Recommendations

- Huawei Talent Online Website
  - https://e.huawei.com/en/talent/#/
- Huawei e-Learning
  - https://e.huawei.com/en/talent/#/search?productTags=&productName=&navType=learningNavKey
- Huawei Certification
  - https://e.huawei.com/en/talent/#/cert?navType=authNavKey
- Find Training
  - https://e.huawei.com/en/talent/#/halp/home?navType=halp

Copyright © Huawei Technologies Co., Ltd. 2021.

Huawei Certification

HCIP-Datacom-Network Automation Developer



Huawei Technologies Co.,Ltd.

Copyright © Huawei Technologies Co., Ltd. 2021. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of 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 Certification

HCIP-Datacom-Network Automation Developer

V1.0

Preface

Introduction

This document is a training material for HCIP-Datacom certification. It is intended for trainees wanted to participate the HCIP-Datacom exam and those who want to obtain the HCIP-Datacom-Network Automation Developer certification. Obtained the certification indicates that you have the ability of network automation.

Content

This document consists of four modules, covering the basics of Python programming, software version control principles and operations, open programmability of Huawei devices, iMaster-NCE northbound openness and iMaster-NCE service openness and programmability.

Module 1 Programming basics. Introduce the course architecture and basic programming knowledge.

Module 2 Device openness and programmability. Introduce the open programmability at the device layer, including SSH, SNMP, NETCONF, Telemetry, and OPS.

Module 3 iMaster-NCE northbound openness. Introduce the northbound open APIs and practices of iMaster-NCE.

Module 4 iMaster-NCE service openness and programmability. Introduce how to use the Specific NE Driver(SND) and Specific Service Plugin(SSP) to implement quick interconnection with new devices and build new services.

Contents

Introduction to Network Programming and Automation 1

Python Programming Basics 27

Git Fundamentals and Practices 51

SSH Principles and Practices 100

SNMP Fundamentals and Practices 161

NETCONF YANG Principles and Practices 204

Telemetry Fundamentals and Practices 255

OPS Fundamentals and Practices 308

SDN Overview 349

RESTful Principles and Practices 396

iMaster NCE-Campus Open API Introduction 454

iMaster NCE-Fabric Open API Introduction 489

Introduction to Network AI 552

NCE Service Openness and Programmability 567



Introduction to Network Programming and Automation

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Foreword

- Networks are evolving to be more open, more flexible, and simpler. Huawei's vision is to help customers build intent-driven networks (IDNs).
- Network openness and programmability are prerequisites of intent-driven networks.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Objectives

- Upon completion of this course, you will be able to describe:
  - Background of network programming and automation.
  - Network openness levels.
  - Open capabilities of Huawei iMaster NCE.
  - Positioning and capabilities of network automation engineers.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

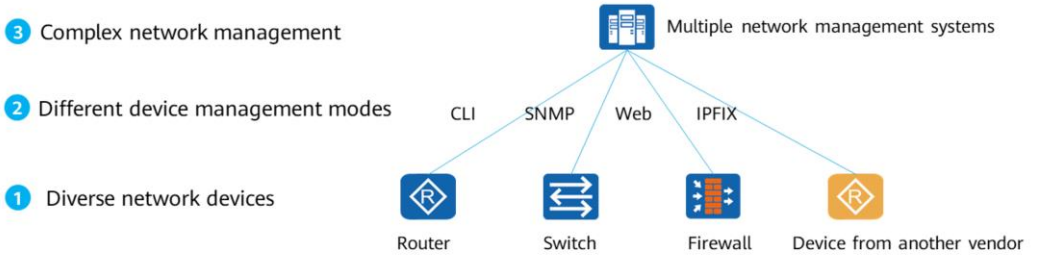
Contents

1. **Network Programming and Automation**
2. Network Automation Engineers
3. Course Overview

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Background: Complex Networks

- From the first day when the computer network is generated, the network is complex and difficult to manage, which is reflected in the following aspects:
  - There are various network devices such as routers, switches, firewalls, and intrusion detection system (IDS).
  - Devices from different or even the same vendors are managed in different ways.
  - Complex devices result in complex network management.

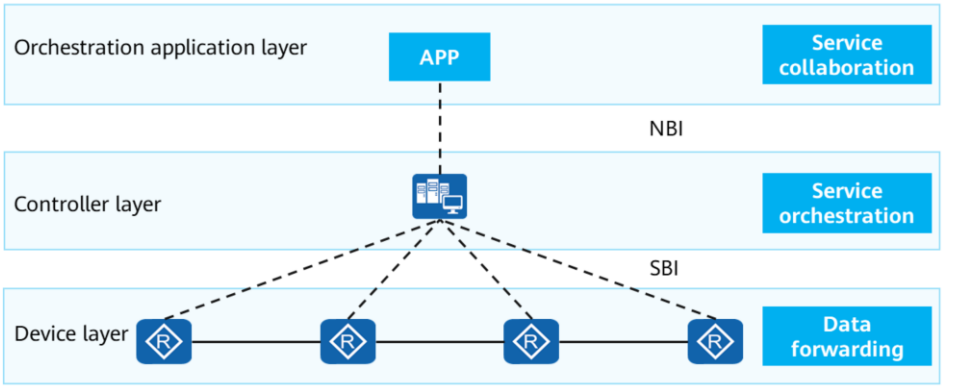


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Different network devices communicate with each other through routing protocols.
- There are different device management modes, such as SNMP, CLI, IPFIX, and Web UI.

Background: Network Architecture Transformation

- Software defined networking (SDN) brings about network architecture transformation. It introduces a network controller to implement centralized control from a global perspective, achieving objectives such as fast service deployment, traffic optimization, and network service openness.

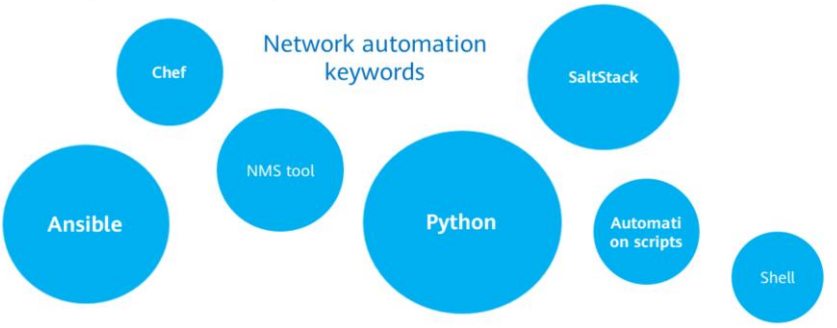


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Orchestration application layer: implements various upper-layer applications of user intents. Typical orchestration applications include OSS and OpenStack. The OSS is responsible for service collaboration on the entire network. The OpenStack is used for network, computing, and storage service collaboration in a data center. There are other orchestration-layer applications. For example, a user wants to deploy a security application. The security application does not care about the device deployment location but invokes a controller NBI, for example, Block (Source IP, DestIP). Then the controller delivers an instruction to network devices. This instruction varies according to the southbound protocol.
- Controller layer: The entity at the controller layer is the SDN controller, which is the core of the SDN network architecture. The control layer is the brain of the SDN system. Its core function is to implement network service orchestration.
- Device layer: The network devices receive instructions from the controller and forward the instructions.
- NBIs: NBIs are used by the controller to interconnect with the orchestration application layer. The main NBIs are RESTful interfaces.
- SBIs: SBIs are protocols used for interaction between the controller and devices, including NETCONF, SNMP, OpenFlow, and OVSDB.

Basic Network Automation

- Network automation means that tools are used to implement automatic network deployment, operation, and maintenance, gradually reducing the dependency on human resources.
- There are many open-source tools that implement network automation in the industry, such as Ansible, SaltStack, Puppet, and Chef. These tools connect to devices through SSH to implement batch operation and management, achieving basic network automation.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- Network automation tools implement basic network automation. That is, tools connect to devices through SSH to implement batch operation and management.

Development of Network Automation

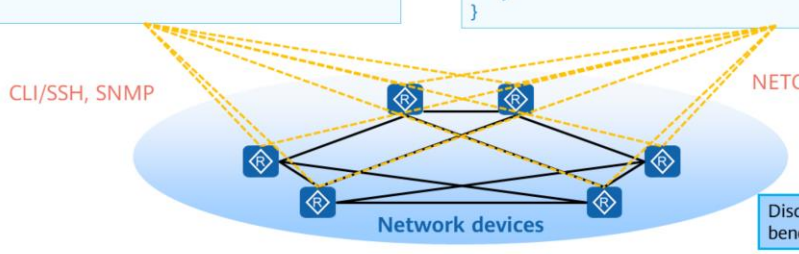
- In basic network automation, networks are managed on the CLI. The pain point is that network devices return unstructured data (text display), which is inconvenient for computers to process. The basic requirement of network automation development is that devices provide structured data, which can greatly promote the development of network automation. Devices provide NETCONF/RESTCONF interfaces to provide data in XML or JSON format.

Unstructured data: easy to understand for humans

Interface	InUti	OutUti	inErrors	outErrors
GigabitEthernet0/0/0	10%	20%	0	0
GigabitEthernet0/0/1	20%	30%	0	0
...				

Structured data: easy to understand for machines

```
{ "Interfaces":
    { "GigabitEthernet0/0/0":
        { "InUti": "10", "OutUti": "20", "inErrors": "0", "outErrors": "0" },
      "GigabitEthernet0/0/1":
        {"InUti": "20", "OutUti": "30", "inErrors": "0", "outErrors":"0"}
    }
}
```



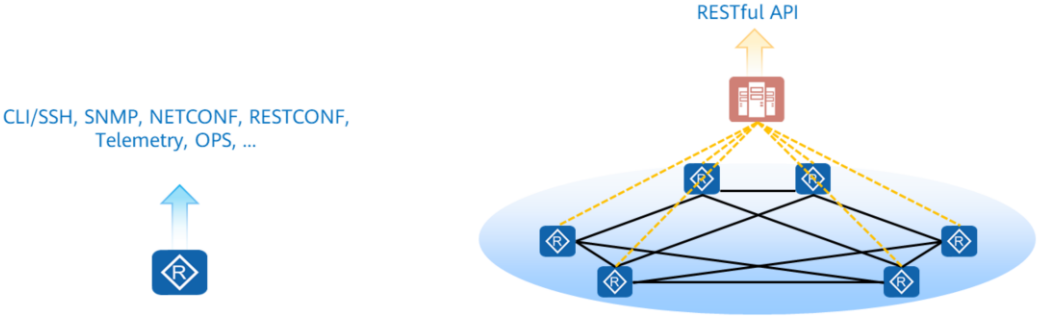
Discussion: What are the benefits of open APIs?

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Unstructured data can be easily understood by humans, but it is difficult for machines to understand and difficult for automatic data collection.
- NETCONF/RESTCONF has many advantages not limited to providing structured data. For details, see NETCONF Principles and Practices.

Network Openness and Programmability

- Network openness and programmability is to use programming methods to implement automated networks on the premise of open networks.
- In the early stage of SDN commercial use and in the future, traditional networks and SDN networks will coexist on a large scale. Network openness is implemented at two levels: device openness and SDN platform openness.



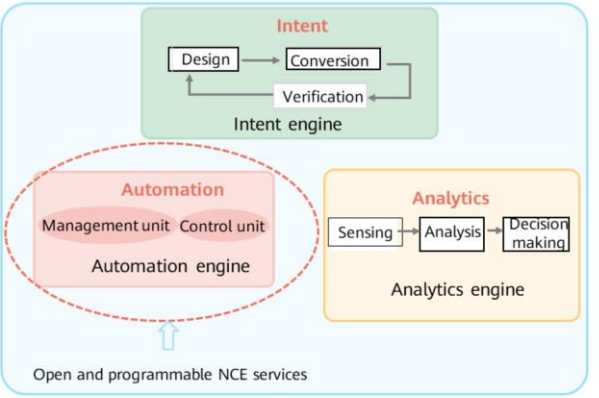
Device-based openness and programmability

SDN-based openness and programmability

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Huawei iMaster NCE, Open and Programmable

- iMaster NCE (NCE for short) is an innovative network cloud engine of Huawei. The overall open programming capabilities of NCE include automation, analytics, and intent engines. The goal is to build a full-lifecycle open programmable architecture to meet customer requirements for openness and programmability. NCE service openness and programmability are designed to help customers implement service automation.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- iMaster NCE is not only a controller, but also provides analysis and network management functions.

Contents

1. Network Programming and Automation
2. **Network Automation Engineers**
3. Course Overview

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Overview and Objectives

- Generally, network engineers focus on network protocol principles, configuration, and management of network devices. System engineers focus on operating system principles, configuration, and management of system services. Development engineers focus on programming languages, algorithms, and other related development capabilities.
- Network automation engineers for network programming and automation must be all-rounders who master skills of these positions to some extent. This part introduces capability requirements on network automation engineers and how to implement network programming and automation.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

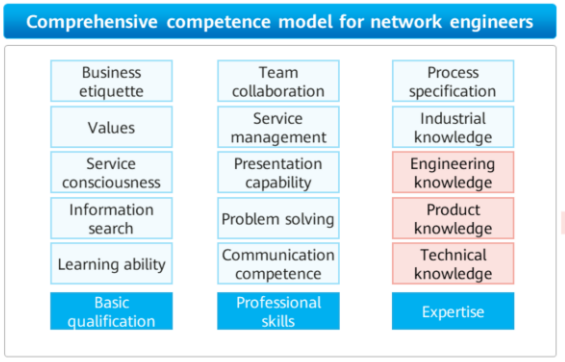
Network Automation Engineers

- Network engineers:
  - Network engineers are professionals who master network technologies, have professional skills, competence, and project management experience in the network engineering field. They are able to fully communicate with customers or other project stakeholders onsite. In addition, they can develop implementation solutions and project plans (recognized by project stakeholders) based on customer requirements and environment factors, fully mobilize resources of all parties to ensure timely and high-quality project implementation, and provide training for stakeholders and deliver engineering documents after the project is implemented.
- Network automation engineers:
  - They are professionals who have network skills, competence, and project management experience in the network automation field. They aim to meet the requirements of automatic deployment, development, and O&M of enterprise networks.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

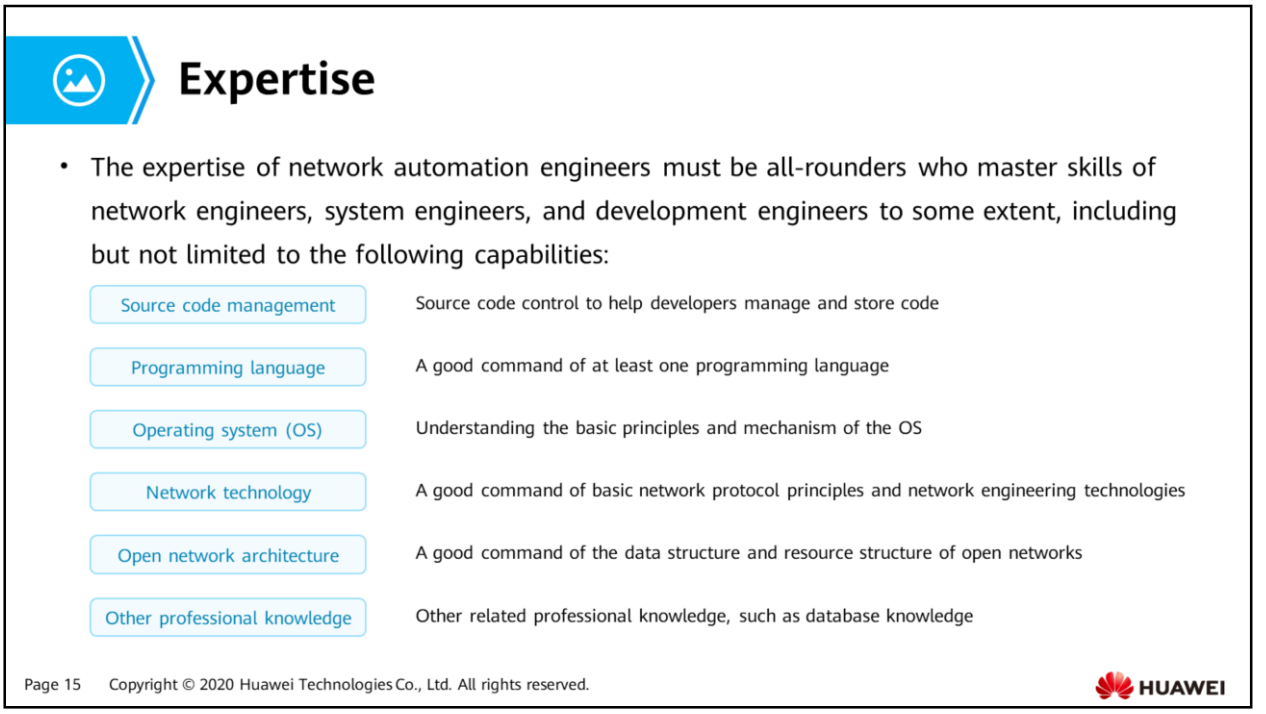
Comprehensive Competence Model

- Compared with the comprehensive competence model of network engineers, network automation engineers have the same basic competence and professional skills, but have different skill requirements in terms of expertise.



In addition to basic network technologies, network automation engineers need to master development-related technology, product, and engineering knowledge.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



- Network automation developers may need to have more professional knowledge, such as database, algorithm, cryptography, software development lifecycle management, development framework, big data, cloud computing, and artificial intelligence (AI), depending on the specific work content and scenario.

Learning Objectives

- This course focuses on the following capabilities of network automation engineers:

Source code management	Git
Programming language	Python and integrated development environment (IDE)
OS	N/A
Network technology	Principles and code implementation of network technologies such as NETCONF, YANG, Telemetry, SNMP, and SSH
Open network architecture	XML/JSON, Huawei NCE northbound RESTful API, and Huawei NCE service openness and programmability
Others	HTTP/TLS, northbound invocation tools

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. Network Programming and Automation
2. Network Automation Engineers
3. **Course Overview**

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Introduction to Course Modules

- This course consists of four modules: programming basics, device openness and programmability, NCE northbound openness, and NCE service openness and programmability.

Programming basics	Device openness and programmability	NCE northbound openness	NCE service openness and programmability
Basic programming knowledge, including: • Python basics • Python advanced knowledge • Source code management	Network programming and automation course module based on device openness capabilities and Python practices: • SSH • SNMP • NETCONF/RESTCONF • Telemetry • Huawei device OPS	Network automation course module based on the northbound openness capability of Huawei NCE, including: • REST and RESTful • HTTP/HTTPS • RESTful invocation tools • Scenario-based invocation practice (campus network and DCN)	Course module based on the NCE service open programming function, including: • Jinja2 • Compilation of an NE driver package • Compilation of a service driver package • Scenario-based service provisioning practices and NBI invoking

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- An open programmability system (OPS) is an open platform that provides representational state transfer (RESTful) application programming interfaces (APIs) to achieve programmability, allowing third-party applications to run on the platform.

Python Overview

- Python is a fully open-source high-level programming language. It was created by Guido van Rossum.
- All coding practices in this course are based on Python.

Advantages of Python:
- Python has elegant syntax, dynamic typing, and interpretative nature. Python allows learners to focus on program logic instead of learning syntax details.
- Python supports both process-oriented and object-oriented programming.
- Python has rich third-party libraries.
- Python can invoke code written in other languages, therefore it is known as "glue language".

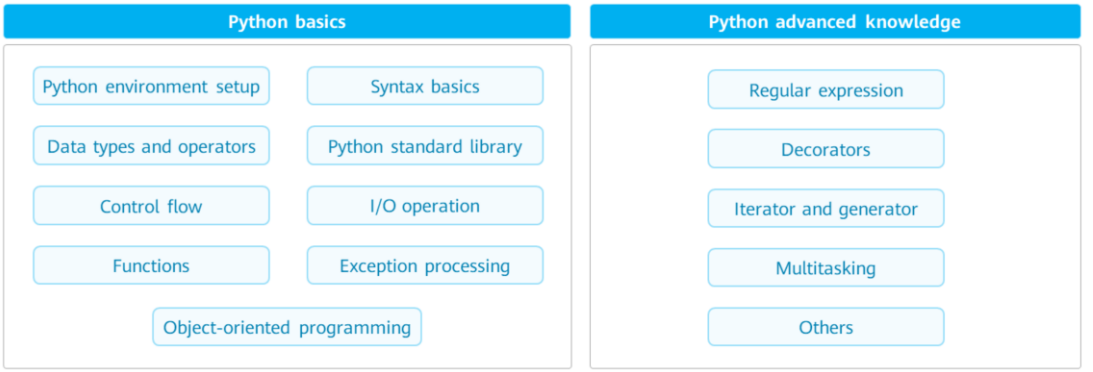
Disadvantages of Python:
- Low running speed. Python is an interpreted language that can run without compilation. The code is translated line by line into machine code that the CPU can understand, which is time-consuming.

In addition to language advantages, Python has abundant third-party libraries. Therefore, Python can be used in many fields, such as AI, data science, apps, and automatic O&M scripts.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Python Learning Map

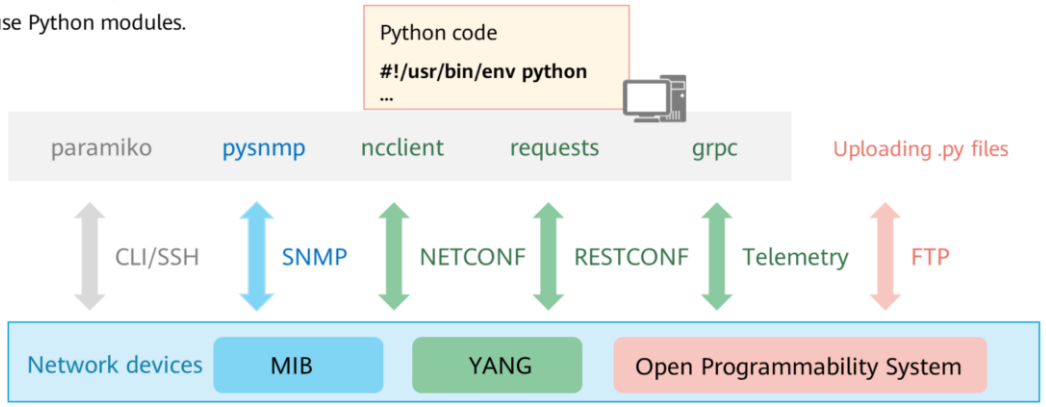
- Python can be used in many fields. To facilitate Python learning and use in the network engineering field, this course divides Python into basics and advanced learning. Python basics enable you to basically implement network programming and automation.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Device Openness and Programmability

- Device openness and programmability aim to provide engineers with guidance on how to implement network automation through programming based on device openness capabilities.
- The following figure shows the open capabilities of Huawei network devices. This course module focuses on how to use Python modules.

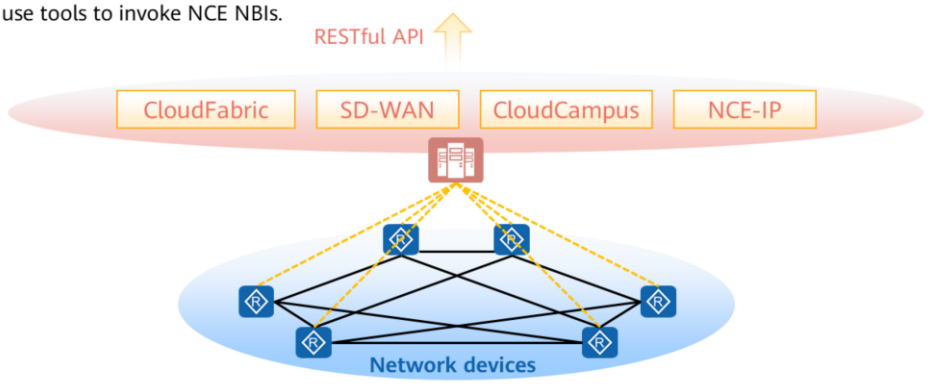


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Part 1 of this course module describes how to use Python modules, including paramiko, pysnmp, ncclient, requests, and grpc, to communicate with devices.
- Part 2 focuses on the OPS. The OPS refers to open programmability provided by Huawei devices. You can upload Python code to a device, and the device runs the code to implement specified functions.

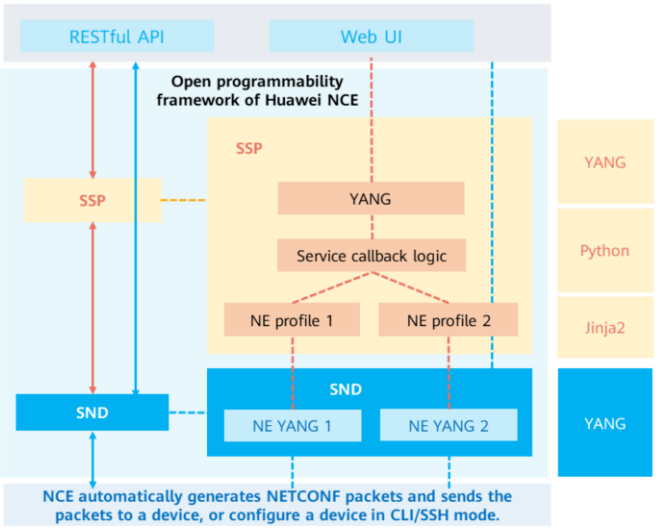
NCE Northbound Openness

- The openness and programmability of the controller provide engineers with guidance on how to implement network automation through programming based on the open capabilities of the SDN controller.
- Huawei iMaster NCE includes controllers and provides northbound RESTful APIs. This course module focuses on how to use tools to invoke NCE NBIs.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Huawei NCE Service Openness and Programmability



- NCE service openness and programmability is a subsystem of NCE. It provides E2E programing capabilities including the openness of NE-layer functions and the openness of network service functions.
- Engineers compile an NE driver package (specific NE driver, SND) and a service driver package (specific service plugin, SSP) to implement the following functions:
  - Multi-vendor device management
  - User-defined network services
  - Web-based service management
  - Automatic exposure of northbound RESTful APIs

Page 23 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- An SND abstracts device capabilities based on a device YANG model. A user can generate an SND based on device YANG files and a few Python code. After the SND is uploaded to NCE, device management and service provisioning can be implemented. SND types include NETCONF SNDs, CLI SNDs, and customized SNDs.
  - NETCONF SND: provides the capability of converting YANG files into NETCONF files.
  - CLI SND: provides the conversion capability from YANG to CLI.
  - Customized SND: provides the capability of converting YANG to other protocols such as RESTCONF.
- An SSP allows user to customize network services (apps), for example, quickly provision L3VPN services. These types of services or application involve multiple devices and protocols and are presented as an SSD. To compile an SSD, an engineer needs to compile service YANG files, Python scripts (service callback logic) for service mapping, and Jinja2 template. The basic principles are as follows (from north to south):
  - A service model automatically generates northbound interfaces or UIs, which are invoked by an external system to initiate a service request.

Quiz

1. (Multiple) Which modules are included in this course? ( )
   A. Programming basics
   B. Device openness and programmability
   C. NCE northbound openness
   D. NCE openness and programmability
2. What are the differences between network automation engineers and network engineers?

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

1. ABCD
2. Network automation engineers: They are professionals who have network skills, competence, and project management experience in the network automation field. They aim to meet the requirements of automatic deployment, development, and O&M of enterprise networks. For details, see P14 to P16.

Summary

- Network programing and automation is to use programming methods to implement automated networks on the premise of open networks. Network openness is implemented at two levels: device openness and SDN platform openness.
- Network automation engineers must be all-rounders who master skills of network engineers, system engineers, and development engineers to some extent to support enterprise network automation.
- This course consists of four modules: programming basics, device openness and programmability, NCE northbound openness, and NCE service openness and programmability. Next, let's learn network programmability.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.


Thank You
www.huawei.com



Python Programming Basics

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Foreword

- Python is a fully open-source high-level programming language with elegant syntax and rich third-party libraries.
- This course aims to help you understand the programming language and Python coding style.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Objectives

- On completion of this course, you will be able to:
  - Describes the classification of programming languages.
  - Describes the Python encoding style.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

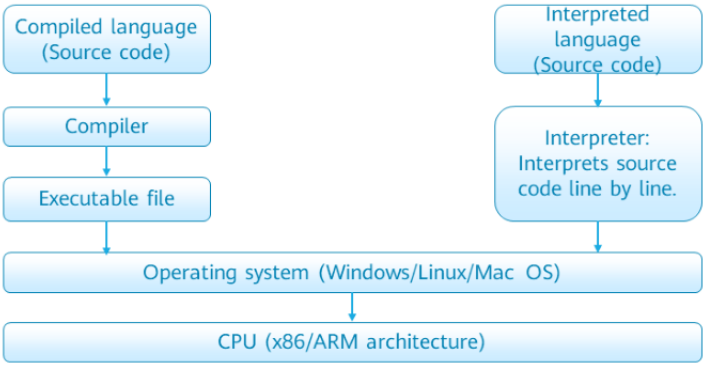
1. **Overview of Programming Language**
2. Python Introduction

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

Programming Languages

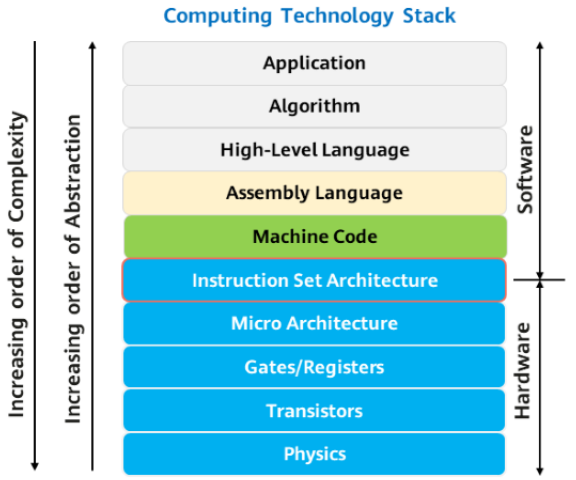
- A programming language is used to write a computer program and control behavior of a computer.
- According to whether compilation is required before execution of a language, the programming language may be classified into the compiled language, and interpreted language that does not need to be compiled.

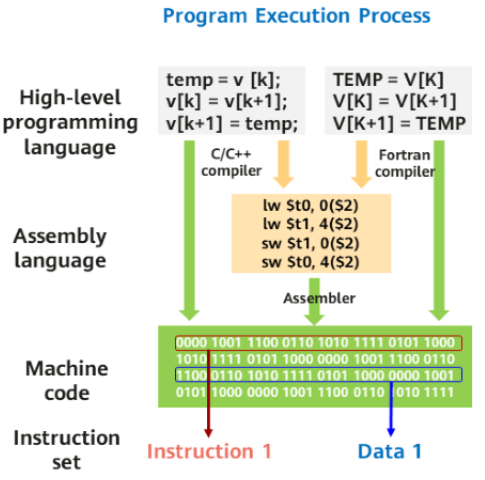


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- Based on language levels, computer languages can also be classified into machine language, assembly language, and high-level language. The machine language consists of 0 and 1 instructions that can be directly identified by a machine. Because machine languages are obscure, hardware instructions 0 and 1 are encapsulated to facilitate identification and memory (such as MOV and ADD), which is assembly language. The two languages are low-level languages, and other languages are high-level languages, such as C, C++, Java, Python, Pascal, Lisp, Prolog, FoxPro, and Fortran. Programs written in high-level languages cannot be directly identified by computers. The programs must be converted into machine languages before being executed.

Computing Technology Stack and Program Execution Process



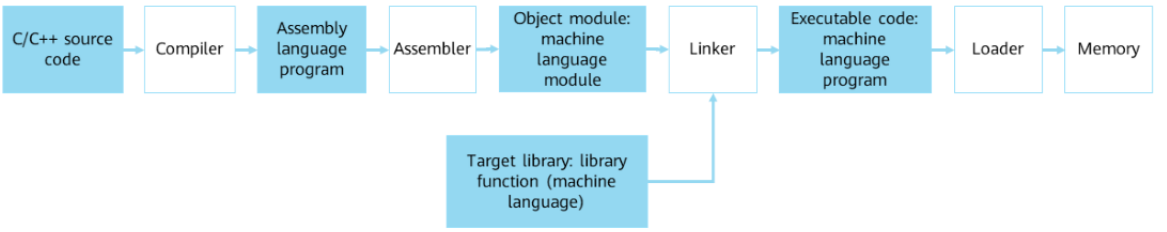


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- A process of executing a computer's technology stack and programs. On the left is the computing technology stack. From the bottom layer of the hardware, physical materials and transistors are used to implement gate circuits and registers, and then the micro architecture of the CPU is formed. The instruction set of the CPU is an interface between hardware and software. An application drives hardware to complete calculation using an instruction defined in the instruction set.
- Applications use certain software algorithms to implement service functions. Programs are usually developed using high-level languages, such as C, C++, Java, Go, and Python. The high-level language needs to be compiled into an assembly language, and then the assembler converts the assembly language into binary machine code based on a CPU instruction set.
- A program on disk is a binary machine code consisting of a pile of instructions and data, that is, a binary file.

High-level Language - Compiled Language

- Compiled language: Before a program in a compiled language is executed, a compilation process is performed to compile the program into a machine language file. The compilation result can be directly used without re-translation during running. Typical compiled languages include C/C++ and Go.
- From source code to program: The source code needs to be translated into machine instructions by the compiler and assembler, and then the linker uses the link library function to generate the machine language program. The machine language must match the instruction set of the CPU, which is loaded to the memory by the loader during running and executed by the CPU.

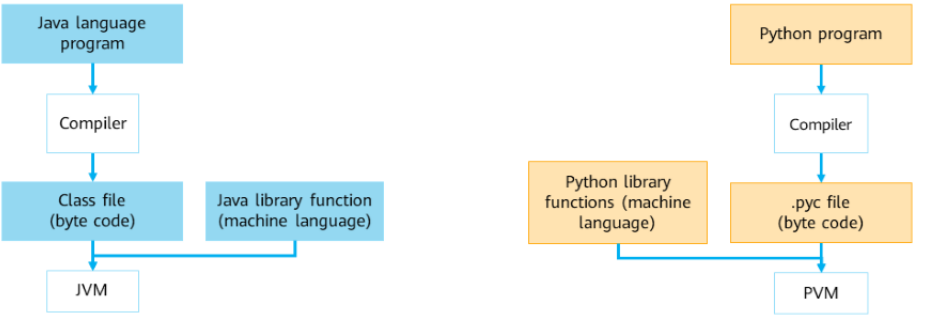


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- Compiled languages are compiled into formats, such as .exe, .dll, and .ocx, that can be executed by machines. Compilation and execution are separated and cannot be performed across platforms. For example, x86 programs cannot run on ARM servers.

High-level Language - Interpreted Language

- Interpreted language: Interpreted language programs do not need to be compiled before running. They are translated line by line when running. Typically, Java and Python are interpreted languages.
- Process from source code to programs: Source code of an interpreted language is generated by the compiler and then interpreted and executed by a virtual machine (VM) (for example, JVM/PVM). The VM shields the differences between CPU instruction sets. Therefore, portability of the interpreted language is relatively good.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- JVM: Java virtual machine
- PVM: Python VM

Contents

1. Overview of Programming Language
2. **Python Introduction**

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

What Is Python?

- Python is a fully-open-source high-level programming language. Its author is Guido Van Rossum.

Advantages of Python:
- Is a dynamically typed interpreted language with elegant syntax. It allows learners to focus on program logic rather than syntax detail learning.
- Supports both process- and object-oriented programming.
- Provides abundant third-party libraries.
- Is nicknamed the glue language because it can call code written in other languages.

Disadvantages of Python:
- Runs slow. Is an interpreted language that runs without being compiled. Code is translated line by line at run time into machine code that the CPU can understand, which is time-consuming.

With support for abundant third-party libraries and advantages of the Python language, Python can be used in many fields, such as AI, data science, apps, and scripts for automated O&M.

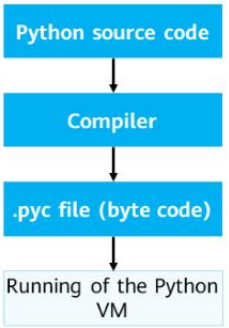
 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- Python is also a dynamically typed language. The dynamically typed language automatically determines the type of variable during program running. The type of a variable does not need to be declared.

Python Code Execution Process

Process of compiling and running a Python program



Operation

1. Install Python and the running environment in an operating system.
2. Compile Python source code.
3. The compiler runs the Python source code and generates a .pyc file (byte code).
4. A Python VM converts the byte code into the machine language.
5. Hardware executes the machine language.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- Python source code does not need to be compiled into binary code. Python can run programs directly from the source code. When Python code is run, the Python interpreter first converts the source code into byte code, and then the Python VM executes the byte code.
- The Python VM is not an independent program and does not need to be installed independently.

Getting Started with Python Code – Interactive Running

- Python runs in either interactive or script mode.
- Interactive programming does not require script files to be created, and code is written in the interactive mode of the Python interpreter.

```
C:\Users\Richard>python
Python 3.7.4 (default, Aug  9 2019, 18:34:13) [MSC v.1915 64 bit (AMD64)] ::
Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
1. Input  -- >>> print ("hello world")
2. Output -- hello world
3. Input  -- >>> a = 1
4. Input  -- >>> b = 2
5. Input  -- >>> print ( a + b )
6. Output -- 3
>>>
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Getting Started with Python Code – Script-based Running

- Code in script mode can run on various Python compilers or in integrated development environments. For example, IDLE, Atom, Visual Studio, Pycharm, and Anaconda provided by Python can be used.

demo.py

```
print ("hello world")
a = 1
b = 2
print ( a + b )
```

1. Input --
2. Output --
3. Output --

```
C:\Users\Richard>python demo.py
hello world
3
```

1 Write a Python script file (.py).

2 Execute the script file.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Code Style Guide for Python

- Code style rules refer to naming rules, code indentation, and code and statement segmentation modes that must be complied with when Python is used to write code. Good style rules help improve code readability and facilitate code maintenance and modification.
- For example, the following rules for using semicolons, parentheses, blank lines, and spaces are recommended:

Semicolon

- A semicolon can be added at the end of a line in Python, but is not recommended to separate statements.
- It is recommended that each sentence be in a separate line.

Blank line

- Different functions or statement blocks can be separated by spaces. A blank line helps differentiate two segments of code, improving code readability.

Parentheses

- Parentheses can be used for the continuation of long statements. Avoid unnecessary parentheses.

Space

- Spaces are not recommended in parentheses.
- You can determine whether to add spaces on both ends of an operator.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Identifier Naming

- A Python identifier represents the name of a constant, variable, function, or another object.
- An identifier is usually composed of letters, digits, and underscores, but cannot start with a digit. Identifiers are case sensitive and must be unique. If an identifier does not comply with the rules, the compiler will output a SyntaxError message when running the code.

1. Assign a value	--	User_ID = 10	print (User_ID)
2. Assign a value	--	user_id = 20	print (user_id)
3. Assign a string	--	User_Name = 'Richard'	print (User_Name)
4. Assign a value	--	Count = 1 + 1	print (Count)
5. Incorrect identifier	--	4_passwd = "Huawei"	print (4_passwd)

print() is a built-in function of Python and is used to output content in parentheses.

Question: What is the output of the print command on the right?

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- Basic data types of Python are Boolean (True/False), integer, floating point, and string. All data (Boolean values, integers, floating points, strings, and even large data structures, functions, and programs) in Python exists in the form of objects. This makes the Python language highly unified.
- The execution results are 10, 20, Richard, 2, and SyntaxError, respectively.
- This presentation does not describe Python syntax. For Python syntax details, see the HCIP course.

Code Indentation

- In Python programs, code indentation represents the scope of a code block. If a code block contains two or more statements, the statements must have the same indentation. For Python, code indentation is a syntax rule that uses code indentation and colons to distinguish between layers of code.
- When writing code, you are advised to use four spaces for indentation. If incorrect indentation is used in the program code, an IndentationError error message is displayed during code running.

Correct indentation --

Correct indentation --

Incorrect indentation --

```
if True:
    print ("Hello")
else:
    print (0)

a = “Python”
    print (a)
```

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- if...else... is a complete block of code with the same indentation.
- print(a) calls parameter a, and it is in the same code block with if...else...clause.

Using Comments

- Comments are explanations added to programs to improve program readability. In the Python program, comments are classified into single-line comments and multi-line comments.
- A single-line comment starts with a pound sign (#).
- A multi-line comment can contain multiple lines, which are enclosed in a pair of three quotation marks ('''...''' or """ ...""").

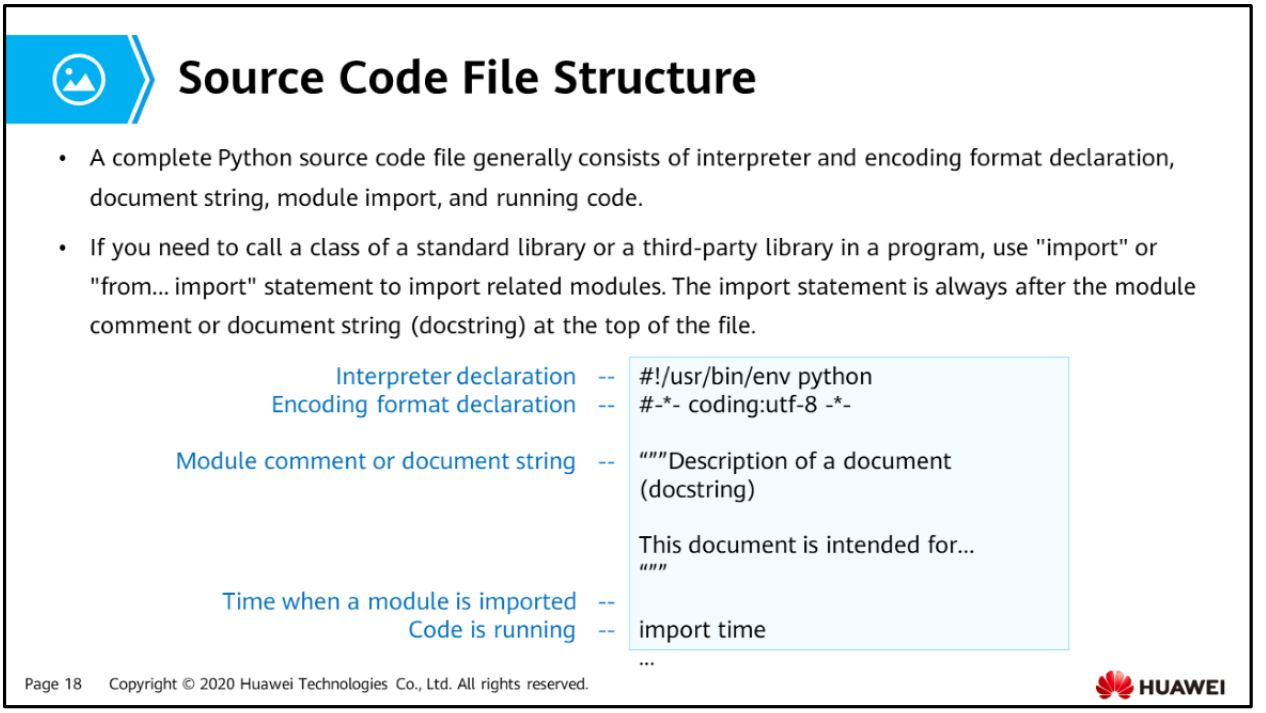
Single-line comment --

Multi-line comment --

```
# Assign a string to a.
a = “Python”
print (a)

“””
The output is Python.
“””
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI



- The interpreter declaration is used to specify the path of the compiler that runs this file (the compiler is installed in a non-default path or there are multiple Python compilers). In the Windows , you can omit the first line of the interpreter declaration in the preceding example.
- The encoding format declaration is used to specify the encoding type used by the program to read the source code. By default, Python 2 uses ASCII code (Chinese is not supported), and Python 3 supports UTF-8 code (Chinese is supported).
- docstring is used to describe the functions of the program.
- time is a built-in module of Python and provides functions related to processing time.

Python Functions and Modules

- A function is a block of organized, reusable code that is used to perform a single, related action. It can improve the modularity of the program and code utilization. The function is defined using the def keyword.
- A module is a saved Python file. Modules can contain definitions of functions, classes, and variables that can then be utilized in other Python programs. The only difference between a module and a regular Python program is that the module is used for importing by other programs. Therefore, a module usually does not have a main function.

demo.py

```
def sit(): #Define a function.
    print ('A dog is now sitting')

sit() #Call a function.
```

Execution result:

```
A dog is now sitting.
```

1 Write a Python file (.py).

test.py

```
import demo #Import a module.

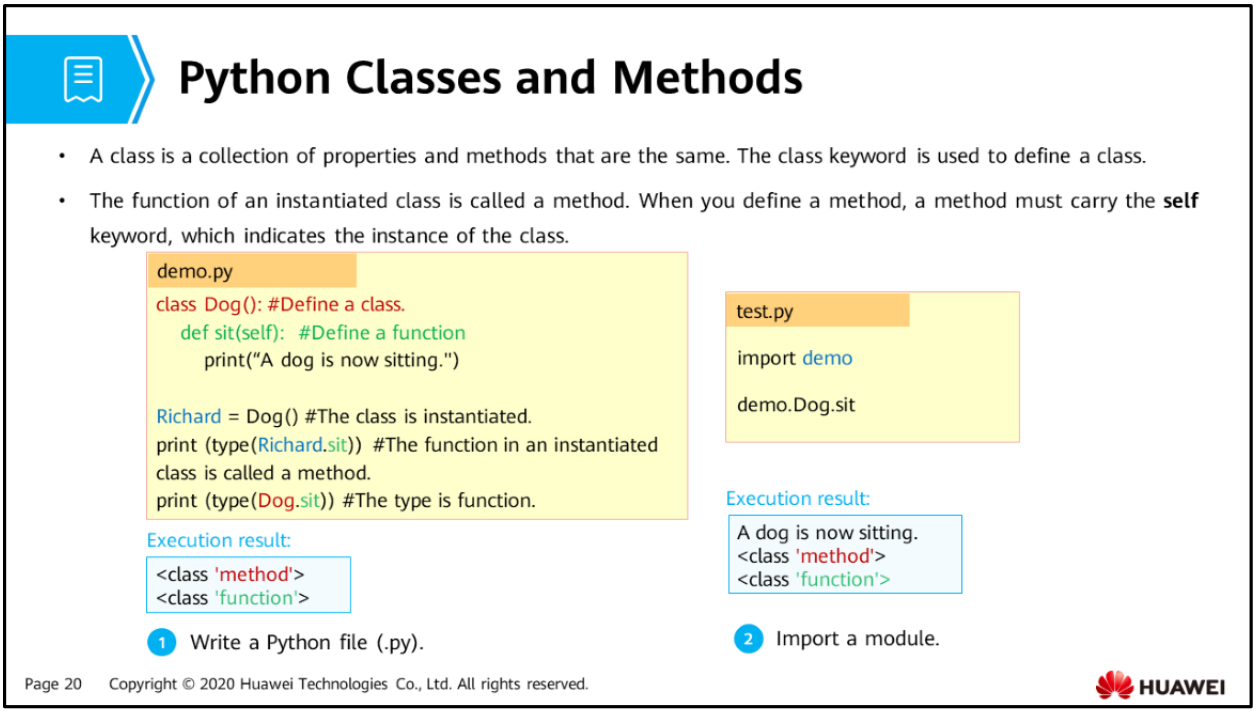
demo.sit()      #Call a function.
```

Execution result:

```
A dog is now sitting.
A dog is now sitting.
```

2 Import a module.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

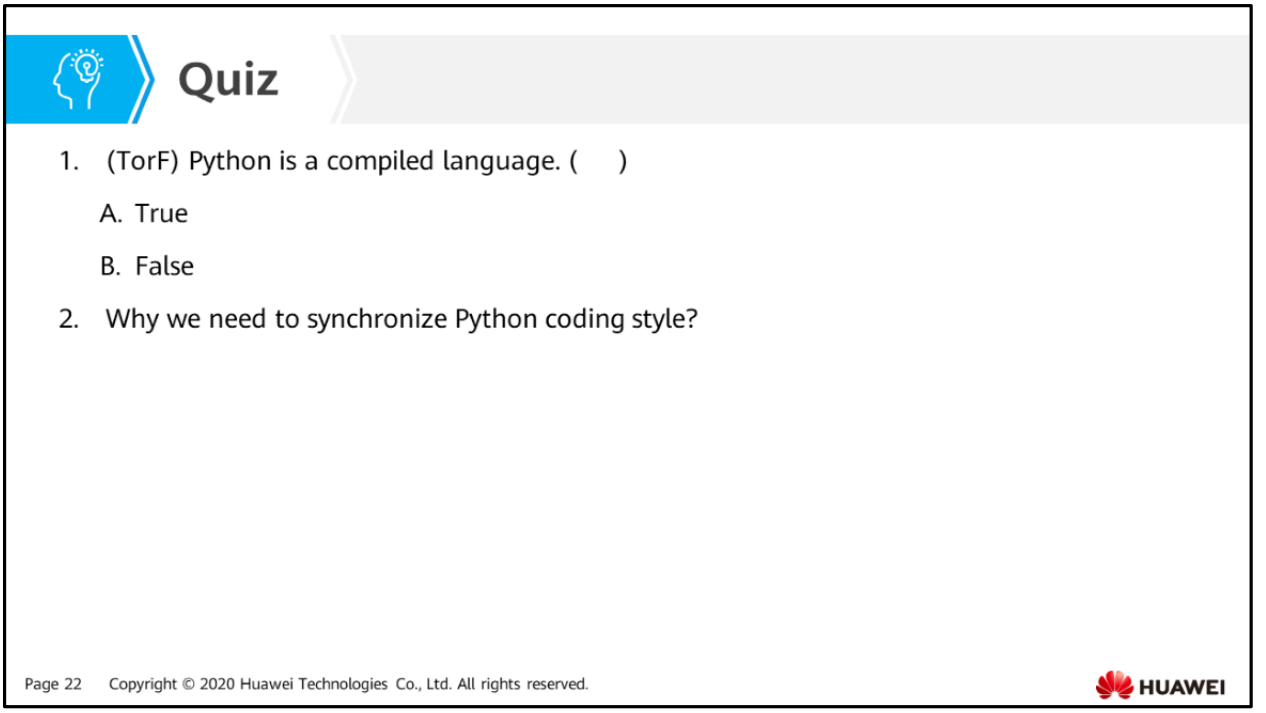


- Official definitions of functions and methods:
- A series of statements which returns some value to a caller. It can also be passed zero or more arguments which may be used in the execution of the body.
- A function which is defined inside a class body. If called as an attribute of an instance of that class, the method will get the instance object as its first argument (which is usually called self).
- For more information about classes, see https://docs.python.org/3/tutorial/classes.html.

Summary

- This section describes the Python language and its coding style.
- To better understand the basic and advanced Python syntax, learn HCIP-Datacom-Python Programming Basics and experience the charm of Python in practice.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



1. B
2. Coding style refer to the naming rules, code indentation, and code segmentation methods that must be followed when Python is used. It is very important to unify coding style in a team. Good coding style help improve code readability, facilitate code maintenance and modification, and improve team work efficiency.

More Information

- For more information about Python, visit https://www.python.org/.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.


Thank You
www.huawei.com



Git Fundamentals and Practices

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Foreword

- Since software is expanding in scale and becoming increasingly complicated, developers have higher requirements on version control during software development.
- There are various version control tools in the industry, such as ClearCase, Visual SourceSafe (VSS), Subversion (SVN), and Git. This course describes the basic concepts and operations of the popular open-source tool Git, and introduces the code hosting practice on HUAWEI CLOUD.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- ClearCase: developed by Rational Software
- Visual SourceSafe (VSS): developed by Microsoft
- Subversion (SVN): open-source software
- Git: open-source software

Objectives

- Upon completion of this course, you will be able to:
  - Describe the differences between centralized and distributed version control systems.
  - Describe Git basic concepts.
  - Perform basic Git operations.
  - Use the GUI-based Git client TortoiseGit.
  - Perform code hosting on HUAWEI CLOUD.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

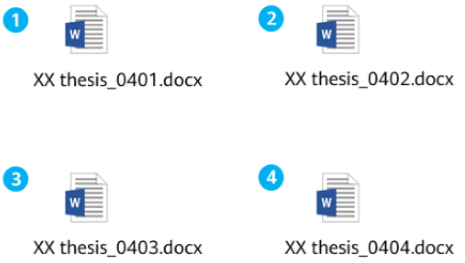
Contents

1. **Version Control Overview**
2. Git Basic Concepts
3. Basic Git Operations: Clone, Pull, Push, and Merge
4. HUAWEI CLOUD-based Code Hosting

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Begin with Your Thesis

- Multiple copies with different modifications:



Question: How to solve this problem?

Pain point:

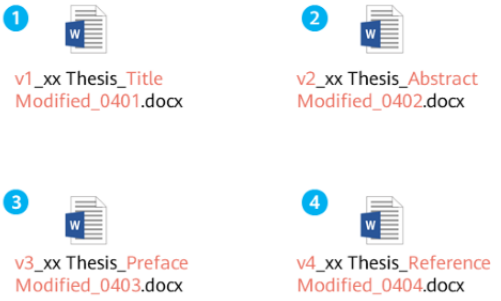
- Multiple copies with different modifications are available. To revert certain modifications, you need to open these files one by one to search for the desired modifications.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- You may have multiple copies for your thesis, and it is very inconvenient to revert certain changes.

Version Control: Add Change Descriptions

- Adding a description for each copy facilitates you to recognize what changes are made in a copy.
- Since multiple copies may be generated in a day, to arrange all these copies in a timeline, specify a version number for each copy.

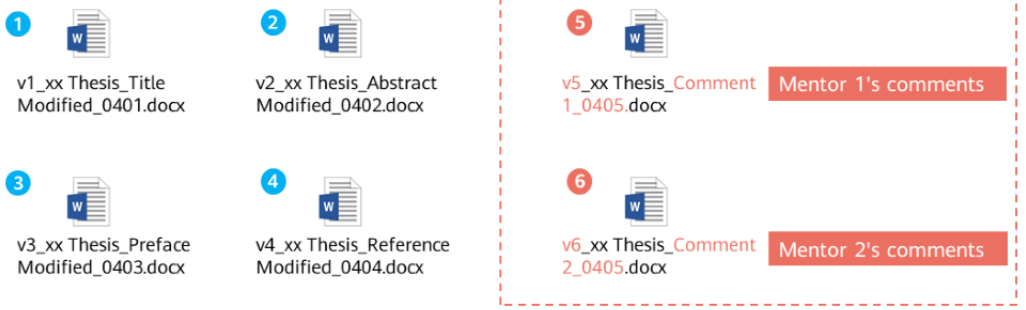


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Adding a description and version number to a file name allows you to quickly know the modifications made on each copy and how do all these copies evolve.

New Problem: What If Multiple Users Modify a Single File

- When a thesis is reviewed and modified by several mentors, a new problem arises.



Pain point:

- The file name does not present the modifier information. In this case, it is difficult to determine the modifier directly through the file name.

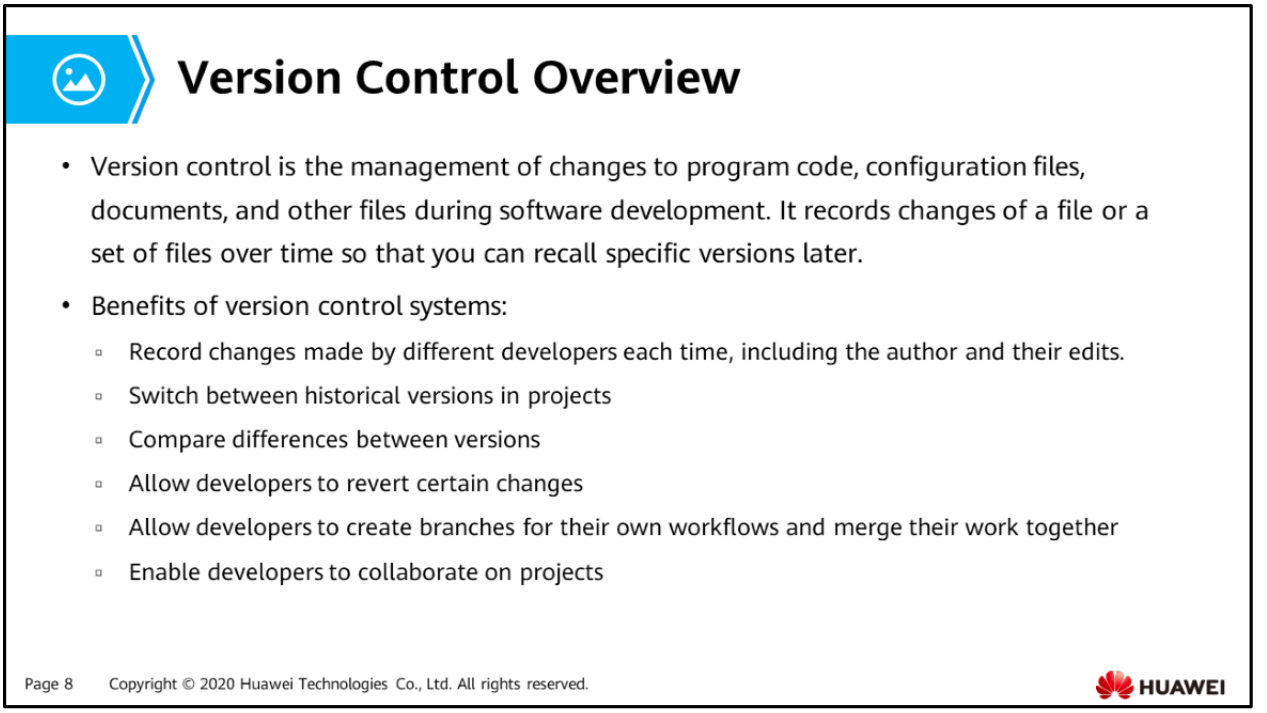
Add the modifier information

v5_xx Thesis_Mentor Zhang_Comment 1_0405.docx
v6_xx Thesis_Mentor Li_Comment 2_0405.docx

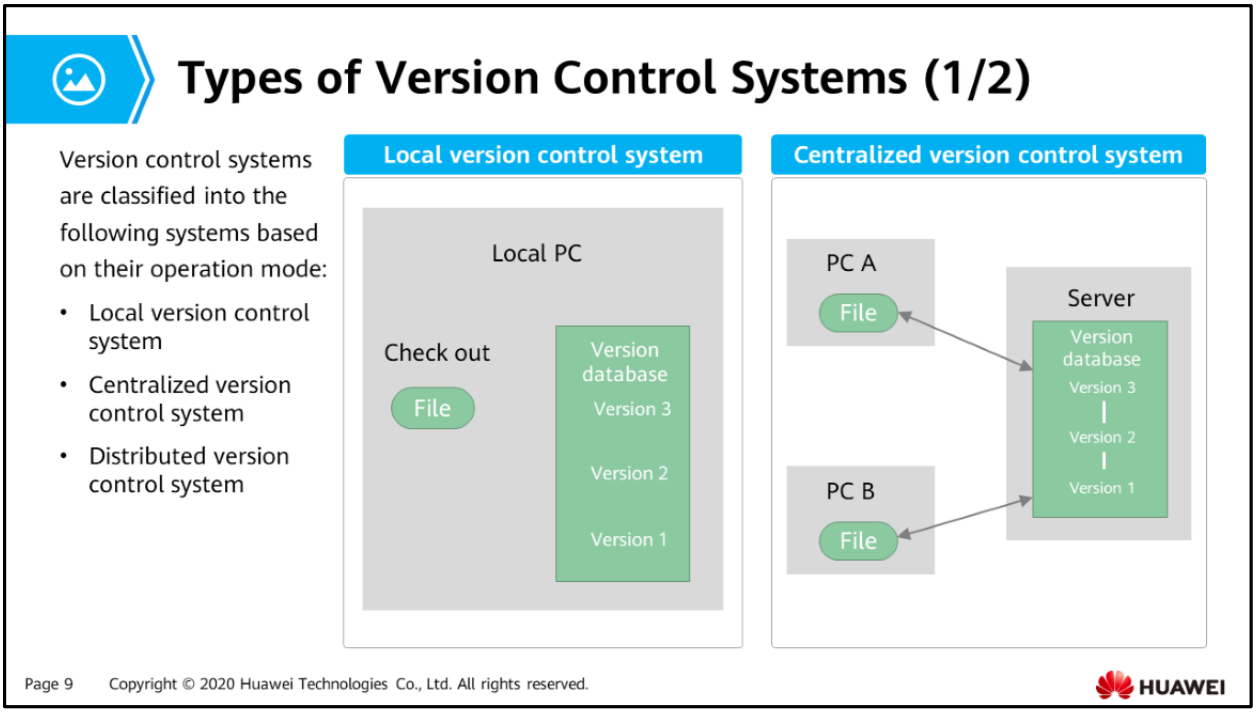
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- When multiple users modify the same file, the file name needs to contain the information about the modifiers.
- Version control systems provide the similar function to the file name in this example.

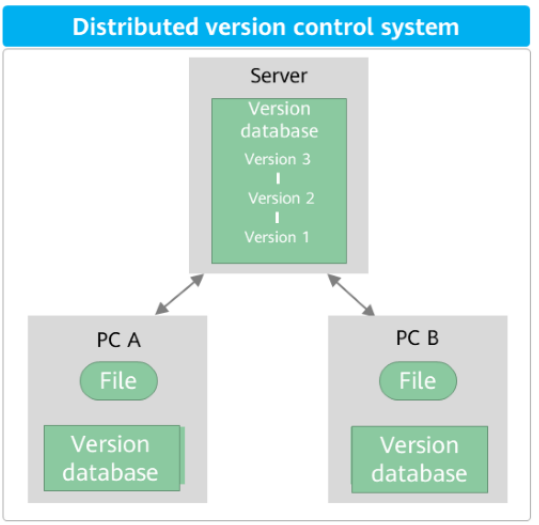


- Version control enables you to roll back a file or the entire project to a previous version, compare differences in files between versions, trace the author of each change, and restore files to the normal status if a bug occurs. Compared with manual operations, version control greatly reduces the workload.



- There are many version control systems in the industry, which are classified into three types: local version control systems, centralized version control systems, and distributed version control systems.
- A local version control system, such as Revision Control System (RCS), is installed on the local PC and has a simple database to keep all file changes. It can re-produce file contents based on the recorded changes, ensuring that no content is lost. Since files are saved on local PCs, multiple users cannot collaborate to work on these files. Considering this drawback, this system is seldom used.
- A centralized version control system, such as Subversion (SVN), has a central server saved with all changes to files. Users connect to this server from clients on their local PCs to retrieve the latest files or commit their changes. Compared with local version control systems, centralized version control systems allow developers to collaborate on projects and allow administrators to control the rights of each developer. However, if a single point of server failure occurs, for example, if the server goes down or the network between the server and developers is disconnected, developers cannot commit their changes to the server. If the server disk is damaged and proper backups have not been kept, all project data will be lost, including the entire project change history. The central server is the heart of centralized version control systems because it stores all information about versioned files. Developers need to retrieve the latest files from the central server before they start to work every day and commit their changes to the server after resolving conflicts if there is any. In this case, developers cannot work without the central server.

Types of Version Control Systems (2/2)



- Local version control system:
  Drawback: All files are stored on one local PC. Multiple users cannot collaborate to work on the files.
- Centralized version control system:
  Drawback: This system requires network connectivity; therefore, developers are unable to work offline. If the central server fails, developers cannot collaborate and even data lost may occur; therefore, security cannot be assured.

- Distributed version control system:
  This system comes into picture to overcome the preceding drawbacks. Developers clone the version database to their own PCs. In this case, each developer owns all versioned files.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- The most obvious difference between distributed version control systems and centralized version control systems is that clients not only retrieve snapshots of the latest files, but also clone the entire remote repository to their local PCs. If developers fail to connect to the remote repository, they can commit changes to their local repository and roll back the files to any historical version. In centralized version control systems, however, since all file changes are stored on the central server, if the server fails or the network is disconnected, clients cannot retrieve file change records from the server. In distributed version control systems, if the remote server dies, any of the client's local repositories can be copied to the server to help restore the server.

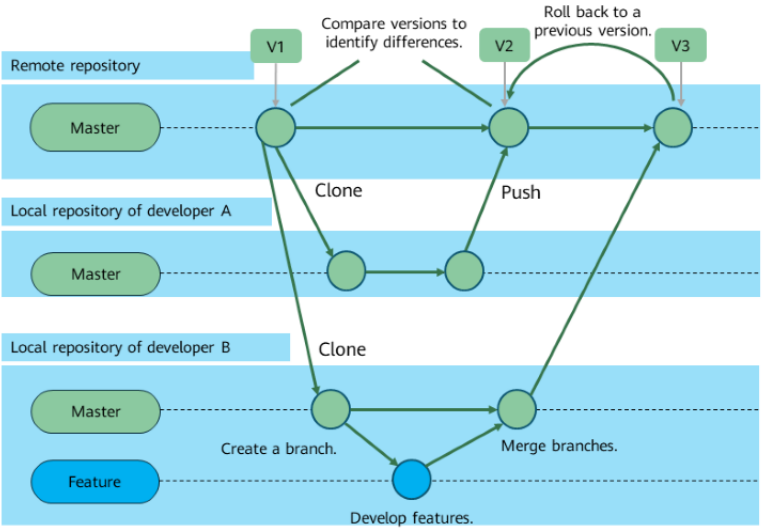
DISTR VCS: Basic Concepts

- Basic concepts:
  - **Remote repository**: a code repository created on the server. It can be described as the heart of the system, where all developers collaborate and store their code.
  - **Local repository**: a code repository on developers' local PCs. Developers write code in their local repositories and then push their changes to the remote repository. In addition, they can retrieve code from the remote repository to obtain changes made by other developers.
  - **Branch**: a copy of code in a repository. Developers can write code in a branch and merge their changes to the target branch. This eliminates the impact on code in branches other than the one that has been checked out during feature development.
  - **Clone**: This operation copies the remote repository to the local PC.
  - **Push**: This operation integrates changes in the local repository to the remote repository.

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- VCS：Version Control System

DISTR VCS: Fundamentals



 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- The preceding figure describes how developers work with a distributed version control system. A remote repository is a server that stores project code and enables all developers in a team to collaborate. A local repository stores code on developers' local PCs.
- Developers A and B pull code from the remote repository to their respective local repository. Master indicates the master branch. Developer B does not develop new features in the master branch, but instead creates a feature branch, which can be considered as a copy of code in the master branch. Developer B then develops new features in the feature branch. After all features are developed, developer B merges the Feature branch into the master branch, that is, merges the newly developed features to the master branch.
- Version control systems enable you to roll back code to any version. For example, if there is a bug in V3, you can easily roll back code to V2. In addition, you can compare versions to identify code differences and check what code is incorporated. Since version control systems record each code submission, you can easily find who makes what changes at what time.

Contents

1. Version Control Overview
2. **Git Basic Concepts**
3. Basic Git Operations: Clone, Pull, Push, and Merge
4. HUAWEI CLOUD-based Code Hosting

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

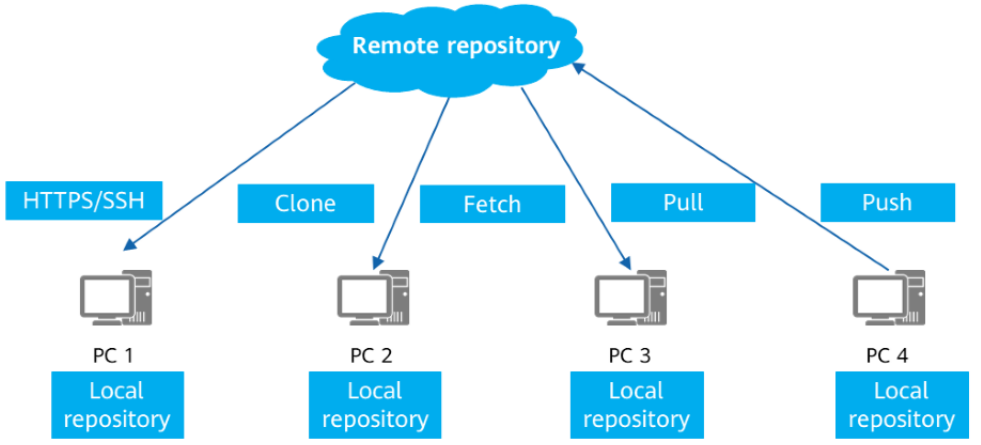
Git Overview

- Git is an open-source distributed software version control system for tracking code changes during software development.
- It was developed by the open source community led by Linus Torvalds in 2002.
- It was originally developed to support the massive open-source code of the Linux kernel.
- It was designed based on the following principles:
  - Fast and easy to design
  - Strong support for non-linear development and thousands of parallel branches
  - Fully distributed
  - Able to efficiently handle large projects like the Linux kernel

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Git Application Scenarios

- Collaborative software development



 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Git supports collaboration between developers during software development. Developers install the Git client on their local PCs, clone the code from the remote repository to their respective local repository, pull the updated code from the remote repository, and push their changes to the remote repository. The data in the local and remote repositories can be transferred using SSH or HTTPS.
- Clone: This operation helps you copy the remote repository to a folder on your local PC. All branches on the remote repository are cloned, and the branch that you are working on will be checked out to a folder on the local PC.
- Fetch: This operation downloads version change records from the remote repository to your local repository but does not check out the changes to the working directory.
- Pull: This operation downloads version change records from the remote repository to your local repository and checks out the changes to the working directory.
- Push: This operation pushes changes in your local repository to target branches in the remote repository.

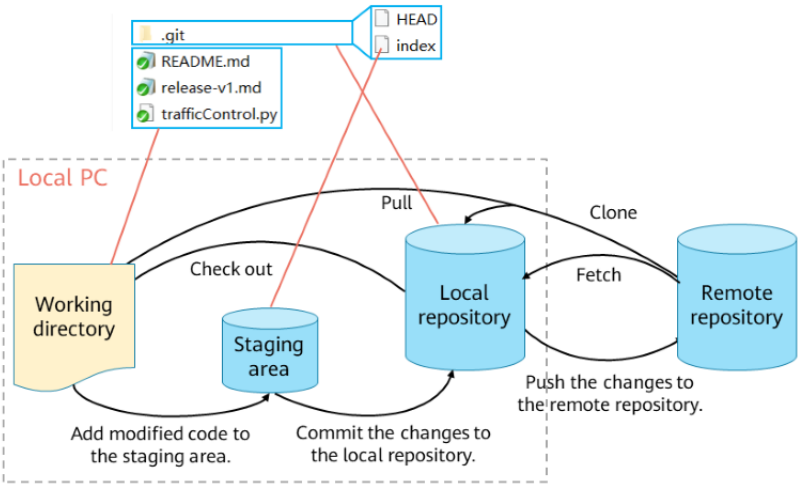
Git Basic Concepts (1/2)

- Before learning Git, you need to understand the following basic concepts:
  - Remote repository: a repository for storing versioned files on the remote server.
  - Local repository: a repository for storing versioned files on a local server.
  - Working directory: developer's working copy. Changes in the working directory can be committed to the local repository.
  - Staging area: area between a working directory and a local repository. Before committing changes to a local repository, add the changes to the staging area first.
  - Branch: a copy of code in a repository.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Git Basic Concepts (2/2)

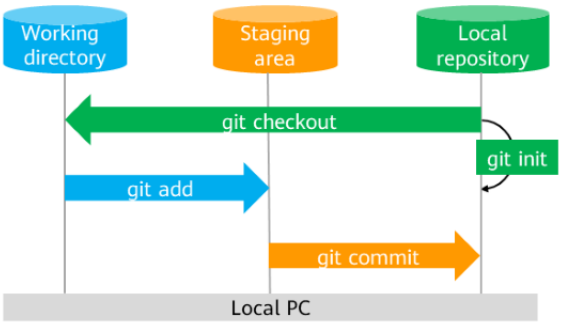
- Checkout: checks out code of a certain version from the local repository to the working directory.
- Fetch: downloads code changes from the remote repository to your local repository but does not check out the changes to the working directory.
- Pull: downloads code changes from the remote repository to your local repository and checks out the changes to the working directory.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- Modified files in the working directory need to be added to the staging area first and then committed to the local repository.
- Checkout: This operation checks out code of a certain version from the local repository to the working directory.

Git Local Working Areas



Three areas exist on the Local PC installed with Git:

1. Working directory: a working copy of files checked out from the local repository.
2. Staging area: isolates the working directory from the local repository and stores files with changes to be committed in the **.git/index** file, in most cases.
3. Git repository: local repository on the local PC, which is a hidden directory named **.git**.

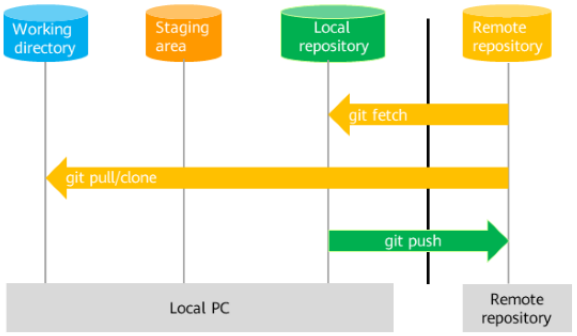
Related commands:

- git init: creates a local repository.
- git add: adds files with changes from the working directory to the staging area.
- git commit: commits the changes that have been staged to the local repository.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- The local repository is the repository cloned from the remote repository on the local PC.
- Three areas exist on the local PC installed with Git: the working directory, staging area, and local repository.
- All historical snapshots are saved in the local repository.
- The working directory is your working copy of files of a certain version from the local repository. You can modify the files in the working directory.
- The staging area stores files with changes to be committed to the local repository.

Remote Repository



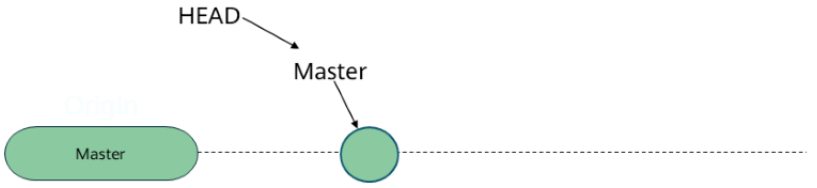
- The remote repository is a Git server (also known as origin). Technically, a remote repository does not differ from a local one.
- It saves code permanently and allows developers to collaborate on projects.
- Differences between remote and local repositories:
  - Connection to the remote repository with the URL
  - SSH access to the remote repository
- Git-based repository managers:
  - Outside China: GitHub and GitLab
  - In China: CodeHub (HUAWEI CLOUD), Gitee, etc.
- Related commands:
  - git clone: clones the remote repository.
  - git push: pushes files from a local branch to the remote repository.
  - git fetch: copies the information that is in the remote repository but is not in the local repository.
  - git pull: copies the information that is in the remote repository but is not in the local repository, and merge it to a local branch.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- The remote repository saves code permanently and allows developers to collaborate on projects. Basic operations relating to the remote repository include clone, fetch, pull, and push.
- The **git clone** command copies the remote repository to the local PC whereas and the **git pull** command copies only the updates involved in the branch that you are working on.

