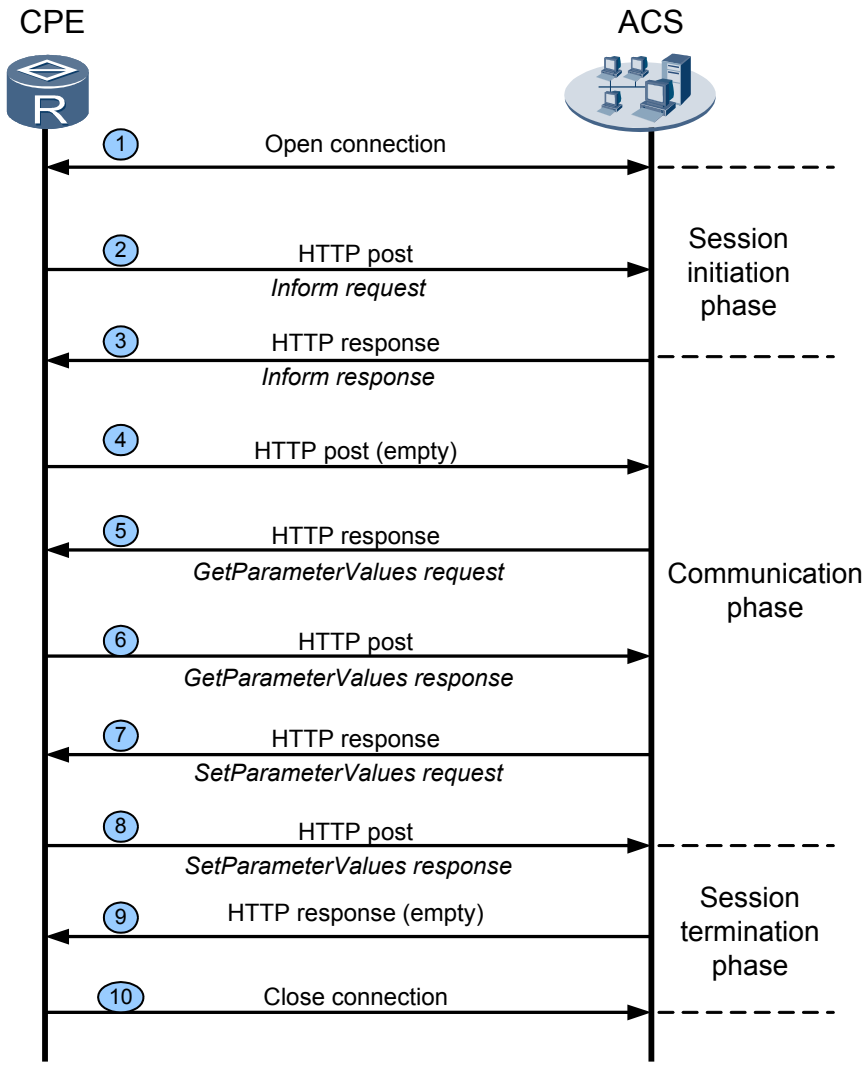
A router is deployed as a CPE.

3.2.2 CWMP Implementation

CWMP Process

Figure 3-2 shows the CWMP working process when the ACS changes a parameter value on the CPE.

Figure 3-2 CWMP process



The CWMP process is as follows:

1. A CPE initiates a session.

If an ACS initiates a session, it sends a Connect request to the CPE (which functions as an HTTP server) to set up a session.

2. The CPE and ACS use Security Socket Layer (SSL) protocol to set up a secure connection.
3. The CPE invokes the Remote Procedure Call (RPC) method Inform to send an Inform request to the ACS, reporting device information and requesting a CWMP connection.
4. After the CPE is authenticated, the ACS sends an Inform response. The Inform method is complete and the CWMP connection is set up.
5. The CPE sends an empty HTTP post message to the ACS, indicating that the CPE does not invoke any more RPC methods supported by the ACS.
6. The ACS invokes the GetParameterValues method to query CPE parameters.
7. The CPE sends a GetParameterValues response containing the queried parameters to the ACS. The GetParameterValues method is complete.
8. The ACS invokes the SetParameterValues method to set CPE parameters.
9. The CPE sends a SetParameterValues response containing the parameter settings to the ACS. The SetParameterValues method is complete.
10. The ACS sends an empty HTTP response to the CPE, indicating that the ACS does not invoke any more RPC methods supported by the CPE.
11. The CPE terminates the connection.

As shown in **Figure 3-2**, the CWMP session goes through three phases.

Session initiation phase

A session can be initiated by a CPE or an ACS.

- A CPE initiates a session in the following scenarios:
  - After startup, the CPE searches for an ACS based on the local configuration or the ACS URL allocated by the Dynamic Host Configuration Protocol (DHCP) server, and then initiates a session.
  - The CPE is configured to send Inform messages at intervals. The CPE will automatically send an Inform message to initiate a session when the interval arrives (1 hour for example).
  - The CPE is configured to send Inform messages at a specified time. The CPE will automatically send an Inform message at the time to initiate a session.
  - If session setup is interrupted unexpectedly and the number of CPE auto-connection retries has not reached the upper limit, the CPE automatically sets up a new connection.
- An ACS initiates a session.

  An ACS can send a Connect request to a CPE at any time. After the CPE authenticates the request, a session between the CPE and the ACS is set up.

  The prerequisite for this method is that the CPE and the ACS have communicated with each other before. During the first communication between the CPE and the ACS, the ACS saves the CPE IP address in the address list. Then it can initiate a session in subsequent communication with the CPE.

CWMP uses security mechanisms to protect communication between a CPE and an ACS. The security mechanisms prevent the transactions between the CPE and the ACS from being tampered and ensure confidentiality of the transactions. CWMP supports the following security mechanisms:

- CPE and ACS authentication:

  - CPE authentication on the ACS side: A CPE sends an Inform request containing an ACS URL to communicate with an ACS. After the CPE is authenticated (the ACS user name and password in the Inform request are the same as those configured on the ACS), a session is set up between the CPE and the ACS.
  - ACS authentication on the CPE side: An ACS sends an HTTP request containing a CPE IP address to communicate with a CPE. After the ACS is authenticated (the CPE user name and password in the HTTP request are the same as those configured on the CPE), a session is set up between the CPE and the ACS.
- Security Socket Layer (SSL) authentication:

  It ensures transaction confidentiality and data integrity and enables the CPE and ACS to authenticate each other using certificates.

Communication phase

After a session is initiated, a CPE or an ACS can send requests to each other to perform operations. For example, the ACS can query and set CPE parameters, and the CPE can upload files to or download files from the file server specified by the ACS.

Session termination phase

Only a CPE can terminate a session.

If the ACS and CPE have sent all necessary requests and received all responses, the CPE terminates the session.

CWMP Operation Methods

An ACS manages and monitors a CPE by performing a series of operations. These operations are called RPC methods in CWMP.

CWMP supports the following standard RPC methods:

- Generic methods: Both the CPE and the ACS must support these methods. **Table 3-1** describes a generic method, which can be invoked by both the CPE and the ACS.

  Table 3-1 Generic method

Method	Description
GetRPCMethods	Used to obtain RPC methods supported by the CPE and the ACS.

- CPE methods: The CPE must support these methods. **Table 3-2** lists the CPE methods, which can be invoked only by the ACS.

  Table 3-2 CPE methods

Method	Description
SetParameterVal-ues	Used by an ACS to set CPE parameters.
GetParameterVal-ues	Used by an ACS to obtain CPE parameter values.

Method	Description
GetParameter-Names	Used by an ACS to discover accessible parameters of a CPE.
SetParameterAttri-butes	Used by an ACS to set attributes of CPE parameters.
GetParameterAttri-butes	Used by an ACS to obtain CPE parameter attributes.
AddObject	Used by an ACS to create instances for a multi-instance object in the CPE data model.
DeleteObject	Used by an ACS to delete instances of a multi-instance object from the CPE data model.
Download	Used by an ACS to request a CPE to download a file from a specified URL and use the downloaded file to replace the local file on the CPE.
Upload	Used by an ACS to request a CPE to upload a file to a specified URL.
Reboot	Used by an ACS to remotely restart a CPE when the CPE is faulty or the CPE software is upgraded.

- ACS methods: The ACS must support these methods. **Table 3-3** lists the ACS methods, which can be invoked only by the CPE.

Table 3-3 ACS methods

Method	Description
Inform	Used by a CPE to send an Inform message to an ACS when the CPE needs to initiate a session with the ACS or periodically send local information to the ACS, or when the CPE bottom-layer configuration changes.
TransferComplete	Used by a CPE to notify an ACS that requested file download or upload is complete no matter whether the file is successfully downloaded or uploaded.

3.2.3 CPE Management

CWMP provides many CPE management functions, which improve CPE operation efficiency and decrease network management problems. Main CPE management functions include:

- Automatic configuration
- File management
- Status and performance monitoring
- Fault diagnosis

Automatic Configuration

CWMP enables an ACS to automatically configure CPEs. When a CPE has set up a session with an ACS, the ACS automatically delivers configurations to the CPE. Automatic configuration parameters include:

- **URL**: address of the ACS
- **Username**: user name used by the CPE to set up a session with the ACS
- **Password**: password used by the CPE to set up a session with the ACS
- **PeriodicInformEnable**: indicates whether Inform messages are sent automatically
- **PeriodicInformInterval**: interval at which Inform messages are sent
- **PeriodicInformTime**: time when Inform messages are sent
- **ConnectionRequestUsername**: CPE user name
- **ConnectionRequestPassword**: CPE password

File Management

CWMP enables CPEs to:

- Upload files

  A CPE can upload the configuration file and log files to the server specified by an ACS to back up important data.

- Download files

  A CPE can use HTTP, HTTPS, or FTP to download web page files, configuration files, system software packages, patch files, license files, and any other files from a file server specified by an ACS. After downloading a file, the CPE checks the validity of the file and processes the file according to the check result. For example, if the downloaded file is a configuration file, the CPE automatically specifies it as the configuration file for next startup and sends the download result (succeeded or failed) to the ACS.

NOTE

- Currently, the CPE does not support file download using digital signature.
- To download a file using HTTPS, the CPE must set up a Secure Sockets Layer (SSL) connection to the ACS.

Status and Performance Monitoring

CWMP enables an ACS to monitor the status and performance parameters of the connected CPEs. Performance and functions vary with CPEs. Therefore, an ACS must be able to identify performance of different CPEs and monitor configurations and configuration changes of each CPE.

CWMP allows network administrators to define monitoring parameters and obtain the CPE status and statistics using an ACS.

NOTE

Currently, the CPE does not support the data model defined in TR-143 among all technical specifications that define status and performance monitoring.

Fault Diagnosis

CWMP enables an ACS to diagnose CPE faults using methods such as ping, traceroute, asynchronous transfer mode (ATM) loopback, and digital subscriber line (DSL) detection.

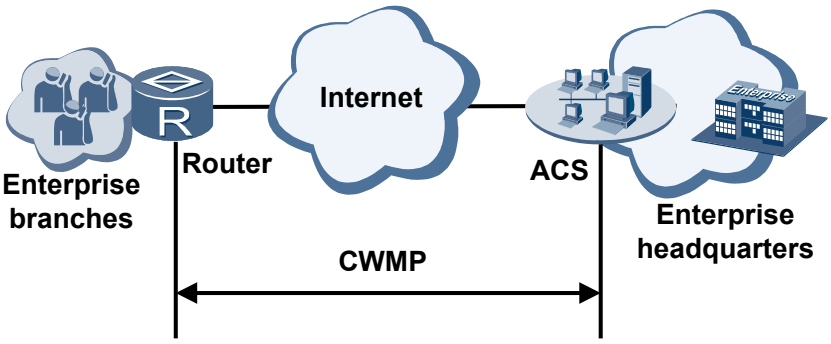
NOTE

Currently, the CPE does not support the data model defined in TR-098 among all technical specifications that define fault diagnosis.

3.3 Applications

As shown in **Figure 3-3**, enterprise branches connect to the Internet through the enterprise gateway router. CWMP is deployed between the enterprise headquarters and branches. The router functions as a CPE. The enterprise headquarters control and manage the router using an ACS. After connecting to the router, the ACS manages the system startup file and configuration file for the router, configures the router, monitors the router status and performance, and diagnoses router faults.

Figure 3-3 CWMP application



3.4 References

Document	Description	Remarks
CPE/ACS Application	The application uses the CPE WAN Management Protocol on the CPE and ACS, respectively. The application is locally defined and not specified as part of the CPE WAN Management Protocol	N/A
RPC Methods	The specific RPC methods that are defined by the CPE WAN Management Protocol	N/A
SOAP	A standard XML-based syntax used here to encode remote procedure calls. Specifically Simple Object Access Protocol (SOAP) 1.1.	Only one authentication password can be configured.

Document	Description	Remarks
HTTP	RFC 2616, Hypertext Transfer Protocol -- HTTP/1.1	N/A
SSL/TLS	The standard Internet transport layer security protocols. Specifically, either SSL 3.0 (Security Socket Layer), or TLS 1.0 (Transport Layer Security).	N/A
TCP/IP	Standard TCP/IP	N/A

4 NTP

About This Chapter

4.1 Introduction to NTP

Definition

The Network Time Protocol (NTP) is an application layer protocol in the TCP/IP protocol suite. NTP is used to synchronize the time among a set of distributed time servers and clients. NTP is implemented based on the Internet Protocol (IP) and User Datagram Protocol (UDP). NTP packets are transmitted using UDP port 123.

Purpose

As network topologies become increasingly complex, clock synchronization becomes more important for devices on the entire network. If a system clock is modified manually by network administrators, the workload is heavy and the modification is error-prone, which affects clock precision. NTP is formulated as a networking protocol for clock synchronization between devices on a network.

NTP applies to the following situations where all the clocks of the devices on a network need to be consistent:

- In network management, analysis of logs or debugging messages collected from different routers requires time for reference.
- An accounting system requires that the clocks of all the devices be consistent.
- When several systems work together to process a complicated event, they have to refer to the same clock to ensure a correct execution order.
- Incremental backup between a backup server and clients requires that their clocks be synchronized.
- Some applications need to obtain the time in which a user logs in a system and a document is modified.

Version Evolution

NTP is evolved from the Time Protocol and the ICMP Timestamp message but is specifically designed to maintain accuracy and robustness. **Table 4-1** shows the NTP version evolution.

Table 4-1 NTP version evolution

Version	Date	Protocol Number	Description
NTPv1	June 1988	RFC 1059	NTPv1 puts forward complete NTP rules and algorithms for the first time, but it does not support authentication and control messages.

Version	Date	Protocol Number	Description
NTPv2	September 1989	RFC 1119	NTPv2 supports authentication and control messages.
NTPv3	March 1992	RFC 1305	NTPv3 uses correctness principles and improves clock selection and filter algorithms, and it is widely used.
NTPv4	June 2010	RFC 5905	NTPv3 only applies to an IPv4 network. As IPv6 develops and network security requirements grow, NTPv4 is produced. NTPv4, an extension of NTPv3, is compatible with NTPv3. ● NTPv4 applies to both IPv4 and IPv6 networks. ● NTPv4 provides a complete encryption and authentication system so it is more secure than NTPv3.

4.2 Principles

4.2.1 Operating Principle

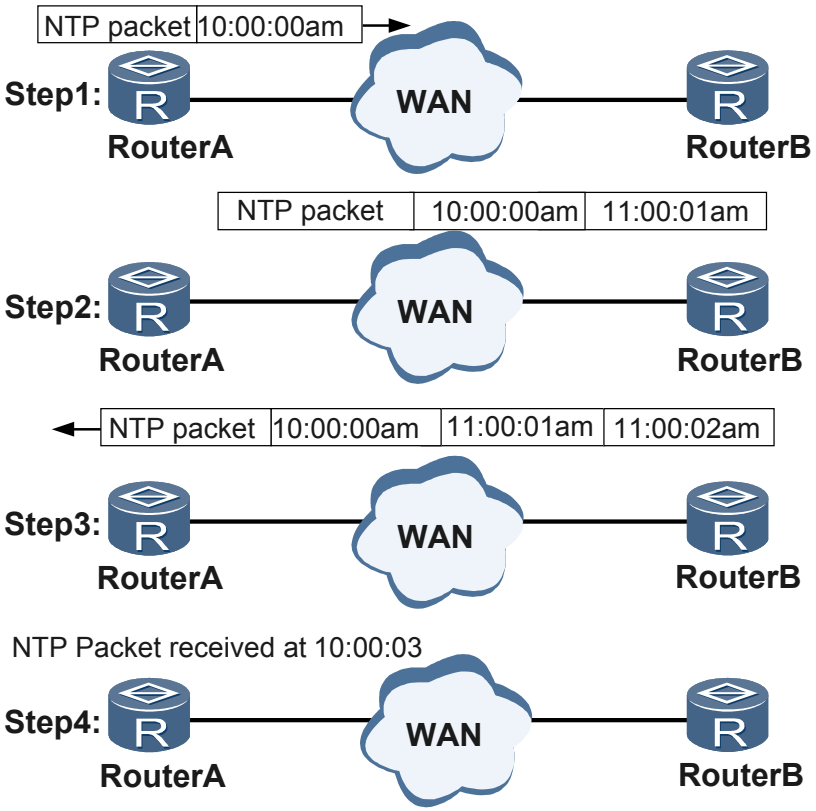
Figure 4-1 shows NTP implementation:

Router A and Router B are connected through a wide area network (WAN). Each of them has its own system clock, which is synchronized automatically through NTP.

Presuming that:

- Before the clocks of Router A and Router B are synchronized, the clock of Router A is 10:00:00 a.m. and the clock of Router B is 11:00:00 a.m.
- Router B acts as an NTP time server, and Router A must synchronize its clock with that of Router B.
- It takes one second to unidirectionally transmit an NTP message between Router A and Router B.
- Both Router A and Router B take one second to process an NTP message.

Figure 4-1 Diagram of NTP implementation



The process of synchronizing the system clock is as follows:

1. Router A sends an NTP message to Router B. The message carries an initial timestamp, 10:00:00 a.m. (T1), indicating the time when it leaves Router A.
2. When the NTP message reaches Router B, Router B adds the timestamp 11: 00:01 a.m. (T2) to the NTP message, indicting the time when Router B receives the message.
3. When the NTP message leaves Router B, Router B adds the transmit timestamp 11:00:02 a.m. (T3) to the NTP message, indicating the time when the message leaves Router B.
4. When Router A receives this response message, it adds a new receive timestamp, 10:00:03 a.m. (T4).

   Router A uses the information in the received message to calculate the following two important parameters:

   - Roundtrip delay of the NTP message: Delay = (T4 - T1) - (T3 - T2)
   - Clock offset of Router A by taking Router B as a reference: Offset = ((T2 - T1) + (T3 - T4))/2
5. After the calculation, Router A knows that the roundtrip delay is 2 seconds and the clock offset of Router A is 1 hour. Router A sets its own clock based on these two parameters to synchronize its clock with that of Router B.

NOTE

The preceding example is only a brief description of the operating principle of NTP. In fact, NTP uses the standard algorithms in RFC 1305 to ensure the precision of clock synchronization.

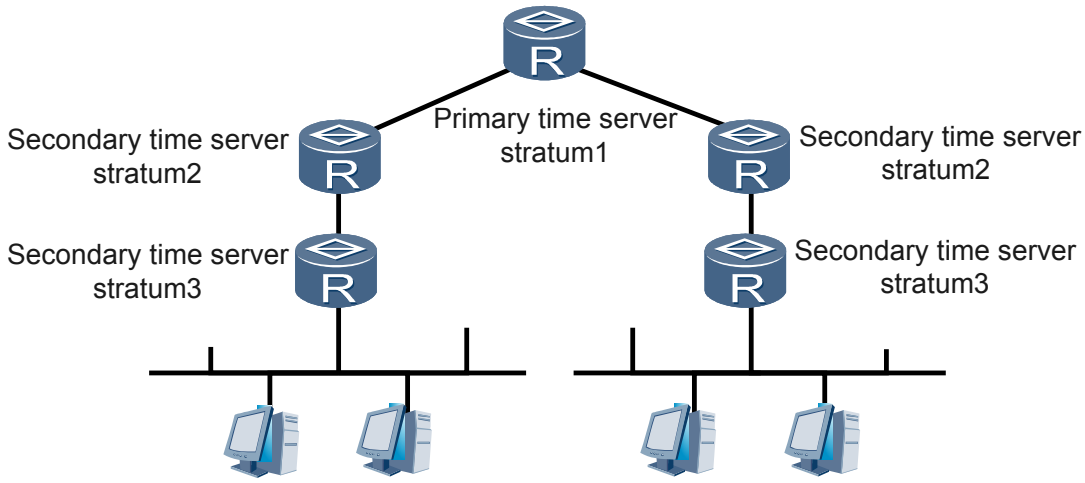
4.2.2 Network Architecture

In a synchronization subnet, the **primary time server** sends time information to other **secondary time servers** using the NTP protocol. The secondary time servers then synchronize their clocks with the primary time server. These servers are hierarchically connected, and each level of the hierarchy is called a **stratum** and assigned a layer number. For example, the primary time server is a stratum 1 server, the secondary time servers are stratum 2 servers, and following strata can be obtained by analogy. A larger clock stratum indicates lower precision.

The NTP network architecture involves the following concepts:

- **Synchronization subnet** consists of the primary time server, secondary time servers, clients, and interconnecting transmission paths, as shown in **Figure 4-2**.
- **Primary time server** directly synchronizes its clock with a standard reference clock using a cable or radio. The standard reference clock is usually a radio clock or the Global Positioning System (GPS).
- **Secondary time server** synchronizes its clock with the primary time server or other secondary time servers on the network. A secondary time server transmits the time information to other hosts on a LAN through NTP.
- **Stratum** is a hierarchical standard for clock synchronization. It represents precision of a clock. The value of a stratum ranges from 1 to 16. A smaller value indicates higher precision. The value 1 indicates the highest clock precision, and 16 indicates that the clock is not synchronized.

Figure 4-2 NTP network architecture



Under normal circumstances, the primary time server and the secondary time servers in a synchronization subnet are arranged in a hierarchical-master-slave structure. In this structure, the primary time server is located at the root, and the secondary time servers are arranged close to leaf nodes. As their strata increase, the precision decreases accordingly. The extent to which the precision of the secondary time servers decreases depends on stability of network paths and the local clock.

NOTE

When the synchronization subnet has multiple primary time servers, the optimal server can be selected using an algorithm.

Such a design ensures that:

- When faults occur in one or more primary/secondary time servers or network paths interconnecting them, the synchronization subnet will automatically be reconstructed into another hierarchical-master-slave structure to obtain the most precise and reliable time.
- When all primary time servers in the synchronization subnet become invalid, a standby primary time server runs.