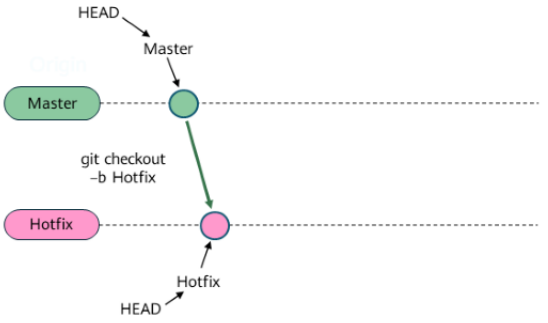
Branching: Creating the Master Branch

- A branch in Git is simply a pointer. The master branch has a master pointer that points to the last commit you made. Other branches also have their own pointers. The HEAD points to the branch that you are currently on.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Branching: Creating a Hotfix Branch



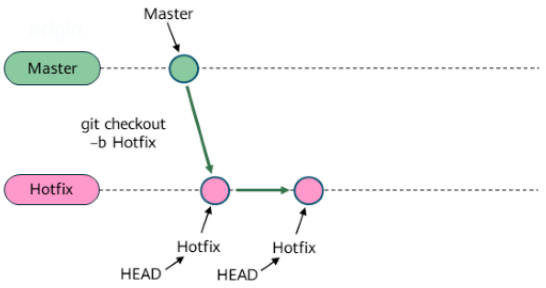
Related commands:

- git checkout -b *<name>*: creates a branch and switch to it.
- git branch *<name>*: creates a branch.
- git switch *<name>*: switches to a branch.

- A branch in Git is simply a pointer. The master branch has a master pointer that points to the last commit you made. Other branches also have their own pointers. The HEAD points to the branch that you are currently on.
- Branch creation: Create branches that are separated from the master branch (which is the mainstream branch), enabling you to start an independent development line. Branches are not physically copied. The files with changes in branches are saved only when they are committed.
- Branch switching: You can switch to different branches in the same repository under the same working directory by modifying the HEAD. When you switch branches, Git replaces the content of your working directory with the last committed snapshot of the destination branch.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Writing Code in the Hotfix Branch



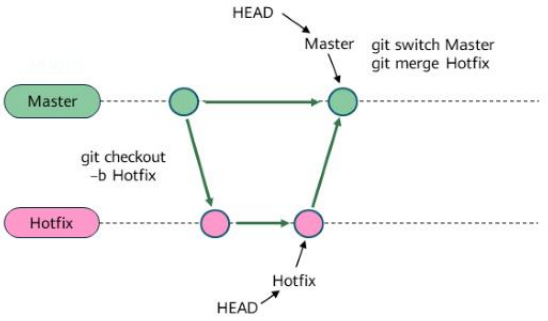
Related commands:

- git checkout -b *<name>*: creates a branch and switch to it.
- git branch *<name>*: creates a branch.
- git switch *<name>*: switches to a branch.

- A branch in Git is simply a pointer. The master branch has a master pointer that points to the last commit you made. Other branches also have their own pointers. The HEAD points to the branch that you are currently on.
- Branch creation: Create branches that are separated from the master branch (which is the mainstream branch), enabling you to start an independent development line. Branches are not physically copied. The files with changes in branches are saved only when they are committed.
- Branch switching: You can switch to different branches in the same repository under the same working directory by modifying the HEAD. When you switch branches, Git replaces the content of your working directory with the last committed snapshot of the destination branch.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Merging Branches



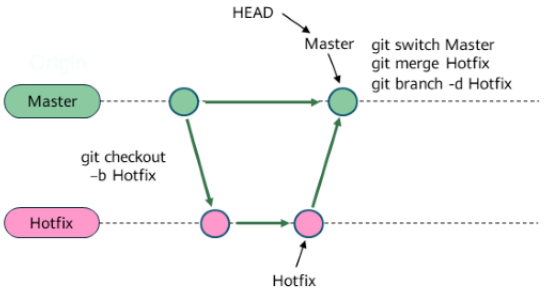
Related commands:

- git checkout -b *<name>*: creates a branch and switch to it.
- git branch *<name>*: creates a branch.
- git switch *<name>*: switches to a branch.
- git merge *<name>*: merges a branch into the branch you have checked out.

- A branch in Git is simply a pointer. The master branch has a master pointer that points to the last commit you made. Other branches also have their own pointers. The HEAD points to the branch that you are currently on.
- Branch creation: Create branches that are separated from the master branch (which is the mainstream branch), enabling you to start an independent development line. Branches are not physically copied. The files with changes in branches are saved only when they are committed.
- Branch switching: You can switch to different branches in the same repository under the same working directory by modifying the HEAD. When you switch branches, Git replaces the content of your working directory with the last committed snapshot of the destination branch.
- Branch merging: Merge the updates in one branch to another.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Deleting the Hotfix Branch



Related commands:

- git checkout -b *<name>*: creates a branch and switch to it.
- git branch *<name>*: creates a branch.
- git switch *<name>*: switches to a branch.
- git merge *<name>*: merges a branch into the branch you have checked out.
- git branch -d *<name>*: deletes a branch.

- A branch in Git is simply a pointer. The master branch has a master pointer that points to the last commit you made. Other branches also have their own pointers. The HEAD points to the branch that you are currently on.
- Branch creation: Create branches that are separated from the master branch (which is the mainstream branch), enabling you to start an independent development line. Branches are not physically copied. The files with changes in branches are saved only when they are committed.
- Branch switching: You can switch to different branches in the same repository under the same working directory by modifying the HEAD. When you switch branches, Git replaces the content of your working directory with the last committed snapshot of the destination branch.
- Branch merging: Merge the updates in one branch to another.
- Branch deletion: Delete the pointer of a branch.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. Version Control Overview
2. Git Basic Concepts
3. **Basic Git Operations: Clone, Pull, Push, and Merge**
4. HUAWEI CLOUD-based Code Hosting

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

Introduction to Git Clients

- Git comes with clients based on the command line interface (CLI) and graphical user interface (GUI). The Git software provides a CLI by default.
- Multiple GUI-based Git clients are available, such as TortoiseGit, SourceTree, SmartGit, and GitEye. TortoiseGit is used here as an example.



TortoiseGit



SourceTree

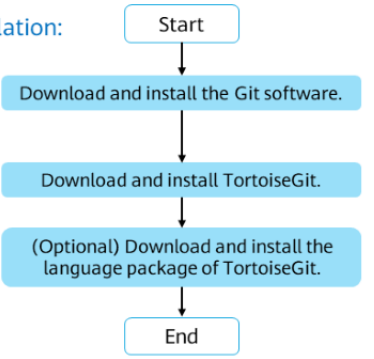


SmartGit



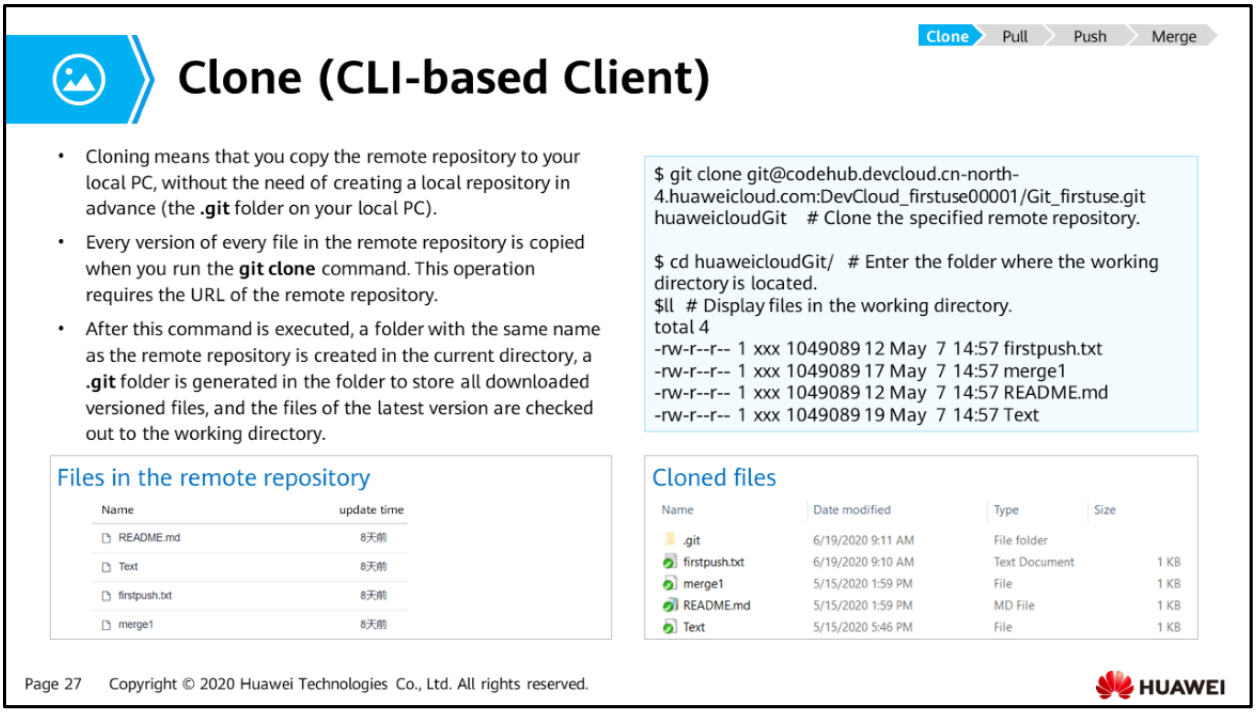
GitEye

Git installation:



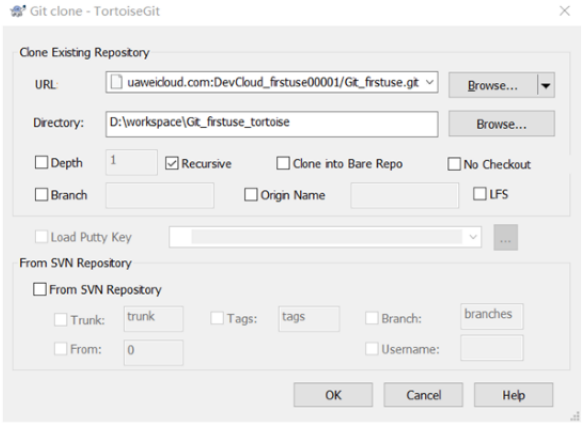
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- TortoiseGit provides a Chinese language package for installation.

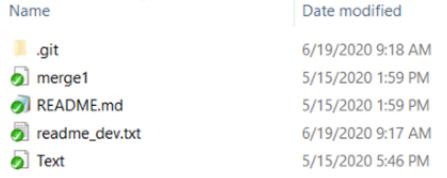


- Git clones a full copy of nearly all data that the remote repository has, but not just the files you need to work on. Every version of every file in the remote repository is copied by default when you run the **git clone** command. If a disk on the server where the remote repository resides gets corrupted, you can use any of the clones on any client to set the server back to the state it was when it was cloned.
- You can use SSH or HTTPS when running the **git clone** command. SSH is used as an example here.

Clone (TortoiseGit)

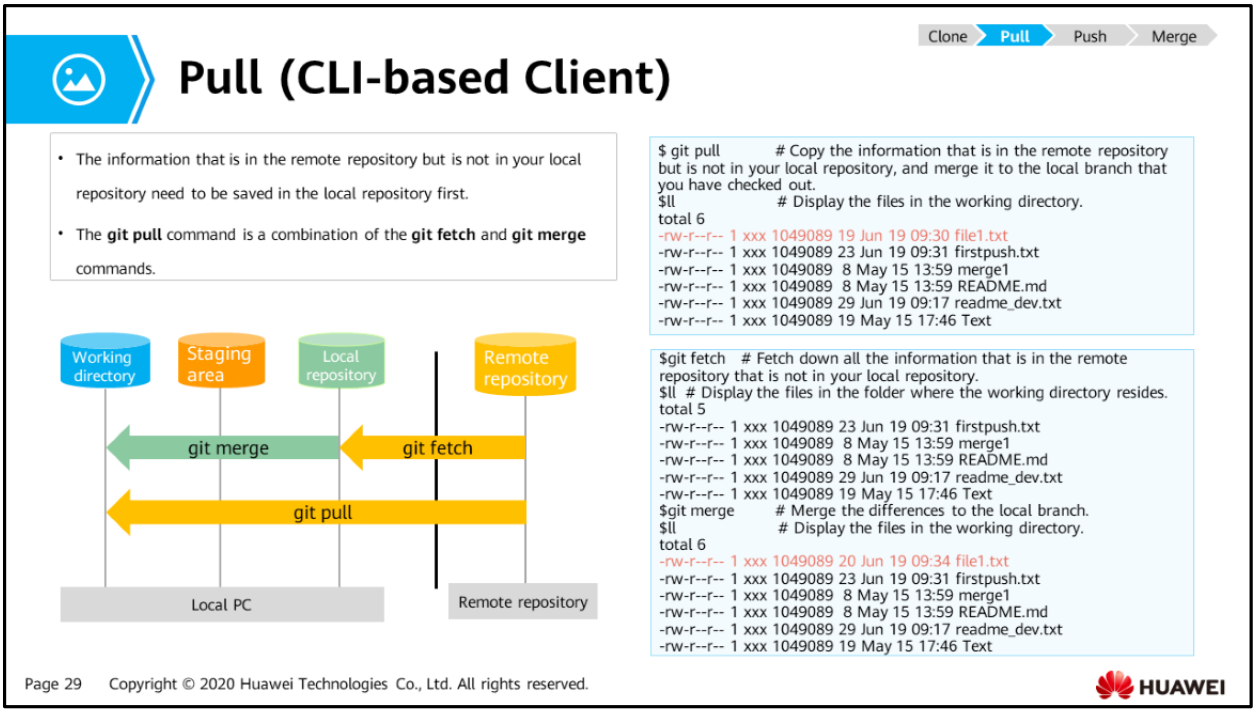


- URL: Remote repository URL
- Directory: Folder where the working directory resides



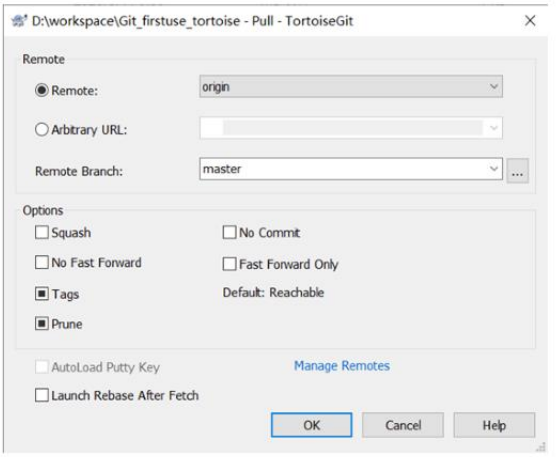
 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

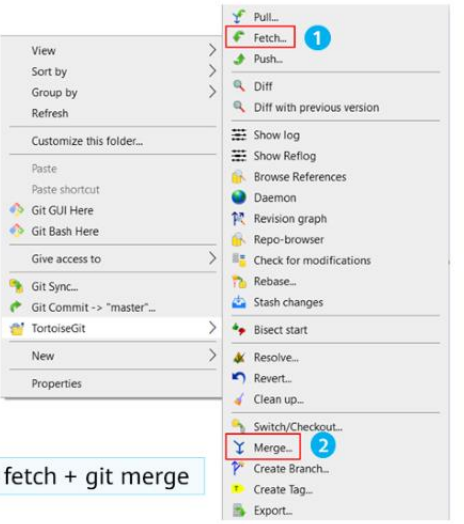


- The **git pull** command pulls the information that is in the remote repository but is not in the local repository, and merge it to the working directory of the branch that you have checked out. The **git pull** command is a combination of the **git fetch** and **git merge** commands. The difference between the **git pull** and **git fetch** commands lies in that the **git fetch** command only copies the information that is in the remote repository but is not in the local repository and does not merge it to the corresponding local branch, whereas the git merge command merges the fetched information to the working directory of the branch that you have checked out.
- In this example, the **file1.txt** file is added in the remote repository. You can run the **git pull** command to copy the **file1.txt** file from the remote repository to the working directory on the local PC. If you run the **git fetch** and **git merge** commands in sequence, the files in the working directory remain unchanged after the **git fetch** command is run and the **file1.txt** file is added to the working directory after the **git merge** command is run.

Pull (TortoiseGit)



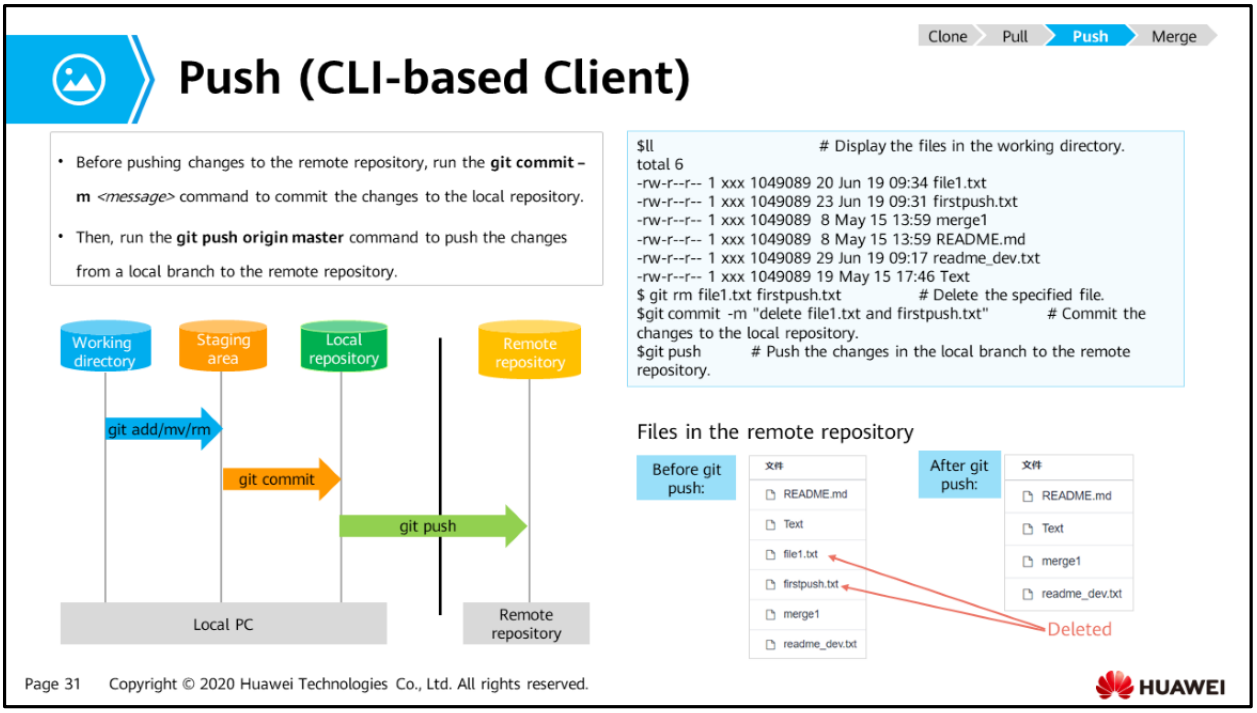
git pull



git fetch + git merge

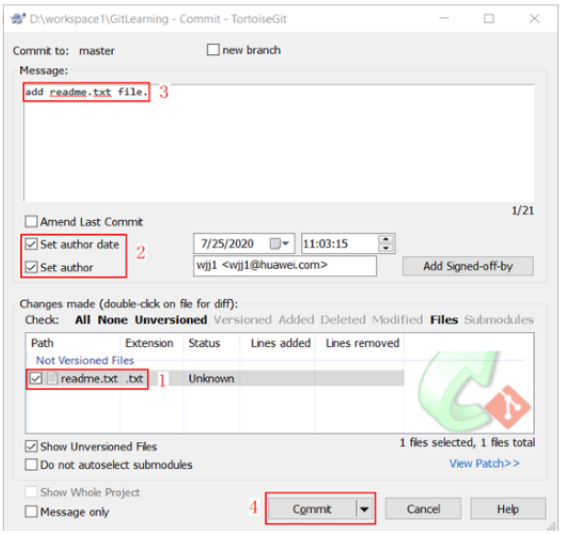
 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

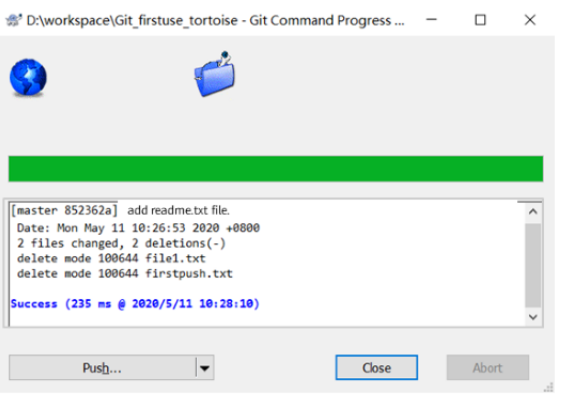
HUAWEI



- The **git push** command pushes the changes made in the local branch that you have checked out to the remote repository. Before pushing changes to the remote repository, you need to commit the changes to the local repository.

Push (TortoiseGit)





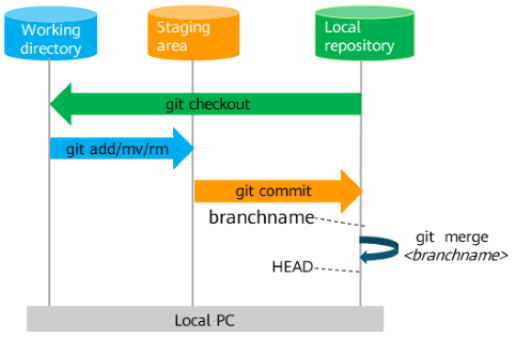
1. Git commit
2. Git push

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

Merge (CLI-based Client)

- The **git merge** command merges a specified branch into the branch you have checked out.
- The merge is likely to run into conflicts. When Git encounters a conflict, it will ask for manual intervention to resolve it and the changes need to be committed again through the **git commit** command.




```
$ git branch hotfix              # Create a hotfix branch.
$ git switch hotfix              # Switch to the hotfix branch.
$ll                              # Display the files in the working directory.
total 2
-rw-r--r-- 1 xxx 1049089 27  7 May 16:12 forgotten_file
-rw-r--r-- 1 xxx 1049089 78  7 May 16:06 method_rename.txt
$ touch file1.txt                # Create a file.
$ git add file1.txt              # Commit the file to the staging area.
$ git commit -m "add file1.txt in branch hotfix"          # Commit the changes to
the local repository.
$ git switch master              # Switch to the master branch.
$ git merge hotfix               # Merge the changes in the hotfix branch to the
master branch.

$ git log --pretty=oneline --graph          # Display the commit history.
* 34cf9d1a2ab5a5ce7343db8d021387aea8b38f51 (HEAD -> master, hotfix) add
file1.txt in branch hotfix
• ee9d29c2f167671ea1d2cae8e169dda613625db9 method_rename.txt modified

$ll                              # Display the files in the remote repository.
total 3
-rw-r--r-- 1 xxx 1049089  7  8 May 10:25 file1.txt
-rw-r--r-- 1 xxx 1049089 27 7 May 16:12 forgotten_file
-rw-r--r-- 1 xxx 1049089 78 7 May 16:06 method_rename.txt
```

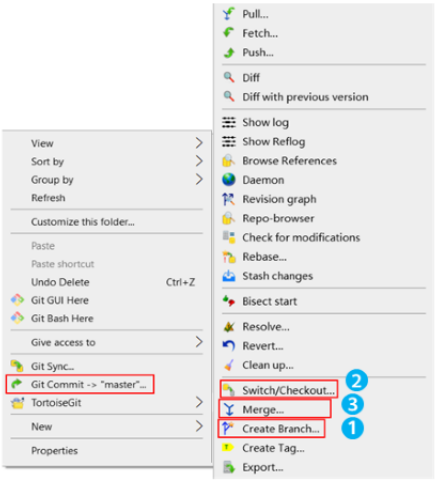
 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

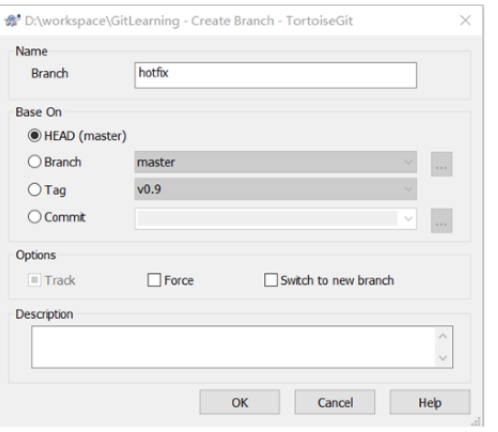
HUAWEI

- The **git merge** command merges a specified branch into the branch you have checked out.
- Create a hotfix branch, modify and commit the changes, switch to the master branch, and merge the changes in the hotfix branch to the master branch. The **file1.txt** file created in the hotfix branch has been merged into the master branch.

Merge (TortoiseGit)

- Create a hotfix branch.

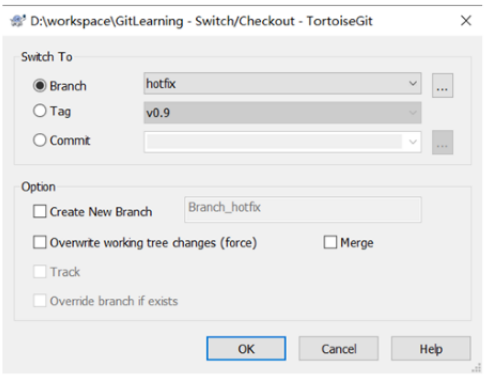




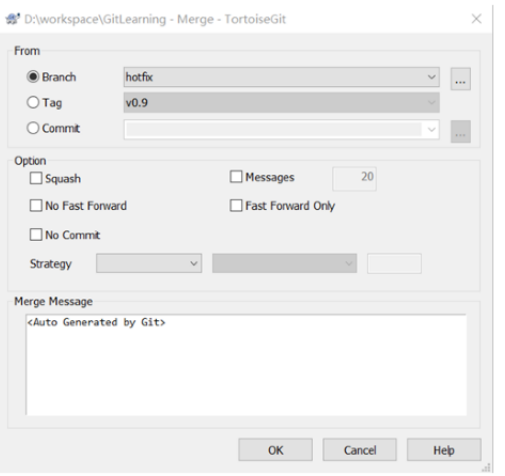
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Merge (TortoiseGit)

- Switch to the hotfix branch and commit the changes in the hotfix branch to the local repository.



- Switch to the master branch and merge the changes in the hotfix branch to the master branch.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

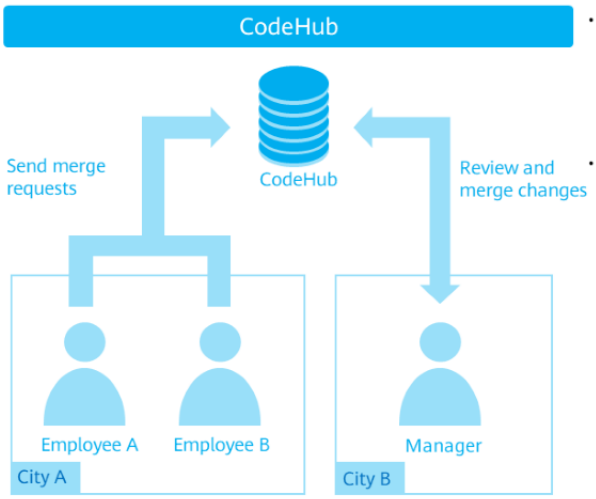
Contents

1. Version Control Overview
2. Git Basic Concepts
3. Basic Git Operations: Clone, Pull, Push, and Merge
4. **HUAWEI CLOUD-based Code Hosting**

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

Code Hosting on HUAWEI CLOUD



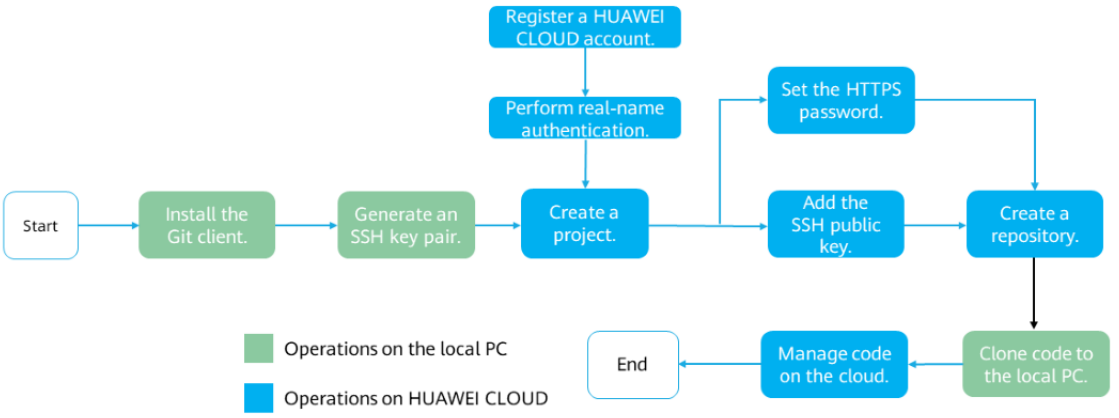
- CodeHub is a Git-based online code hosting service for software developers. It is a cloud code repository that provides such functions as security management, member and permission management, branch protection and merging, online editing, and statistics. The service aims to address issues such as cross-distance collaboration, multi-branch concurrent development, code version management, and security.
- Major functions:
  - Online code operations: You can read, modify, and commit code online anytime, anywhere.
  - Permission management: You can add, delete, and modify project members, and assign permissions to them.
  - Online branch management: You can create, switch, and merge branches.
  - Online code hosting: You can clone, update, push, compare codes.
  - Statistics services: You can view repository operation logs and member contribution statistics
  - IP address whitelists: Unauthorized code downloads are blocked to ensure data security.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Currently, Codehub is available only at the China site. You can select the China site and switch the language to English.

Code Hosting Procedure

- Code hosting procedure on HUAWEI CLOUD



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Generating an SSH Key Pair on the Local PC

- Git can use the SSH protocol to manage versioned code files. An SSH key pair is used to establish a secure connection between a PC and the code hosting service. Different users may use different computers. Before using SSH to connect to a code repository, you need to configure an SSH key pair on your local PC.
- Command: ssh-keygen –t rsa –C "*<key comment>*"
- If you press **Enter** without entering the password, the generated private key file **id_rsa** is stored in plain text on the local PC.
- If **passphrase** is set, the generated private key file is saved in cipher text on the local PC.

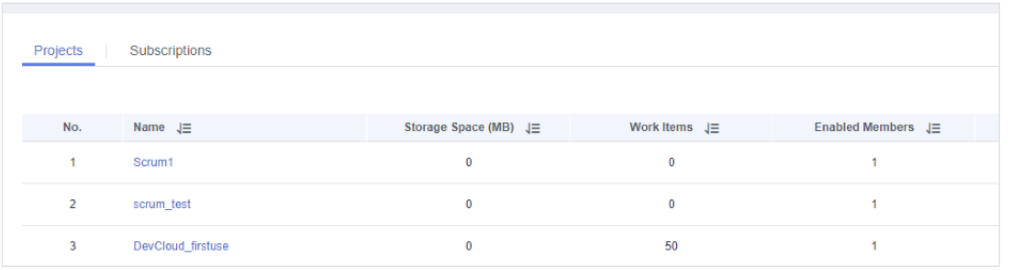
```
$ ssh-keygen  -t rsa -C "wjj1@huawei.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/xxx/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/xxx/.ssh/id_rsa
Your public key has been saved in /c/Users/xxx/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:kZUYzfaGAIOzEWDf7pirF+W6uT3WtqZH3nYUmSuVJuU
wjj1@huawei.com
The key's randomart image is:
+---[RSA 3072]----+
| o..oo..=..      |
| . .+. .oo= .    |
|   .+. oo = +    |
|   ... .o E      |
|    o. S  = o    |
|   .+. . . o     |
|   oo.+ . o      |
|   o+o * o .     |
| .o=+o*.o .      |
+----[SHA256]-----+
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- When using SSH to connect to the remote repository, developers must generate an SSH key pair locally and add the SSH public key to the code hosting service on HUAWEI CLOUD.

Creating a Project on HUAWEI CLOUD

- Since CodeHub depends on ProjectMan, you need to subscribe to ProjectMan before using CodeHub. Service provisioning needs to be performed by region. Therefore, select a proper region before service provisioning.
- After a project is created, you can create code repositories under this project.



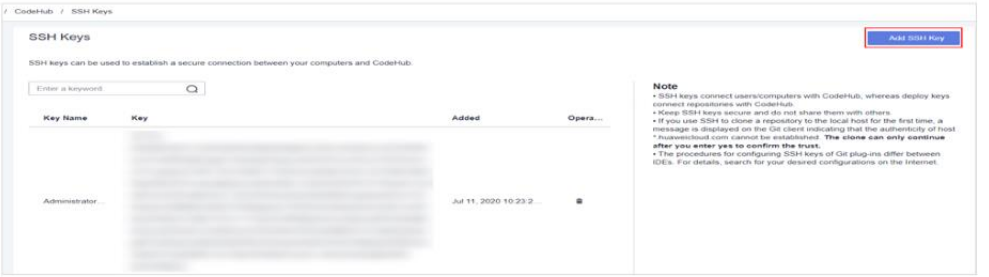
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- CodeHub depends on ProjectMan. Subscribe to ProjectMan before using CodeHub.

Adding the SSH Public Key

- Copy the generated SSH public key, log in to the CodeHub home page, click **Add SSH Key**, and add the SSH public key as prompted.

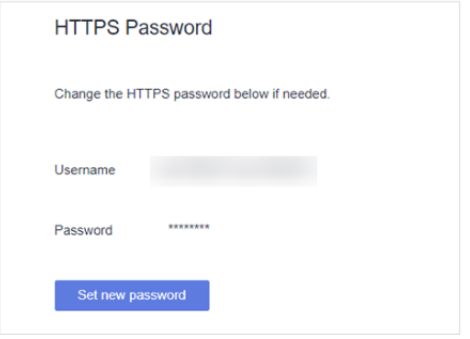
```
$ cat ~/.ssh/id_rsa.pub
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQC4NjZfIQB468a/W/pqYGFc25IkQvg9i7m6ytMJ3YkbzBEfPtUw+VzXW9gnHi3WC5JEw9GLjFvDK+HhNjJj
Cj3GCbe6WwcFqSZA6CnQtcNChpMKqZAiY1UrsGVVxRWrZzvRItHaltax7B/B8fEmsBn1BaMC3n1aRN7YNRkrzQIGAEt+ss+kAYnwOb3+WBEX4/0u
oN9S3aVjVrkcIA4wdToOCL2EHvKHax17yj3319KXCpGzq92jIbp04KcToWqf75bNb6d5fjD2pfWY/K4YcgJ7MCk1hGL6w34ucCxuiCvC54GseWUnfA
pUYCse0n78WzwUXbFKJQUxwgRg/VNHito4cX2vSqH7V0SB8nQMMN0eDLzk57WpGSsf4mUOxCa6EhDaOh03l0s14S0vMQokKGgVtfpHfHdquJ
53OiOc8k7V81vNpEbE7ufmWPIkbD6HtwbZ2NBvSh4rabY0QpJ9q/rdcCp9X12ajmG7POBo1H/5PuEamPRA7WR6eoP2aaqft78=
```



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Setting HTTPS Password on HUAWEI CLOUD

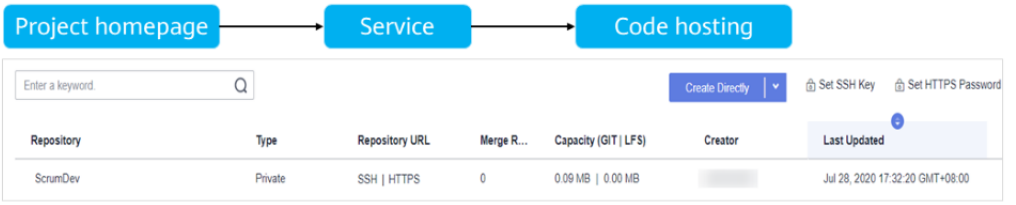
- HTTPS is another protocol used by Git to manage versioned code. An HTTPS password is a credential for local hosts to interact with the code hosting server using the HTTPS protocol.
- Log in to the CodeHub home page, click **Set new password**, and set the HTTPS password as prompted.
- No password is required when you download codes from public code repositories using HTTPS.



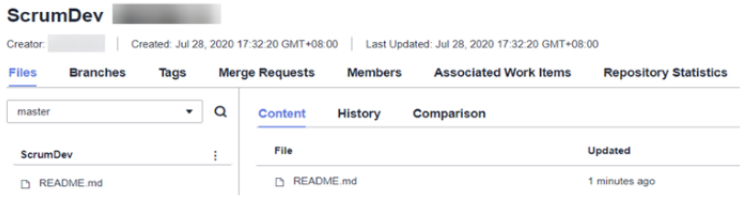
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Creating a Code Repository on HUAWEI CLOUD

- A code repository is an object of code hosting and is the basis for collaborative software development.



- Files in the repository on the cloud:



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Cloning Code to the Local PC

- Click **Clone with ssh/download** on the code repository page, copy the SSH or HTTPS address of the code repository, and run the **git clone** command on the local Git client to clone code from the code repository on the cloud to the local PC.

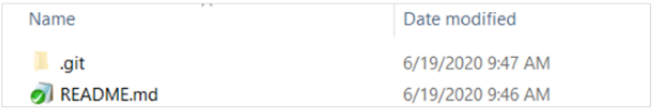
SSH

```
$ git clone git@codehub.devcloud.cn-north-4.huaweicloud.com:DevCloud_firstuse00001/Git_firstuse.git
```

HTTPS

```
$ git clone https://codehub.devcloud.cn-north-4.huaweicloud.com/DevCloud_firstuse00001/Git_firstuse.git
```

- Modify codes on the local PC and commit changes to the code repository on the cloud.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Managing Code on HUAWEI CLOUD

- On HUAWEI CLOUD, you can manage branches, including creating, switching, and merging branches. You can also read, modify, and commit code online. Experience more functions in code repositories on HUAWEI CLOUD.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



Quiz

1. (Single) What type of version control system is Git? ( )

   A. Local version control system

   B. Centralized version control system

   C. Distributed version control system

   D. Open version control system

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

1. C

Summary

- Version control systems allow developers to manage versioned code effectively in software development.
- Version control systems are classified into local version control systems, centralized version control systems, and distributed version control systems. Git is a distributed version control system.
- Git has CLI- and GUI-based clients available. Basic Git operations include clone, pull, push, and merge.
- HUAWEI CLOUD provides a Git-based online code hosting service for software developers to resolve issues such as cross-region collaboration, multi-branch concurrent development, code version management, and code security.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

More Information

- Git: https://git-scm.com/
- TortoiseGit: https://tortoisegit.org/
- CodeHub operation guide: https://support.huaweicloud.com/codehub/index.html

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.


Thank You
www.huawei.com



SSH Principles and Practices

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Foreword

- To achieve efficient O&M and enhance agility amid increasingly complex service requirements and network architecture, network automation is gaining momentum and ever evolving. Currently, Secure Shell (SSH) is the most common method used by engineers to log in to devices for remote management. As such, engineers are expected to learn about and use an automation tool to implement SSH remote login, simulate man-machine interaction with O&M personnel, and automatically transfer files.
- In this course, we will use the Python Paramiko module to write automation scripts to implement SSH-based preliminary network automation.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Objectives

- Upon completion of this course, you will be able to:
  - Describe the basic concepts and working principles of SSH.
  - Understand the concept of Paramiko.
  - Master the composition and common methods of Paramiko.
  - Grasp the common methods for implementing Paramiko.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

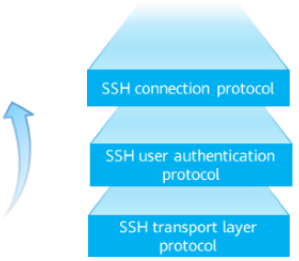
Contents

1. **Introduction to SSH**
   - **Overview of SSH**
     - Working Principles of SSH
     - Overview of Paramiko
2. Paramiko Component Architecture
3. SSH Practices

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Overview of SSH

- Secure Shell (SSH) is a protocol for secure remote login and other secure network services over an insecure network.
- SSH consists of the following sub-protocols: SSH transport layer protocol, SSH user authentication protocol, and SSH connection protocol.



Establishes a session connection.

Authenticates users (password and key).

Negotiates the version and algorithm and exchanges keys.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Transport Layer Protocol > User Authentication Protocol > Connection Protocol

SSH Transport Layer Protocol

- SSH transport layer protocol is a secure transport protocol. The SSH transport layer is usually established over TCP/IP connections. It can also be established over any other reliable data flow.
- The SSH transport layer protocol negotiates all key exchange algorithms, public key algorithms, symmetric encryption algorithms, and message authentication algorithms.

Algorithm Type	Function	Name
Key exchange algorithm	Generates session keys.	diffie-hellman-group14-sha1, diffie-hellman-group1-sha1, etc.
Public key algorithm	Performs digital signature and user authentication.	ssh-rsa, ssh-dss, etc.
Symmetric encryption algorithm	Encrypts sessions.	aes128-ctr, 3des-cbc, etc.
Message authentication algorithm	Verifies data integrity.	hmac-sha1, hmac-md5, etc.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Perfect Forward Secrecy (PFS) is a property of secure communication protocols. PFS was proposed by Christoph G. Gunther in 1990. PFS is essentially defined as the cryptographic property of a key-establishment protocol in which the compromise of a session key or long-term private key after a given session does not cause the compromise of any earlier session. PFS ensures that any future disclosure of passwords or keys cannot be used to decrypt any communications sessions recorded in the past, even if the attacker proactively intervenes.
- The SSH transport layer protocol uses the Diffie-Hellman key exchange algorithm to implement PFS.
- For details, see section 9.3.7 "Forward Secrecy" in RFC 4251 (https://www.ietf.org/rfc/rfc4251.txt).

SSH User Authentication Protocol

- The SSH user authentication protocol authenticates the client-side user to the server. It runs over the transport layer protocol.
- The SSH user authentication protocol provides two authentication methods: password authentication and public key authentication.
  - Password authentication: The client uses the user name and password for authentication before successfully logging in to the server.
  - Public key authentication: The server decrypts the digital signature of the client by using a public key.

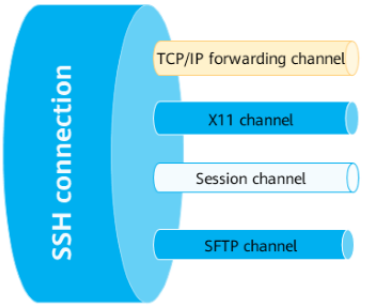


 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- A public key is used to decrypt information to ensure message authenticity and integrity. Therefore, the receiver knows that the information comes from someone who has a private key. The encrypted information is called a digital signature. The public key is in the form of a digital certificate.
- When the SSH user authentication protocol is started, it receives a session ID from the SSH transport layer protocol. The session ID uniquely identifies a session and is a part of the digital signature to indicate the ownership of the private key.

SSH Connection Protocol

- The SSH connection protocol multiplexes several logical channels into a single encrypted tunnel. It provides interactive login sessions, remote execution of commands, forwarded TCP/IP connections, and forwarded X11 connections.
- The SSH connection protocol runs on top of the SSH transport layer protocol and user authentication protocol.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- A TCP/IP connection can forward network data of other TCP ports through SSH channels, ensuring security.
- Data of Telnet, SMTP, IMAP, and other TCP/IP-based insecure protocols can be forwarded through SSH, which prevents the transmission of user names, passwords, and privacy information in plaintext and therefore enhances security. In addition, if the firewall restricts the use of some network ports but allows the SSH connection, communication can be implemented through the SSH TCP/IP connection.
- In X11, X refers to the X protocol, and 11 is the eleventh version of the X protocol. The Linux graphical user interface (GUI) is based on the X protocol at the bottom layer. When remote interaction with graphical applications on the Linux server is required, a method for enhancing communication security is to use SSH to display the GUI on the local client through the X11 tunnel.
- A session is a remote execution of a program. A program can be a shell, an application, a system command, or some built-in subsystems. Multiple session channels can be active at the same time. An interactive login session can be implemented using the invoke_shell() method, and the remote command can be implemented using the exec_command() method, which will be described in detail later.

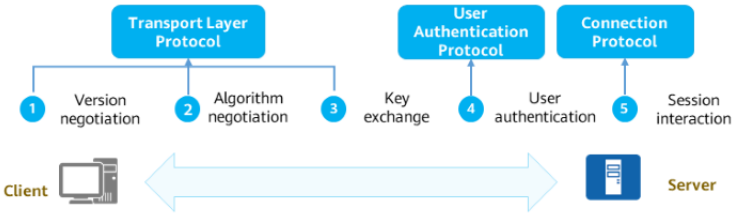
Contents

1. **Introduction to SSH**
   - Overview of SSH
   - **Working Principles of SSH**
   - Overview of Paramiko
2. Paramiko Component Architecture
3. SSH Practices

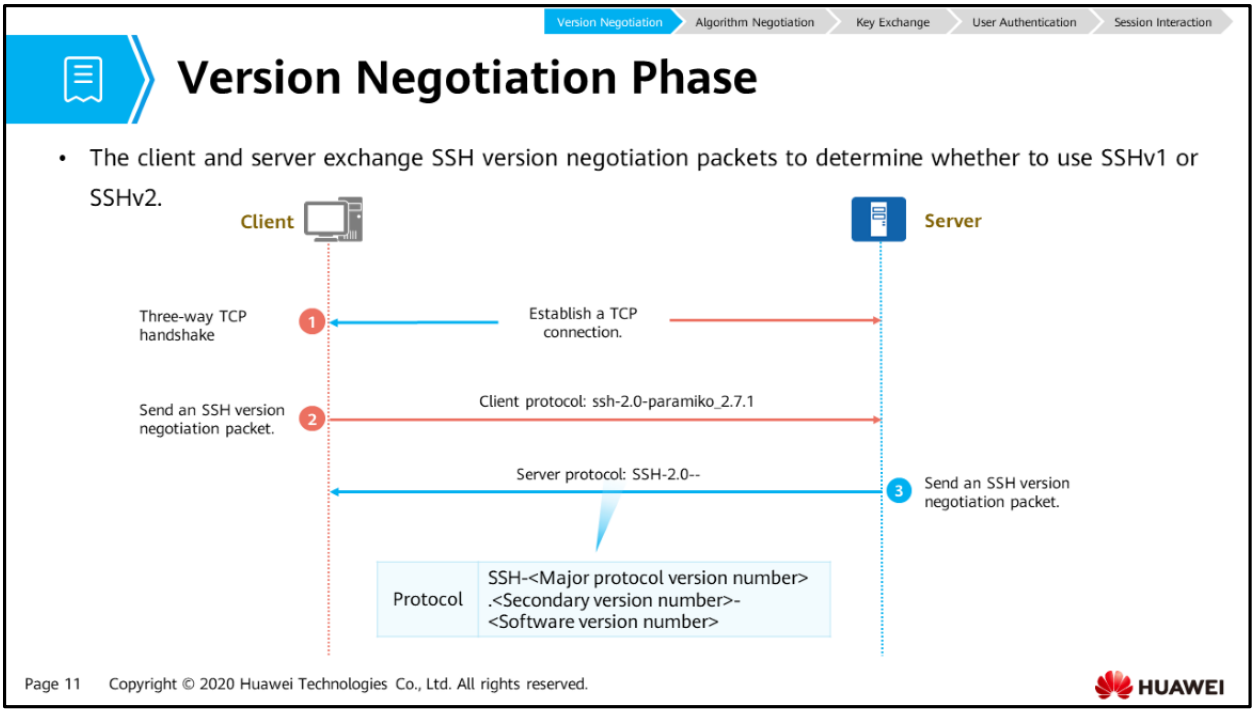
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Working Principles of SSH

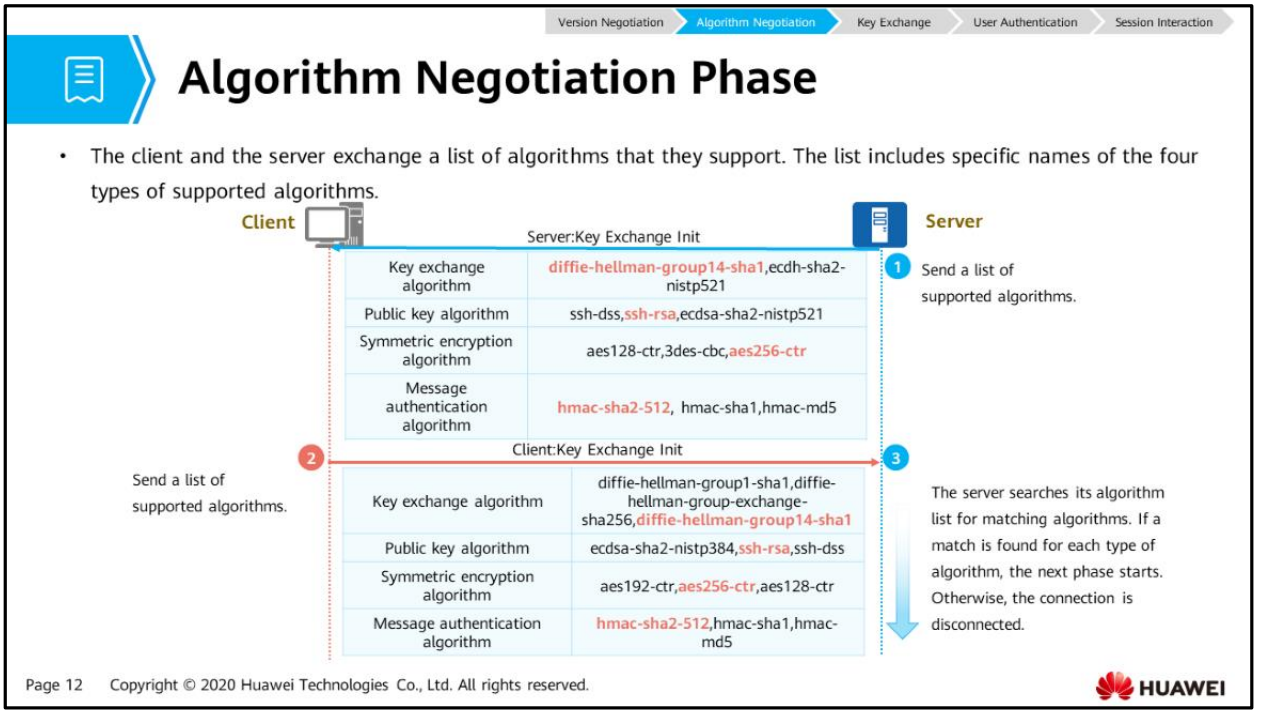
- In the entire communication process, to implement a secure SSH connection, the server and client go through the following five phases:
  - Version negotiation phase: Two versions of SSH are available: SSH Version 1 (SSHv1) and SSH Version 2 (SSHv2). The server and client determine the version to be used through negotiation.
  - Algorithm negotiation phase: SSH supports multiple encryption algorithms. The server and client negotiate the encryption algorithm to be used based on the algorithms that they support.
  - Key exchange phase: A session key is generated by using a key exchange algorithm. The subsequent sessions between the server and client are encrypted by using the session key.
  - User authentication phase: The SSH client sends an authentication request to the server, and the server authenticates the SSH client.
  - Session interaction phase: After the authentication succeeds, the server and client exchange information.



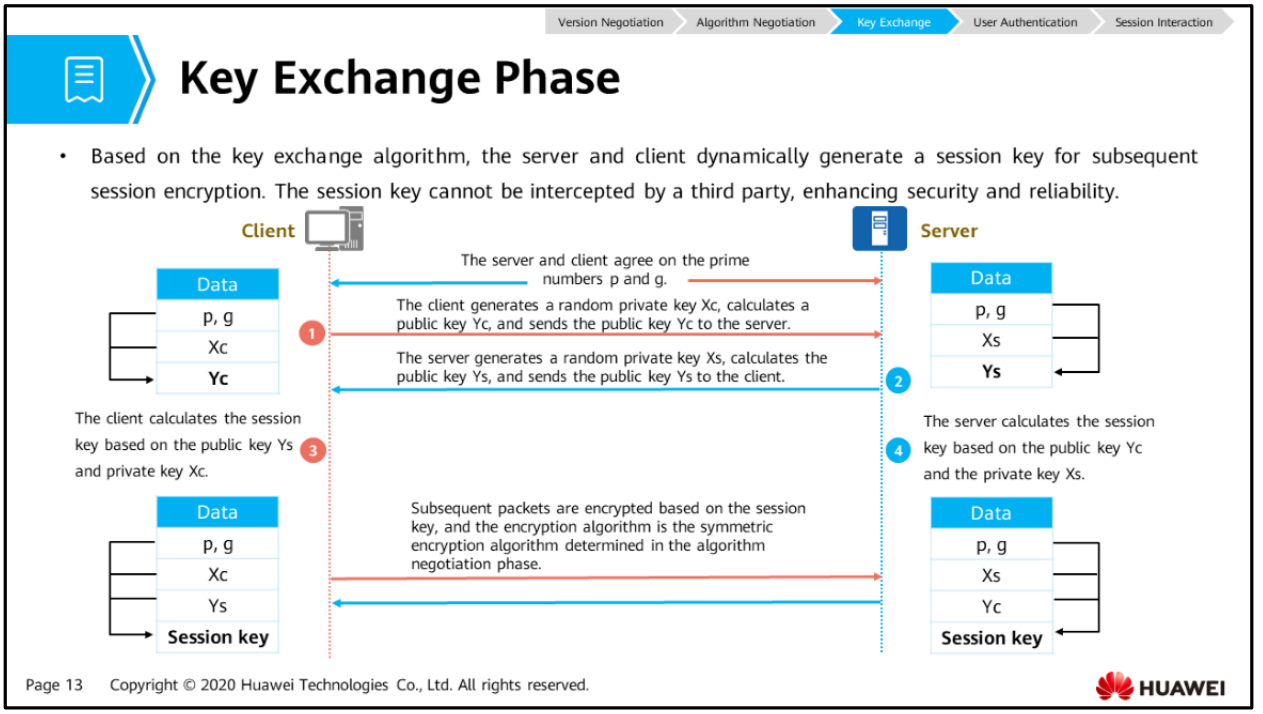
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI



- Port 22 is enabled on the server, waiting for the client to connect. The client initiates a TCP connection to the server. The two parties complete the handshake and establish a connection. The client sends a packet to the server. The packet contains the version field, in the format of Major version number.Secondary version number-Software version number. After receiving the packet, the server parses it to obtain the protocol version number. If the protocol version number of the client is earlier than that of the server and the server supports the earlier version of the client, the server uses the protocol version number of the client. Otherwise, the server uses its own protocol version number.



- The algorithm negotiation process is as follows: The server obtains the first algorithm from the algorithm list of the client and searches its own algorithm list for the same algorithm. If the same algorithm is found, the negotiation succeeds, and the server continues to negotiate the algorithm of the next type. Otherwise, the server searches its own algorithm list for the next algorithm in the client's algorithm list until a match is found.



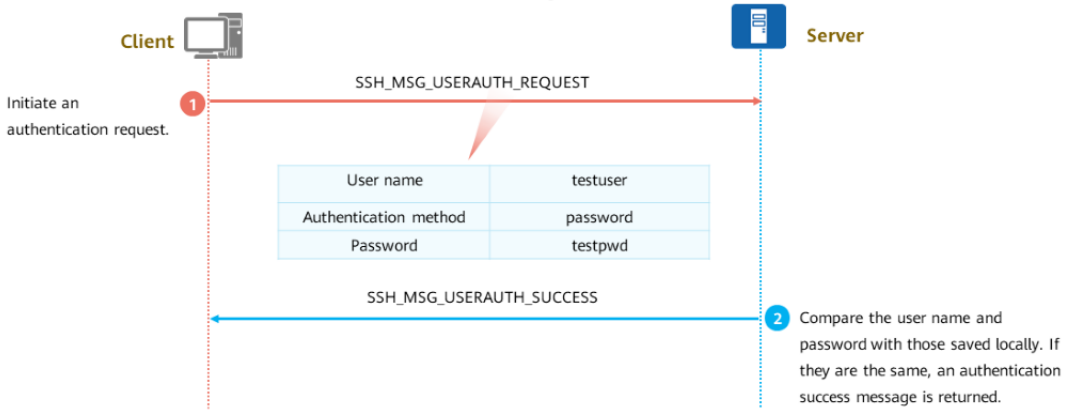
1. The client and the server first agree on two public prime numbers p and g.
2. The client and server each randomly generate a private key Xc and Xs, respectively.
3. The client and server each calculate their own public key Yc and Ys, respectively.
4. The client and server exchange their own public key.
5. The client and server calculate the session key for encryption based on the public and private keys.

The Diffie-Hellman key exchange algorithm is used for key exchange, which is based on the mathematical discrete logarithm and is not described in this course. During key exchange, the private keys Xc and Xs are not transferred and, due to the difficulty in computing discrete logarithms, they cannot be decrypted by other users even if p, g, Yc, and Ys are obtained. This ensures the confidentiality of the session keys.

Note that the public and private keys generated in this phase are used only to generate session keys and are irrelevant to subsequent user authentication. After the key exchange phase is complete, all subsequent packets are encrypted based on the session keys.

User Authentication Phase: Password Authentication

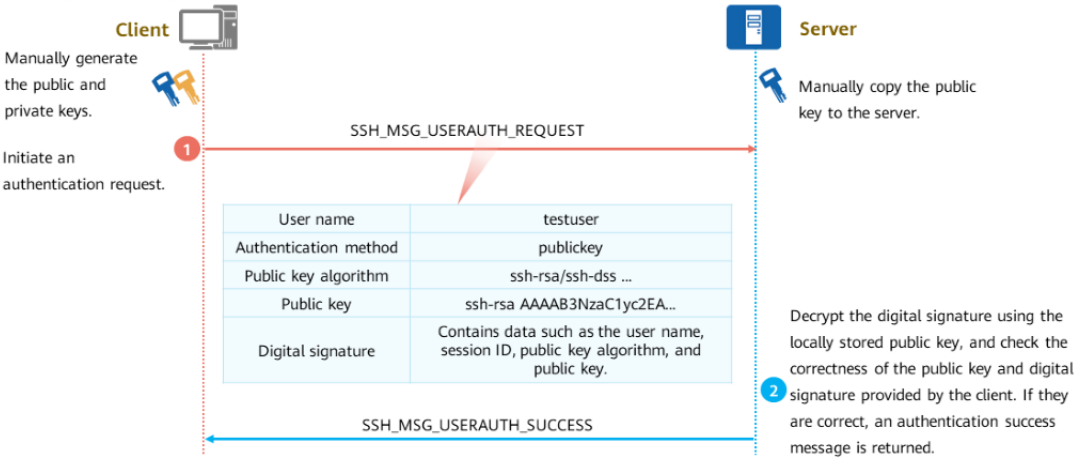
- There are two user authentication modes: password authentication and public key authentication.
- During password authentication, the client sends an authentication request carrying the user name and password, and the server authenticates the received user information against the local user information.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

User Authentication Phase: Public Key Authentication

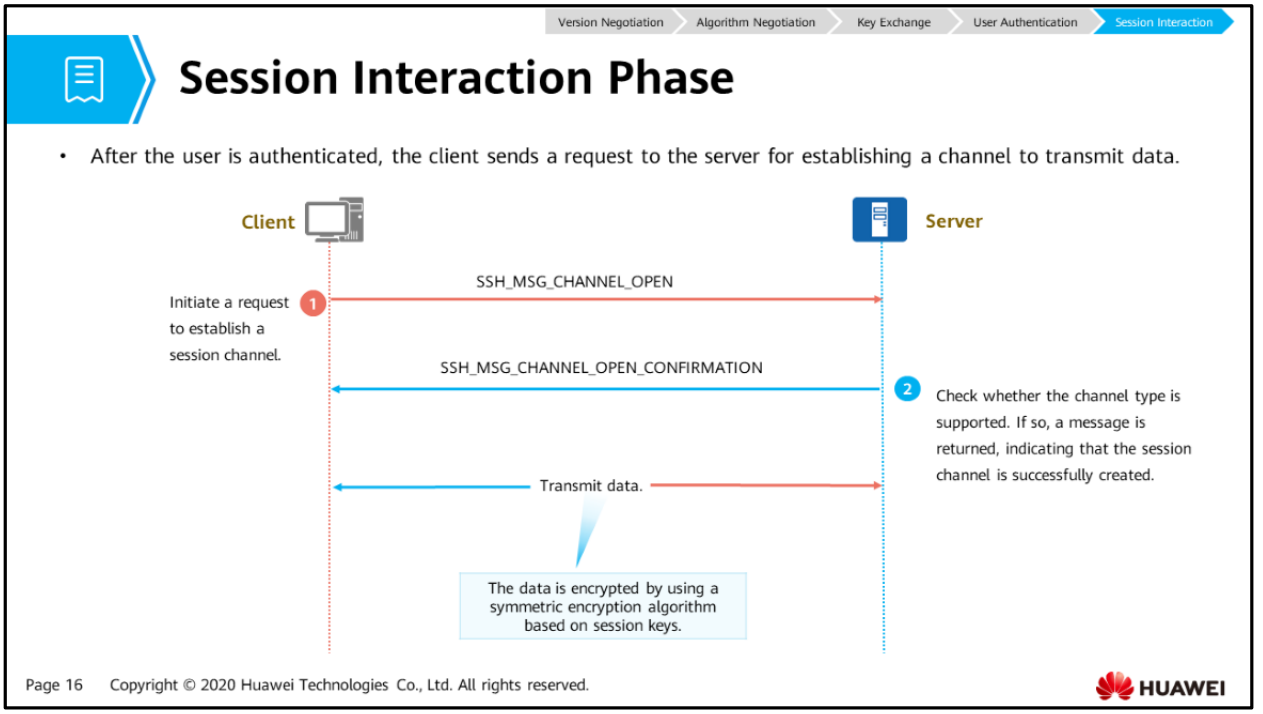
- During public key authentication, a client sends an authentication request carrying a digital signature, and the server decrypts the digital signature based on the public key to implement authentication.



 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- The digital signature is encrypted by client's private key. To see the content, we need public key to decrypt it.



- The channel types include session, x11, forwarded-tcpip, and direct-tcpip.
- For details, see section 4.9.1 "Connection Protocol Channel Types" in RFC4250 at https://www.ietf.org/rfc/rfc4250.txt.
- Different ssh logical channels can multiplex one ssh session.

Contents

1. **Introduction to SSH**
   - Overview of SSH
   - Working Principles of SSH
   - **Overview of Paramiko**
2. Paramiko Component Architecture
3. SSH Practices

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Overview of Paramiko

- Paramiko is a Python module that implements the SSHv2 protocol. It supports password authentication and public key authentication and implements functions such as secure remote command execution and file transfer.
- Engineers can compile Python code based on the Paramiko module to implement SSH functions.



 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- In HCIA courses, we learned how to use the telnetlib module for Telnet remote connections. In the production environment, the more secure Paramiko module is recommended for SSH remote connections.

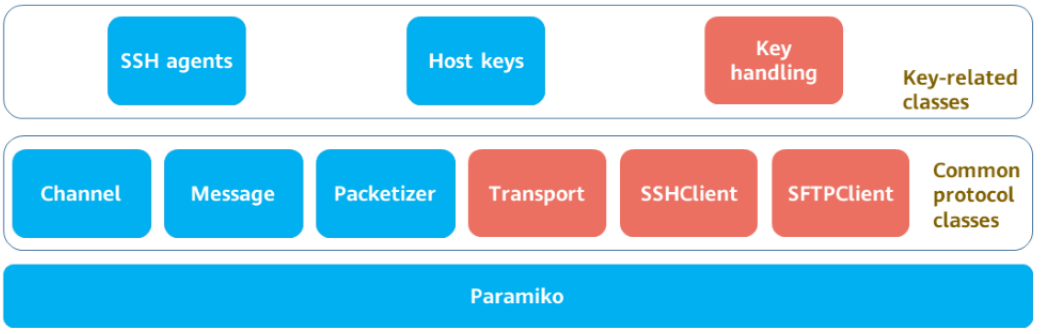
Contents

1. Introduction to SSH
2. **Paramiko Component Architecture**
   - **Paramiko Component Architecture**
   - Transport Class and Its Methods
   - Key Handling Class and Its Methods
   - SFTPClient Class and Its Methods
   - SSHClient Class and Its Methods
3. SSH Practices

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Paramiko Component Architecture

- The following figure shows the components of the Paramiko module. SSHClient and SFTPClient are its most commonly used classes, which provide the SSH and SFTP functions, respectively.
- This course describes the methods of the Transport, key handling, SSHClient, and SFTPClient classes.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Common Paramiko Classes

- Channel: This class is used to create a secure channel over the SSH transport layer.
- Message: An SSH message is a stream of bytes that encodes some combinations of strings, integers, bools, and infinite-precision integers (known in Python as longs).
- Packetizer: This class is used for packet handling.
- Transport: This class is used to create a transport session object over an existing socket or socket-like object.
- SFTPClient: This class creates an SFTP session connection through an open SSH transport session and performs remote file operations.
- SSHClient: This class is an advanced representation of a session with the SSH server. This class integrates the Transport, Channel, and SFTPClient classes.

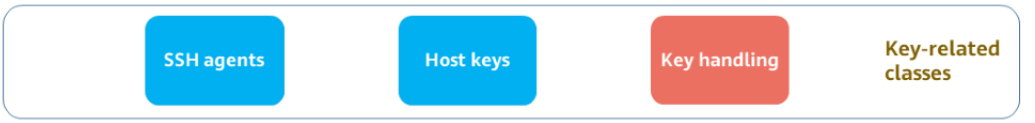


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- The Channel class provides methods for executing commands, requesting X11 sessions, sending data, and opening interactive sessions. Generally, these common methods from the Channel class have been packaged in the SSHClient class.
- The Message class provides methods for writing bytes to a stream and extracting bytes.
- The Packetizer class provides methods for checking handshakes and obtaining channel IDs.
- The Transport class provides methods such as public key authentication, private key authentication, and channel opening.
- The SSHClient class provides methods for establishing connections and opening interactive sessions.
- The SFTPClient class provides methods such as file upload and download.

Key-Related Classes of the Paramiko Module

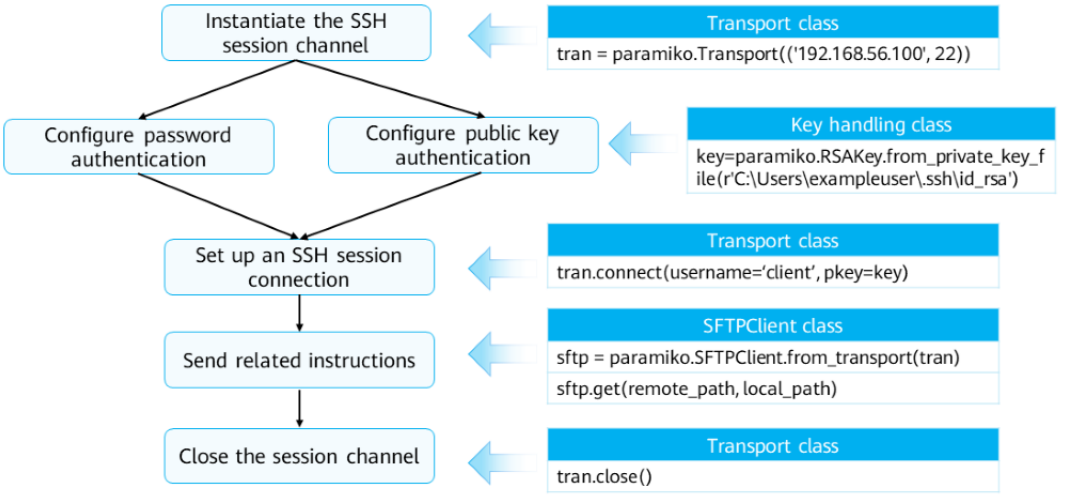
- SSH Agent: This class is used for the SSH agent.
- Host keys: This class is related to the OpenSSH known_hosts file and is used to create a host keys object.
- Key handling: This class is used to create instances of the corresponding key type, for example, RSA keys and DSS (DSA) keys.



 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- OpenSSH is a free open-source implementation of the SSH protocol. It provides server programs and client tools. OpenSSH is integrated in all Linux operating systems. OpenSSH records the public key of each computer that a user has accessed in **~/.ssh/known_hosts**. When the same computer is accessed next time, OpenSSH checks the public key. If the public keys are different, OpenSSH generates a warning to prevent man-in-the-middle attacks.

Process of Using Paramiko



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- This course describes methods of four classes: Transport, key handling, SSHClient, and SFTPClient.
- This process uses the Paramiko SFTP session as an example. Because the SSHClient class integrates the Transport, Channel, and SFTPClient classes, the preceding methods can be implemented by the SSHClient class. This is especially true for SSH sessions.

Contents

1. Introduction to SSH
2. **Paramiko Component Architecture**
   - Paramiko Component Architecture
   - **Transport Class and Its Methods**
   - Key Handling Class and Its Methods
   - SFTPClient Class and Its Methods
   - SSHClient Class and Its Methods
3. SSH Practices

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Transport Class and Its Methods

- Transport class: An SSH transport connects to a stream (usually a socket) to negotiate and encrypt sessions and perform authentication. Channels can then be created based on the encrypted sessions. Multiple channels can be multiplexed in a single session connection (in fact, this is often the case, such as port forwarding).
- The following is an example of the method:

```
tran = paramiko.Transport(('192.168.56.100', 22))
tran.connect(username='client',  password='test')
```

Common Method	Function
Transport(sock)	Creates a Transport object and instantiates the SSH session channel.
connect(username='',password=None,pkey=None)	Establishes an SSH session connection and uses a password or private key for identity authentication.
close()	Closes the session.

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- For ease of use, you can use an address (as a tuple) or a host string as the **sock** parameter. The host string is the host name with an optional port, separated by a colon (:). If a port is transferred, it is converted to a tuple in the format (*host name, port*).

Contents

1. Introduction to SSH
2. **Paramiko Component Architecture**
   - Paramiko Component Architecture
   - Transport Class and Its Methods
   - **Key Handling Class and Its Methods**
   - SFTPClient Class and Its Methods
   - SSHClient Class and Its Methods
3. SSH Practices

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Key Handling Class and Its Methods

- The key handling class is used to create instances of the corresponding key type, for example, RSA keys and DSS (DSA) keys. This class provides methods for reading and writing keys.
- The following is an example of the method:

```
key=paramiko.RSAKey.from_private_key_file(r'C:\Users\exampleuser\.ssh\id_rsa')
```

Common Method	Function
RSAKey.from_private_key_file(filename)	Reads the RSA private key from a file to create a key object.
DSSKey.from_private_key_file(filename)	Reads the DSS private key from a file to create a key object.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. Introduction to SSH
2. **Paramiko Component Architecture**
   - Paramiko Component Architecture
   - Transport Class and Its Methods
   - Key Handling Class and Its Methods
   - **SFTPClient Class and Its Methods**
   - SSHClient Class and Its Methods
3. SSH Practices

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

SFTPClient Class and Its Methods

- The SFTPClient class creates an SFTP session connection through an open SSH transport session and performs remote file operations.
- The following is a typical example:

```
key=paramiko.RSAKey.from_private_key_file(r'C:\Users\exampleuser\.ssh\id_rsa')
tran = paramiko.Transport(('192.168.56.100', 22))
tran.connect(username='client', pkey=key)
sftp = paramiko.SFTPClient.from_transport(tran)
local_path=r'C:\Users\exampleuser\.ssh\vrptest.cfg'
remote_path= '/vrpcfg.cfg'
sftp.get(remote_path, local_path)
```

Common Method	Function
from_transport()	Creates an SFTP session connection through an open Transport session channel.
get()	Downloads a specified file.
put()	Uploads a specified file.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Method from_transport

- from_transport(): This method creates an SFTP client channel from the enabled Transport session channel.
- The following is an example of the method:

```
t = paramiko.Transport(('192.168.56.100', 22))
sftp = paramiko.SFTPClient.from_transport(t)
```

Parameter	Description
T	An authenticated and enabled Transport session, in the format of (hostname,port).
windows_size	Size of the SFTP session window. This parameter is optional.
max_packet_size	Maximum size of the SFTP session window. This parameter is optional.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Method get

- get(): This method copies a remote file (specified by **remotepath**) from the SFTP server to the destination path (specified by **localpath**) on the local host. Any exception raised by operations will be passed through.
- The following is an example of the method:

```
local_path=r'C:\Users\exampleuser\.ssh\vrptest.cfg'
remote_path= '/vrpcfg.cfg'
sftp.get(remote_path, local_path)
```

Parameter	Description
remotepath	Remote file.
localpath	Destination path on the local host. The path must contain the file name. If only a directory is specified, an error may occur.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Method put

- put(): This method copies a local file (specified by **localpath**) from the local host to the destination path (specified by **remotepath**) on the SFTP server. Any exception raised by operations will be passed through.
- The following is an example of the method:

```
local_path=r'C:\Users\exampleuser\.ssh\vrptest.cfg'
remote_path= '/vrpcfg.cfg'
sftp.put(localpath, remotepath)
```

Parameter	Description
localpath	Local file.
remotepath	Destination path on the SFTP server. The path must contain the file name. If only a directory is specified, an error may occur.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. Introduction to SSH
2. **Paramiko Component Architecture**
   - Paramiko Component Architecture
   - Transport Class and Its Methods
   - Key Handling Class and Its Methods
   - SFTPClient Class and Its Methods
   - **SSHClient Class and Its Methods**
3. SSH Practices

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

SSHClient Class and Its Methods

- The SSHClient class is an advanced representation of a session with an SSH server. This class contains the Transport, Channel, and SFTPClient classes for session channel establishment and authentication. The following is a typical example:

```
client=paramiko.client.SSHClient()
client.connect(hostname=192.168.56.100',port=22,username='client',password='123456')
stdin,stdout,stderr=client.exec_command('ls  -l')
```

Common Method	Function
connect()	Connects to the remote server and implements authentication.
set_missing_host_key_policy()	Specifies a policy to be used when the connected server does not have a known host key.
load_system_host_key()	Loads the host key from the system file.
exec_command()	Runs Linux commands on the remote server.
invoke_shell()	Starts an interactive shell session on the remote server.
open_sftp()	Creates an SFTP channel in a session connection.
close()	Closes a connection.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Method connect

- connect(): This method is used to connect to a remote server and implement authentication.
- The following is an example of the method:

```
client.connect(hostname='192.168.56.100',port=22,username='client',key_filename='id_rsa')
client.connect(hostname='192.168.56.100',port=22,username='client',password='123456')
```

Parameter	Description
hostname	Target host to be connected. Only this parameter is mandatory.
port	Specified port. The default value is 22.
username	User name for authentication. This parameter is left empty by default.
password	Password of the user to be authenticated. This parameter is left empty by default.
key_filename	Private key file name or list. This parameter is left empty by default.
pkey	Private key used for identity authentication.
...	...

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Method set_missing_host_key_policy

- set_missing_host_key_policy(): This method specifies a policy to be used when the connected server does not have a known host key.
- The following is an example of the method:

```
client.set_missing_host_key_policy(paramiko.client.AutoAddPolicy())
```

Parameter	Description
AutoAddPolicy	Automatically adds the host name and host key to the local HostKeys object, without depending on the configurations of the load_system_host_keys method. That is, when a new SSH connection is set up, you do not need to enter **yes** or **no** for confirmation.
WarningPolicy	Logs a Python-style warning for an unknown host key and accepts it. This method provides functions similar to AutoAddPolicy. The difference lies in that this method will display a message, indicating that the connection is a new connection.
RejectPolicy	Automatically rejects the unknown host name and key. This method depends on the configuration of the load_system_host_keys method. This is the default option.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- OpenSSH records the public key of each computer that a user has accessed in **~/.ssh/known_hosts**. When the same computer is accessed next time, OpenSSH checks the public key. If the public keys are different, OpenSSH generates a warning to prevent man-in-the-middle attacks. Generally, when a client connects to the SSH server for the first time, you need to enter **Yes** or **No** for confirmation.

Method load_system_host_keys

- load_system_host_keys(): This method loads the host key from the system file. If no parameter is specified, the system attempts to read the key from the **known hosts** file on the local host.
- The following is an example of the method:

```
client.load_system_host_keys()
```

Parameter	Description
filename	File name. This parameter is left empty by default.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Method exec_command

- exec_command(): This method is used to run Linux commands on a remote server.
- The following is an example of the method:

```
stdin,stdout,stderr=client.exec_command('ls  –l')
```

Parameter	Description
command	Linux command to be executed.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Method invoke_shell

- invoke_shell(): This method starts an interactive shell session based on the SSH session connection.
- The following is an example of the method:

```
cli = client.invoke_shell()
```

Parameter	Description
None	None

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Method open_sftp

- open_sftp(): This method creates and opens an SFTP session on the SSH server.
- The following is an example of the method:

```
sftp=client.open_sftp()
```

Parameter	Description
None	None

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

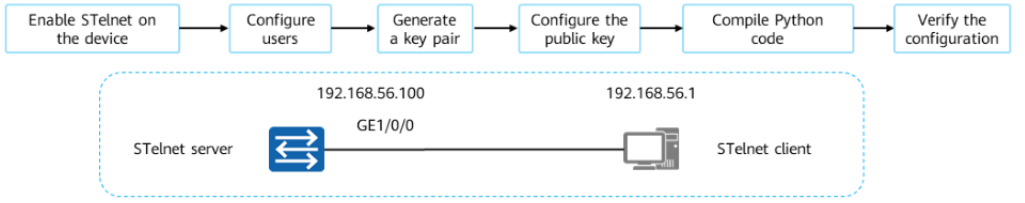
1. Introduction to SSH
2. Paramiko Component Architecture
3. **SSH Practices**
   - **Practices in SSH Python Scripts**
   - Practices in SFTP Python Scripts

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

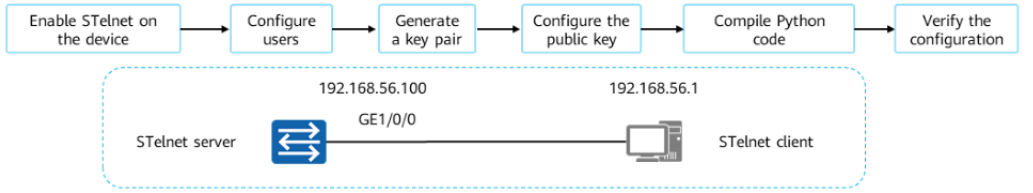
Case: Using SSH to Log In to a Device

- Description:
- As shown in the figure below, after the STelnet server function is enabled on the switch that functions as the SSH server, the PC functioning as the SSH client can log in to the SSH server in password or RSA authentication mode.
- This case uses RSA user authentication as an example to describe how to configure a client so that it logs in to a server through SSH using the Paramiko module of Python.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

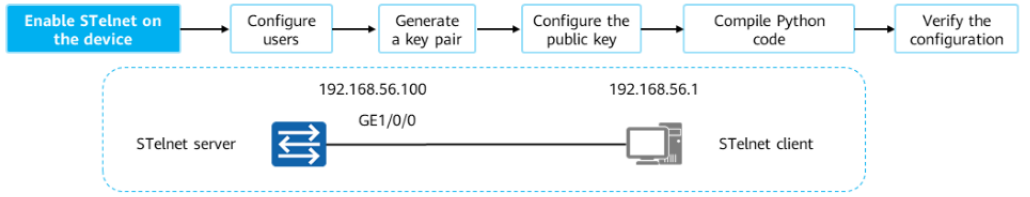
Configuration Roadmap



- Configuration on the server:
  - Configure STelnet. Specially, configure a management IP address, enable the STelnet function, and configure the user interface.
  - Configure users. Specially, create a local user and an SSH user, and configure the service type and authentication mode for the users.
  - Configure a public key. Specially, add the public key generated by the client and allocate it to the user.
- Configuration on the client:
  - Generate a key pair. Specifically, generate a public key and a private key locally.
  - Compile Python code.
  - Verify the configuration.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Case: Configuring STelnet on the Server



1. Configure an IP address for the management network port on the server.

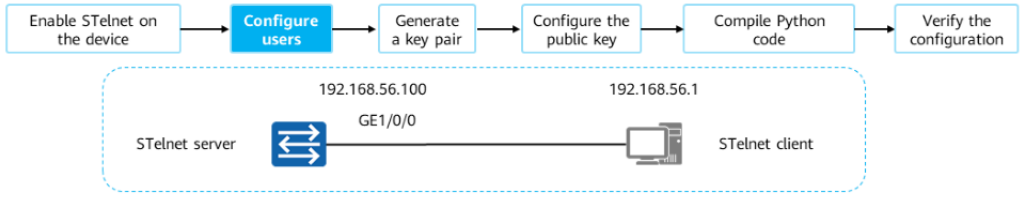
```
<HUAWEI>system-view immediately
[HUAWEI] sysname SSH Server
[SSH Server] interface GE 1/0/0
[SSH Server-GE1/0/0] ip add 192.168.56.100 24
[SSH Server-GE1/0/0] quit
```

2. Enable STelnet on the server and configure the VTY user interface.

```
[SSH Server] stelnet server enable
[SSH Server] user-interface vty 0 4
[SSH Server-ui-vty0-4] authentication-mode aaa
[SSH Server-ui-vty0-4] protocol inbound ssh
[SSH Server-ui-vty0-4] user privilege level 3
[SSH Server-ui-vty0-4] quit
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Case: Configuring Users on the Server



3. Create a local user on the server, add the user to the administrator group, and configure the service type for the user.

```
[SSH Server] aaa
[SSH Server-aaa] local-user client password irreversible-cipher Huawei@123
[SSH Server-aaa] local-user client user-group manage-ug
[SSH Server-aaa] local-user client service-type ssh
[SSH Server-aaa] quit
```

4. Create an SSH user on the server and configure the authentication mode and service type for the user.

```
[SSH Server] ssh user client
[SSH Server] ssh user client authentication-type rsa
[SSH Server] ssh user client service-type stelnet
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Case: Creating an RSA Key Pair on the Client

Enable STelnet on the device → Configure users → **Generate a key pair** → Configure the public key → Compile Python code → Verify the configuration

5. On the client, use Git Bash to create an RSA key pair (private key id_rsa and public key id_rsa.pub) and check the public key.

```
exampleuser@exampleuser  MINGW64 ~
$ ssh-keygen  -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/exampleuser/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/exampleuser/.ssh/id_rsa
Your public key has been saved in /c/Users/exampleuser/.ssh/id_rsa.pub
```

Generate an RSA public/private key pair. --

Set the path for storing the key. (Press **Enter** to use the default path.) --

Enter the pass phrase. (Press **Enter** to use the default pass phrase.) --

Enter the pass phrase again. (Press **Enter** to use the default pass phrase.) --

Path for storing the private key file of the client --

Path for storing the public key file of the client --

Check the public key. --

```
$ cat /c/Users/exampleuser/.ssh/id_rsa.pub
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Case: Configuring a Public Key on the Server

Enable STelnet on the device → Configure users → Generate a key pair → **Configure the public key** → Compile Python code → Verify the configuration

6. On the server, add the public key generated by the client and allocate it to the user.

```
[SSH Server] rsa peer-public-key  rsa01 encoding-type openssh
[SSH Server-rsa-public-key]  public-key-code  begin
[SSH Server-rsa-public-key-rsa-key-code]  ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQDwLRx8MmuNs500dRemhFHdDbBmxco8Bp+wyqwaGuHJZBCjyFQV6AB+ezu5t0eWE3mw57IZfg
mvR+MjBcliZv/x3l8oUMLcQKlKslYQDtvfUCZd+za1suXAPB/dyPKMhYPAzSDA7K+xqCWlmU3q06vxHEPLMv4A5IX54rKtBnK92fWjl9ACU+ak0
ZlHxbKwOFn1tr0GJBazcInEs9DKGwkTTqJdu9+5hI5NxXTSbM3an53805ZbCU18xPy57g7MZC89vbdsag/uvQmFkLJ3arts/Om2R7fhR92EU/SN
PmVy+qDEdwZEVdubdqJInW+8zzVkPGlnb2oH5hwH78Ksklbxb0fEfmGR0mS1ZAi3ZHUGcEEjuFZona3+5Z0Un2OPxfXwvoljVDusbYcugJHo9
Ssurz05GzVuamQZlcO2JYY6FhtLUAImtXGQ80MpTjB0lcprkAZCib8agYOtVQNTZ7iB0g2EcBN9UTyMz7sh8RtrBDj445r+XPaDE8LmpDRKHMk
=
[SSH Server-rsa-public-key-rsa-key-code]  public-key-code  end
[SSH Server-key-code]  peer-public-key  end
[SSH Server] ssh user client assign rsa-key rsa01
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- For details about the commands, refer to the product documentation at https://support.huawei.com/enterprise/en/doc/EDOC1000097293/466984de?idPath=24030814|21432787|21430822|22318704|9794900.

Case: Compiling Python Code on the Client

Enable STelnet on the device → Configure users → Generate a key pair → Configure the public key → **Compile Python code** → Verify the configuration

7. Compile and run Python code on the client to log in to the server through SSH.

```
Import module                                          --  import paramiko
                                                           import time
Instantiate SSH objects.                               --  ssh = paramiko.SSHClient()
Allow connections to unknown hosts.                    --  ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
Set up an SSH session connection.                      --  ssh.connect(hostname='192.168.56.100',port=22,username='client',key_filename=r'C:
                                                           \Users\exampleuser\.ssh\id_rsa')
Open an interactive session.                           --  cli = ssh.invoke_shell()
Send the command for canceling screen splitting.       --  cli.send('screen-length  0 temporary\n')
Send the command for displaying the current configuration. --  cli.send('display cu\n')
Set the pause duration to 3 seconds.                   --  time.sleep(3)
Instantiate the received data.                         --  dis_cu = cli.recv(999999).decode()
Print the command output.                              --  print(dis_cu)
Close the SSH connection.                              --  ssh.close()
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Case: Verifying the Configuration on the Client

Enable STelnet on the device → Configure users → Generate a key pair → Configure the public key → Compile Python code → **Verify the configuration**

8. Run the code. The current configuration of the SSH server is displayed.

```
Info: The max number of VTY users is 5, the number of current VTY users online is 1, and total
number of terminal users online is 2.
    The current login time is 2020-05-12 11:04:56.
<SSH Server>screen-length 0 temporary
Info: The configuration takes effect on the current user terminal interface only.
<SSH Server>display cu
!Software Version V200R005C10SPC607B607
!Last configuration was updated at 2020-05-12 10:46:40+00:00
!Last configuration was saved at 2020-05-12 10:46:42+00:00
#
...
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

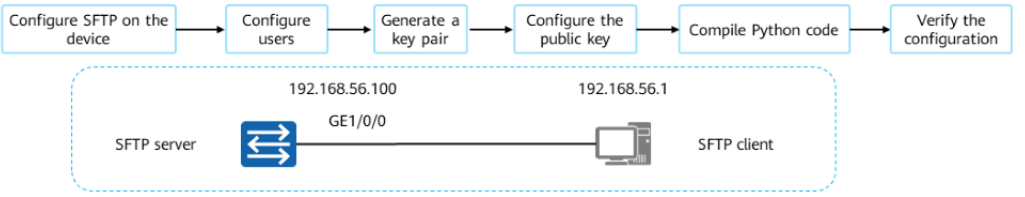
1. Introduction to SSH
2. Paramiko Component Architecture
3. **SSH Practices**
   - Practices in SSH Python Scripts
   - **Practices in SFTP Python Scripts**

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

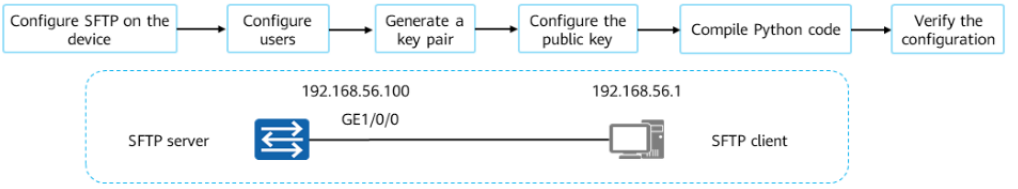
Case: Using SFTP to Upload and Download Files

- Description:
- SSH File Transfer Protocol (SFTP) is a secure file transfer protocol based on SSH. SFTP not only provides all functions of FTP, but also has higher security and reliability.
- As shown in the figure below, after the SFTP server function is enabled on the switch that functions as the SFTP server, the PC functioning as a client can log in to the SFTP server in password or RSA authentication mode to upload or download files.
- This case uses RSA user authentication as an example to describe how to upload and download files on the client through SFTP using the Paramiko module of Python.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

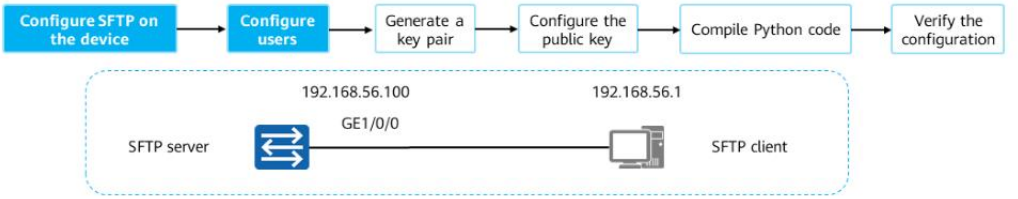
Configuration Roadmap



- Configuration on the server:
  - Configure SFTP. Specifically, configure the management IP address and enable SFTP on the device.
  - Create a user. Specifically, create an SSH user and configure the service type, authentication mode, and SFTP path.
  - Configure a public key. Specially, add the public key generated by the client and allocate it to the user.
  - Verify the configuration. Specifically, check the uploaded files.
- Configuration on the client:
  - Generate a key pair. Specifically, generate a public key and a private key locally.
  - Compile Python code.
  - Verify the configuration. Specifically, check the downloaded files.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Case: Configuring SFTP and Users on the Server



1. Configure the management IP address for the SFTP server and enable the SFTP server function.

```
<HUAWEI>system-view immediately
[HUAWEI] sysname SFTP Server
[SFTP Server] interface GE 1/0/0
[SFTP Server-GE1/0/0] ip add 192.168.56.100 24
[SFTP Server-GE1/0/0] quit
[SFTP Server] sftp server enable
```

2. Create the SSH user **client** and configure the authentication type and service type for the user.

```
[SFTP Server] ssh user client
[SFTP Server] ssh user client authentication-type rsa
[SFTP Server] ssh user client service-type sftp
[SFTP Server] ssh user client sftp-directory cfcard:
[SFTP Server] ssh authorization-type default root
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Case: Creating an RSA Key Pair on the Client



3. On the client, use Git Bash to create an RSA key pair (private key **id_rsa** and public key **id_rsa.pub**) and check the public key.

Generate an RSA public/private key pair.	--	exampleuser@exampleuser MINGW64 ~ $ ssh-keygen -t rsa Generating public/private rsa key pair.
Set the path for storing the key. (Press **Enter** to use the default path.)	--	Enter file in which to save the key (/c/Users/exampleuser/.ssh/id_rsa):
Enter the pass phrase. (Press **Enter** to use the default pass phrase.)	--	Enter passphrase (empty for no passphrase):
Enter the pass phrase again. (Press **Enter** to use the default pass phrase.)	--	Enter same passphrase again:
Path for storing the private key file of the client	--	Your identification has been saved in /c/Users/exampleuser/.ssh/id_rsa
Path for storing the public key file of the client	--	Your public key has been saved in /c/Users/exampleuser/.ssh/id_rsa.pub
Check the public key.	--	$ cat /c/Users/exampleuser/.ssh/id_rsa.pub

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Case: Configuring a Public Key on the Server

Configure SFTP on the device → Configure users → Generate a key pair → **Configure the public key** → Compile Python code → Verify the configuration

4. On the server, add the public key generated by the client and allocate it to the user.

```
[SFTP Server] rsa peer-public-key rsa01 encoding-type openssh
[SFTP Server-rsa-public-key] public-key-code begin
[SFTP Server-rsa-public-key-rsa-key-code] ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQDwLRx8MmuNs500dRemhFHdDbBmxco8Bp+wyqwaGuHJZBCjyFQV6AB+ezu5t0eWE3mw57IZfgm
vR+MjBcliZv/x3l8oUMLcQKlKslYQDtvfUCZd+za1suXAPB/dyPKMhYPAzSDA7K+xqCWlmU3q06vxHEPLMv4A5IX54rKtBnK92fWjl9ACU+ak0ZlH
xbKwOFn1tr0GJBazcInEs9DKGwkTTqJdu9+5hI5NxXT5bM3an53805ZbCU18xPy57g7MZC89vbdsag/uvQmFkLJ3arts/Om2R7fhR92EU/SNPmV
y+qDEdwZEVdubdqJInW+8zzVkPGlnb2oH5hwH78Ksklbxb0fEfmGR0mS1ZAi3ZHUGcEEjuFZona3+5Z0Un2OPxfXwvoljVDusbYcugJHo9Ssurz0
5GzVuamQZlcO2JYY6FhtLUAImtXGQ80MpTjB0lcprkAZCib8agYOtVQNTZ7iB0g2EcBN9UTyMz7sh8RtrBDj445r+XPaDE8LmpDRKHMk=
[SFTP Server-rsa-public-key-rsa-key-code] public-key-code end
[SFTP Server-key-code] peer-public-key end
[SFTP Server] ssh user client assign rsa-key rsa01
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- For details about the commands, refer to the product documentation at https://support.huawei.com/enterprise/en/doc/EDOC1000097293/466984de?idPath=24030814|21432787|21430822|22318704|9794900.

Case: Compiling Python Code on the Client

Configure SFTP on the device → Configure users → Generate a key pair → Configure the public key → **Compile Python code** → Verify the configuration

5. Compile and run Python code on the client to log in to the server through SFTP and upload and download files.

```
Import module                          --  import paramiko
Create an RSA key object.              --  key=paramiko.RSAKey.from_private_key_file(r'C:\Users\exampleuser\.ssh\id_rsa')
Instantiate the session channel.       --  tran = paramiko.Transport(('192.168.56.100', 22))
Set up an SSH session connection.      --  tran.connect(username='client',  pkey=key)
Set up an SFTP channel.                --  sftp = paramiko.SFTPClient.from_transport(tran)
Set the local path.                    --  local_path=r'C:\Users\exampleuser\.ssh\vrptest.cfg'
Set the remote path.                   --  remote_path='/vrpcfg.cfg'
Perform the download operation.        --  sftp.get(remote_path, local_path)
Perform the upload operation.          --  sftp.put(local_path,'/test.cfg')
Close the session.                     --  tran.close()
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Case: Verifying the Configuration on the Server and Client

Configure SFTP on the device → Configure users → Generate a key pair → Configure the public key → Compile Python code → **Verify the configuration**

6. Run the code. The client successfully downloads the specified file to the local host.

vrptest.cfg 2020/5/13 17:17

7. Run the dir command on the server. The specified file is successfully uploaded to the server.

```
<SFTP Server>dir
Directory of cfcard:/

 Idx  Attr     Size(Byte)  Date        Time        FileName
   0  dr-x             -  May 13 2020 16:27:30   $_checkpoint
   1  dr-x             -  Apr 28 2020 20:20:09   $_install_mod
   2  dr-x             -  Apr 28 2020 20:20:37   $_license
   3  dr-x             -  May 13 2020 11:01:17   $_security_info
   4  dr-x             -  May 13 2020 15:26:44   $_system
   5  -rw-         2,428  May 13 2020 17:17:13   test.cfg
   6  -rw-         2,493  May 12 2020 17:10:15   vrpcfg.cfg
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Quiz

1. (Multiple) Which of the following phases are involved in the SSH protocol? ( )
   A. Version negotiation phase
   B. Algorithm negotiation phase
   C. Key exchange phase
   D. User authentication phase
   E. Session interaction phase
2. Which method is used by Paramiko to enable an SSH interactive session?
3. Which method is used by Paramiko to establish an SSH session connection?

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

1. ABCDE
2. Method invoke_shell() of the SSHClient class
3. Method connect of the SSHClient class or method connect of the Transport class

Summary

- This course describes the concepts of Paramiko and SSH, and illustrates the working principles of SSH.
- This course also describes the components and common methods of Paramiko. In the last part, this course uses example scripts of Python SSH and SFTP to show the use and practices of Paramiko methods, thereby implementing preliminary network automation based on SSH.
- For more information, visit Paramiko's official website, read SSH RFC documents, and learn upper-layer SSH libraries such as Fabric. Fabric is developed based on Paramiko and is further encapsulated to improve SSH-based application deployment and system management efficiency.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

More Information

- Paramiko official websites
  - docs.paramiko.org/en/latest/index.html
  - www.paramiko.com
- SSH RFC documents
  - https://tools.ietf.org/html/rfc4251.html
  - https://tools.ietf.org/html/rfc4252.html
  - https://tools.ietf.org/html/rfc4253.html
  - https://tools.ietf.org/html/rfc4254.html
- Fabric official website
  - https://fabric-chs.readthedocs.io/zh_CN/chs/tutorial.html

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.


Thank You
www.huawei.com



SNMP Fundamentals and Practices

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Foreword

- As a network expands in scale, network management becomes complex and inefficient because various types and models of devices from different vendors coexist on the network. To simplify network management, the Simple Network Management Protocol (SNMP) is developed to standardize network management interfaces and protocols and uniformly manage all devices on the network.
- To cope with diversified and changing O&M requirements, network administrators need to have certain programming capabilities to improve O&M efficiency. PySNMP implements SNMP functions and is a powerful tool for network administrators to improve O&M capabilities.
- This course describes basic functions, fundamentals, and basic configurations of SNMP. It also describes PySNMP and its practices.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Objectives

- Upon completion of this course, you will be able to:
  - Describe basic SNMP concepts.
  - Describe basic SNMP functions.
  - Understand SNMP fundamentals.
  - Perform basic SNMP configurations.
  - Use PySNMP to perform SNMP operations.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

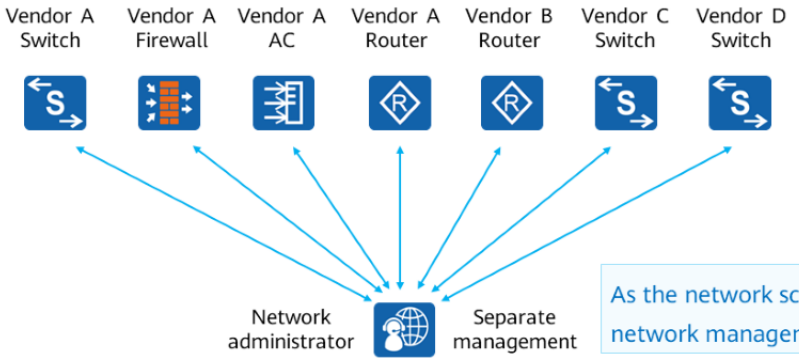
Contents

1. **Overview of SNMP**
2. SNMP Fundamentals
3. Basic SNMP Configurations
4. Introduction to PySNMP

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Overview of SNMP (1/3)

- When various types and models of devices from different vendors coexist on a network, separate network management is complex and inefficient.

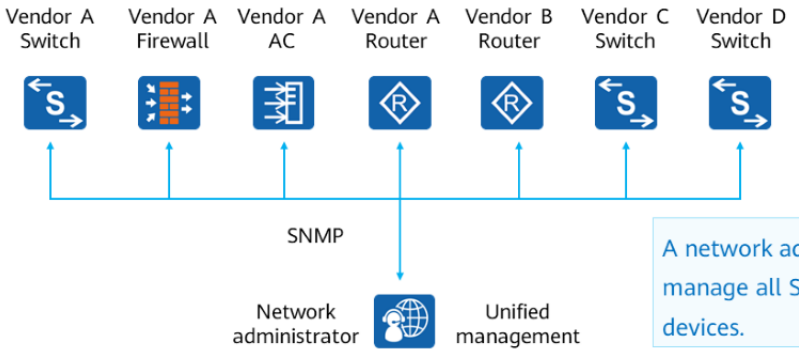


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Different vendors use their proprietary protocols to manage their own devices, resulting in complex and inefficient network management.

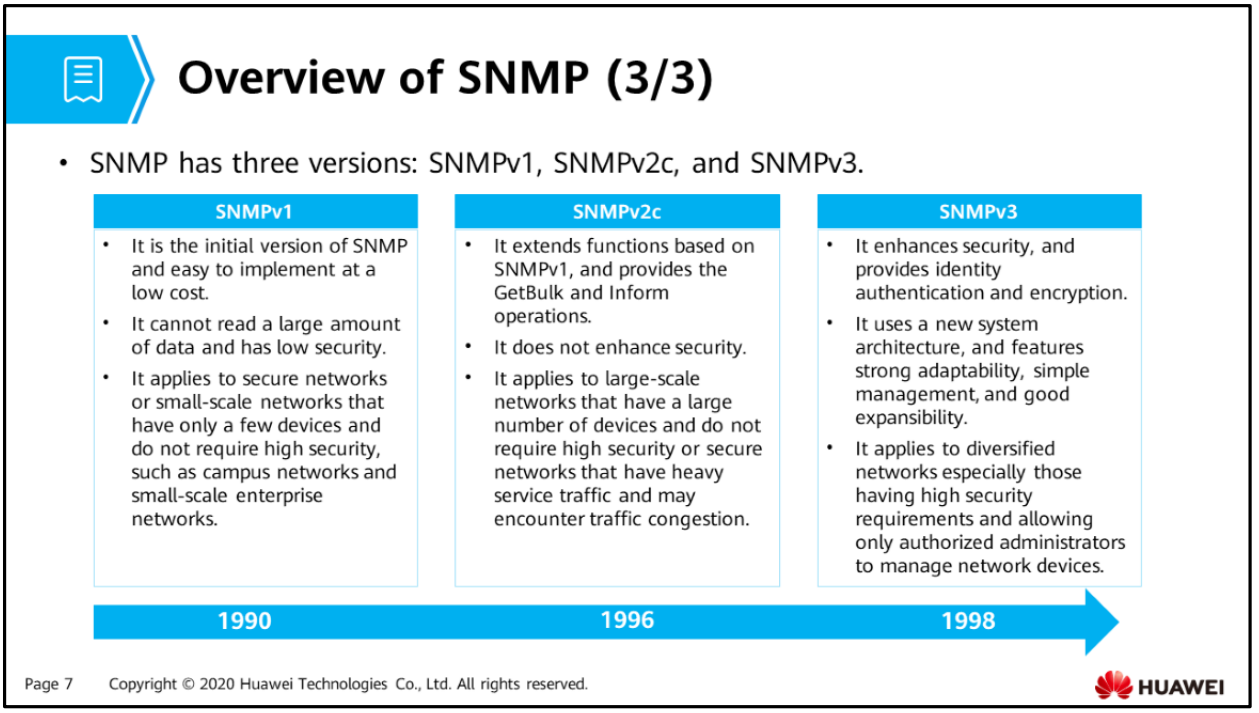
Overview of SNMP (2/3)

- The Internet Engineering Task Force (IETF) defines SNMP to implement unified management of devices.



A network administrator can uniformly manage all SNMP-capable network devices.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



- SNMPv2c
  - GetBulk operation: is performed by a network management system (NMS) to query information about managed devices in batches.
  - Inform operation: is performed by a managed device to notify the NMS of a fault or event on the device and requires an acknowledgement from the NMS.
- SNMPv3
  - Strong adaptability: It adapts to both simple and complex networks.
  - Simple management: It consists of multiple independent sub-systems and applications. When a fault occurs in an SNMP system, it is easy to locate the sub-system where the fault is originated according to the fault type.
  - Good expansibility: Modules can be added to expand an SNMP system. For example, a module can be added to the security sub-system to run a new security protocol.

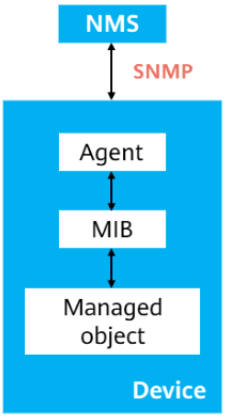
Contents

1. Overview of SNMP
2. **SNMP Fundamentals**
3. Basic SNMP Configurations
4. Introduction to PySNMP

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

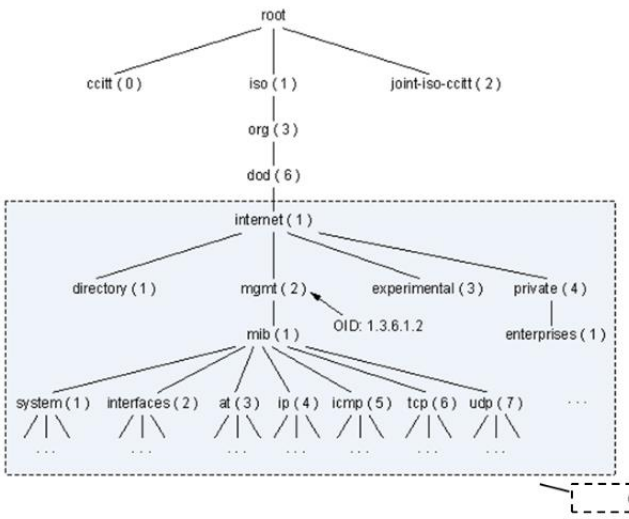
SNMP Management Model

- Basic SNMP concepts:
  - Network management system (NMS): a system that uses SNMP to manage network devices
  - Agent: a process running on a managed device that interacts with the NMS
  - Management information base (MIB): a database containing the variables (information that can be queried and set by the agent) maintained by a managed device
  - Managed object: an object to be managed on a network device, for example, a hardware component (such as an interface card) or a set of parameters configured for a piece of hardware or software
  - Device: a device managed by the NMS



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

MIB



- A MIB is a database containing the variables (information that can be queried and set by the agent) maintained by a managed device. It defines the attributes of managed objects, including:
  - Object identifier (OID)
  - Status
  - Access permission
  - Data type
- A MIB uses a data structure containing all objects that may be managed on the network. Because the data structure is similar to a tree structure, a MIB is also called an object naming tree.

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- A MIB is independent of a specific network management protocol. Device vendors can integrate SNMP agent software into their products (for example, routers), but they must ensure that this software complies with relevant standards after new MIBs are defined. Users can use the same network management software to manage routers containing MIBs of different versions. However, the network management software cannot manage a router that does not support the MIB function.
- MIBs fall into public and private ones.
  - Public MIBs are defined by RFCs and used for structure design of public protocols and standardization of interfaces. Most device vendors need to provide SNMP interfaces according to the specifications defined in RFCs.
  - Private MIBs are the necessary supplement of public MIBs. Some enterprises develop proprietary protocols or special functions, they can design private MIBs to improve the management functions of SNMP interfaces. In addition, private MIBs enable third-party NMS software to manage devices that use proprietary protocols or have special functions. For example, OIDs of Huawei MIB objects start with 1.3.6.1.4.1.2011.

Common MIB Objects

- Objects used for query or modification

OID	Object Name	Syntax	Maximum Access	Description
1.3.6.1.2.1.2.1	ifNumber	Integer	read-only	Number of network interfaces in the system (regardless of the interface status).
1.3.6.1.4.1.2011.5.25.41.1.2.1.1.3	hwIpAdEntNetMask	IpAddress	read-create	Subnet mask of an IP address.

- Object used for trap notification

OID	Object Name	Bound Variable	Description
1.3.6.1.6.3.1.1.5.3	linkDown	ifIndex ifAdminStatus ifOperStatus ifDesc	It is detected that the ifOperStatus object has changed to the Down state from another state (not notPresent) which is represented by the value of ifOperStatus.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- The maximum access of a MIB object specifies the operations that an NMS can perform on the MIB object. The options include:
  - not-accessible: No operation can be performed.
  - read-only: Read information.
  - read-write: Read information and modify configurations.
  - read-create: Read information, and modify, create, and delete configurations.
- When generating a trap, a device reports the type of the current trap together with some variables. For example, when sending a linkDown trap, a device also sends variables such as the index and current configuration status of the involved interface.
  - ifIndex: indicates the interface index.
  - ifAdminStatus: indicates the interface management status, that is, whether the interface is shut down. The options include undo shutdown and shutdown.
  - ifOperStasuts: indicates the interface operational status, that is, the link layer protocol status of the interface. The options include Up and Down.
  - ifDesc: indicates the interface description.

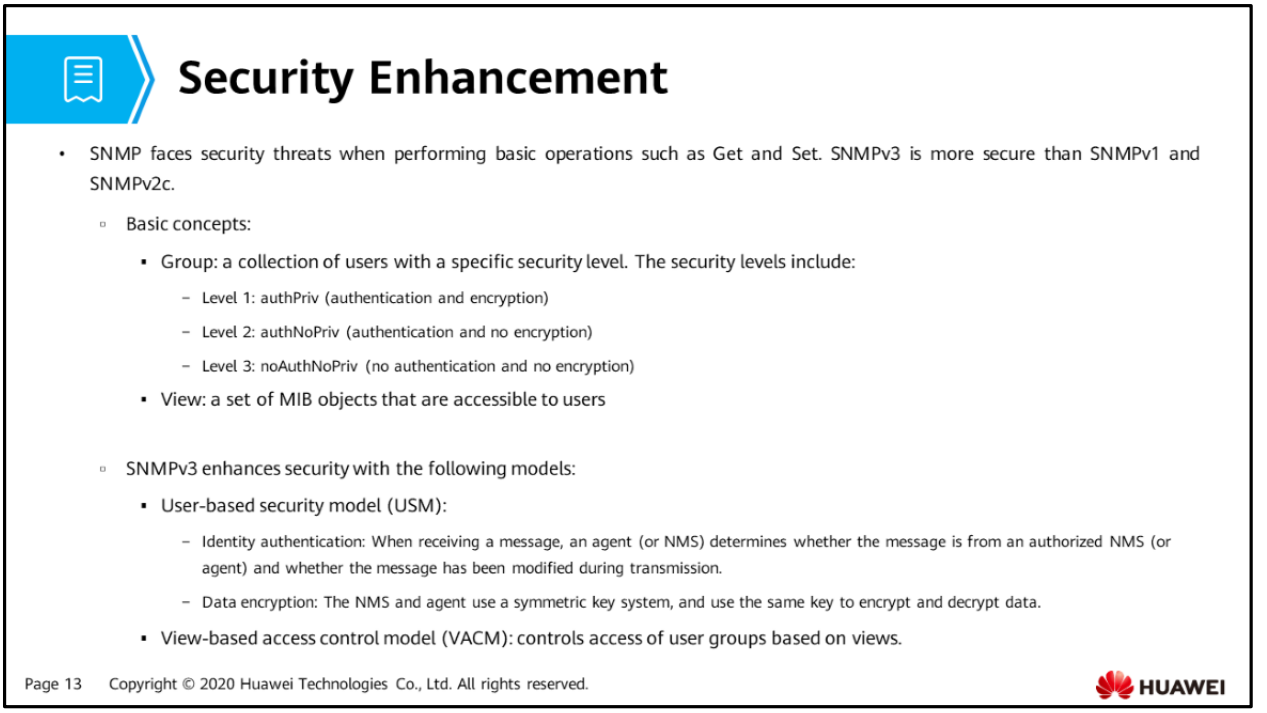
Basic SNMP Operations

- An NMS uses SNMPv3 to send query and set operation instructions to managed devices, receive operation responses, and listen on trap information sent by managed devices.

Function	SNMPv3 Operation	Description
Query	Get	Obtains one or more variables from an SNMP agent.
	GetNext	Obtains the next variable from an SNMP agent in lexicographical order.
	GetBulk	Queries information from an SNMP agent in batches.
Set	Set	Requests an SNMP agent to set one or more variables.
Trap	Trap	An SNMP agent sends traps to notify the NMS of events generated by the managed device.
	Inform	Provides similar functions to the trap operation. The difference is that the inform operation requires an acknowledgement from the NMS. Therefore, informs occupy more system resources than traps.
Response	Response	A response message is sent by an SNMP agent in response to the Get or Set operation or by an NMS in response to an inform.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

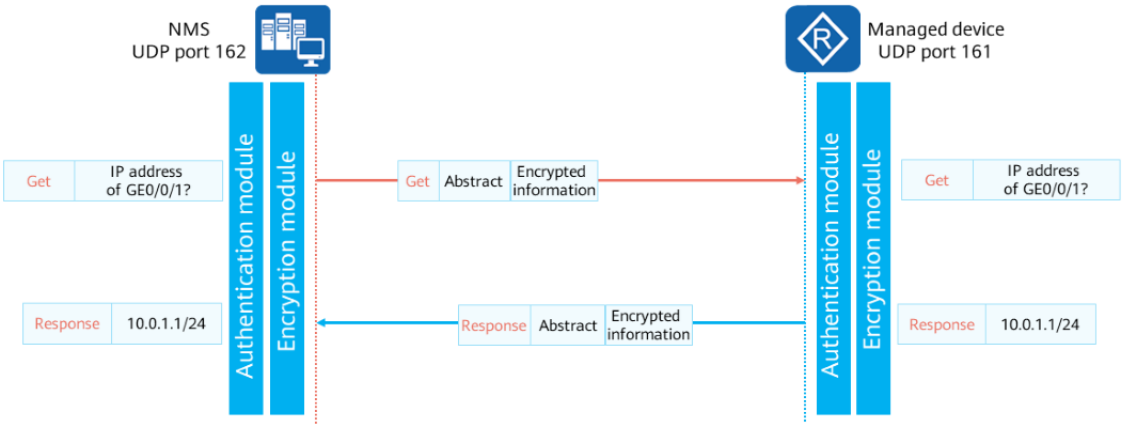
- This document describes SNMPv3 operations.



- SNMPv1 and SNMPv2c provide user authentication based on community names, which has the following security risks:
  - Most network product vendors deliver their devices with the default SNMP read-only community name being Public and the default read-write community name being Private. In most cases, network management personnel do not modify the default settings.
  - SNMP community names are transmitted in plain text on the network, which may cause information leakage.

SNMP Operation: Get

- Get operation: obtains one or more variables from the SNMP agent of a managed device. The following shows how an NMS obtains the IP address of a specified interface on a managed device through the Get operation.

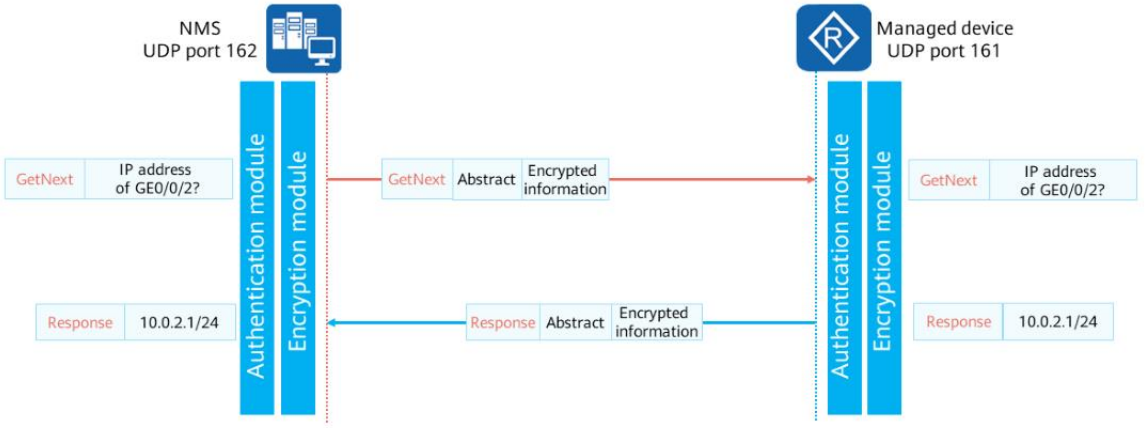


 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- The listening UDP port of a managed device is 161, so Get, GetNext, GetBulk, and Set packets are destined for port 161.
- The listening UDP port of the NMS is 162, so Trap and Inform packets are destined for port 162.
- The authentication and encryption modules provide the same identification and encryption functions as the USM in SNMPv3 to encrypt the sent SNMP packets and authenticate received the SNMP packets respectively.

SNMP Operation: GetNext

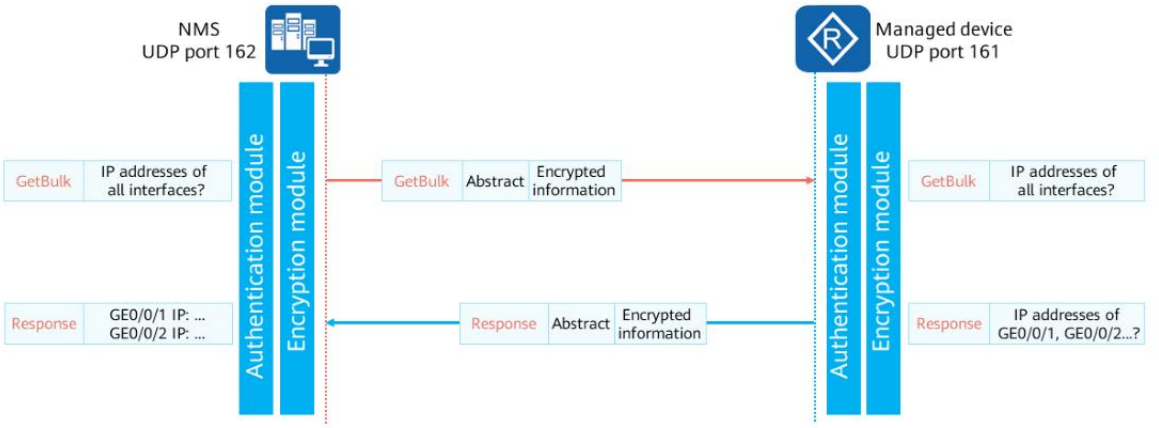
- GetNext: obtains the next variable from an SNMP agent in lexicographical order. The following shows how an NMS obtains the IP address of the next interface on a managed device through the GetNext operation.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

SNMP Interaction: GetBulk

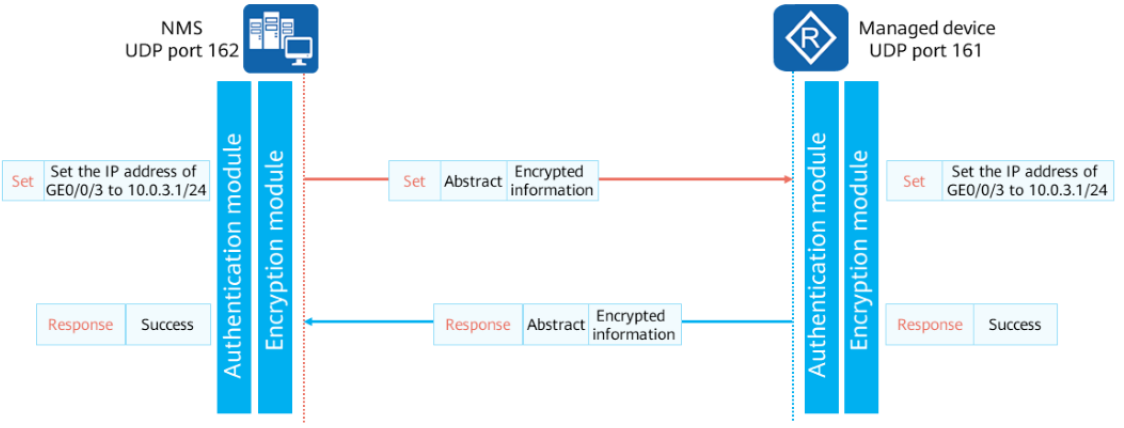
- GetBulk: queries information from an SNMP agent in batches. The following shows how an NMS queries IP addresses of all interfaces on a managed device through the GetBulk operation.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

SNMP Interaction: Set

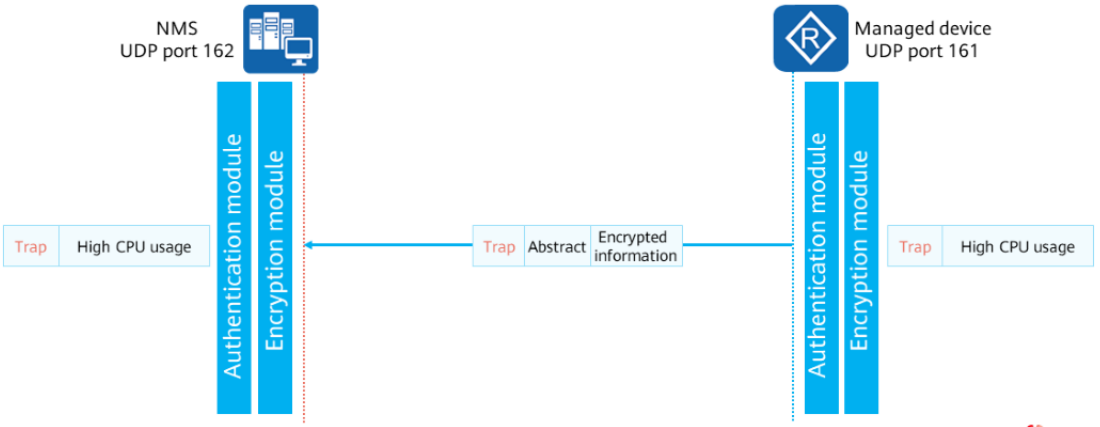
- Set: sets one or more variables through an SNMP agent. The following figure shows how an NMS sets an IP address for a specified interface on a managed device through the Set operation.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

SNMP Interaction: Trap

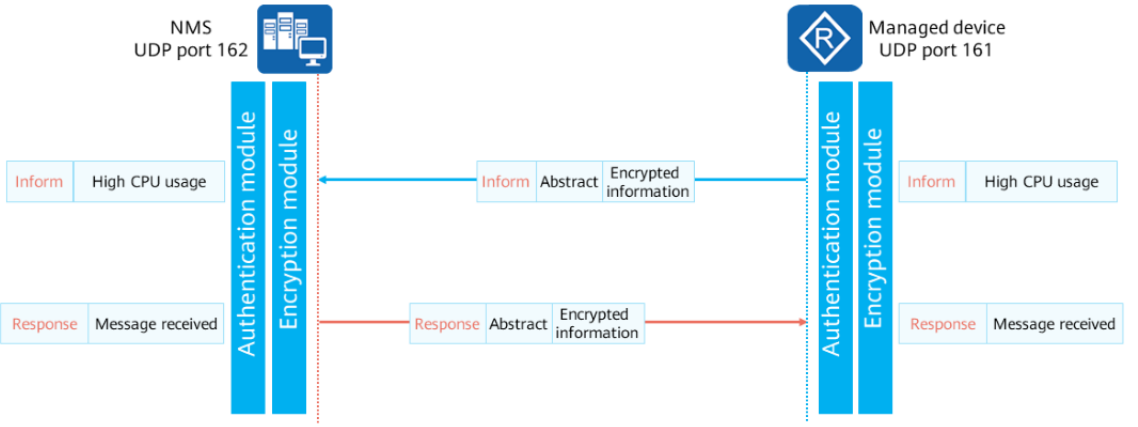
- Trap: The SNMP agent of a managed device sends traps to notify an NMS of certain events generated by the managed device. The following shows how the SNMP agent of a managed device sends a trap to notify an NMS of high CPU usage of the device.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

SNMP Interaction: Inform

- Inform: provides similar functions to the trap operation. The difference is that the inform operation requires an acknowledgement from the NMS. The following shows how the SNMP agent of a managed device sends an inform to notify an NMS of high CPU usage of the device.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. Overview of SNMP
2. SNMP Fundamentals
3. **Basic SNMP Configurations**
4. Introduction to PySNMP

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Basic SNMP Configurations (1/3)

1. Enable the SNMP agent function.

   [Huawei] **snmp-agent**

2. Configure the SNMP version.

   [Huawei] **snmp-agent sys-info version [v1 | v2c | v3]**

   The SNMP version used on the device must be the same as that on the NMS.

3. Create or update a MIB view.

   [Huawei] **snmp-agent mib-view** *view-name* { **exclude** | **include** } *subtree-name* [**mask** *mask*]

4. Create an SNMP group by mapping SNMP users to SNMP views.

   [Huawei] **snmp-agent group v3** *group-name* **{ authentication | noauth | privacy } [read-view** *view-name* **| write-view** *view-name* **| notify-view** *view-name* **]**

   This command is used to create an SNMPv3 group, specify the authentication and encryption modes, and configure one or more of the following views: read-only view, read-write view, and notification view. It is a mandatory command on networks that require high security.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Basic SNMP Configurations (2/3)

5. Add a user to an SNMP group.

```
[Huawei] snmp-agent usm-user v3 user-name group group-name
```

6. Configure the authentication mode and key for an SNMPv3 user.

```
[Huawei] snmp-agent usm-user v3 user-name  authentication-mode { md5 | sha | sha2-256 }
```

7. Configure the encryption mode and key for an SNMPv3 user.

```
[Huawei] snmp-agent usm-user v3 user-name privacy-mode { aes128 | des56 }
```

8. Set parameters for the device to send traps.

```
[Huawei] snmp-agent target-host trap-paramsname paramsname v3 securityname securityname {
authentication | noauthnopriv | privacy }
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Basic SNMP Configurations (3/3)

9. Configure the target host of traps.

```
[Huawei] snmp-agent target-host trap-hostname hostname address ipv4-address trap-paramsname paramsname
```

10. Enable all trap functions.

```
[Huawei] snmp-agent trap enable
```

This command only enables the device to send traps. It must be used together with the **snmp-agent target-host** command, which specifies the host to which traps are to be sent.

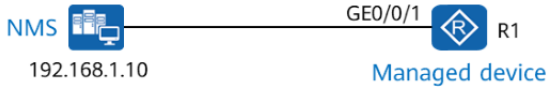
11. Configure the source interface from which traps are to be sent.

```
[Huawei] snmp-agent trap source interface-type interface-number
```

An IP address must have been configured for this source interface.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

SNMPv3 Configuration Example (on a Managed Device)



- Enable SNMP on R1 and set the SNMP version to **v3**.
- Create the SNMPv3 group **test**, and set both the encryption and authentication modes to **privacy**.
- Create the SNMPv3 user **R1**, and set both the authentication and encryption keys to **HCIP-Datacom123**.
- Create the trap sending parameter list **param** and set the user security name to **sec**.
- Set the IP address of the target host for receiving traps to **192.168.1.10**.
- Enable the trap function and specify GE0/0/1 as the source interface from which traps are to be sent.

Configurations on R1:

```
[R1]snmp-agent
[R1]snmp-agent sys-info version v3
[R1]snmp-agent group v3 test privacy
[R1]snmp-agent usm-user v3 R1 test authentication-mode md5 HCIP-Datacom123 privacy-mode aes128 HCIP-Datacom123
[R1]snmp-agent target-host trap-paramsname param v3 securityname sec privacy
[R1]snmp-agent target-host trap-hostname nms address 192.168.1.10 trap-paramsname param
[R1]snmp-agent trap source GigabitEthernet 0/0/1
[R1]snmp-agent trap enable
Info: All switches of SNMP trap/notification will be open.
Continue? [Y/N]:y
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. Overview of SNMP
2. SNMP Fundamentals
3. Basic SNMP Configurations
4. **Introduction to PySNMP**

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

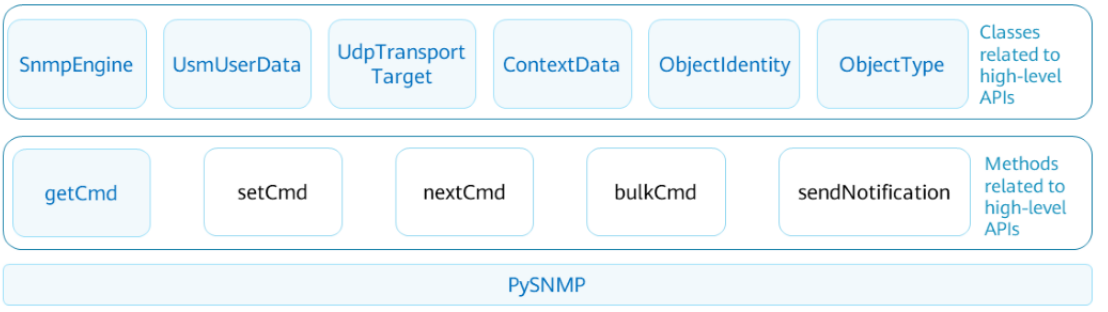
Introduction to PySNMP

- PySNMP is a third-party module of Python and implements all functions of SNMPv1, SNMPv2c, and SNMPv3. The latest version of PySNMP is v4.4.12. Users can use Python to implement all SNMP operations by using PySNMP.
- PySNMP provides easy-to-use high-level APIs to simplify the coding process and improve the coding efficiency.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

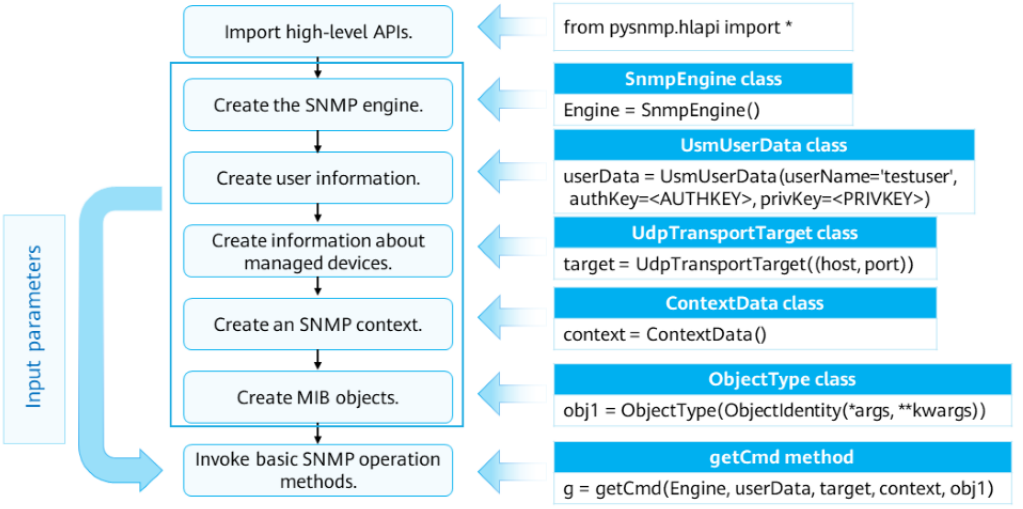
High-Level API Architecture of PySNMP

- The following shows the composition of PySNMP high-level APIs. Basic SNMPv3 operations involve the SnmpEngine, UsmUserData, UdpTransportTarget, ContextData, ObjectIdentity, and ObjectType classes, as well as the getCmd, setCmd, nextCmd, bulkCmd and sendNotification methods. This document describes only the getCmd method. The other methods are used similarly.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

PySNMP Encoding Process



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- The SnmpEngine, UsmUserData, UdpTransportTarget, ContextData, and ObjectType class instances of PySNMP high-level APIs are used as input parameters of PySNMP methods such as getCmd.
- The ObjectType class is initialized using the ObjectIdentity class instance.

SnmpEngine Class

- SnmpEngine class: is a central, umbrella object in PySNMP. In PySNMP, all SNMP operations involve the SnmpEngine class instance.
- Using this class:

```
engine = SnmpEngine()
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

UsmUserData Class

- UsmUserData class: provides SNMPv3 USM configuration for SNMPv3 systems. This class can be used to create SNMPv3 users and the corresponding authentication and encryption algorithms.
- Using this class:

```
userData= UsmUserData(
                        'admin',                                # User name
                        authKey = 'Admin@123',                  # Authentication key
                        privKey = 'Huawei@123',                 # Encryption key
                        authProtocol = usmHMACSHAAuthProtocol,  # Authentication algorithm
                        privProtocol = usmAesCfb128Protocol     # Encryption algorithm
                        )
```

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- For options of the authentication and encryption algorithms, visit http://snmplabs.com/pysnmp/docs/api-reference.html?highlight=usmuserdata#pysnmp.hlapi.UsmUserData.

UdpTransportTarget Class

- UdpTransportTarget class: specifies the IP address and port number of a managed device.
- Using this class:

```
target = UdpTransportTarget(("192.168.56.101", 161))        # (host, port) indicates the IP address and port number of a managed device.
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

ContextData Class

- ContextData class: specifies SNMP context.
- Using this class:

```
context = ContextData()                    # If no parameter is specified when the ContextData class instance is initialized, the context object is empty.
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

ObjectIdentity Class

- ObjectIdentity class: represents the OID of a MIB object.
- Using this class:

```
oid1 = ObjectIdentity('SNMPv2-MIB',' sysName', 0)          # sysName object instance
oid2 = ObjectIdentity('1.3.6.1.2.1.1.5.0')                  # Use the OID string of the sysName object
                                                              to initialize the ObjectIdentity class.
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

ObjectType Class

- ObjectType class: represents a MIB object, which is initialized using the ObjectIdentity object.
- Using this class:

```
obj1 = ObjectType(ObjectIdentity('SNMPv2-MIB',' sysName', 0))          # Instantiate the ObjectType object of sysName.
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Method getCmd

- getCmd method: implements the SNMP Get operation. The returned value is a generator.
- Method declaration: getCmd(snmpEngine, authData, transportTarget, contextData, *varBinds)

Parameter	Description
snmpEngine	SnmpEngine class instance
authData	UsmUserData class instance
transportTarget	UdpTransportTarget class instance
contextData	ContextData class instance
*varBinds	ObjectType class instance

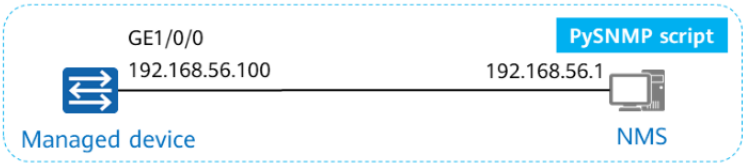
- Using this method:

```
g = getCmd(snmpEngine, authData, transportTarget, contextData, *varBinds)        # g is a generator.
# When the Python next method is used, a Get operation is generated. The obtained result is saved in varBinds. The other three returned values indicate error information.
errorIndication, errorStatus, errorIndex, varBinds =next(g)
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

PySNMP Practice (1/4)

- You can use the PySNMP classes and methods to perform the SNMP Get operation. The following case describes how to use Python to write code for the SNMP Get operation.
- Case description: A managed device is connected to an NMS and has SNMP enabled. PySNMP is used on the NMS to compile a Python script to obtain the sysName of the managed device. The following figure shows the networking.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

PySNMP Practice (2/4)

- The managed device delivers the following configurations to enable the SNMP service:

```
snmp-agent usm-user v3 admin group dc-admin                          # Create an SNMP user admin.
snmp-agent usm-user v3 admin authentication-mode sha                  # Set the authentication algorithm to HMAC-SHA-96.
Huawei@123                                                           # Set the authentication password.
Huawei@123
Set the snmp-agent usm-user v3 admin privacy-mode aes128             # Set the encryption algorithm to AES-128.
Huawei@123                                                            # Set the encryption password.
Huawei@123

snmp-agent trap source ME0/0/0                                       # Specify the source interface for sending traps.
snmp-agent mib-view included nt iso                                  # Create a MIB view.
snmp-agent mib-view included rd iso
snmp-agent mib-view included wt iso
snmp-agent mib-view included iso-view iso
snmp-agent group v3 dc-admin privacy read-view rd write-view wt notify-view nt        # Create a user group.
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

PySNMP Practice (3/4)

- Compile a Python script:

```
from pysnmp.hlapi import *                    # Import high-level APIs of PySNMP.

userData = UsmUserData(                       # SNMP user information
                        'admin',
                        'Huawei@123',
                        'Huawei@123',
                        authProtocol=usmHMACSHAAuthProtocol,
                        privProtocol=usmAesCfb128Protocol)
g = getCmd(
        SnmpEngine(),
        userData,
        UdpTransportTarget(("192.168.56.100", 161)),
        ContextData(),
        ObjectType(ObjectIdentity('SNMPv2-MIB','sysName',0))
        )
errorIndication, errorStatus, errorIndex, varBinds =next(g)      # Send a Get request to obtain the sysName value.
for i in varBinds:
          print(i)                                               # Print the sysName value.
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Code explanation:
  - Use the "from...import" statement to import high-level APIs of PySNMP.
  - Create the UsmUserData class instance — userData, and set the user name, authentication algorithm, encryption algorithm, authentication key, and encryption key.
  - Invoke the getCmd method and initialize the SnmpEngine, UdpTransportTarget, ContextData, and ObjectType class instances when they are input as parameters. Compared to initializing the class instances before the getCmd is invoked, this simplifies the coding process.
  - Use the "errorIndication, errorStatus, errorIndex, varBinds =next(g)" statement to perform the next operation on the generator g obtained using the getCmd method. The obtained information about MIB objects is saved in varBinds.
  - Finally, use loop statements to print information about all MIB objects in varBinds. In this example, only one MIB object is available in varBinds.

PySNMP Practice (4/4)

- Run the Python script on the NMS to implement the Get operation. The result is as follows:

```
SNMPv2-MIB::sysName.0 = CE1
```

- Summary: The values of other MIB objects on the managed device can also be obtained using the Python script that is used to implement a Get operation. The scripts for implementing the Set, GetNext, GetBulk, Trap, and Inform operations by using PySNMP are similar to this Python script.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- For the values of MIB objects on a managed device, refer to the product documentation of the managed device.

Quiz

1. (Multiple) Compared with SNMPv1 and SNMPv2c, what are the improvements of SNMPv3 in terms of security? ( )

   A. Identity authentication

   B. Encryption

   C. View-based access control

   D. Authentication based on community names

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

1. ABC

Summary

- SNMPv3 is more secure than SNMPv1 and SNMPv2c.
- PySNMP can implement SNMP functions, helping engineers improve O&M efficiency.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

More Information

- For more information about the SNMPv3 framework, see RFC 3411.
- For more information about the USM for SNMPv3, see RFC 5590.
- For more information about the VACM for SNMPv3, see RFC 3415.
- PySNMP projects: https://github.com/etingof/pysnmp
- PySNMP documentation: http://snmplabs.com/pysnmp/

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.


Thank You
www.huawei.com



NETCONF YANG Principles and Practices

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Foreword

- For device configuration management, engineers are more accustomed to using the CLI to interact with devices. This mode is simple, direct, and easy to understand. However, in the network automation field, the CLI mode has problems such as low efficiency and difficult command understanding.
- To simplify device configuration and management, Internet Engineering Task Force (IETF) has set up work groups and released NETCONF, YANG, and RESTCONF standards.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Objectives

- On completion of this course, you will be able to:
  - Describe the problems and challenges of traditional network configuration management.
  - Describe the basic principles of NETCONF, YANG, and RESTCONF.
  - Describe the differences and relationships between NETCONF, YANG, and RESTCONF.
  - Understand the working principles of NETCONF and RESTCONF.

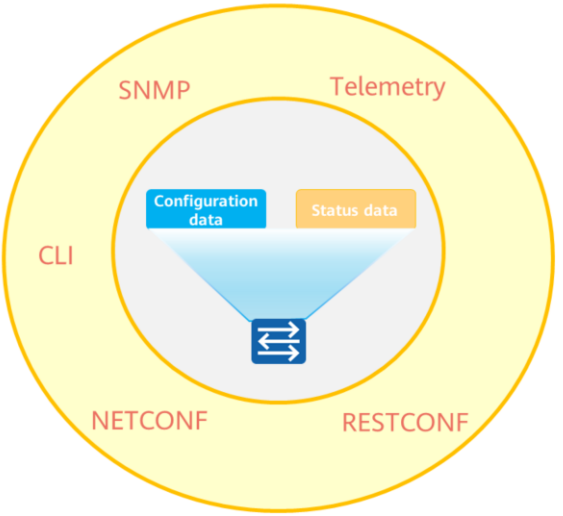
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. **Network Management Technology Background**
2. NETCONF Protocol
3. Modeling Language of YANG
4. RESTCONF Protocol

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Introduction to Device Data

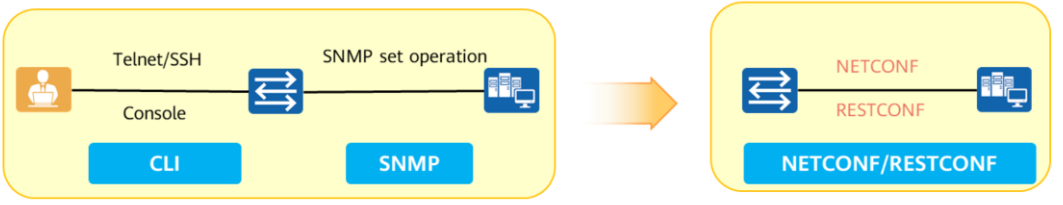


- Data is the carrier of information. Information that can be obtained from a running device is classified into configuration data and status data.
  - Configuration data is writable. It can change the system status, for example, from the initial state to the current state.
  - Status data is read-only non-configuration data in a system, for example, status information and statistics information.
- There are multiple methods to modify device data, either based on command lines or based on specific protocols. The CLI and SNMP are the most typical methods for obtaining device data.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Configuring a Network Device

- You can configure and manage a device in multiple ways. You can use the console cable to directly connect to the device or use SSH to remotely log in to the device, and then use the CLI to configure the device. You can also use the NMS server to set parameters in the MIB node of the device through SNMP set.
- With the increase of network scale and complexity, the preceding two methods cannot satisfy the configuration management requirements. To resolve this problem, NETCONF based on Extensible Markup Language (XML) is introduced.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- SNMP is based on UDP and is stateless, unordered, and unreliable for configuration management.
- SNMP can be configured for only one object, not for one service. During the concurrent configuration of multiple objects, if some objects are successfully configured but some objects fail to be configured, unknown impacts will be caused on the network.
- The SNMP interface is difficult to understand.

Typical Configuration Management Methods

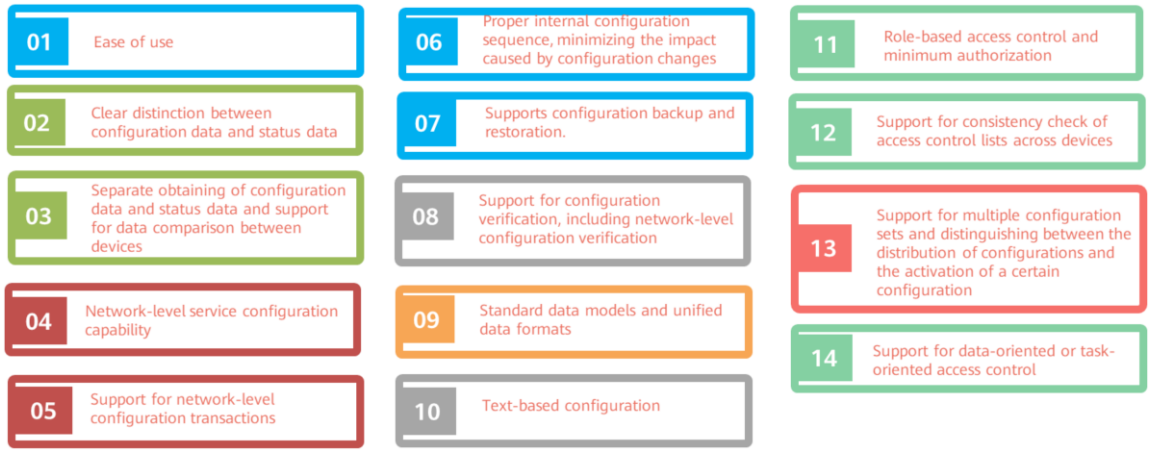
	Advantage	Disadvantage
CLI	1. It uses a text interface, which is easy to understand. 2. Based on Telnet/SSH, it is easy to use.	1. Vendors have different definitions. Carriers need to learn and develop adaptation scripts for each vendor. 2. Configuration scripts are unstructured, unpredictable, and easy to change, which makes parsing complex. CLI scripts are difficult to maintain, and automatic parsing is difficult to implement.
SNMP	1. It uses an interface between machines. The data model file (MIB) is available. 2. It is mainly used for network monitoring.	1. Based on the UDP protocol, it is stateless, unordered, and unreliable. 2. Configuration can be performed on objects only one by one, but does not orient to a service. During the concurrent configuration of multiple objects, if some objects are successfully configured but some objects fail to be configured, unknown impacts may be caused on the network. 3. SNMP manages only a single device and does not support network-level configuration or multi-device configuration collaboration. 4. The binary interface is difficult to understand. 5. It is insecure.

No matter how configuration automation technologies develop, the CLI is still used in essence.

SNMP is not suitable for configuration management.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

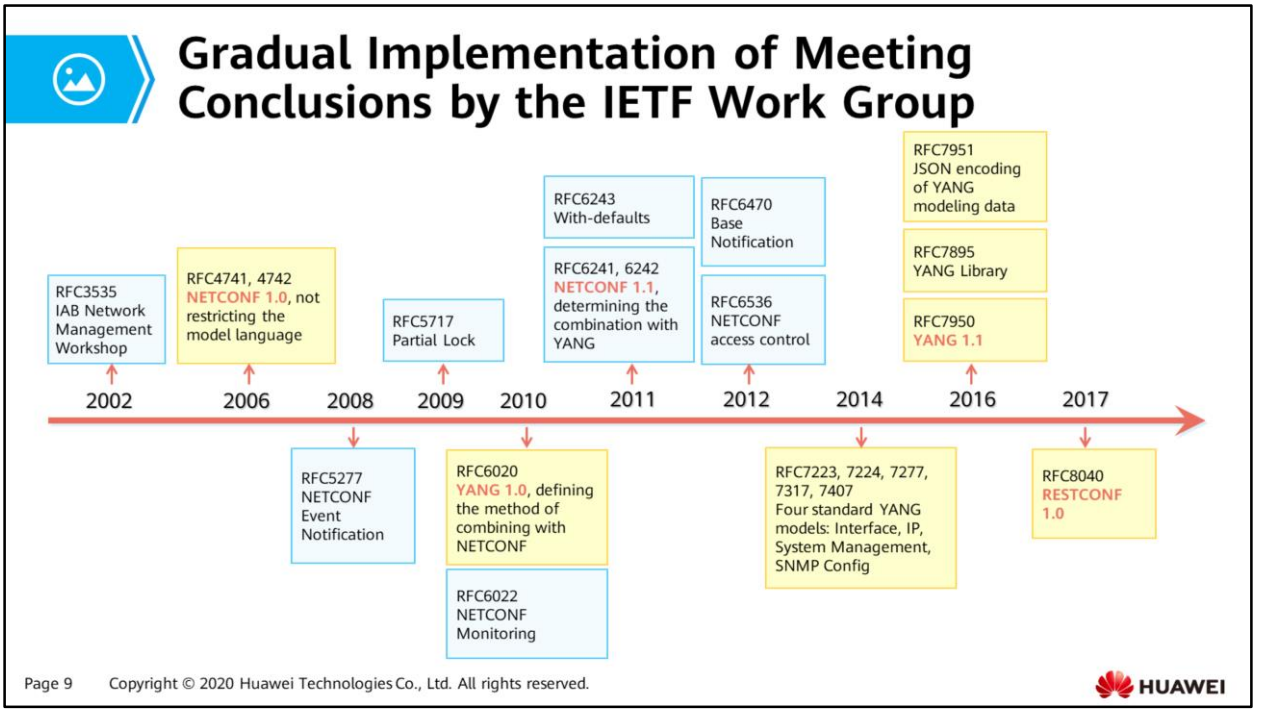
Requirements for Network Management



- At the 2002 IAB Network Management Workshop, 14 essential requirements for network management were raised.

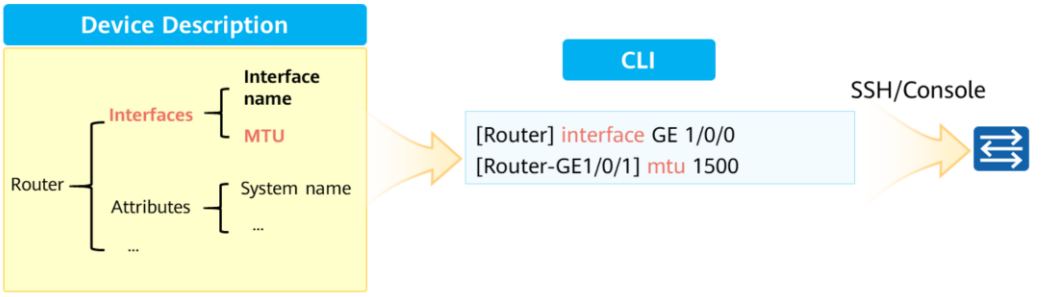
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- For details, see RFC 3535.
- Different IETF work groups and drafts gradually meet 14 requirements.



- The IETF gradually implements the conclusions of the IAB meeting. Different work groups gradually improve the 14 requirements.
- NETCONF 1.0 has no requirements on the model language. The combination between NETCONF 1.1 and YANG is determined.

Process for an Engineer to Configure Devices

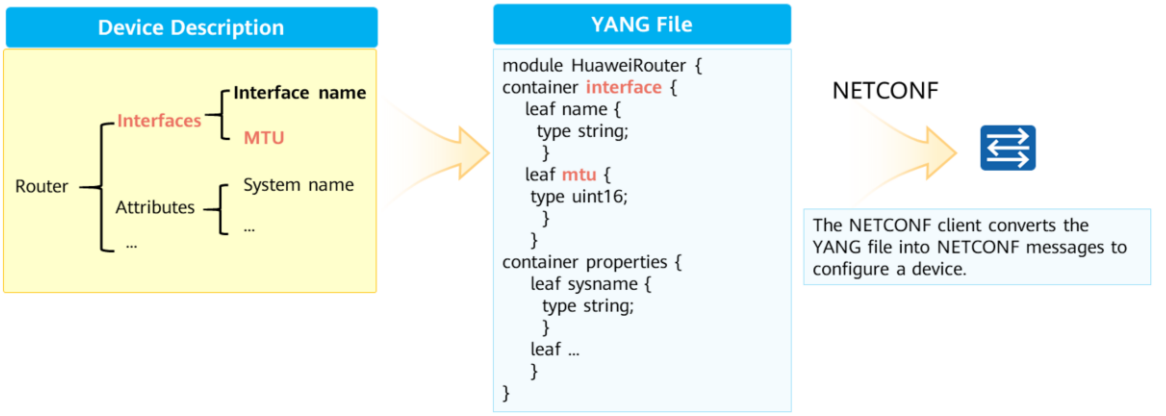


- The command line is an interactive language between humans and devices. Engineers query the product documentation and configure a device using the CLI.
- Command lines are nested to implement device configuration.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Process for NETCONF to Configure Devices

- A YANG file describes device data in another way.
- The YANG model uses the module-container-leaf structure to describe devices. For example, the YANG model defines field types and specifications for router interfaces and attributes.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- This example is not a real example. The YANG model does not take the entire device as one YANG file. Instead, the YANG model splits it into multiple YANG files by function.

Contents

1. Network Management Technology Background
2. **NETCONF Protocol**
3. Modeling Language of YANG
4. RESTCONF Protocol

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

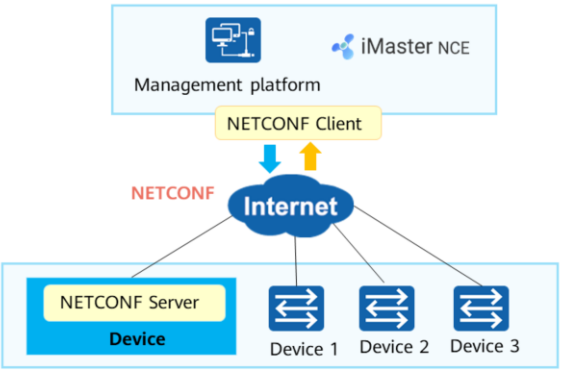
HUAWEI

NETCONF Overview

- NETCONF provides a network device management mechanism. You can use NETCONF to add, modify, or delete configurations of network devices, and obtain configurations and status of network devices.

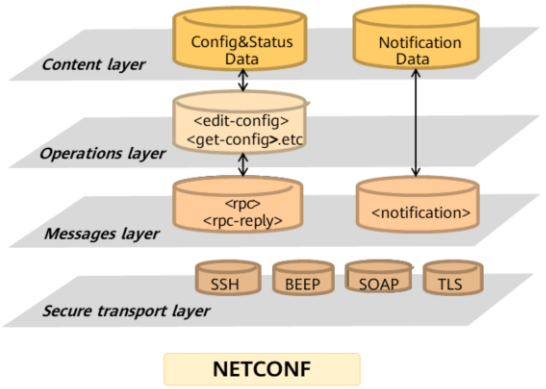
NETCONF has three objects:

- NETCONF client
- NETCONF server
- NETCONF message



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

NETCONF Protocol Framework



NETCONF is partitioned into four layers, as described in the following table.

- Secure transport layer: provides a communication path for interaction between the client and server. Currently, Huawei uses SSH as the transport protocol of NETCONF.
- Messages layer: provides a simple RPC request and response mechanism independent of the transport protocol layer. The NETCONF client uses an <rpc> element to encapsulate RPC request information and sends the information to the NETCONF server. Upon receipt, the NETCONF server uses an <rpc-reply> element to encapsulate RPC response information and returns the information to the NETCONF client.
- Operations layer: defines a group of basic operations as RPC invoking methods. These operations constitute basic NETCONF capabilities.
- Content layer: describes configuration data involved in network management. The configuration data depends on vendors' devices. Currently, mainstream data models include the Schema model and YANG model.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Complete NETCONF Packet

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">   Message
  <edit-config>                                                          Operations
    <target>
      <running/>
    </target>
    <default-operation>merge</default-operation>
    <error-option>rollback-on-error</error-option>
    <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
      <isiscomm xmlns="http://www.Huawei.com/netconf/vrp/Huawei-isiscomm">
        <isSites>
          <isSite xc:operation="merge">                                  Content
            <instanceId>100</instanceId>
            <description>ISIS</description>
              <vpnName>_public_</vpnName>
            </isSite>
          </isSite>
        </isiscomm>
      </config>
  </edit-config>
</rpc>
```

- This figure shows a complete NETCONF packet.
- Based on the SSH secure transmission, a packet contains information at the messages layer, operations layer, and content layer.
- NETCONF packets are encoded in XML format.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

XML Encoding

- XML is the encoding format of NETCONF. NETCONF uses text files to represent complex hierarchical data.