When all primary time servers in the synchronization subnet become invalid, other secondary time servers are synchronized among themselves. These secondary time servers become independent of the synchronization subnet and automatically run at the last determined time and frequency. When a router with a stable oscillator becomes independent of the synchronization subnet for an extended period of time, its timing error can be kept less than several milliseconds in a day because of highly precise calculations.

4.2.3 Operating Mode

A device may use multiple NTP operating modes to perform time synchronization.

- **Unicast Server/Client Mode**
- **Symmetric Peer Mode**
- **Broadcast Mode**
- **Multicast Mode**
- **Manycast Mode**

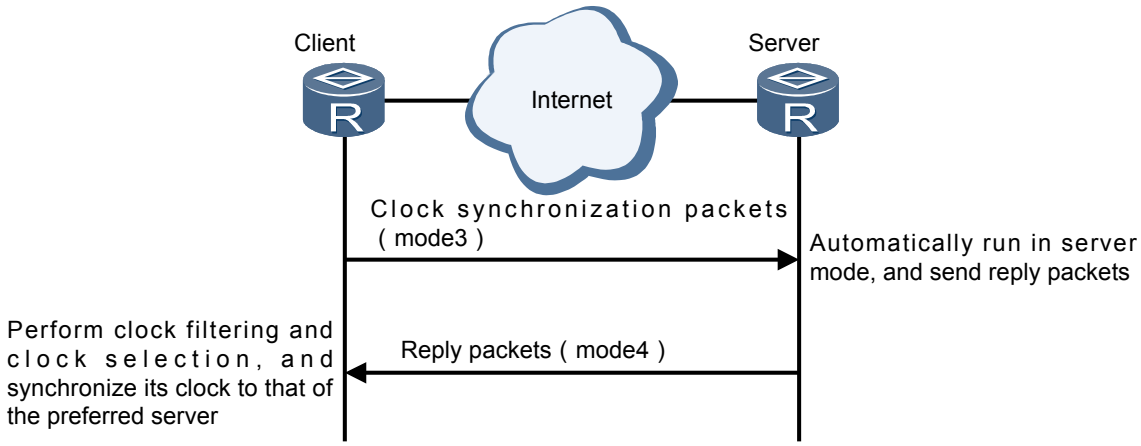
You can select an appropriate operating mode as required. When an IP address of the NTP server or peer device cannot be determined or a large number of devices require synchronization on a network, the broadcast or multicast mode can be used for clock synchronization. In server and peer mode, the devices synchronize their clocks with a specified server or peer, which increases clock reliability.

Unicast Server/Client Mode

The unicast server/client mode runs on a higher stratum on a synchronous subnet. In this mode, devices need to obtain the IP address of the server in advance.

- Client: A host running in client mode (client for short) periodically sends packets to the server. The Mode field in the packets is set to 3, indicating that the packets are coming from a client. After receiving a reply packet, the client filters and selects clock signals, and synchronizes its clock with the server that provides the optimal clock. A client does not check the reachability and stratum of the server. Usually, a host running in this mode is a workstation on a network. It synchronizes its clock with the clock of a server but does not change the clock of the server.
- Server: A host running in server mode (server for short) receives the packets from clients and responds to the packets received. The Mode field in reply packets is set to 4, indicating that the packets are coming from a server. Usually, the host running in server mode is a clock server on a network. It provides synchronization information for clients but does not change its own clock.

Figure 4-3 Unicast Client/Server Mode



During and after the restart, the host operating in client mode periodically sends NTP request messages to the host operating in server mode. After receiving the NTP request message, the server swaps the position of destination IP address and source IP address, and the source port number and destination port number, fills in the necessary information, and sends the message to the client. The server does not need to retain state information when the client sends the request message. The client freely adjusts the interval for sending NTP request messages according to the local conditions.

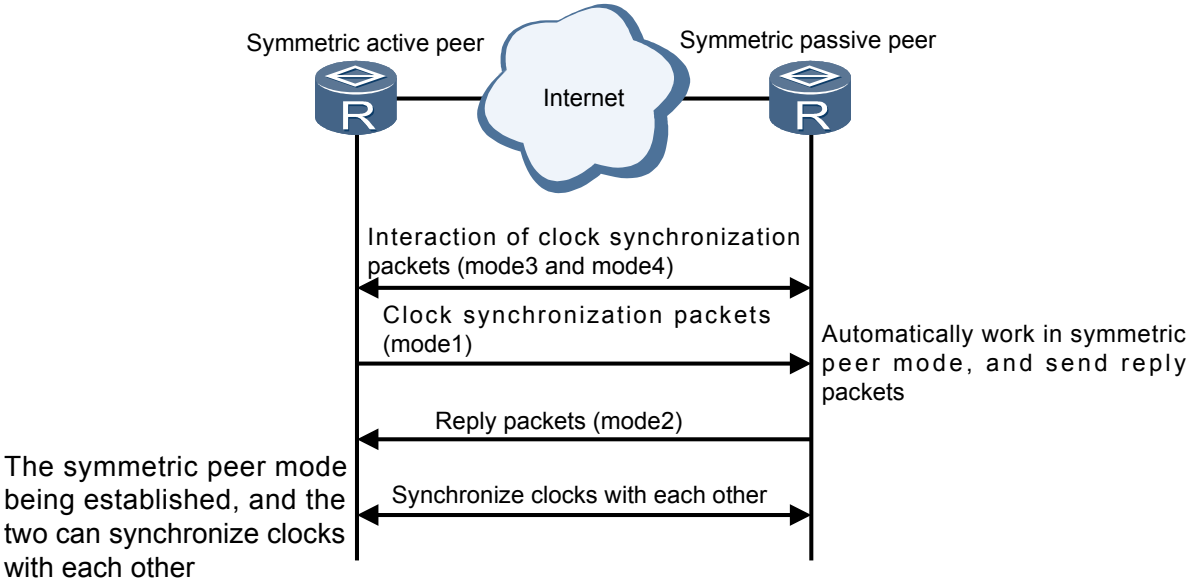
Symmetric Peer Mode

The peer mode runs on a lower stratum on a synchronous subnet. In this mode, a symmetric active peer and a symmetric passive peer can synchronize with each other. The symmetric peer with a higher stratum (a lower level) synchronizes with a symmetric peer with a lower stratum (a higher level).

In symmetric peer mode, the symmetric active peer initiates an NTP packet with the Mode field set to 3 (the client mode), and the symmetric passive peer responds with an NTP packet with the Mode field set to 4 (the server mode). This interaction creates a network delay so that devices at both ends enter the symmetric peer mode.

- Symmetric active peer: A host that functions as a symmetric active peer sends packets periodically. The value of the Mode field in a packet is set to 1. This indicates that the packet is sent by a symmetric active peer, without considering whether its symmetric peer is reachable and which stratum its symmetric peer is on. The symmetric active peer can provide time information about the local clock for its symmetric peer, or synchronize the time information about the local clock based on that of the symmetric peer clock.
- Symmetric passive peer: A host that functions as a symmetric passive peer receives packets from the symmetric active peer and sends reply packets. The value of the Mode field in a reply packet is set to 2. This indicates that the packer is sent by a symmetric passive peer. The symmetric passive peer can provide time information about the local clock for its symmetric peer, or synchronize the time information about the local clock based on that of the symmetric peer clock.

Figure 4-4 Symmetric peer mode



The prerequisite for having a host run in symmetric passive mode is that: The host receives an NTP packet from a symmetric peer running in symmetric active peer mode. The symmetric active peer has a stratum lower than or equal to that of the host, and is reachable from the local host.

NOTE

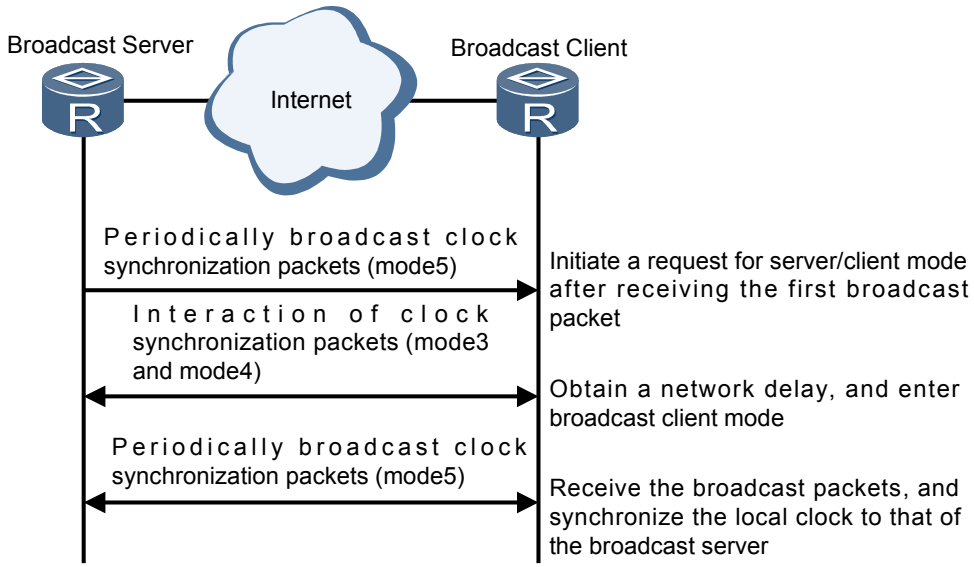
The symmetric passive peer does not need to be configured. A host sets up a connection and sets relevant state variables only when it receives an NTP packet.

Broadcast Mode

The broadcast mode is applied to the high speed network that has multiple workstations and does not require high accuracy. In a typical scenario, one or more clock servers on the network periodically send broadcast packets to the workstations. The delay of packet transmission in a LAN is at the milliseconds level.

- Broadcast server: A host that runs in broadcast mode sends clock synchronization packets to the broadcast address 255.255.255.255 periodically. The value of the Mode field in a packet is set to 5. This indicates that the packet is sent by a host that runs in broadcast mode, without considering whether its peer is reachable and which stratum its peer is on. The host running in broadcast mode is usually a clock server running high-speed broadcast media on the network, which provides synchronization information for all of its peers but does not alter the clock of its own.
- Broadcast client: The client listens to the clock synchronization packets sent from the server. When the client receives the first clock synchronization packet, the client and server exchange NTP packets whose values of Mode fields are 3 (sent by the client) and the NTP packets whose values of Mode fields are 4 (sent by the server). In this process, the client enables the server/client mode for a short time to exchange information with the remote server. This allows the client to obtain the network delay between the client and the server. Then, the client returns the broadcast mode, and continues to sense the incoming clock synchronization packets to synchronize the local clock.

Figure 4-5 Broadcast mode

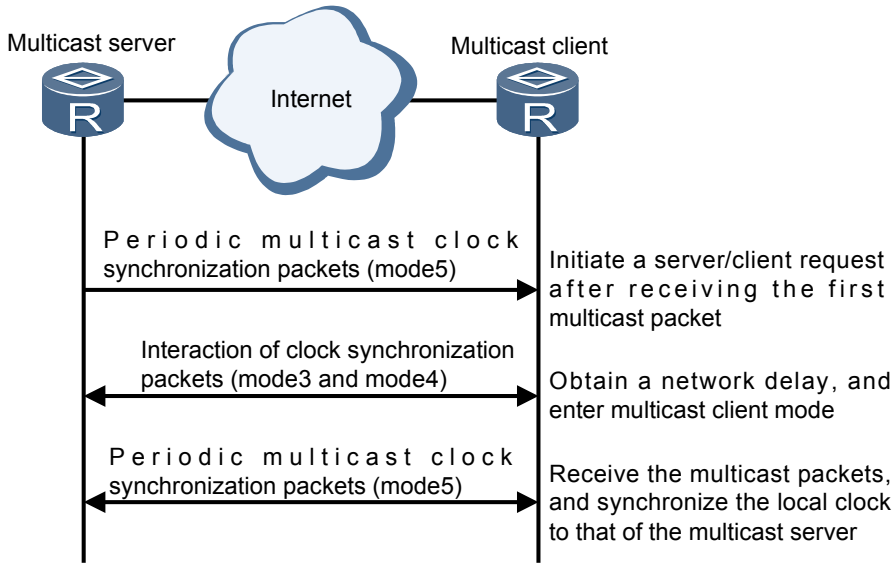


Multicast Mode

Multicast mode is useful when there are large numbers of clients distributed in a network. This normally results in large number of NTP packets in the network. In the multicast mode, a single NTP multicast packet can potentially reach all the clients on the network and reduce the control traffic on the network.

- Multicast server: A server running in multicast mode sends clock synchronization packets to a multicast address periodically. The value of the Mode field in a packet is set to 5. This indicates that the packet is sent by a host that runs in multicast mode. The host running in multicast mode is usually a clock server running high-speed broadcast media on the network, which provides synchronization information for all of its peers but does not alter the clock of its own.
- Multicast client: The client listens to the multicast packets from the server. When the client receives the first broadcast packet, the client and server exchange NTP packets whose values of Mode fields are 3 (sent by the client) and the NTP packets whose values of Mode fields are 4 (sent by the server). In this process, the client enables the server/client mode for a short time to exchange information with the remote server. This allows the client to obtain the network delay between the client and the server. Then, the client returns the multicast mode, and continues to sense the incoming multicast packets to synchronize the local clock.

Figure 4-6 Multicast mode



Manycast Mode

Manycast mode is applied to a small set of servers scattered over the network. Clients can discover and synchronize to the closest manycast server. Manycast can especially be used where the identity of the server is not fixed and a change of server does not require reconfiguration of all the clients in the network.

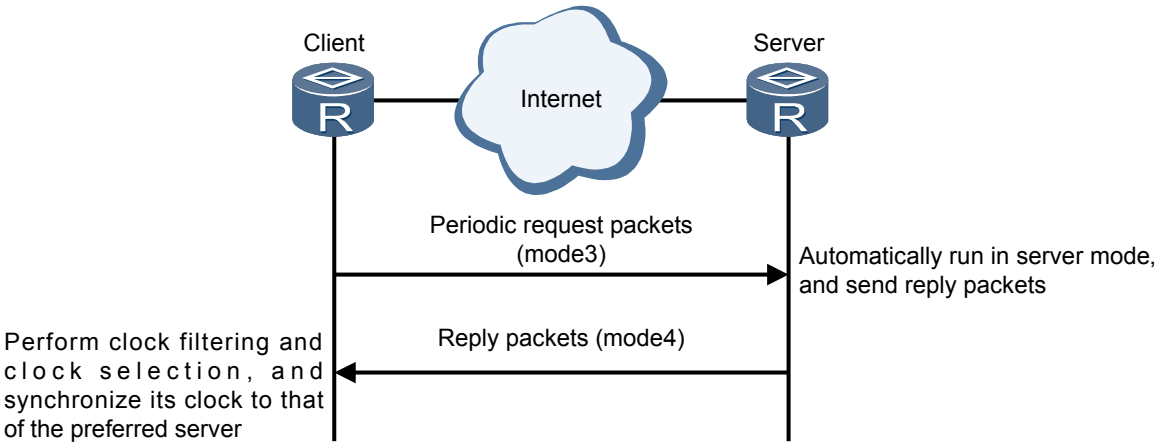
- Manycast client: The client in manycast mode periodically sends request packets (the Mode field is set to 3) to an IPv4/IPv6 multicast address. After receiving a reply packet, the client filters and selects clock signals, and synchronizes its clock with the server that provides the optimal clock.
- Manycast server: The manycast server continuously listens to the packets. If a server can be synchronized, the server returns a packet (the Mode field is set to 4) by using the unicast address of the client as the destination address.

To prevent the client from constantly sending NTP request packets to the manycast server and reduce the load of the server, the NTP protocol defines a minimum number of connections. In manycast mode, the client records the number of connections established every time it synchronizes clock with the server. The minimum number of connections is the minimum number of connections called during a synchronization process. If the number of connections called by the client reaches the minimum number during subsequent synchronization processes and the synchronization is completed, the client considers that the synchronization is completed. After that, the client sends a packet every time a timeout period expires to maintain the connection. The NTP protocol uses the time to live (TTL) mechanism to ensure that the client can successfully synchronize with the server. Every time the client sends an NTP packet, the TTL of the packet increases (the initial value as 1) until the minimum number of connections is reached or the TTL value reaches the upper limit. If the TTL reaches the upper limit or the number of connections called by the client reaches the minimum number, but connections called by the client still cannot complete the synchronizing process, the client stops data transmission in a timeout period to eliminate all connections. Then the client repeats the preceding process.

NOTE

In NTP implementation, a peer structure is established for each synchronization source, and these peer structures are stored in a chain in a Hash form. Each peer structure is corresponding to a connection. A single device supports a maximum of 128 connections. When the number of connections exceeds 128, no new connection can be established.

Figure 4-7 Manycast mode



4.2.4 Security Mechanism

When a time server on a synchronization subnet is faulty or encounters a malicious attack, timekeeping on other clock servers on the subnet should not be affected. To meet this requirement, NTP provides the following security mechanisms to ensure network security: access authority, Kiss-o'-Death (KOD) and NTP authentication.

Access Authority

A device provides access authority, which is simpler and more secure, to protect a local clock.

NTP access control is implemented based on an access control list (ACL). NTP supports four levels of access authority, and a corresponding ACL rule can be specified for each level. If an NTP access request hits the ACL rule for a level of access authority, they are successfully matched and the access request enjoys the access authority at this level.

When an NTP access request reaches the local end, the access request is successively matched with the access authority from the maximum one to the minimum one. The first successfully matched access authority takes effect. The matching order is as follows:

1. peer: indicating the maximum access authority. A time request may be made for the local clock and a control query may be performed on the local clock. The local clock can also be synchronized to a remote server.
2. server: indicating that a time request may be made for the local clock and a control query may be performed on the local clock, but the local clock cannot be synchronized with the clock of the remote server.
3. synchronization: indicating that only a time request can be made for the local clock.
4. query: indicating the minimum access authority. Only a control query can be performed on the local clock.

5. limited: taking effect only when the KoD function is enabled. The rate of incoming packets is controlled and the kiss code is sent after the KoD function is enabled.

KOD

When a server receives a large number of client access packets within a specified period of time and cannot bear the load, the KOD function can be enabled on the server to perform access control. KOD is a brand new access control technology that is put forward in NTPv4, and it is used by the server to provide information, such as a status report and access control, for the client.

A KOD packet is a special NTP packet. When the Stratum field in an NTP packet is 0, the packet is called a KOD packet and the ASCII message it conveys is called kiss code and represents access control information. Currently, only two types of kiss codes are supported: DENY and RATE.

After the KOD function is enabled on the server, the server sends kiss code DENY or RATE to the client based on the configuration.

NOTE

After the KOD function is enabled, the corresponding ACL rule needs to be configured. When the ACL rule is configured as **deny**, the server sends the kiss code DENY. When the ACL rule is configured as **permit** and the rate of NTP packets received reaches the configured upper limit, the server sends the kiss code RATE.

- When the client receives kiss code DENY, the client terminates all connections to the server and stops sending packets to the server.
- When the client receives kiss code RATE, the client immediately reduces its polling interval to the server and continues to reduce the interval each time it receives a RATE kiss code.

Authentication

The NTP authentication function can be enabled on networks demanding high security. Different keys may be configured in different operating modes.

When a user enables the NTP authentication function in a certain NTP operating mode, the system records the key ID in this operating mode.

- **Sending process**

  The system determines whether authentication is required in this operating mode. If authentication is not required, the system directly sends a packet. If authentication is required, the system encrypts the packet using the key ID and an encryption algorithm and sends it.

  NOTE

  Currently, devices support only the MD5 key authentication algorithm.

- **Receiving process**

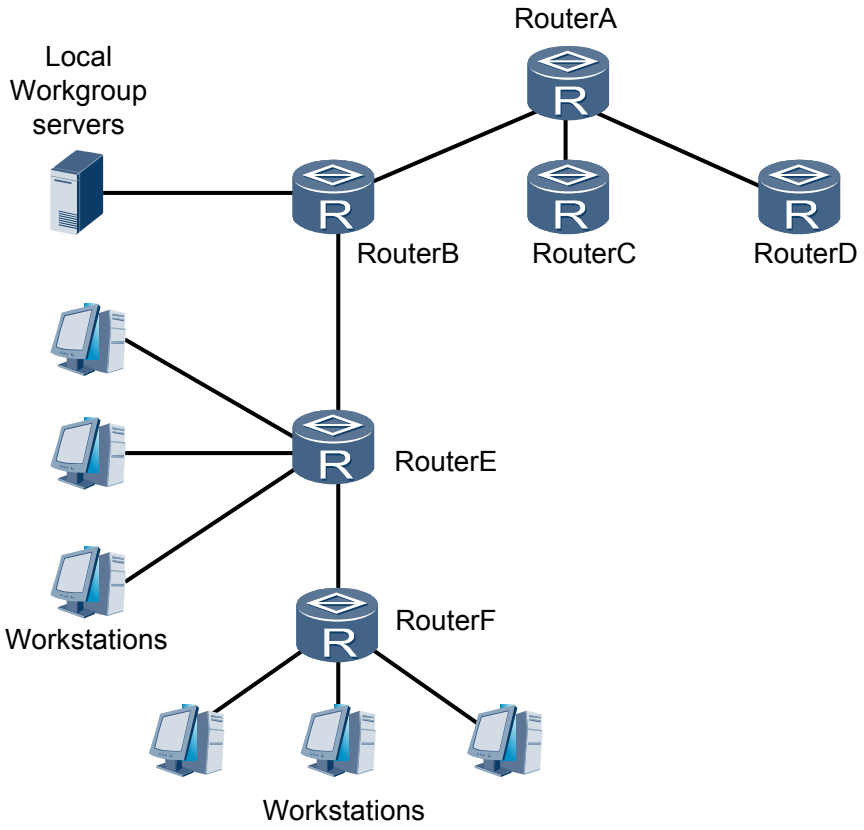
  After receiving a packet, the system determines whether the packet needs to be authenticated. If the packet does not need to be authenticated, the system directly performs subsequent processing on the packet. If the packet needs to be authenticated, the system authenticates the packet using the key ID and a decryption algorithm. If the authentication fails, the system directly discards the packet. If the authentication succeeds, the system processes the received packet.

4.3 Application

Typical Application

On the network as shown in **Figure 4-8**, Router A accessing a standard clock is used as the NTP master clock server to achieve synchronization of clocks on the entire network. Router A is configured as the unicast server, and Router B, Router C and Router D are configured as unicast clients. Router E acts as a symmetric peer of the upstream Router B and downstream Router F.