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Learners</to>
  <from>Huawei</from>
  <heading>Reminder</heading>
  <body>Don't forget Reading!</body>
</note>
```

The header of an XML encoding file is as follows:

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

In the preceding format:

- **<?**: indicates the start of an instruction.
- **xml**: identifies an XML file.
- **version**: indicates the XML version. "1.0" indicates that the XML1.0 standard version is used.
- **encoding**: indicates the character set encoding format. Only UTF-8 encoding is supported.
- **?>**: indicates the end of an instruction.

The XML document forms a tree structure, unfolding from the root.

The <note><to><from><heading><body> tag is private. The XML language does not have predefined tags, allowing users to customize tags and document structures.

Content is nested in the tag format. The slash (/) indicates the end of the current tag.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Transport layer > Messages layer > Operations layer > Content layer

Transport Layer and Messages Layer

- NETCONF uses SSH to implement secure transmission and uses Remote Procedure Calls (RPCs) to implement communication between the client and server.



The RPC framework is independent of the transport layer, and is used to indicate NETCONF requests and responses.

- <rpc>: encapsulates NETCONF requests from the client to the server. The header defines the message-id identification sequence.
- <rpc-reply>: response message sent by the user server to the <rpc>. The values of message-id in the headers are the same.
  - <rpc-error> is sent in <rpc-reply>. One <rpc-reply> element can contain multiple <rpc-error> elements.
  - If <ok> is sent in <rpc-reply>, no error or data is returned.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- For details, see RFC 6241.



Basic Operations of NETCONF

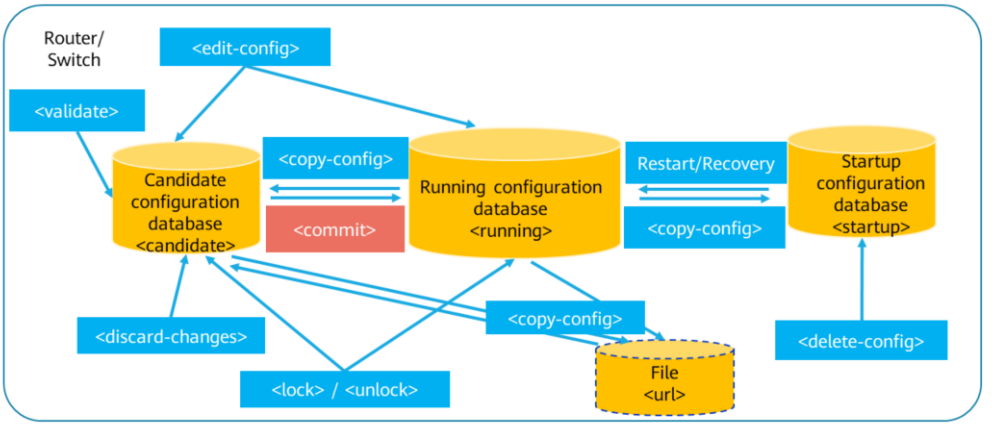
- NETCONF defines a series of operations:

Operation	Scenario Type	Function Description
<get-config>	Querying data	Queries configuration data.
<get>		Queries the current configuration and status data of the device.
<edit-config>	Editing data	Creates, modifies, or deletes configuration data.
<copy-config>	Backup/Recovery	Exports configuration data, or replaces one set of configuration data with another set of configuration data.
<delete-config>		Deletes the configuration data set and clears the startup configuration.
<lock>	Locking/Unlocking	Locks and exclusively occupies the permission to modify the configuration data set.
<unlock>		Unlocks the configuration data set and cancels the exclusive permission to modify the configuration data set.
<commit>	Transaction	Commits configuration data in the <candidate> data set to the current running configuration data.
<cancel-commit>		Cancels the trial run of configuration committing.
<discard-changes>		Discards the uncommitted configuration data in <candidate>.
<validate>		Checks whether the syntax and semantics of the specified configuration data are correct.
<close-session>	Session	Normally ends the NETCONF session.
<kill-session>		Forcibly terminates other NETCONF sessions. You must have the administrator rights to perform this operation.

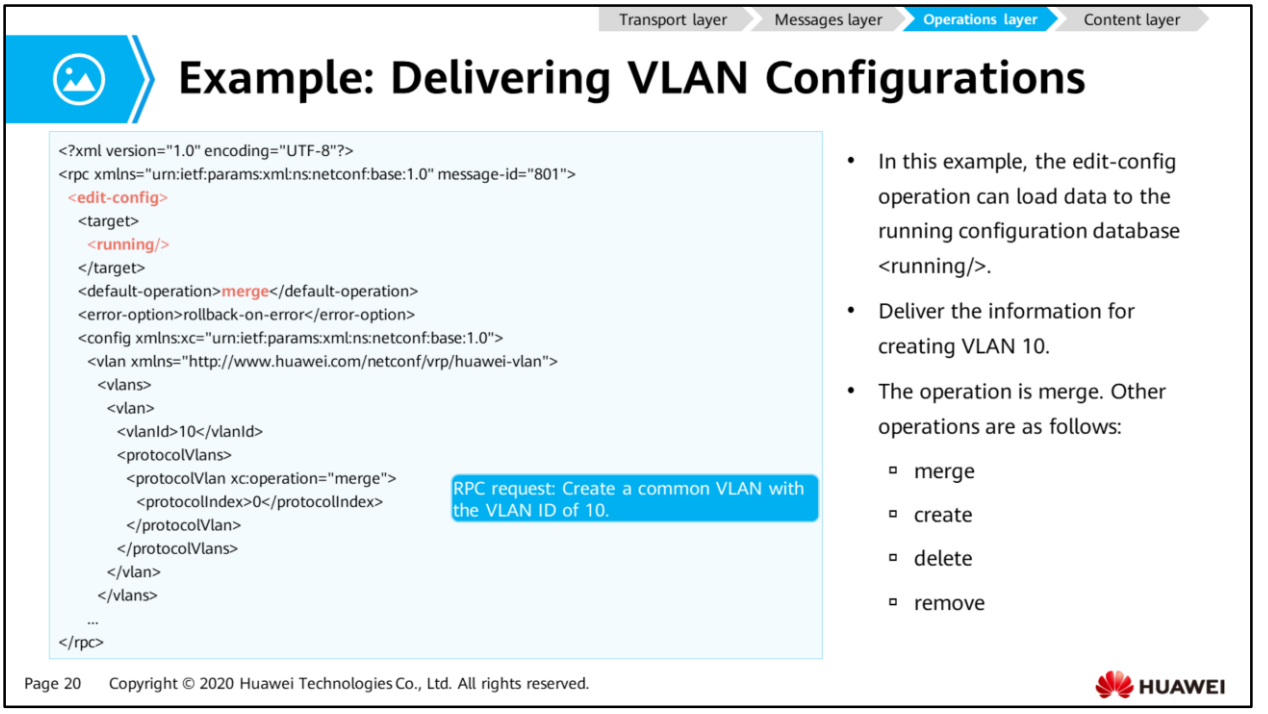
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Operation Objects of NETCONF

- Operation objects have three configuration databases. You can flexibly read and edit the candidate, running, and startup configuration databases to deliver, verify, and roll back the overall configuration.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



- <config> may contain the optional attribute <operation>, which is used to specify an operation type for a configuration item. If the <operation> attribute is not carried, the <merge> operation is performed by default. The <operation> attribute values are as follows:
  - **merge**: In the database, modify the existing data or create data that does not exist. This is the default operation.
  - **create**: Add configuration data to the configuration database only when the configuration data to be created does not exist in the configuration database. If the configuration data exists, <rpc-error> is returned, in which the <error-tag> value is **data-exists**.
  - **delete**: Delete a specified configuration data record from the configuration database. If the data record exists, the data record is deleted. If the data record does not exist, <rpc-error> is returned, in which the <error-tag> value is **data-missing**.
  - **remove**: Delete a specified configuration data record from the configuration database. If the data exists, the data is deleted. If the data does not exist, a success message is returned.

Content Layer of NETCONF

- The NETCONF content layer configures data for devices. Configuration data requires a modeling language, which depends on the implementation of each vendor. NETCONF 1.0 has no requirements on the model language. NETCONF 1.1 clearly defines the combination with YANG and starts to standardize the NETCONF content format.

NETCONF has two modeling languages: Schema and YANG.

- Schema is a set of defined rules for describing XML documents. A schema file defines all the management objects on a managed device, the constraints and hierarchical relationships between these management objects, and the read and write permissions of these management objects.
- YANG is a data modeling language developed to design NETCONF-oriented configuration data, status data models, RPC models, and notification mechanisms.

- Currently, Huawei devices support the following content layers:
  - Huawei-YANG
  - NETCONF Schema
  - IETF-YANG
  - OpenConfig-YANG

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Schema is a language that Huawei extends private syntax based on the W3C XML standard. Before the NETCONF standard is bound to the YANG model, VRPV8 has implemented Schema.
- Huawei-YANG has the most abundant content.

Transport layer > Messages layer > Operations layer > Content layer

Example: Delivering VLAN Configurations (HUAWEI-YANG)

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="801">
 <edit-config>
  <target>
   <running/>
  </target>
  <default-operation>merge</default-operation>
  <error-option>rollback-on-error</error-option>
  <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
   <vlan xmlns="http://www.huawei.com/netconf/vrp/huawei-vlan">
    <vlans>
     <vlan>
      <vlanId>10</vlanId>
      <protocolVlans>
       <protocolVlan xc:operation="merge">
        <protocolIndex>0</protocolIndex>
       </protocolVlan>
      </protocolVlans>
     </vlan>
    </vlans>
    ...
</rpc>
```

RPC request: Create a common VLAN with the VLAN ID of 10.

- In this example, the edit-config operation can load data to the running configuration database <running/>.
- <config></config>: content layer information.
- The namespace of the content layer is http://www.huawei.com/netconf/vrp/huawei-vlan.
- The content layer structure is as follows: <vlans> contains <vlan>, including <vlanId> and <protocalVlans>.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- For details, see the *NETCONF YANG API Reference* released at the official website.

Example: Delivering VLAN Configurations (Schema)

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="801">
 <edit-config>
  <target>
   <running/>
  </target>
  <default-operation>merge</default-operation>
  <error-option>rollback-on-error</error-option>
  <config>
   <vlan xmlns="http://www.huawei.com/netconf/vrp" content-version="1.0" format-version="1.0">
    <vlans>
     <vlan operation="merge">
      <vlanId>10</vlanId>
      <vlanType>common</vlanType>
     </vlan>
    </vlans>
   </vlan>
  </config>
 </edit-config>
</rpc>
```

RPC request: Create a common VLAN with the VLAN ID of 10.

- This example also describes how to create VLAN 10, but the content is in the Schema mode.
- HUAWEI-YANG is recommended, because it has more comprehensive support features.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

NETCONF Configuration

1. Enter the NETCONF view.

   ```
   [Huawei] netconf
   ```

   Run the **netconf** command in the system view to enter the NETCONF view.

2. Configure the port number for NETCONF services.

   ```
   [Huawei-netconf] protocol inbound ssh port 830
   ```

   Enable NETCONF services on port 830 of the SSH server.

3. Enable NETCONF.

   ```
   [Huawei] snetconf server enable
   ```

   Enable the NETCONF services of the SSH server on TCP port 22.

   - Either **snetconf server enable** or **protocol inbound ssh port 830** command can be used to enable NETCONF. If both commands are executed, the client can use port 22 or 830 to establish a NETCONF connection with the server.
   - NETCONF uses SSH as the transport protocol. Therefore, you must configure SSH before using NETCONF.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. Network Management Technology Background
2. NETCONF Protocol
3. **Modeling Language of YANG**
4. RESTCONF Protocol

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

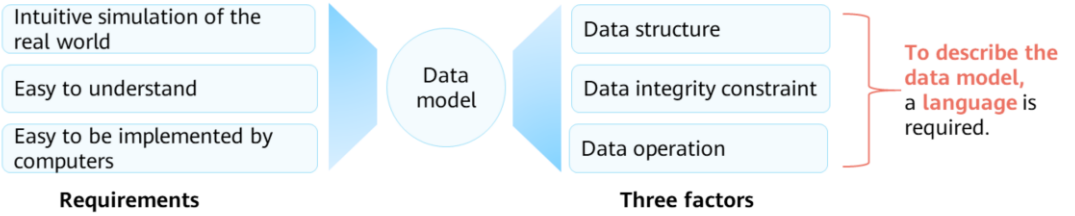
HUAWEI

Origin of YANG

- Although the NETCONF protocol is standardized, data content is not standardized. How can we describe a piece of data?



A data model is an abstraction and expression of data



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



YANG Language Overview

- Yet Another Next Generation (YANG) is a data modeling language.
- The YANG model defines the hierarchical structure of data and can be used for NETCONF-based operations. Modeling objects include configuration, status data, remote procedure calls, and notifications. This allows a complete description of all data exchanged between a NETCONF client and server. YANG has the following features:
  - Hierarchical tree-like structure modeling.
  - Data models are presented as modules and sub-modules.
  - It can be converted to the YANG Independent Notation (YIN) model based on the XML syntax without any loss.
  - Defines built-in data types and extensible types.

The YANG model is presented as a **.yang** file.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- YANG originates from NETCONF but is not only used for NETCONF. Although the YANG modeling language is unified, YANG files are not unified.
- YANG files can be classified into three types: vendor-specific YANG files, IETF-defined YANG files, and OpenConfig YANG files.
- The Config&Status Data, Notification Data, and bottom-layer RPC messages in NETCONF can be modeled using the YANG model. YANG model files can be converted into XML/JSON files using a tool and then encapsulated into NETCONF/RESTCONF messages.
- For details, see RFC 7950.

Introduction to YANG Files – Module

```
module example-system {
    yang-version 1.1;
    namespace "urn:example:system";
    prefix "sys";

    organization "Example Inc.";
    contact "joe@example.com";
    description
     "The module for entities implementing the Example system.";

    revision 2007-06-09 {
     description "Initial revision.";
    }

    container system {
        leaf host-name {...}
        container login{...}
    }
    container X {...}
  }
```

- A YANG file can be defined as a module or submodule. Modules and submodules can reference other modules' model files to use the data types and structures defined by other modules.
- Each module contains multiple declarations, including:
  - YANG version, which is 1.0 or 1.1.
  - YANG namespace, which is IETF YANG or Openconfig YANG.
  - Organization information
  - Contact
  - File description
  - Change history

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- For details, see RFC 7950.

Introduction to YANG Files – Leaf Node

A leaf node is used to define a variable of a simple specified type and is declared using the **leaf** keyword.

- A leaf node has sub-declarations and values but no sub-nodes. In this example, **host-name** contains two sub-declarations: type and description.
- **type** indicates the value type. In this example, the value is **string**.
- **description** indicates the description.

YANG model

```
leaf host-name {
    type string;
    description
      "Hostname for this system.";
  }
```

XML encoding

```
<host-name>my.example.com</host-name>
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- For more information, see RFC 7950.

Introduction to YANG Files – Leaf List

Leaf List is used to define an array variable and is declared using the **leaf-list** keyword.

- In this example, **domain-search** has two sub-declarations: type and description.
- **type** indicates the value type. In this example, the value is **string**.
- **description** indicates the description.

YANG model

```
leaf-list domain-search {
      type string;
      description
        "List of domain names to search.";
    }
```

XML encoding

```
<domain-search>high.example.com</domain-search>
<domain-search>low.example.com</domain-search>
<domain-search>everywhere.example.com</domain-search>
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Introduction to YANG Files – List Node

- A list node is used to define a higher-level data node. Each list node is uniquely identified by a key and can contain multiple leaf nodes.

YANG model

```
list user {
      key "name";
      leaf name {
        type string;
      }
      leaf full-name {
        type string;
      }
      leaf class {
        type string;
      }
    }
```

XML encoding

```
<user>
  <name>glocks</name>
  <full-name>Goldie  Locks</full-name>
  <class>intruder</class>
</user>
<user>
  <name>snowey</name>
  <full-name>Snow  White</full-name>
  <class>free-loader</class>
</user>
<user>
  <name>rzell</name>
  <full-name>Rapun  Zell</full-name>
  <class>tower</class>
</user>
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Introduction to YANG Files – Container Node

YANG model

```
container system {
    container login {
      leaf message {
        type string;
        description
          "Message given at start of login session.";
      }
      list user {
          key "name";
          leaf name {
            type string;
          }
          leaf full-name {
            type string;
          }
          leaf class {
            type string;
          }
        }
}
```

XML encoding

```
<system>
  <login>
    <message>Good  morning</message>
  </login>
</system>
```

- Container nodes are used to define data sets in a larger scope. Each container node has only different sub-nodes, but does not have any value. These sub-nodes can be container, leaf, leaf-list, or list nodes.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Introduction to YANG Files – Grouping

- Grouping is used to define nodes that can be reused. Generally, grouping is used together with **uses**. In this example, **target** defines **leaf address** and **port**. If **use target** is specified in the container peer, the leaf model is reused.

```
grouping target {
  leaf address {
    type inet:ip-address;
    description "Target IP address.";
  }
  leaf port {
    type inet:port-number;
     description "Target port number.";
  }
}
container peer {
  container destination {
    uses target;
  }
}
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Introduction to YANG Files – Configuration Data and Status Data

```
list interface {
      key "name";
      config true;

      leaf name {
        type string;
      }
      leaf speed {
        type enumeration {
          enum 10m;
          enum 100m;
          enum auto;
        }
      }
      leaf observed-speed {
        type uint32;
        config false;
      }
 }
```

YANG files can declare data types. The **config** declaration is used to distinguish configuration data from status data.

- **config true** indicates configuration data.
- **config false** indicates status data.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- For details, see RFC 7950.

Introduction to YANG Files – Data Type

Name	Description
binary	Any binary data
bits	A set of bits or flags
boolean	"true" or "flase"
decimal64	64-bit single decimal number
empty	A leaf that does not have any value
enumeration	One of an enumerated to an abstract identity
instance-identifier	A reference to an abstract identity
int8	8-bits signed integer
int16	16-bit signed integer
int32	32-bit signed integer
int64	64-bit signed integer
leafref	A reference to a leaf instance
string	A character string
uint8	8-bit unsigned integer
uint16	16-bit unsigned integer
uint32	32-bit unsigned integer
uint64	64-bit unsigned integer
union	Choice of member types

- The YANG model supports built-in default data types and extended data types. The **typedef** declaration is used to define extended data types.

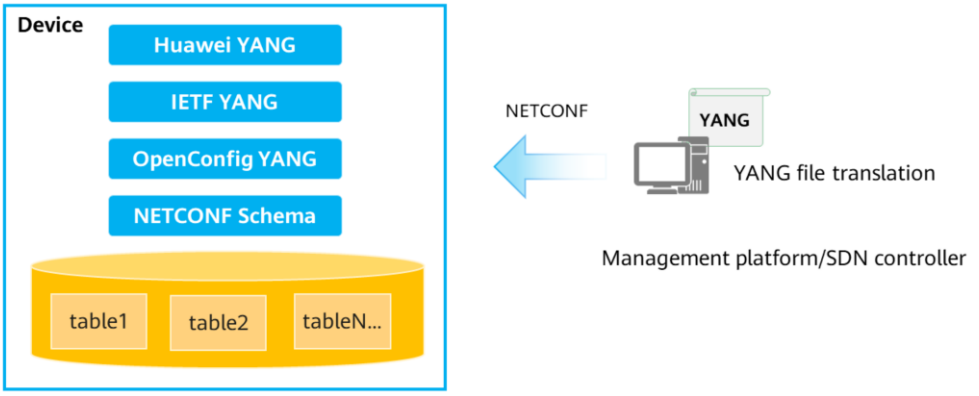
```
typedef percent {
  type uint8 {
    range "0 .. 100";
  }
}

leaf completed {
  type percent;
}
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

FAQ: How Do I Load a YANG Model?

- A YANG file is loaded on the NETCONF client (such as the NMS or SDN controller). A tool is used to convert YANG files into NETCONF messages and deliver the messages to devices.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. Network Management Technology Background
2. NETCONF Protocol
3. Modeling Language of YANG
4. **RESTCONF Protocol**

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

Origin of RESTCONF

Doctoral thesis of Roy Thomas Fielding in 2000

"Architectural Styles and the Design of Network-based Software Architectures"

REST: Representational State Transfer
The state of a resource is transferred in a representation mode on the network.

Resource: data. Unique URI | Representation mode | State transfer: HTTP operation



REST: Client-server interaction framework

In 2017, IETF released RFC 8040.

Internet Engineering Task Force (IETF)
Request for Comments: 8040
Category: Standards Track
ISSN: 2070-1721

A, Bierman
YumaWorks
M. Bjorklund
Tail-f Systems
K.Watsen
Juniper.Networks
January 2017

RESTCONF Protocol

Abstract

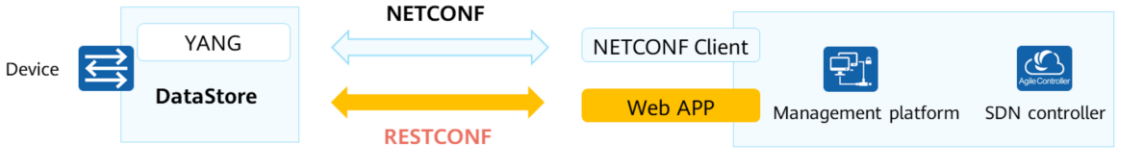
This documents describes an HTTP-based protocol that provides a programmatic interface for accessing data defined in YANG, using the datastore concepts defined in the Network Configurastion Protocol (NETCONF).

REST + Configuration

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

RESTCONF Overview

- RESTCONF allows web applications to access configuration data, status data, and event notifications of network devices in a modular and scalable manner. It has the following features:
  - RESTCONF uses HTTP methods to perform operations (CRUD) on data defined by YANG.
  - YANG files can be shared by NETCONF and RESTCONF.
  - The data encoding format can be XML or JSON.

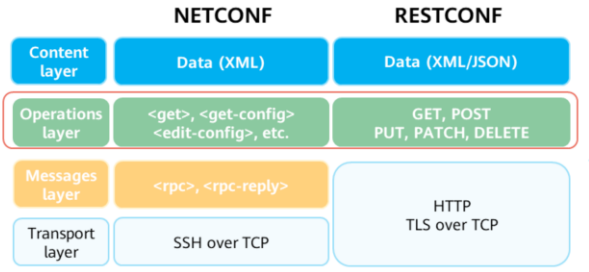


Compared with NETCONF, RESTCONF uses different operation methods and data encoding formats.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- NETCONF and RESTCONF can coexist.
- CRUD: Create, Remove, Update, Delete.

Comparison Between RESTCONF and NETCONF



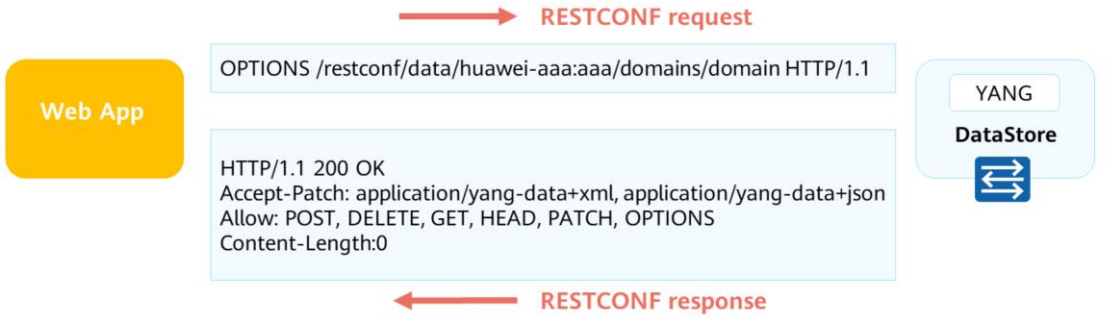
RESTCONF	NETCONF Operation
GET	<get-config>, <get>
POST	<edit-config> (nc:operation="create")
	invoke an RPC operation
PUT	<edit-config> (nc:operation="create/replace")
	<copy-config> (PUT on datastore)
PATCH	<edit-config> (nc:operation depends on PATCH content)
DELETE	<edit-config> (nc:operation="delete")

- NETCONF operates multiple configuration libraries of a device. The transaction mechanism and rollback mechanism are available.
- RESTCONF uses HTTP operations, which are stateless, have no transaction mechanism, and does not support rollback.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Typical RESTCONF Interaction

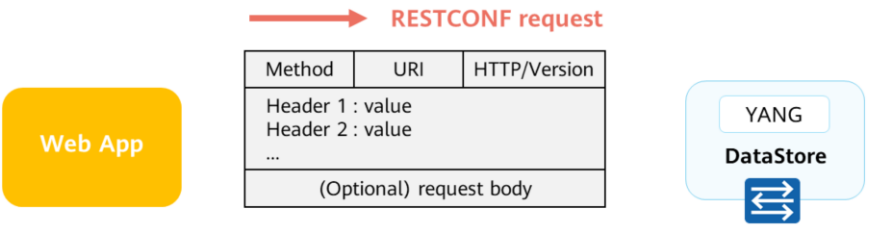
- A complete RESTCONF interaction includes a request and a response.
- In this example, the client uses the OPTIONS method to obtain the operations supported by the device.
- The device response supports the following operations: POST, DELETE, GET, HEAD, PATCH, and OPTIONS.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- NETCONF and RESTCONF can coexist.

RESTCONF Request Packet



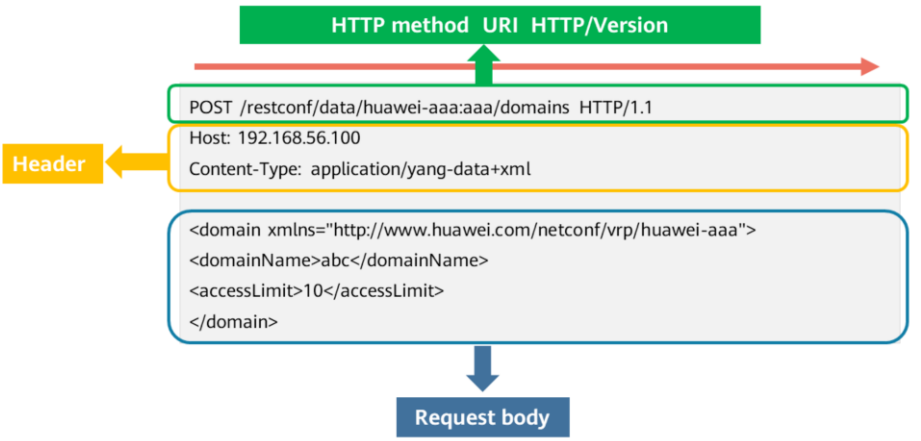
Description	
Method	HTTP method, performed on the resource identified by the request URI.
URI	Uniform Resource Identifier.
HTTP/Version	HTTP version.
Header : value	Header of a request packet, which has specific field requirements. The format is header field and value.
Request body	(Optional) Request body. Some methods do not carry body information.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- A request header may contain multiple fields, such as Accept, Authorization, Host, and From. For details, see RFC 2916.

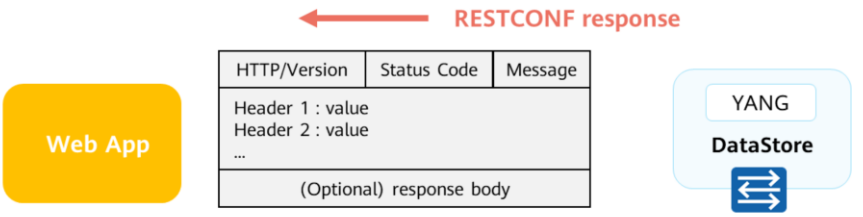
RESTCONF Request Example

- The IP address of the RESTCONF request object is 192.168.56.100. Change the value of **domainName** to **abc** and the value of **accessLimit** to **10**.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

RESTCONF Response Packet



Description	
HTTP/Version	HTTP method, performed on the resource identified by the request URI.
Status code	HTTP status code.
Message	HTTP status message.
Header : value	Header of a response packet, which has specific field requirements. The format is header field and value.
Response body	(Optional) Response body. Some methods do not carry body information.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Header information contains details about Response Header and Entity Header. For details, see sections 6.2 and 7.1 in RFC 2916.



HTTP Status Code

- An HTTP status code is a three-digit number indicating the status of a response from the server. It is used to return the operation result to the client.

Status Code		Description	
1XX	(Informational) The request is received.	100 Continue	If the request is received, go to the next step.
2XX	(Successful) The request is successful.	200 OK	Success. The response body is available.
		201 Created	A resource is successfully created, and the URI of the newly created resource exists in the location field.
		204 No Content	Success. The response body is unavailable.
3XX	(Redirection) Further operations need to be performed.	301 Moved Permanently	A new URI is allocated to the target resource, and all future resources will be associated with the new URI.
4XX	(Client Error) Request error.	400 Bad Request	The request body is incorrect and carries error information.
		401 Unauthorized	Authorization failed. For example, the certificate does not match.
		403 Forbidden	Access denied. The possible cause is that the user attempts to perform operations beyond the permission or the login user name or password is incorrect.
		404 Not Found	The requested resource cannot be found.
5XX	(Server Error) Server error.	500 Internal Server Error	The request cannot be executed due to an internal server error. The user needs to resend the request later.
		501 Not Implemented	The function has not been implemented.

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

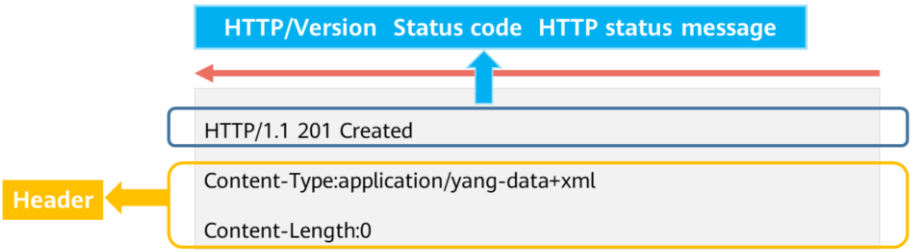
HUAWEI

- For details, see 6 Response Status Code in RFC 7231.

RESTCONF Response Example

RESTCONF response packet

- If status code 201 is returned, the resource is successfully created.
- **Content-Type** and **Content-Length** in the header describe body information. The body data type is XML, and the content length is 0.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Quiz

1. Which protocol is used at the transport layer of Huawei NETCONF?
2. What is the relationship between NETCONF, RESTCONF, and YANG?

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

1. Huawei uses SSH as the transport layer protocol. Before enabling the NETCONF function on a device, you need to create an SSH user as the NETCONF user for login.
2. YANG is a modeling language used to describe the content layer of NETCONF and RESTCONF. The difference between NETCONF and RESTCONF is as follows: RESTCONF constructs the transport layer, messages layer, and operations layer based on HTTP, while NETCONF has defined the operations layer and uses SSH as the transport layer and RPC as the messages layer.

Summary

- NETCONF provides a network device management mechanism. You can use NETCONF to add, modify, or delete configurations of network devices, and obtain configurations and status of network devices.
- The YANG model defines the hierarchical structure of data and can be used for NETCONF-based operations.
- RESTCONF uses HTTP methods to perform operations (CRUD) on data defined by YANG.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

More Information

- For details about NETCONF, see RFC 6241.
- For details about YANG, see RFC 7950.
- For details about RESTCONF, see RFC 8040.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Recommendations

- https://e.huawei.com/cn/talent/#/home

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.


Thank You
www.huawei.com



Telemetry Fundamentals and Practices

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Foreword

- Telemetry is a network device monitoring technology that periodically samples statistics and status data on network devices. This course describes the technical background, concepts, framework, and technical fundamentals of telemetry, as well as the configuration procedure of static telemetry subscription.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Objectives

- On completion of this course, you will be able to:
  - Describe the background of the telemetry technology.
  - Describe the telemetry concepts and framework.
  - Distinguish static and dynamic telemetry subscriptions.
  - Describe the concepts and functions of gRPC.
  - Configure telemetry data sampling.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

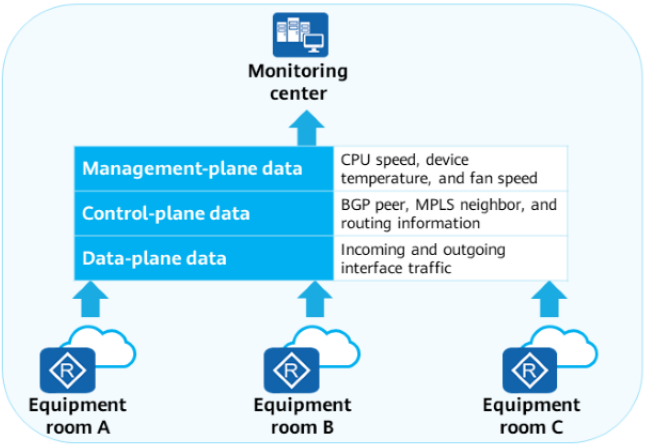
Contents

1. **Introduction to Telemetry**
   - **Technical Background**
   - Overview
2. Telemetry Technical Fundamentals
3. Telemetry Configuration and Practice

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Technical Background: Introduction to Network Device Monitoring

- An O&M platform mainly provides unified monitoring and performance management for network devices. It monitors device data on the data, control, and management planes.



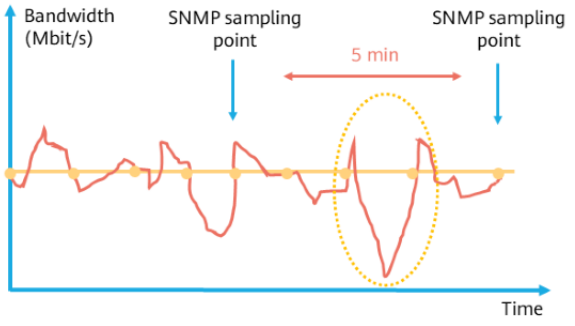
- Device monitoring data can be obtained through SNMP, CLI, Syslog, NetStream, and sFlow.
- NetSteam and sFlow deal with network traffic monitoring and are mainly applicable to data-plane data.
- SNMP is the most popular protocol for device monitoring.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Technical Background: Bottlenecks of Traditional Data Sampling Technologies (1/2)

- The Nyquist Sampling Theorem states that a signal must be sampled at more than twice the highest frequency component of the signal to ensure that no information will be lost. Data loss may occur if the data is sampled through SNMP every 5 minutes.

Long sampling period



- The red wavy line represents a normal interface bandwidth change trend. Traffic exceptions cannot be detected through periodic SNMP-based data collection.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Technical Background: Bottlenecks of Traditional Data Sampling Technologies (2/2)

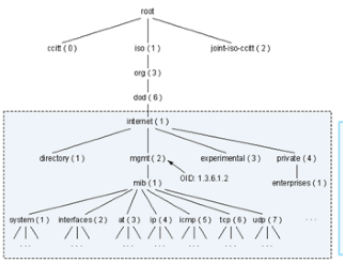
- Traditional SNMP-based data collection uses the pull mode, in which one request and one response are involved in each interaction. To collect sampled data, the network management system (NMS) needs to send a query request to the device. After parsing the request, the device sends the sampled data to the NMS. Each vendor customizes its Management Information Base (MIB), resulting in poor compatibility.

Low collection efficiency



- Traditional data collection uses the pull mode.

Poor data model compatibility



- SNMP data is based on MIBs which are defined by vendors separately.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- MIB is a standard for network management data.

Summary: Traditional Network O&M Faces Many Challenges

- On a traditional network, data is sampled in pull mode at an interval of 5 to 15 minutes on average. Data sampling in pull mode at a shorter interval will result in network breakdown.

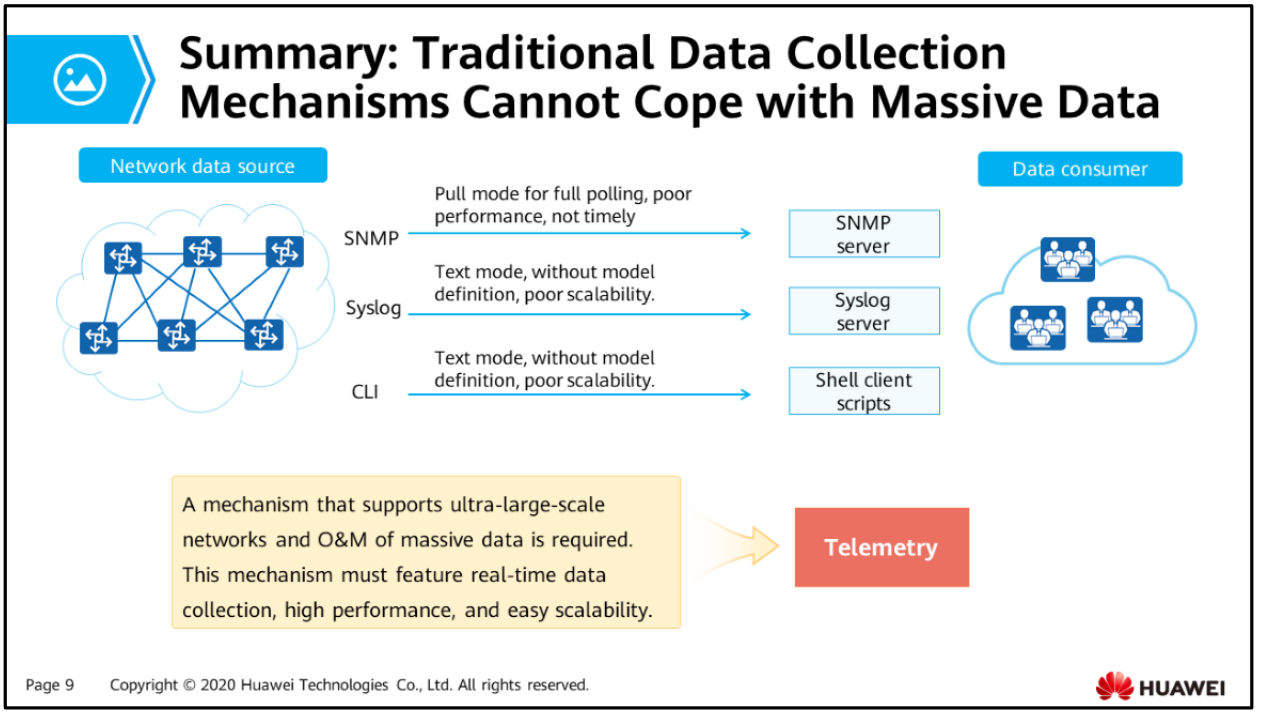


Problems in traditional network O&M

- The large sampling interval results in low network visibility.
- SNMP-based O&M systems are inefficient.
- Real-time end-to-end monitoring, like IT O&M, cannot be achieved, and complete historical data cannot be retained.
- Network problems caused by a large number of microbursts cannot be detected.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- A microburst refers to a situation in which a large amount of burst data is received within a very short time (milliseconds), so that the burst data rate is tens or hundreds times higher than the average rate or even exceeds the port bandwidth. The NMS or network performance monitoring software calculates the real-time network bandwidth at an interval of seconds to minutes. At such an interval, the network traffic seems to be stable. However, packet loss may have occurred due to microbursts.



- SNMP queries are performed in a question-answer manner. If 1000 interactions are performed within 1 minute, SNMP parses 1000 query request packets. Telemetry avoids repeated queries. This is because subscription needs to be performed only once and then devices can continuously push data to the NMS.

Contents

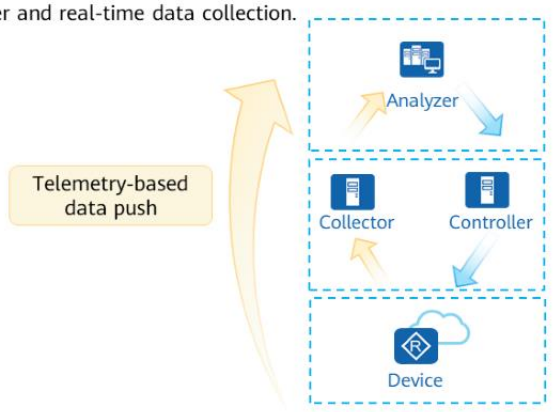
1. **Introduction to Telemetry**
   - Technical Background
   - **Overview**
2. Telemetry Technical Fundamentals
3. Telemetry Configuration and Practice

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

Telemetry Overview

- Telemetry, also called network telemetry, is a technology that remotely collects data from physical or virtual devices at a high speed. Devices periodically send interface traffic statistics, CPU usage, and memory usage to collectors in push mode. Compared with the traditional pull mode (question-answer interaction), the push mode provides faster and real-time data collection.

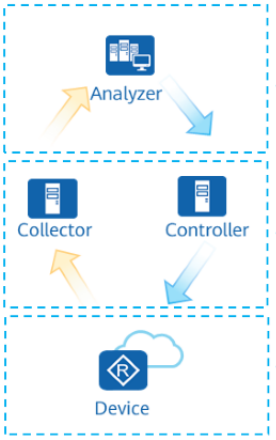


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- There is also a view in the industry that SNMP is considered as a traditional telemetry technology, and telemetry is currently referred to as streaming telemetry or model-driven telemetry.
- Telemetry packs the data to be sent, improving transmission efficiency.

Telemetry Technical Features

- Telemetry works in push mode. YANG models define the telemetry data structure, and data sampling can be performed at an interval of sub-seconds.



Open APIs are available to third-party data analysis and processing companies, providing stronger data storage and processing capabilities.

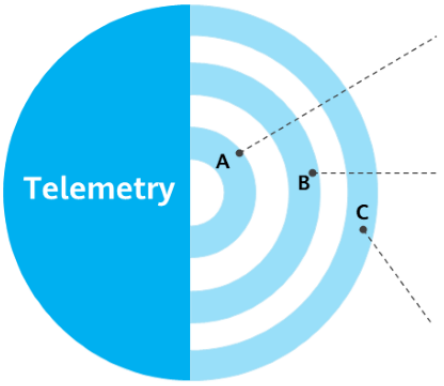
The traditional NMS is decoupled into a collector and a controller. In addition, communication protocols and management applications are decoupled.

- Improves devices' data collection capabilities.
- Supports the subscription mechanism.
- Constructs standard data models.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Telemetry Advantages

- Telemetry obtains abundant monitoring data in push mode in a timely manner. The data helps quickly locate network faults. In addition, telemetry provides a uniform data stream format, which simplifies data collection and analysis to solve problems in traditional network O&M.



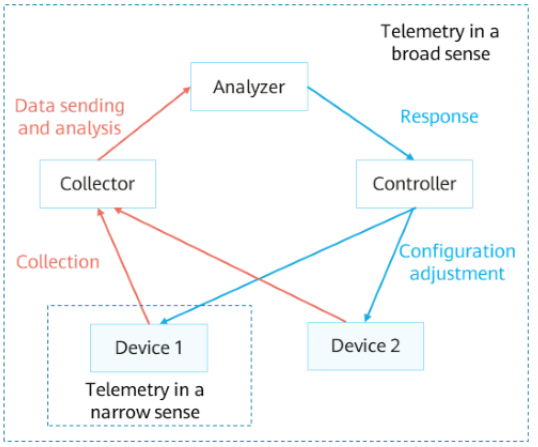
Refined monitoring
The collected data is of high precision and various types, which can fully reflect the network status.

Fast fault locating
Users want faults to be quickly located in seconds or even sub-seconds on a complex network.

Proactive data reporting
For telemetry, subscription needs to be performed only once. Then devices can continuously report data, reducing the pressure on devices to process query requests.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Telemetry Network Model



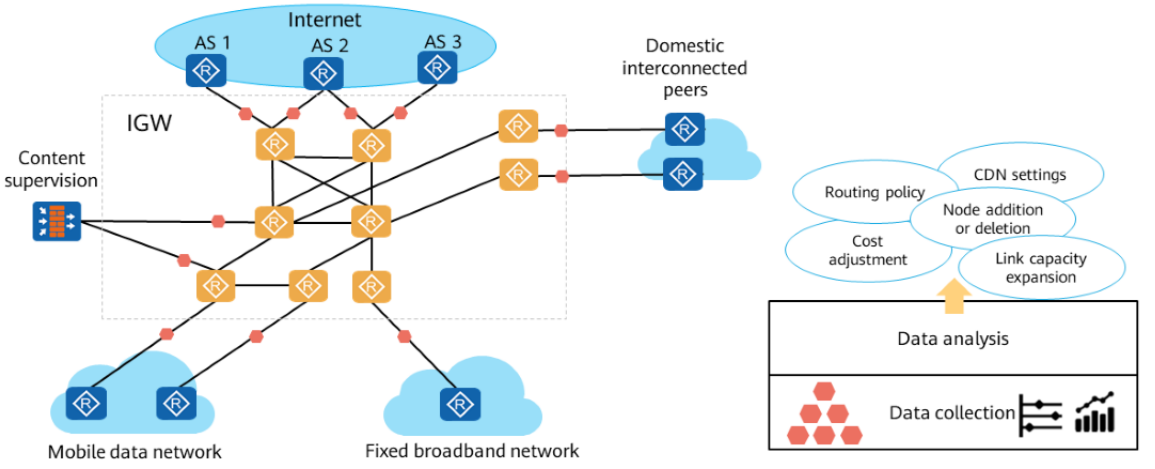
- The telemetry network model is defined in two ways:
  - In a broad sense, telemetry is a self-closed-loop system consisting of network devices, a collector, an analyzer, and a controller.
  - In a narrow sense, telemetry is the function of sending sampled device data to a collector.

Using telemetry, a collector can collect a large amount of device data and send the data to an analyzer for comprehensive analysis. The analyzer sends the results to a controller, which then adjusts device configurations accordingly, enabling the analyzer to determine whether the device status after adjustment meets expectations in real time.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Telemetry Application Scenario: WAN

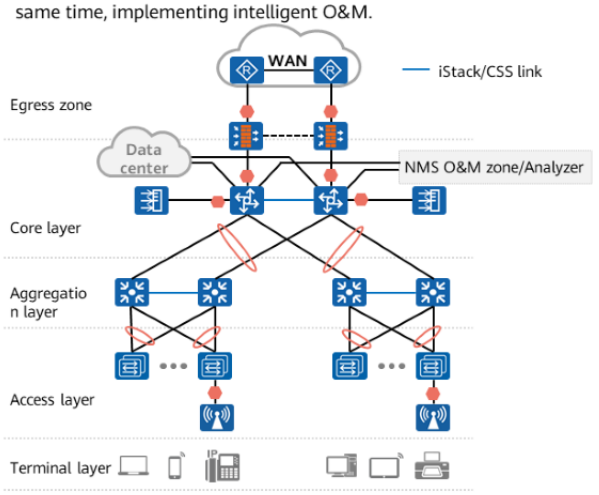
- Interface data of routers is reported in real time to help you gain insight into the traffic direction.



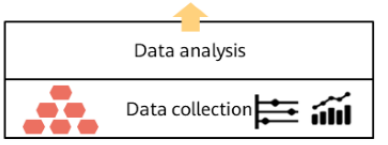
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Telemetry Application Scenario: Campus Network

- On a campus network, telemetry can be used to monitor and report device status and wired and wireless data to the analyzer at the same time, implementing intelligent O&M.



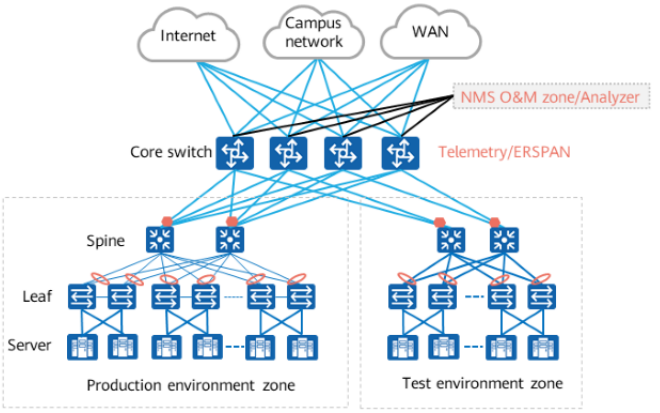
Measurement Object	Metric
AP	CPU usage, memory usage, and number of online clients
Radio	Number of online users, channel usage, noise, traffic, backpressure queue, interference rate, and power
User	RSSI, negotiated rate, packet loss rate, and latency
Device/Board	CPU and memory usages
Port	Numbers of sent and received broadcast, multicast, and unicast packets, numbers of discarded received and sent packets, and numbers of received and sent error packets



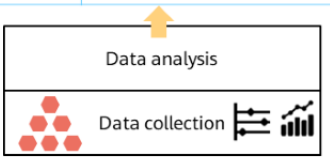
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Telemetry Application Scenario: Data Center

- Upon the receipt of data sent through telemetry, the analyzer calculates the forwarding path, forwarding latency, and link latency of packets. In addition, it analyzes the application interaction relationship and associates applications with network paths. The analyzer can collect statistics, analyze data, and display analysis results, implementing intelligent O&M.



Measurement Object	Metric
Device	CPU and memory usages
Board	CPU, memory, FIB entry, and MAC entry usages
Chip	TCAM usage
Port	Number of received/sent packets, number of bytes, number of lost packets, number of error packets, and numbers of broadcast, multicast, and unicast packets
Queue	Buffer size
Optical module	Receive/Transmit optical power, current, voltage, and temperature
Packet loss behavior	Detection of packet loss due to congestions



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- The collector in the data center collects device performance data through telemetry and collects device flow mirroring data through ERSPAN.

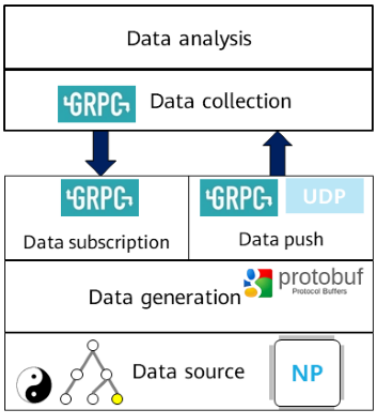
Contents

1. Introduction to Telemetry
2. **Telemetry Technical Fundamentals**
3. Telemetry Configuration and Practice

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Telemetry Framework

- The framework of telemetry in a narrow sense consists of four modules: data source, data generation, data subscription, and data push.

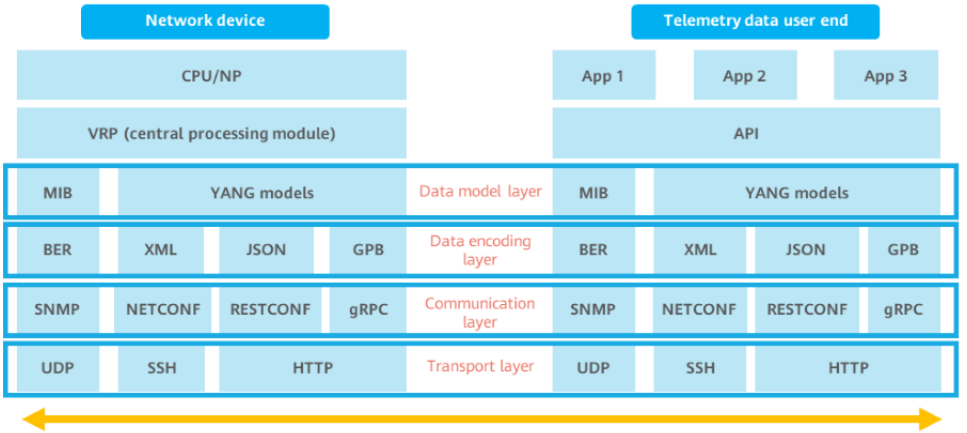


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- For details about the framework, see the corresponding RFC draft at https://tools.ietf.org/html/draft-song-ntf-02.
- Google Remote Procedure Call (gRPC) is an open-source remote procedure call (RPC) system developed by Google.
- User Datagram Protocol (UDP) provides a method for an application to send encapsulated IP packets without establishing a connection.
- Protocol buffers (Protobuf) is a mechanism for serializing structured data.

Telemetry Protocol Stack

- The telemetry protocol stack is divided into the transport layer, communication layer, data encoding layer, and data model layer.

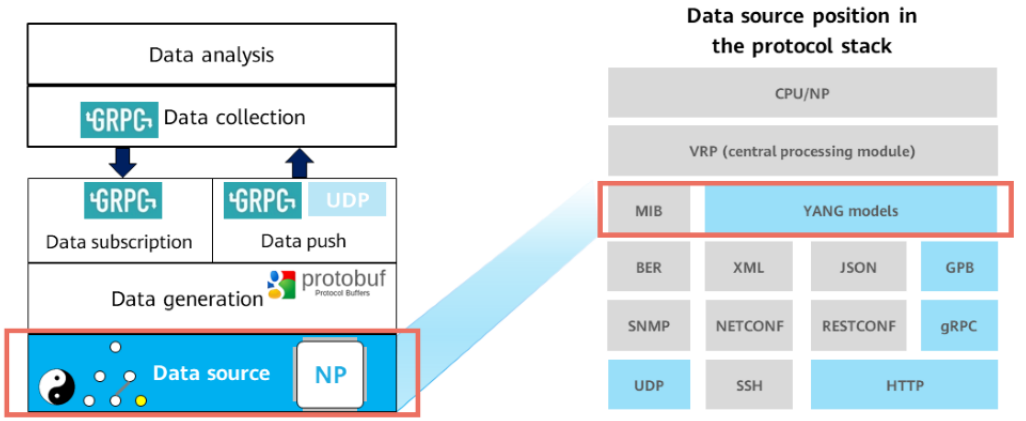


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- A YANG model is similar to a menu for a fast-food restaurant. If a customer wants to eat a hamburger or fried chicken, the customer writes an A4 paper purchase list with one hamburger and two fried chickens according to the menu, folds the list into a stamp-sized note, puts it in the GPB envelope, and gives the envelope to the messenger gRPC at the door. The gRPC then rides on an HTTP/2 electric motorcycle and goes to the fast food restaurant. The messenger gRPC gives the GPB envelope to the restaurant boss, and the restaurant manager opens the envelope to check whether the items that the customer orders are on the restaurant menu.

Telemetry Data Source

- The telemetry data source defines the data that can be obtained. Telemetry uses various types of YANG models to define data sources available to devices, such as Huawei-YANG, IETF-YANG, and OpenConfig-YANG.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

YANG Model and Sampling Path

Top node in the YANG model

Top container in the YANG model

Level-2 container in the YANG model

Node to be sampled

```
module huawei-debug {
  namespace "urn:huawei:yang:huawei-debug";
  prefix debug;
  import huawei-extension {
    prefix ext;
  }
  container debug {
    description
      "Debug information.";
    container cpu-infos {
      description
        "CPU information of boards.";
      list cpu-info {
        ext:entry-from "system";
        key "position";
        description
          "CPU information of a board.";
        leaf position {
          type string {
            length "1..32";
          }
          description
            "Board position.";
        }
        leaf overload-threshold {
          type uint32 {
            range "4..100";
          }
          units %;
          default "90";
          description
            "Overload threshold.";
        }
```

The preceding information forms a sampling path:

huawei-debug:debug/cpu-infos/cpu-info

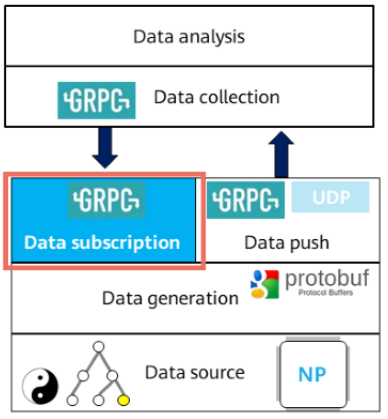
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- For more information about the YANG model, visit https://datatracker.ietf.org/doc/rfc7895/.

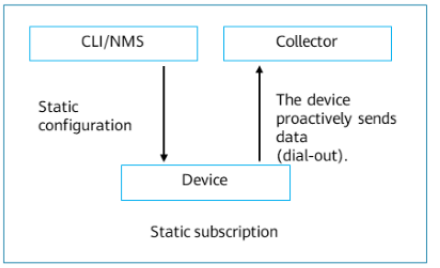
Telemetry Data Subscription

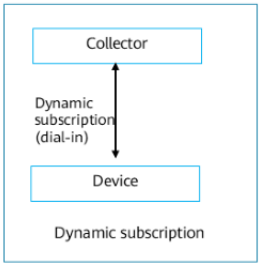
- Telemetry data subscription defines the data receive end and transmit end, as well as their interactions.



Huawei provides two subscription modes for telemetry: static and dynamic.

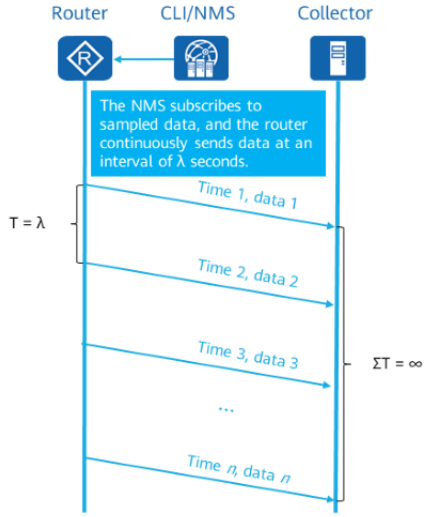
- In static telemetry subscription, a device functions as a client and a collector functions as the server. The device proactively initiates a connection to the collector to collect and send data. This mode is applicable to long-term inspection.
- In dynamic telemetry subscription, a collector functions as a client and initiates a connection to a device functioning as a server to collect data. This mode is applicable to short-term monitoring.





Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Static Telemetry Subscription

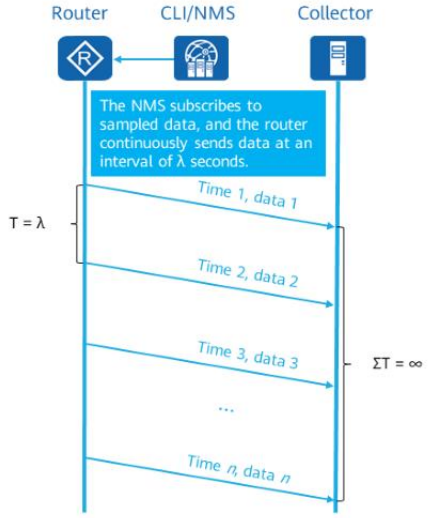


Static Telemetry Subscription

1. The NMS delivers static subscription configurations to the device, including the sampling path, sampling interval, destination to which the sampled data is reported, and data. The configurations can be delivered in the following modes:
   - Through the CLI
   - Through the NETCONF model **openconfig-telemetry.yang**
2. The device starts to sample data, establishes a connection with the collector, and pushes the sampled data to the collector in the following modes:
   - Through gRPC using the RPC method **dataPublish**
   - Through UDP

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

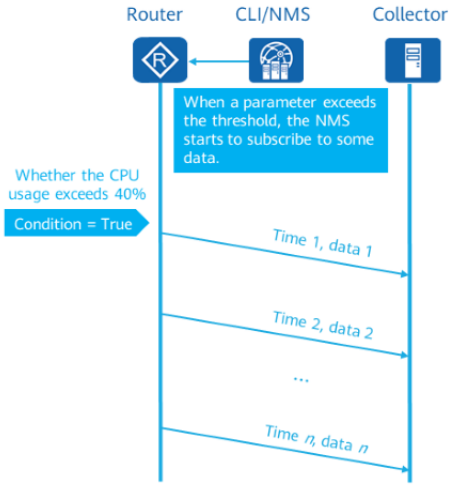
Static Telemetry Subscription



Configure the collector.	system-view Telemetry destination-group ipv4-address IP address, port number, protocol, and encryption mode for the collector
Configure the data to be sampled.	system-view telemetry sensor-group sensor-path Sensor sampling path
Create a subscription.	system-view telemetry subscription sensor-group Associate the sampling sensor group, and configure a sampling period, redundancy suppression, and a heartbeat interval.
Verify the configuration	display telemetry sensor display telemetry destination display telemetry subscription display telemetry sensor-path

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Customized Event Reporting

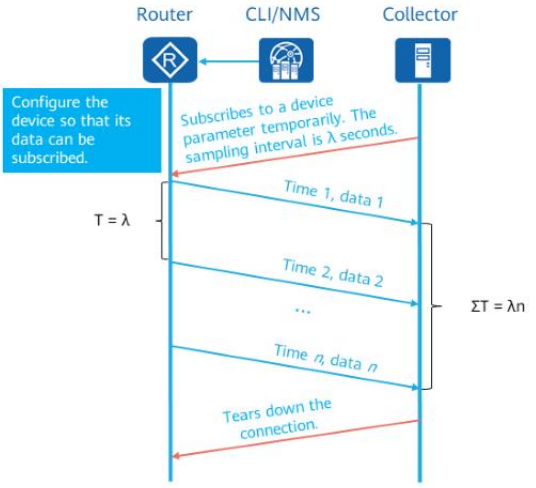


After a specific parameter exceeds the threshold, static subscription is triggered.

Configure a customized event in the static subscription.	system-view telemetry sensor-group Sensor-path //Configures a customized event. Example: sensor-path huawei-devm:devm/cpuInfos/cpuInfo condition express op-field systemCpuUsage op-type gt op-value 40

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Dynamic Telemetry Subscription Using gRPC



Dynamic Telemetry Subscription

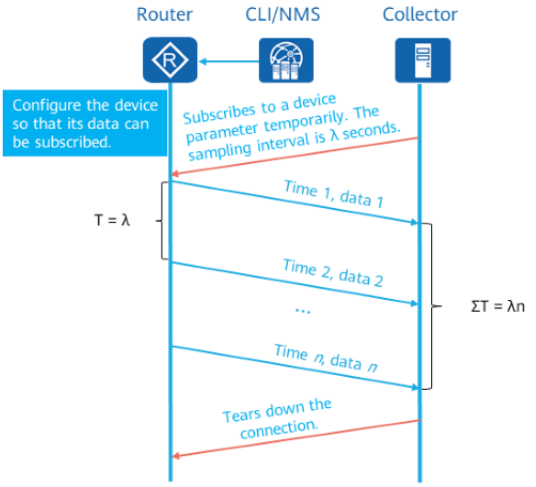
1. The collector initiates a gRPC connection request to the router.
2. The collector initiates a dynamic subscription request to the router.
   - RPC method: Subscribe
   - Specify the sampling path and sampling interval.
3. The router starts to sample data, sends the sampled data (as the response to the dynamic subscription request) to the collector.

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- gRPC is a RPC system developed by Google.

Dynamic Telemetry Subscription Using gRPC

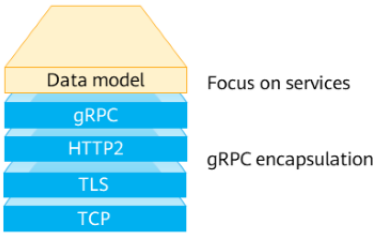


Parameter-specific data collection and on-demand data push	
Configure the collector.	system-view grpc grpc server // Enters the gRPC server view. source-ip // Specifies a source IP address to be listened on. server-port // Specifies the number of a port to be listened on. service enable // Enables the gRPC service. (Optional) acl (Optional) idle-timeout (Optional) ssl-policy (Optional) ssl-verify peer commit
Verify the configuration.	display telemetry dynamic-subscription

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

gRPC

- gRPC, developed by Google, is a language-neutral, platform-neutral, and open-source remote procedure call (RPC) system. It can be used to perform secondary development for both communication parties (NMS and devices). This development focuses on services and shortens interconnection development periods.
- gRPC supports the following languages: C++, Node.js, Python, Ruby, Objective-C, PHP, C#, Java, and Go.
- gRPC is based on HTTP/2, which is better than HTTP/1 in performance. HTTP/2 features include bidirectional streaming, flow control, header compression, and multiplexing request of a single connection.



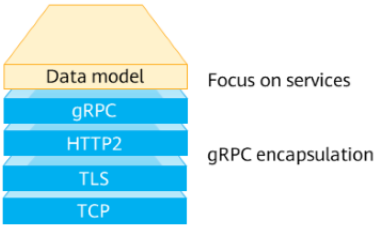
 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- For more information about gRPC, visit https://doc.oschina.net/grpc?t=58009.

gRPC Protocol Stack

- The gRPC protocol stack consists of the following layers:
  - TCP layer: This layer provides connection-oriented and reliable data links.
  - Transport Layer Security (TLS) layer: This layer ensures secure communication between devices and collectors using the TLS protocol, which is optional.
  - HTTP/2 application layer: gRPC is based on the HTTP/2 protocol. To be specific, gRPC uses HTTP/2 features such as bidirectional streams, flow control, and multiplexing.
  - gRPC layer: This layer defines the protocol interaction format for RPCs. Public RPC methods are defined in common .proto files, for example, **huawei-grpc-dialout.proto**.
  - Data model layer: This layer carries encoded service data. The data can be encoded in Google Protocol Buffers (GPB), XML, and JSON.



 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- For more information about gRPC, visit https://doc.oschina.net/grpc?t=58009.



gRPC Service Methods

- gRPC defines four service methods (distinguished by streams).
  - Unary RPCs: The client sends a single request to the server and gets a single response back.

    Example: rpc Cancel(CancelArgs) returns(CancelReply) {};
  - Server streaming RPCs: The client sends a request to the server, and the server continuously sends a stream back.

    Example: rpc Subscribe(SubsArgs) returns(stream SubsReply) {};
  - Client streaming RPCs: The client continuously sends a stream to the server and waits for a response from the server.

    Example: rpc LotsOfGreetings(stream HelloRequest) returns (HelloResponse) {};
  - Bidirectional streaming RPCs: Both the client and server send a stream.

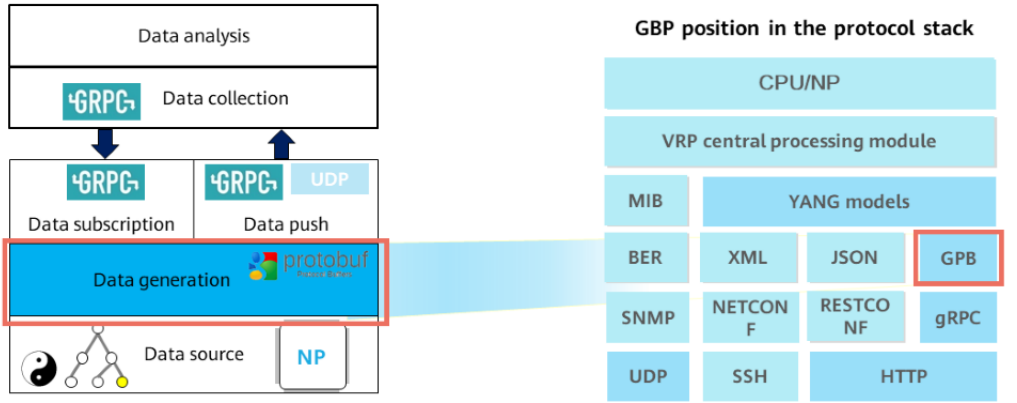
    Example: rpc dataPublish(stream serviceArgs) returns(stream serviceArgs) {};

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- For more information about gRPC, visit https://doc.oschina.net/grpc?t=58009.

Telemetry Data Generation

- With telemetry, data is encoded using GPB, which provides a mechanism for serializing structured data flexibly, efficiently, and automatically. Similar to XML and JSON, GPB is a binary encoding mode with better performance.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

GPB Encoding

- GPB uses a .proto file to describe the message format and uses digits to replace flag names (keyword names).

Before GPB Encoding

```
{
 1:"HUAWEI"
 2:"s4"
 3:"huawei-ifm:ifm/interfaces/interface"
 4:46
 5:1515727243419
 6:1515727243514
 7{
  1[{
   1: 1515727243419
   2 {
    5{
     1[{
      5:1
      16:2
      25:"Eth-Trunk1"
     }]
    }
   }
  }]
 }
}
```

huawei-telemetry.proto

```
syntax = "proto3";
package telemetry;

message Telemetry {
  string node_id_str = 1;
  string   subscription_id_str = 2;
  string   sensor_path = 3;
  string   proto_path = 13;
  uint64   collection_id = 4;
  uint64   collection_start_time = 5;
  uint64   msg_timestamp = 6;
  TelemetryGPBTable data_gpb = 7;
  uint64 collection_end_time = 8;
  uint32   current_period = 9;
  string   except_desc = 10;
  string   product_name = 11;
  Encoding   encoding =12;
  string   data_str = 14;

}
```

After GPB Decoding

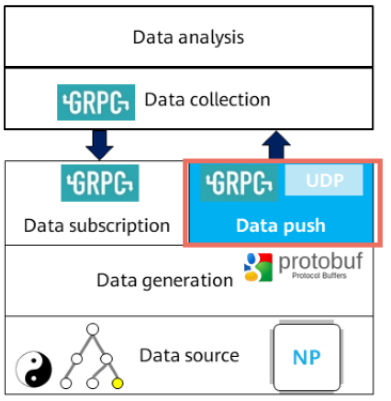
```
{
 "node_id_str":"HUAWEI",
 "subscription_id_str":"s4",
 "sensor_path":"huawei-ifm:ifm/interfaces/interface",
 "collection_id":46,
 "collection_start_time":"2018/1/12 11:20:43.419",
 "msg_timestamp":"2018/1/12 11:20:43.514",
 "data_gpb":{
  "row":[{
   "timestamp":"2018/1/12 11:20:43.419",
   "content":{
    "interfaces":{
     "interface":[{
      "ifAdminStatus":1,
      "ifIndex":2,
      "ifName":"Eth-Trunk1"
     }]
    }
   }
  }]
 }
}
```

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

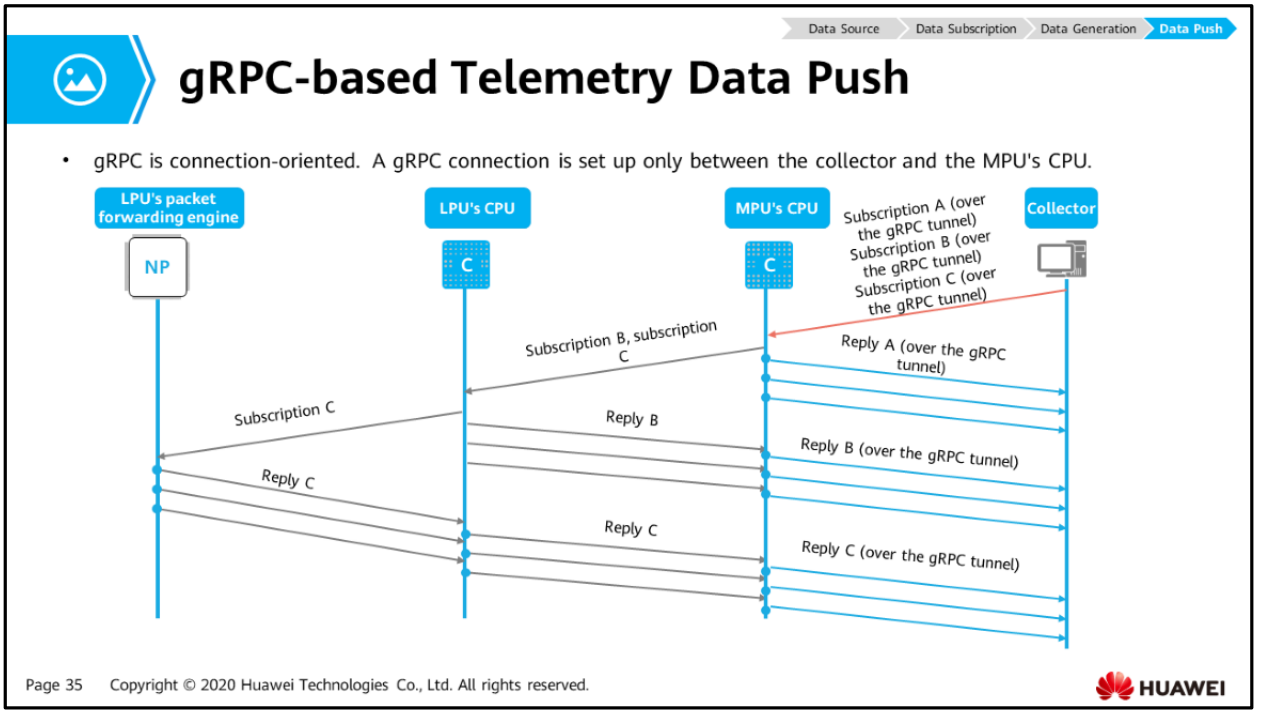
- GPB transmits data in binary mode with a small number of bytes for each transmission, and therefore stands out from other encoding methods, such as XML and JSON, in terms of transmission efficiency. Data collection efficiency is a key concern of Telemetry.
- For more information, see https://developers.google.com/protocol-buffers/.

Telemetry Data Push

- Telemetry data can be pushed in gRPC or UDP mode.



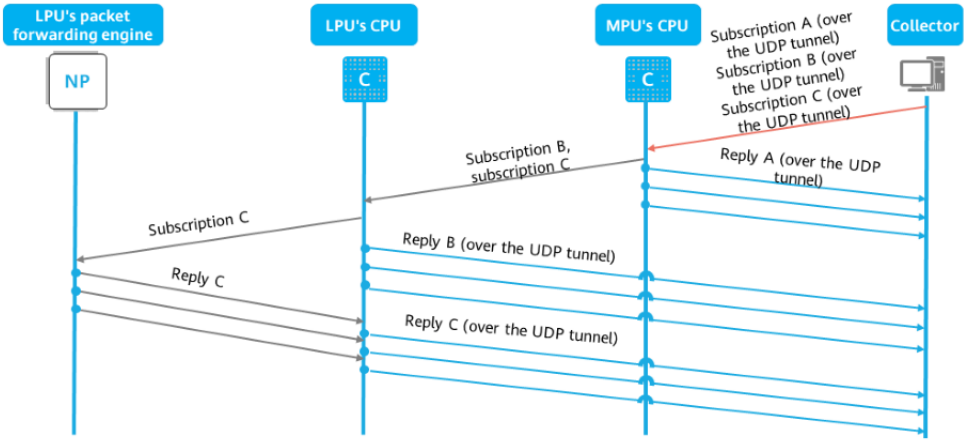
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



1. The collector functions as the gRPC client, and the device functions as the gRPC server.
2. The collector constructs data in GPB or JSON format based on the subscribed event, compiles a .proto file through Protocol Buffers, establishes a gRPC channel with the device, and sends a request message to the device using gRPC.
3. After receiving the request, the device parses the .proto file using Protocol Buffers to restore the data for processing.
4. After data sorting is complete, the device re-compiles the data using Protocol Buffers and sends a response to the collector using gRPC.
5. The collector receives the response message. So far, the gRPC interaction ends.

UDP-based Telemetry Data Push

- Compared with gRPC, data push based on UDP is optimized. In addition to the UDP tunnel with the MPU's CPU, the collector establishes a UDP tunnel with the LPU's CPU. This reduces the pressure on the MPU's CPU and improves efficiency.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Format of Pushed Data (Based on gRPC)

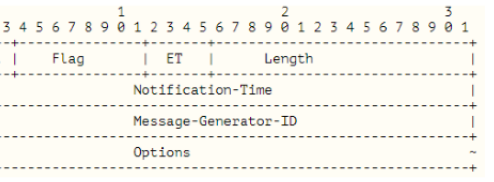
- The gRPC encapsulation layer is provided by the gRPC open-source software. The following figure shows the protocol stack for pushing data based on gRPC.

Hierarchy		Description
TCP layer		Underlying communication protocol, which is based on TCP connections.
TLS layer		This layer is optional. It is based on the TLS 1.2–encrypted channel and bidirectional certificate authentication.
HTTP/2 layer		gRPC is based on HTTP/2, which is better than HTTP/1 in performance. HTTP/2 features include bidirectional streaming, flow control, header compression, and multiplexing request of a single connection.
gRPC layer		Defines the protocol interaction format for RPCs.
Data model layer	RPC layer	The device proactively initiates an RPC request to the collector, which is defined in the **huawei-grpc-dialout.proto** file. A collector proactively initiates an RPC request for dynamic subscription to a device, which is defined in the **huawei-grpc-dialin.proto** file.
	Telemetry layer	This is defined in the **huawei-telemetry.proto** file.
	Service data layer	Service data at this layer needs to be encoded by the .proto file of the corresponding service, and the NMS needs to decode the data using the .proto file of the corresponding service as well. For example, if the sampling path is **huawei-debug:debug/cpu-infos/cpu-info**, the **huawei-debug.proto** file needs to be used for data encoding and decoding.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Format of Pushed Data (Based on UDP)

- The format of the data pushed based on UDP is as follows.

Hierarchy		Description
UDP layer		Underlying communication protocol, which is connectionless based on UDP.
DTLS layer		This layer is optional. It is based on the TLS 1.2–encrypted channel and bidirectional certificate authentication.
Message header layer		This layer describes the message header format of pushed telemetry data. The IETF definition is available at https://datatracker.ietf.org/doc/draft-ietf-netconf-udp-pub-channel/01/?include_text=1 The draft has six versions ranging from 00 to 05. Currently, telemetry is implemented based on the 01 version, as shown in the following figure. 
Notification message layer	Telemetry layer	This is defined in the **huawei-telemetry.proto** file.
	Service data layer	Service data at this layer needs to be encoded by the .proto file of the corresponding service, and the NMS needs to decode the data using the .proto file of the corresponding service as well. For example, if the sampling path is **huawei-debug:debug/cpu-infos/cpu-info**, the **huawei-debug.proto** file needs to be used for data encoding and decoding.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

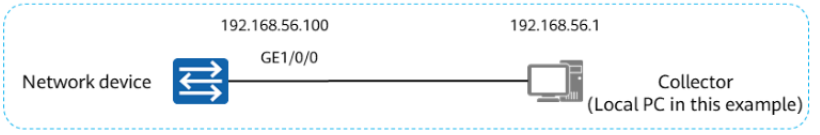
1. Introduction to Telemetry
2. Telemetry Technical Fundamentals
3. **Telemetry Configuration and Practice**

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Case: Configuring Static Telemetry Subscription

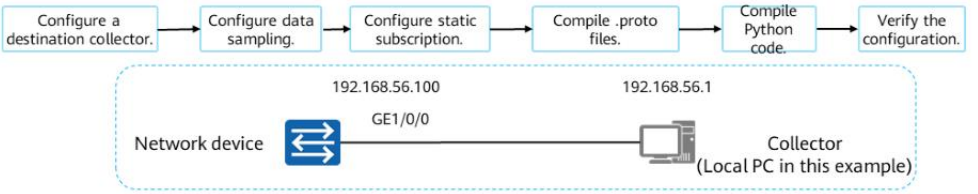
Description:

- A company has deployed a CE12800 switch with the management IP address 192.168.56.100. To better collect device performance data, it is required that the device push CPU information to the collector in static telemetry subscription mode.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

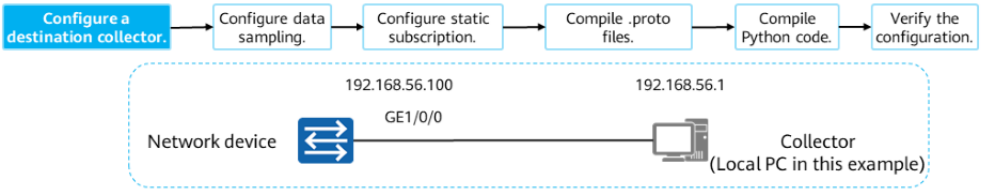
Configuration Roadmap



- Device configuration:
  - Enter the telemetry view.
  - Configure a destination collector for receiving sampled data.
  - Configure data sampling.
  - Configure static telemetry subscription.
- Collector configuration:
  - Prepare and compile .proto files.
  - Compile Python code: Import modules, create a gRPC server, obtain telemetry data, and decode telemetry data.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Configuring a Destination Collector



1. Enter the telemetry view.

```
<CE1> system-view immediately
[CE1] telemetry
[CE1-telemetry]
```

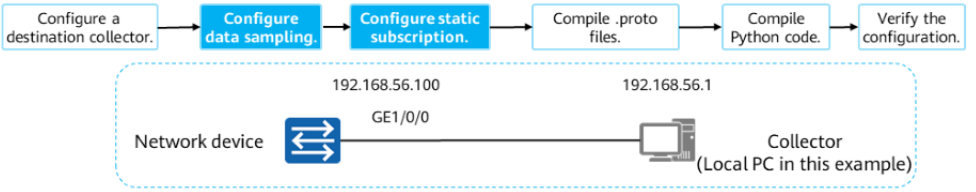
2. Configure a destination collector for receiving sampled data.

   Create a destination group where a destination collector belongs. In this example, create a destination group named **Dest1** and set the IP address and port number of the destination collector to 192.168.56.1 and 20000, respectively.

```
[CE1-telemetry] destination-group Dest1
[CE1-telemetry-destination-group-Dest1] ipv4-address
192.168.56.1 port 20000 protocol grpc no-tls
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Configuring Data Sampling and Static Subscription



3. Configure data sampling.

   When configuring static subscription to collect sampled data, create a sampling sensor group and specify a sampling path. In this example, create a sampling group named **Sensor1** and configure a sampling path of CPU information.

```
[CE1-telemetry] sensor-group Sensor1
[CE1-telemetry-sensor-group-Sensor1] sensor-path
huawei-devm:devm/cpuInfos/cpuInfo
```

4. Configure static subscription.

   Create a subscription by associating the destination group and the sampling sensor group. In this example, the destination group **Dest1** is associated with the sampling sensor group **Sensor1**, and the sampling interval is set to 1000 ms. After the configuration is complete, the device continuously pushes data to the destination collector.

```
[CE1-telemetry]subscription Sub1
[CE1-telemetry-subscription-Sub1]destination-group Dest1
[CE1-telemetry-subscription-Sub1]sensor-group Sensor1
sample-interval 1000
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Compiling .proto Files for the Collector (1/3)



5. Compile .proto files.
   - After the device pushes the sampled data to the collector using telemetry, the collector must connect to the device to obtain the sampled data. gRPC and GPB knowledge and development experience are required for interconnection.
   - The .proto files corresponding to the device are released. The collector needs to perform secondary development based on these .proto files to connect to the device. You can download the Huawei device .proto files from the Huawei official website.
     - Common .proto file:
       - **huawei-grpc-dialin.proto**: The collector functions as the client to send an RPC request to the device to subscribe to sampled data for dynamic telemetry subscription.
       - **huawei-grpc-dialout.proto**: The device functions as the client to initiate an RPC request to the collector to push data for static telemetry subscription.
       - **huawei-telemetry.proto**: After sampling service data, the device uses the telemetry header to encapsulate the data, facilitating interconnection with the collector.
     - Service .proto files:
       - **huawei-xxx.proto**: After sampling service data, the device encodes the sampled data in GPB method using this file, and the NMS decodes data using this file as well. Example: **huawei-mac.proto** and **huawei-mpls.proto**

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Compiling .proto Files for the Collector (2/3)

Configure a destination collector. → Configure data sampling. → Configure static subscription. → **Compile .proto files.** → Compile Python code. → Verify the configuration.

- Compile .proto files to obtain the method invoked by the server. You can create a Python script, for example, **run_codegen.py**, and compile .proto files with the following codes:

```
from grpc_tools import protoc
protoc.main(
    (
        '',
        '-I./protos',
        '--python_out=.',
        '--grpc_python_out=.',
        './protos/huawei-grpc-dialout.proto', #dialout file path
    )
)
```

```
protoc.main(
    (
        '',
        '-I./protos',
        '--python_out=.',
        '--grpc_python_out=.',
        './protos/huawei-telemetry.proto', # telemetry file path
    )
)
```

```
protoc.main(
    (
        '',
        '-I./protos',
        '--python_out=.',
        '--grpc_python_out=.',
        './protos/huawei-devm.proto', # devm file path
    )
)
```

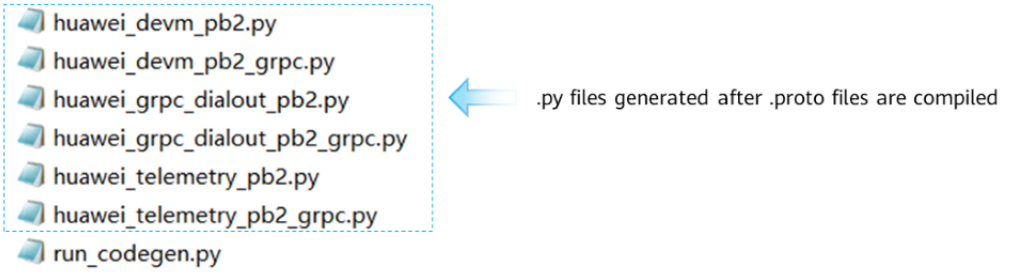

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- After the files are compiled successfully, multiple Python files are generated in the current folder.

Compiling .proto Files for the Collector (3/3)

Configure a destination collector. → Configure data sampling. → Configure static subscription. → **Compile .proto files.** → Compile Python code. → Verify the configuration.

- Run the **run_codegen.py** script file to obtain related files generated in the current directory after .proto files are compiled in Python.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Compiling Python Code (1/2)

Configure a destination collector. → Configure data sampling. → Configure static subscription. → Compile .proto files. → **Compile Python code.** → Verify the configuration.

6. Compile Python code to obtain device information.

Import modules. --

```
from concurrent import futures
import time
import importlib
import grpc
import huawei_grpc_dialout_pb2_grpc
import huawei_telemetry_pb2
```

```
_ONE_DAY_IN_SECONDS = 60 * 60 * 24


def serve():
server = grpc.server(futures.ThreadPoolExecutor(max_workers=10))
huawei_grpc_dialout_pb2_grpc.add_gRPCDataserviceServicer_to_server(Telemetry_CPU_Info(), server)
server.add_insecure_port('192.168.56.1:20000')
server.start()
try:
    while True:
      time.sleep(_ONE_DAY_IN_SECONDS)
  except KeyboardInterrupt:
    server.stop(0)
```

Create a gRPC server object. --
Register the data listening service. --
Set the socket listening port. --
Start the gRPC server. --
Infinite loop listening --

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- The gRPC module is installed by running the **pip install grpc** command.
- The **huawei_grpc_dialout_pb2_grpc** and **huawei_telemetry_pb2** modules are generated after the .proto files are compiled.

Compiling Python Code (2/2)

Create a class to inherit the Servicer method. --

Define the initialization method. --

Inherit the dataPublish method. --

Dynamically load a module for obtaining data through telemetry. --

Dynamically load a module for obtaining data through telemetry. --

Decode and print data. --

```
class Telemetry_CPU_Info(huawei_grpc_dialout_pb2_grpc.gRPCDataserviceServicer):
  def __init__(self):
      return
  def dataPublish(self, request_iterator, context):
      for i in request_iterator:
          print ('############ start ############\n')
          telemetry_data = huawei_telemetry_pb2.Telemetry.FromString(i.data)
          print (telemetry_data)

          for row_data in  telemetry_data.data_gpb.row:
              print ('-----------------')
              print ('The proto path is :'+telemetry_data.proto_path)
              print ('-----------------')
              module_name = telemetry_data.proto_path.split('.')[0]
              root_class  = telemetry_data.proto_path.split('.')[1]

              decode_module = importlib.import_module(  module_name+'_pb2')
              print (decode_module)
              decode_func = getattr(decode_module,root_class).FromString
              print ('----------- content is -----------\n')
              print (decode_func(row_data.content))
              print ('----------- done -----------------')
if __name__ == '__main__':
  serve()
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

Verifying the Configuration

Configure a destination collector. → Configure data sampling. → Configure static subscription. → Compile .proto files. → Compile Python code. → **Verify the configuration.**

7. Verify the configuration.

   Run the following script to check whether the CPU information is successfully obtained.

```
---------- content is ----------

cpuInfos {
  cpuInfo {
    entIndex: 16842753
    interval: 8
    ovloadThreshold: 90
    position: "1"
    systemCpuUsage: 6
    unovloadThreshold: 75
  }
}

---------- done ----------
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Quiz

1. (Multiple) Which of the following methods can be used to push data using telemetry? ( )
   A. TCP
   B. UDP
   C. gRPC
2. What are telemetry data subscription modes?

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

1. BC
2. Static subscription and dynamic subscription

Summary

- Telemetry remotely collects data from physical or virtual devices at a high speed.
- In a narrow sense, the telemetry framework consists of four modules: data source, data generation, data subscription, and data push.
- The data source uses YANG models to define devices. Data is generated using the GPB encoding method. Data subscription can be implemented in static or dynamic mode. Sampled data can be pushed based on gRPC or UDP.
- Python scripts can be run to subscribe to sampled device data through telemetry.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

More Information

- Protobuf official website
  - https://developers.google.com/protocol-buffers/
- gRPC official website
  - https://www.grpc.io/

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.


Thank You
www.huawei.com



OPS Fundamentals and Practices

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Foreword

- A conventional network device is a closed system that provides fixed functions and services and is not dynamic or flexible. With the rapid development and popularization of networks, diversified and differentiated requirements emerge one after another. Conventional network devices cannot meet these requirements. Some customers require devices with specific openness so that they can develop their own functions and deploy proprietary management policies to implement automatic management and maintenance, lowering management costs.
- Huawei has launched the open programmability system (OPS) to provide openness and programmability capabilities for network devices and enable users to perform secondary development, fully unleashing the potential of devices.
- This course introduces the OPS, describes its principles and application scenarios, and provides a case.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Objectives

- On completion of this course, you will be able to:
  - Describe basic OPS concepts.
  - Describe basic OPS functions.
  - Understand the OPS working principles.
  - Implement the basic OPS configuration.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. **OPS Overview**
2. OPS Fundamentals
3. OPS Application Scenarios
4. OPS Configuration Practices

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

OPS Overview (1)

- In the following scenarios:
  - During network deployment, services need to be configured for a large number of devices after hardware installation.
  - The network administrator wants devices to automatically take actions when alarms are generated.
  - The network is disconnected when the network administrator wants to remotely deliver commands and periodically collect device information through compiled scripts.



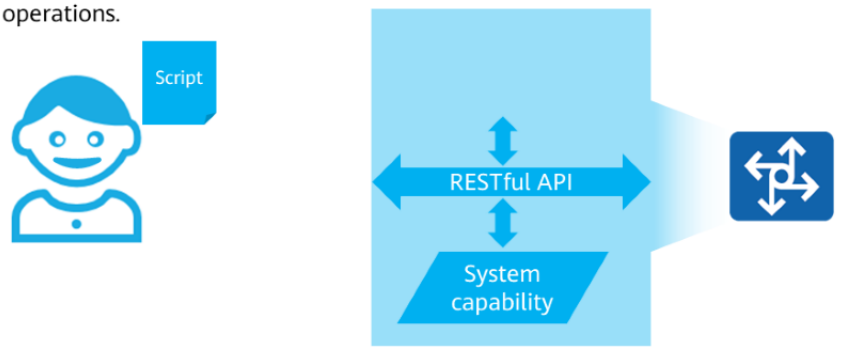
Conventional network devices cannot effectively solve the problems in these scenarios and meet the requirements.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Traditional network devices are relatively closed and cannot meet flexible and differentiated network management requirements.

OPS Overview (2)

- Huawei offers the OPS.
- The OPS allows users and third-party developers to develop and deploy network management policies using open RESTful APIs. It implements rapid service expansion, automatic function deployment, and intelligent device management, helping reduce network operation and maintenance costs and simplify network operations.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- With the OPS, you can compile scripts based on their requirements and import the scripts to network devices for running, which is flexible and efficient.

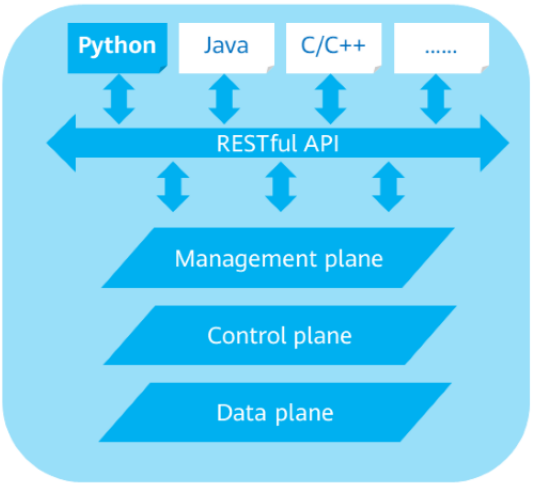
Contents

1. OPS Overview
2. **OPS Fundamentals**
3. OPS Application Scenarios
4. OPS Configuration Practices

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

OPS Architecture

- The OPS is developed on Huawei Versatile Routing Platform (VRP). It uses open RESTful APIs to work with VRP's management plane, control plane, and data plane, allowing for function expansion on a device.
- The OPS provides managed objects (MOs) to open devices. Each MO is uniquely identified by a Uniform Resource Identifier (URI). The client can perform operations on objects using standard HTTP methods, such as GET, PUT, POST, and DELETE.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- The VRP system is developed by Huawei based on years of research and network application experience and its intellectual property rights is owned by Huawei.
- Managed object (MO): an object that can be used to manage network devices by invoking RESTful APIs, such as CPU information, system information, and interface information.
- Uniform Resource Identifier (URI): identifies a specific resource. In the OPS, URIs are used to identify MOs. For example, the URI of the CPU information is /devm/cpuInfos/cpuInfo, which uniquely identifies the CPU information.
- Uniform resource locator (URL): A URL is a URI that can be used to present a resource and specify how to locate the resource, for example, http://www.ietf.org/rfc/rfc2396.txt and ftp://ftp.is.co.za/rfc/rfc1808.txt.
- Huawei network devices that support the OPS provides a running environment for Python scripts. Scripts in Java and C/C++ languages are not supported.

Introduction to RESTful APIs

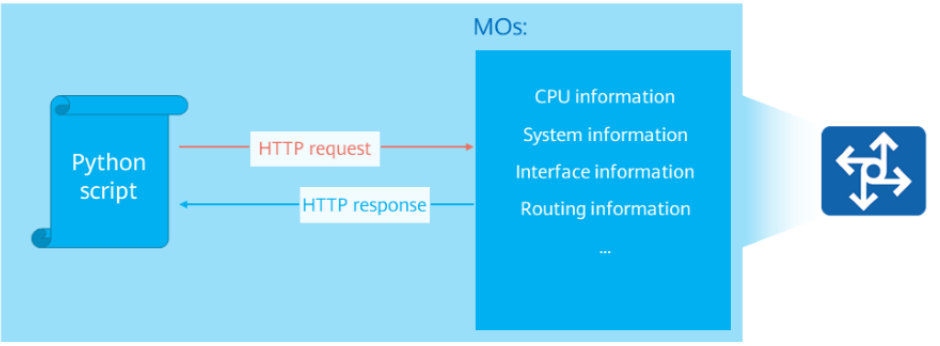
- Representational State Transfer (REST) is a style of software architecture. The design concepts and principles are as follows:
  - All elements on networks can be abstracted as resources.
  - Each resource has a unique resource ID. Operations on resources do not change the resource IDs.
  - Standard methods are used to operate resources. The core operations are GET, PUT, POST and DELETE defined in HTTP.
  - All operations are stateless.
- RESTful API: application programming interfaces (APIs) that comply with the REST architecture.
- The OPS defines a set of RESTful APIs and uses URIs to identify open MOs. You can access MOs using standard HTTP methods (GET, PUT, POST, and DELETE).

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- An API is a particular set of rules and specifications that are used for communication between software programs.
- For more information about RESTful, see the HCIP Programming and Automation Course — RESTful Fundamentals and Practices.

OPS RESTful API Interaction Example

- With RESTful APIs defined by the OPS, you can write Python scripts to send HTTP requests to perform operations on MOs of network devices. Network devices will return HTTP response messages based on the operation results.

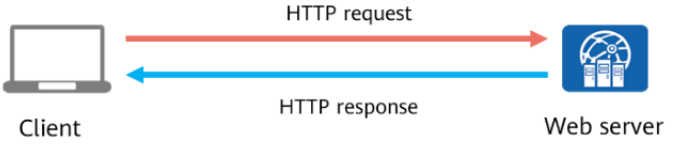


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- The OPS allows you to compile Python scripts, install the scripts on network devices, and send HTTP requests when the scripts are running to manage network devices.

Overview of HTTP

- Hyper Text Transfer Protocol (HTTP) is the most widely used network transfer protocol on the Internet. It is used to transfer hypertext from the World Wide Web server to the local browser.
- HTTP uses the client/server (C/S) architecture model and uses TCP/IP to transfer data, such as HTML files, image files, and query results. HTTP uses the TCP port 80 by default.

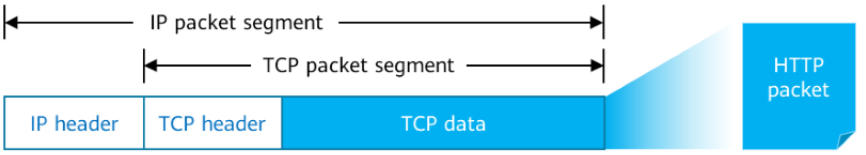


 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Currently, the implementation of RESTful APIs uses the HTTP standard specifications. Therefore, this section briefly describes HTTP.

HTTP Packet Format (1)

- HTTP packets are carried over TCP. The following figure shows the position of HTTP packets in TCP packets.



- An HTTP packet (request message and response message) consists of the following:
  - Start line: describes operations to be performed in a request packet and the execution result in a response packet.
  - Header field: describes additional information about the request and response packets. For example, the header field Host in a request packet specifies the Internet host of the requested resource, and the header field Content-Type in the response packet header indicates the type of the content in the packet body.
  - Body: The body of a request packet is the data to be sent by the client to the web server. The body of a response packet is the data to be returned by the web server to the client, such as HTML texts, images, and videos.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HTTP Packet Format (2)

- The syntax of HTTP packets is as follows:

	Request Packet Format	Response Packet Format
Start line	<method> <request-URL> <version>	<version> <status-code> <reason-phrase>
Header field	<headers>	<headers>
Body	<entity-body>	<entity-body>

The request and response packets differ only in the syntax of the start line.

- method: action that the client expects the server to perform on a resource. It is a single word, for example, GET or POST.
- request-URL: path of the requested resource on the server.
- version: HTTP version used by packets. The format is as follows: HTTP/<major version No.>.<minor version No.>.
- status-code: HTTP status code, which indicates the execution result of a request operation. The status code is a three-digit number.
- reason-phrase: reason phrase, which explains the status code in human-readable mode.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- <headers> and <entity-body> are the header field and body of the packet on the previous page.

HTTP Packet Format (3)

- HTTP packet instance

	Request Packet	Response Packet
Start line	GET / HTTP/1.1	HTTP/1.1 200 OK
Header field	Host: www.example.com User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) Accept: text/html,application/xhtml+xml Accept-Language: zh-CN,zh;q=0.9	Date: Mon, 15 Jun 2020 10:21:47 GMT Server: VWebServer Last-Modified: Mon, 15 Jun 2020 10:11:12 GMT ETag: "e211-5a81ca574d000" Accept-Ranges: bytes Vary: Accept-Encoding Content-Length: 13358 Content-Type: text/html
Body	The GET method is usually used to request the server to send a resource whose body is empty.	<HTML> <HEAD> <TITLE>Example Homepage</TITLE> </HEAD> <BODY> <!– Body goes here --> </BODY> </HTML>

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Header field:
  - Host: contains host name and port number of the web server.
  - User-Agent: contains information about the user agent originating the request.
  - Accept: specifies response media types that are acceptable.
  - Accept-Language: indicate the set of natural languages that are preferred in the response.
  - Date: represents the date and time at which the message was originated.
  - Server: contains information about the software used by the origin server to handle the request.
  - Last-Modified: last modified date for the requested object.
  - ETag: specifies an identifier for a specific version of a resource, often a message digest.
  - Accept-Ranges: allows a server to indicate that it supports range requests for the target resource.
  - Vary: describes what parts of a request message, aside from the method, Host header field, and request target, might influence the origin server's process for selecting and representing this response.
  - Content-Length: indicates length of the response body in octets.
  - Content-Type: indicates Multipurpose Internet Mail Extensions (MIME) type of

content.

- For details about the header fields, see RFC 2616.

OPS RESTful API Description Example

- The HTTP requests sent by a user must be constructed based on the RESTful APIs defined by the OPS. The API for obtaining system information is described as follows:

Request example including the HTTP header (XML format):

```
GET /system/systemInfo HTTP/1.1
Host: localhost
Accept-Encoding: identity
Content-Length: 66
Content-Type: application/xml
Accept: application/xml

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

For details about the RESTful APIs of the OPS, see the *RESTful API Reference*.

Response example including the HTTP header (XML format):

```
HTTP/1.1 200 OK
Cache-Control: no-cache
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?> <systemInfo>
  <sysName>HUAWEI</sysName>
  <platformName>VRP</platformName>
  <platformVer>v800r007c00</platformVer>
  <productName>Device</productName>
  <productVer>V800R007C00</productVer>
  <patchVer></patchVer>
  <esn>123456789</esn>
  <mac>00e0-fc34-1234</mac>
</systemInfo>
```

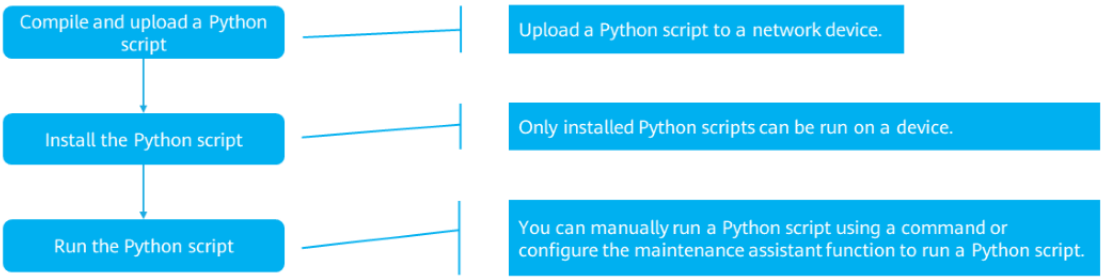
 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- The formats of the OPS RESTful API request and response packets are similar to those of the HTTP request and response packets described in the previous slide.
- Extensible Markup Language (XML) is designed to transmit and store data.
- Currently the OPS RESTful APIs use the XML format to transmit data. In a later version, the APIs can use the JavaScript Object Notation (JSON) format to transmit data. Therefore, the body of the OPS RESTful API request and response packets is in XML format.
- You can download RESTful API Reference on the network device page of http://support.huawei.com.

Usage of Python Scripts

- Currently, the OPS RESTful APIs can be used only in the Embedded Running Environment (ERE). You can compile Python scripts based on OPS APIs and install them on the devices that provide a Python running system. When a Python script is running, an HTTP request is sent to the system to manage devices.
- The following figure shows the Python script execution process.



 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- The maintenance assistant is a function of Huawei network devices. You can set the trigger conditions and the Python script to be executed when the conditions are met. The system monitors device running in real time. When the specified trigger condition is met, the network device system automatically executes the Python script to complete the actions defined in the script. For more information about the maintenance assistant, see the Huawei network device product documentation.

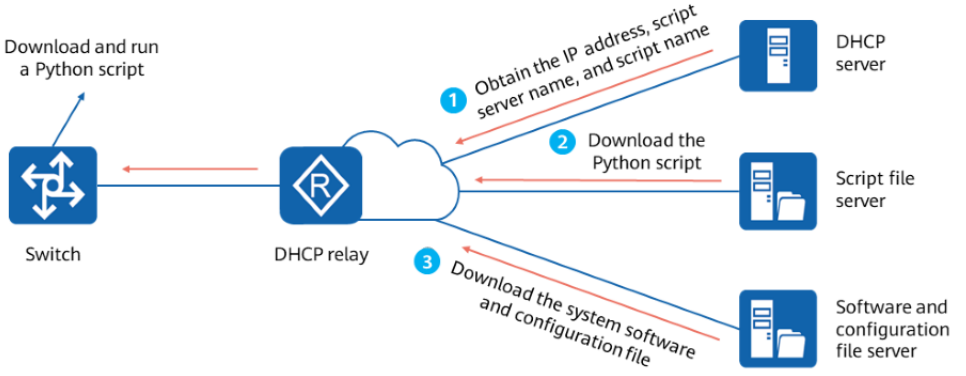
Contents

1. OPS Overview
2. OPS Fundamentals
3. **OPS Application Scenarios**
4. OPS Configuration Practices

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Scenario 1: Automatic Deployment of Unconfigured Devices

- The OPS can implement automatic deployment of unconfigured devices without the need of on-site installation, reducing labor costs and improving deployment efficiency.

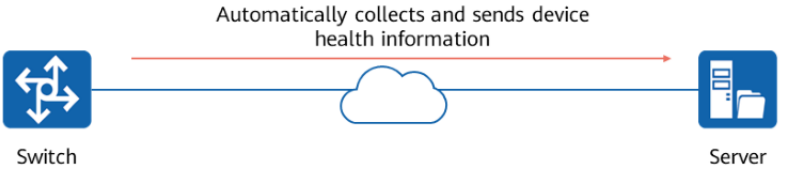


 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- DHCP server: allocates the temporary IP address, default gateway, and script file server address to the device to be automatically deployed.
- DHCP relay agent: forwards packets exchanged between the device to be automatically deployed and the DHCP server when they are located on different network segments.
- Script file server: stores scripts (Python) required for automatic network device deployment. By running the script files, a network device can obtain information such as the IP address of the software and configuration file server, version file, and configuration file.
- Software and configuration file server: stores system software, configuration files, and patch files required for automatic network device deployment.

Scenario 2: Automatic Health Check

- During conventional device health check, you need to log in to a device and run commands to check the hardware and service running status.
- With the OPS function, a device can automatically run the health check commands, periodically collects health check results, and sends the results to a server for analysis. This function reduces maintenance workload. These commands are delivered through the Python scripts installed in the network device system instead of being remotely delivered. Therefore, you do not need to worry about network disconnection.

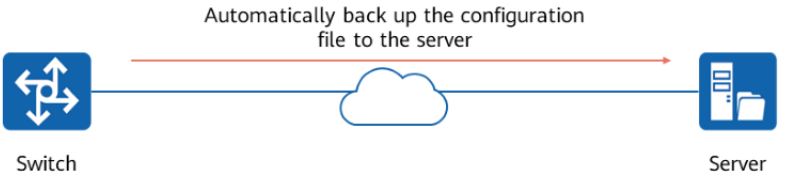


 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- A Python script can be compiled to deliver commands. When the network is disconnected, the execution result is temporarily stored on the device. After the network is recovered, the execution result is transmitted to the server. Therefore, the impact of network disconnection can be mitigated.

Scenario 3: Automatic Backup of Configuration Files

- A network device subscribes to the configuration file saving events through the maintenance assistant. After the configuration is saved, the device automatically runs the Python script and sends its configuration file to the server for backup, reducing the manual backup workload.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

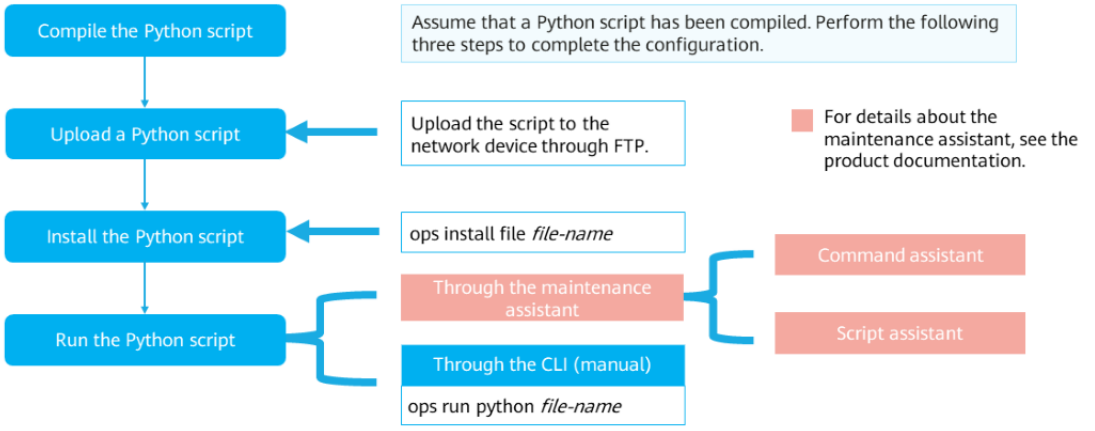
Contents

1. OPS Overview
2. OPS Fundamentals
3. OPS Application Scenarios
4. **OPS Configuration Practices**

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Configuration Roadmap

- The configuration roadmap is as follows: The core is to compile Python scripts, which will be detailed later. For the operation modes and involved commands of the other three steps, see the following figure.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Compiling a Python Script (1)

- Python scripts are used to deliver HTTP requests to the system for management. Huawei provides Python script templates. You only need to search for RESTful APIs based on the functions to be implemented and modify code.
- Example of a Python script template

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import traceback
import httplib
import string

class OPSConnection(object):
    """Make an OPS connection instance."""

#: an initialization class that creates an HTTP connection
 def __init__(self, host, port = 80):
        self.host = host
        self.port = port
        self.headers = {
(See the next slide for the rest of code.)
```

- The import statement is used to import third-party modules required in the Python script template.
- The OPSConnection class is used to invoke RESTful APIs. This class defines methods for establishing HTTP connections and does not need to be modified.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Compiling a Python Script (2)

```
            "Content-type": "text/xml",
            "Accept":       "text/xml"
            }
        self.conn = None

# Disable an HTTP connection.
    def close(self):
        """Close the connection"""
        self.conn.close()

    # Create device resources.
    def create(self, uri, req_data):
        """Create operation"""
        ret = self.rest_call("POST", uri, req_data)
        return ret

    # Delete device resources.
    def delete(self, uri, req_data):
        """Delete operation"""
        ret = self.rest_call("DELETE", uri, req_data)
        return ret
(See the next slide for the rest of code.)
```

- Methods defined in the OPSConnection class:
  - def _init_(): is an initialization method that creates an HTTP connection.
  - def close(): shuts down an HTTP connection.
  - def create(): creates device resources.
  - def delete(): deletes device resources.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Compiling a Python Script (3)

```
# Query device resources.
def get(self, uri, req_data = None):
    """Get operation"""
    ret = self.rest_call("GET", uri, req_data)
    return ret

# Modify device resources.
def set(self, uri, req_data):
    """Set operation"""
    ret = self.rest_call("PUT", uri, req_data)
    return ret

# Invoke classes internally.
def rest_call(self, method, uri, req_data):
    """REST call"""
    print('|---------- request: ----------|')
    print('%s %s HTTP/1.1\n' % (method, uri))
    if req_data == None:
        body = ""
    else:
        body = req_data
```

- Methods defined in the OPSConnection class:
  - def get(): queries device resources.
  - def set(): modifies device resources.
  - def rest_call(): indicates a class method that is invoked by a method mentioned above (create, delete, get, or set).
- You do not need to modify the methods defined in the OPSConnection class.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Compiling a Python Script (4)

```
            print(body)
        if self.conn:
            self.conn.close()
        self.conn = httplib.HTTPConnection(self.host, self.port)

        self.conn.request(method, uri, body, self.headers)
        response = self.conn.getresponse()
        response.status = httplib.OK    # stub code
        ret = (response.status, response.reason, response.read())
        print('|----------------------------------  response: ----------------------------------|')
        print('HTTP/1.1 %s %s\n\n%s' % ret)
        print('|-------------------------------------------------------------------------------|')
        return ret

# Define the function for obtaining system startup information.
def get_startup_info(ops_conn):

    uri = "/cfg/startupInfos/startupInfo"

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

- You need to compile a method to invoke the RESTful API through an instance of the OPSConnection class to manage devices. For example, the def get_startup_info(ops_conn) method in this document is invoked to obtain system startup information. You can specify the method name.
- uri = "/cfg/startupInfos/startupInfo" is the URI of the system startup information. Different MOs have different URIs. You need to change the URI as required.

The code in red needs to be modified based on the site requirements when you compile a Python script.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Compiling a Python Script (5)

```
<startupInfo>
</startupInfo>
'''
# The response data of the system startup information is similar to the following:
    '''
    <?xml version="1.0" encoding="UTF-8"?>
    <rpc-reply>
      <data>
        <cfg xmlns="http://www.huawei.com/netconf/vrp" format-version="1.0"
content-version="1.0">
          <startupInfos>
            <startupInfo>
              <position>6</position>
              <nextStartupFile>flash:/vrpcfg.cfg</nextStartupFile>
              <configedSysSoft>flash:/system-software.cc</configedSysSoft>
              <curSysSoft>flash:/system-software.cc</curSysSoft>
              <nextSysSoft>flash:/system-software.cc</nextSysSoft>
              <curStartupFile>flash:/vrpcfg.cfg</curStartupFile>
              <curPatchFile>NULL</curPatchFile>
              <nextPatchFile>NULL</nextPatchFile>
            </startupInfo>
```

- You can obtain the values of uri and req_data from *RESTful API Reference* based on the site requirements.
- The system startup information in this document is described in the *RESTful API Reference* as follows:

Operation	URI	Description
GET	/cfg/startupinfos/startupinfo	Obtain system startup information

Request example

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


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Compiling a Python Script (6)

```
        </startupInfos>
      </cfg>
    </data>
  </rpc-reply>
  '''

  # Execute a GET operation request.
  ret, _, rsp_data = ops_conn.get(uri, req_data)
  if ret != httplib.OK:
      return None

  return rsp_data

# main() function defines the operations to be performed during script running. You can modify the function according to service requirements.
def main():
  """The main function."""
  host = "localhost"
  try:
      # Establish an HTTP connection.
      ops_conn = OPSConnection(host)
      # Invoke the function for obtaining system startup information.
```

- According to the *RESTful API Reference*, the HTTP operation corresponding to the RESTful API for obtaining system startup information is GET. Therefore, the get method of the OPSConnection class instance is invoked.
- You can change the get method to the create, delete, or set method of the OPSConnection class based on the device management function to be implemented.
- req_data indicates the request content, and rsp_data indicates the response message after the network device performs the request operation.
- In the main() method, host indicates the loop address. Currently, RESTful APIs can be invoked only inside the device, that is, the value is localhost.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Compiling a Python Script (7)

```
        rsp_data = get_startup_info(ops_conn)
        # Disable an HTTP connection.
        ops_conn.close()
        return

    except:
        errinfo = traceback.format_exc()
        print(errinfo)
        return

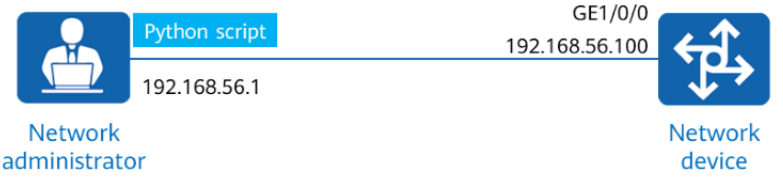
if __name__ == "__main__":
    main()
```

- You can invoke the method compiled based on the RESTful API in the main() method, for example, the get_startup_info method in this document, to complete the desired function.
- The main() method can flexibly implement various management functions of a network device by using various RESTful APIs and structures of Python such as loops and if-else.
- The last line of the script indicates that the main() method is executed.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Case: Obtaining CPU Information of a Device

- Description: A network administrator compiles a Python script and uploads the script to a network device to obtain CPU information by using the OPS function of the device. The networking is as follows.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Compiling a Python Script (1)

- Based on the function of obtaining device CPU information to be implemented, obtain the URI, request message, and response message by referring to the *RESTful API Reference*. The result is as follows.

URI information:

Operation	URI	Description
GET	/devm/cpuInfos/cpuInfo	Query a board's CPU usage

Request message:

```
<?xml version='1.0'  encoding='UTF-8'?>
  <cpuInfo>
    <position></position>
    <entIndex></entIdex>
    <systemCpuUsage></systemCpuUsage>
    <ovloadThreshold><ovloadTreshold>
    <unovloadThreshold></unovloadThreshold>
</cpuInfo>
```

Response message:

```
<?xml version="1.0" encoding="UTF-8"?>
<cpuInfo>
   <position chassis="1" slot="17">17</position>
   <entIndex>17891329</entIndex>
   <systemCpuUsage>5</systemCpuUsage>
   <ovloadThreshold>95</ovloadThreshold>
   <unovloadThreshold>75</unovloadThreshold>
</cpuInfo>
```

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- After knowing the format of the response message, you can parse the response message in the Python script. In this case, the response message is only displayed. You can try to parse the response message to implement more complex functions.

Compiling a Python Script (2)

- Modify the related code in the Python script template based on the URI and request content. Other code in the template does not need to be modified.

Define the get_cpu_info method.

```
def get_cpu_info(ops_conn):
    uri = "/devm/cpuInfos/cpuInfo"                #URI corresponding to the CPU information

    req_data = \                                  #Set the content of the request for obtaining CPU information.
      '''<?xml version='1.0' encoding='UTF-8'?>
   <cpuInfo>
     <position></position>
     <entIndex></entIndex>
     <systemCpuUsage></systemCpuUsage>
     <ovloadThreshold></ovloadThreshold>
     <unovloadThreshold></unovloadThreshold>
   </cpuInfo>
      '''
   ret, _, rsp_data = ops_conn.get(uri, req_data)   #Execute the get operation.
   if ret != httplib.OK:
      return None

   return rsp_data
```

Invoke the get_cpu_info method in the main method.

```
def main():
   """The main function."""

   host = "localhost"
   try:
      # Establish an HTTP connection.
      ops_conn = OPSConnection(host)
      # Invoke the function for obtaining CPU information.
      rsp_data = get_cpu_info(ops_conn)
      # Disable an HTTP connection.
      ops_conn.close()
      return

   except:
      errinfo = traceback.format_exc()
      print(errinfo)
      return
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Uploading a Python Script

- After the Python script is compiled, save it as cpu_demo.py. The network administrator enables the FTP server on the local PC, and the network device functions as the FTP client to download the Python script.

```
<CE1>ftp 192.168.56.1                                          # Connect to the FTP server.
Trying 192.168.56.1 ...
Press CTRL + K to abort
Connected to 192.168.56.1.
220 WFTPD 2.0 service (by Texas Imperial Software) ready for new user
User(192.168.56.1:(none)):1                                    # Enter the user name.
331 Give me your password, please
Enter password:                                                # Enter the password.
230 Logged in successfully
[ftp]get cpu_demo.py                                           # Obtain the cpu_demo.py script file.
Warning: The file may not transfer correctly in ASCII mode.
500 Unidentified command SIZE cpu_demo.py
200 PORT command okay
150 "D:\FTPSERVER\cpu_demo.py" file ready to send (4605 bytes) in ASCII mode
..
226 Transfer finished successfully.                            # FTP transmission is complete.

FTP: 4605 byte(s) received in 1.135 second(s) 3.962Kbyte(s)/sec.
```

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- For details about how to enable the FTP server on the local PC, you can easily search the way from a search engine.

Compiling a Python Script

- After the Python script is uploaded, run the ops install file file-name command to install the script.

```
<CE1>ops install file cpu_demo.py                    # Install the Python script.
```

- Verify the installation result.

```
<CE1>display ops script                              # Check the installed Python script.
-------------------------------------------------------------------------------
  Index      Size(Byte)  Filename
-------------------------------------------------------------------------------
      0            4605  cpu_demo.py
-------------------------------------------------------------------------------
```

The **cpu_demo.py** script has been successfully installed.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Compiling a Python Script (1)

- Run the ops run python file-name command to run the Python script.

```
<CE1>ops run python cpu_demo.py
|---------------------------------- request: ----------------------------------|          #  Content of the get operation
GET /devm/cpuInfos/cpuInfo HTTP/1.1

<?xml version='1.0' encoding='UTF-8'?>
    <cpuInfo>
      <position></position>
      <entIndex></entIndex>
      <systemCpuUsage></systemCpuUsage>
      <ovloadThreshold></ovloadThreshold>
      <unovloadThreshold></unovloadThreshold>
   </cpuInfo>

|---------------------------------- response: ----------------------------------|         # Content of the response message
HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply>
  <data>
    <devm xmlns="http://www.huawei.com/netconf/vrp" format-version="1.0" content
-version="1.0">

(See the next slide for the rest of code.)
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Running the Python Script (2)

```
    <cpuInfos>                                                  # Returned CPU information. In this case, the network device has
      <cpuInfo>                                                   two cpus.
        <position chassis="1" slot="1">1</position>             # position indicates the CPU position information.
        <entIndex>16842753</entIndex>                           # entIndex indicates the CPU index information.
        <systemCpuUsage>4</systemCpuUsage>                      # systemCpuUsage indicates the CPU usage.
        <ovloadThreshold>90</ovloadThreshold>                   # ovloadThreshold indicates the CPU overload clearance threshold.
        <unovloadThreshold>75</unovloadThreshold>               # unovloadThreshold indicates the CPU non-overload threshold.
      </cpuInfo>
      <cpuInfo>
        <position chassis="1" slot="17">17</position>
        <entIndex>17891329</entIndex>
        <systemCpuUsage>4</systemCpuUsage>
        <ovloadThreshold>90</ovloadThreshold>
        <unovloadThreshold>75</unovloadThreshold>
      </cpuInfo>
    </cpuInfos>
  </devm>
 </data>
</rpc-reply>

|-----------------------------------------------------------------------------|
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Quiz

1. (Multiple) The OPS function uses standard HTTP methods to access managed objects (MOs). Which of the following methods are supported? ( )

   A. GET

   B. PUT

   C. POST

   D. DELETE

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

1. ABCD

Summary

- The OPS is a unique function of Huawei network devices. It uses RESTful APIs and standard HTTP methods to access MOs on devices, helping users easily cope with various network O&M requirements.
- Huawei provides Python script templates. You only need to search for RESTful APIs based on the functions to be implemented and modify code, which is easy to use.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



More Information

- For more REST information, see the paper written by Dr. Roy Fielding.
  - https://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm
- For more information about HTTP, see RFC 2616.
- Python official website: https://www.python.org
- RESTful API reference: https://support.huawei.com

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- The REST software architecture was first mentioned by Roy Fielding in his doctoral paper. Roy Fielding is one of the major authors of the HTTP specifications.


Thank You
www.huawei.com



SDN Overview

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Foreword

- The open ecosystem of the computing industry brings booming development of multiple fields. The network industry is seeking transformation and development, where Software Defined Networking (SDN) is a concept that attracts great attention.
- This course aims to help engineers understand SDN concepts and learn Huawei SDN solutions.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Objectives

- On completion of this course, you will be able to:
  - Describe the development of SDN.
  - Describe Huawei SDN solutions.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

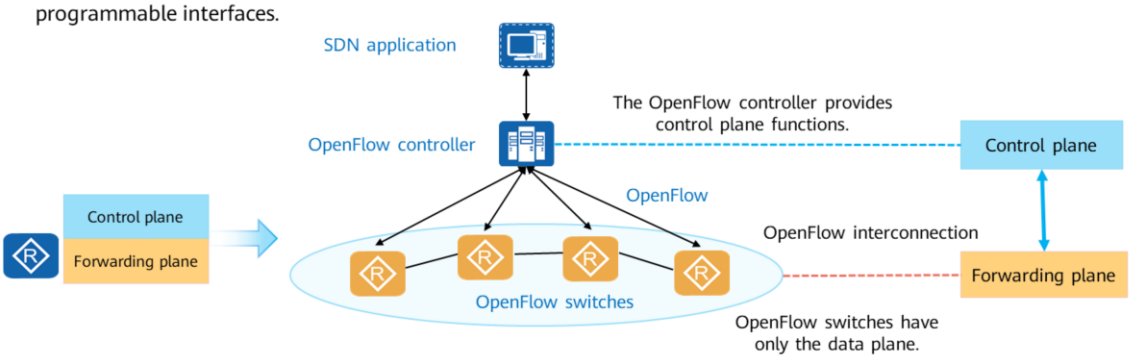
Contents

1. **SDN Overview**
2. Introduction to Huawei iMaster NCE

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

SDN Origin

- SDN was developed by the Clean Slate Program at Stanford University as an innovative new network architecture. The core of SDN is to separate the control plane from the data plane of network devices to implement centralized control of the network control plane and provide good support for network application innovation.
- SDN has three characteristics in initial phase: forwarding-control separation, centralized control, and open programmable interfaces.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- OpenFlow was defined in the initial phase of SDN. With technology development, many other southbound interface (SBI) protocols are defined between the controller and network devices.

Essential Requirements of SDN

- The essence of SDN is to make networks more open, flexible, and simple. It builds a core brain for a network and implements fast service deployment, traffic optimization, or network service openness through centralized control in the global view.
- Highlights:
  - Provides centralized management, simplifying network management and O&M.
  - Shields technical differences, simplifying network configuration and reducing O&M costs.
  - Offers automatic optimization, improving network utilization.
  - Deploys services rapidly, shortening the service provisioning time.
  - Builds an open network, supporting open and programmable third-party applications.

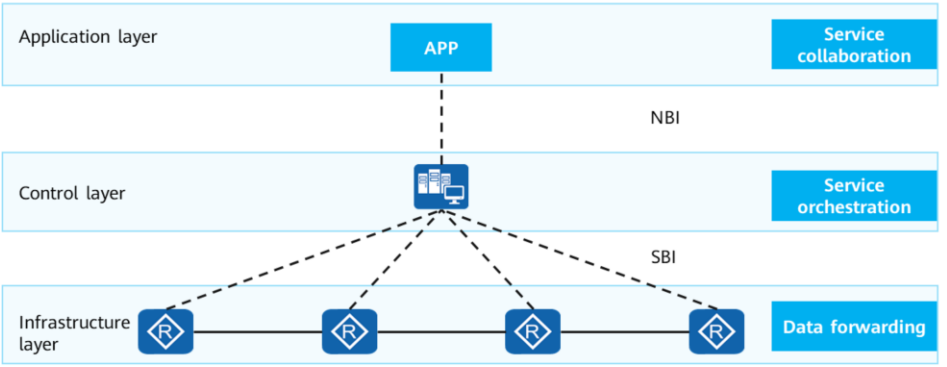
SDN transforms the network architecture.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- SDN is a broader concept, not limited to OpenFlow. Separation between the control and data planes is a method rather than the essence of SDN.

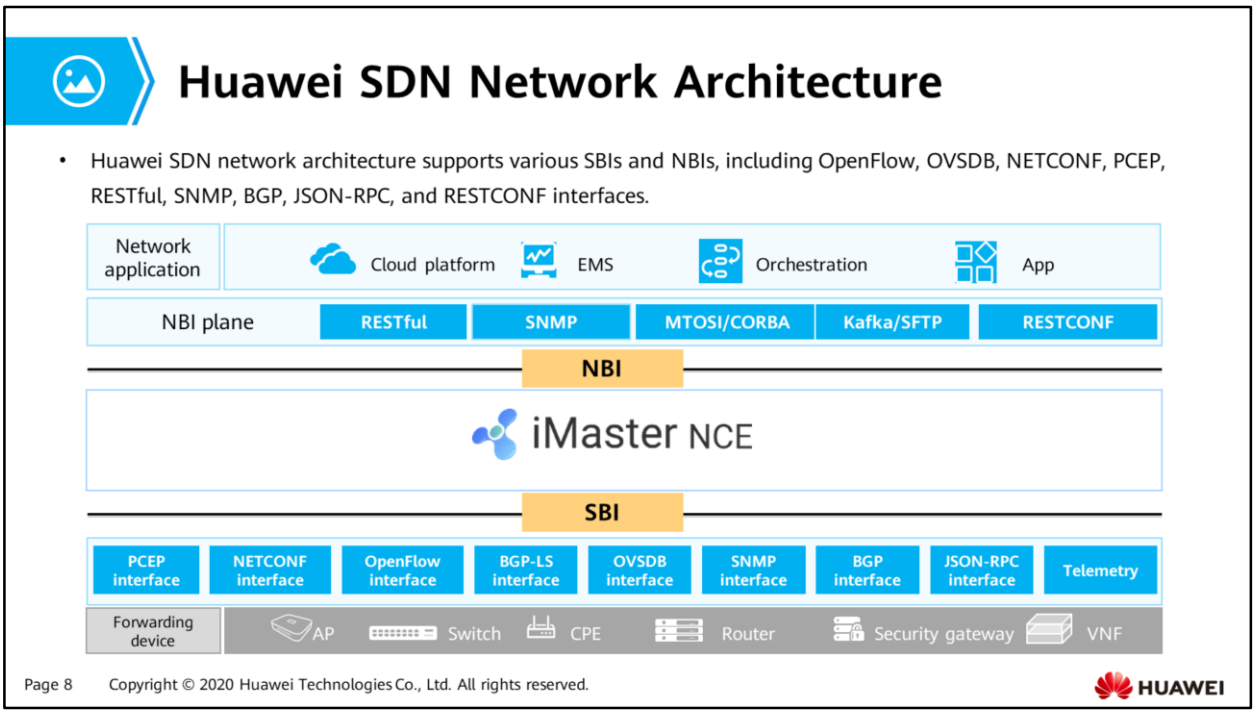
SDN Architecture

- An SDN architecture consists of the application layer, control layer, and infrastructure layer. To communicate between these layers, SDN uses open interfaces where the northbound interfaces (NBIs) communicate between the infrastructure and control layers and the southbound interfaces (SBIs) communicates between the application and the control layers. OpenFlow is an SBI protocol.



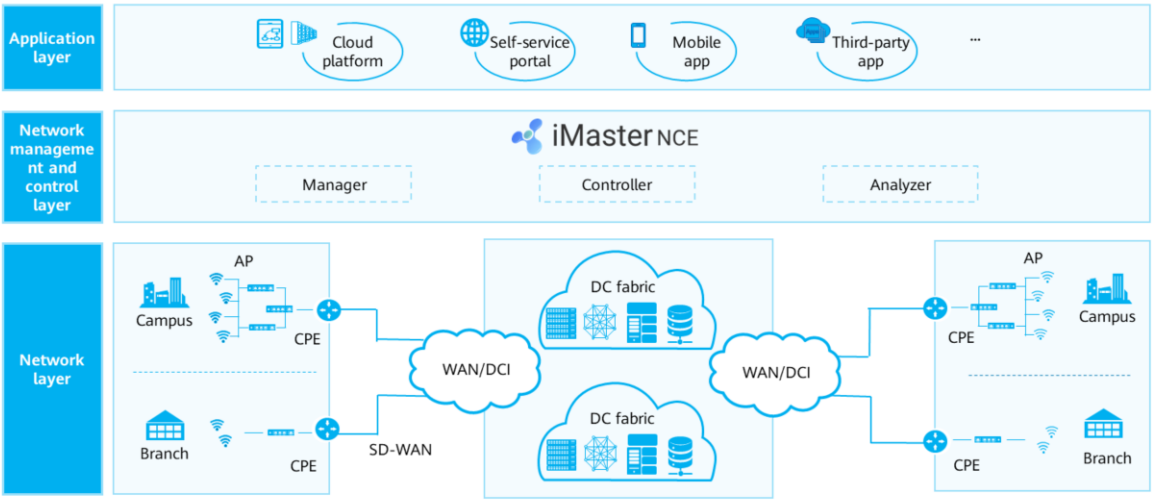
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Application layer: provides various upper-layer applications for service intents, such as OSS and OpenStack. The OSS is responsible for service orchestration of the entire network, and OpenStack is used for service orchestration of network, compute, and storage resources in a DC. There are also other applications at this layer. For example, a user deploys a security app. This app invokes NBIs of the controller, such as Block (Source IP，DestIP), regardless of the device locations. Then the controller delivers different instructions to network devices based on different southbound protocols.
- Control layer: The SDN controller is deployed at this layer and is the core of the SDN network architecture. The control layer is the brain of the SDN system and implements network service orchestration.
- Infrastructure layer: A network device receives instructions from the controller and performs data forwarding.
- NBI: NBIs, mainly RESTful APIs, are used by the controller to interconnect with the application layer.
- SBI: SBIs are used by the controller to interact with devices through protocols such as NETCONF, SNMP, OpenFlow, and OVSDB.



- Cloud platform: resource management platform in a cloud DC. The cloud platform manages network, compute, and storage resources. OpenStack is the most mainstream open-source cloud platform.
- The Element Management System (EMS) manages one or more telecommunication network elements (NEs) of a specific type.
- Container-based orchestration: The container-based orchestration tool can also provide the network service orchestration function. Kubernetes is a mainstream tool.
- MTOSI or CORBA is used to interconnect with the BSS or OSS. Kafka or SFTP can be used to connect to a big data platform.

Huawei SDN Solution - Integrating Management, Control, and Analysis to Build an Intent-Driven Network



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

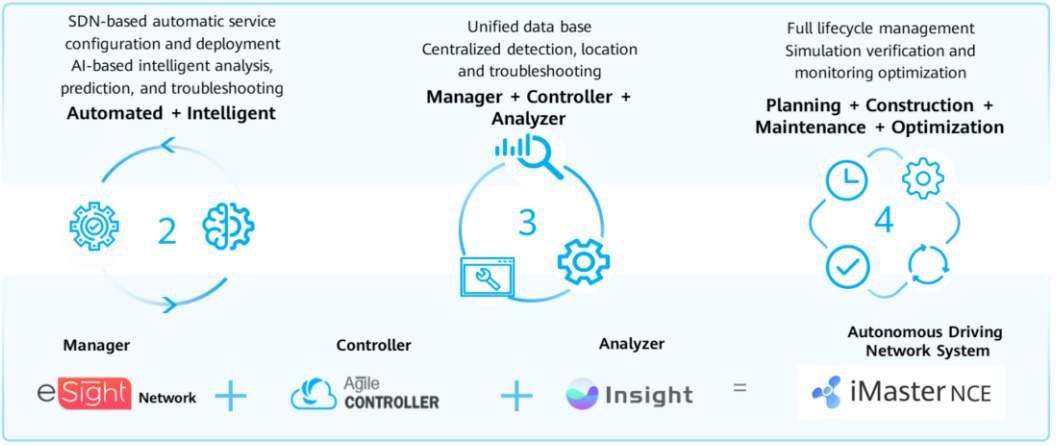
Contents

1. SDN overview
2. **Introduction to Huawei iMaster NCE**
   - iMaster NCE-Campus
   - iMaster NCE-Fabric

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Introduction to iMaster NCE

- Huawei iMaster NCE is an industry intelligent network automation platform that integrates management, control, analysis, and AI capabilities.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- iMaster NCE effectively connects physical networks with business intents and implements centralized management, control, and analysis of global networks. It enables resource cloudification, full lifecycle automation, and data analytics-driven intelligent closed-loop management according to business and service intents and provides open network APIs for rapid integration with IT systems.
- Huawei iMaster NCE can be used in the enterprise data center network (DCN), enterprise campus, and enterprise branch interconnection (SD-WAN) scenarios to make enterprise networks simpler, smarter, open, and secure, accelerating enterprise service transformation and innovation.

iMaster NCE Application



iMaster NCE

Data center	iMaster NCE-Fabric *
Enterprise campus	iMaster NCE-Campus *
SD-WAN	iMaster NCE-WAN
IP WAN	iMaster NCE-IP
WAN transmission	iMaster NCE-T

* Introduced in this document

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

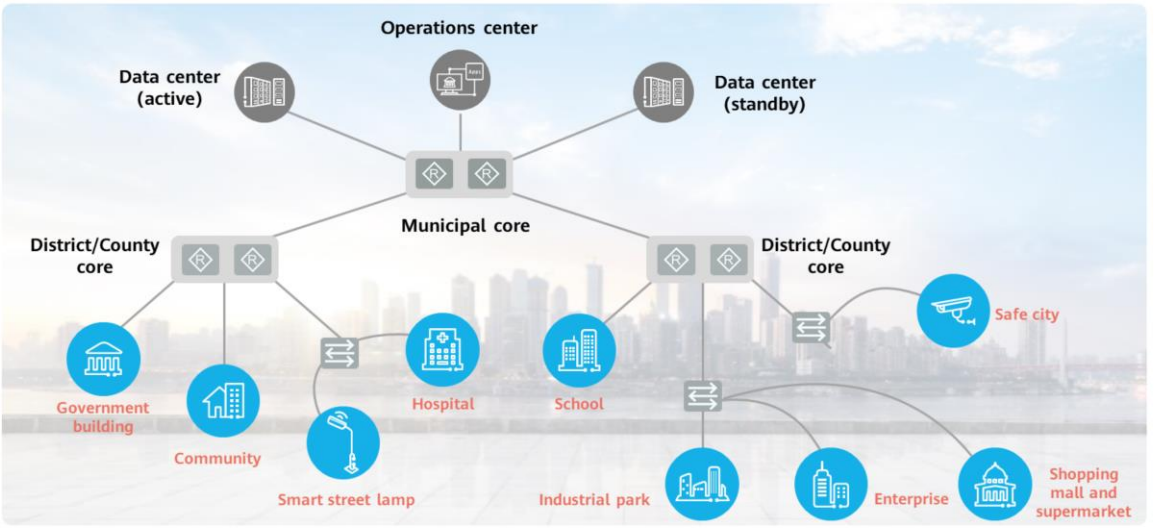
Contents

1. SDN overview
2. **Introduction to Huawei iMaster NCE**
   - **iMaster NCE-Campus**
   - iMaster NCE-Fabric

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

Introduction to a Campus



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- Campuses are everywhere, including factories, government buildings and facilities, shopping malls, office buildings, school campuses, and parks. According to statistics, 90% of urban residents work and live in campuses, 80% of gross domestic product (GDP) is created in campuses, and each person stays in campuses for 18 hours every day.
- Campus networks, as the infrastructure for campuses to connect to the digital world, are an indispensable part of campus construction and play an increasingly important role in daily working, R&D, production, and operation management.
- Campus networks vary according to the size and industry characteristics. From the industry perspective, typical industry campus networks include education campus networks, government campus networks, commercial campus networks, office campus networks, and manufacturing campus networks.

What Is a Campus Network?



- A campus network generally refers to the internal network of an enterprise or organization, which is connected to the wide area network (WAN) and data center networks.
- A campus network is built to ensure that key enterprise services are running more efficiently.
- Based on the scale, campus networks can be classified into large-, medium-, and small-scale campus networks.
- Some enterprises have branches dispersed in different areas. Each branch network can be considered as a single campus network.

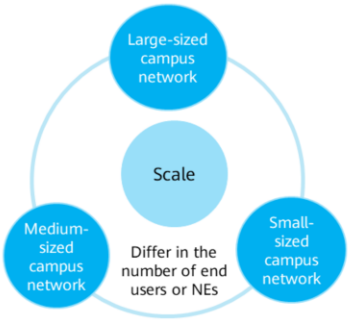
 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Large- and medium-sized campus network: is a network where more than 2000 terminals are present.
  - The network usually covers multiple buildings. Multiple aggregation layers are used to centralize networks of multiple buildings to the core layer.
  - The access layer needs to provide high-density wired interfaces and a large number of access points (APs). In addition, the access layer needs to provide differentiated access control based on terminal types.
  - To ensure service continuity, the campus network needs to support advanced capabilities such as the virtual network, free mobility, and security visibility.
  - A campus network is typically equipped with a data center, which provides service computing and storage capabilities for the campus network.
  - Typical examples: large-scale enterprise office networks, campus networks for higher education institutions, networks for large-scale government agencies, and airport networks.
- Small- and medium-sized campus network: is a network where no more than 2000 terminals are present.
  - Network layers are simple.
  - Due to the limitations of traditional network management technologies, the branches are often managed separately.
  - Typical examples: campus networks for primary/secondary schools, networks for

small-scale government agencies, networks for small-scale supermarkets, and networks for micro stores.

Campus Network Classification (1/3)

- Number of end users > 2000
- Number of NEs > 100

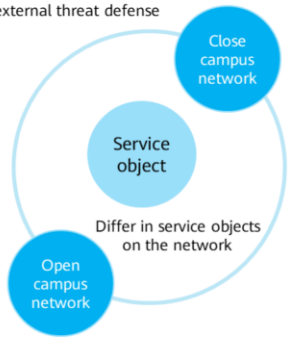


- Number of end users: 200 to 2000
- Number of NEs: 25 to 100

- Number of terminal users < 200
- Number of NEs < 25

Networks of different scales have different requirements and pain points.

- User: internal personnel only
- Security requirements: network access control and external threat defense



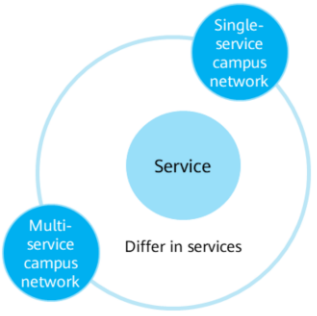
- Users: both internal and external personnel, such as the general public
- Security requirements: access control, identity identification, behavior control, and threat defense

A campus network usually has both close and open subnets.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Campus Network Classification (2/3)

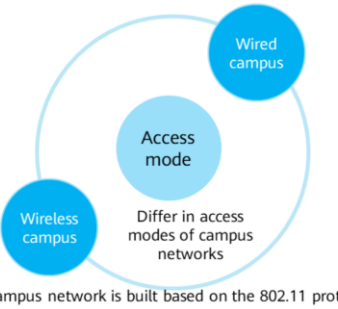
- Single service
- Simple network architecture



- Carries a large number of services and the network scale is large. Different services need to be isolated and guaranteed.
- Has a complex and virtualized network architecture.

How complex the network architecture is depends on the services running on the campus network.

- Each device connected to the network must be connected to the preset network interface using cables.
- The hierarchical architecture with clear logic facilitates troubleshooting.



- A wireless campus network is built based on the 802.11 protocol (Wi-Fi), and is also called WLAN.
- AP deployment and installation affect signal coverage. Interference and conflicts may occur and need to be optimized periodically.

Currently, most campus networks are hybrid networks of the wired and wireless networks.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

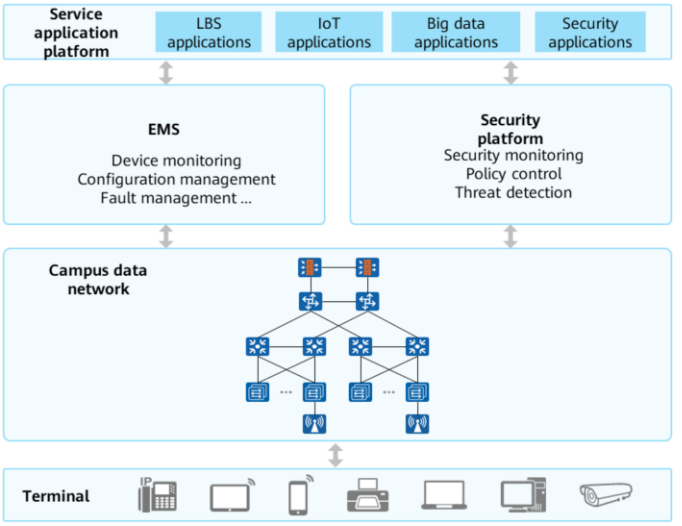
Campus Network Classification (3/3)

- To meet the requirements of different industries, the campus network architecture is designed based on the characteristics of the industry that the campus network serves. In this case, a campus network solution dedicated for a specific industry is launched.