Figure 4-8 Typical networking

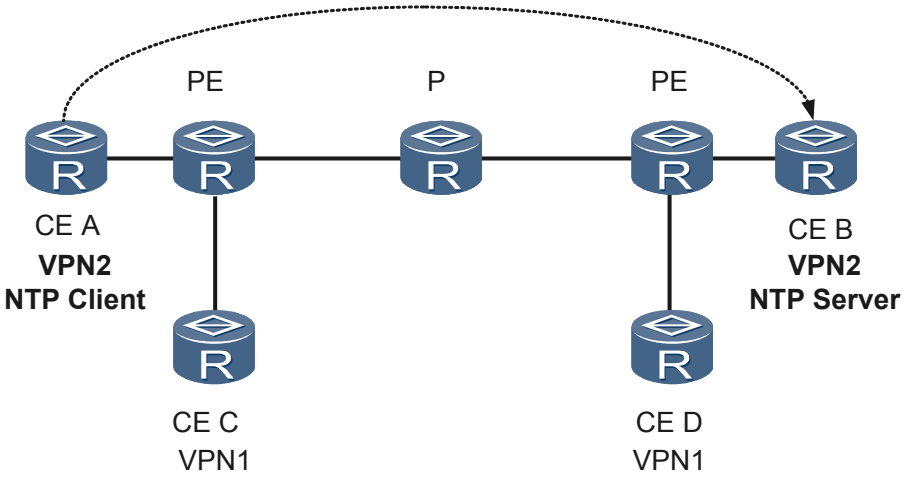


Application in VPN Networking

NTP supports time synchronization on an MPLS VPN network. Specifically, network devices (CE and PE) at different geographical locations can achieve time synchronization through NTP as long as they belong to the same VPN.

Figure 4-9 shows application of the NTP service on a VPN network. Both CE A and CE B belong to VPN 2. CE B is used as an NTP unicast server, CE A is used as an NTP unicast client, and NTP time synchronization can be implemented between CE B and CE A.

Figure 4-9 VPN



4.4 Reference Standards and Protocols

The following table provides reference standards and protocols for NTP.

Document No.	Description
RFC 1305	Network Time Protocol (Version 3) Specification, Implementation and Analysis
RFC 5905	Network Time Protocol Version 4: Protocol and Algorithms Specification
RFC 5906	Network Time Protocol Version 4: Autokey Specification

5 NQA

About This Chapter

5.1 Introduction to NQA

Definition

Network quality analysis (NQA) is a feature that the system provides to monitor network quality of service (QoS) in real time and to locate and diagnose network faults. Independent of the hardware, NQA functions on the link layer to measure the performance of protocols running at the network layer, transport layer, and application layer.

Purpose

To visualize the quality of network services and allow users to check whether the quality of network services meets requirements, the following measures must be taken:

- Collect data on network devices to describe the quality of network services.
- Deploy probe devices to monitor the quality of network services.

The preceding measures require devices to provide statistical parameters such as the delay, jitter, and packet loss ratio and require dedicated probe devices. These requirements increase investments on devices.

NQA can precisely test the network operating status and output statistics without using dedicated probe devices, effectively saving costs.

NQA measures the performance of different protocols running on the network. It allows you to collect network operation indexes in real time, such as the total HTTP connection delay, TCP connection delay, DNS resolution delay, file transmission rate, FTP connection delay, and DNS resolution error ratio.

5.2 Principles

Constructing a test instance

NQA requires two test ends, an NQA client and an NQA server (or called the source and destination). The NQA client (or the source) initiates an NQA test. You can configure test instances through command lines or the NMS. Then NQA places the test instances into test queues for scheduling.

Starting a test instance

When starting an NQA test instance, you can choose to start the test instance immediately, at a specified time, or after a delay. A test packet is generated based on the type of a test instance when the timer expires. If the size of the generated test packet is smaller than the minimum size of a protocol packet, the test packet is generated and sent out with the minimum size of the protocol packet.

Processing a test instance

After a test instance starts, the protocol-related running status can be collected according to response packets. The client adds a timestamp to a test packet based on the local system time before sending the packet to the server. After receiving the test packet, the server sends a response packet to the client. The client then adds a timestamp to the received response packet based on

the current local system time. This helps the client calculate the round-trip time (RTT) of the test packet based on the two timestamps.

NOTE

In a jitter test instance, both the client and server add a timestamp to the sent and received packets based on the local system time. In this manner, the client can calculate the jitter value.

You can view the test results to learn about the network operating status and service quality.

5.3 Test Mechanism

5.3.1 DHCP Test

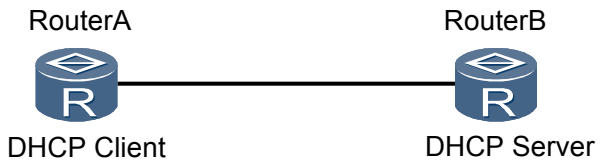
An NQA DHCP test is performed using User Datagram Protocol (UDP) packets. The NQA client simulates a DHCP client to initiate a DHCP request on a specified interface. According to whether the interface obtains an IP address, you can determine whether DHCP servers are available on the network segment where the interface resides and measure the time the interface takes to obtain an IP address.

Figure 5-1 shows the process of a DHCP test:

1. The client (RouterA) broadcasts a DHCP Discovery packet through the interface that needs to obtain an IP address to query a DHCP server. The Discovery packet is broadcast to the network segment where the interface resides.
2. After receiving the Discovery packet, the DHCP server (RouterB) returns a DHCP Offer packet carrying its own IP address, to the client.
3. The client broadcasts a DHCP Request packet to the network segment where the interface resides. The Request packet contains the IP address of the DHCP server.
4. After receiving the Request packet, the DHCP server returns a DHCP ACK packet carrying an IP address assigned to the interface.

   After receiving the DHCP ACK packet, the client calculates the time taken to obtain an IP address from the DHCP server by subtracting the time the client sends the Discovery packet from the time the client receives the ACK packet.

Figure 5-1 DHCP test scenario



A DHCP test only uses an interface to send DHCP packets and releases the DHCP lease after obtaining an IP address for the interface. Therefore, the DHCP test does not consume address resources of the DHCP server. The interface used in a DHCP server must be in Up state.

The DHCP test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

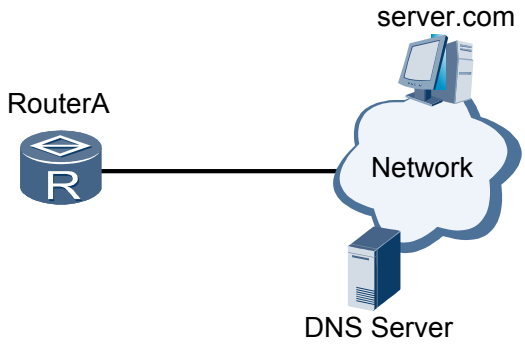
5.3.2 DNS Test

An NQA DNS test is performed using UDP packets. The NQA client simulates a DNS client to send a DNS request to a specified DHCP server. According to whether DNS resolution succeeds and the time taken for DNS resolution, you can determine whether the DNS server is available and measure the DNS resolution speed.

Figure 5-2 shows the process of a DNS test:

1. The DNS client (RouterA) sends a DNS Query packet to the DNS server, requesting the server to resolve a specified DNS name.
2. After receiving the Query packet, the DNS server constructs a Response packet and sends it to the client.
3. After receiving the Response packet, the client calculates the difference between the time the client sends the Query packet and the time the client receives the Response packet to obtain the time taken to resolve the DNS name. This can reflect DNS protocol performance on the network.

Figure 5-2 DNS test scenario



A DNS test only simulates the DNS resolution process but not saves the mapping between domain names and IP addresses.

The DNS test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

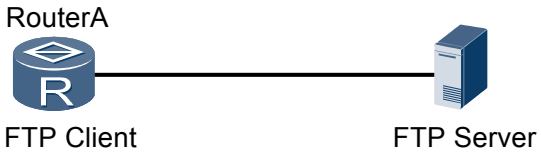
5.3.3 FTP Test

An NQA FTP test is performed using TCP packets. According to the test results, you can determine whether an FTP client can establish a connection with a specified FTP server and measure the time taken to download a specified file or upload a specified file to the FTP server.

Figure 5-3 shows the process of an FTP test. An NQA FTP test obtains the responding speed in two phases:

- Control connection setup: You can obtain the time taken by the client (RouterA) to set up a TCP control connection with the FTP server through three-way handshake and the time taken to exchange signals through the control connection.
- Data connection setup: You can obtain the time taken by the client (RouterA) to download a specified file from the FTP server or upload a specified file to the FTP server through the data connection.

Figure 5-3 FTP test scenario



In an FTP test, the following data can be calculated based on the information in the packets received by the client:

- Minimum, maximum, and average time taken to set up a control connection
- Minimum, maximum, and average time taken to set up a data connection

FTP tests support file upload and download. During a file download, the downloaded file is not saved to the local file system, and only the time taken to download the file is calculated. After the file download time is obtained, the occupied memory is automatically released. During a file upload, the file with fixed size and contents but not the local file are uploaded to the FTP server. The name of the file to be uploaded is specified and the data in the file is specified by the system. If the specified file name is the same as an existing file name on the server, the specified file overwrites the existing file. After an FTP test is complete, the file is not deleted. Therefore, FTP tests are independent of the local file system.

The FTP test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

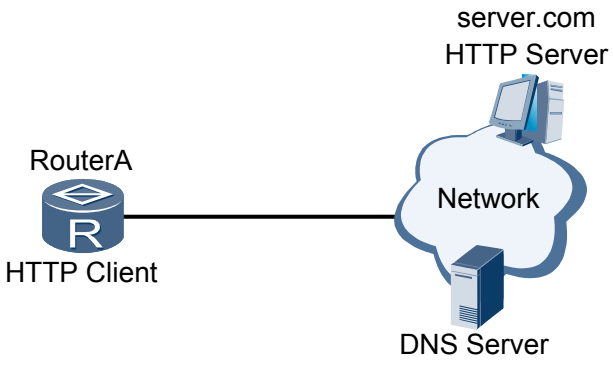
5.3.4 HTTPTest

An NQA HTTP test detects whether the client can set up a connection with a specified HTTP server. According to the test results, you can determine whether a device provides the HTTP service and measure the time taken to set up a connection.

Figure 5-4 shows the process of an HTTP test. An NQA HTTP test obtains the responding speed in three phases:

- DNS resolution: You can obtain the DNS resolution time, the period from when the client (RouterA) sends a DNS packet to the resolver to resolve the name of the HTTP server to an IP address to the time when the client receives a DNS resolution packet containing the IP address.
- TCP connection setup: You can obtain the time taken to set up a TCP connection between the client and the HTTP server through three-way handshake.
- TCP transaction: You can obtain the transaction time, the period from the time the client sends a Get or Post packet to the HTTP server to the time the client receives a response packet from the HTTP server.

Figure 5-4 HTTP test scenario



In an HTTP test, the following data can be calculated based on the information in the packets received by the client:

- Minimum, maximum, and total time of DNS resolution
- Minimum, maximum, and total time taken to set up a TCP connection
- Minimum, maximum, and total HTTP transaction time

The HTTP test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

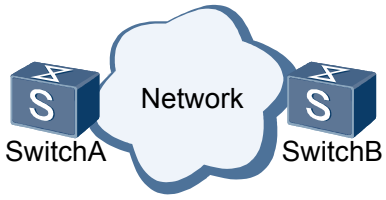
5.3.5 ICMP Jitter Test

An ICMP jitter test is implemented using ICMP packets to obtain the delay, jitter, and packet loss ratio based on the timestamp in test packets. The jitter time equals the interval for receiving two consecutive packets minus the interval for sending the two packets.

Figure 5-5 shows the process of an ICMP jitter test:

1. The source (SwitchA) sends packets to the destination (SwitchB) at a specified interval.
2. After receiving a packet, the destination adds a timestamp to the packet and sends it back to the source.
3. After receiving the returned packet, the source calculates the jitter by subtracting the interval at which the source sends two consecutive packets from the interval at which the destination receives the two consecutive packets.

Figure 5-5 ICMP jitter test scenario



NOTE

In an ICMP jitter test, the interval for sending packets is configurable and defaults to 20 ms; the number of packets to be sent each time is configurable and defaults to 60.

The following data can be calculated based on information in the packets received by the source:

- Maximum, minimum, and average jitter of the packets from the source to the destination and from the destination to the source
- Maximum unidirectional delay from the source to the destination or from the destination to the source

In an ICMP jitter test, you can set the number of consecutive packets to be sent in a single test instance. This setting allows you to simulate the actual traffic of specified data within a specified period.

The ICMP jitter test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.3.6 ICMP Test

An NQA ICMP test detects whether there are reachable routes from the source to the destination. An ICMP test has similar functions as the ping command except that the ICMP test provides more output information:

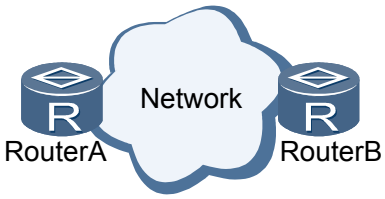
- By default, the system saves results of the latest five tests.
- The test results include the average delay, packet loss ratio, and time the last packet is correctly received.

Figure 5-6 shows the process of an ICMP test:

1. The source (RouterA) constructs an ICMP Echo Request packet and sends it to the destination (RouterB).
2. After receiving the ICMP Echo Request packet, the destination responds the source with an ICMP Echo Reply packet.

   The source then can calculate the time for communication between the source and the destination by subtracting the time the source sends the ICMP Echo Request packet from the time the source receives the ICMP Echo Reply packet. The calculated data can reflect the network operating status.

Figure 5-6 ICMP test scenario



The ICMP test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.3.7 LSP Jitter Test

An NQA LSP jitter test measures the jitter, delay, and packet loss ratio on LDP LSPs or TE LSPs based on the timestamps in test packets.

Figure 5-7 shows the process of an LSP jitter test:

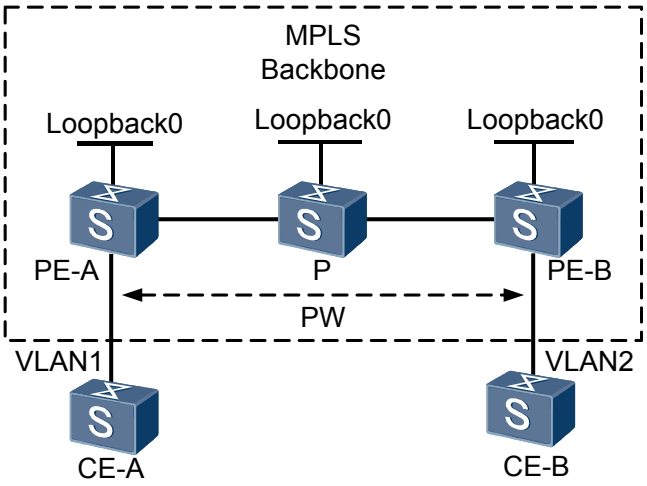
1. The source constructs a UDP MPLS Echo Request packet and fills in the destination IP field with an IP address on network segment 127.0.0.0/8. The source then searches for the

corresponding LSP and forwards the packet through the LSP in the MPLS domain at a certain interval. If a matching TE LSP is found, the packet can be sent from a tunnel interface and then forwarded along a specified CR-LSP.

2. The destination monitors port 3503, adds a timestamp to each received packet, and sends an MPLS Echo Reply packet to the source.
3. After receiving the MPLS Echo Reply packet, the source calculates the jitter by subtracting the interval at which the source sends two consecutive packets from the interval at which the destination receives the two consecutive packets.

   The source can also calculate the maximum, minimum, and average jitter time in the transmission of packets from the source to the destination. This data can reflect the network operating status.

Figure 5-7 LSP jitter test scenario



The LSP jitter test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.3.8 LSP Ping Test

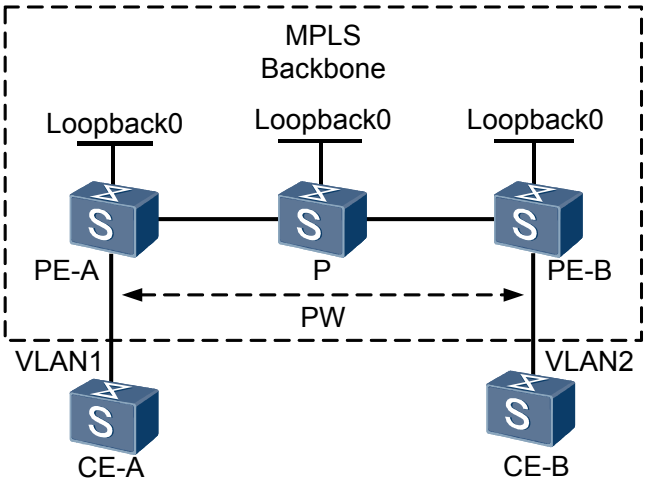
An NQA LSP ping test checks the reachability of LDP LSPs or TE LSPs.

Figure 5-8 shows the process of an LSP ping test:

1. The source constructs a UDP MPLS Echo Request packet and fills in the destination IP field with an IP address on network segment 127.0.0.0/8. The source then searches for the corresponding LDP LSP based on the configured remote LSR ID and forwards the packet through the LDP LSP in the MPLS domain. For a TE LSP, the packet can be sent from a tunnel interface and then forwarded along a specified CR-LSP.
2. The destination monitors port 3503 and sends an MPLS Echo Reply packet to the source.

   After receiving the MPLS Echo Reply packet, the source calculates the time taken for communication between the source and the destination by subtracting the time the source sends the MPLS Echo Request packet from the time the source receives the MPLS Echo Reply packet. This data can reflect the MPLS network operating status.

Figure 5-8 LSP ping test scenario



The LSP ping test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.3.9 LSP Trace Test

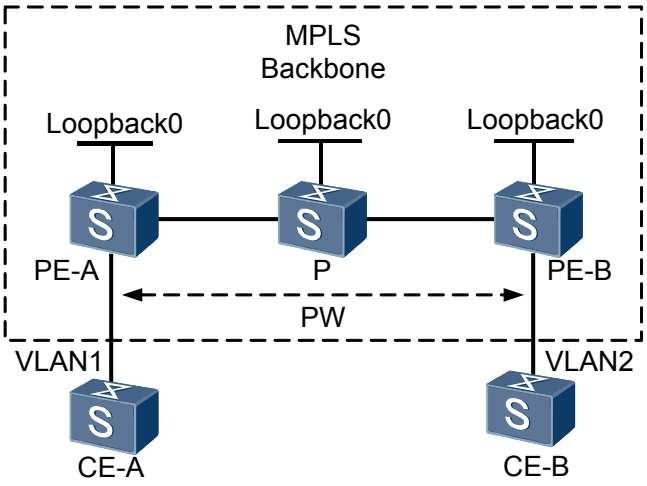
An NQA LSP trace test detects the forwarding paths of LDP LSPs or TE LSPs and collects statistics about each device along a forwarding path.

Figure 5-9 shows the process of an LSP trace test:

1. The source constructs a UDP MPLS Echo Request packet and fills in the destination IP field with an IP address on network segment 127.0.0.0/8. The source then searches for the corresponding LSP. For a TE LSP, the packet can be sent from a tunnel interface and then forwarded along a specified CR-LSP. The MPLS Echo Request packet should contain the downstream mapping TLV that carries LSP downstream information on the current node, including next-hop IP address and outbound label. The TTL of the first sent MPLS Echo Request packet is 1.
2. The MPLS Echo Request packet is forwarded through the specified LSP in the MPLS domain. After the packet reaches the first hop of the LSP, the TTL decreases to 0 and times out. The firs hop then returns an MPLS Echo Reply packet.
3. The source continues to send an Echo Request packet, with the TTL increasing by 1, This process is repeated until all the LSRs along the LSP return their responses. Then the traceroute process ends.

   According to the MPLS Echo Reply packet received from each hop, the source obtains the LSP forwarding path from the source to the destination and collects statistics about each device along the forwarding path. These statistics can reflect the LSP status.

Figure 5-9 LSP trace test scenario



The LSP trace test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.3.10 MAC Ping Test

An NQA MAC ping test is a detection tool provided by Ethernet OAM and is implemented based on 802.1ag. A MAC ping test is initiated by an MEP and is performed between the MEP and MP in the same MA. The destination is an MEP or MIP of the same level as the MEP initiating the MAC ping test in the same or different MAs.

Figure 5-10 shows the process of initiating an 802.1ag MAC ping test from MEP1 to MEP2:

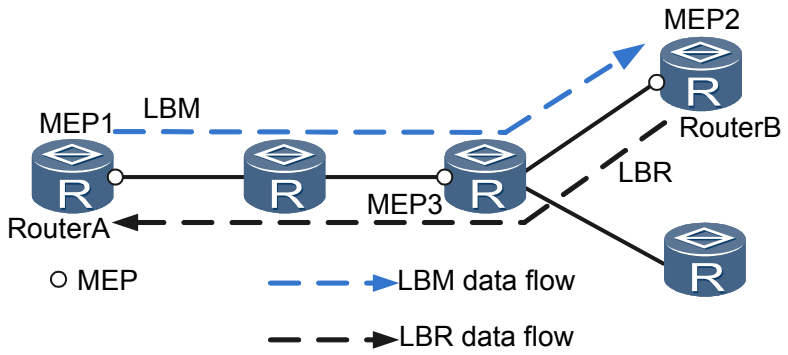
1. MEP1 sends a Loopback Message (LBM) to MEP2.
2. After receiving the LBM, MEP2 responds with a Loopback Reply (LBR). MEP1 calculates the time taken to perform the ping operation to analyze network performance.

   Within a specified timeout period:

   - If MEP1 does not receive the LBR message from MEP2, MEP1 considers the link between itself and MEP2 unreachable.
   - If MEP1 receives the LBR message from MEP2, MEP1 calculates the transmission delay from MEP1 to MEP2 based on the timestamp carried in the message.

   During a MAC ping test, the source can send multiple LBMs continuously and then check whether LBR messages are returned. In a MAC ping test, statistics about Ethernet OAM performance, including the average delay, jitter, and packet loss ratio, can be collected based on the timestamps in the test packets. These statistics can reflect Ethernet network performance.

Figure 5-10 MAC ping test scenario



The MAC ping test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.3.11 MTrace Test

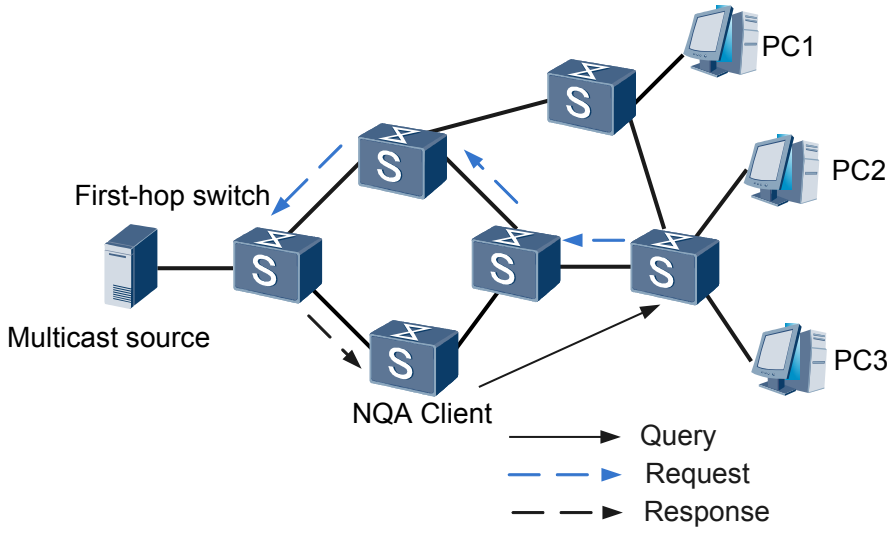
An NQA MTrace test detects the multicast forwarding path from a multicast source to a destination host and collects statistics about each device along the multicast forwarding path.

Figure 5-11 shows the process of an MTrace test:

1. The multicast querier (NQA client) sends an IGMP Tracert Query packet to the last-hop switch connected to the destination host.
2. After the first-hop switch connected to the multicast source receives the IGMP Tracert Request packet, the switch sends an IGMP Tracert Response packet to the multicast querier.

   After receiving the IGMP Tracert Response packet, the multicast querier obtains the multicast forwarding path from the multicast source to the destination host and information about each switch along the multicast forwarding path. This information can reflect the multicast forwarding path from the multicast source to the destination host.

Figure 5-11 MTrace test scenario



The MTrace test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.3.12 PWE3 Ping Test

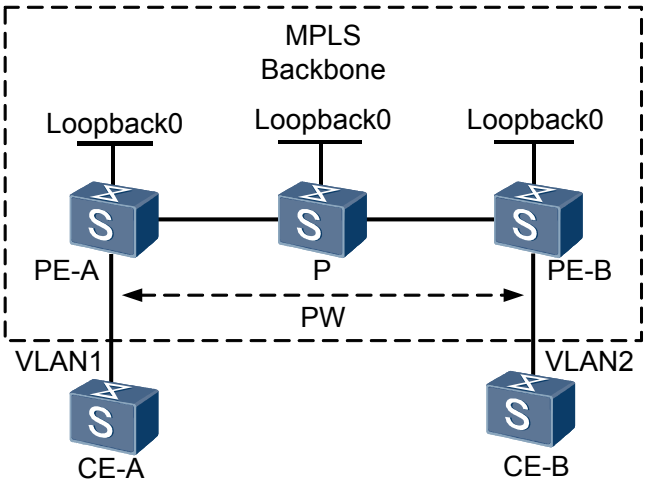
An NQA PWE3 ping test checks the reachability of MPLS-based PWs.

Figure 5-12 shows the process of a PWE3 ping test:

1. The source selects a specified PW according to the configured PW ID to send an MPLS Echo Request packet. After the packet reaches the remote PE, the remote PE responds to the source with an MPLS Echo Reply packet with the destination address as the IP address of the interface that sends the MPLS Echo Request packet.
2. The source forwards data using the PW only when it receives an MPLS Echo Reply packet from the remote PE.

   After receiving the MPLS Echo Reply packet, the source calculates the time taken for communication between the source and the destination by subtracting the time the source sends the MPLS Echo Request packet from the time the source receives the MPLS Echo Reply packet. The communication time can reflect the PW status.

Figure 5-12 PWE3 ping test scenario



The PWE3 ping test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.3.13 PWE3 Trace Test

An NQA PWE3 trace test detects the MPLS-based PW path and collects statistics about each device along the path.

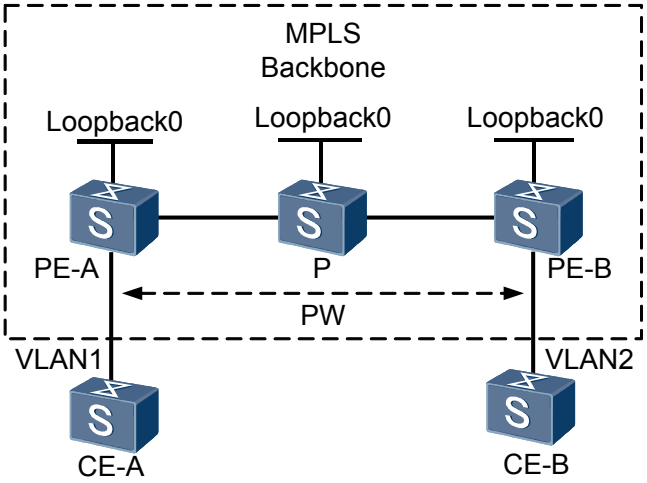
Figure 5-13 shows the process of a PWE3 trace test:

1. The source sends an MPLS Echo Request packet with the TTL as 1 through a specified PW. After the packet reaches the first-hop device on the PW, its TTL decreases to 0 and expires, and the first-hop device returns an MPLS Echo Reply packet.
2. After receiving the MPLS Echo Reply packet from the first-hop device, the source continues to send an MPLS Echo Request packet along the specified PW, with the TTL as 2. After the packet reaches the second-hop device on the PW, its TTL decreases to 0 and expires, and the second-hop device returns an MPLS Echo Reply packet.

3. The preceding process is repeated until the source collects information about each device on the PW.

   According to the MPLS Echo Reply packet received from each hop, the source obtains the PW path from the source to the destination and collects statistics about each device along the path. This can reflect the PW status.

Figure 5-13 PWE3 trace test scenario



The PWE3 trace test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.3.14 RTP Test

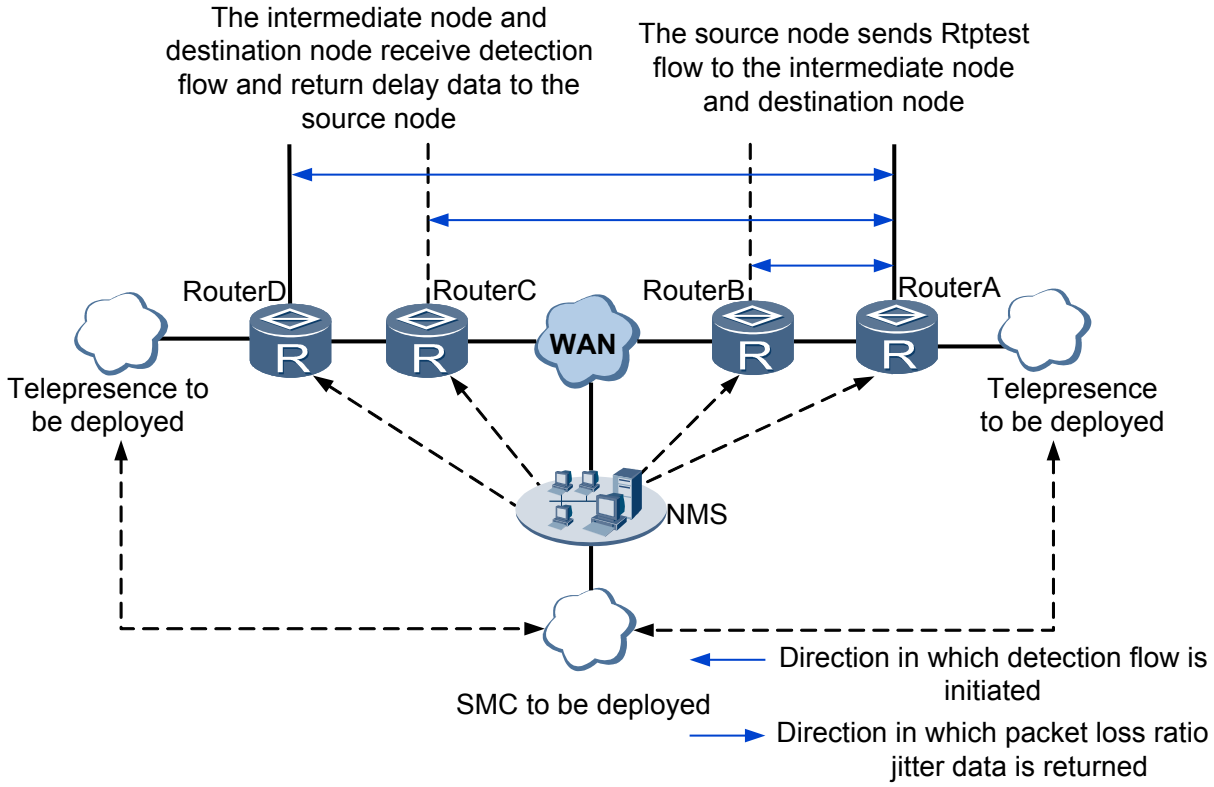
The Real-Time Transport Protocol (RTP) test includes the Rtptest test instance and Rtpsnoop test instance.

Rtptest and Rtpsnoop tests are tools for detecting the Telepresence network quality. Before deploying a Telepresence system or starting a Telepresence conference, you can use the Rtptest and Rtpsnoop test instances to detect the packet loss ratio, jitter, and delay, and use the statistics to evaluate whether the network quality meets the Telepresence service requirements. During a Telepresence conference, the Rtpsnoop test instance can be configured on an intermediate node to detect network indexes such as packet loss ratio, jitter, and DSCP to locate faults rapidly. Faults can be rectified using specified measures to ensure the Telepresence conference is working properly.

Telepresence Conference Delay Detection

Figure 5-14 shows the process of detecting the delay in a Telepresence conference.

Figure 5-14 Process of detecting the Telepresence conference delay

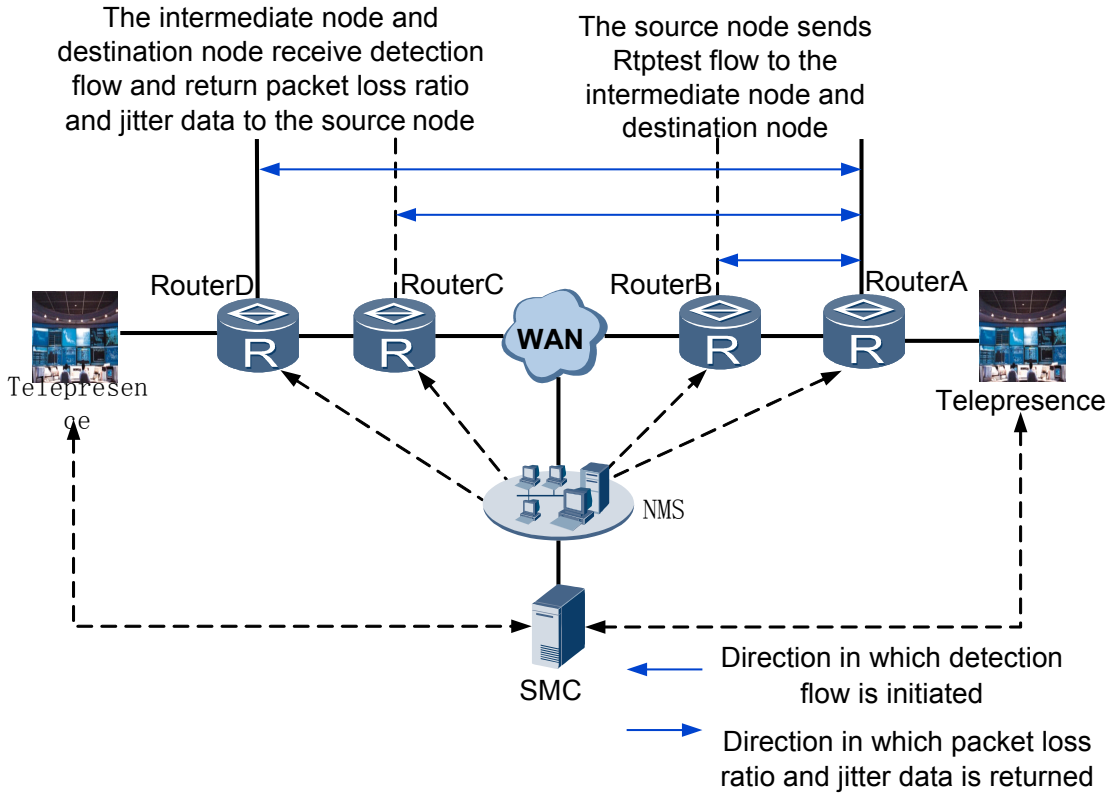


1. The tracert path from the source switch (SwitchA) to the destination switch (SwitchD) can be obtained using the NMS or command lines. SwitchA reports the tracert results to the NMS.
2. Using the NMS or command lines, the Rtptest test instance is configured on SwitchA to detect the source IP address, destination IP address, source UDP port number, destination UDP port number, and IP DSCP value of traffic.
3. Using the NMS or command lines, the intermediate switches (SwitchB and SwitchC) and destination switch (SwitchD) are configured to prepare to receive test traffic.
4. SwitchA constructs an RTP packet and starts to send test traffic.
5. The intermediate switches and destination switch receive the RTP packet and send a reply packet to SwitchA.
6. SwitchA receives the reply packet and calculates the delay in receiving the packet.
7. The NMS server sends a message to SwitchA to obtain the delay statistics. SwitchA sends the delay statistics to the NMS server.

Telepresence conference packet loss ratio and jitter detection

Figure 5-15 shows the process of detecting the packet loss ratio and jitter of a Telepresence conference.

Figure 5-15 Process of detecting the packet loss ratio and jitter of a Telepresence conference



1. The tracert path from the source switch (SwitchA) to the destination switch (SwitchD) can be obtained using the NMS or command lines. SwitchA reports the tracert results to the NMS.
2. Using the NMS or command lines, the Rtptest test instance is configured on SwitchA to detect the source IP address, destination IP address, source UDP port number, destination UDP port number, and IP DSCP value of traffic.
3. Using the NMS or command lines, the intermediate switches (SwitchB and SwitchC) and destination switch (SwitchD) are configured to detect the Rtpsnoop test instance.
4. The NMS sends a message to the intermediate switches and destination switch, starts the Rtpsnoop test instance, and notifies the switches of the snooping time.
5. The intermediate switches and destination switch calculate the packet loss ratio and jitter, and report the IP DSCP values of packets.
6. The NMS server sends a message to the intermediate switches and destination switch to obtain detection statistics and IP DSCP values of packets.
7. The intermediate switches and destination switch send statistics about the packet loss ratio, jitter, and packet priority change to the NMS. The NMS identifies links of low network quality.

5.3.15 SNMP Test

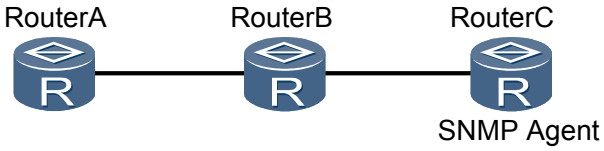
An NQA SNMP test is performed using UDP packets to measure the time taken for communication between an NQA client and an SNMP agent.

Figure 5-16 shows the process of an SNMP test:

1. The source (RouterA) sends a request packet to the SNMP agent (RouterC) to obtain the system time.
2. After receiving the request packet, the SNMP agent queries the system time, constructs a reply packet, and sends it to the source.

   After receiving the reply packet, the source calculates the time taken for communication between the source and the SNMP agent by subtracting the time the source sends the request packet from the time the source receives the reply packet. This can reflect network SNMP performance.

Figure 5-16 SNMP test scenario



The SNMP test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.3.16 TCP Test

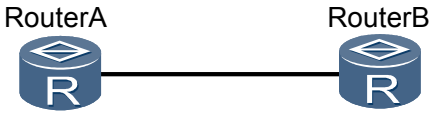
An NQA TCP test measures the time taken to set up a TCP connection between an NQA client and a TCP server through three-way handshake.

Figure 5-17 shows the process of a TCP test:

1. RouterA (NQA client) sends a TCP SYN packet to RouterB (TCP server) to set up a TCP connection.
2. After receiving the TCP SYN packet, RouterB accepts the request and responds RouterA with a TCP SYN ACK packet.
3. After receiving the SYN ACK packet, RouterA sends an ACK packet to RouterB. Subsequently, a TCP connection is successfully set up.

   Then RouterA can calculate the time taken to set up the TCP connection with RouterB by subtracting the time RouterA sends the TCP SYN packet to the time RouterA receives the TCP SYN ACK packet. This can reflect network TCP performance.

Figure 5-17 TCP test scenario



Frequent TCP tests will consume too many resources and affect device performance.

The TCP test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.3.17 Trace Test

An NQA trace test detects the forwarding path between the source and the destination and collects statistics about each device along the forwarding path. A trace test has similar functions

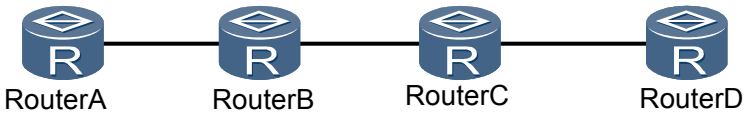
as the tracert command except that the trace test provides more output information, including the average delay, packet loss ratio, and time the last packet is received.

Figure 5-18 shows the process of a trace test:

1. The source (RouterA) constructs a UDP packet, with the TTL as 1, and sends the packet to the destination (RouterD).
2. After the first-hop router (RouterB) receives the UDP packet, it checks the TTL field and finds that the TTL decreases to 0. Then RouterB returns an ICMP Time Exceeded packet.
3. After the source receives the ICMP Time Exceeded packet, it obtains the IP address of the first-hop router and reconstructs a UDP packet, with the TTL as 2.
4. After the second-hop router (RouterC) receives the UDP packet, it checks the TTL field and finds that the TTL decreases to 0. Then RouterC returns an ICMP Time Exceeded packet.
5. The preceding process is repeated until the packet reaches the last-hop router, which then returns an ICMP Port Unreachable packet to the source.

   According to the ICMP packet received from each hop, the source obtains information about the forwarding path from the source to the destination and statistics about each device along the forwarding path. These statistics can reflect the forwarding path status.

Figure 5-18 Trace test scenario



The trace test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.3.18 UDP Test

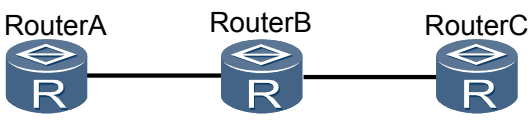
An NQA UDP test measures the time taken for communication between the source and the destination (UDP server).

Figure 5-19 shows the process of a UDP test:

1. The source (RouterA) constructs a UDP packet and sends it to the destination (RouterC).
2. After receiving the UDP packet, the destination returns the packet to the source.

   After receiving the UDP packet, the source calculates the time taken for communication between the source and the destination by subtracting the time the source sends the UDP packet from the time the source receives the UDP packet. This can reflect network UDP performance.

Figure 5-19 UDP test scenario



The UDP test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.3.19 UDP Jitter Test

A UDP Jitter test is performed using UDP packets to obtain the delay, jitter, and packet loss ratio based on the timestamp in test packets. The jitter time equals the interval for receiving two consecutive packets minus the interval for sending the two packets. **Figure 5-20** shows the process of a UDP jitter test:

1. The source (RouterA) sends packets to the destination (RouterB) at a specified interval.
2. After receiving a packet, the destination adds a timestamp to the packet and sends it back to the source.
3. After receiving the MPLS Echo Reply packet, the source calculates the jitter by subtracting the interval at which the source sends two consecutive packets from the interval at which the destination receives the two consecutive packets.

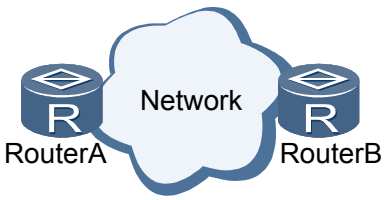
   NOTE

   In a UDP jitter test, the maximum number of test packets to be sent each time is configurable, which equals the number of jitter tests (probe-count) multiplied by the number of test packets sent each time (jitter-packetnum).

   The following data can be calculated based on information in the packets received by the source:

   - Maximum, minimum, and average jitter of the packets from the source to the destination and from the destination to the source
   - Maximum unidirectional delay from the source to the destination or from the destination to the source

Figure 5-20 UDP jitter test scenario



In a UDP jitter test, you can set the number of consecutive packets to be sent in a single test instance. This setting allows you to simulate the actual traffic of specified data within a specified period. For example, you can set the source to send 3000 UDP packets at an interval of 20 ms. Then G.711 traffic can be simulated within 1 minute.

The UDP jitter test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.3.20 UDP Jitter (Hardware-based) Test

A UDP jitter (hardware-based) test is performed using UDP packets and is a supplement to the UDP jitter. It uses the sub-core to transmit packets and add timestamps to packets. This test has the following advantages:

- Reduces the interval for sending packets. The minimum interval for sending packets can be 10 ms.
- Increases the number of concurrent test instances.

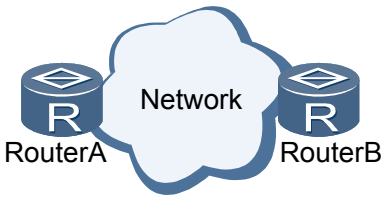
- Improves the accuracy of delay and jitter calculation.

These advantages enable the UDP jitter (hardware-based) test to accurately reflect the network status and improve device efficiency.

Table 5-1 Differences between UDP jitter and UDP jitter (hardware-based)

Comparison	UDP Jitter	UDP Jitter (Hardware-based)
Interval for sending packets	The minimum value is 20 ms.	The minimum value is 10 ms.
Jitter calculation	Timestamps are added to packets on the MPU.	Timestamps are added to packets on the LPU, which is more precise.

Figure 5-21 UDP jitter (hardware-based) test scenario

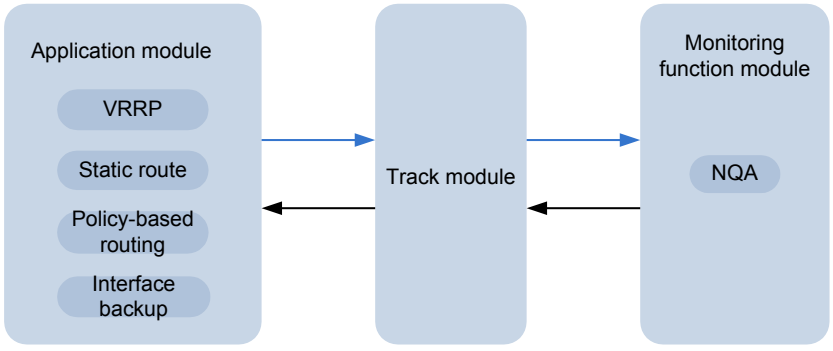


The UDP jitter (hardware-based) test results and historical records are collected in test instances. You can run commands to view the test results and historical records.

5.4 NQA Association Mechanism

The NQA association function creates association entries to monitor the probes in a test instance and to trigger other modules to respond accordingly when the number of probe failures reaches a specified value. **Figure 5-22** shows the implementation of the association function.

Figure 5-22 Implementation of the association function



The association function involves three modules: application module, track module, and monitoring function module (such as NQA module). The track module resides between the application module and NQA module. When the status of the monitored entry changes, the NQA

module reports the change to the track module, which then instructs the application module to process the change accordingly. This process implements the association function.

The following uses a static route as an example.