Enterprise Campus Network	Education Campus Network	Government Campus Network	Business Campus Network
			
• Refers to an Ethernet-based enterprise office network. • Focuses on network reliability and advancement, to continuously improve employees' office experience and ensure the efficiency and quality of operation and production.	• Classification: campus networks for primary/secondary schools and for higher education institutions. • Campus networks for higher education institutions are complex, since they may require teaching research networks, student networks, and operational dormitory networks. • High requirements on manageability, security, and advancement.	• Refers to the internal network of government agencies. • Has high security requirements. Generally, the intranet and extranet are isolated to ensure the absolute security of confidential information.	• Applicable to shopping malls, supermarkets, hotels, and parks. • Used to serve consumers and also contains subnets for internal office work. • Provides Internet access services and builds a business intelligence system to improve user experience, reduce O&M costs, improve business efficiency, and realize value transfer.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

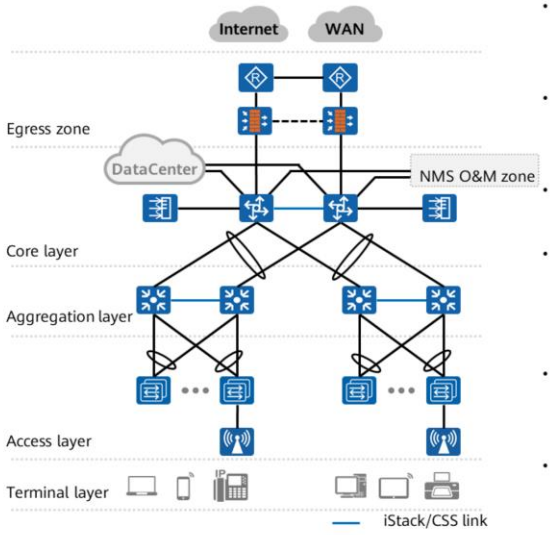
Typical Logical ARCH of a Campus Network



- Service applications can be developed based on the EMS to build a service application platform for the campus network.
- The next-generation EMS achieves automation, intelligence, and capability openness based on the traditional NMS and can interact with the security platform.
- The security platform can identify and defend against advanced persistent threats, display the network-wide security situation, and interwork with network devices to isolate and block threat flows after detecting threats.
- It is built based on Ethernet or WLAN technologies, and has various network devices deployed.
- It carries multiple services, such as office, IoT, and video conferencing.
- Wired and wireless terminals from both internal and external networks may access the campus network.
- Terminal admission and security management need to be considered.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

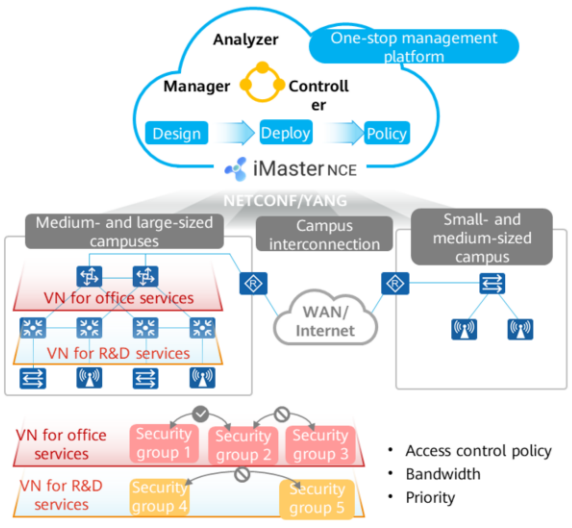
Typical Physical ARCH of a Campus Network



- **Core layer**: This layer is the backbone area of a campus network, which is the data switching core. It connects the various parts of the campus network, such as the data center, aggregation layer, and campus egress.
- **Aggregation layer**: This layer is a middle layer of a campus network, and is responsible for data aggregation or switching. Some basic network functions, such as routing, QoS, and security, are also provided at this layer.
- **Access layer**: As the edge of a campus network, this layer connects end users to the campus network.
- **Egress zone**: This zone is the edge between the campus internal network and the external network. Through this zone, internal users can access the public network and external users (including customers, partners, branches, and remote users) can access the internal network.
- **Data center area**: This area has servers and application systems deployed to provide data and application services for internal and external users of the enterprise.
- **Demilitarized zone (DMZ)**: This zone has public servers deployed to provide access services to external guests (non-employees). The access permission to this area is strictly controlled.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Huawei CloudCampus Solution



Fast network deployment, improving deployment efficiency by 600%

- Device plug-and-play: simplified device deployment, scenario-based navigation, and template-based configuration
- Simplified network deployment: network resource pooling, multi-purpose network, and automatic service provisioning

Fast service provisioning, improving user experience by 100%

- Free mobility: GUI-based policy configuration, allowing users to access the network anytime and anywhere with unchanged roaming permission and user experience
- Intelligent terminal identification: anti-spoofing for terminal access, with an intelligent terminal identification accuracy of over 95%
- Intelligent HQoS: application-based scheduling and shaping, and refined bandwidth management, ensuring service experience of key users.

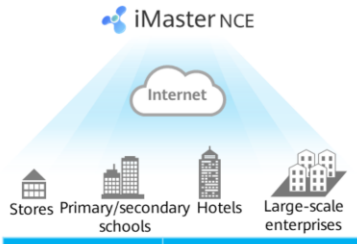
Fast intelligent O&M, improving network performance by over 50%

- Real-time experience visualization: telemetry-based network experience visualization at each moment, for each user, and in each area
- Precise fault analysis: proactively identifying 85% of typical network issues and providing suggestions, and comparing and analyzing real-time data to predict faults
- Intelligent network optimization: predictive optimization of wireless networks based on historical data, improving network-wide performance by over 50% (certified by Tolly).

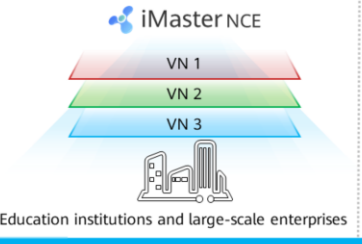
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Cover All Campus Network Scenarios

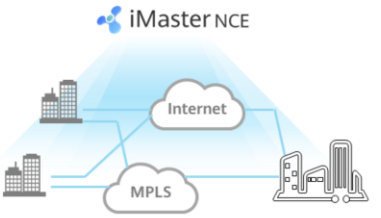
Simple-service campus



Multi-service campus

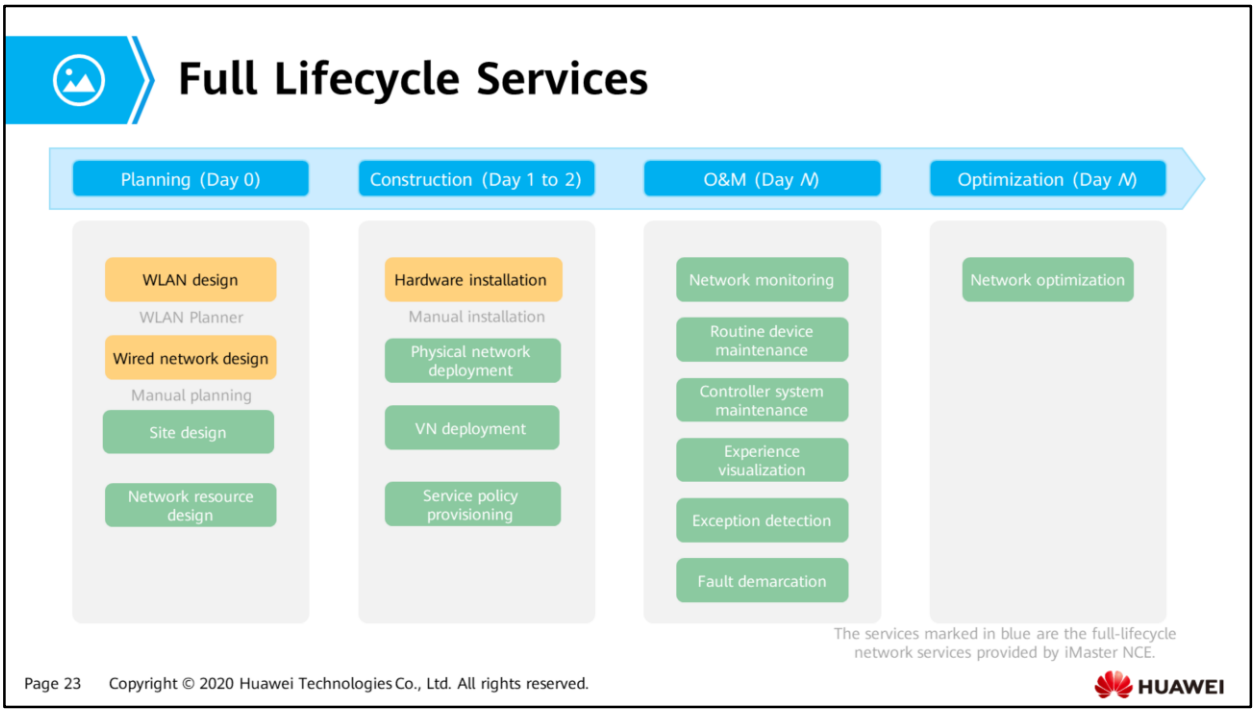


Multi-branch campus



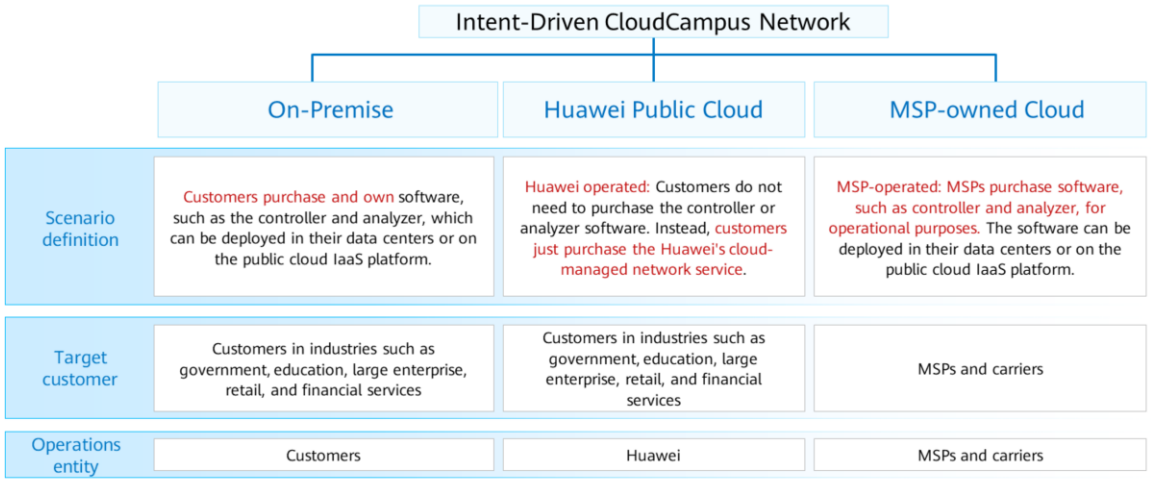
	Simple-Service Campus	Multi-Service Campus	Multi-Branch Campus
Network structure	Network covering a single campus, which is mainly used for Internet access	Complex network covering multiple areas and carrying various services	Wired and wireless networks for Internet access are available in the headquarters and branches. VPNs are built between the headquarters and branches for interconnection.
General requirements	Management of various network devices and user authentication	Management of various network devices, user authentication, and service isolation	Management of various network devices (such as APs, switches, firewalls, and ARs), user authentication, and VPN management
Typical scenario	Multi-branch campuses and small-sized enterprises, such as hotels and primary/secondary schools	Education institutions, government agencies, and large-scale enterprises	Large-scale enterprises and financial branches

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

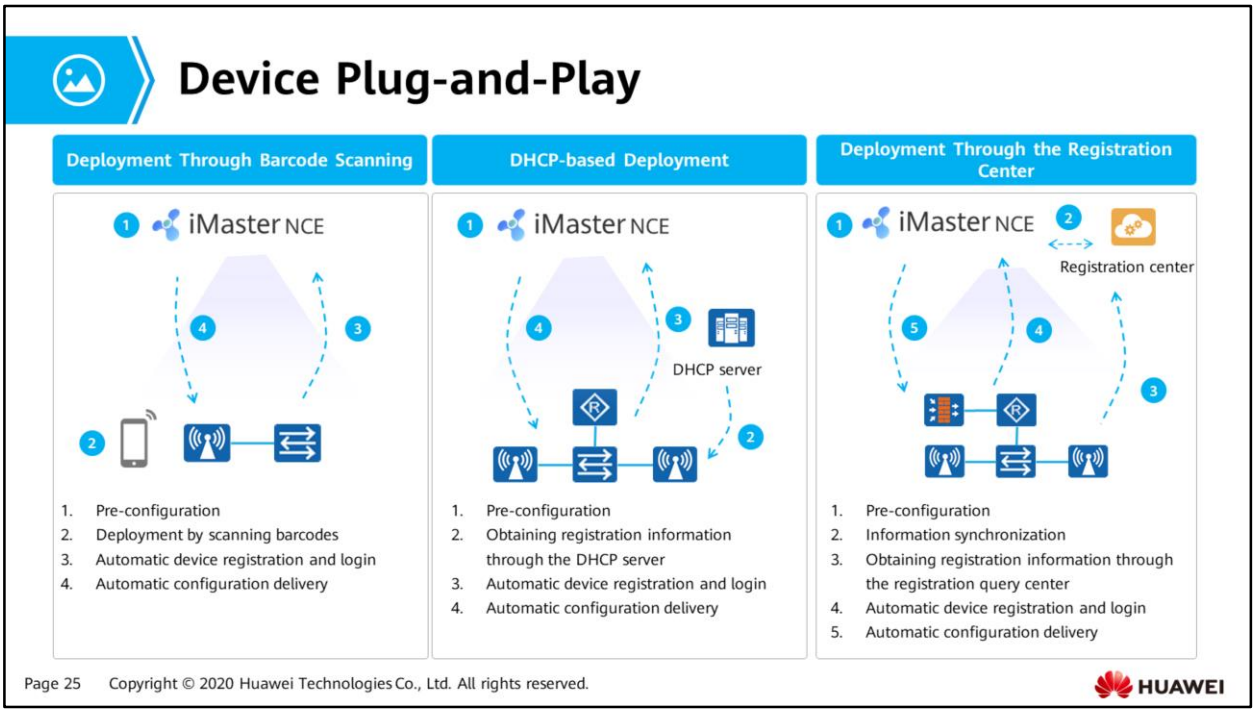


- Full Lifecycle: Planning, Construction, O&M, and Optimization

Three CloudCampus Deployment Scenarios

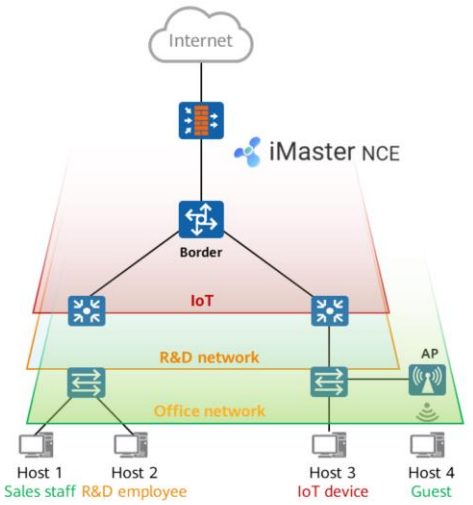


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



- The device plug-and-play feature is applicable to multiple deployment modes, including deployment through barcode scanning, DHCP-based deployment, and deployment through the registration center.
- Registration center: is one of the main components of the Huawei CloudCampus Solution for query of the device management mode and the home cloud management platform. According to the query result, a device determines whether to change to the cloud-based management mode and which cloud management platform it must register with. For example, all the APs that support cloud-based management will have the Huawei device registration query center's URL (register.naas.huawei.com) and port number (10020) pre-configured.

One Network for Multiple Purposes

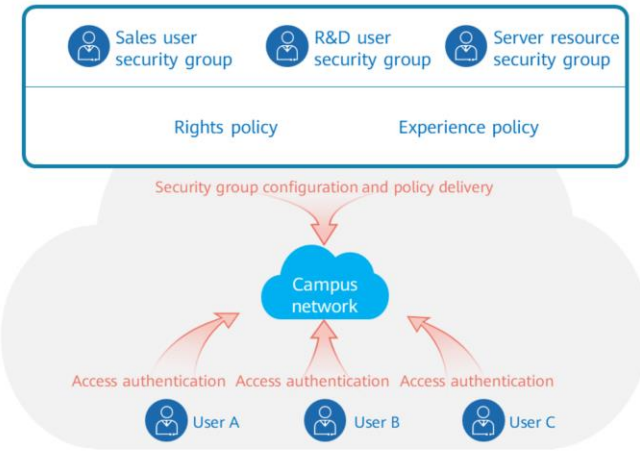


- By leveraging the virtualization technology, multiple VNs are created on a physical campus network, each deployed with a unique service, such as office, R&D, and IoT.
- iMaster NCE manages network-wide devices in a centralized manner. Administrators can configure networks on the iMaster NCE web UI.
- iMaster NCE translates service configuration intents of an administrator into configuration commands and delivers them to devices through NETCONF to implement network autonomous driving.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Free Mobility

- Free mobility: Enables users to have consistent network rights and security policies regardless of their locations and IP addresses.



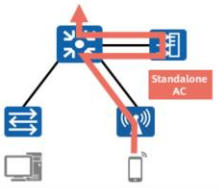
1. Configure security groups. A security group is a group of users for which the same security policy is used.
2. Define security group-based permission control policies and user experience policies and deliver the policies to network devices.
4. After user traffic enters a network, network devices enforce policies based on the source and destination security groups of the traffic.
3. A security group is authorized to a user after the user passes access authentication.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- Free Mobility: Policy Management Based on Security Groups

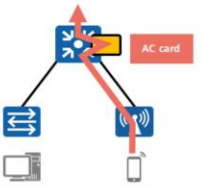
Wired and Wireless Convergence

WLAN with a Standalone AC



- This mode poses a wireless traffic bottleneck and increases faulty nodes.
- Wired and wireless management is independent.
- Wired and wireless authentication points are separated.

WLAN with an AC Card



- An AC card is installed on a switch to provide AC functions.
- Hardware-level convergence only

Wired and wireless authentication points are separately deployed, policy control is decentralized, and traffic forwarding is separated, resulting in difficulties in fault troubleshooting and network management.

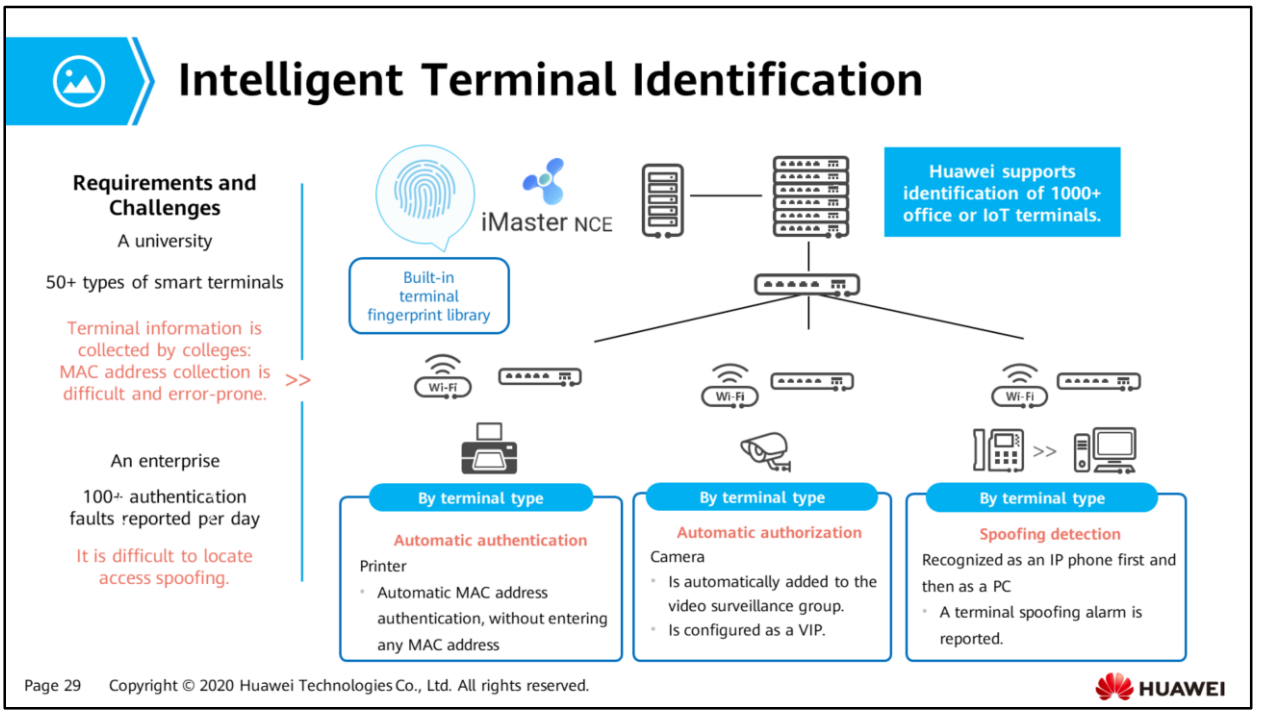
Wired and Wireless Convergence (Integrated AC)



The switch integrates the AC function, eliminating wireless traffic forwarding bottlenecks and reducing fault nodes. Wired and wireless devices are centrally managed.

- Unified management and converged forwarding of wired and wireless services
- Converged management for wired and wireless users and gateway convergence
- Converged authentication points for wired and wireless access
- Unified wired and wireless policy enforcement

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



- Intelligent Terminal Identification, Ensuring Secure Access

AI-Powered Intelligent O&M

As-Is: device-centric network management

Traditional NMS

NMS
- Topology management
- Performance management
- Alarm management
- Configuration management

SNMP
Network data collection within minutes



- **Device-centric O&M method**: User experience cannot be detected.
- **Fault-triggered responses**: Potential faults cannot be identified.
- Rely on professional engineers to **locate faults onsite**

To-Be: user experience-centric AI-powered intelligent O&M

iMaster NCE CampusInsight

Telemetry
Network data collection within seconds

Intelligent network analyzer
- Visualized experience management
- User journey playback
- Potential fault identification
- Root cause identification
- Predictive network optimization



Experience visualization: telemetry-based data collection within seconds, visualized experience of each user and each application in real time

Potential fault identification and root cause location within minutes
- Identify potential faults based on dynamic baselines and big data correlation analysis.
- Accurately locate root causes of faults through KPI association analysis and protocol tracing.

Predictive network optimization: AI is used to intelligently analyze the AP load trend and implement predictive optimization on the wireless network.

The efficiency is improved by using algorithms. With scenario-based continuous learning and expert experience, intelligent O&M frees O&M personnel from complex alarms and noises, making O&M more automated and intelligent.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

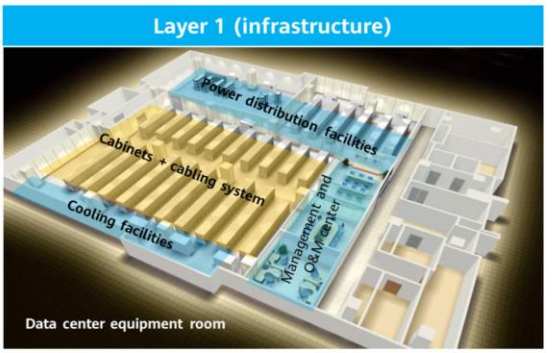
Contents

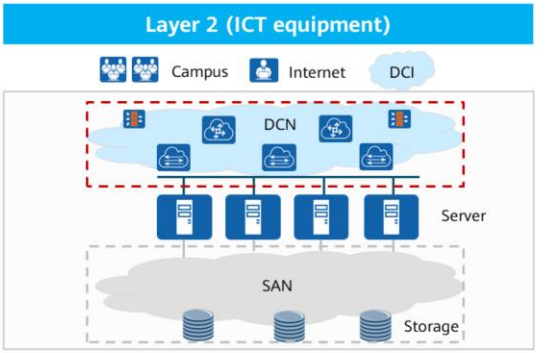
1. SDN Overview
2. **Introduction to Huawei iMaster NCE**
   - iMaster NCE-Campus
   - **iMaster NCE-Fabric**

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Introduction to a Data Center

- A data center is an entire suit of facilities, including buildings. It includes the computer system and the devices related to the computer system (for example, communication and storage devices), as well as redundant data communication connections, environmental control devices, monitoring devices, and various safety devices.
- A data center uses a layered architecture. In most cases, L1 refers to the data center's infrastructure, and Layer 2 refers to the data center's ICT equipment.





Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- The concepts, topology, and architecture of a data center in this course are based on layer 2 or above.

Devices in a Data Center

- Devices in a data center are classified into computing, storage, and network devices (excluding data center infrastructure).

1 Computing device

Server



- FusionServer server
- TaiShan server
- Atlas AI computing platform

2 Storage device

Storage array

Cloud storage device



- OceanStor Dorado all-flash storage
- OceanStor V5 hybrid-flash storage
- FusionStorage cloud storage

3 Network device

Router

Switch

Firewall

Load balancer

IDS/IPS

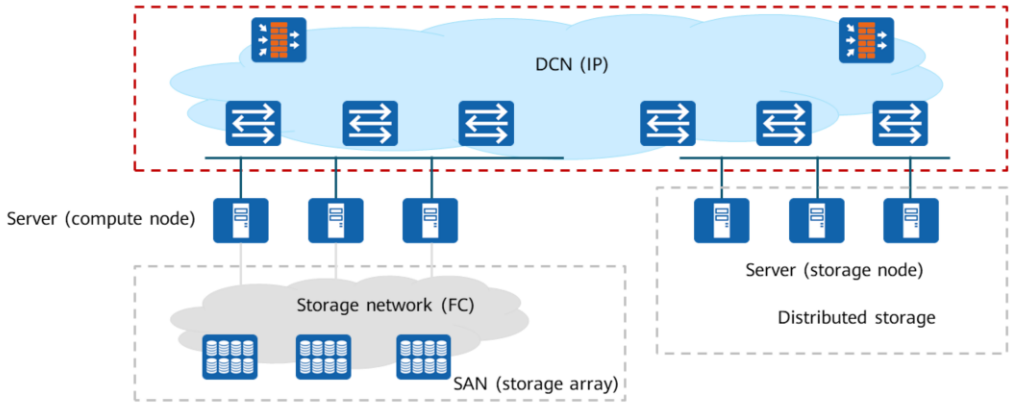
• • •

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Huawei devices are used as an example here.

Typical Data Center Networking

- A typical data center networking consists of a data center network (DCN), a storage area network (SAN), and servers.

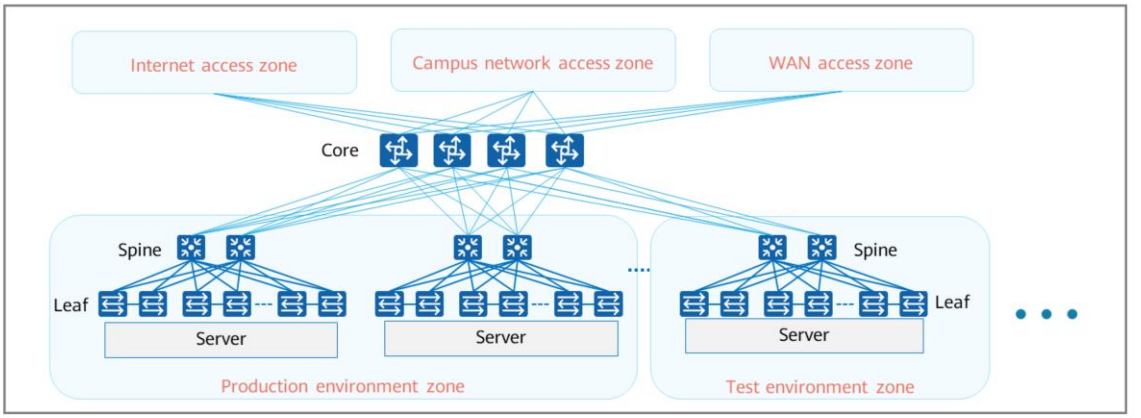


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- DCN: enables communication between computing instances in a DC and between computing units and external egresses.
- SAN: consists of storage arrays and Fiber Channel (FC) switches to provide block storage. The storage network that uses the FC protocol is called FC SAN, and the storage network that uses the IP protocol is called IP SAN.
- Distributed storage: The deployment mode of distributed storage is different from that of storage array. Data is stored on multiple independent servers (storage nodes). It is also used as cloud storage.
- Server (compute node): provides computing services.

DCN

- DCN: enables communication between units in a data center and between units and external egresses.
- A data center network consists of various network devices and is divided into different areas based on services.

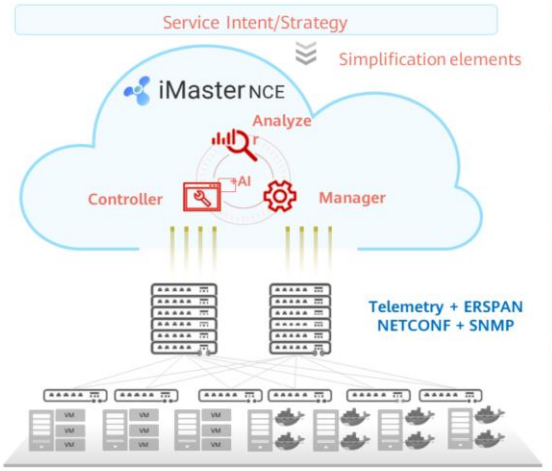


 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Data center networks can be flexibly divided into different zones based on enterprise or industry requirements. For example, a financial data center network can be divided into production zone 1, production zone 2, test zone 1, test zone 2, big data zone, operation management zone, and other zones.
- In this example, the data center data network is divided in to the following zones:
  - Internet access zone: transmits Internet access traffic from users.
  - Campus network access zone: transmits traffic from users on enterprise campus networks.
  - WAN access zone: connects to the enterprise-built WAN. Data centers and campuses in other cities may reside on the other side of this zone.
  - Production environment zone: connects to the production environment.
  - Test environment area: connects to the test environment.

Huawei CloudFabric Solution

- iMaster NCE-Fabric provides full-lifecycle services covering planning, construction, O&M, and optimization.



Integrated planning and construction
- The planning tool interworks with iMaster NCE-Fabric to implement integrated planning and construction.
- Zero Touch Provisioning (ZTP)

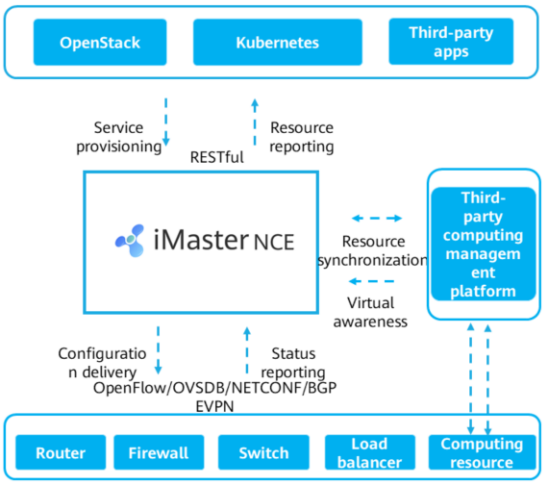
Simplified deployment
- Service intent self-understanding and conversion
- Network change simulation and evaluation, eliminating human errors

Intelligent O&M
- Rapid fault detection and location based on knowledge graph and expert experiences
- Fast fault rectification based on expert experiences and simulation analysis

Real-time optimization
- Local traffic inference and online model training and optimization based on AI Fabric
- User behavior prediction and resource optimization suggestions

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Open Architecture, Building a DCN Ecosystem



Northbound ecosystem

- Provides standard Neutron/GBP RESTful APIs to enable abstraction of fabric configuration.
- Interconnects with standard OpenStack and Kubernetes.

Eastbound and westbound ecosystem

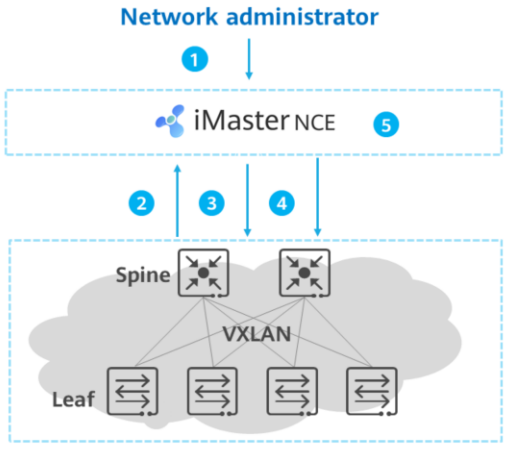
- Supports interconnection with third-party computing management platforms and works with compute resources to implement on-demand VM on-boarding and elastic scaling.

Southbound ecosystem

- Provides standard APIs for service provisioning and device configuration.
- Provides southbound plug-ins, through which third-party devices can interoperate with Huawei devices.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Simplified ZTP Deployment

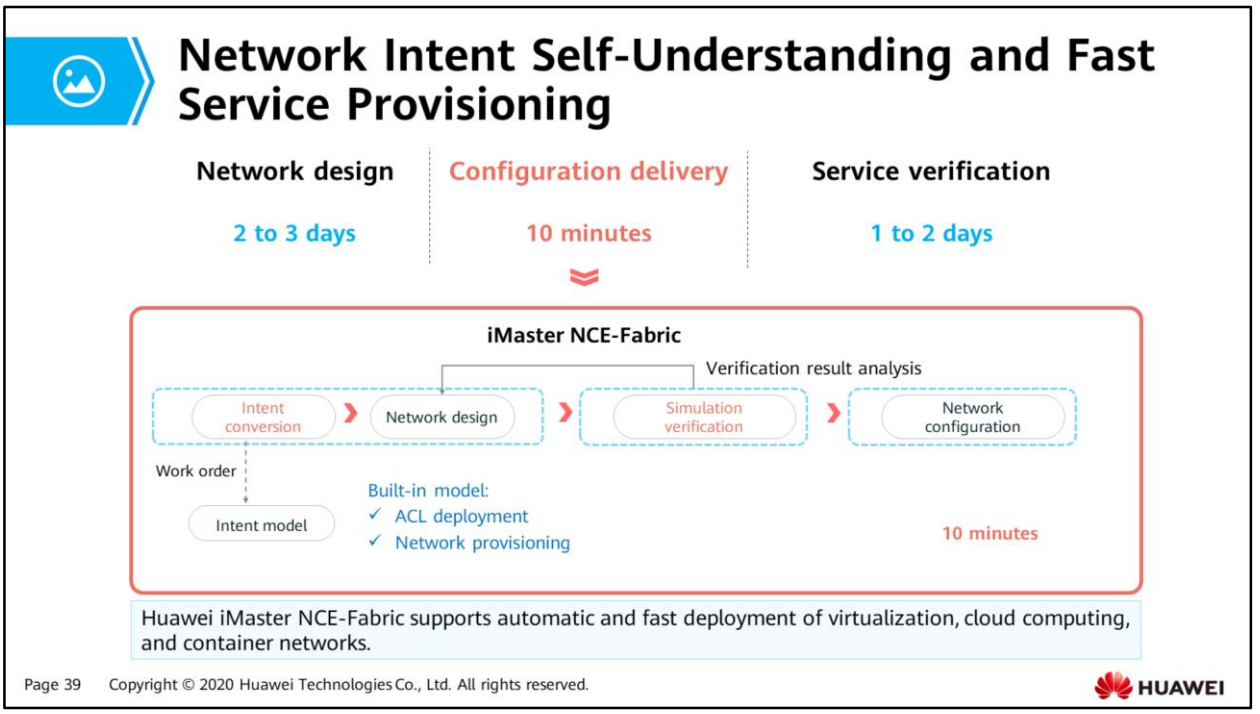


ZTP process:

1. The network administrator starts the ZTP task on the iMaster NCE-Fabric GUI.
2. A device automatically obtains an IP address to access iMaster NCE-Fabric.
3. iMaster NCE-Fabric determines the device role (spine or leaf node), delivers configurations such as the management IP address, SNMP configuration, and NETCONF configuration to the device, and manages it through the management IP address.
4. iMaster NCE-Fabric globally delivers interconnection configurations as well as OSPF or BGP configurations.
5. The device goes online successfully, and the administrator checks network-wide information on iMaster NCE-Fabric.

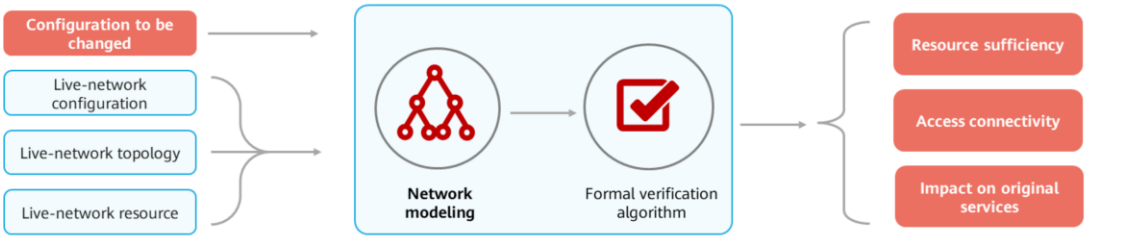
Note: The DC uses the spine-leaf architecture.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



- iMaster NCE-Fabric can connect to customer IT systems to match an intent model for customer intents and deliver configurations to devices through NETCONF to implement fast service deployment.
- iMaster NCE-Fabric can interconnect with the mainstream cloud platform OpenStack, virtualization platform vCenter and System Center, and the container orchestration platform Kubernetes.

Network Change Simulation and Change Risk Prediction



Data collection/upload

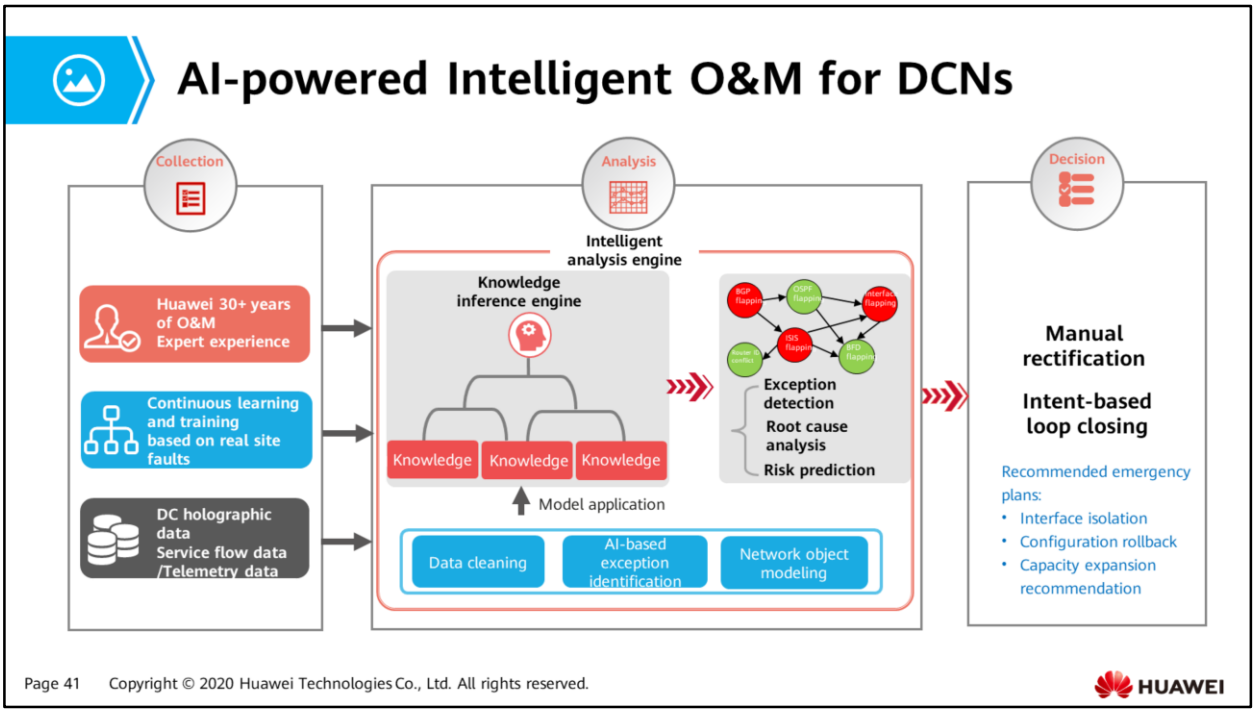
Modeling and computing

- Establish physical, logical, and application network models.
- Use the formal verification method to solve the problem.

Verification result

- Check whether resources on the live network are sufficient and whether the network is connected.
- Analyze and display the impact of changes on original services.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



- iMaster NCE-FabricInsight provides AI-powered intelligent O&M capabilities for data centers.

Quiz

1. (Multiple) Which of the following deployment scenarios are supported by CloudCampus? ( )

   A. On-premise scenario

   B. Huawei public cloud scenario

   C. MSP-built cloud scenario

   D. Campus scenario

2. (Single) Which of the following is the latest product in Huawei CloudFabric Solution? ( )

   A. iMaster NCE-Campus

   B. iMaster NCE-Fabric

   C. Agile Controller-DCN

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

1. ABC

2. B

Summary

- SDN innovates the network architecture. It uses a controller to build a more open, flexible, and simplified network.
- Huawei IDN solution covers various network scenarios, including enterprise campus networks, data center networks, SD-WAN, enterprise WAN IP networks, and enterprise transport networks.
- This course introduces the CloudCampus and CloudFabric solutions. For more information about Huawei IDN solutions, see HCIP-Datacom certification materials in each domain.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

More Information

- For more information about Huawei IDN solutions, see:

  https://e.huawei.com/en/products/network-management-and-analysis-software.

- For more information about Huawei solution certifications, see HCIP-Datacom courses of each domain.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Recommendations

- Huawei Talent: https://e.huawei.com/en/talent/#/home

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.


Thank You
www.huawei.com



RESTful Principles and Practices

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Foreword

- In the network communications field, network openness is an inevitable trend of industry development. In a fully connected, intelligent world, opening more interfaces means quick connections with other networks. Network openness not only transforms networks, but also further segments the industry chain, bringing new opportunities for industry development. Nowadays, REST is selected and preferred by a growing number of companies, and likewise, RESTful application programming interfaces (APIs) are growing in popularity across the entire network.
- This course describes the concepts and functions of REST and RESTful, working principles of HTTP, and typical practices in invoking RESTful APIs.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Objectives

- Upon completion of this course, you will be able to:
  - Have full knowledge of the background, concepts, and highlights of REST.
  - Understand the relationship between REST, RESTful, and RESTCONF.
  - Master the HTTP packet format and field descriptions.
  - Understand the relationship between HTTP, HTTPS, and HTTP/2.
  - Grasp basic capabilities for invoking RESTful APIs.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

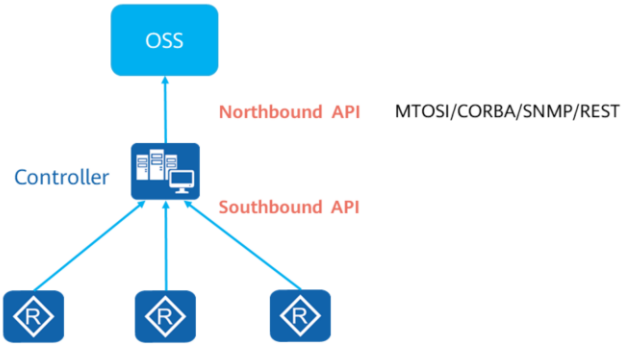
Contents

1. **REST and RESTful**
   - **Background of REST**
   - Overview of REST
2. Working Principle of HTTP
3. Practices in Invoking RESTful APIs

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Basic Concepts of Southbound and Northbound

- Southbound and northbound are relative concepts. Generally, the interfaces provided by an upper-layer system for a lower-layer system are called southbound APIs, and the interfaces provided by a lower-layer system for an upper-layer system are called northbound APIs.
- APIs provide the interaction function to implement data transmission.

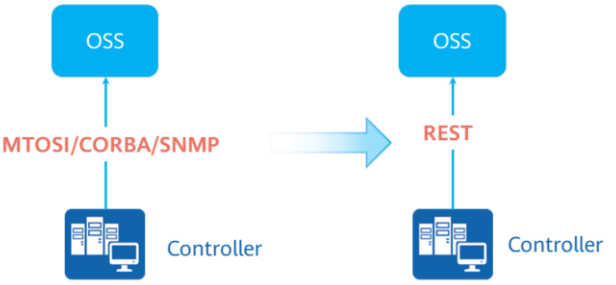


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- The operation support system (OSS) is a necessary support platform for telecom services.
- MTOSI: Multi-Technology Operations System Interface
- Common Object Request Broker Architecture (CORBA) is a standard object-oriented application program system specification formulated by Object Management Group (OMG).

Development of Northbound APIs

- There are many northbound API standards, for example, MTOSI and CORBA. Vendors' controllers need to open northbound APIs compliant with specific standards to easily interconnect with carriers' integrated network management platform.
- Devices may use diverse northbound APIs such as SNMP, CORBA, and REST. Nowadays, these APIs are gradually unified into the REST style.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

API Evolution: Unified URI Naming Conventions

- REST standardizes universal resource identifier (URI) naming conventions. Resource-oriented URI names are easy to understand.
  - A URI identifies the location of a resource.

Without REST: URIs were randomly named.	With REST: URIs are independent and methods are clear.
URIs without constraints on naming conventions: • Reserving a meeting room: /reserve/meetingroom/B25R • Canceling a reserved meeting room: /cancel/meetingroom/B25R • Querying the meeting room status: /meetingroom/B25R?method=query	URIs with naming conventions of REST: • Reserving a meeting room: /meetingroom/B25R, with the calling method being POST • Canceling a reserved meeting room: /meetingroom/B25R, with the calling method being DELETE • Querying the meeting room status: /meetingroom/B25R, with the calling method being GET

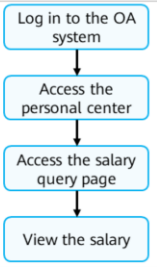
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

API Evolution: Stateless Design

- REST uses a stateless design to enhance system scalability.
  - Stateful: The server needs to save and maintain the state information of previous requests. Each subsequent state depends on the previous state.
  - Stateless: The server sends back the same response for invocation of the same request, function, or method, without depending on other requests. The server does not need to maintain state information, facilitating expansion. At least one URI is available to locate a resource.

Without REST: both stateful and stateless used

A client can obtain the required information only after multiple requests. The figure on the right shows an example of logging in to the office automation (OA) system to view the salary. The process is complex, affecting user experience.
The required information cannot be directly located using only one URI.



With REST: stateless

In the REST architecture, the required result can be obtained with only one request. For example, the salary can be queried using the following URIs:

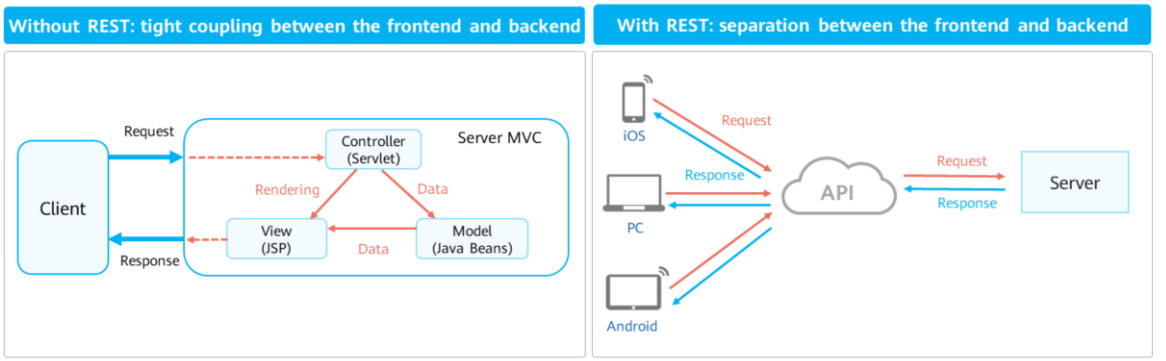
- Salary of Tom: http://oa.company.com/salary/tom
- Salary of Jerry: http://oa.company.com/salary/jerry

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- Stateful request: A server generally needs to save and maintain the status information of previous requests. Each request can use information about the previous requests by default.
- Stateless request: The processing result on the server must be based on the information carried in the same request.

API Evolution: Separation Between Frontend and Backend (1)

- REST provides good cross-platform compatibility and separates the frontend from the backend.
  - The frontend mainly involves the pages that can be viewed by users.
  - The backend mainly involves the logical functions related to data processing.

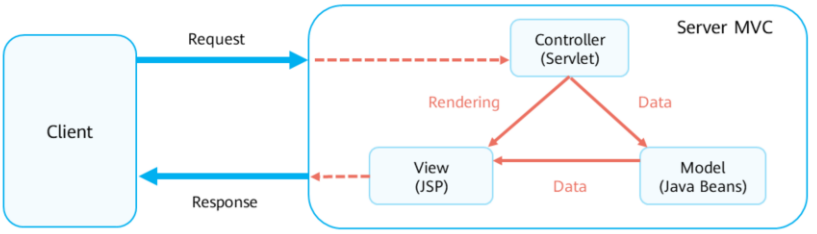


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- An API is a set of predefined functions or methods for connecting different components of a software system. It is a set of routines that can be accessed by applications and developers based on software or hardware without having to access the source code or understand the details of the internal working mechanism. For example, if a computer needs to invoke information in a mobile phone, we simply need to connect the computer and the mobile phone by using a data cable. In this example, the interfaces on the computer and mobile phone at both ends of the data cable.

Interface Evolution: Separation Between Frontend and Backend (2)

- In early web projects, rendering is generally performed on the server. A client directly obtains and interacts with the rendered view on the server.



Advantage: The entire page is sent back with only one request, featuring a high loading speed.

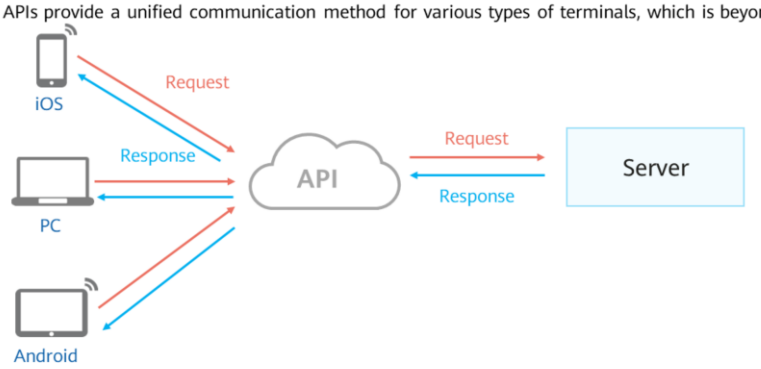
Disadvantage: Components are closely associated with each other. If a view is modified on the frontend, the logical modules of the backend must be modified accordingly.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- Rendering refers to the process of transforming views such as HTML into visual images that can be seen by human eyes.
- For web applications of early days, a view is a graphical user interface (GUI) composed of HTML elements. For today's web applications, the GUI incorporates new elements such as Adobe Flash, XHTML, XML/XSL, and WML.
- The Model-View-Controller (MVC) is a pattern for creating web applications and consists of three parts: controller, model, and view.
  - The controller is responsible for processing user interaction in an application. Generally, the controller reads data from the view, controls user input, and sends data to the model.
  - The model is responsible for processing the data logic of an application. Generally, the model stores and reads data from the database.
  - The view is responsible for processing data display. Generally, a view is created based on model data.

API Evolution: Separation Between Frontend and Backend (3)

- REST develops rapidly with the emergence of the mobile Internet. The server cannot render a GUI for each type of terminal. Therefore, rendering on the server cannot meet the requirements of mobile Internet. To maintain compatibility with various types of terminals, REST emerges and becomes pervasive.
- With REST, the server only processes data, and the client is responsible for GUI rendering. In this way, the frontend and backend are separated.
- REST-compliant RESTful APIs provide a unified communication method for various types of terminals, which is beyond the reach of traditional APIs.



 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- An API is a set of predefined functions or methods for connecting different components of a software system. It is a set of routines that can be accessed by applications and developers based on software or hardware without having to access the source code or understand the details of the internal working mechanism.
- Separation between the frontend and backend has become an industry standard for the Internet projects in the industry. It lays a solid foundation for the large-scale distributed architecture, elastic computing architecture, microservice architecture, and multi-terminal services (such as browsers, vehicle-mounted terminals, Android, and iOS). The key to separation between the frontend and backend is that the frontend page invokes the RESTful API of the backend for data interaction.

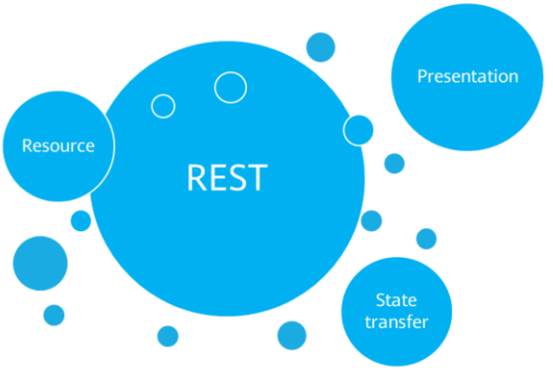
Contents

1. **REST and RESTful**
   - Background of REST
   - **Overview of REST**
2. Working Principle of HTTP
3. Practices in Invoking RESTful APIs

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Overview of REST

- Representational State Transfer (REST) was proposed by Roy Thomas Fielding, HTTP's leading designer, in his doctoral dissertation in 2000. In short, REST is a design style.



REST uses the client-server interaction framework and involves the following concepts:

- Resource
- Presentation
- State transfer

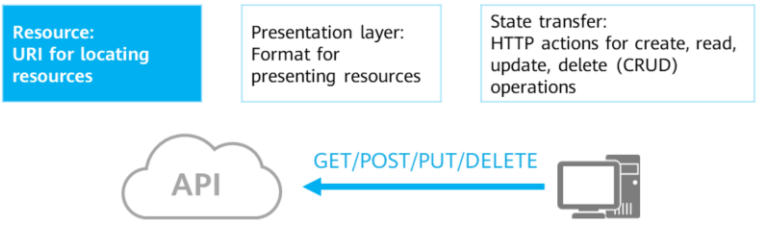
 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- Abstract of Roy's doctoral dissertation *Architectural Styles and the Design of Network-based Software Architectures*:
  This dissertation explores a junction on the frontiers of two research disciplines in computer science: software and networking. Software research has long been concerned with the categorization of software designs and the development of design methodologies, but has rarely been able to objectively evaluate the impact of various design choices on system behavior. Networking research, in contrast, is focused on the details of generic communication behavior between systems and improving the performance of particular communication techniques, often ignoring the fact that changing the interaction style of an application can have more impact on performance than the communication protocols used for that interaction. My work is motivated by the desire to understand and evaluate the architectural design of network-based application software through principled use of architectural constraints, thereby obtaining the functional, performance, and social properties desired of an architecture.

REST Concepts — Resource

- A resource is an information entity. All things on the network can be abstracted as resources. Each resource has a URI. For example, ports, network elements (NEs), boards, and equipment rooms are resources.
- Resources and operations are the core of REST. The URI is used to locate resources, and HTTP actions (GET, POST, PUT, and DELETE) are used to describe operations.

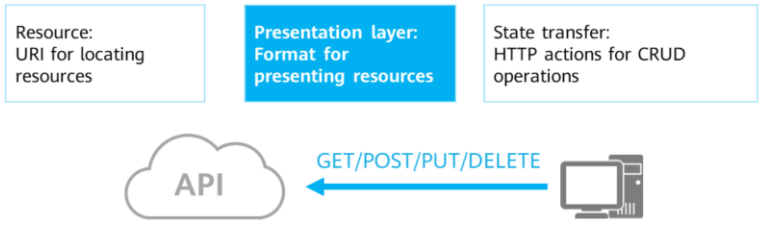


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- REST is short for Representational State Transfer, in which the main entity — resource — is not presented.

REST Concepts — Presentation Layer

- Representation is the way a resource is presented.
- For example, a text can be presented in TXT, HTML, XML, JSON, or even binary format; an image can be presented in JPG, PNG, or other formats.

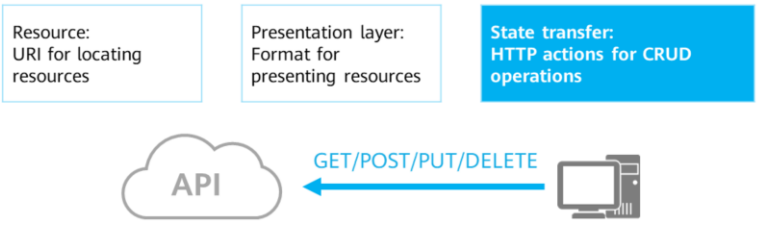


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- A URI represents only a resource entity but not its presentation.

REST Concepts — State Transfer

- Accessing a URI represents an interaction between the client and the server. In this process, data and status changes are involved.
- Hypertext Transfer Protocol (HTTP) is a stateless protocol. This means that all states are stored on the server side, not on the client side. Therefore, the client performs operations on the server by using a certain method, so that a state transfer occurs on the server. Such a state transfer depends on the presentation layer, so it is called "Representational State Transfer".
- The method used by the client is HTTP.

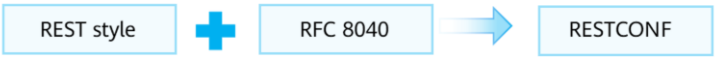


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

RESTful and RESTCONF

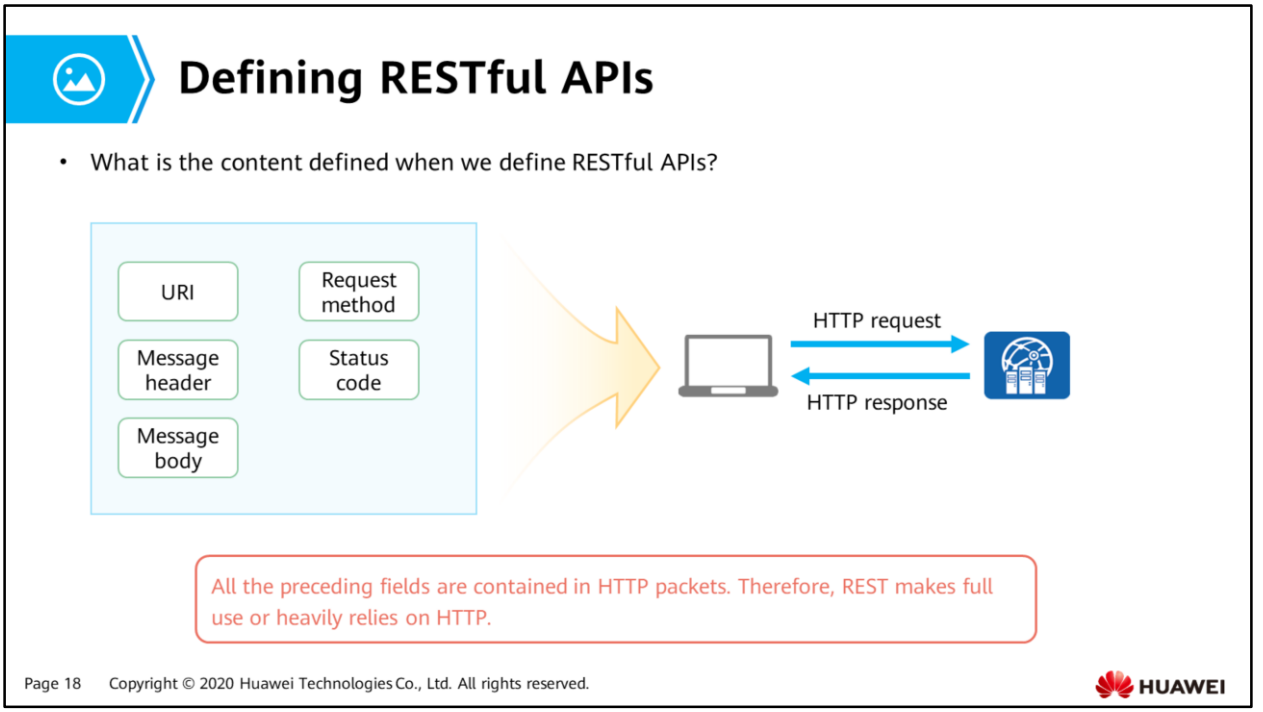
- RESTful APIs comply with the REST design style. There are no mandatory requirements on RESTful APIs, and therefore RESTful APIs can be defined freely.
- RESTCONF APIs also comply with the REST design style. Unlike RESTful APIs, RESTCONF APIs must comply with RFC 8040 defined by Internet Engineering Task Force (IETF). RFC 8040 defines RESTCONF APIs and their specifications. RESTCONF APIs are based on HTTP and are used to access data defined in Yet Another Next Generation (YANG). RESTCONF allows web applications to access configuration data, status data, and event notifications of network devices in a modular and scalable manner.

REST + Configuration

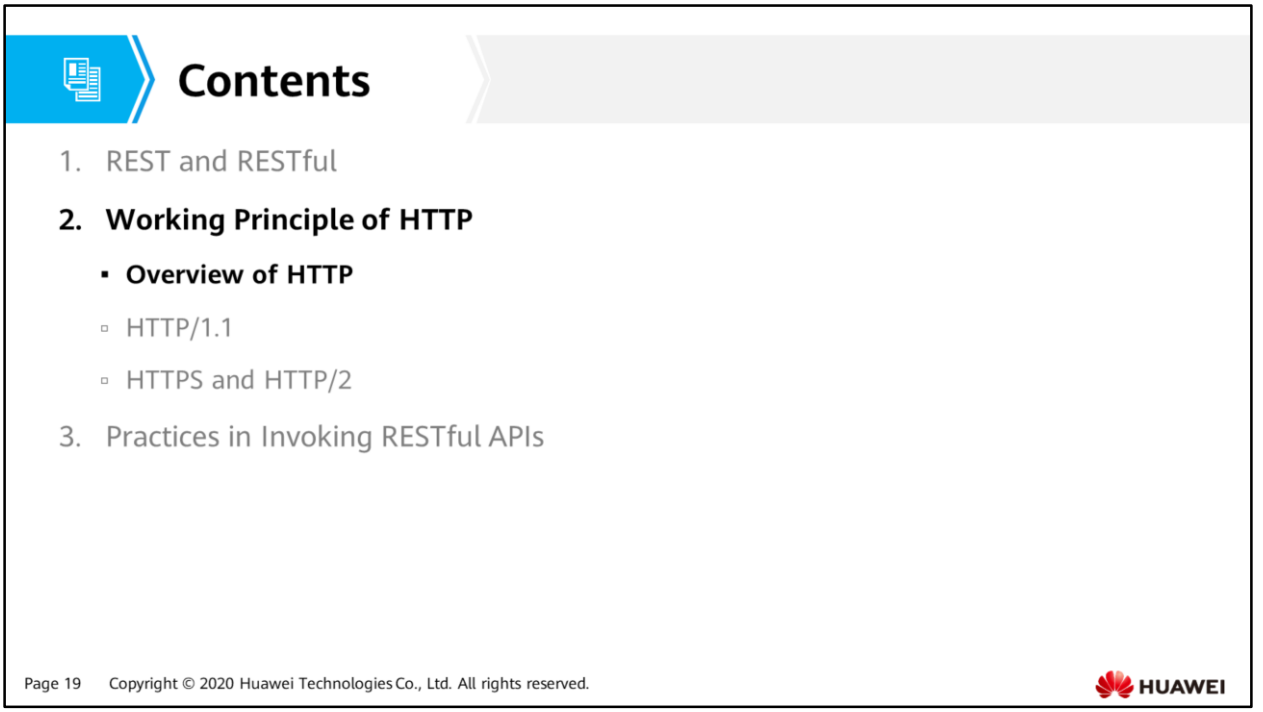


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- YANG defines the storage content and configuration of data.



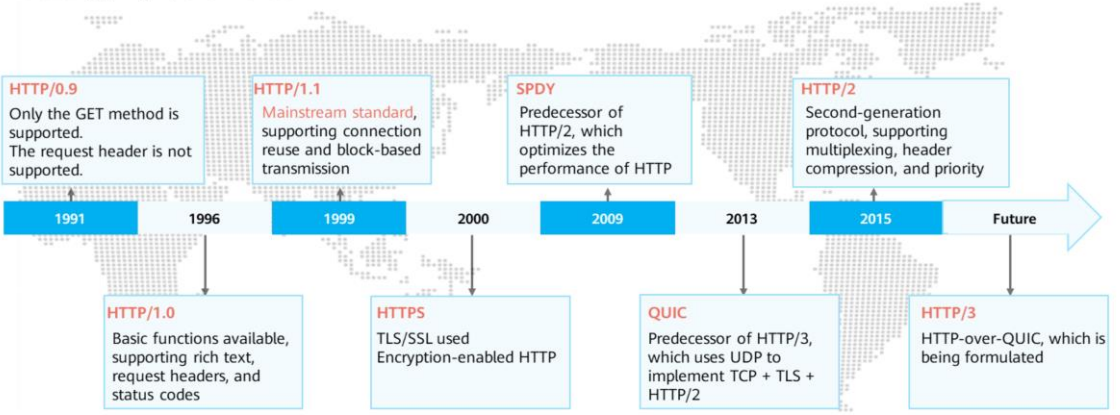
- Relationship between the URI and URL
  The URL is a subset of the URI. The former must be an absolute path, while the latter can be an absolute path or a relative path. For example, http://127.0.01:8080/AppName/rest/product/1 is a URL, and AppName/rest/product/1 is a URI.



- As mentioned earlier, REST makes full use or heavily relies on HTTP. Next, we will move on to HTTP.

Development of HTTP

- HTTP was proposed by Tim Berners-Lee in 1990. After the WWW Alliance was established, the IETF working group further optimized and released the HTTP protocol. Over the evolution from HTTP/0.9 to HTTP/3, HTTP offers increasingly high performance.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- SPeeDY (SPDY) is a TCP-based application-layer protocol developed by Google. Its objective is to optimize the performance of HTTP and shorten the loading time of web pages and improve security by using technologies such as compression, multiplexing, and priority. The core idea of SPDY is to minimize the number of TCP connections. SPDY is an enhancement to HTTP, instead of a protocol for replacing HTTP.
- Quick UDP Internet Connection (QUIC) is a UDP-based low-delay Internet transport layer protocol developed by Google. In November 2016, the IETF convened the first meeting of the QUIC working group, which attracted wide attention from the industry. This means that QUIC starts its standardization process and will become a next-generation transport layer protocol.

Overview of HTTP

- Hypertext Transfer Protocol (HTTP) is an application layer protocol for distributed, collaborative, and hypermedia information systems. HTTP has been used by the World Wide Web since its inception.
- It uses TCP/IP to transmit data and uses TCP port 80 by default. Websites starting with http:// are standard HTTP services.

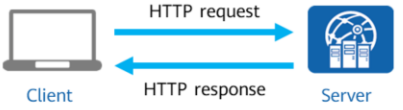


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- The data transmitted using HTTP can be HTML, images, texts, and so on.

Working Principle of HTTP

- HTTP is based on the client/server (C/S) architecture. The HTTP request and response process is as follows:
  - A client establishes a TCP connection with the server.
  - The client sends an HTTP request, which consists of a request line, request header, empty line, and request data.
  - The server receives the request and sends back an HTTP response, which consists of a status line, response header, empty line, and response body.
  - The TCP connection is released.
  - The client parses the status line, response header, and response body in sequence and displays the response packet. If the body contains HTML data, the client formats the data based on the HTML syntax and displays the data.

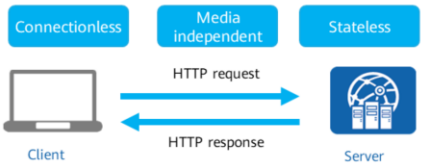


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- An HTTP client is usually a browser. A web server can be an Apache server or an Internet Information Services (IIS) server.
- When a TCP connection is released, if the value of the **Connection** field in the packet header is **close**, the server proactively closes the TCP connection, and the client passively closes and releases the TCP connection. If the value of **Connection** is **keepalive**, the connection lasts for a period of time and can continue to receive requests.

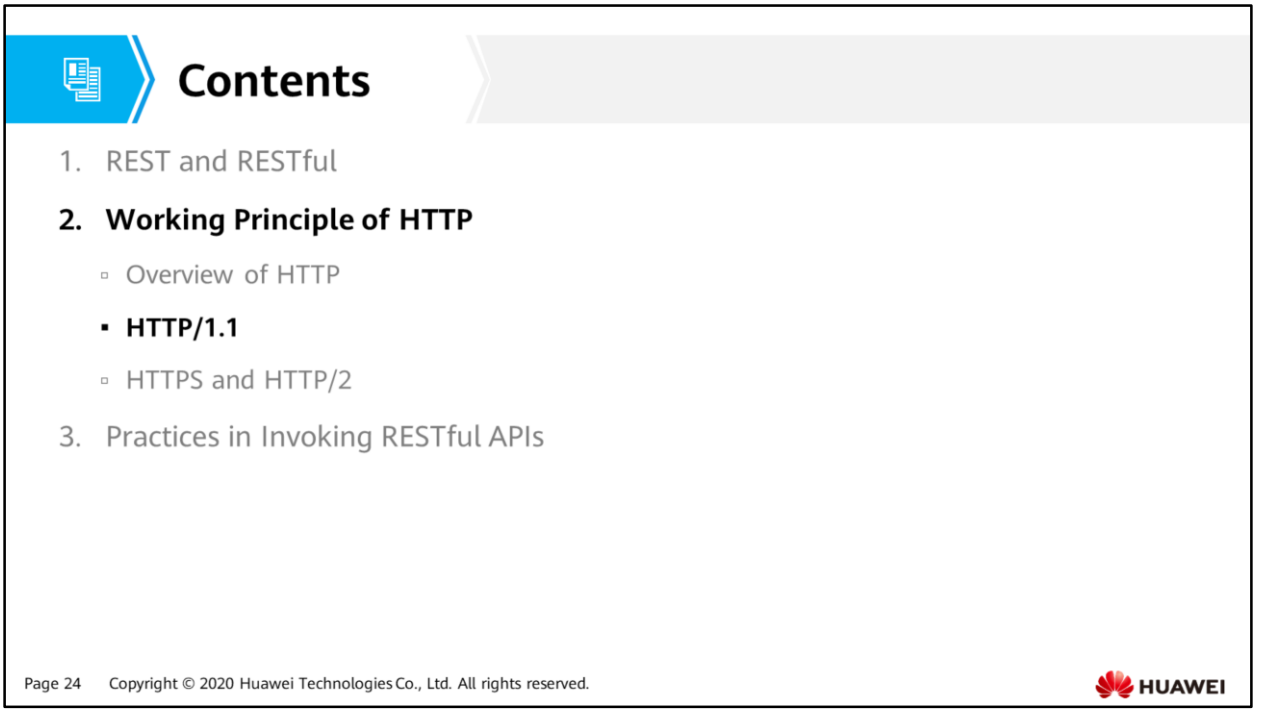
Features of HTTP

- HTTP has the following features:
  - Connectionless: Only one request is processed for each connection. After processing the client's request, the server disconnects from the client.
  - Media independent: Any type of data can be sent by HTTP as long as both the client and the server know how to handle the data content. It is required for the client as well as the server to specify the content type using appropriate MIME-type.
  - Stateless: The server and client are aware of each other only during a current request. Afterwards, both of them forget about each other. This facilitates quick processing of a large number of transactions and enhances the protocol scalability.



 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

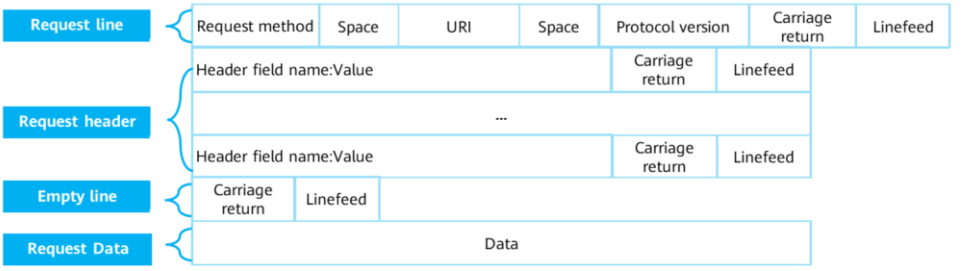
- The browser differentiates the displayed content such as HTML, XML, GIF, and flash based on MIME-type.
- Advantages of the connectionless feature: This mode saves the transmission time and improves the concurrent performance. No persistent connection is established. Instead, one response is made to each request. However, if a connection is repeatedly established and torn down, the efficiency is affected. In HTTP/1.1, a TCP connection is maintained between the browser and the server for a period of time and will not be disconnected immediately after a request ends.
- Stateless means that, if the processing of subsequent packets requires the previously exchanged information, the information must be retransmitted. Although HTTP/1.1 is a stateless protocol, cookies are introduced to implement the function of maintaining status information.
- A cookie is a text file stored on a client. This file is associated with a specific web page and saves the information about the web page accessed by the client.



- HTTP/1.1 has been widely used since it was proposed in 1999 and has become a mainstream standard for more than 20 years. In the following part, we will introduce HTTP packets, which are based on HTTP/1.1.

Client Request Message

- An HTTP request message sent by a client consists of a request line, request header, empty line, and request data. The following figure shows the format of a request message.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Client Request Message - Request Line

- A request line consists of the request method field, URI field, and HTTP version field.
  - Request method: HTTP request method, for example, GET and POST. An HTTP client (for example, a browser) must specify the request type when sending a request to the server.
  - URI: A URI identifies the resource involved in a request.
  - Protocol version: The protocol version allows the sender to indicate the format of a message and its capacity for understanding further HTTP communication.

Example of a request line: GET http://www.w3.org/pub/WWW/TheProject.html HTTP/1.1

Request method	Space	URI	Space	Protocol version	Carriage return	Linefeed
Header field name:Value				Carriage return	Linefeed	
...						
Header field name:Value				Carriage return	Linefeed	
Carriage return	Linefeed					
Data						

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HTTP Request Method

- HTTP requests can use multiple request methods. HTTP 1.0 defines three request methods: GET, POST, and HEAD. HTTP 1.1 provides six new request methods: OPTIONS, PUT, PATCH, DELETE, TRACE, and CONNECT.

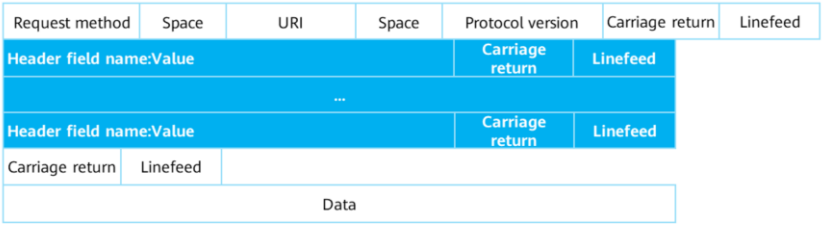
Method	Function
GET	Requests the specified page information. The server returns the specific data.
POST	Submits data, for example, a form.
HEAD	Similar to the GET method. However, the response does not contain any specific data. HEAD is used to obtain the header.
PUT	Updates and modifies data.
DELETE	Deletes a specified page.
CONNECT	Implements HTTP proxy.
OPTIONS	Allows the client to check the server performance.
TRACE	Echoes back the request received by the server. This is used for testing or diagnosis.
PATCH	Partially updates known resources.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Client Request Message - Request Header

- The request header allows the client to send additional information about the request to the server. These fields act as request modifiers whose semantics are equivalent to the parameters invoked in the programming language method.