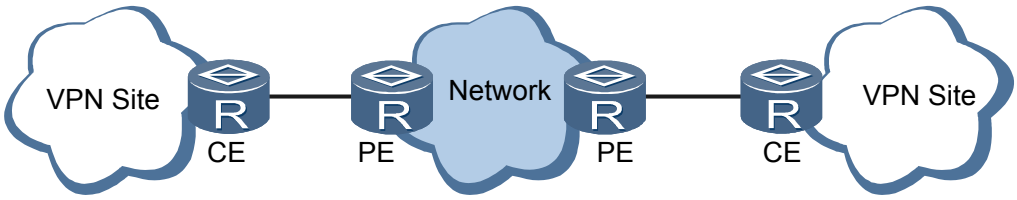
Assume that a static route with next hop 192.168.0.88 is configured. If next hop 192.168.0.88 is reachable, the static route is valid. If the next hop is unreachable, the static route is invalid. Association between the NQA module, track module, and application module can determine the validity of the static route in real time. If the NQA module finds that next hop 192.168.0.88 is unreachable, it notifies the unreachability to the static route module through the track module. The static route module then determines whether the static route is invalid.

5.5 Applications

Performing Network Diagnosis

You may often encounter such problems as intermittent network disconnections, failure to access websites, slow Internet access, and slow file downloading. When this occurs, you need to collect statistics about the device to locate the faults. These statistics need to be provided by the device.

Figure 5-23 Performing network diagnosis



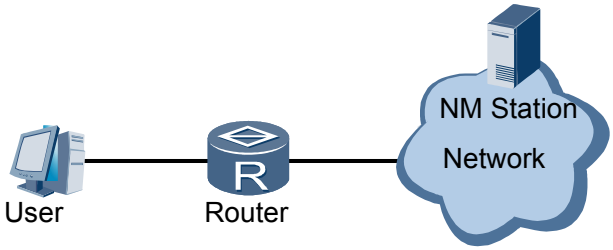
As shown in **Figure 5-23**, users in different places connect to each other over a VPN network. However, users reflect that the network is disconnected intermittently and the network connection is slow.

You can deploy NQA on PEs to analyze network quality. Perform an ICMP test between the PEs and CEs to check the continuity of the network. After confirming that the network is correctly connected, perform a jitter test to measure the network jitter. Then perform the same tests between the PEs. Analyze the test data and the faults that users encounter for fault location.

Learning About Network Service Quality

NQA helps you learn about network service quality.

Figure 5-24 Learning about network service quality



As shown in **Figure 5-24**, users You can perform an NQA test to obtain statistics about the network operating status, which helps learn about network service quality.

5.6 References

The following lists the references for NAQ.

Document	Description	Remarks
RFC 1889	RTP: A Transport Protocol for Real-Time Applications	-
RFC 2925	Definitions of Managed Objects for Remote Ping, Traceroute, and Lookup Operations	-
RFC 2131	Dynamic Host Configuration Protocol	-
RFC 1035	DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION	-
RFC 414	FILE TRANSFER PROTOCOL (FTP) STATUS AND FURTHER COMMENTS	-
RFC 1945	Hypertext Transfer Protocol – HTTP/1.0	-
RFC 2616	Hypertext Transfer Protocol - HTTP/1.1	-
RFC 792	INTERNET CONTROL MESSAGE PROTOCOL	-
RFC 4379	Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures	-
IEEE 802.1AG DRAFT6.1	IEEE 802.1AG DRAFT6.1	-
RFC 792	INTERNET CONTROL MESSAGE PROTOCOL	-
DRAFT-FENNER-TRACEROUTE-IPM-07	DRAFT-FENNER-TRACEROUTE-IPM-07	-
RFC 1157	A Simple Network Management Protocol (SNMP)	-
RFC 3414	User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)	-
RFC 1905	Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)	-
RFC793	Transmission Control Protocol	TCP/UDP test
RFC 862	Echo Protocol	TCP/UDP test
RFC 1393	Traceroute Using an IP Option	Trace test

6 LLDP

About This Chapter

6.1 Introduction to LLDP

Definition

The Link Layer Discovery Protocol (LLDP) is a standard Layer 2 topology discovery protocol defined in IEEE 802.1ab. LLDP collects local device information including the management IP address, device ID, and port ID and advertises the information to neighbors. Neighbors save the received information in their management information bases (MIBs). The network management system (NMS) can use data in MIBs to query the link status.

Purpose

An NMS must be capable of managing multiple network devices with diverse functions and complex configurations. Most NMSs can detect Layer 3 network topologies, but they cannot detect detailed Layer 2 topologies or detect configuration conflicts. A standard protocol is required to exchange Layer 2 information between network devices.

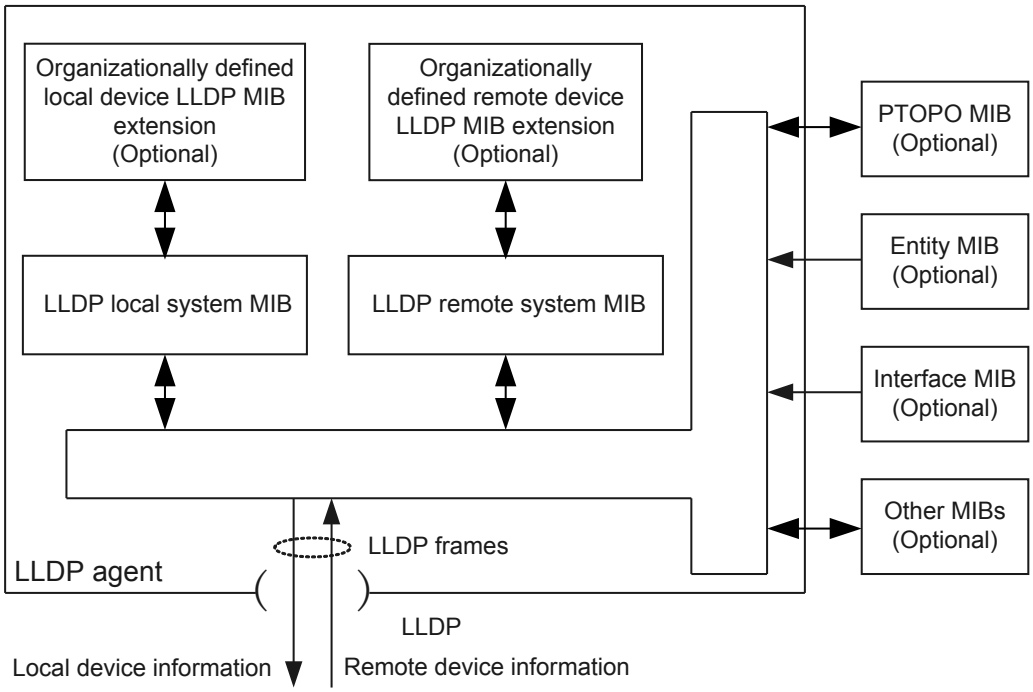
The LLDP protocol provides a standard link-layer discovery method. Layer 2 information obtained from LLDP allows the NMS to detect the topology of neighboring devices, and display paths between clients, switches, routers, application servers, and network servers. The NMS can also detect configuration conflicts between network devices and identify causes of network failures. Enterprise users can use an NMS to monitor the link status on devices running LLDP and quickly locate network faults.

6.2 Principles

6.2.1 LLDP Implementation

LLDP collects and sends local device information to neighbors, and the local device saves information received from neighbors to standard MIBs. **Figure 6-1** shows how LLDP is implemented.

Figure 6-1 LLDP block diagram



LLDP is implemented as follows:

- The LLDP module uses an LLDP agent to interact with the Physical Topology MIB, Entity MIB, Interfaces MIB, and other MIBs to update the LLDP local system MIB and LLDP local extended MIB.
- The LLDP agent encapsulates local device information in LLDP frames and sends the LLDP frames to neighbors.
- After receiving LLDP frames from neighbors, the LLDP agent updates the LLDP remote system MIB and LLDP remote extended MIB.
- By exchanging LLDP frames with neighbors, the local device can obtain information about neighbors, including remote interfaces connected to the local device and MAC addresses of neighbors.

The LLDP local system MIB stores local device information, including the device ID, port ID, system name, system description, port description, and management address.

The LLDP remote system MIB stores neighbor information, including the device ID, port ID, system name, system description, port description, and management address of each neighbor.

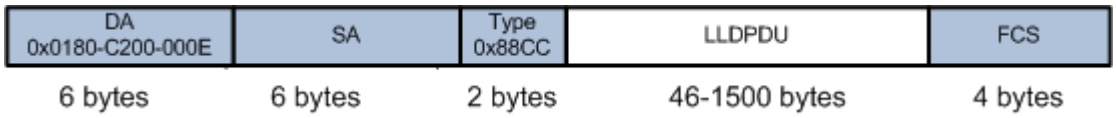
An LLDP agent performs the following tasks:

- Maintains the LLDP local system MIB and LLDP remote system MIB.
- Obtains and sends LLDP local system MIB information to neighbors when the local device status changes. An LLDP agent also obtains and sends LLDP local system MIB information to neighbors at periodic intervals if the local device status does not change.
- Identifies and processes received LLDP frames.
- Sends LLDP traps to the NMS when information in the LLDP local system MIB or LLDP remote system MIB changes.

6.2.2 LLDP Frame Format

An LLDP frame is an Ethernet frame encapsulated with an LLDP data unit (LLDPDU). **Figure 6-2** shows the LLDP frame format.

Figure 6-2 LLDP frame format



An LLDP frame contains the following fields:

- DA: destination MAC address, a fixed multicast MAC address 0x0180-C200-000E.
- SA: source MAC address, the MAC address of the sender
- Type: packet type, 0x88CC in LLDP frames.
- LLDPDU: LLDP data unit, body of an LLDP frame.
- FCS: frame check sequence.

LLDPDU

An LLDPDU contains local device information and is encapsulated in an LLDP frame. Each LLDPDU consists of several information elements known as TLVs that each includes Type, Length, and Value fields. The local device encapsulates its local information in TLVs, constructs an LLDPDU with several TLVs, and encapsulates the LLDPDU in the data field of an LLDP frame. **Figure 6-3** shows the LLDPDU structure.

Figure 6-3 LLDPDU structure

As shown in **Figure 6-3**, an LLDPDU has four mandatory TLVs: Chassis ID TLV, Port ID TLV, Time to Live TLV, and End of LLDPDU TLV. Other TLVs are optional, and a device can determine whether to encapsulate them in an LLDPDU.

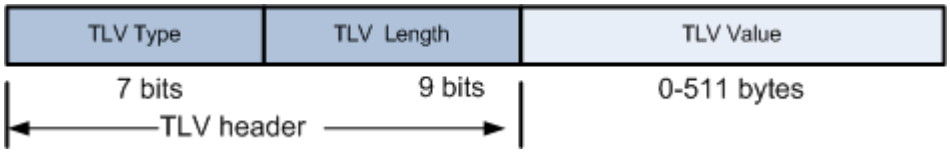
When LLDP is disabled on an interface or an interface is shut down, the interface sends a shutdown LLDPDU to the neighbors. In the shutdown LLDPDU, the value of the Time to Live TLV is 0. A shutdown LLDPDU contains no optional TLVs.

TLV Structure

An LLDPDU is formed by TLVs, and each TLV is an information element.

Figure 6-4 shows the structure of a TLV.

Figure 6-4 TLV structure



A TLV contains the following fields:

- TLV Type (7 bits): type of a TLV. Each TLV type has a unique value. For example, the value of End of LLDPDU TLV is 0, and the value of Chassis ID TLV is 1.
- TLV Length (9 bits): size of a TLV.
- TLV Value (0-511 bytes): The first bit indicates the sub-type of a TLV, and the other bits are the TLV content.

TLV Type

LLDPDUs can encapsulate basic TLVs, TLVs defined by IEEE 802.1 working groups, TLVs defined by IEEE 802.3 working groups, and Media Endpoint Discovery (MED) TLVs. Basic TLVs are used for basic device management functions. The TLVs defined by IEEE 802.1 and IEEE 802.3 working groups, and MED TLVs defined by other organizations are used for enhanced device management functions. A device determines whether to encapsulate organizationally specific TLVs.

- Basic TLVs

  Four basic TLVs are mandatory in LLDP implementation and must be encapsulated in an LLDPDU.

  Table 6-1 Basic TLVs

TLV	Description	Mandatory
Chassis ID TLV	Bridge MAC address of the device sending an LLDPDU.	Yes
Port ID TLV	Port from which an LLDPDU is sent. ● If an LLDPDU does not contain any MED TLVs, the Port ID TLV identifies the port name. ● If an LLDPDU contains a MED TLV, the Port ID TLV identifies the port MAC address. If the port has no MAC address, the Port ID TLV identifies the bridge MAC address.	Yes
Time To Live TLV	Time to live (TTL) of the local device information stored on the neighbor device.	Yes
End of LLDPDU TLV	End of an LLDPDU.	Yes
Port Description TLV	Character string that describes the port sending an LLDPDU.	No
System Name TLV	System name.	No
System Description TLV	Character string that describes the system.	No

TLV	Description	Mandatory
System Capabilities TLV	Main functions of the system and the functions that have been enabled.	No
Management Address TLV	Address used by the NMS to identify and manage the local device. Management IP addresses uniquely identify network devices, facilitating layout of the network topology and network management.	No

- TLVs defined by the IEEE 802.1 working group

  Table 6-2 TLVs defined by the IEEE 802.1 working group

TLV	Description
Port VLAN ID TLV	VLAN ID of a port.
Port And Protocol VLAN ID TLV	Protocol VLAN ID of a port.
VLAN Name TLV	Name of the VLAN on a port.
Protocol Identity TLV	Protocol types that a port supports.

- TLVs defined by the IEEE 802.3 working group

  Table 6-3 TLVs defined by the IEEE 802.3 working group

TLV	Description
EEE TLV	Whether a port supports Energy-Efficient Ethernet (EEE).
Link Aggregation TLV	Whether a port supports link aggregation and has link aggregation enabled.
MAC/PHY Configuration/ Status TLV	Rate and duplex mode of a port, whether the port supports auto-negotiation, and whether auto-negotiation is enabled on the port.
Maximum Frame Size TLV	Maximum frame length that a port supports. The value is the maximum transmission unit (MTU) of the port.
Power Via MDI TLV	Power capabilities of a port, for example, whether a port supports PoE and whether a port supplies or demands power.

- MED TLVs

  MED TLVs are related to voice over IP (VoIP) applications and provide functions such as basic configuration, network policy configuration, address management, and directory

management. These TLVs meet the requirements of voice device manufacturers for cost efficiency, easy deployment, and easy management. Use of these TLVs allows the deployment of voice devices on Ethernet network. This brings great convenience for manufacturers, sellers, and users of voice devices.

Table 6-4 LLDP-MED TLVs

TLV	Description
LLDP-MED Capabilities TLV	Type of a device and types of LLDP-MED TLVs that can be encapsulated in an LLDPDU.
Inventory TLV	Manufacturer of the device.
Location Identification TLV	Location of the local device.
Network Policy TLV	VLAN ID, Layer 2 priority, and DSCP of a voice VLAN.
Extended Power-via-MDI TLV	Power capability of the system.
Hardware Revision TLV	Hardware version of a media endpoint (ME). This TLV can only be queried on the local device and cannot be sent to neighbor devices.
Firmware Revision TLV	Firmware version of an ME device. This TLV can only be queried on the local device and cannot be sent to neighbor devices.
Software Revision TLV	Software version of an ME device. This TLV can only be queried on the local device and cannot be sent to neighbor devices.
Serial Number TLV	Serial number of an ME device. This TLV can only be queried on the local device and cannot be sent to neighbor devices.
Model Name TLV	Model name of an ME device. This TLV can only be queried on the local device and cannot be sent to neighbor devices.
Asset ID TLV	Asset identifier for directory management and assertion tracing of an ME device. This TLV can only be queried on the local device and cannot be sent to neighbor devices.

- DCBX TLV

  Data center bridging (DCB) information. Neighboring nodes on data center networks use the Data Center Bridging Exchange (DCBX) protocol to exchange and negotiate DCB information so that they have the same DCB information. This prevents packet loss on the data center network. DCBX encapsulates DCB information in DCBX TLVs and uses LLDP frames to exchange DCB information between neighboring nodes.

6.2.3 LLDP Working Modes

LLDP works in either TxRx or Disable mode.

TxRx: Able to Send and Receive LLDP Frames

LLDP frame transmission

- After LLDP is enabled on a device, the device periodically sends LLDP frames to neighbors. When the local configuration changes, the device sends LLDP frames to notify neighbors of the changes. To reduce the number of LLDP frames sent when the local information changes frequently, the device waits for a period before sending the next LLDP frame.
- The device starts fast transmission of LLDP frames in the following scenarios: when it receives an LLDP frame with device information not in its MIB (a new neighbor is discovered), when LLDP is enabled, or a when port transitions from Down to Up state. When fast transmission starts, the local device sends LLDP frames at 1-second intervals. After a specified number of LLDP frames have been sent at this interval, the device reverts the previous transmission interval.

LLDP frame reception

An LLDP-capable device checks the validity of received LLDP frames and the TLVs in those frames. When determining that an LLDP frame and its TLVs are valid, the local device saves neighbor information and sets the aging time of neighbor information on the local device to the TTL value carried in the received LLDPDU. If the TTL value carried in the received LLDPDU is 0, the neighbor information ages out immediately.

Disable: Unable to Send or Receive LLDP Frames

When LLDP is disabled, the device does not send or receive LLDP frames.

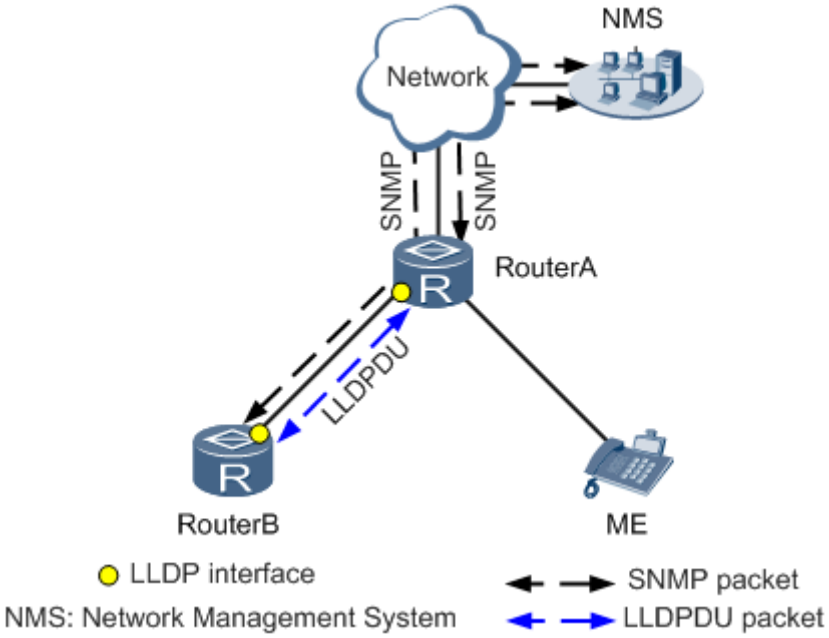
6.2.4 LLDP Networking

LLDP has the following networking modes:

- Single-neighbor networking

  In this networking, interfaces between two routers or interfaces between a router and a media endpoint (ME) are directly connected, and each interface has only one neighbor. As shown in **Figure 6-5**, RouterA is directly connected to RouterB and ME. Each interface on RouterA and RouterB has only one neighbor.

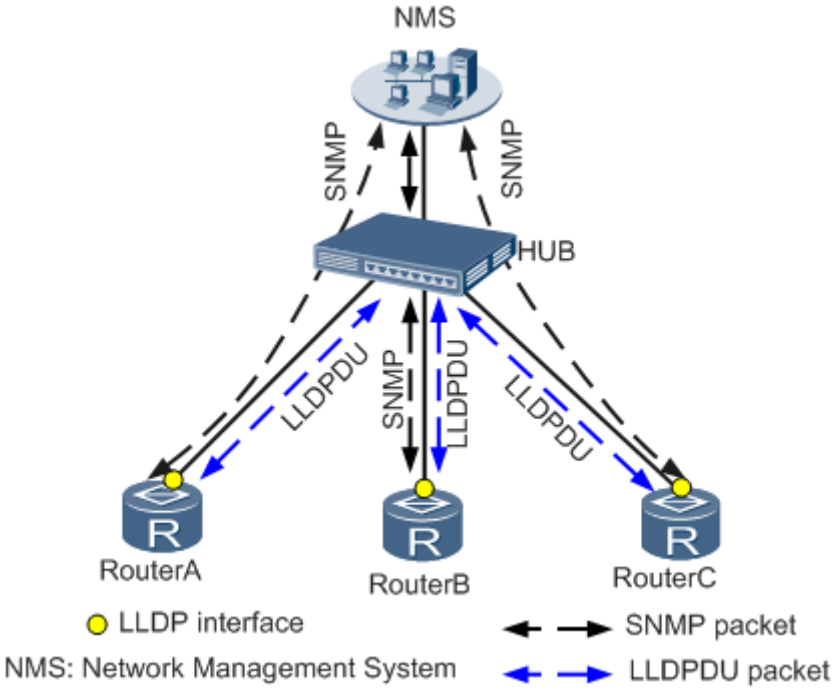
Figure 6-5 Single-neighbor networking



- Multi-neighbor networking

  Interfaces between routers are not directly connected, and each interface has more than one neighbor. As shown in **Figure 6-6**, RouterA, RouterB, and RouterC are connected through a hub. Interfaces on the routers each have more than one neighbor.

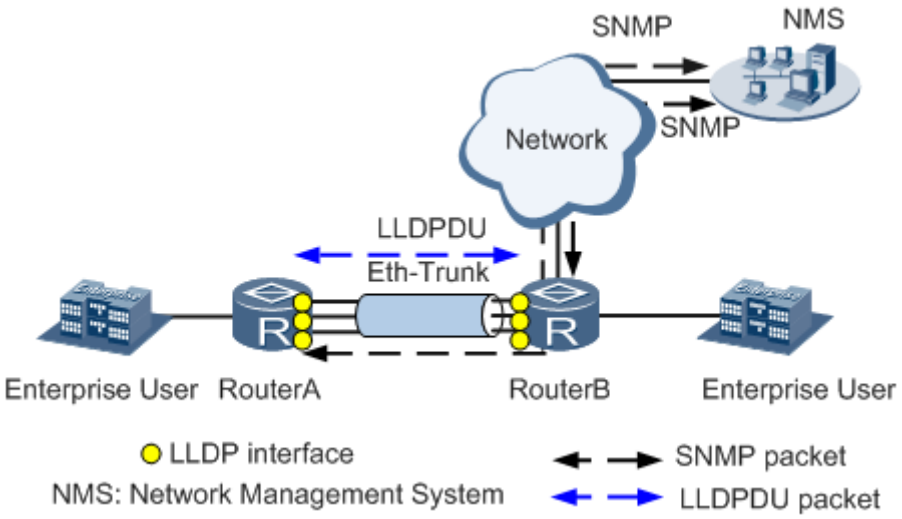
Figure 6-6 Multi-neighbor networking



- Link aggregation networking

  Interfaces between routers are directly connected and bundled into a link aggregation group. Each interface in a link aggregation group has only one neighbor. As shown in **Figure 6-7**, the interfaces connecting RouterA and RouterB are bundled into a link aggregation group, and each interface has only one neighbor.

  Figure 6-7 Link aggregation networking



6.3 References

The following table lists the references for this document.

Document	Description	Remarks
IEEE 802.1ab	IEEE Standard for Local and Metropolitan Area Networks: Station and Media Access Control Connectivity Discovery	-
IEEE 802.3at	802.3at Data Terminal Equipment(DTE) Power via the Media Dependent Interface(MDI) Enhancements	-

7 NetStream

About This Chapter

7.1 Introduction to NetStream

Definition

NetStream is a traffic statistics and analysis technology.

Purpose

The Internet provides users with high bandwidth and supports more services and applications. Enterprises require fine-grained management and accounting, which poses higher requirements on traffic statistics and analysis. Traditional traffic statistics technologies such as SNMP and port mirroring cannot meet these requirements because of their limitations (see **Table 7-1**). A new technology is required to better support network traffic statistics.

NetStream has been developed to address this problem. NetStream collects classified statistics about service traffic and resource usage, and sends the statistics to a dedicated server or a network management system (NMS) that has NetStream software installed for further analysis.

Table 7-1 Implementation and limitations of the traditional traffic statistics methods

Traffic Statistics Method	Implementation	Limitation
Statistics based on IP packets	Saves counter indexes in the routing table on a device to count the number of bytes and packets that pass through the device.	This method applies to collection of statistics about simple information instead of various information.
Statistics based on access control lists (ACLs)	Precisely matches flows based on ACLs and then collects statistics.	This method requires large capacity of ACLs and cannot collect statistics about flows that match no ACL rule.
Statistics using SNMP	Uses SNMP to implement simple statistics functions, such as interface statistics, IP packet statistics, and the ACL matching statistics.	The statistics function is not strong enough and collects statistics from the NMS using continuous polling, wasting CPU and network resources.
Statistics based on port mirroring	Duplicates traffic passing through a port and sends the duplicated traffic to a dedicated server for statistics and analysis.	This method requires high costs because a dedicated server is required to collect statistics. In addition, this method occupies a port. Statistics cannot be collected on a port that does not support port mirroring.

Traffic Statistics Method	Implementation	Limitation
Statistics based on the traffic duplication at the physical layer	Duplicates traffic using an optical splitter or other devices at the physical layer and then sends the duplicated traffic to a dedicated server for statistics.	This method requires high costs because a dedicated server and dedicated hardware devices must be purchased.

Benefits

- Accounting

  NetStream provides detailed data for accounting based on resource usage (such as usage of links, bandwidths, and time segments). The data includes the number of packets, number of bytes, IP addresses, time, types of service (ToSs), and application types. An enterprise can calculate expenses of each department and distribute operation costs based on the data to effectively use resources.
- Network monitoring

  NetStream monitors network traffic almost in real time. NetStream can be deployed on an interface connected to the Internet to monitor outgoing traffic almost in real time and analyze bandwidth usage of services. The traffic monitoring information helps network administrators determine the network running status and discover inappropriate network structures or performance bottlenecks on networks. Enterprises can easily plan and allocate network resources.
- User monitoring and analysis

  NetStream allows network administrators to obtain network resource usage of users so that they can efficiently plan and allocate network resources and ensure network running security.

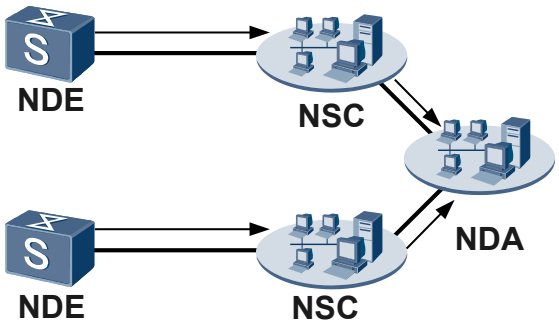
7.2 Principles

7.2.1 Basic Principles of NetStream

Components of a NetStream System

As shown in **Figure 7-1**, three roles are involved in a NetStream system: NetStream data exporter (NDE), NetStream collector (NSC), and NetStream data analyzer (NDA).

Figure 7-1 Networking diagram of a NetStream system



- NDE

  An NDE analyzes and processes network flows, extracts flows that meet conditions for statistics, and exports the statistics to the NSC. The NDE can perform operations (such as aggregation) over the statistics before exporting them to the NSC. A device configured with NetStream functions as the NDE in a NetStream system.

- NSC

  An NSC is a program running on the Unix or Windows operating system. The NSC parses packets from the NDE and saves statistics to the database. The NSC can collect data exported from multiple NDEs, and filter and aggregate the data.

- NDA

  An NDA is a traffic analysis tool. It extracts statistics from the NSC, processes the statistics, and generates a report. This report provides a basis for services such as traffic accounting, network planning, and attack monitoring. The NDA provides a graphical user interface (GUI) for users to easily obtain, check, and analyze the collected data.

NOTE

In practice, the NSC and NDA are integrated on a NetStream server.

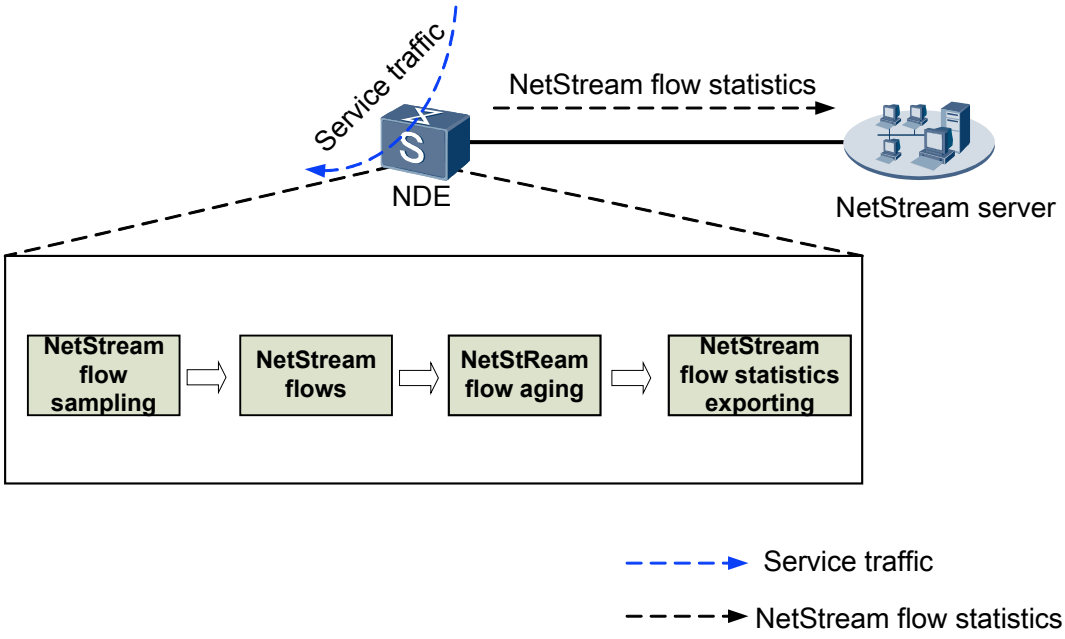
NetStream Working Mechanism

A NetStream system works as follows:

1. An NDE periodically exports detailed data about flows to an NSC.
2. The NSC processes the data and sends it to an NDA.
3. The NDA analyzes the data for applications such as accounting and network planning.

In most cases, datacom products function as NDEs in a NetStream system. This document mainly describes NDE implementation.

Figure 7-2 Diagram for implementing NetStream



As shown in **Figure 7-2**, an NDE is properly forwarding service traffic. The NetStream module on the NDE samples packets (see **NetStream Packet Sampling**), creates a flow based on the collected data (see **NetStream Flows**), ages out the flow (see **NetStream Flow Aging**), and exports the flow statistics (see **NetStream Flow Statistics Exporting**). In this manner, the NDE periodically exports detailed data about flows to the NSC.

7.2.2 NetStream Packet Sampling

Incoming traffic and outgoing traffic are sampled for statistics. You can set an interval for sampling packets so that only statistics about sampled packets are collected. The statistics show the flow status on the entire network. The sampling function reduces NetStream impact on device performance.

The following sampling modes are available:

- Packet-based random sampling

  The NDE randomly samples a packet from a specified number of packets transmitted. For example, if the number of packets is set to 100, the NDE randomly samples a packet from every 100 packets. This mode applies to sampling regular traffic.

- Packet-based regular sampling

  The NDE samples a packet every time when a specified number of packets are transmitted. For example, if the number of packets is set to 100, the NDE samples a packet after every 100 packets are transmitted. If the NDE samples the fifth packet at the first time, the NDE samples the one hundred and fifth packet, the two hundred and fifth packet, and so on. This mode applies to network traffic accounting.

- Time-based random sampling

  The NDE randomly samples a packet in a specified interval. For example, if the interval is set to 100, the NDE randomly samples a packet in every 100 ms. This mode applies to sampling regular traffic.

- Time-based regular sampling

  The NDE samples a packet at a specified interval. For example, the interval is set to 100. If the NetStream module samples a packet at the fifth second at the first time, the NDE samples a packet at the one hundred and fifth second, the two hundred and fifth second, and so on. This mode applies to networks with a large volume of traffic.

7.2.3 NetStream Flows

NetStream provides packet statistics based on flows. NetStream supports statistics about IP packets (including UDP, TCP, and ICMP packets) and MPLS packets.

- For IPv4 packets, IPv4 NetStream defines a flow based on the destination IP address, source IP address, destination port number, source port number, protocol number, ToS, and inbound or outbound interface. Packets with the same 7-tuple information are marked as one flow.
- For IPv6 packets, IPv6 NetStream defines a flow based on the destination IP address, source IP address, destination port number, source port number, protocol number, traffic class, flow label, and inbound or outbound interface. Packets with the same 8-tuple information are marked as one flow.
- For MPLS packets, the NDE collects statistics about IPv4/IPv6 information contained in the packets. If statistics about IP information are collected, the NetStream defines a flow based on the MPLS label stack and IP information.

7.2.4 NetStream Flow Aging

NetStream flow aging is the prerequisite for exporting flow statistics to the NSC. After NetStream is enabled on a device, flow statistics are stored in the NetStream cache on the device. When a NetStream flow is aged out, the NDE exports the flow statistics in the cache to the NSC using NetStream packets of a specified version.

NetStream flows are aged out in the following modes:

- Regular aging
  - Active aging

    Packets are added to a flow continuously in a specified period since the first packet is added to the flow. After the active aging timer expires, the flow statistics are exported. Active aging enables the NDE to periodically export the statistics about the flows that last for a long period.
  - Inactive aging

    If no packet is added to a flow in a specified period after the last packet is added to the flow, the NDE exports flow statistics to the NetStream server. Inactive aging clears unnecessary entries in the NetStream cache so that the system can fully leverage statistical entries. Inactive aging enables the NDE to export the statistics about flows that last for a short period. Once adding packets to a flow stops, the NDE exports the flow statistics to save memory space.
- FIN- or RST-based aging

  The FIN or RST flag in a TCP packet indicates that a TCP connection is terminated. When receiving a packet with the FIN or RST flag, the NDE immediately ages the corresponding NetStream flow.

- Byte-based aging

  The number of bytes is recorded for each flow in the NetStream cache. When the number of bytes of a flow exceeds the specified upper limit, the flow overflows. Therefore, when finding that the number of bytes of a flow exceeds the specified upper limit, the NDE immediately ages the flow to prevent a byte counting error. The hardware byte counter is a 64–bit counter, and the upper limit for bytes is 4294967295 bytes (about 3.9 GB).

- Forced aging

  You can run commands to forcibly age all flows in the NetStream cache.

  Forced aging is used when existing flows do not meet aging conditions but the latest statistics are required or when some flows fail to be aged out due to abnormal NetStream services.

7.2.5 NetStream Flow Statistics Exporting

After aging flows in the NetStream cache, the NDE exports the flow statistics to a specified NSC for further analysis. Original, aggregation, and flexible flow statistics are exported as packets of V5, V8, or V9.

Flow Statistics Exporting Modes

Original flow statistics exporting

In original flow statistics exporting mode, the NDE collects statistics about all flows. After the aging timer expires, the NDE exports statistics about each flow to the NetStream server.

This mode enables the NetStream server to obtain detailed statistics about each flow. However, this mode increases the network bandwidth and CPU usage. In addition, these statistics occupy much memory space of the NDE, which increases the cost.

Aggregation flow statistics exporting

The NDE aggregates flow statistics with the same aggregation entry values and exports the aggregation flow statistics to a specified NetStream server. This mode greatly saves network bandwidth. The NDE supports the aggregation modes described in **Table 7-2**.

For example, there are four original TCP flows. They have the same source port number, destination port number, and destination IP address, but different source IP addresses. The **protocol-port** mode is used. Aggregation entries in this mode include protocol number, source port number, and destination port number. The four TCP flows have the same protocol number, source port number, and destination port number, so only one aggregation flow statistical record is recorded in the aggregation flow statistics table.

Table 7-2 Aggregation modes

Aggregation Mode	Aggregation Entries
as	Source AS number, destination AS number, index of the inbound interface, and index of the outbound interface
as-tos	Source AS number, destination AS number, inbound interface index, outbound interface index, and ToS
protocol-port	Protocol number, source port number, and destination port number

Aggregation Mode	Aggregation Entries
protocol-port-tos	Protocol number, source port number, destination port number, ToS, inbound interface index, and outbound interface index
source-prefix	Source AS number, source mask length, source prefix, and inbound interface index
source-prefix-tos	Source AS number, source mask length, source prefix, ToS, and inbound interface index
destination-prefix	Destination AS number, destination mask length, destination prefix, and outbound interface index
destination-prefix-tos	Destination AS number, destination mask length, destination prefix, ToS, and outbound interface index
prefix	Source AS number, destination AS number, source mask length, destination mask length, source prefix, destination prefix, inbound interface index, and outbound interface index
prefix-tos	Source AS number, destination AS number, source mask length, destination mask length, source prefix, destination prefix, ToS, inbound interface index, and outbound interface index
mpls-label	Label value (a maximum of four layers)

Flexible flow statistics exporting

Flexible flows are created based on customized configuration. Users can collect flow statistics based on the protocol type, DSCP field, source IP address, destination IP address, source port number, destination port number, or flow label as required. The NDE exports the flow statistics to the NetStream server. Compared to original flow statistics exporting, flexible flow statistics exporting occupies less traffic and provides users with a flexible way to collect NetStream statistics.

Versions of Exported Packets

At present, the versions of NetStream exported packets are V5, V8, and V9. Other versions are in the experimental stage and have not been put to commercial use. NetStream exported packets of all the versions are transmitted using UDP.

- V5: The packet format is fixed. NetStream packets in this format contain the original flow statistics collected based on 7-tuple information.
- V8: The packet format is fixed. NetStream packets in this version support the aggregation exporting format.
- V9: The NetStream packet format is defined in profiles. Statistical items can be combined, and therefore statistics are exported more flexibly. V9 supports the exporting of BGP next hop information and MPLS statistics.

Mapping Between Flow Statistics Exporting Modes and Packet Versions

Statistics about a NetStream flow are exported based on a specified flow statistics exporting mode and a specified packet version. Each flow statistics exporting mode maps a packet version, as shown in **Table 7-3**.

Table 7-3 Mapping between flow statistics exporting modes and packet versions

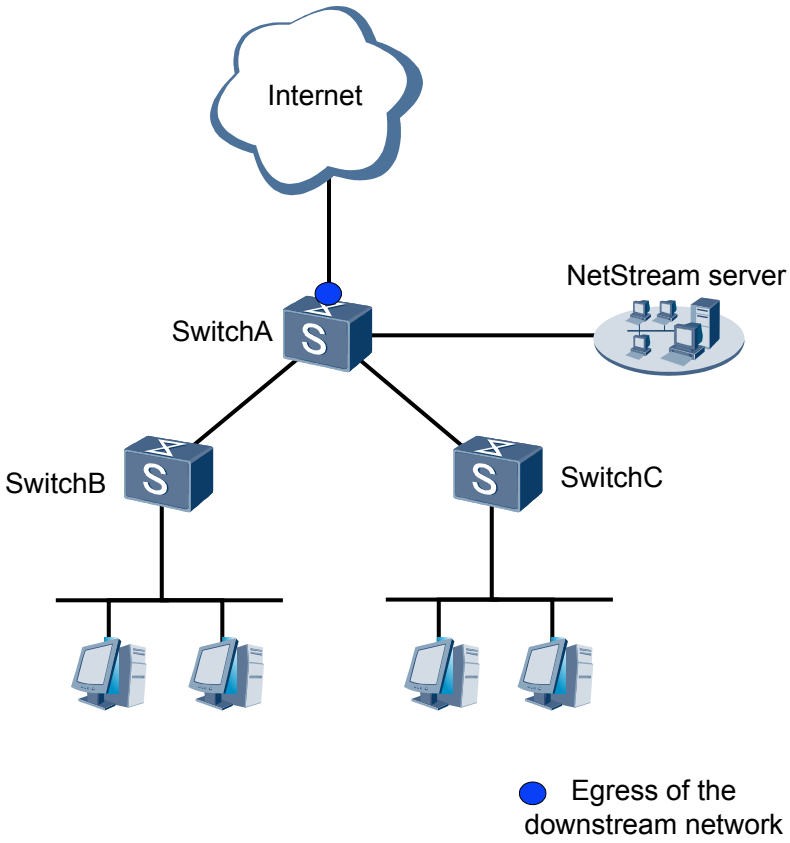
Flow Statistics Exporting Mode	Packet Version
Original flow statistics exporting	V5 and V9 By default, the version of exported packets carrying IPv4 flow statistics is V5 and the version of exported packets carrying IPv6 flow statistics is V9. To export packets carrying MPLS flow statistics, set the version to V9.
Aggregation flow statistics exporting	V8 and V9 By default, V8 supports exported packets carrying IPv4 aggregation flow statistics and V9 supports exported packets carrying MPLS aggregation flow statistics.
Flexible flow statistics exporting	V9

7.3 Applications

Network Monitoring

On a network shown in **Figure 7-3**, SwitchA connects the downstream network to the Internet. A large number of communication packets are stored on SwitchA. Network administrators want to know the bandwidths occupied by services. The NetStream function needs to be configured on SwitchA to monitor real-time traffic on the interface connecting to the Internet. The traffic monitoring information helps network administrators determine the network running status and discover inappropriate network structures or performance bottlenecks on networks.

Figure 7-3 Networking diagram of NetStream



7.4 References

The following table lists the references of this document.

Document	Description	Remarks
RFC 3917	Requirements for IP Flow Information Export (IPFIX)	-
RFC 3954	Cisco Systems NetFlow Services Export Version 9	-

8 sFlow

About This Chapter

8.1 Introduction to sFlow

Definition

Sampled Flow (sFlow) is a traffic monitoring technology that collects and analyzes traffic statistics.

Purpose

Compared with carrier networks, enterprise networks have a smaller scale, provide flexible networking, and are prone to attacks. Due to these characteristics, enterprise networks often encounter service exceptions. Enterprise users require a traffic monitoring technique on interfaces of devices to locate unexpected traffic and the source of attack traffic in a timely manner so that they can quickly rectify faults to ensure stable running of the network.

sFlow is developed to achieve the preceding purpose. sFlow provides interface-based traffic analysis and displays traffic statistics in graphs or reports, facilitating preventive maintenance especially on enterprise networks without specialized network administrators.

NetStream is a technology that collects and analyzes statistics on network flows. Network devices need to preliminarily collect and analyze network flows, and store statistics in the cache. When the cache overflows or flow statistics expire, the statistics are exported. Compared with NetStream, sFlow does not require a cache, network devices only sample packets, and a remote collector collects and analyzes traffic statistics. Therefore, sFlow has the following advantages over NetStream:

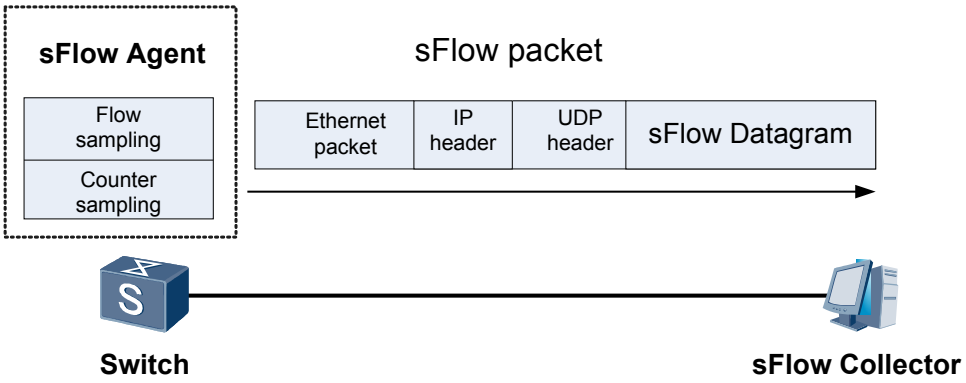
- Saves resources and lowers costs. No cache is required, and a small number of network devices are used, which lower costs.
- Flexible collector deployment. A collector collects and analyzes traffic statistics based on various traffic characteristics as required. The collector is deployed flexibly.

8.2 Principles

Architecture of an sFlow System

As shown in **Figure 8-1**, the sFlow system involves an sFlow agent embedded in the device and a remote sFlow collector. The sFlow agent obtains traffic statistics from an sFlow-enabled interface using sFlow sampling and encapsulates them into sFlow packets. When an sFlow packet buffer overflows or an sFlow packet expires, the sFlow agent sends the sFlow packets to the sFlow collector. The sFlow collector analyzes the sFlow packets and displays the traffic statistics in a report.

Figure 8-1 sFlow system



sFlow Sampling

An sFlow agent provides two sampling modes: flow sampling and counter sampling.

Flow sampling

In flow sampling, an sFlow agent samples packets in one direction or both directions on an interface based on the sampling ratio, and parses the packets to obtain information about packet data content. **Table 8-1** lists the main fields in flow sampling packets. Flow sampling focuses on traffic details to monitor and parse traffic behaviors on the network.

Flow sampling samples packets on an interface, and currently supports only random sampling. In random sampling mode, the sFlow agent allocates a random value to each packet processed by an interface. The random value ranges from 0 to N. The threshold is set to n ranging from 0 to N. When the random value is smaller than the threshold, the sFlow agent samples packets. The actual sampling ratio is n/(N+1).