Example of a request header:
Accept: text/html
Accept-Encoding: gzip, deflate, br
Connection: keep-alive



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

Request Header Fields

Request Header Field	Description
Accept	Media types which are acceptable for the response.
Accept-Encoding	Data encoding format that can be used by the client for decoding, for example, gzip.
Accept-Language	Natural languages that are preferred as a response to the request. This field is mandatory if the server can provide more than one language.
Authorization	Authorization information, which is usually contained in the response to the WWW-Authenticate header sent by the server.
Content-Type	Media type of the data. The default value is text/plain.
Content-Length	Length of the request message body.
Host	Name of the host to be accessed by the client.
Referer	Resource from which the client accesses the server. This field contains a URI, indicating that the client accesses the requested page from the page represented by the URI.
User-Agent	Information about the user who sends the request and the client type.
Cookie	Used by the client to save the data returned by the server. Generally, the user identity information is saved.
Connection	Whether to disconnect the connection after the request is processed.

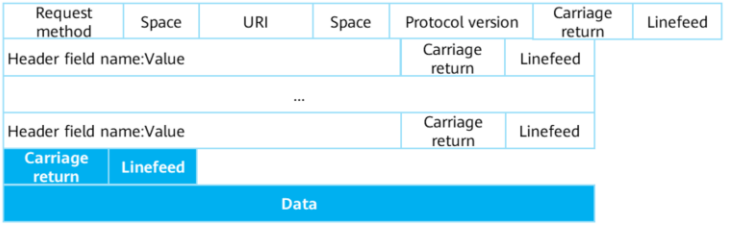
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- In HTTP 1.0, each connection involves only one request and response and is closed after the request is processed. HTTP 1.0 does not have the **Host** field. In HTTP 1.1, multiple requests and responses can be transmitted in the same connection, and multiple requests can be processed concurrently.
- WWW-Authenticate is a simple and effective user identity authentication technology in the early stage.
- The browser differentiates the displayed content such as HTML, XML, GIF, and flash based on MIME-type.
- For more information, refer to RFC HTTP 1.1 at https://www.ietf.org/rfc/rfc2616.html.

Client Request Message — Request Data

- Empty line: The server is notified of the end of a request header through an empty line.
- Request data: If the GET method is used, the request data is empty. If the POST method is used, the request data is the data to be submitted.

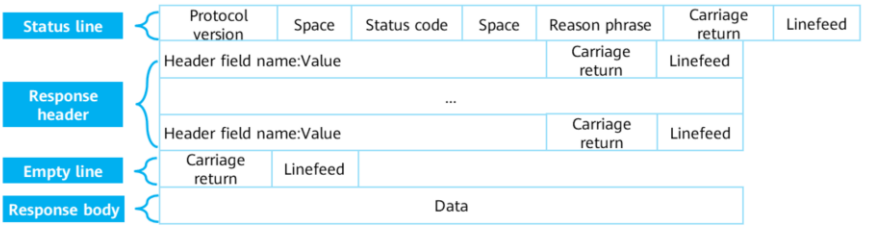
Example of request data:
user=admin&password=123456



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

Server Response Message

- An HTTP response consists of four parts: status line, response header, empty line, and response body.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

Server Response Message — Status Line

- The first line of the response message is the status line, which consists of the protocol version, status code, and reason phrase, separated by a space character.
  - Protocol version: The protocol version allows the sender to indicate the format of a message and its capacity for understanding further HTTP communication.
  - Status code: a three-digit result code, which is used to return the operation result to the client.
  - Reason phrase: a short text description of the status code, which facilitates understanding.

Example of a status line: HTTP1.1 200 OK

<table>
<tr><th>Protocol version</th><th>Space</th><th>Status code</th><th>Space</th><th>Reason phrase</th><th>Carriage return</th><th>Linefeed</th></tr>
<tr><td colspan="4">Header field name:Value</td><td>Carriage return</td><td>Linefeed</td><td></td></tr>
<tr><td colspan="7">...</td></tr>
<tr><td colspan="4">Header field name:Value</td><td>Carriage return</td><td>Linefeed</td><td></td></tr>
<tr><td>Carriage return</td><td>Linefeed</td><td colspan="5"></td></tr>
<tr><td colspan="7">Data</td></tr>
</table>

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



Status Code

- An HTTP status code is a three-digit number indicating the status of a response from the server. It is used to return the operation result to the client.

Status Code		Description	
1XX	(Informational) The request is received.	100 Continue	If the request is received, go to the next step.
2XX	(Successful) The request is successful.	200 OK	Success. The response body is available.
		201 Created	Resource created successfully.
		204 No Content	Success. No response body is available.
3XX	(Redirection) Further operations need to be performed.	301 Moved Permanently	A new URI is allocated to the target resource, and all future resources will be associated with the new URI.
4XX	(Client Error) Request error.	400 Bad Request	The request body is incorrect and carries error information.
		401 Unauthorized	Authorization failed. For example, the certificate does not match.
		403 Forbidden	Access denied. The possible cause is that the user attempts to perform operations beyond the permission or the login user name or password is incorrect.
		404 Not Found	The requested resource cannot be found.
5XX	(Server Error) Server error.	500 Internal Server Error	The request cannot be executed due to an internal server error. The user needs to resend the request later.
		501 Not Implemented	The function has not been implemented.

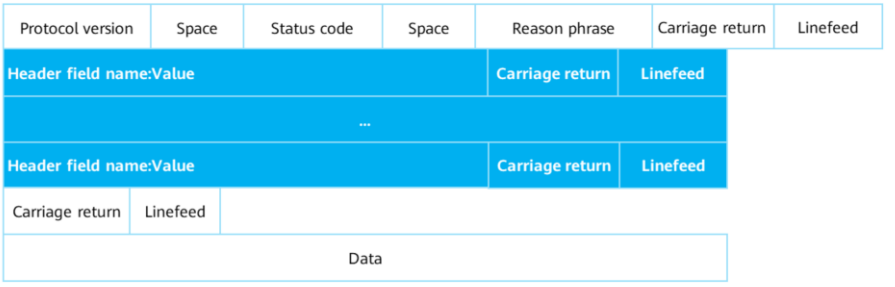
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Status line > Response header > Response body

Server Response Message — Response Header

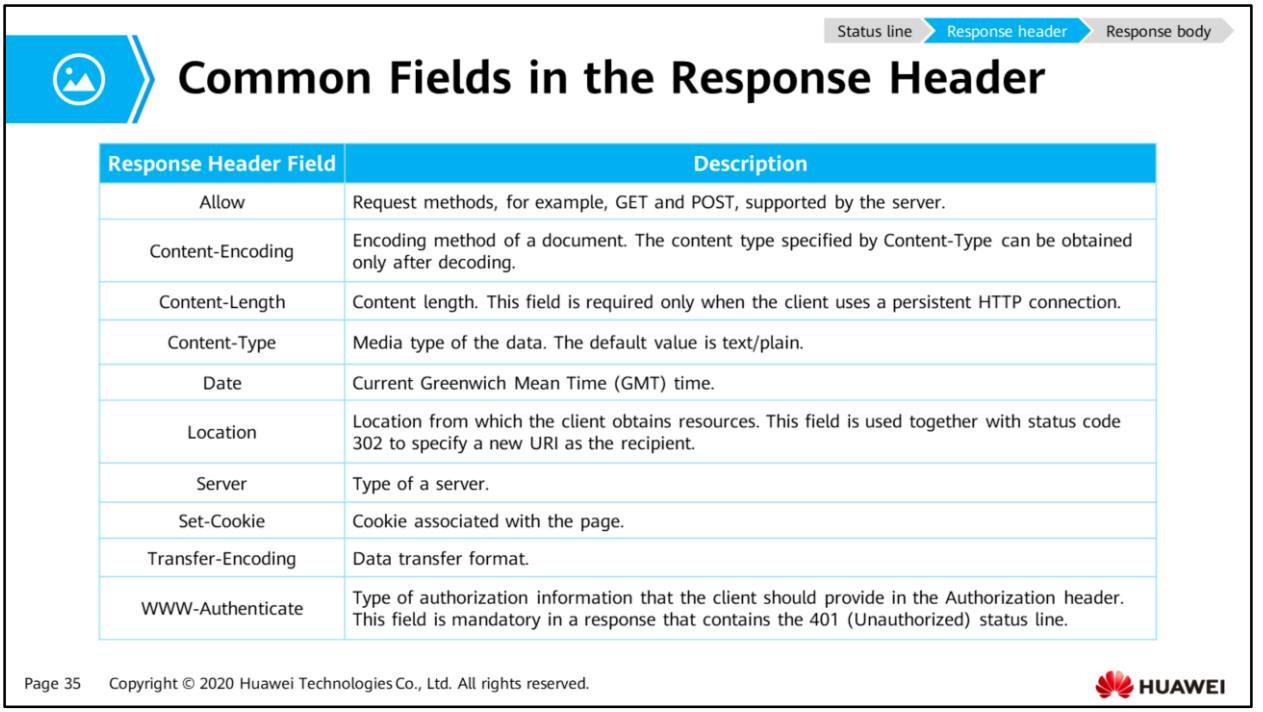
- A response header allows the server to transfer additional information about the response. These header fields provide information about the server and the resource identified by the URI.

Example of a response header: Server: JSP3/2.0.14



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

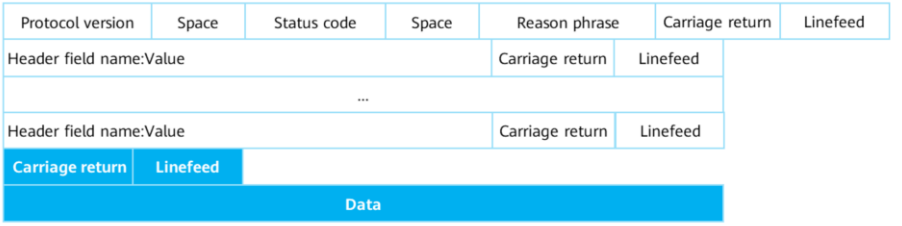
- The response header describes the basic information about the server and data. The server uses the response header to notify the client of how to process the data that it replies to.



- The HTTP response header is often combined with the status code. For example, the status code 302 (indicating that the location has changed) is usually used together with the Location header, and the status code 401 (Unauthorized) must be used together with a WWW-Authenticate header. The response header can be used to set the cookie, specify the date, instruct the client to refresh the page at the specified interval, and so on.

Server Response Message — Response Body

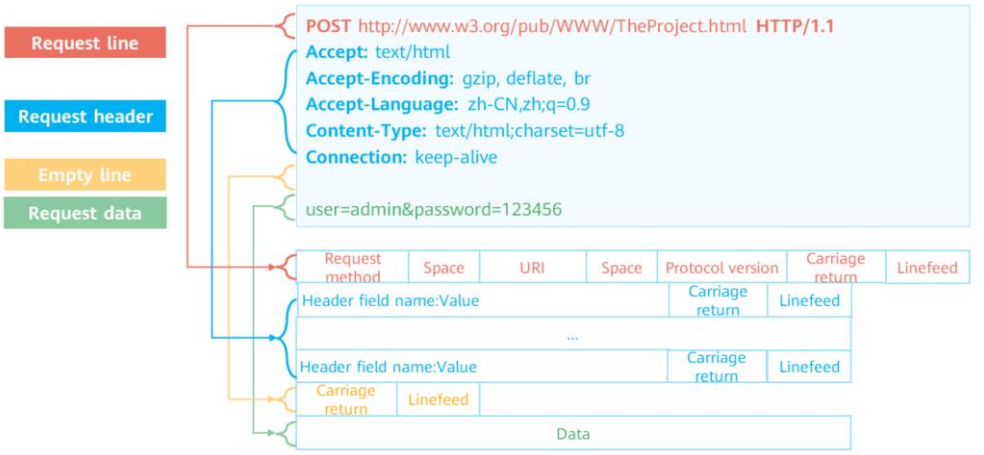
- Empty line: It is used to inform the server of the end of the response header.
- Response body: message body of the response. If data of the pure data type is requested by the client, pure data is returned. If an HTML page is requested, the HTML code is returned.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Client Request Message — Example

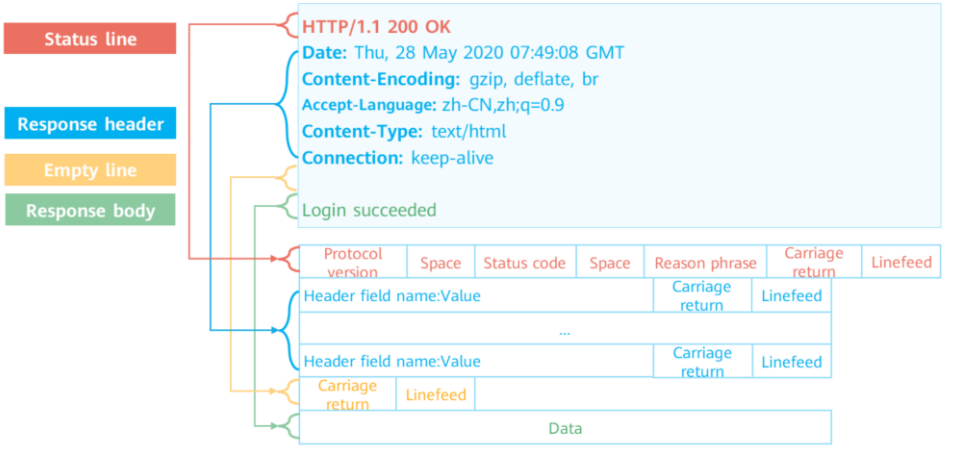
- A client sends a request containing the user name and password to the server for login authentication.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Server Response Message — Example

- The server sends a response to the client, indicating that the authentication is successful.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

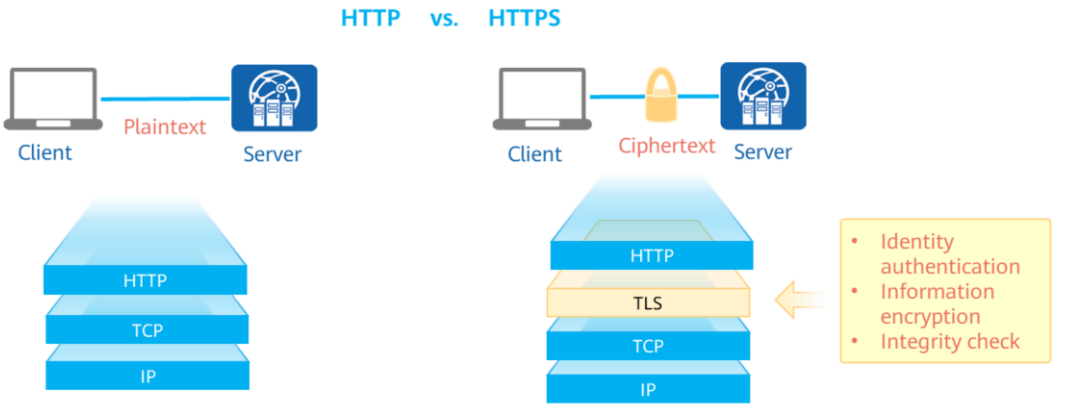
Contents

1. REST and RESTful
2. **Working Principle of HTTP**
   - Overview of HTTP
   - HTTP/1.1
   - **HTTPS and HTTP/2**
3. Practices in Invoking RESTful APIs

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Overview of HTTPS

- Based on HTTP, Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS) uses SSL/TLS for encryption, enhancing security.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- HTTP transmits information in plaintext, which may pose risks of information interception, tampering, and hijacking. Transport Layer Security (TLS) provides identity authentication, information encryption, and integrity check functions, and therefore can prevent such problems.

Overview of SSL/TLS

- Secure Sockets Layer (SSL) and Transport Layer Security (TLS) are security protocols that ensure security and data integrity for network communication.
- TLS is the successor of SSL. SSL and TLS encrypt data transmitted on network connections between the transport layer and application layer without affecting the TCP and HTTP protocols. Therefore, if HTTPS is used, it does not require much modification on HTTP pages.

Risks of HTTP communication without using SSL/TLS

1. Eavesdropping: HTTP transmits information in plaintext, and third parties can obtain communication data.
2. Tampering: Third parties can tamper with the communication content.
3. Pretending: Third parties can pretend others to participate in communications.

Features of HTTPS communication using SSL/TLS

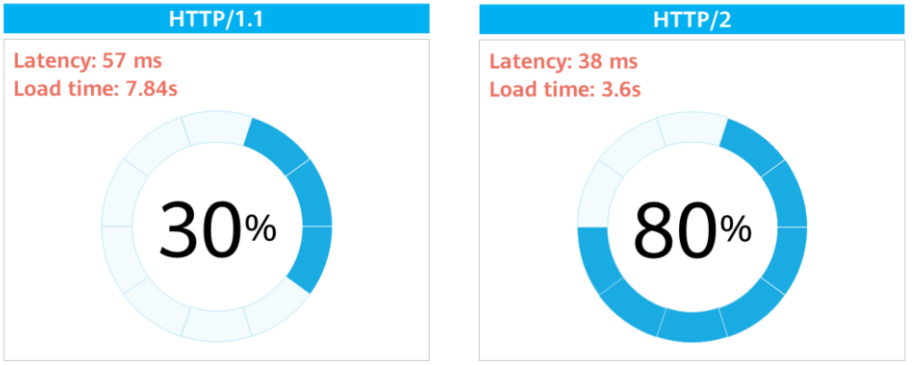
1. Data integrity: HTTPS uses SSL/TLS to encrypt and transmit all HTTP packets. All information is encrypted and cannot be intercepted or tampered with by third parties.
2. Data verification: Communication data is verified. If data is tampered with, the communication parties can immediately detect the tampering.
3. Trusted data source: Identity certificates are used to prevent identity spoofing.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- For more information, refer to the RFC document at https://www.ietf.org/rfc/rfc5246.html.

Overview of HTTP/2

- HTTP/2 is the second major version of the HTTP protocol. It was initially named HTTP 2.0 and is based on the SPeeDY (SPDY) protocol.
- HTTP/2 greatly improves web performance without changing HTTP semantics, methods, status codes, URIs, and header fields.

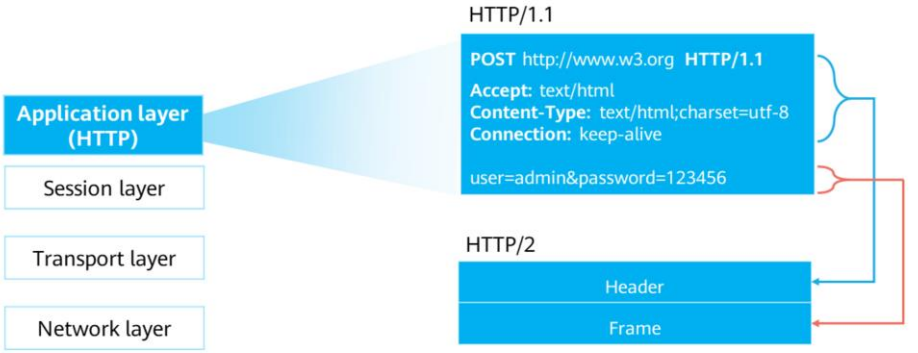


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- SPDY is a TCP-based application-layer protocol developed by Google. SPDY aims to optimize the performance of HTTP and shorten the loading time of web pages and improve security by using technologies such as compression, multiplexing, and priority. The core idea of SPDY is to minimize the number of TCP connections. SPDY is an enhancement to HTTP, instead of a protocol for replacing HTTP.

Core of HTTP/2 — Binary Transmission

- The core of performance enhancement in HTTP/2 is binary transmission.
- In HTTP/1, data is transmitted in text mode, which has some defects. This is because various scenarios need to be considered to achieve robustness, due to the diversity of text formats. On the contrary, the binary format involves only 0s and 1s, featuring significant convenience and robustness.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- Enhancements to HTTP/2:
- Header compression: The HPACK algorithm is used to compress headers to reduce the header size and improve performance.
- Multiplexing: A request message can be divided into frames, which are sent in sequence and are reassembled at the other end. In HTTP/1.1, when a client sends multiple requests through a TCP connection, the server can only respond to the requests in sequence. Subsequent requests may be blocked.
- Resource pushing: In addition to responding to client requests, the server can push additional resources to clients.
- Priority: HTTP/2 defines complex priority rules. A browser can request multiple resources at a time and specify priorities to help the server determine how to process these resources, avoiding resource competition.

Contents

1. REST and RESTful
2. Working Principle of HTTP
3. **Practices in Invoking RESTful APIs**

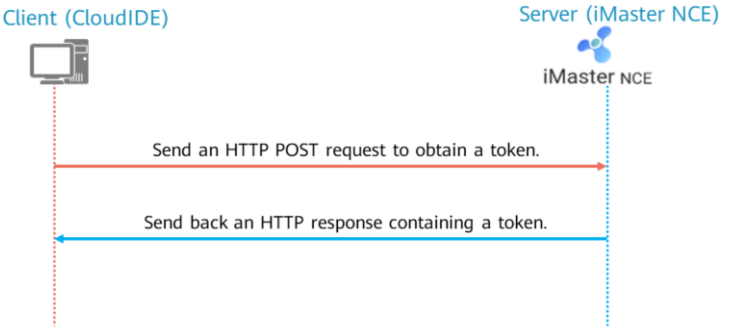
 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

Case: Invoking RESTful APIs to Obtain Tokens

Description:

- Huawei iMaster NCE is a network automation and intelligence platform that integrates management, control, analysis, and artificial intelligence (AI) functions. In the SDN network architecture, iMaster NCE centrally manages network devices. You can implement intent-driven networks based on open northbound APIs of iMaster NCE.
- This case describes an example of how to use the Python Requests module to log in to iMaster NCE and invoke service interfaces.



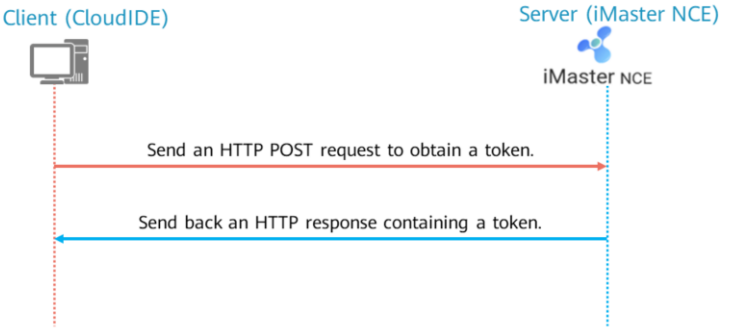
 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- In this case, two objects are involved in the networking: CloudIDE and iMaster NCE.
- CloudIDE is a cloud-based development environment provided by HUAWEI CLOUD. The local environment can also be used to write code.
- A sandbox environment is available to iMaster NCE in the datacom developer community.
- A token is a character string used for authentication on APIs.
- For details, see HCIP-Datacom-Northbound Openness Lab Guide.

Case: Invoking RESTful APIs to Obtain Tokens

Configuration roadmap:

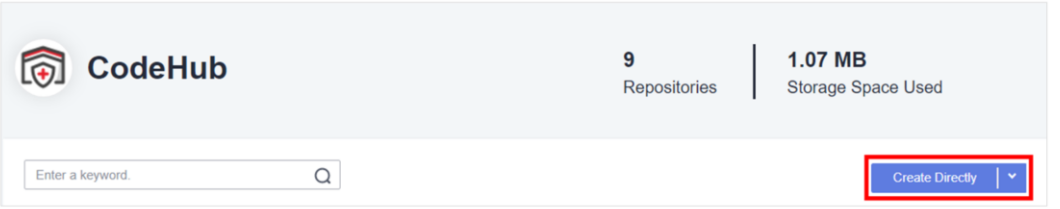
- Prepare the environment, including the IDE and sandbox environment.
- Compile the code for invoking Python RESTful APIs.
- Verify the configuration.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Procedure: Create a Code Repository

1. Access the CodeHub console and create a project and a code repository on the cloud.



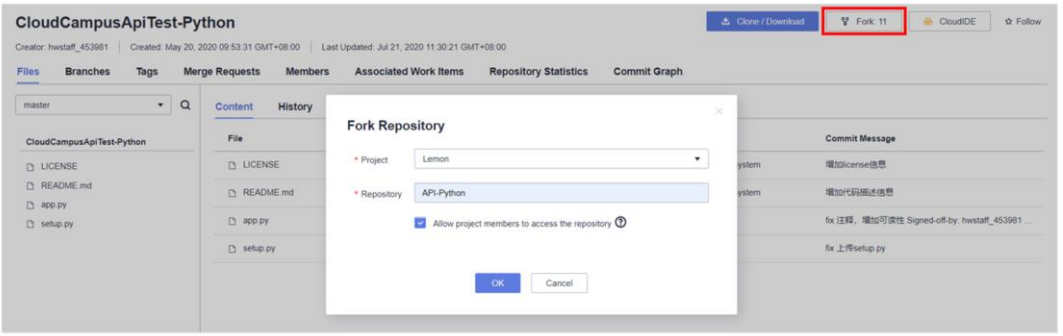
 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- For more operations, see the CodeHub guide at https://support.huaweicloud.com/codehub/index.html.

Procedure: Fork the Complete Code

2. Fork the complete code to the repository. In this example, the complete code has been uploaded to DevCloud on HUAWEI CLOUD. You can use Huawei CloudIDE to download the code to the local host. (Alternatively, you can perform online programming.)



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

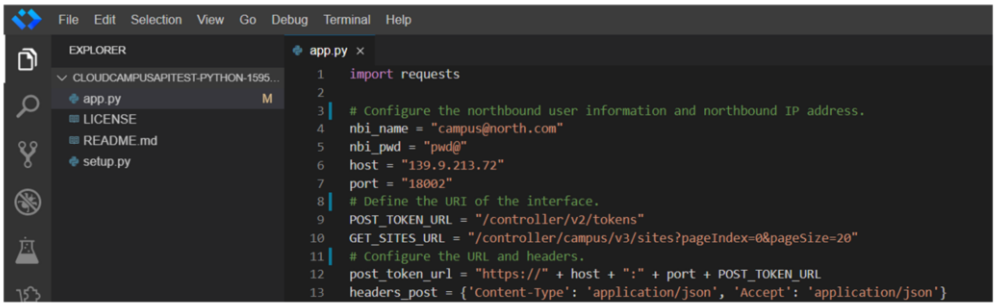
- Code in DevCloud on HUAWEI CLOUD: https://devcloud.cn-north-4.huaweicloud.com/codehub/project/68494a8ad06b4eea9a1c3f18be115161/codehub/620653/home

Procedure: Start the CloudIDE

3. Click **CloudIDE** to open the code.



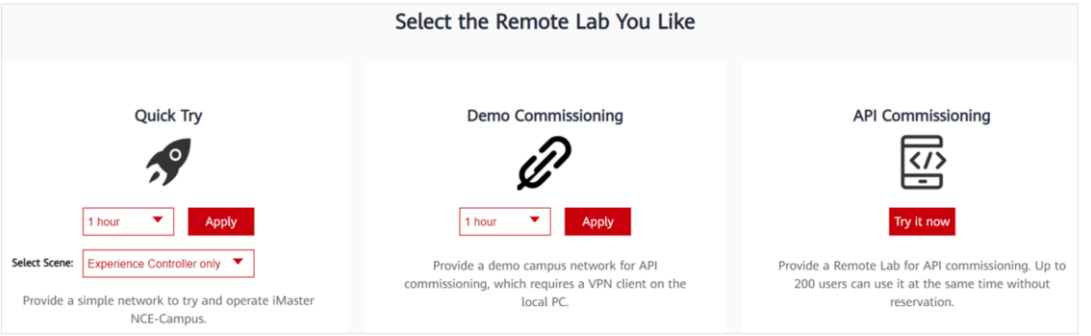
4. Check the code in CloudIDE.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Procedure: Reserve a Sandbox Environment

5. Reserve a sandbox environment.
   - The developer community provides the sandbox experience. The intent-driven campus network sandbox environment is used as an example.
   - You can experience the sandbox environment in three ways: **Quick Try**, **Demo Commissioning**, and **API Commissioning**. Here, **Quick Try** in CloudIDE is used as an example.

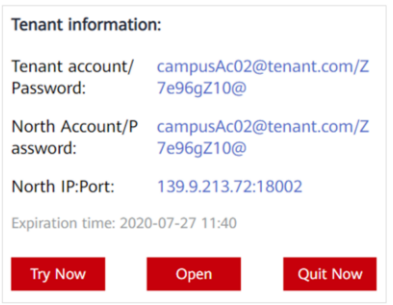


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Website for reserving a sandbox environment: https://devzone.huawei.com/openecosystem/experienceView/campus.html

Step: Check Sandbox Information

6. Check sandbox information.
   - **Tenant account**: account for logging in to iMaster NCE.
   - **North Account**: account used by a location application to interact with iMaster NCE.
   - **Northbound IP:Port**: URL for logging in to iMaster NCE. You can click **Try Now** to log in to iMaster NCE.
   - **Password**: password for logging in to the sandbox environment.



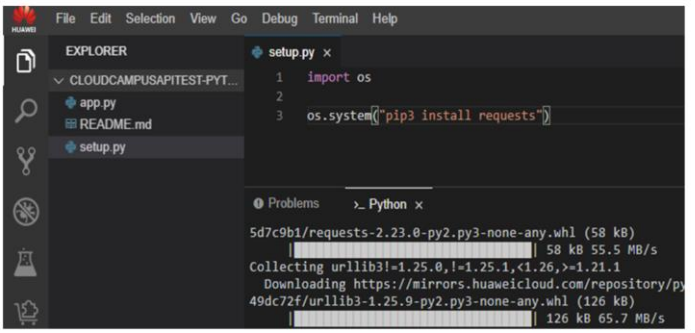
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Procedure: Install the Requests Module

7. Install the Requests module.
   - The Python Requests module is required in this experiment. Create and run the **setup.py** script in CloudIDE to install the Requests module. Alternatively, run **pip3 install requests** on the CloudIDE Python CLI to install the Requests module.

```
import os

os.system("pip3 install requests")
```



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- To run the code, right-click setup.py and choose Run Python File in Terminal from the shortcut menu.

Procedure: Compile a Python Script

8. Compile a Python script.

```
import requests

nbi_name = "campus02@north.com"
nbi_pwd = "5aAZ4lh44@"
host = "139.9.213.72"
port = "18002"

POST_TOKEN_URL = "/controller/v2/tokens"

post_token_url = "https://" + host + ":" + port + POST_TOKEN_URL
headers_post = {'Content-Type': 'application/json', 'Accept': 'application/json'}

r = requests.post(post_token_url, headers=headers_post, json={"userName":
nbi_name, "password": nbi_pwd}, verify=False)

token_id = r.json()['data']['token_id']
print("1.[Get Token Id]")
print("[post_token_url]:"+post_token_url)
print("[token_id]:"+token_id)
```

Configure the northbound user information and northbound address.

Define the URI of an interface.

Configure the URL.

Configure the request header.

Initiate a request, whose body is in JSON format.

Parse the returned data to obtain the token.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Procedure: Verify the Configuration

- The token (specified by token_id) is obtained by invoking the RESTful API.

```
1.[Get Token Id]
[Post_token_url]: https://139.9.213.72:18002/controller/v2/tokens
[Token_id]: x-
rs5jvy7z47vsum042r2qc9vtaq44o9hgqohf3uca7z9clio5vvvz86al3s5dvs0aphpck5defw45uq
pi3xdgddmn482kgajutf7vlfs649rw6k5c6r6l0a5i1g4b5j9c
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



Quiz

1. (Single) Which of the following HTTP request methods does not need to contain request data in an HTTP request? ( )

   A. POST

   B. GET

   C. PUT

   D. DELETE

2. What is the relationship between REST and RESTful?

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

1. B
2. RESTful APIs comply with the REST design style.

Summary

- REST is an architectural or design style rather than a standard. It provides only design principles and constraints.
- REST makes full use of HTTP. The core of REST is to perform operations (such as GET, POST, PUT, and DELETE) on resources in multiple forms.
- By invoking RESTful APIs, users can write Python scripts to implement automation as required.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

More Information

- REST dissertation
  - https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm
- RFC HTTP/1.1
  - https://www.ietf.org/rfc/rfc2616.html
- RFC HTTP/2
  - https://www.ietf.org/rfc/rfc7540.html
- RFC HTTPS
  - https://www.ietf.org/rfc/rfc2818.html

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.


Thank You
www.huawei.com



iMaster NCE-Campus Open API Introduction

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Foreword

- iMaster NCE-Campus is a next-generation SDN controller for enterprise campuses and is a core component of Huawei CloudCampus Solution.
- iMaster NCE-Campus centrally manages and controls enterprise campuses and provides automatic deployment, security defense, and visualized O&M. It helps customers build service-centric dynamic scheduling capabilities of network services.
- iMaster NCE-Campus provides standard-compliant open RESTful APIs for partners to develop innovative applications and continuously create value for enterprise customers.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Objectives

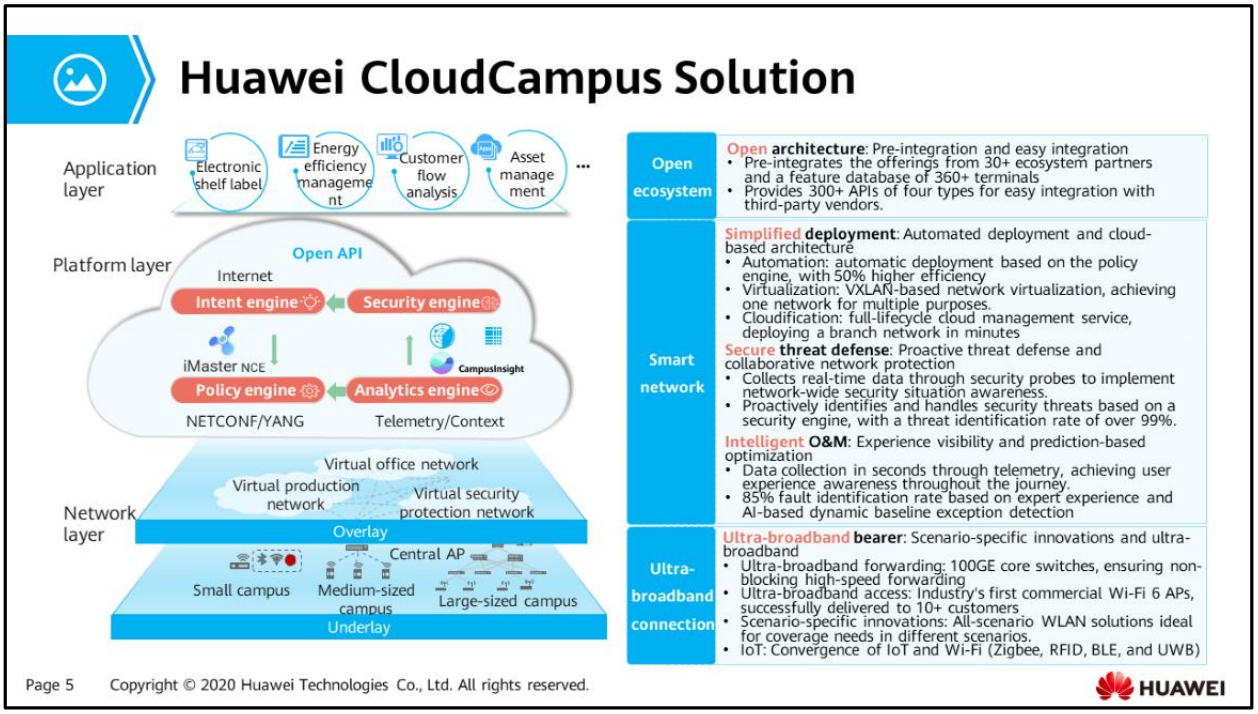
- Upon completion of this course, you will be able to:
  - Describe Huawei CloudCampus Solution.
  - Describe main open APIs of iMaster NCE-Campus.
  - Describe typical application scenarios of iMaster NCE-Campus.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

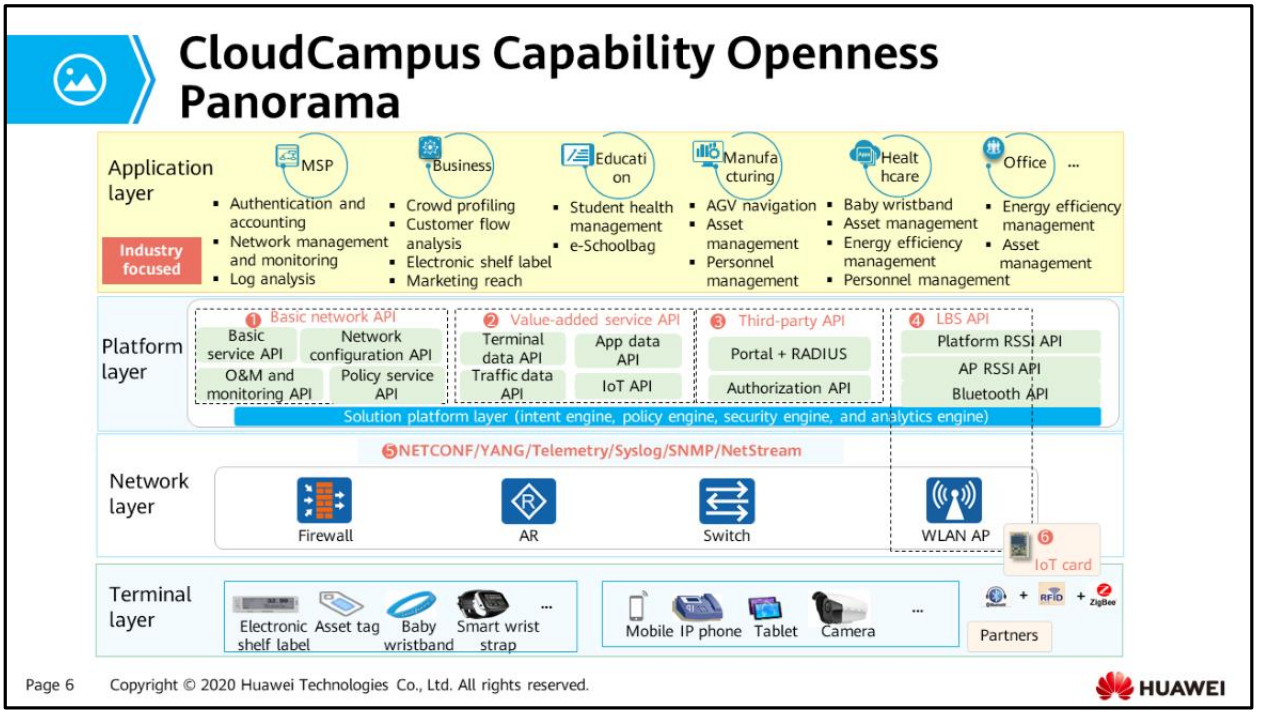
Contents

1. **Solution Overview**
2. Typical Scenarios and Open Capabilities
3. Openness and Cooperation Cases

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

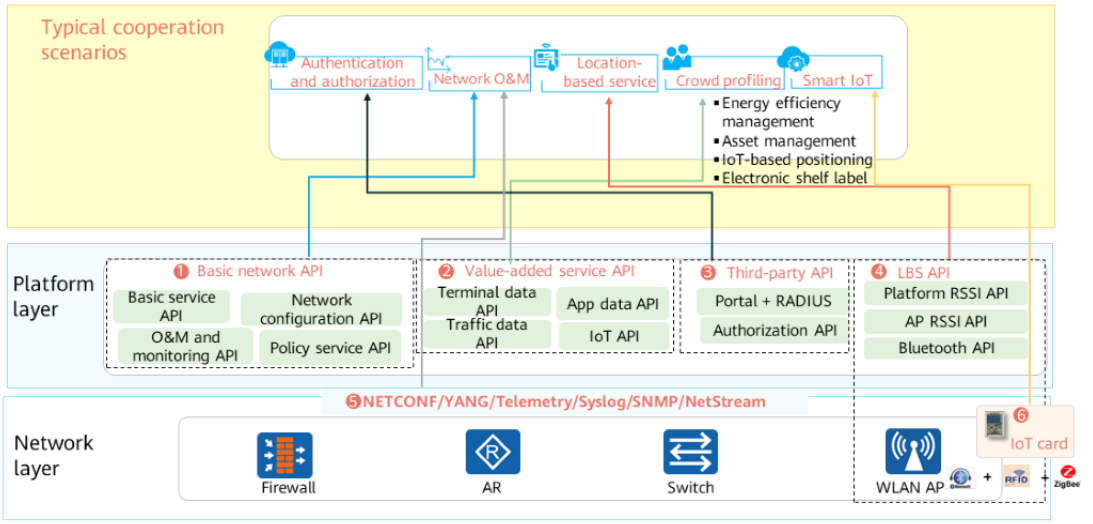


- iMaster NCE-Campus is Huawei's next-generation autonomous driving network management and control system for campus networks. It is a network automation and intelligent platform integrating manager, controller, analyzer, and AI functions. As such, this platform drives enterprise cloudification and digital transformation, and creates a shortcut to more automated network management and intelligent network O&M.



- Application layer: It focuses on mainstream scenarios and provides industry-wide applications. At this layer, network service data is shared, meeting value-added data and operation requirements.
- Platform layer: It provides four types of APIs and supports industry-standard network interconnection protocols.
- Network layer: It provides various open interfaces, such as NETCONF, YANG, and Telemetry, improving device manageability. APs are compatible with third-party IoT cards to implement IoT.
- Terminal layer: It supports access of IoT terminals (such as ZigBee, RFID, and BLE), and access of wired and wireless terminals (such as mobile phones, IP phones, tablets, and cameras).

Typical Cooperation Scenarios and Open APIs



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

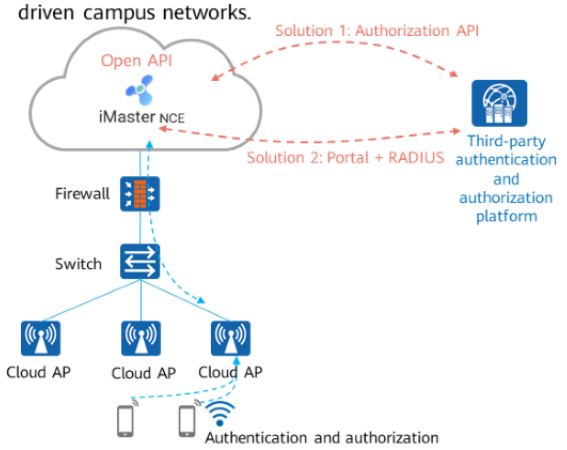
Contents

1. Solution Overview
2. **Typical Scenarios and Open Capabilities**
   - **Authentication and Authorization**
   - LBS
   - Crowd Profiling
   - Network O&M
   - Smart IoT
3. Openness and Cooperation Cases

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Introduction to Authentication and Authorization

- Authentication and authorization: A third-party authentication and authorization platform interworks with iMaster NCE-Campus to perform network access authentication and accounting for terminals that access Huawei intent-driven campus networks.

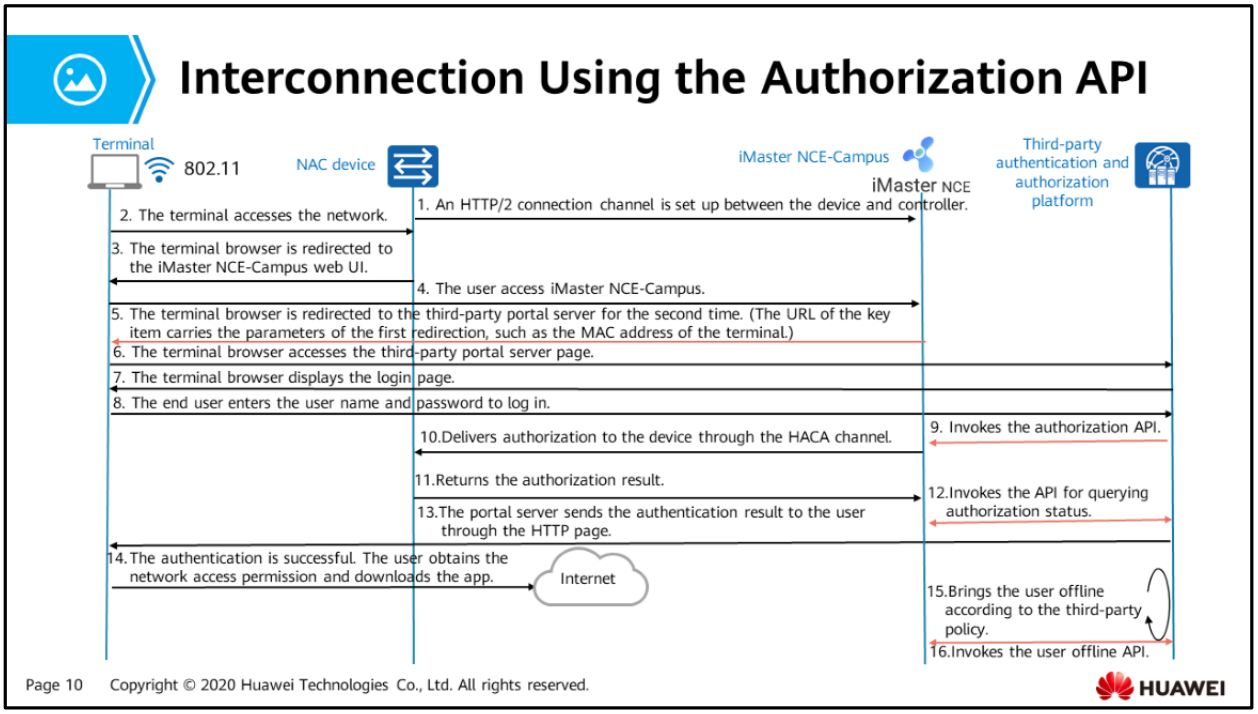


- iMaster NCE-Campus provides two integration solutions for authentication and authorization using the third-party authentication and authorization platform. Tenants can select a solution based on the site requirements.

Integration Solution	Suggestions
Portal + RADIUS	iMaster NCE-Campus functions as a RADIUS client to interact with the third-party authentication and authorization platform for user authentication and authorization. Accounting is recommended in this solution.
Authorization API	After users are authenticated, the third-party authentication and authorization platform invokes the authorization API to instruct iMaster NCE-Campus to authorize the users. Accounting is not recommended in this solution.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- The tenant or MSP wants to use an existing or third-party authentication platform to authenticate user identities and authorize users for network access authentication through the web page (authentication portal). For example, an MSP provides a unified access authentication page for tenants.



- To access the Internet, a user connects to the SSID of a Wi-Fi network and logs in to the portal pushed by a developer app. The developer app calls the authorization API of Huawei iMaster NCE-Campus to deliver the user's Wi-Fi access permission to the AP. The user then can access the Internet.
- NAC is short for network access control.
- Huawei Agile Cloud Authentication (HACA) is based on the mobile Internet protocol HTTP/2.

Interconnection Using the Authorization API

Sample Code (Windows)

The API uses HTTPS and the port 18008.
This is a RESTful API, and **Content-Type** is **application/json**.
The following shows only a part of the API URL. For example, if the address of iMaster NCE-Campus is **https://10.186.40.148:18008/**, the complete URL of the authorization API is **https://10.186.40.148:18008/controller/cloud/v2/northbound/accessuser/haca/authorization**.

Request

POST
/controller/cloud/v2/northbound/accessuser/haca/authorization

```
{
    "deviceMac":"AA:AA:AA:00:00:89",
    "deviceEsn":"AA50082935AAAA000088",
    "ssid":"YWFh",
    "policyName":"dfdsffd",
    "terminalIpV4":"10.11.11.11",
    "terminalMac":"ff-ff-ff-ff-ff-f1",
    "username":"liao",
    "nodeIp":"10.186.40.148",
    "apMac":"AA:AA:AA:00:00:89"
}
```

Response

STATUS CODE 200

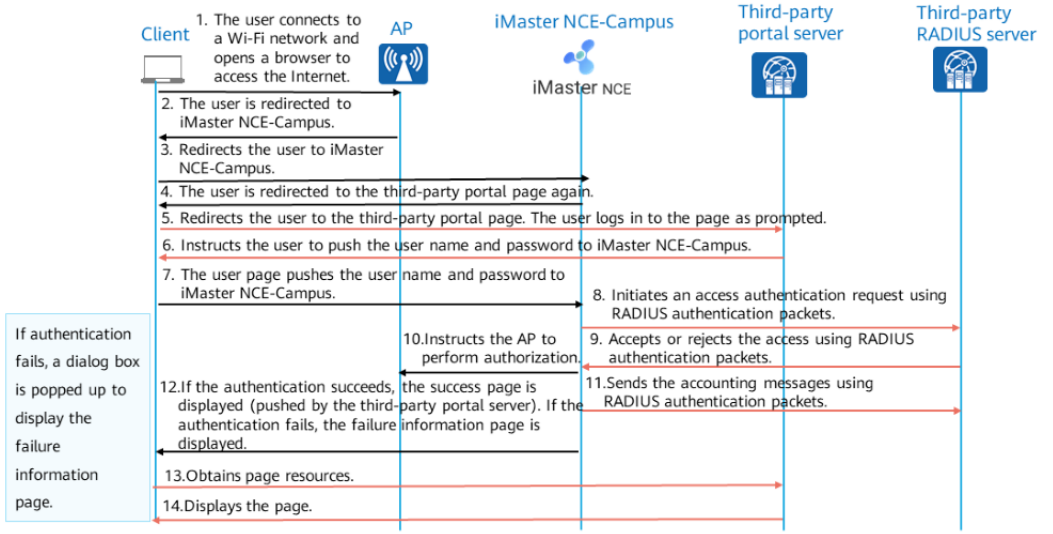
```
{
  "errcode": "",
  "errmsg": "",
    "psessionid":
    "846d65a4e1bb38c350a280f46e6d2f6ab0a688df7edbb72b77bc5d03e3
    926cd6"
}
```

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- For more information about API-based authentication and authorization, visit **https://devzone.huawei.com/cn/enterprise/campus/apiSolution.html**.

Interconnection Using Portal and RADIUS



 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- For details about RADIUS-based authentication, visit https://devzone.huawei.com/cn/enterprise/campus/radiusSolution.html.

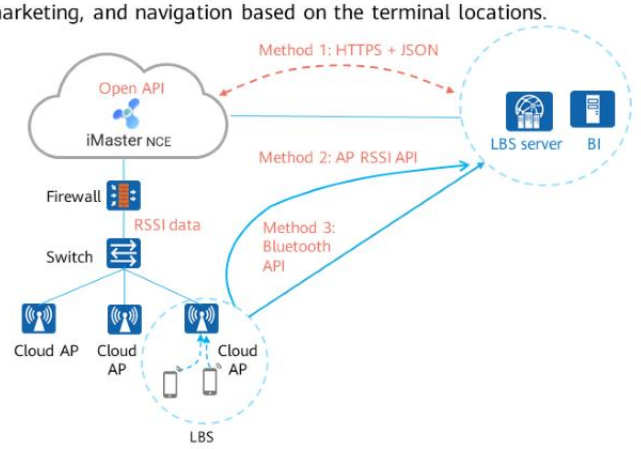
Contents

1. Solution Overview
2. **Typical Scenarios and Open Capabilities**
   - Authentication and Authorization
   - **LBS**
   - Crowd Profiling
   - Network O&M
   - Smart IoT
3. Openness and Cooperation Cases

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

LBS Introduction

- A location-based service (LBS) is deployed on a third-party LBS application platform and used to detect and locate terminals managed on Huawei intent-driven campus networks. The LBS provides the customer flow analysis, Wi-Fi marketing, and navigation based on the terminal locations.



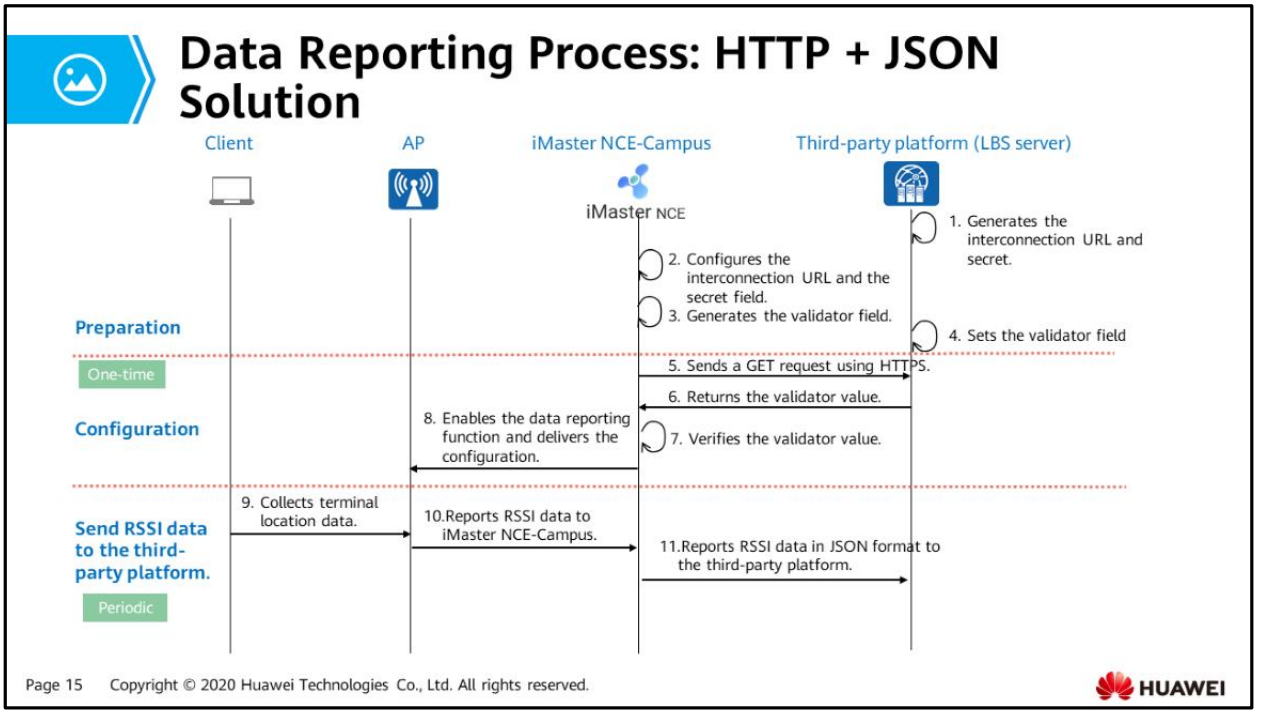
Application Scenario

- Huawei provides location data of Wi-Fi and Bluetooth terminals to a third-party partner. The partner parses the location data and provides VAS applications, such as heat map, track tracing, and customer flow analysis, for end customers.

Integration Solution	Suggestions
HTTPS + JSON	Applicable to the interconnection with an LBS system deployed based on software as a service (SaaS).
AP RSSI API	Applicable to the on-premise scenario.
Bluetooth API	Applicable to locating Bluetooth terminals and reporting terminal data.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- Location-based service (LBS) uses various locating technologies to obtain the current locations of devices and pushes information and basic service for these devices through mobile Internet.
- iMaster NCE-Campus aggregates the terminal location data collected by cloud APs and periodically sends the data to the third-party LBS platform. After parsing and analyzing the location data with a series of algorithms, the LBS platform provides VASs, such as heatmap, tracking, and customer flow analysis, for customers.
- Remarks: Partners need to meet related standards based on application scenarios, such as EU General Data Protection Regulation (GDPR).



- iMaster NCE-Campus can directly report terminal location data to a third-party LBS platform. In this solution, iMaster NCE-Campus function as a relay agent.
- For details about this process, see "Wi-Fi Terminal Location Practice in Huawei CloudCampus Solution" in the *HCIP-Datacom-NCE Northbound Openness Lab Guide*.

Data Example: HTTP + JSON Solution

Sample Code (Windows)

```
{
    "data":[
        {
            "apMac":"4C:FA:CA:D8:23:A0",
            "terminallist":[
                {
                    "terminalMac":"88:19:08:F1:88:45",
                    "rssi":-68,
                    "timestamp":1557460789000
                },
                {
                    "terminalMac":"90:2E:1C:6A:2A:57",
                    "rssi":-57,
                    "timestamp":1557460789000
                }
                ]
        }
    ],
    "secret":"Test@1234",
    "type":"Aplocation"
}
```

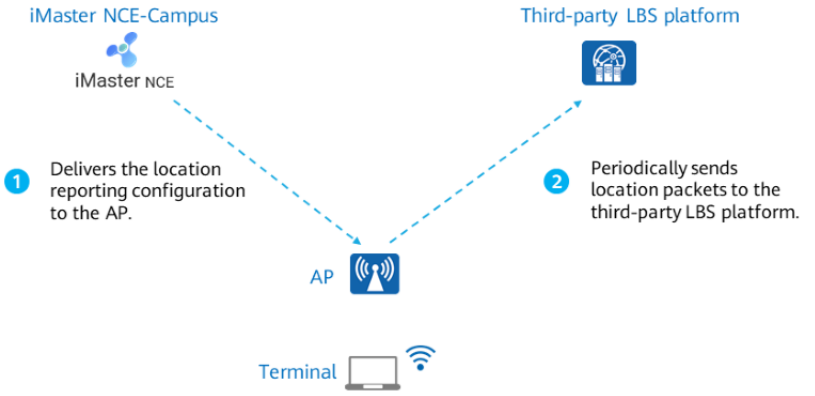
- secret: Used by a third-party LBS platform to verify the data sent by iMaster NCE-Campus. If the secret does not match that associated with the reported URL, the data is discarded. The value is generated by the third-party LBS platform, and is a string of 32 hexadecimal characters.
- validator: Before sending data, iMaster NCE-Campus checks whether the validator returned by the peer end is correct. If the validator does not match that associated with the reported URL, iMaster NCE-Campus does not send Wi-Fi terminal location data to the third-party LBS platform.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- The validator value is in UUID format and is generated by iMaster NCE-Campus.
- For more examples, visit https://developer.huaweicloud.com/techfield/network.html#CloudCampus.

Data Reporting Process: AP RSSI API Solution

- iMaster NCE-Campus allows WLAN APs to report terminal location data directly to a third-party LBS platform. iMaster NCE-Campus delivers the location reporting configuration to APs, and then the APs periodically send location packets to the third-party LBS platform.

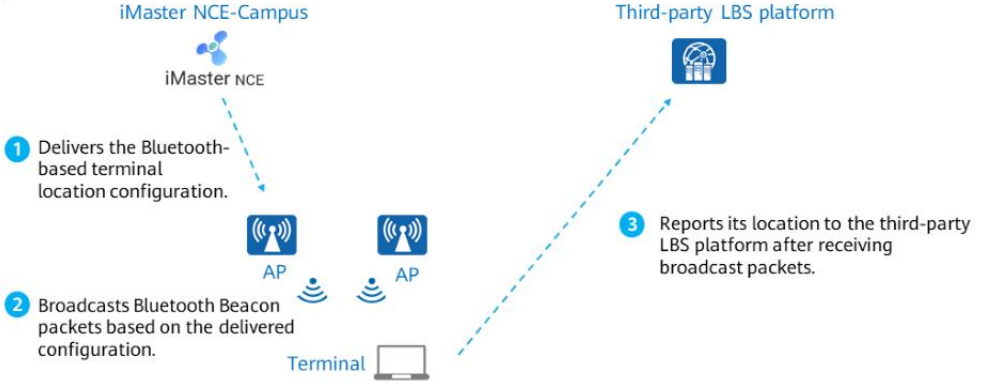


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- For more about the AP location reporting solution, see https://devzone.huawei.com/cn/enterprise/campus/lbsWiFiSolution.html#Wi-Fi Terminal Data Reporting Process.

Data Reporting Process: Bluetooth API Solution

- iMaster NCE-Campus allows WLAN APs to report terminal location data through Bluetooth Beacon packets. iMaster NCE-Campus delivers the Bluetooth-based terminal location configuration to APs. The APs then broadcast Bluetooth Beacon packets. After receiving the broadcast packets, terminals report their locations to the third-party LBS platform.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- For more about the Bluetooth API solution, see https://devzone.huawei.com/cn/enterprise/campus/lbsBluetoothSolution.html.

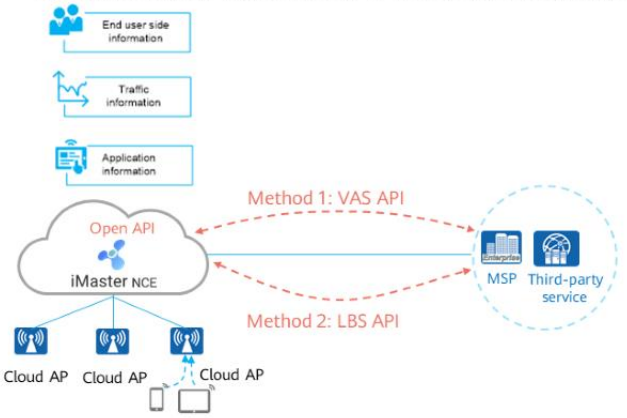
Contents

1. Solution Overview
2. **Typical Scenarios and Open Capabilities**
   - Authentication and Authorization
   - LBS
   - **Crowd Profiling**
   - Network O&M
   - Smart IoT
3. Openness and Cooperation Cases

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Crowd Profiling

- Crowd profiling: A third-party big data analysis platform analyzes the locations and status of terminals detected and managed on Huawei intent-driven campus networks to obtain the characteristics of crowd profiles in an area, so that target content can be pushed to the customers to provide personalized services.



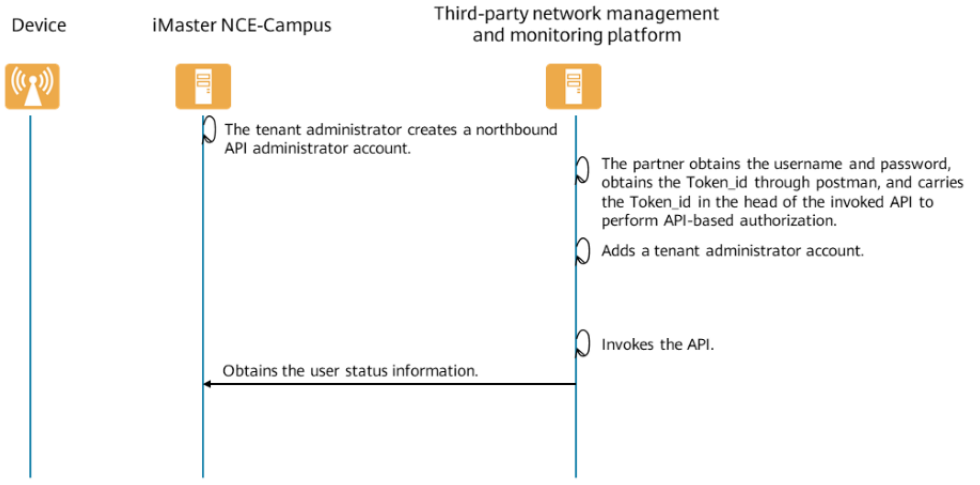
Application Scenario

- The business industry has the requirements for precision marketing. Direct marketing can tag specific people based on crowd profiles, and push target content to provide personalized services based on these tags. Crowd profiling requires two types of data: one is the data left online; the other type of data is left in brick-and-mortar stores (offline data), which can be collected using Wi-Fi probes and from network access data of customers.

Integration Solution	Suggestions
LBS API	Terminal location information is provided.
VAS API	Terminal login and logout information, network application information, and network traffic information are provided.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Invoking a VAS API



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- For more about VAS APIs, visit https://devzone.huawei.com/cn/enterprise/campus/valueAddedApi.html.

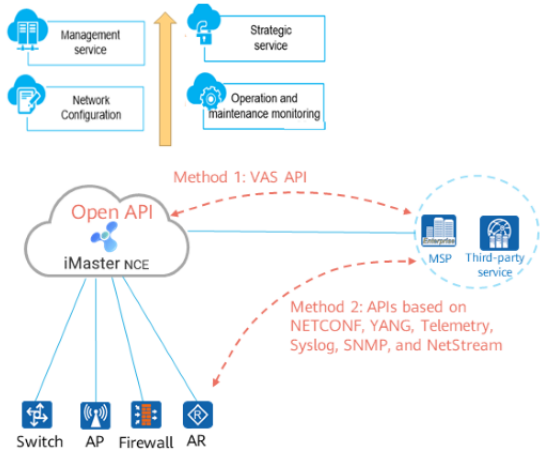
Contents

1. Solution Overview
2. **Typical Scenarios and Open Capabilities**
   - Authentication and Authorization
   - LBS
   - Crowd Profiling
   - **Network O&M**
   - Smart IoT
3. Openness and Cooperation Cases

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Network O&M Overview

- Network O&M: Third-party network management platforms (such as MSP-owned or mainstream network management and monitoring platforms) can manage or monitor devices managed by iMaster NCE-Campus.



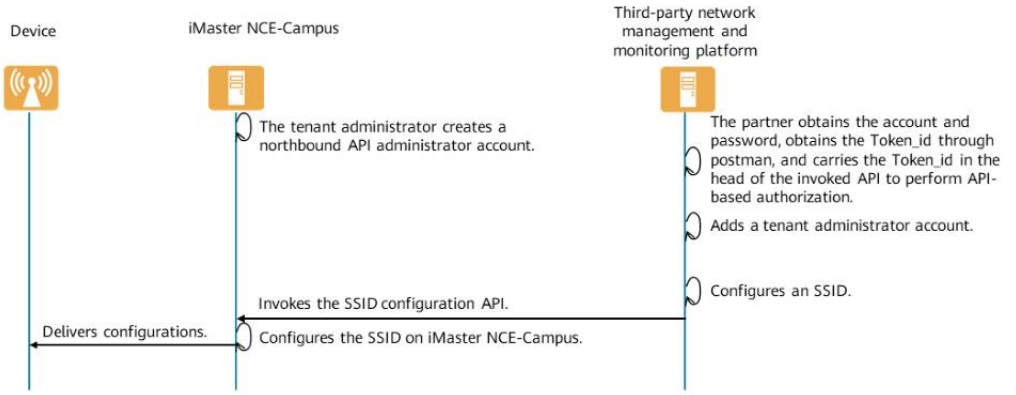
Application Scenario

- MSPs and customers use their existing network management platform to manage or monitor devices that are supposed to be managed by iMaster NCE-Campus, such as creating tenant administrator accounts, managing devices, configuring networks for specified devices, and monitoring device status and alarms.
- Currently, two methods are available to implement network O&M:
  - Network service APIs: The third-party network management platform manages and monitors devices through RESTful APIs.
  - Traditional device APIs based on NETCONF, YANG, Telemetry, and SNMP: Network management platforms can directly configure, manage, and maintain devices through these APIs.

Integration Solution	Suggestions
Network service API	Applicable to interconnection with various third-party private SaaS platforms.
APIs based on NETCONF, YANG, Telemetry, and SNMP	Applicable to interconnection with traditional network management platforms.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Interconnecting with a Third-Party Network Management and Monitoring Platform



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- For more basic network solutions, visit https://devzone.huawei.com/cn/enterprise/cloudcampus/quickStart.html#network.

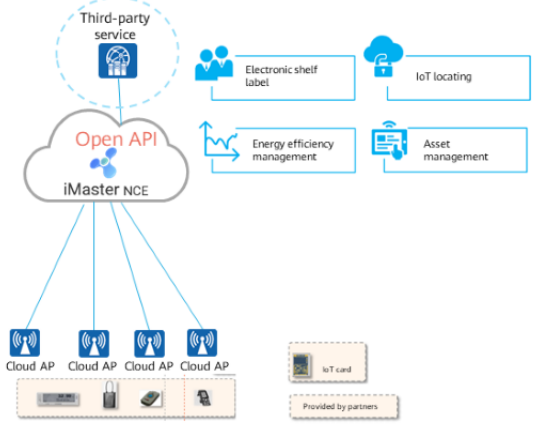
Contents

1. Solution Overview
2. **Typical Scenarios and Open Capabilities**
   - Authentication and Authorization
   - LBS
   - Crowd Profiling
   - Network O&M
   - **Smart IoT**
3. Openness and Cooperation Cases

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Introduction to Smart IoT

- Smart IoT: A third party expands an AP into an IoT base station through the open card capability of the AP to implement functions such as energy efficiency management, asset management, IoT locating, and electronic shelf label.



Application Scenario

- When partners want to deploy IoT applications (such as electronic shelf label, IoT locating, energy efficiency management, and asset management), they can use Huawei's network infrastructure to provide IoT signal coverage (ZigBee, Bluetooth, and RFID), without the need to deploy a secondary IoT network.

- Based on Huawei CloudCampus Solution, partners can develop IoT cards and IoT service software on the cards to provide IoT services based on the network infrastructure.
- Huawei provides the network infrastructure, open AP hardware, and basic IoT card management and monitoring.
  - Partners develop IoT cards, card software, and IoT service software.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Smart IoT: Open Cards

Built-in card



IoT slot: allows IoT cards to be installed to provide extended functions.



AP4050DN-E

External card



MT800





 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- For more about the smart IoT solution, visit https://devzone.huawei.com/cn/enterprise/cloudcampus/quickStart.html#iot.

Contents

1. Solution Overview
2. Typical Scenarios and Open Capabilities
3. **Openness and Cooperation Cases**

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

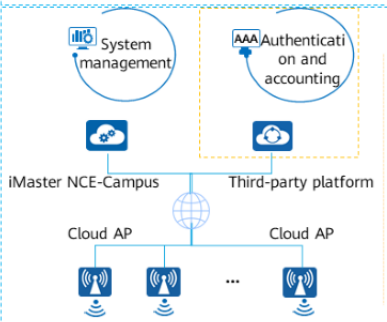
Unified Cloud-based Wi-Fi Management in XX

Background

- XX provides Wi-Fi services over more than 800 cities and has 1.5 million users.

Challenges

- How to ensure Wi-Fi signal quality and user access experience in diversified tenant environments.
- **O&M for massive devices.** A large number of tenants have devices scattered in difference places, resulting in high installation and O&M costs.



Solution Implementation

iMaster NCE-Campus implements plug-and-play and fast deployment of devices and provides full-lifecycle cloud-based and centralized management, simplifying O&M. It provides open APIs to seamlessly interconnect with third-party authentication and accounting management platforms, implementing secure network management.

Cooperation Scenario and API

Authentication and authorization: Portal + RADIUS API

Customer Benefits

- **Network coverage**: High-quality Wi-Fi coverage
- **Cloud-based management**: 80% lower OPEX
- **Smooth upgrade**: Securer customer investment

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

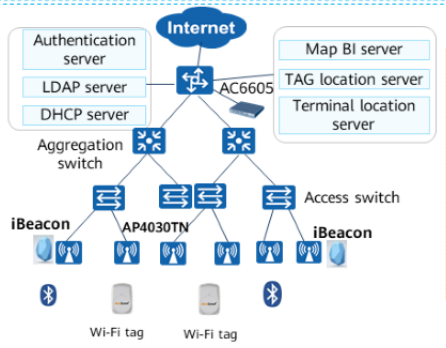
XX Builds the Future Business

Background

- XXX is a large city complex that integrates industry R&D, apartments, hotels, and businesses.
- It is a landmark building group in XXX. Wireless networks are an important infrastructure.

Challenges

- **Consumer experience**: The customer requires high-density, high-quality wireless network access to provide precise indoor navigation in more than 160 stores and 30,000 parking spaces.
- **Smart operations**: The customer wants to use big data analytics technology to describe consumer behavior and visualize staff locations in support of marketing and operational decision-making.



Solution Implementation

Three-RF APs have an independent radio to provide LBS services for Wi-Fi terminals and tags. Bluetooth beacons in the underground garage achieve a locating precision of 1 m to 3 m. Open APIs connect to app platform to implement customer flow analysis, indoor navigation, and precision marketing, laying a solid foundation for operational decision-making.

Cooperation Scenario and API

LBS: AP RSSI API

Customer Benefits

- Network quality: High-quality Wi-Fi network
- Customer flow analysis: Big data-based precision marketing
- Indoor navigation: Improved customers' shopping experience

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

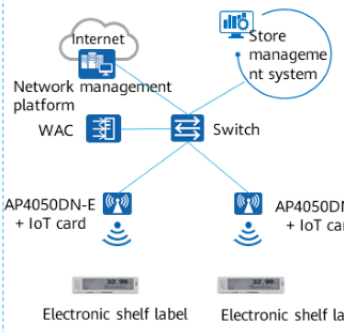
Digital Transformation of the XX Supermarket

Background

- XX is a well-known chain supermarket in China.
- One of China's top 500 enterprises and leading enterprise in China's national merchandising and agricultural industrialization

Challenges

- **Shopping experience**: The customer wants to provide high-quality wireless network access to improve customers' shopping experience.
- **Manpower**: The customer wants to reduce manpower and O&M costs, and improve the accuracy of supermarket services.



Dual-network convergence

Huawei AP4050DN-E provides built-in slots for IoT modules to converge Wi-Fi and IoT networks, achieving unified planning of the two networks.

Electronic shelf label

Electronic shelf labels interwork with the supermarket management system and customer's Enterprise Resource Planning (ERP) system to display prices in real-time and provide interactive functions such as code scanning, coupon claiming, and out-of-stock warning.

Wireless Internet service

Huawei AP4050DN-E helps build a high-quality wireless network for customers to scan QR codes for shopping or entertainment.

Cooperation Scenario and API

Smart IoT: IoT card

Customer Benefits

- **Converged management**: Converged high-quality Wi-Fi and IoT networks
- **Smart operations**: ESL-based smart operation supermarket
- **Wireless Internet access**: Improved customers' shopping experience

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

Quiz

1. (Multiple) Which of the following layers are available for openness on Huawei intent-driven campus networks? ( )

   A. Application layer

   B. Platform layer

   C. Network layer

   D. Terminal layer

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

1. ABCD

Summary

- This course describes the architecture, cooperation scenarios, and API packet structure involved in Huawei CloudCampus Solution to help readers understand the basic ecosystem openness capabilities of Huawei CloudCampus Solution and provide guides for interconnection between iMaster NCE and third-party applications based on these capabilities. It also helps MSPs, VAS application partners, and customers quickly implement system interconnection and service convergence, and helps customers and partners leverage network values.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

More Information

- Developer community:
  - https://developer.huaweicloud.com/techfield/network.html#CloudCampus

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.


Thank You
www.huawei.com



iMaster NCE-Fabric Open API Introduction

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Foreword

- iMaster NCE-Fabric is a next-generation SDN controller for enterprise DC markets and is a core component of Huawei CloudFabric Solution.
- iMaster NCE-Fabric can centrally manage cloud data center networks and provide automatic mapping from the application network to the physical network, resource pooling, and visualized O&M, helping customers build service-centric dynamic network service scheduling capabilities.
- iMaster NCE-Fabric provides standard RESTful API openness capabilities, which can be used for:
  - Seamless interconnection with customers' service systems, enabling customers to flexibly deploy and schedule network resources based on their service development.
  - Cooperation with partners to develop innovative apps, continuously creating value for enterprise customers.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Objectives

- Upon completion of this course, you will be able to:
  - Understand the basic functions of iMaster NCE-Fabric.
  - Understand constraints and specifications of iMaster NCE-Fabric open APIs.
  - Understand the main open APIs and typical application scenarios of iMaster NCE-Fabric.
  - Complete the example of invoking Neutron APIs for intra-VPC communication.

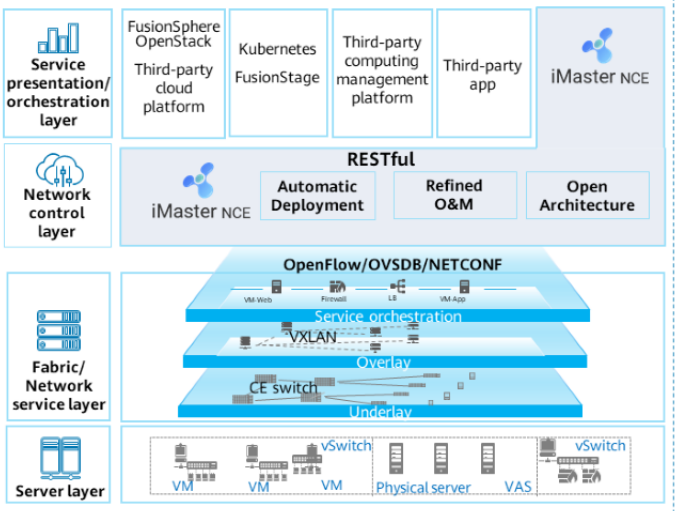
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. **iMaster NCE-Fabric Overview**
2. General Restrictions and Specifications of iMaster NCE-Fabric Open APIs
3. iMaster NCE-Fabric Open APIs and Application Scenarios
4. Example of Invoking Neutron APIs for Intra-VPC Communication

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

iMaster NCE-Fabric Overview



iMaster NCE-Fabric

- Abstracts network resources and services in the northbound and adapts to various devices and networks in the southbound

Automatic

- Application-based automatic network configuration and graphical configuration, implementing minute-level service provisioning
- Zero touch configuration
- VM-level physical path visualization

Refined O&M

- Security resource pool management
- Dynamic and flexible scheduling and orchestration of service resources
- RBAC-based system security protection

Reliable

- Industry's top management capability, achieving geo-redundancy with clusters
- Data consistency check, ensuring a highly reliable network running

Open

- Interconnection with the standard OpenStack/Huawei FusionSphere and Kubernetes/Huawei FusionStage
- Association with third-party computing management platforms
- Interconnection with third-party app systems such as the customer's service system

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

iMaster NCE-Fabric Key Features



Service provisioning in multiple scenarios

- Cloud-network integration
- Network virtualization – computing
- Network virtualization – hosting
- Container network



Multi-fabric network management

- Network overlay
- Hybrid overlay
- Multicast overlay
- IPv6 overlay



Flexible multi-service orchestration

- Service Function Chain (SFC)
- Microsegmentation



Comprehensive refined O&M

- Network-wide resource visualization
- Three-level topology visibility
- Service path visibility
- Loop detection
- One-click emergency

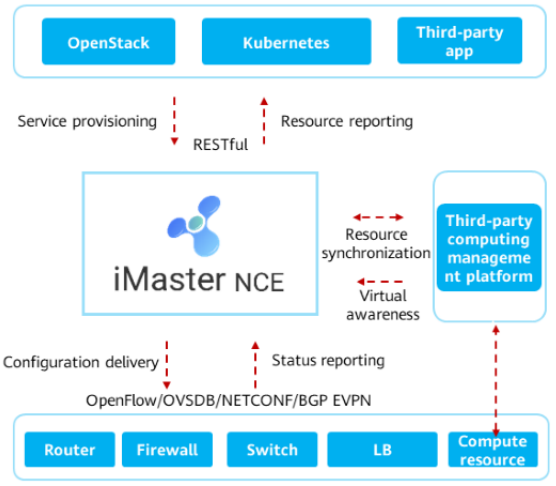


Multi-DC pooling management

- Remote management of DCs using a single controller cluster
- DC management using active and standby controller clusters
- Independent deployment of multiple sets of controller clusters

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

iMaster NCE-Fabric Open Architecture



Northbound ecosystem

- Provides the standard Neutron/RESTful APIs for VPC orchestration to enable abstraction of fabric configuration.
- Provides interoperability with OpenStack and Kubernetes.

Eastbound and westbound ecosystem

- Provides compatibility with third-party computing management platforms and associates with computing resources to implement on-demand VM go-online and elastic scaling.

Southbound ecosystem

- Provides standard device configuration APIs for service provisioning.
- Interconnects with Huawei devices through plug-ins. Third parties can develop plug-ins for interconnection with iMaster NCE-Fabric.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. iMaster NCE-Fabric Overview
2. **General Restrictions and Specifications of iMaster NCE-Fabric Open APIs**
3. iMaster NCE-Fabric Open APIs and Application Scenarios
4. Example of Invoking Neutron APIs for Intra-VPC Communication

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

URL Format

- URL: http(s)://IP:port/controller/product/version/interface type/{module name}
- The value of each field in the URL (product, version, interface type, and module name) is case-sensitive.

Parameter	Description
http(s)	HTTP and HTTPS are supported, and HTTPS is recommended.
IP	Controller IP address
port	Port number. For example, in https://189.200.44.100:18002, 18002 is the port number.
controller	Fixed value
Product	Product name: dc
Version	API version: v2 and v3
Interface type	API type name: neutronapi, logicnetwork, multicast, and so on.
Module name	Service module of an API type, for example, network and subnet under the Neutron API.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HTTP Request Mode/Header Parameters

- HTTP request mode:
  - GET: Obtain detailed information about specified resources. You can choose an Internet media type as needed (for example, XML and JSON). This document uses JSON.
  - PUT: Update a specified element.
  - POST: Treat URI as a resource group and create an element in it.
  - DELETE: Delete a specified element.
- Parameter format: application/json;charset=UTF-8
- HTTP header: The following three parameters must be added to the header before APIs (except the authentication interface) are invoked.
  - Content-Type: format of the data received by the entity header. Value: application/json
  - Accept: type of data received by the request header. Value: application/json
  - X-ACCESS-TOKEN: variable name of the token in the HTTP header. Value: token ID obtained through the authentication API.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Service_name of the Neutron API

- The northbound Neutron API of iMaster NCE-Fabric contains the service_name field. This field is a string containing 1 to 255 bytes. This field specifies the name of a cloud platform and helps iMaster NCE-Fabric identify the cloud platform from which services are delivered.
- The service_name field is mandatory for the creation operation of all modules through the Neutron API.
  - L2-L3: subnet, network, port, router, router interface
  - L4-L7: all services
- The following uses network as an example.