Table 8-1 Main fields in flow sampling packets

Field	Description
Raw packet	Records the entire packet or part of the packet header, encapsulates the recorded raw packets to an sFlow packet, and sends the sFlow packet to the collector.
Ethernet Frame Data	Analyzes Ethernet headers in Ethernet frames, encapsulates the analyzed Ethernet header to an sFlow packet, and sends the sFlow packet to the collector.
IPv4 Data	Records IPv4 header information in IPv4 packets.
IPv6 Data	Records IPv6 header information in IPv6 packets.
Extended Switch Data	Records VLAN translation and 802.1Q priority mapping information in Ethernet frames. VLAN 0 is an invalid VLAN.
Extended Router Data	Records routing information for packets.

Counter sampling

An sFlow agent periodically obtains traffic statistics on an interface. **Table 8-2** lists the main fields in counter sampling packets. Compared with flow sampling, counter sampling focuses on traffic statistics on an interface rather than traffic details.

Table 8-2 Main fields in counter sampling packets

Field	Description
Generic Interface Counters	Records basic information and traffic statistics on an interface.
Ethernet Interface Counters	Records traffic statistics on an Ethernet interface.
Processor Information	Records CPU usage and memory usage of a device.

sFlow Packet Format

Figure 8-1 shows the sFlow packet format. sFlow packets are encapsulated in UDP packets. By default, sFlow packets are transmitted by known port 6343. sFlow packets use the following packet header formats: Flow sample, Expanded Flow sample, Counter sample, and Expanded Counter sample. Expanded Flow sample and Expanded Counter sample are added to sFlow version5 and are extensions to Flow sample and Counter sample, but they are not compatible with earlier versions. All expanded sampling packets must be encapsulated with the expanded sampling packet header.

8.3 Applications

Network Monitoring

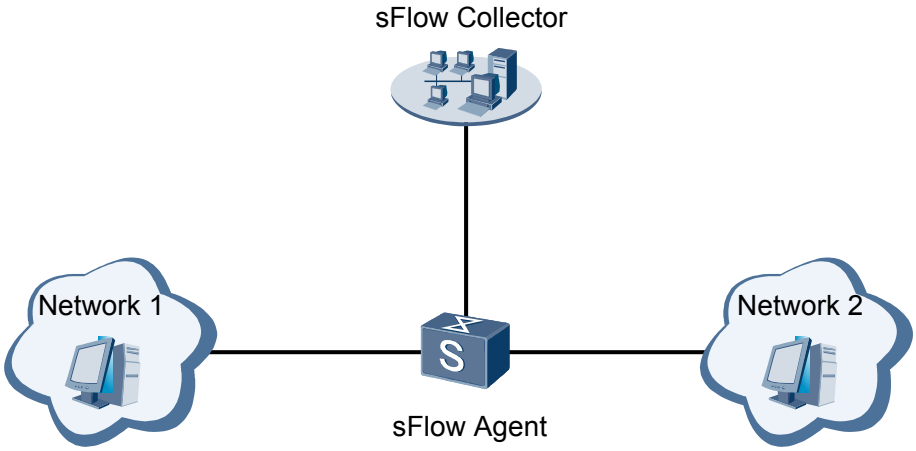
Network maintenance personnel often use the traffic monitoring technique to monitor networks.

Enterprise network users often have requirements for traffic on an interface and device running. They require a traffic monitoring technique on an interface to locate unexpected traffic and the source of attack traffic immediately so that they can rectify faults quickly to ensure stable running of the network.

sFlow is an interface-based traffic analysis technique that collects packets on an interface based on the sampling ratio. In flow sampling, an sFlow agent analyzes the packets including the packet content and forwarding rule, and encapsulates the original packets and parsing result into sFlow packets. Then the sFlow agent sends the sFlow packets to an sFlow collector. In counter sampling, an sFlow agent periodically collects traffic statistics on an interface, CPU usage, and memory usage. sFlow focuses on traffic on an interface, traffic forwarding, and device running, so it can be used to monitor and locate network exceptions. The sFlow collector displays the traffic statistics in a report, which helps you locate faults.

As shown in **Figure 8-2**, before collecting traffic statistics on an interface and analyzing the collected traffic statistics, configure an sFlow agent and connect the sFlow agent to an sFlow collector.

Figure 8-2 Networking diagram for sFlow application



8.4 References

The following table lists the references of this document.

Document	Description	Remarks
sFlow version 5	Inmon sFlow version 5	-
RFC 3176	Inmon sFlow version 4	-
RFC 1014	XDR: External Data Representation Standard	sFlow data standard

9 HGMP

About This Chapter

9.1 Introduction to HGMP

Definition

The Huawei Group Management Protocol (HGMP) is a Huawei proprietary protocol designed for automatic operation and centralized management of a large number of devices scattered in a wide range.

Purpose

As the network scale increases, a large number of access devices are deployed at the edge of a network. Managing these devices is complicated and takes a large amount of time. Each device requires an IPv4 address. This worsens shortage of IPv4 addresses. HGMP is developed to solve the preceding problems.

Benefits

- Save public network addresses.
- Simplify configuration and management. A network administrator can manage and maintain all devices in a cluster by configuring a public IP address on one device, and does not need to log in to each device.
- Provide the topology discovery and display function, which facilitates network monitoring and debugging.
- Upgrade software, set parameters, and download files for multiple devices at the same time, without restricted by the network topology and distances.

9.2 Principles

9.2.1 Roles in a Cluster

Roll Assignment

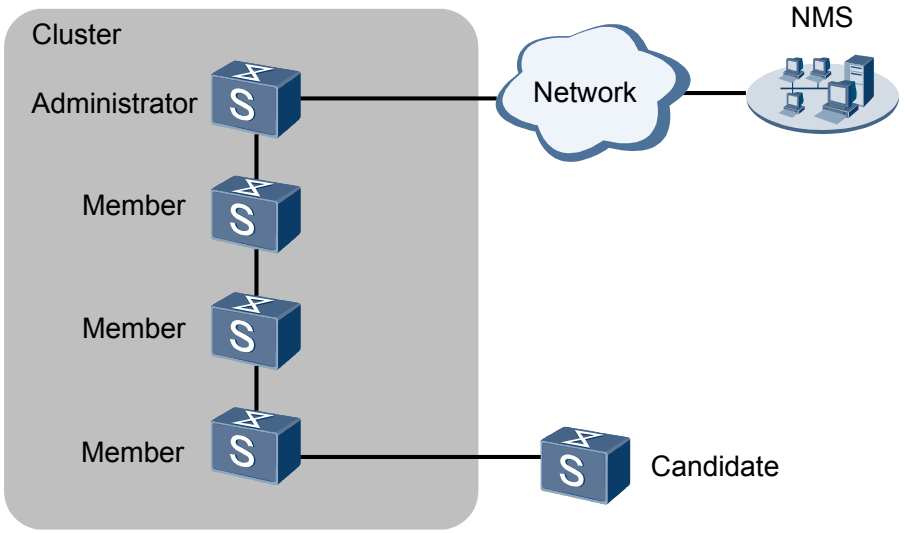
Switches in a cluster play the following roles depending on their positions and functions:

- Administrator switch: functions as an interface for management of the entire cluster. Only the administrator switch needs to be configured with a public IP address in the cluster. Each cluster has only one administrator switch. Other switches in the cluster can be configured, managed, and monitored through the administrator switch. The administrator switch collects relevant information to discover and determine candidate switches and add member switches to the cluster.
- Member switch: is a switch being managed in the cluster.
- Candidate switch: is a switch that has not joined any cluster but is a potential member switch of a cluster. The administrator switch has collected the topology information of the candidate switch but has not added the candidate switch to the cluster.

NOTE

The administrator, member, or candidate switch is only a role and does not indicate an actual switch type.

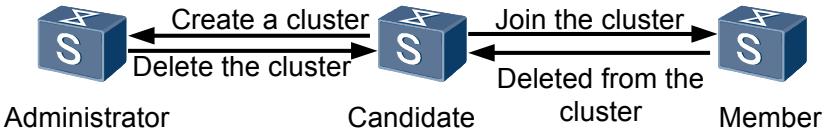
Figure 9-1 Typical networking diagram of a cluster



As shown in **Figure 9-1**, the switch that is configured with a public IP address and performs the management function is an administrator switch. Other switches being managed are member switches. The switch that has not joined the cluster but has cluster capabilities is a candidate switch. The administrator switch and the member switches form the cluster.

Role Switching

Figure 9-2 Role switching in a cluster



As shown in **Figure 9-2**, the role of a switch can be switched according to the following rules:

- When a cluster is created on a candidate switch, the candidate switch becomes the administrator switch of the cluster. The administrator switch becomes a candidate switch only when the cluster is deleted.
- When a candidate switch joins a cluster, the candidate switch becomes a member switch. When the member switch is deleted from the cluster, the member switch becomes a candidate switch again.

9.2.2 HGMP Principles

HGMP uses two important protocols to manage a cluster:

- Neighbor Discover Protocol (NDP)
- Network Topology Discover Protocol (NTDP)

With the NDP and NTDP protocols, the administrator switch collects topology information, establishes a cluster, and maintains the cluster. The cluster establishment and maintenance are

independent of each other. The topology collection process starts before the cluster is established. A cluster is established and maintained as follows:

- All the switches use NDP to obtain information about neighbor switches, including software versions, host names, MAC addresses and interface names of the neighbor switches.
- The administrator switch uses NTDP to collect switch information and connection information within a specified number of hops and determines candidate switches according to the collected topology information.
- The administrator switch adds the candidate switches to the cluster according to the candidate switch information collected using NTDP. Then the cluster is established.
- After the cluster is established, the administrator switch can be used to operate and manage the entire cluster. For example, incremental configuration, batch configuration delivery, plug-and-play, batch restart, and configuration synchronization can be performed.
- As is required, the administrator switch can add or delete member switches to maintain the entire cluster.

9.2.3 NDP

The Neighbor Discovery Protocol (NDP) obtains information about directly-connected neighbor switches. A cluster is established and maintained based on neighbor switch information discovered and maintained using NDP.

NDP has the following characteristics:

- Information about neighbor switches includes switch types, hardware versions, software versions, interfaces, switch IDs, address information, switch capabilities, and hardware description.
- The NDP protocol discovers only directly-connected neighbors.
- The NDP protocol runs at the data link layer and supports different network layer protocols.
- The NDP protocol is not limited by the Spanning Tree Protocol (STP), and NDP packets can pass through an STP-blocked interface.
- The NDP packets are not forwarded.

The working principles of NDP are described as follows:

- A switch running NDP periodically sends NDP packets to neighbors. The NDP packets carry the local end information and the aging time of the NDP packets on the receiving switches. The switch receives the NDP packets sent from neighbor switches but does not forward the NDP packets.
- The switch running NDP stores and maintains an NDP neighbor information table and creates an entry for each neighbor in the table. When the switch discovers a new neighbor (receives the first NDP packet sent by the neighbor), the switch creates an entry for the new neighbor. If the received NDP information of a neighbor is different from existing information on the switch, the switch updates the matching entry and aging time. If the received NDP information is the same as the existing information, the switch only updates the aging time. If the switch does not receive the NDP information of a neighbor when the aging time expires, the switch deletes the matching entry.
- After NDP is enabled globally and on an interface of a switch, the switch immediately sends three NDP packets from the interface. Multiple NDP packets are sent to ensure network reliability. When NDP is disabled on an interface, the switch immediately sends an NDP

packet with the aging time set to 0 from the interface. Meanwhile, NDP information about the interface receiving this NDP packet ages out rapidly.

9.2.4 NTDP

The Network Topology Discovery Protocol (NTDP) collects network topology information. This protocol provides the administrator switch with information about the switches that can be added to a cluster.

NDP provides an adjacency table for NTDP. According to the adjacency information, NTDP sends and forwards NTDP topology requests to collect NDP information and connection information of all the switches within a specified range.

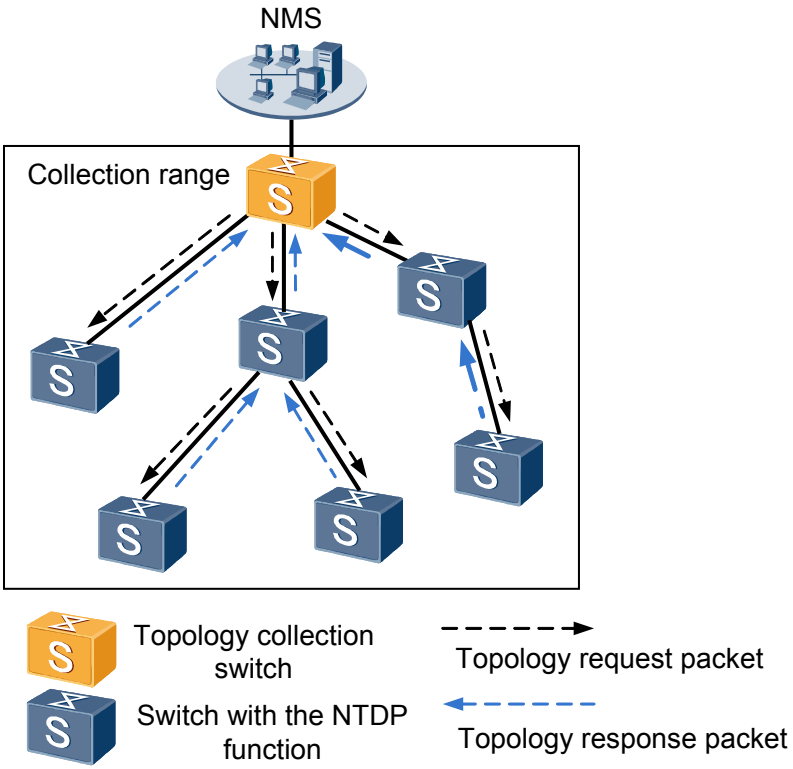
NOTE

After a cluster is established, member switches send handshake messages to notify the administrator switch of topology changes that they discover using NDP. The administrator switch can use NTDP to collect topology information of switches within a specified number of hops to promptly reflect the changes in the network topology.

As shown in **Figure 9-3**, the process of collecting cluster topology information is as follows:

- The topology collection switch periodically sends NTDP topology request packets from an NTDP-capable interface to collect topology information.
- Each switch receiving a topology request packet immediately sends a topology response packet to the topology collection switch, replicates the topology request packet on an NTDP-capable interfaces, and sends the replicated topology request packets to adjacent switches. A topology response packet includes basic local information and NDP information about all the adjacent switches.
- After receiving a topology request packet, the adjacent switches perform the same operations until the topology request packets reach all the switches within a specified number of hops.
- After complete topology information is collected, the administrator switch or the network management system (NMS) can describe the network topology according to the topology information.

Figure 9-3 NTDP implementation



NMS: Network Management System

Transmission of NTDP packets has the following characteristics:

- Topology request packets are broadcast, with the destination MAC address as the HGMP multicast MAC address. You can configure the HGMP multicast MAC address as required.
- Topology response packets are unicast, with the destination MAC address as the unicast MAC address of the switch initiating the topology request.

When the topology request packets are spread on the network, a large number of switches on the network receive the topology request packets and simultaneously send topology response packets. To avoid network congestion and reduce loads on the topology collection switch, switches use the following measures to control the request packet transmission:

- After receiving a topology request packet, the switch whose topology information needs to be collected forwards the packet from the first interface after a delay, but does not immediately forward the packet.
- After being forwarded from the first interface, the topology request packet is successively forwarded from other interfaces on the switch after a delay.

9.2.5 Multicast MAC Address and Management VLAN

Multicast MAC Address

Cluster management packets are transmitted in the format of multicast packets. These multicast packets have a fixed destination address of 0x0180-C200-000A.

The multicast MAC address can be configured within the following ranges:

- 0x0180-C200-0003 to 0x0180-C200-0007
- 0x0180-C200-0009 to 0x0180-C200-0010
- 0x0180-C200-0020 to 0x0180-C200-002F

NOTE

Some multicast MAC addresses have been used by other protocols. Therefore, multicast MAC addresses used in HGMP are not contiguous.

Management VLAN

Management VLAN is the VLAN where cluster protocol packets are transmitted. To perform cluster management using HGMP, configure the management VLAN. The management VLAN restricts cluster management range and have the following functions:

- Except NDP packets, all the data of a cluster is transmitted within the management VLAN. The cluster data is isolated from other packets, which increases security.
- The administrator switch and member switches can communicate with each other through the management VLAN.

The management VLAN ID can be configured as required. By default, the management VLAN ID is 1.

As required by HGMP, interfaces (including cascading interfaces) connecting the administrator switch and member switches or candidate switches must allow the management VLAN to pass through. If an interface does not allow the management VLAN to pass through, the switch connected to the interface cannot join the cluster.

NOTE

When a candidate switch is connected to the administrator switch through another candidate switch, the interfaces connecting the two candidate switches are cascaded interfaces.

9.2.6 Cluster Establishment and Maintenance

Requirements for establishing a cluster are as follows:

- All the switches to be added to a cluster support HGMP cluster operations.
- At least one switch can function as the administrator switch.
- The administrator switch is configured first, and the cluster is established by adding member switches.
- The cluster is identified by the MAC address of the administrator switch. When the administrator switch sends a restart request, HGMP identifies the cluster by the cluster name. Different clusters must be configured with different names.
- If a switch belongs to cluster A, it cannot be a member switch or a candidate switch of cluster B.

Adding a Member Switch

A member switch can be added automatically or manually. The processes in automatic and manual modes are the same, but the triggering conditions are different:

- In automatic mode, after the administrator switch is specified, the system triggers the process of adding a member switch.

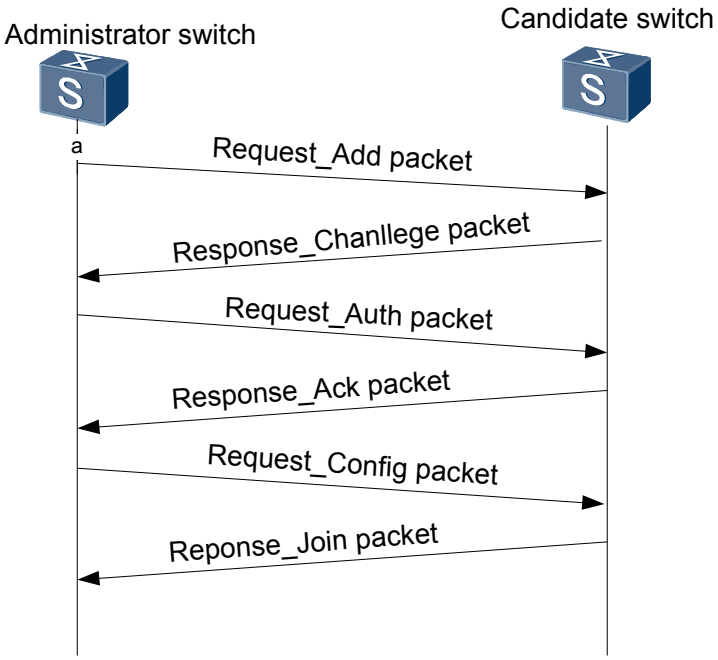
- In manual mode, after the administrator switch is specified, the user uses command lines or NMS to add a member switch.

A member switch can be added through the following two processes:

1. **Add a member switch through authentication.**

   If a candidate switch is configured with a password, the administrator switch needs to be authenticated when sending a Request_Add packet to the candidate switch. **Figure 9-4** shows the detailed process.

   Figure 9-4 Adding a member switch through authentication



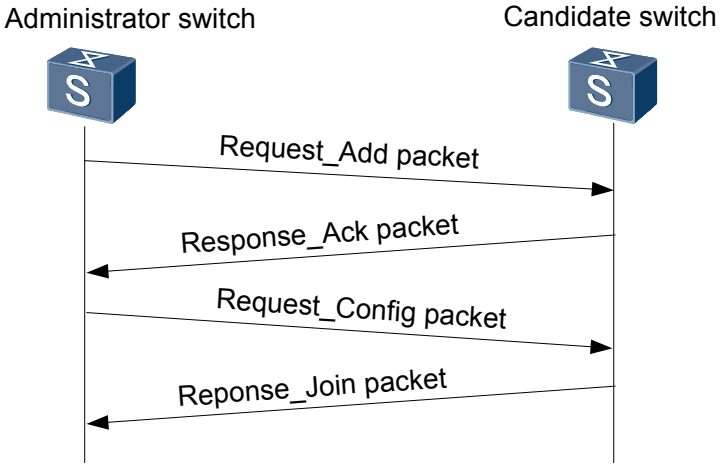
   Process description:

   The password in the Request_Auth packet and configuration information requiring encryption in the Request_Config packet are encrypted using the MD5 algorithm. The administrator switch and the candidate switch use the retransmission mechanism. If either of them fails to receive the response of the peer within the timeout interval, it considers that the request packet is lost on the link and retransmits the request packet. If the administrator switch or the candidate switch fails to receive the response after sending a packet for the maximum number of times, it stops the processing and the candidate switch fails to join the cluster.

2. **Add a member switch without authentication.**

   If a candidate switch is not configured with a password, the administrator switch does not need to be authenticated when sending a Request_Add packet to the candidate switch. **Figure 9-5** shows the detailed process.

Figure 9-5 Adding a member switch without authentication



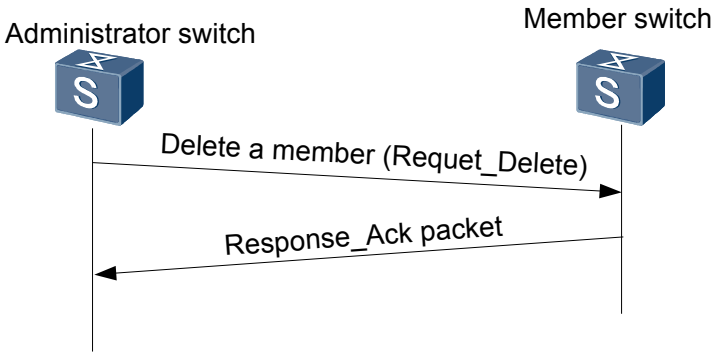
Process description:

After receiving a Request_Add packet from the administrator switch, the candidate switch directly returns a Response_Ack packet instead of a Response_Challenge packet. After the administrator switch confirms that the candidate switch agrees to join the cluster, the administrator switch delivers unencrypted configuration information to the candidate switch. The administrator switch and the candidate switch use the retransmission mechanism. If either of them fails to receive the response of the peer within the timeout interval, it considers that the request packet is lost on the link and retransmits the request packet. If the administrator switch or the candidate switch fails to receive the response after sending a packet for the maximum number of times, it stops the processing and the candidate switch fails to join the cluster.

Deleting a Member Switch

Figure 9-6 shows the process of deleting a member switch, which is initiated by the administrator switch.

Figure 9-6 Deleting a Member Switch



Process description:

To delete a member switch, the administrator switch deletes the stored information about the member switch, sends a Request_Delete packet to the member switch, and waits for the member switch to confirm the deletion. If the administrator switch does not receive the Response_Ack packet from the member switch within the timeout interval, it retransmits the Request_Delete

packet for the specified number of times. When the administrator switch needs to be deleted, it sends a Request-Delete packet to all the member switches.

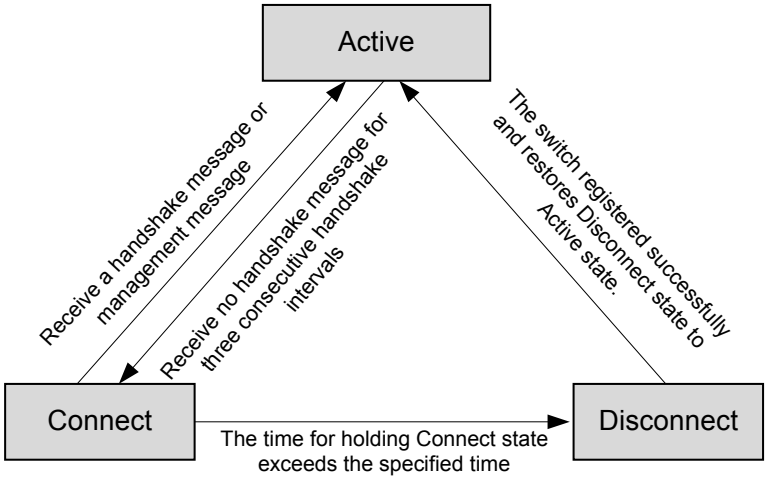
9.2.7 Cluster Communication

Communication within a Cluster

State machine transitions of switches

In an HGMP cluster, the administrator switch and member switches exchange handshake messages to maintain the connection status in real time. **Figure 9-7** shows the connection status between the administrator switch and a member switch.

Figure 9-7 State machine of the administrator switch and a member switch



Process description:

- After a cluster is established and a candidate switch becomes a member switch by joining the cluster, the administrator switch stores the state information about the member switch and sets the state to Active. The member switch stores its own state information and sets its own state to Active.
- The administrator switch and the member switch periodically send handshake messages to each other. The administrator switch does not respond after receiving a handshake message sent by the member switch and maintains the Active state of the member switch. The member switch does not respond after receiving a handshake message sent by the administrator switch and maintains its Active state.
- If the administrator switch does not receive any handshake message from the member switch within three handshake intervals, the administrator switch changes the state of the member switch from Active to Connect. Likewise, if the member switch does not receive any handshake message from the administrator switch within three handshake intervals, the member switch changes its own state from Active to Connect.
- If the administrator switch receives a handshake or management message from the member switch in Connect state within the holding time, the administrator switch changes the state of the member switch back to Active. Otherwise, the administrator switch changes the state of the member switch to Disconnect and considers that it has disconnected from the member switch. If the member switch in Connect state receives a handshake or management packet

from the administrator switch within the holding time, the member switch changes its own state into Active. Otherwise, the member switch changes its state into Disconnect.

- When communication between the administrator switch and the member switch is restored, the member switch in Disconnect state joins the cluster again. After successfully joining the cluster, the state of the member switch is restored to Active on both the administrator switch and member switch.

Processing of topology changes

When the cluster topology changes because of an abnormal member switch, the member switch adjacent to the abnormal member switch discovers the change first. When detecting that a neighbor is abnormal, a member switch sends a handshake message containing the topology change information to the administrator switch. The administrator switch obtains the topology change information from the handshake message and starts to collect the topology information again. At the same time, the administrator switch instructs the NMS to update the topology information.

If the neighbor of a switch is not in the cluster, the administrator switch collects the topology information periodically to update the topology information. Changes of the topology do not affect composition of the cluster. That is, when the topology changes, members in the cluster do not change automatically. The newly discovered candidate switches cannot join the cluster automatically and need to be added to the cluster manually. When the topology changes due to abnormal links, member switches are not deleted automatically and must be deleted manually.

In addition to collecting the local topology information randomly, the administrator switch periodically collects the global topology information because the local topology information cannot show all changes in the global topology.

Communication Between Cluster Switches and Devices Outside the Cluster

When a cluster is established, switches in the cluster use public IP addresses or private IP addresses to communicate within the cluster.

- Public IP address being used

  If cluster members are assigned public IP addresses, they can directly communicate with devices outside the cluster.

- Private IP address being used

  1. When a user sends a packet with a private IP address to a device outside the cluster, the administrator switch replaces the private source IP address and port number of the packet with a public IP address and port number that can be identified by external devices on the outbound interface.
  2. When receiving a packet from an external device, the administrator switch searches the address-and-port mapping table to replace the destination IP address and port number of the packet with a private IP address and port number that can be identified on the private network.

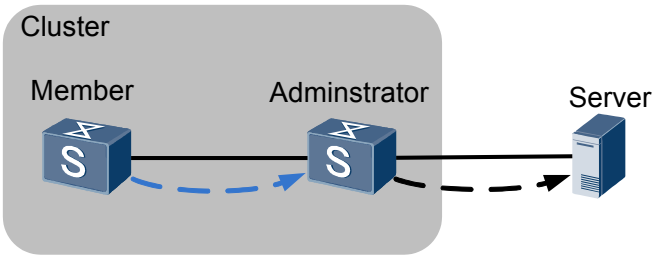
A member switch accesses a device outside the cluster.

A member switch can access a device outside the cluster in the following two modes:

- Redirection mode

  The switches need to support the redirection function. Member switches access a device outside the cluster using the redirection function. **Figure 9-8** shows the access process.

Figure 9-8 A member switch accesses a device outside the cluster in redirection mode

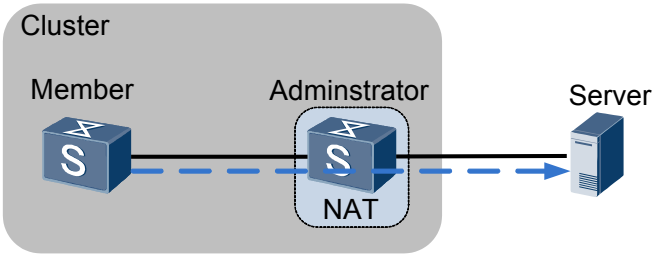


To access a device outside the cluster, a member switch establishes a connection with the administrator switch and accesses the device outside the cluster through the administrator switch.

- Network address translation (NAT) mode

  A member switch accesses a device outside the cluster through the NAT function. **Figure 9-9** shows the access process.

Figure 9-9 A member switch accesses a device outside the cluster in NAT mode



In NAT mode, when the administrator switch receives a packet that a member switch sends to a device outside the cluster, the administrator switch performs NAT on the packet and forwards the packet to the device outside the cluster. The following information in the packet is translated:

1. Source port number (translated to the port number reserved for the cluster)
2. Source IP address (translated to the public IP address of the administrator switch)

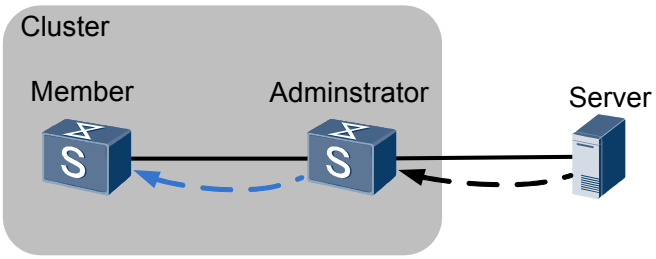
A device outside the cluster accesses a member switch.

A device outside the cluster can access a member switch in the following two modes:

- Redirection mode

  Devices outside the cluster access member switches using the redirection function. **Figure 9-10** shows the access process.

Figure 9-10 A device outside the cluster accesses a member switch in redirection mode

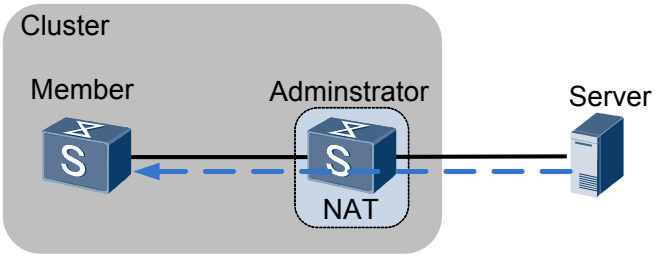


When a device outside a cluster needs to access a member switch in redirection mode, the device establishes a connection with the administrator switch, and accesses the member switch through the administrator switch.

- NAT mode

  Devices outside the cluster access member switches through the NAT function. **Figure 9-9** shows the access process.

Figure 9-11 A device outside the cluster accesses a member switch in NAT mode



When a device outside a cluster accesses a member switch in NAT mode, it sends a packet with the public IP address of the administrator switch as the destination IP address. After receiving the packet, the administrator switch performs NAT on the packet and then forwards the packet to the member switch. The following information in the packet is translated:

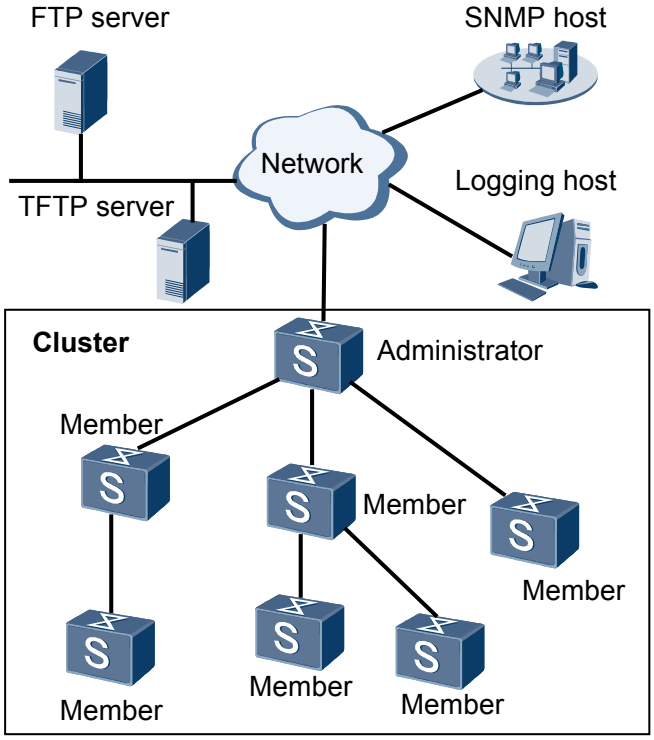
1. Destination IP address (translated to the cluster IP address of the member switch).
2. Destination port number (translated to the port number reserved for the cluster).
3. Source IP address (translated to the cluster IP address of the administrator switch).

9.3 Application

9.3.1 Batch Configuration Delivery

Ethernet technology is widely used on metropolitan area networks (MANs) and enterprise networks. As the network scale increases, a large number of access devices are deployed at the edge of a network. Managing these devices is complicated and takes a large amount of time. Each device requires an IPv4 address. This worsens shortage of IPv4 addresses. HGMP is developed to address the preceding problems and is used in banks, electrical power systems, campuses, and enterprises. The devices scattered on a network can be added to a cluster for centralized management, which greatly improves maintenance efficiency and saves public IP addresses. **Figure 9-12** shows the HGMP application.

Figure 9-12 HGMP application



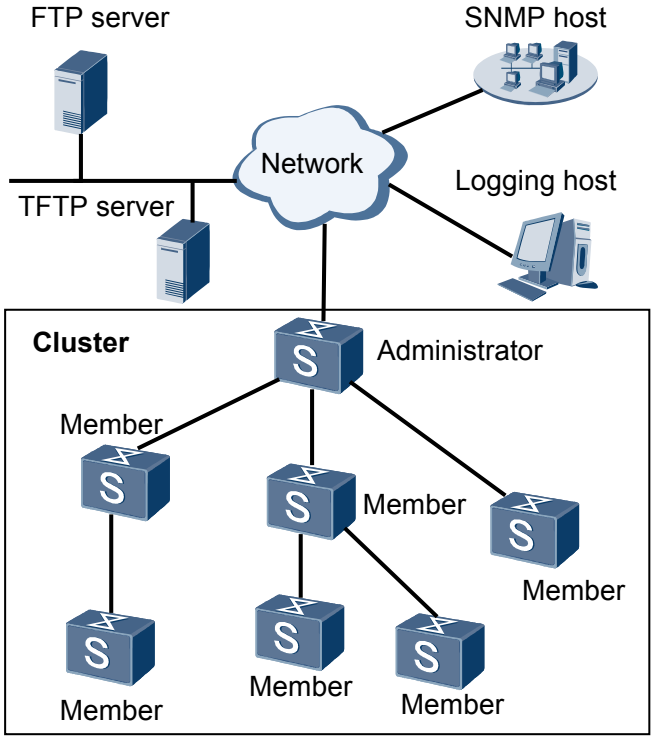
Batch configuration delivery

- HGMP provides the batch delivery function that allows some or all the devices in a cluster to download configuration files from a File Transfer Protocol (FTP) server.
- To prevent the FTP server from being congested because excessive clients download files at the same time, you can set the number of member switches allowed to simultaneously download files through FTP in HGMP.
- A query of the batch delivery result list can show the files obtained by member switches.
- The batch delivery result list stores only the latest result. The later result overrides the previous result.
- The configuration results are saved in batches.

9.3.2 Batch Restart

Ethernet technology is widely used on metropolitan area networks (MANs) and enterprise networks. As the network scale increases, a large number of access devices are deployed at the edge of a network. Managing these devices is complicated and takes a large amount of time. Each device requires an IPv4 address. This worsens shortage of IPv4 addresses. HGMP is developed to address the preceding problems and is used in banks, electrical power systems, campuses, and enterprises. The devices scattered on a network can be added to a cluster for centralized management, which greatly improves maintenance efficiency and saves public IP addresses. **Figure 9-13** shows the HGMP application.

Figure 9-13 HGMP application



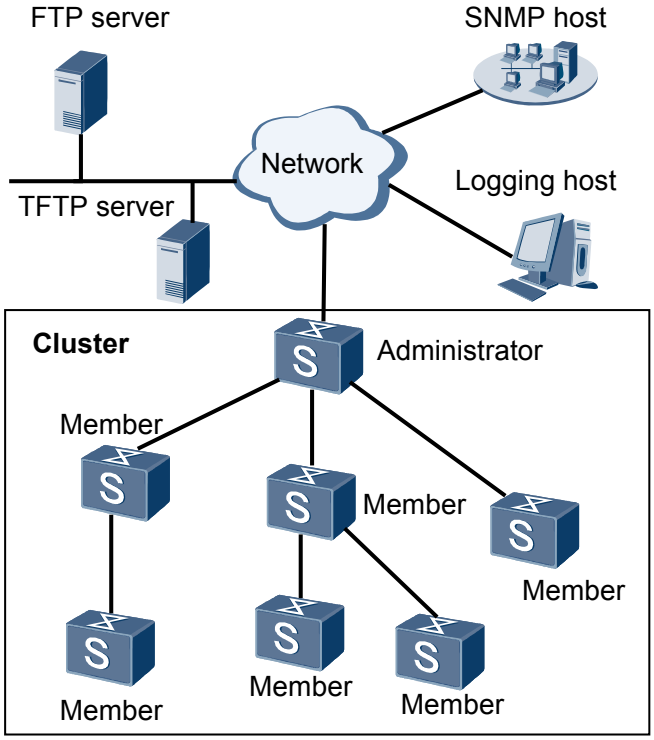
Batch restart

- HGMP provides the batch restart function that allows some or all member switches to restart when their software or configurations are upgraded.
- During batch restart, member switches do not save the current configuration.
- After receiving the batch restart command, member switches wait 1 second to guarantee the flooding of control packets over the cluster.

9.3.3 Incremental Configuration

Ethernet technology is widely used on metropolitan area networks (MANs) and enterprise networks. As the network scale increases, a large number of access devices are deployed at the edge of a network. Managing these devices is complicated and takes a large amount of time. Each device requires an IPv4 address. This worsens shortage of IPv4 addresses. HGMP is developed to address the preceding problems and is used in banks, electrical power systems, campuses, and enterprises. The devices scattered on a network can be added to a cluster for centralized management, which greatly improves maintenance efficiency and saves public IP addresses. **Figure 9-14** shows the HGMP application.

Figure 9-14 HGMP application



Incremental configuration

In a cluster, some member switches may have the same configurations such as VLAN configuration and interface shutdown/undo shutdown. Through incremental configuration, some or all the member switches in a cluster can be delivered with the same configurations. You only need to edit the command list on the administrator switch, and this switch then performs the configuration based on the commands. In addition, the command execution results of each member switch can be queried.

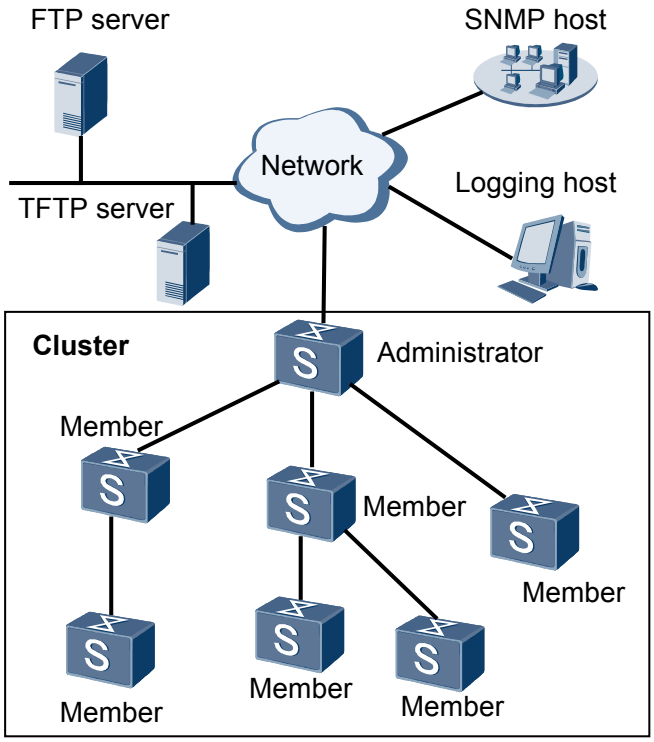
- Incremental configuration can be performed only on the administrator switch.
- Incremental configuration is used to configure member switches in a batch and is performed on all the selected switches at one time.
- After incremental configuration is performed, a result list is returned to report the command output on each member switch. If an error occurs when a command is executed, the command can be located according to the sequence number.
- Only the last result of the incremental configuration command is saved. The previous result is automatically replaced by the latest result.
- You can edit a configuration command list in the incremental configuration view. The commands are executed in the specified views and the command execution sequence is the same as that on a switch.

9.3.4 Plug-and-Play

Ethernet technology is widely used on metropolitan area networks (MANs) and enterprise networks. As the network scale increases, a large number of access devices are deployed at the edge of a network. Managing these devices is complicated and takes a large amount of time. Each device requires an IPv4 address. This worsens shortage of IPv4 addresses. HGMP is

developed to address the preceding problems and is used in banks, electrical power systems, campuses, and enterprises. The devices scattered on a network can be added to a cluster for centralized management, which greatly improves maintenance efficiency and saves public IP addresses. **Figure 9-15** shows the HGMP application.

Figure 9-15 HGMP application



Plug-and-play

Manual configurations must be performed on a switch that needs to join a cluster. When a great number of switches need to join a cluster, you can use the plug-and-play function to simplify the process. Use a Product Adaptive File (PAF) to configure the switches, and connect switches to the cluster devices physically. After that, the switches can join the cluster automatically.

- The PAF is required to complete basic configurations of member switches.
- Plug-and-play needs to be enabled on the administrator switch.
- The interfaces connecting the administrator switch and member switches need to be added to the management VLAN in trunk mode.
- The administrator switch must be enabled to periodically collect NTDP packets.

NOTE

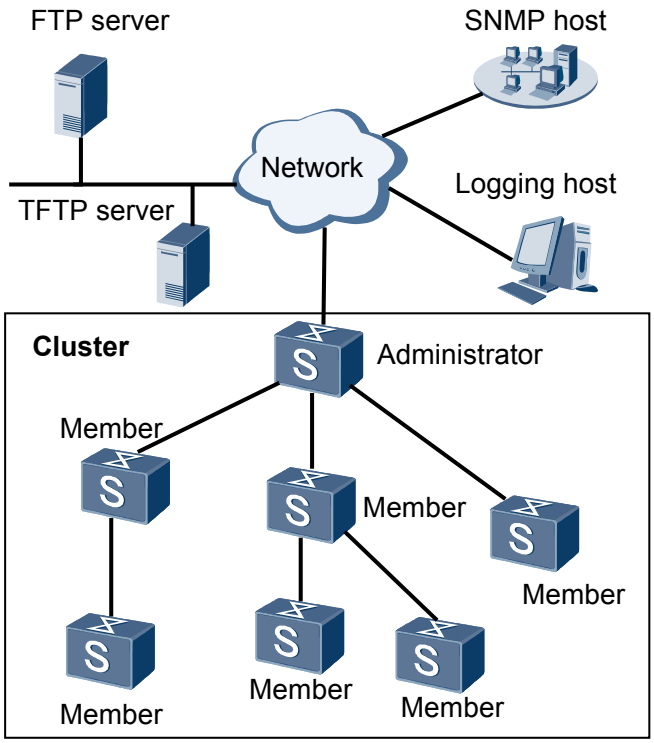
A PAF is similar to an ini file used in a Windows operating system. A PAF determines the features that can be provided by a product and capabilities (specifications) of each feature.

9.3.5 Configuration Synchronization

Ethernet technology is widely used on metropolitan area networks (MANs) and enterprise networks. As the network scale increases, a large number of access devices are deployed at the edge of a network. Managing these devices is complicated and takes a large amount of time. Each device requires an IPv4 address. This worsens shortage of IPv4 addresses. HGMP is

developed to address the preceding problems and is used in banks, electrical power systems, campuses, and enterprises. The devices scattered on a network can be added to a cluster for centralized management, which greatly improves maintenance efficiency and saves public IP addresses. **Figure 9-16** shows the HGMP application.

Figure 9-16 HGMP application



Configuration synchronization

You can save the configuration files of member switches to a specified FTP server through configuration synchronization.

- The administrator switch can deliver the specified configuration file to the specified member switches through the batch delivery function.
- When a device is replaced with a new device of the same type without changing the physical topology of the cluster, the administrator switch automatically delivers the configuration file of the old device to the new one.
- Enable plug-and-play when you replace the devices and have configuration files delivered automatically. Ensure that the physical interfaces of the old and new devices are the same; otherwise, the configuration file of the old device cannot take effect.
- Specify an FTP server before configuration synchronization.

9.4 References

The HGMP protocol is a Huawei proprietary protocol.

NOTE

HGMP can manage only the switches that support the HGMP function.