```
{
        "service_name": "openstack11",
        "network":
        {
                "shared": "false",
                "name": "liudongtest",
                "admin_state_up": "true",
                "tenant_id": "6fbe9263116a4b68818cf1edce16bc4"
        }
}
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Response Value

Normal Response Values	Special Response Values
• The status code 201 is returned if the POST operation succeeded. • The status code 204 is returned if the DELETE operation succeeded. • The status code 200 is returned if the GET operation succeeded. • The status code 200 is returned if the PUT operation succeeded.	• The status code 208 is returned if the object ID already exists but the content differs during creation. The status code 201 is returned if the object content duplicates. • The status code 400 is returned if the object ID does not exist during updating. • The status code 404 is returned if the object ID does not exist during query. • The status code 204 is returned if the object ID does not exist during deletion. • The URL format is **context_path/path**. The status code 404 is returned if **context_path** or **path** does not exist.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Authentication API (1)

- As the first API invoked by a third-party application system, the authentication API returns a token ID. Before invoking any other API, the third-party application system must set X-ACCESS-TOKEN in the HTTP header to a token ID. Otherwise, the system will display "403: Authentication failed." when another API is invoked. The token is effective in a certain time period, and needs to be acquired again from the authentication API when the validity period expires.
- URL: https://{ip}:{port}/controller/v2/tokens
- Method: POST
- Request body

Parameter	Type	Mandatory	Description
userName	string	Yes	Name of a northbound API user. The northbound API user is added to iMaster NCE-Fabric using the **admin** account.
password	string	Yes	Password of a northbound API user. The northbound API user is added to iMaster NCE-Fabric using the **admin** account.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Authentication API (2)

- Response body

Parameter	Type	Mandatory	Description
errcode	string	Yes	Returned error code. The value 0 indicates a success. For others, see description of **errmsg**.
errmsg	string	Yes	Information contained in the returned error message.
data	data	Yes	Returned token information.

- data structure

Parameter	Type	Mandatory	Description
expiredDate	date	Yes	Expiration time of the token. Format: 2016-01-13 19:52:40.
token_id	string	Yes	Token ID.

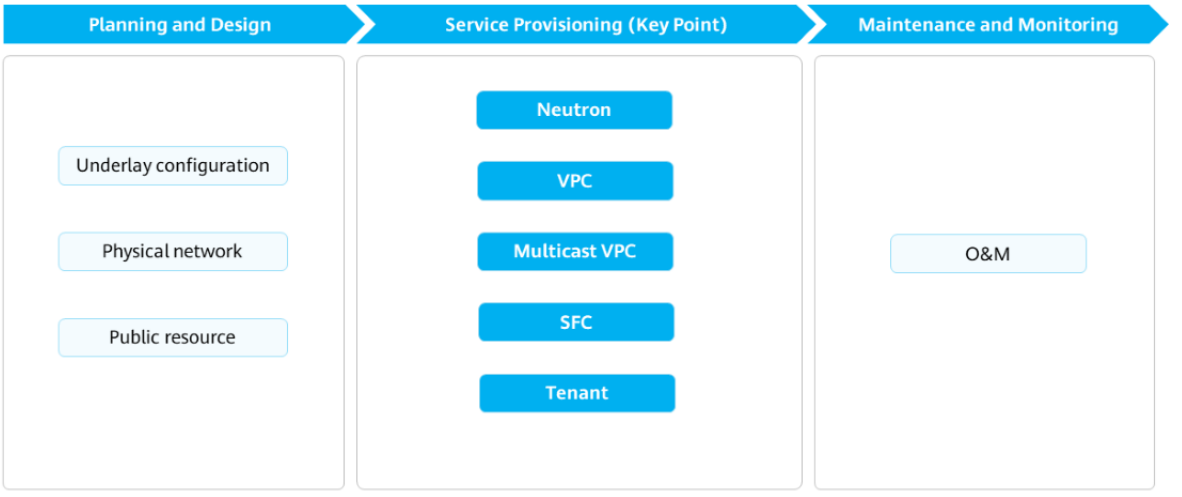
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. iMaster NCE-Fabric Overview
2. General Restrictions and Specifications of iMaster NCE-Fabric Open APIs
3. **iMaster NCE-Fabric Open APIs and Application Scenarios**
   - Northbound Open Neutron APIs
   - Northbound Open APIs for VPC Services
   - Northbound Open APIs for Multicast VPC Services
   - Northbound Open APIs for SFCs
   - Other Open APIs
4. Example of Invoking Neutron APIs for Intra-VPC Communication

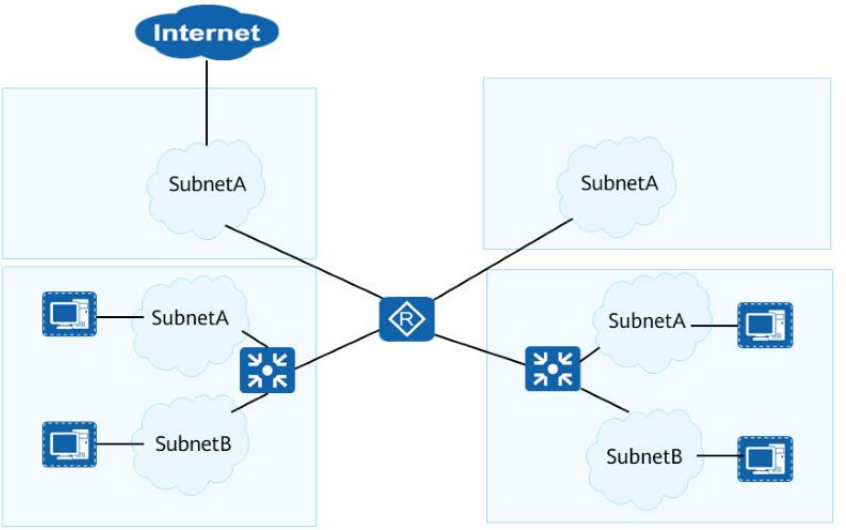
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Overview of iMaster NCE-Fabric Open API



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Typical Basic Networking



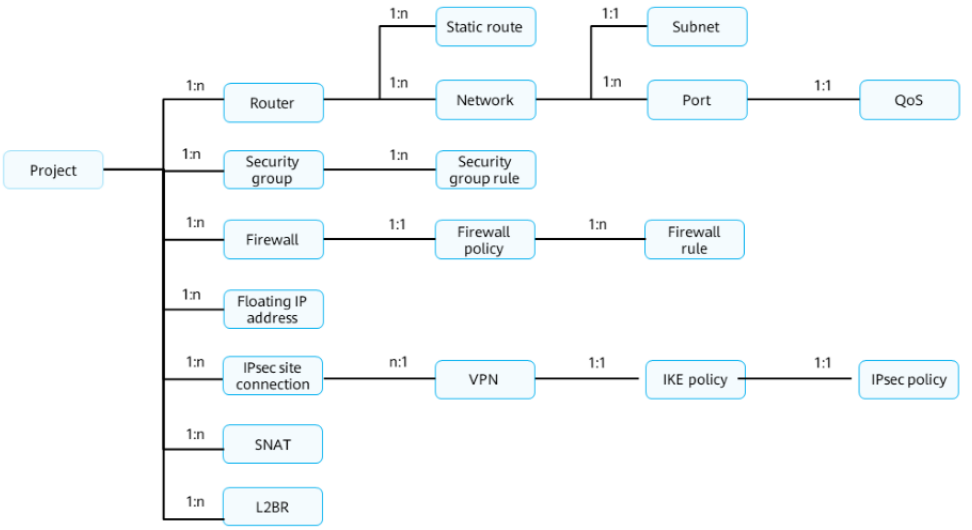
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. iMaster NCE-Fabric Overview
2. General Restrictions and Specifications of iMaster NCE-Fabric Open APIs
3. **iMaster NCE-Fabric Open APIs and Application Scenarios**
   - **Northbound Open Neutron APIs**
   - Northbound Open APIs for VPC Services
   - Northbound Open APIs for Multicast VPC Services
   - Northbound Open APIs for SFCs
   - Other Open APIs
4. Example of Invoking Neutron APIs for Intra-VPC Communication

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Mapping Between Neutron Service Modules



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- When iMaster NCE-Fabric is interconnected with a cloud platform, the cloud platform invokes northbound APIs to provision services. Project, router, firewall, and security group in the figure are all network models of the cloud platform.
- Neutron APIs are encapsulated based on the OpenStack Neutron model and match the network service models in the industry. Neutron APIs are recommended if you are familiar with the OpenStack Neutron models.
- Projects are created on the cloud platform without invoking the RESTful interface.
- For details about OpenStack principles, see the OpenStack Foundation at https://www.openstack.org/ or learn Huawei HCIP-Cloud Computing-OpenStack at https://e.huawei.com/cn/talent/.

Mapping Between Neutron Service Modules and RESTful APIs (1)

Service Module	Description	Access Path
Router	vRouters are used for interconnection between subnets and accessing external networks.	https://{ip}:{port}/controller/dc/v2/neutronapi/routers
Static route	A static route is manually configured by the administrator because it cannot automatically adapt to network topology changes.	https://{ip}:{port}/controller/dc/v2/neutronapi/routers/{router_id}/add_routes
Network	An isolated Layer 2 network domain. Hosts on one network are visible at Layer 2. A user can have multiple networks. Subnets and ports must be associated with networks which are the basis. VXLAN and VLAN are supported.	https://{ip}:{port}/controller/dc/v2/neutronapi/networks
Subnet	A subnet indicates a network segment. Each subnet has a CIDR value and is associated with a network. The IP address can be selected from the CIDR or a specified pool. A subnet may have a gateway, a group of DNSs, and host routes. Subnets are invisible at Layer 2 and communicate with each other through Layer 3 routing on a Layer 3 gateway (router). A network supports only one subnet.	https://{ip}:{port}/controller/dc/v2/neutronapi/subnets
Port	A port belongs to a network and indicates an access point which is used to connect to VMs or routers. The NICs of VMs are connected to the port and the VMs access the network using the port. If an IP address is bound to the port, the port belongs to a subnet.	https://{ip}:{port}/controller/dc/v2/neutronapi/ports
QoS	Quality of service. When a network is overloaded or congested, QoS ensures that important services can be processed in a timely manner and the network can run efficiently. Currently, QoS supports only rate limiting.	https://{ip}:{port}/controller/dc/v2/neutronapi/qoss
Security group	Trusted VMs communicate with each other through security groups on OpenStack. VMs in the same security group use the same security policies. A security group works on a VM port and is a port-level vFW. It filters incoming and outgoing port traffic.	https://{ip}:{port}/controller/dc/v2/neutronapi/security-groups
Security group rule	A policy for implementing traffic filtering. One security group can have multiple security group rules.	https://{ip}:{port}/controller/dc/v2/neutronapi/security-group-rules

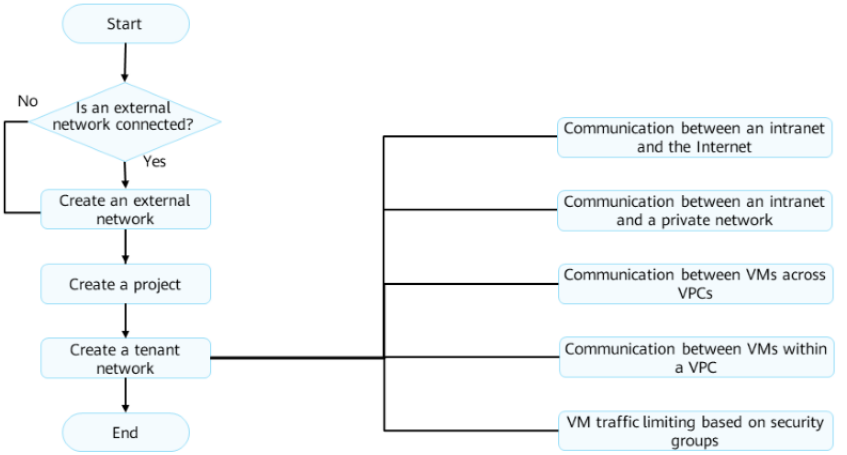
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Mapping Between Neutron Service Modules and RESTful APIs (2)

Service Module	Description	Access Path
Firewall	A firewall contains a series of components deployed among multiple networks or security zones, and monitors, restricts, and modifies data flows passing through it to ensure network security.	https://{ip}:{port}/controller/dc/v2/neutronapi/fw/firewalls
Firewall policy	In the Neutron system, a firewall policy contains a group of rules arranged in a specific sequence.	https://{ip}:{port}/controller/dc/v2/neutronapi/fw/firewall_policies
Firewall rule	Firewall rules are used to determine whether to allow traffic to pass through.	https://{ip}:{port}/controller/dc/v2/neutronapi/fw/firewall_rules
Floating IP address	Floating IP addresses can be dynamically allocated to VMs and translated during communication between internal and external networks.	https://{ip}:{port}/controller/dc/v2/neutronapi/floatingips
IPsec site connection	Connection objects define the routing mode, key mode, authentication mode, and even the parameters and service subnets of the connected sites. IPsec tunnels are set up between gateways to provide a secure connection between LANs.	https://{ip}:{port}/controller/dc/v2/neutronapi/vpn/ipsec-site-connections
VPN	A virtual private network (VPN) is built on a public network for encrypted communication. The VPN gateway enables remote access through data encryption and address translation.	https://{ip}:{port}/controller/dc/v2/neutronapi/vpn/vpnservices
IKE policy	An IKE policy defines parameters for IKE negotiation used during the first phase.	https://{ip}:{port}/controller/dc/v2/neutronapi/vpn/ikepolicies
IPsec policy	An IPsec policy defines ESP negotiation parameters used during the second phase.	https://{ip}:{port}/controller/dc/v2/neutronapi/vpn/ipsecpolicies
Internal router interface	An internal router interface is connected to a router. This interface functions as the gateway of a subnet.	https://{ip}:{port}/controller/dc/v2/neutronapi/routerinterface/{router_id}/add_router_interface
SNAT	Source network address translation (SNAT) is used to translate the source address of an IP data packet to another address.	https://{ip}:{port}/controller/dc/v2/neutronapi/snat
L2BR	Layer 2 Bridge (L2BR) implements Layer 2 communication between VXLAN VMs and VLAN physical servers.	https://{ip}:{port}/controller/dc/v2/neutronapi/l2brs

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Configuration Process

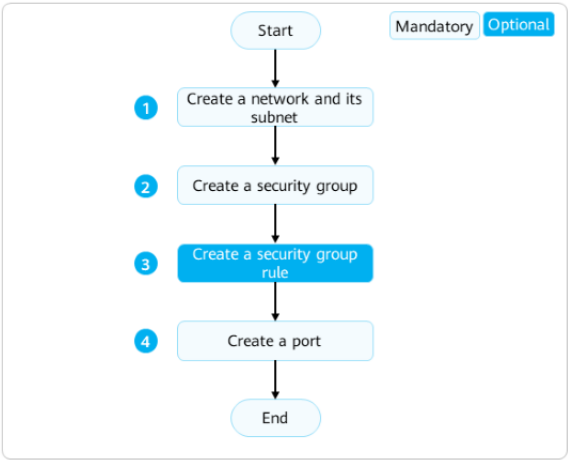


- Only the tenant network is created by invoking the API. Other steps can be performed on the cloud platform.
- The process of creating a tenant network varies according to users' service requirements.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Traffic Limiting of VMs Using Security Groups

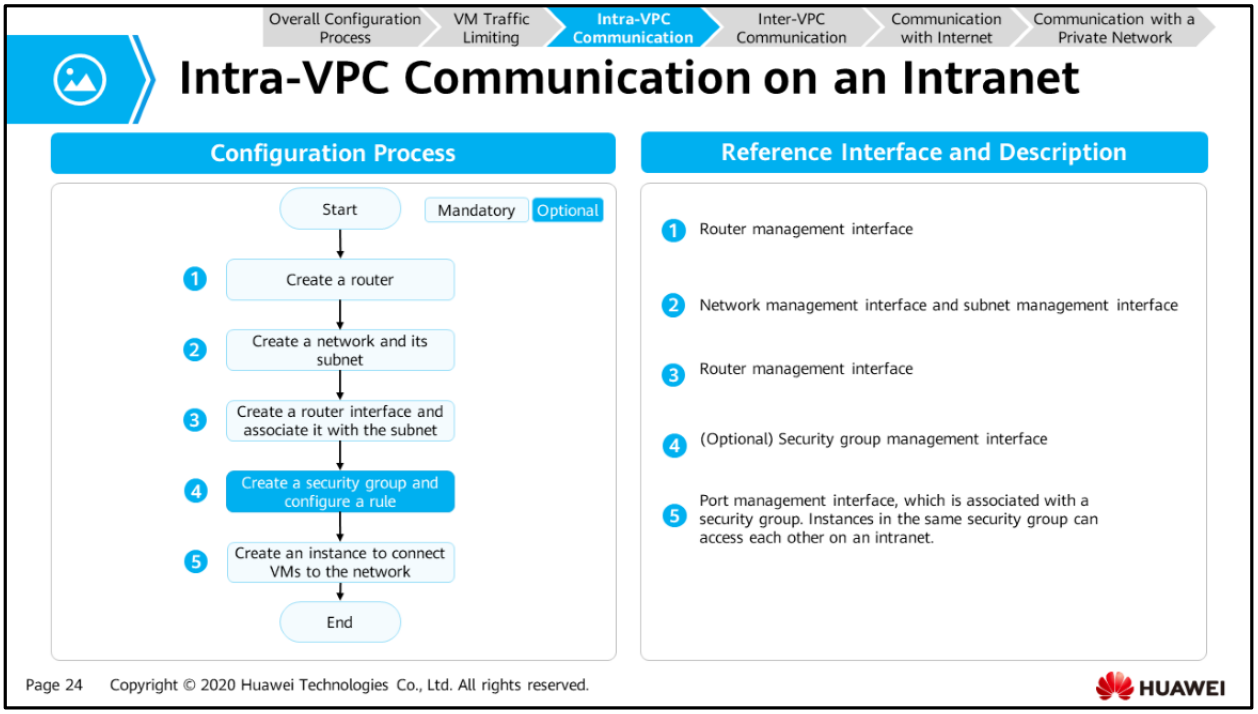
Configuration Process



Reference Interface and Description

1. Network management interface and subnet management interface
2. Security group management interface
3. (Optional) Security group rule management interface
4. Port management interface, which is associated with a security group. Instances in the same security group can access each other on an intranet.

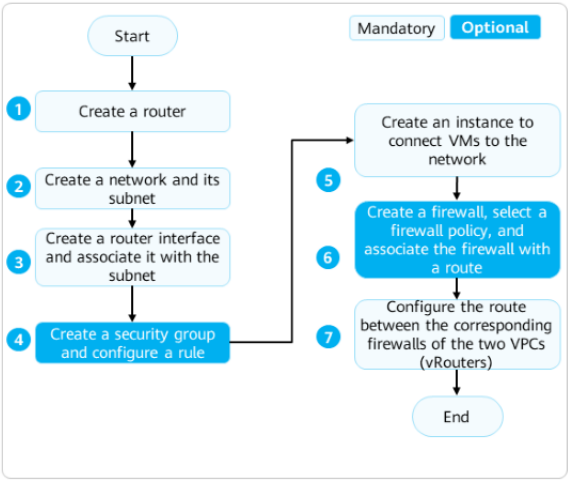
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



- A virtual private cloud (VPC) contains logical NEs abstracted from a physical network. These logical Nes are orchestrated based on service requirements to form a virtual network. Different VPCs are logically isolated but share the same physical network. In this way, physical network resources can be shared after being pooled.

Inter-VPC Communication on an Intranet

Configuration Process



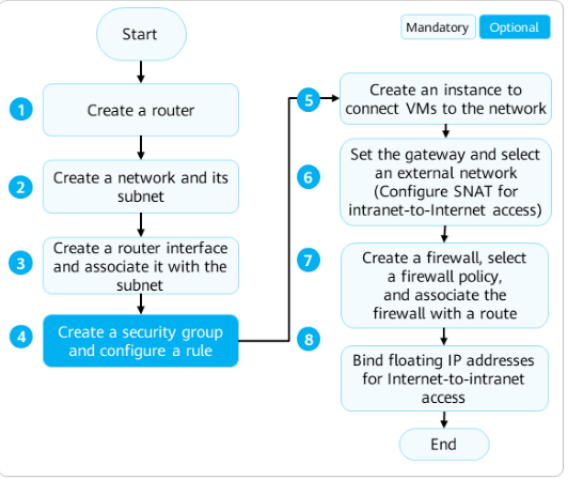
Reference Interface and Description

1. Router management interface
2. Network management interface and subnet management interface
3. Router management interface
4. (Optional) Security group management interface
5. Port management interface, which is associated with a security group. Instances in the same security group can access each other on an intranet.
6. (Optional) Firewall management interface
7. Static route management interface

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Communication Between an Intranet and the Internet

Configuration Process



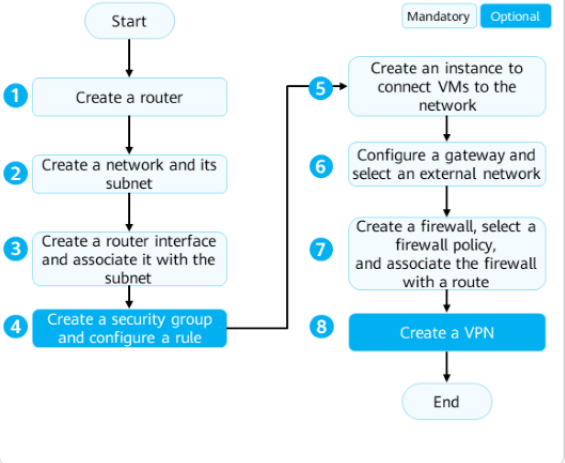
Reference Interface and Description

1. Router management interface
2. Network management interface and subnet management interface
3. Router management interface
4. (Optional) Security group management interface
5. Port management interface, which is associated with a security group. Instances in the same security group can access each other on an intranet.
6. SNAT interface
7. Firewall management interface
8. Floating IP address interface

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Communication Between an Intranet and an External Private Network Through IPsec VPN

Configuration Process



Reference Interface and Description

1. Router management interface
2. Network management interface and subnet management interface
3. Router management interface
4. (Optional) Security group management interface
5. Port management interface, which is associated with a security group. Instances in the same security group can access each other on an intranet.
6. Distributed router management interface
7. Firewall management interface
8. (Optional) IPsec VPNaaS interface. This step is required only when VPN access is involved.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Section Summary

- Neutron APIs are encapsulated based on the OpenStack Neutron model and match network service models in the industry. Neutron APIs are recommended if you are familiar with the OpenStack Neutron model.
- After receiving the request for calling Neutron APIs, iMaster NCE-Fabric invokes the corresponding VPC APIs based on the mapping between the Neutron service model and iMaster NCE-Fabric service model.
- Learn the mapping between Neutron service modules and RESTful APIs.
- Use Neutron APIs to orchestrate VPCs to implement the following typical service scenarios:
  - Traffic limiting of VMs using security groups
  - Intra-VPC communication on an intranet
  - Inter-VPC communication on an intranet
  - Communication between an intranet and the Internet
  - Communication between an intranet and an external private network through IPsec VPN

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

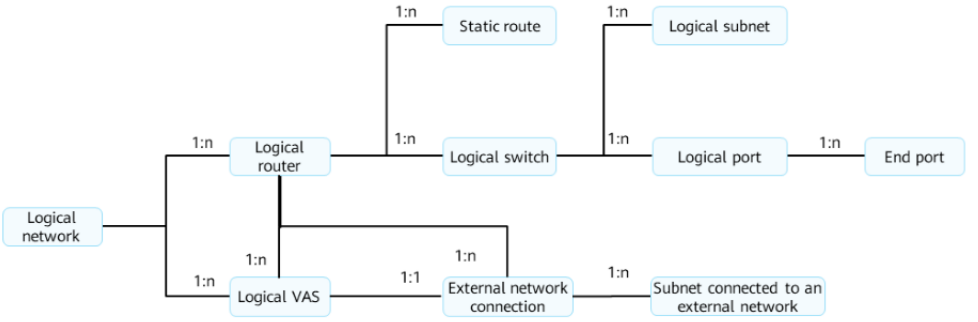
For more API information, see "Secondary Development" in the iMaster NCE-Fabric Product Documentation.

Contents

1. iMaster NCE-Fabric Overview
2. General Restrictions and Specifications of iMaster NCE-Fabric Open APIs
3. **iMaster NCE-Fabric Open APIs and Application Scenarios**
   - Northbound Open Neutron APIs
   - **Northbound Open APIs for VPC Services**
   - Northbound Open APIs for Multicast VPC Services
   - Northbound Open APIs for SFCs
   - Other Open APIs
4. Example of Invoking Neutron APIs for Intra-VPC Communication

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Mapping Between Main Service Modules of a VPC



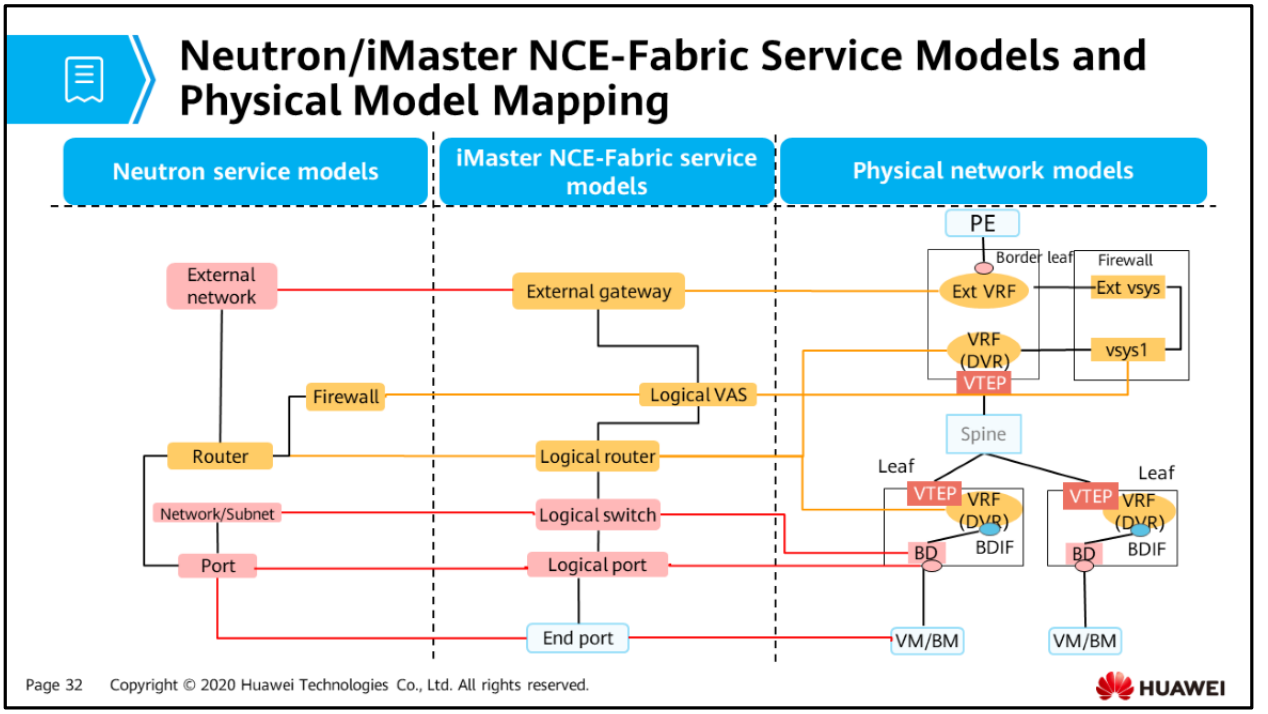
- The northbound open APIs for VPC services are encapsulated based on the iMaster NCE-Fabric service models. The content of the APIs is the same as that of the APIs invoked when performing operations on the iMaster NCE-Fabric GUI. Third parties can use the APIs based on the site requirements.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Mapping Between VPC Service Modules and the RESTful APIs

Service Module	Description	Access Path
Logical network	A logical network (VPC) provides isolated VMs and network environments to meet the network isolation requirements of different departments. Each VPC can provide independent vFWs, elastic IP addresses, security groups, super VLANs, IPsec VPNs, and NAT gateways. (Some functions can be provided by vFWs.)	https://{ip}:{port}/controller/dc/v3/logicnetwork/networks
Logical router	A logical router implements interworking between subnets and access to external networks.	https://{ip}:{port}/controller/dc/v3/logicnetwork/routers
Static route	A static route is manually configured by the administrator because it cannot automatically adapt to network topology changes.	https://{ip}:{port}/controller/dc/v3/logicnetwork/routers/router/{id}/route-sets
Logical switch	An isolated Layer 2 network domain. Hosts on one network are visible at Layer 2. A user can have multiple logical switches. Logical subnets and logical ports must be associated with logical switches.	https://{ip}:{port}/controller/dc/v3/logicnetwork/switchs
Logical subnet	A logical subnet indicates a network segment. Each subnet must have a CIDR and be associated with a network. Subnets are invisible at Layer 2 and communicate with each other through Layer 3 routes by passing through a Layer 3 gateway. One logical switch supports only one logical subnet.	https://{ip}:{port}/controller/dc/v3/logicnetwork/subnets
Logical port	A logical port belongs to a logical switch and is classified into User Network Interface (UNI) or Network to Network Interface (NNI). UNIs are used to connect user VMs/BMs to a VXLAN network. NNIs are used to connect VXLAN networks.	https://{ip}:{port}/controller/dc/v3/logicnetwork/ports
End port	An end port indicates a VM or BM on the user side and is associated with a logical port to access a VXLAN network.	https://{ip}:{port}/controller/dc/v3/logicnetwork/endports
Logical VAS	A unified abstracted logical unit of L4-L7 VASs, such as a firewall and an LB. Logical VASs need to be associated with logical routers and logical networks.	https://{ip}:{port}/controller/dc/v3/servicenetwork/vass
External network connection	A VPC can be associated with multiple external networks, which can be configured as a public external network and multiple private external networks. Different external networks connect to different service egresses.	https://{ip}:{port}/controller/dc/v3/servicenetwork/external-connects
Subnet connected to an external network	Specifies a logical subnet used to connect to an external network, that is, CIDR.	https://{ip}:{port}/controller/dc/v3/servicenetwork/external-connects/external-connect/{id}/action/add-cidrs

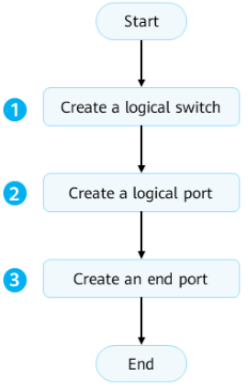
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



- Border leaf (BL): A border leaf node is connected to an egress device.

Intra-subnet Communication in a VPC

Configuration Process



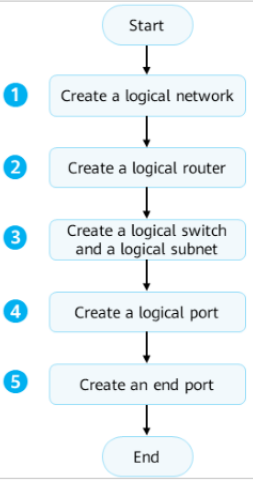
Reference Interface and Description

1. Logical switch management interface, which is used to create a large Layer 2 BD on a VXLAN.
2. Logical port management interface. Multiple Layer 2 sub-interfaces can be created based on service scenarios.
3. End port management interface. End ports correspond to VMs/BMs. Multiple end ports can be created based on service scenarios and associated with their logical ports.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Inter-subnet Communication in a VPC

Configuration Process



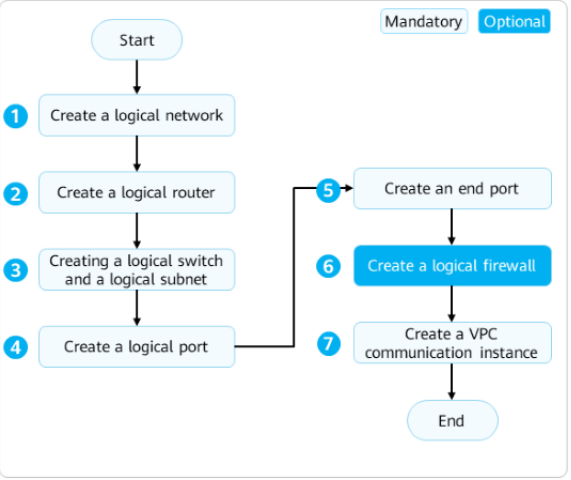
Reference Interface and Description

1. Logical network management interface used to create a VPC.
2. Logical router management interface used to create a VRF.
3. Logical switch management interface and logical subnet management interface. Multiple logical switches and subnets can be created and logical ports can be created on logical switches.
4. Logical port management interface.
5. End port management interface.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Inter-VPC Communication on an Intranet

Configuration Process



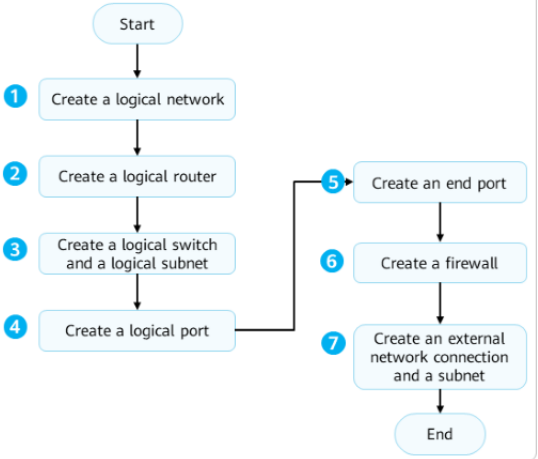
Reference Interface and Description

1. Logical network management interface. Create two VPCs or two VRFs under a VPC based on service requirements. VPC communication and intra-VPC VRF mutual access are supported.
2. Logical router management interface used to create a VRF.
3. Logical switch management interface and logical subnet management interface.
4. Logical port management interface.
5. End port management interface.
6. (Optional) Logical VAS management interface. Select **Firewall**, associate the logical VAS with the logical router, and create a logical firewall instance.
7. VPC communication instance management interface. You need to associate the VPC communication instance with the source and destination logical routers. You can also associate it with the source and destination logical firewalls.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Communication Between an Intranet and the Internet

Configuration Process



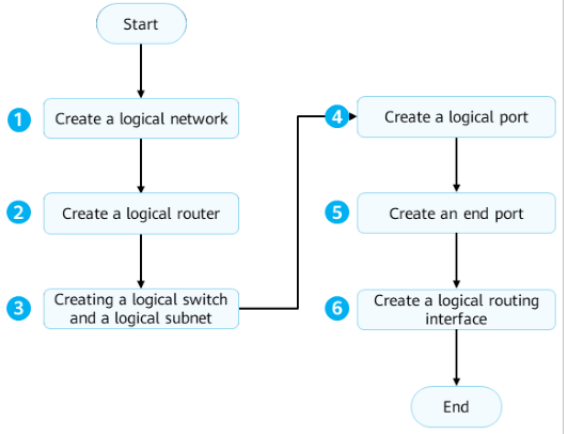
Reference Interface and Description

1. Logical network management interface used to create a VPC.
2. Logical router management interface used to create a VRF.
3. Logical switch management interface and logical subnet management interface.
4. Logical port management interface.
5. End port management interface.
6. (Optional) Logical VAS management interface. Select **Firewall**, associate the logical VAS with the logical network and the logical router, and create a logical firewall instance.
7. External network connection management interface and subnet interface. The external network connection is associated with the logical router, logical firewall, and Layer 3 shared egress gateway. SNAT is enabled when an intranet accesses the Internet.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Communication Between an Intranet and an External Private Network

Configuration Process



Reference Interface and Description

1. Logical network management interface used to create a VPC.
2. Logical router management interface used to create a VRF.
3. Logical switch management interface and logical subnet management interface.
4. Logical port management interface.
5. End port management interface.
6. Logical routing interface management interface, which is associated with the logical router and Layer shared egress gateway.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Section Summary

- APIs for VPC services are encapsulated based on the iMaster NCE-Fabric service models. The content of the APIs is the same as that of the APIs invoked when performing operations on the iMaster NCE-Fabric GUI. Third parties can use the APIs based on the site requirements.
- Learn mapping between Neutron service models, iMaster NCE-Fabric logical models, and physical models.
- Learn mapping Between VPC Service Modules and the RESTful APIs.
- Use VPC APIs for orchestration to implement the following service scenarios:
  - Intra-subnet Communication in a VPC
  - Inter-subnet Communication in a VPC
  - Inter-VPC Communication on an Intranet
  - Communication Between an Intranet and the Internet
  - Communication Between an Intranet and an External Private Network

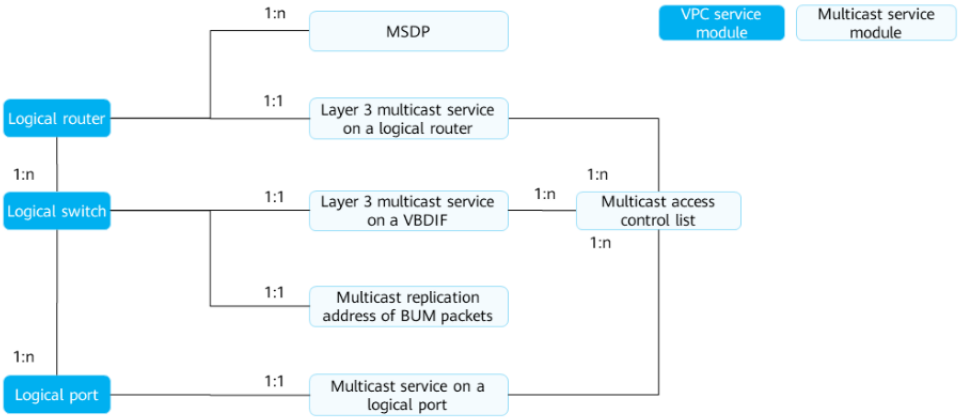
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. iMaster NCE-Fabric Overview
2. General Restrictions and Specifications of iMaster NCE-Fabric Open APIs
3. **iMaster NCE-Fabric Open APIs and Application Scenarios**
   - Northbound Open Neutron APIs
   - Northbound Open APIs for VPC Services
   - **Northbound Open APIs for Multicast VPC Services**
   - Northbound Open APIs for SFCs
   - Other Open APIs
4. Example of Invoking Neutron APIs for Intra-VPC Communication

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Mapping Between Service Modules of a Multicast VPC (IPv4)



- Use the Neutron API or VPC API to create a VPC instance, and then use a multicast VPC API to create a multicast overlay service.
- The following scenarios are supported: both the multicast source and requester are inside a DC, the multicast source is inside a DC and the requester is outside the DC, the requester is inside a DC and the multicast source is outside the DC.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

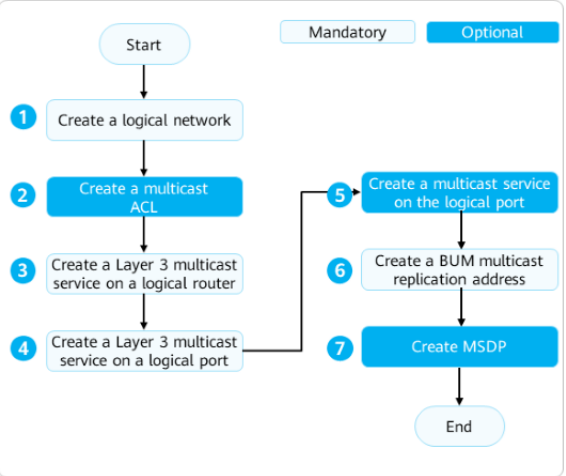
Mapping Between VPC Service Modules and RESTful APIs

Service Module	Description	Access Path
Layer 3 multicast service on a logical router	The multicast capability is enabled on a logical router. The MVPN, internal RP, multicast group replication address, and SSM mapping are also configured.	https://{ip}:{port}/controller/dc/v3/multicast/logicrouter
Layer 3 multicast service on a logical port	PIM and IGMP protocols and policies are configured based on VBDIFs of logical switches, VBDIFs of external networks, and logical routing interfaces.	https://{ip}:{port}/controller/dc/v3/multicast/logicinterface
MSDP	MSDP is used by a border leaf node in a DC to establish an MSDP peer relationship with an external RP/CRP to synchronize SA information.	https://{ip}:{port}/controller/dc/v3/multicast/msdp
Multicast replication address of BUM packets	Multicast replication addresses of BUM packets are configured based on logical switches. Multicast replication reduces bandwidth usage between any two VTEPs based on ingress replication, optimizing traffic forwarding.	https://{ip}:{port}/controller/dc/v3/multicast/bum
Multicast service on a logical port	A multicast group filtering policy is configured based on logical ports.	https://{ip}:{port}/controller/dc/v3/multicast/logicport
Multicast ACL	An ACL is configured to control PIM-SM and IGMP policies.	https://{ip}:{port}/controller/dc/v3/multicast/template/acl

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Multicast VPC Service Scenario

Configuration Process



Reference Interface and Description

1. Use the Neutron API or VPC API to create a VPC and the corresponding network object instances. (Optional) Create an external network, and enable the multicast capability for the tenant and VPC to which the external network belongs.
2. (Optional) Multicast ACL management interface.
3. Layer 3 multicast service management interface of a logical router, which is associated with a logical router and configured with a multicast replication address and an internal RP address.
4. Layer 3 multicast service management interface of a logical port, which is associated with a logical switch and an external network.
5. (Optional) Multicast service management interface of a logical port, which is used to set a multicast group filtering policy.
6. Management interface for a BUM multicast replication address, which is associated with a logical switch. Either this interface or the interface created in step 4 can be selected.
7. (Optional) MSDP management interface, which is used by the border leaf node to establish an MSDP peer relationship with the external RP/CRP.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Section Summary

- Layer 3 IPv4 multicast is supported in the distributed gateway scenario.
- The following scenarios are supported: both the multicast source and requester are inside a DC, the multicast source is inside a DC and the requester is outside the DC, the requester is inside a DC and the multicast source is outside the DC.
- The mapping between multicast VPC service modules and RESTful APIs is listed.
- In the scenario where a multicast VPC is used to orchestrate multicast services, the Neutron API or VPC API needs to be used to create a VPC and the corresponding network object instances. The multicast capability needs to be enabled for the tenant and VPC of the network object instances.

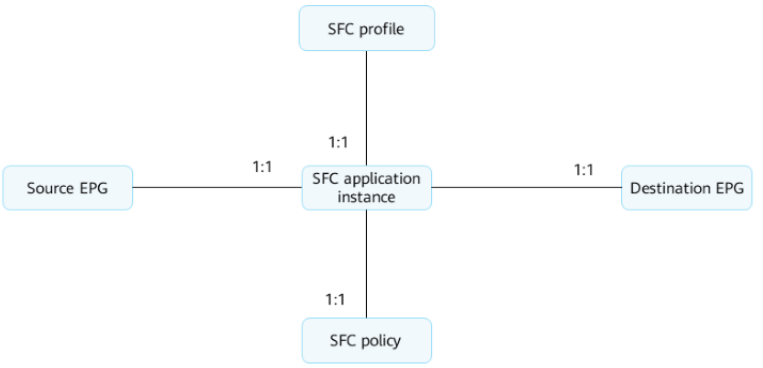
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. iMaster NCE-Fabric Overview
2. General Restrictions and Specifications of iMaster NCE-Fabric Open APIs
3. **iMaster NCE-Fabric Open APIs and Application Scenarios**
   - Northbound Open Neutron APIs
   - Northbound Open APIs for VPC Services
   - Northbound Open APIs for Multicast VPC Services
   - **Northbound Open APIs for SFCs**
   - Other Open APIs
4. Example of Invoking Neutron APIs for Intra-VPC Communication

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Mapping Between Main Service Modules in an SFC



- Use the Neutron API or VPC API to create a VPC instance, and then use the SFC API to create an SFC service.

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

- Service Function Chain (SFC)

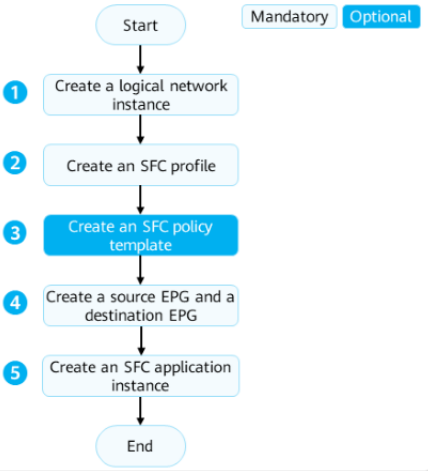
Mapping Between SFC Modules and RESTful APIs

Service Module	Description	Access Path
SFC profile	An SFC profile defines the service functions (SFs) through which packets pass between the SFC ingress (source EPG) and SFC egress (destination EPG) and their sequences. In addition, the SFC profile can be reused.	https://{ip}:{port}/controller/dc/v3/sfco/sfcs
SFC policy	A policy is used to ensure that only the EPG traffic that meets the requirements enters an SFC. The policy template can be reused for creating a policy.	https://{ip}:{port}/controller/dc/v3/sfco/contracts
EPG	An End Point Group (EPG) is a group of devices with the same characteristics. When creating an SFC, you can configure EPGs to determine the start point and end point for the SFC.	https://{ip}:{port}/controller/dc/v3/sfco/epgs
SFC application instance	An SFC application instance is defined based on the service access path. An SFC consists of the source EPG, SFs, and destination EPG. The source EPG is the ingress of an SFC, while the destination EPG is the egress. The service data to be processed enters the service function path (SFP) from the source EPG, and the processing result is obtained from the destination EPG.	https://{ip}:{port}/controller/dc/v3/sfco/scapps

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

SFC Scenario

Configuration Process



Reference Interface and Description

1. Use the Neutron API or VPC API to create a VPC and the corresponding network object instances.
2. SFC profile management interfaces.
3. (Optional) SFC policy management interfaces.
4. EPG management interfaces.
5. SFC application instance management interface, which is associated with an SFC profile, source EPG, and destination EPG.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Section Summary

- Microsegmentations and SFCs use the same service model. The API orchestration logic used for creating a microsegmentation is the same as that used for creating an SFC.
- Learn the mapping between SFC service modules and RESTful APIs.
- In the scenario where an SFC API is used to orchestrate an SFC and a microsegmentation, you need to use a Neutron API or VPC API to create a VPC and the corresponding network object instances.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. iMaster NCE-Fabric Overview
2. General Restrictions and Specifications of iMaster NCE-Fabric Open APIs
3. **iMaster NCE-Fabric Open APIs and Application Scenarios**
   - Northbound Open Neutron APIs
   - Northbound Open APIs for VPC Services
   - Northbound Open APIs for Multicast VPC Services
   - Northbound Open APIs for SFCs
   - **Other Open APIs**
4. Example of Invoking Neutron APIs for Intra-VPC Communication

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Mapping Between Other Service Modules and RESTful APIs

Interface Type	Service Module	Description	Access Path
Underlay configuration	M-LAG	Configure the trunk module, status, and trunk member interfaces.	https://{ip}:{port}/sdn/acdcn/v3/underlay/ethTrunk-configure/.....
	Batch configuration	Deliver configurations to multiple devices in batches.	https://{ip}:{port} /controller/dc/v3/underlay/commands/execute
	ZTP	Automatically deploy underlay functions.	https://{ip}:{port}/sdn/acdcn/v3/underlay/ztp/.....
Physical network	Physical topology	Query the device list, host link list, host list, device link list, port list, resource pool topology, and DCN network topology.	https://{ip}:{port}/acdcn/v3/topoapi/dcntopo/.....
	Fabric resource pool	Manage fabric resource pools.	https://{ip}:{port}/controller/dc/v3/physicalnetwork/fabricresource/.....
Public resource	External gateway	An external gateway is used for communication between VMs/BMs in a DC and the external network.	https://{ip}:{port}/controller/dc/v3/publicservice/external-gateways
Tenant	Tenant	Network administrators need to create a tenant and allocate network resources to the tenant so that the tenant can log in to the system using the associated account to view the network resource usage and adjust the existing services.	https://{ip}:{port}/controller/dc/v3/tenants
O&M	Connectivity test	MAC ping is used to check whether the ARP detection system is normal. IP ping is used to check the network reachability between a VM, BM, container, or device and the destination IP address is normal.	https://{ip}:{port}/controller/dc/v3/oam/ping
	Path detection	Single-path detection: Check the actual physical path of a specific service flow between VMs, BMs, or devices. Multi-path detection: Check actual physical paths between two NVE devices.	https://{ip}:{port}/controller/dc/v3/oam/trace/.....
	Emergency plan	Atomic template: An atomic template supports operations on only one type of objects. Integration template: An integration template can be integrated with a maximum of 10 atomic templates of different types which are executed in sequence.	https://{ip}:{port}/controller/dc/v3/oam/emergency-plan/.....

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

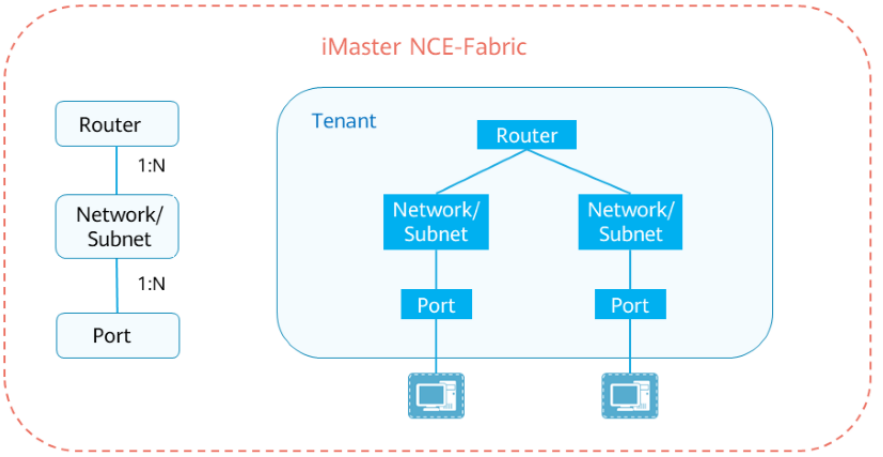
1. iMaster NCE-Fabric Overview
2. General Restrictions and Specifications of iMaster NCE-Fabric Open APIs
3. iMaster NCE-Fabric Open APIs and Application Scenarios
4. **Example of Invoking Neutron APIs for Intra-VPC Communication**

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

Example

- In this example, the Neutron API of iMaster NCE-Fabric is invoked to implement access between intranets of a router.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Creating a Router

Url	https://ip:18002/controller/dc/v2/neutronapi/routers
Method	post
RequestBody	{ "router": { "tenant_id": "f9943c4c-a1c9-58b4-1fac-110000000005", "admin_state_up": "true", "name": "router_yao", "id": "f9943c4c-a1c9-58b4-1fac-110000000004", "status": "ACTIVE" }, "service_name": "FusionSphere" }

Tenant ID, which is specified and delivered by the cloud platform

Virtual router ID, which is specified and delivered by the cloud platform

Cloud platform name

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Creating Intranet 1

Url	https://ip:18002/controller/dc/v2/neutronapi/networks
Method	post
RequestBody	{ "service_name": "FusionSphere", "network": { "tenant_id": "f9943c4c-a1c9-58b4-1fac-110000000005", "shared": "false", "provider:physical_network": "physnet1", "provider:segmentation_id": 2009, "provider:network_type": "vlan" , "router:external": "false", "admin_state_up": "true", "name": "net_yao", "id": "f9943c4c-a1c9-58b4-1fac-110000000009", "status": "ACTIVE" } }

Whether the network is an external network. In this scenario, the value must be false.

Network ID, which is specified and delivered by the cloud platform.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Creating a Subnet for Intranet 1

Url	https://ip:18002/controller/dc/v2/neutronapi/subnets
Method	post

RequestBody

```
{
   "subnet": {
      "tenant_id": "f9943c4c-a1c9-58b4-1fac-110000000005",
      "enable_dhcp": "false",
      "allocation_pools": [
         {
            "start": "172.172.1.2",
            "end": "172.172.1.254"
         }
      ],
      "network_id": "f9943c4c-a1c9-58b4-1fac-110000000009",
      "ip_version": 4,
      "name": "sub_yao",
      "cidr": "172.172.1.0/24",
      "gateway_ip": "172.172.1.1",
      "id": "f9943c4c-a1c9-58b4-1fac-110000000010"
   },
   "service_name": "FusionSphere"
}
```

Available IP address pool of the subnet

Network ID. The value is the ID of intranet 1.

Subnet ID, which is specified and delivered by the cloud platform.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Creating a Port for Intranet 1

Url	https://ip:18002/controller/dc/v2/neutronapi/ports
Method	post

RequestBody

```
{
   "port": {
      "tenant_id": "f9943c4c-a1c9-58b4-1fac-110000000005",
      "mac_address": "fa:16:3e:9e:ff:55",
      "id": "f9943c4c-a1c9-58b4-1fac-110000000011",
      "device_id": "f9943c4c-a1c9-58b4-1fac-110000000004",
      "device_owner": "network:router_interface",
      "network_id": "f9943c4c-a1c9-58b4-1fac-110000000009",
      "fixed_ips": [
         {
            "subnet_id": "f9943c4c-a1c9-58b4-1fac-110000000010",
            "ip_address": " 172.172.1.1 "
         }
      ],
      "name": ""
   },
   "service_name": "FusionSphere"
}
```

Port ID, which is specified and delivered by the cloud platform.

ID of the logical router. The value must be the same as that created previously.

Network ID. The value is the ID of intranet 1.

Subnet ID. The value is the subnet ID of intranet 1.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Creating Intranet 2

Url	https://ip:18002/controller/dc/v2/neutronapi/networks
Method	post

RequestBody

```
{
   "service_name": "FusionSphere",
   "network": {
      "tenant_id": "f9943c4c-a1c9-58b4-1fac-110000000005",
      "shared": "false",
      "provider:physical_network": "physnet1",
      "provider:segmentation_id": 2010,
      "provider:network_type": “vlan" ,
      "router:external": "false",
      "admin_state_up": "true",
      "name": "net_yao",
      "id": "f9943c4c-a1c9-58b4-1fac-110000000012",
      "status": "ACTIVE"
   }
}
```

Whether the network is an external network. In this scenario, the value must be false.

Network ID, which is specified and delivered by the cloud platform.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Creating a Subnet for Intranet 2

Url	https://ip:18002/controller/dc/v2/neutronapi/subnets
Method	post

RequestBody

```
{
    "subnet": {
        "tenant_id": "f9943c4c-a1c9-58b4-1fac-110000000005",
        "enable_dhcp": "false",
        "allocation_pools": [
            {
                "start": "172.172.2.2",
                "end": "172.172.2.254"
            }
        ],
        "network_id": "f9943c4c-a1c9-58b4-1fac-110000000011",
        "ip_version": 4,
        "name": "sub_yao",
        "cidr": "172.172.2.0/24",
        "gateway_ip": "172.172.2.1",
        "id": "f9943c4c-a1c9-58b4-1fac-110000000012"
    },
    "service_name": "FusionSphere"
}
```

Available IP address pool of the subnet

Network ID. The value is the ID of intranet 2.

Subnet ID, which is specified and delivered by the cloud platform.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Creating a Port for Intranet 2

Url	https://ip:18002/controller/dc/v2/neutronapi/ports
Method	post

RequestBody

```
{
    "port": {
        "tenant_id": "f9943c4c-a1c9-58b4-1fac-110000000005",
        "mac_address": "fa:16:3e:9e:ff:55",
        "id": "f9943c4c-a1c9-58b4-1fac-110000000013",
        "device_id": "f9943c4c-a1c9-58b4-1fac-110000000004",
        "device_owner": "network:router_interface",
        "network_id": "f9943c4c-a1c9-58b4-1fac-110000000011",
        "fixed_ips": [
            {
                "subnet_id": "f9943c4c-a1c9-58b4-1fac-110000000012",
                "ip_address": "172.172.2.1"
            }
        ],
        "name": ""
    },
    "service_name": "FusionSphere"
}
```

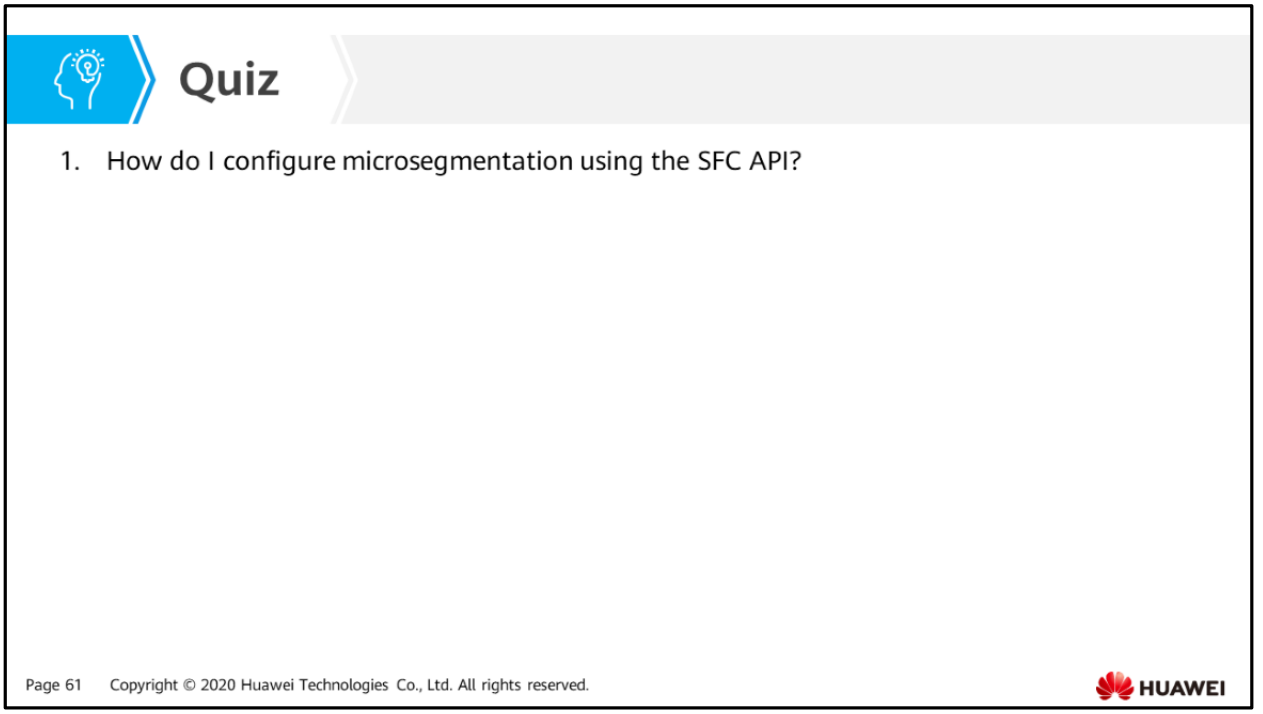
- Port ID, which is specified and delivered by the cloud platform.
- ID of the logical router. The value must be the same as that created previously.
- Network ID. The value is the ID of intranet 2.
- Subnet ID. The value is the subnet ID of intranet 2.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Summary

- This example demonstrates how to invoke the Neutron API to implement inter-network segment access within a VPC.
- Before demonstrating this example, ensure that the fabric network and tenant have been created.
- When designing the API invoking process in actual projects, you can refer to the configuration process and specific API specifications in the typical service scenarios listed in Part 3.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



1. Microsegmentations and SFCs use the same service model. The API orchestration logic used for creating a microsegmentation is the same as that used for creating an SFC. In the scenario where an SFC API is used to orchestrate an SFC and a microsegmentation, you need to use a Neutron API or VPC API to create a VPC and the corresponding network object instances, such as a logical router, logical switch, and logical port.

More Information

- Huawei enterprise service website - Data Center Network Solution
  - https://e.huawei.com/en/solutions/business-needs/enterprise-network/data-center-network
- Section "Secondary Development" in the iMaster NCE-Fabric Product Documentation
  - https://support.huawei.com/enterprise/en/network-management-and-analysis-software/imaster-nce-fabric-pid-250938411/doc

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.


Thank You
www.huawei.com



Introduction to Network AI

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Foreword

- AI is enabling a wide range of industries. The combination of AI and networks is an inevitable trend.
- This course briefly introduces the current situation of the AI industry and Huawei's network AI engine iMaster NAIE.

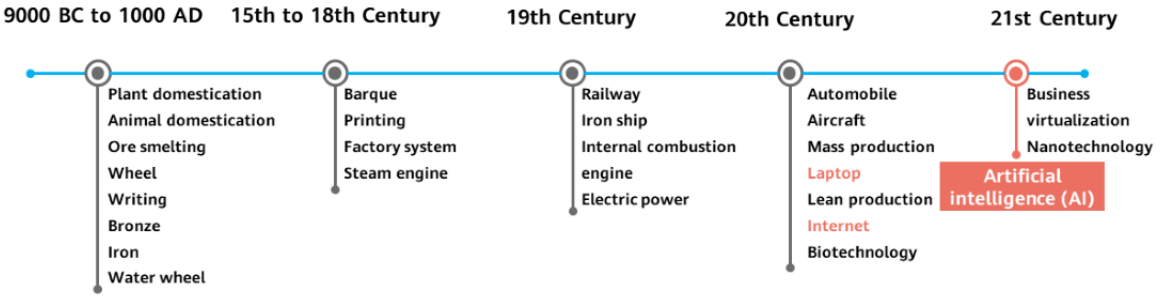
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Objectives

- Upon completion of this course, you will be able to:
  - Describe the requirements of network AI.
  - Describe the iMaster NAIE services of Huawei network AI engine.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

AI: New General Purpose Technology



AI is a multi-functional technology that can be applied almost everywhere to support techno-complementarity and generate technical overflow effect.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- General Purpose Technology (GPT) is the main driving force for economic and social transformation. From the agricultural society to the industrial society and then to the information society, the production mode, life mode, and management mode of the human society have undergone tremendous changes and experienced unprecedented economic and social transformation. For a long time, people have been thinking and exploring the drivers of economic and social development and transformation. From the first technological revolution represented by steam engine to the second technological revolution represented by electricity technology, looking at the industrial and technological revolution of the past 300 years, we can see that science and technology are important sources for promoting sustained economic growth. AI has become a new general purpose technology. Currently, popular AI technologies are being implemented, enabling a wide range of industries.
- https://en.wikipedia.org/wiki/General_purpose_technology
- Richard G. Lipsey, etc., Economic Transformations: General Purpose Technologies and Long-Term Economic Growth

AI Industry Status

	Amazon	Microsoft	Alibaba	Baidu	Huawei
Self-developed AI hardware	Inferentia inference chip	None	Hanguang 800 inference chip	Kunlun series training & inference chip	Ascend series training & inference chip
Self-developed learning framework	None	CNTK	None	PaddlePaddle	MindSpore
Machine learning platform	AWS SegaMaker	Azure ML	Alibaba PAI	Baidu BML	ModelArts
AI solution	Solution+AI	Solution+ AI	Data intelligence City brain, industrial brain, agricultural brain, and environment brain	Industry applications Smart industry, smart retail, enterprise service, smart government, and smart education	Intelligent twins Network intelligent twins, city intelligent twins, autonomous driving cloud services, and industrial intelligent twins

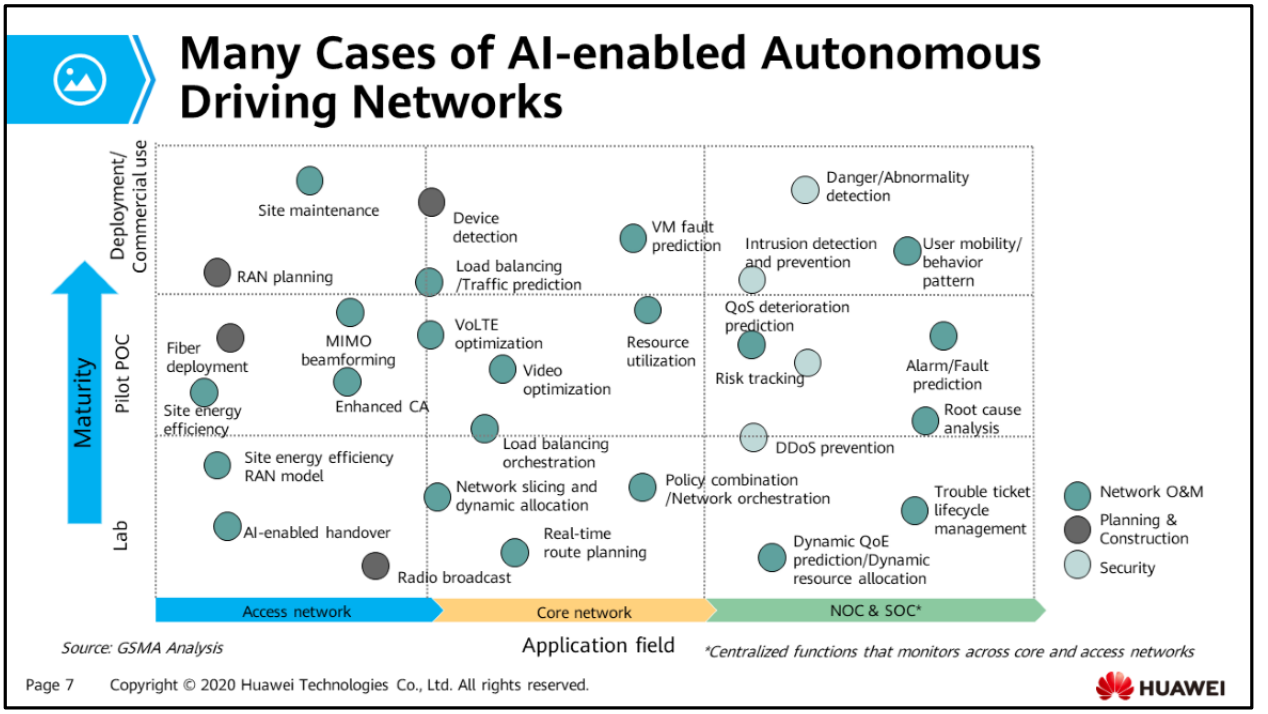
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

AI-based Network Becomes a Consensus in the Industry



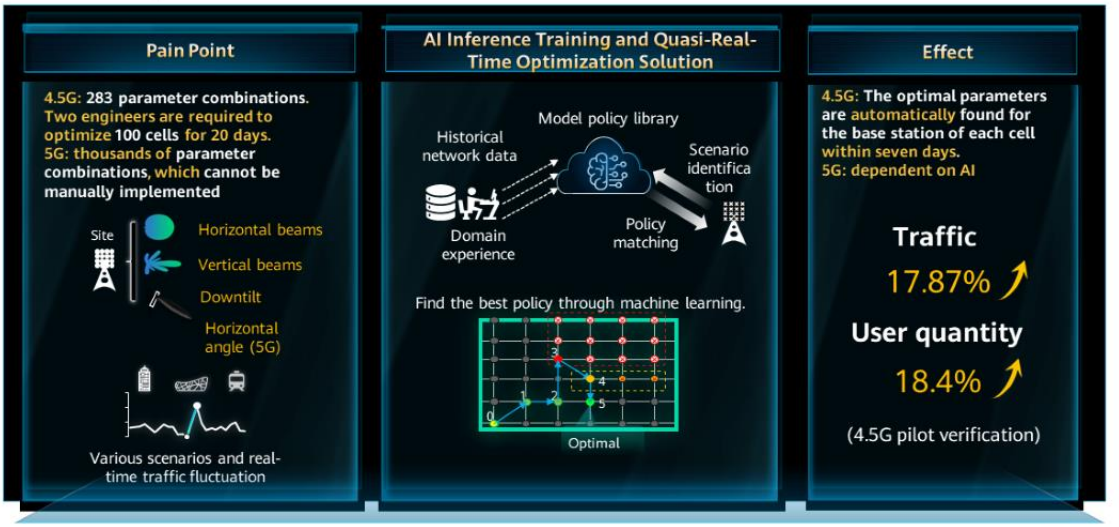
Definition/Level	L0 Manual operation	L1 Tool-assisted automation	L2 Partially autonomous networks	L3 Restricted autonomous networks	L4 Highly autonomous networks	L5 Fully autonomous networks
Repeated execution	Man	Man-Machine	Machine	Machine	Machine	Machine
Context awareness	Man	Man	Man-Machine	Machine	Machine	Machine
Analysis and decision-making	Man	Man	Man	Man-Machine	Man-Machine	Machine
Service experience	Man	Man	Man	Man	Man-Machine	Machine
Scope	N/A	Subtask level	Unit level	Domain level	Service level/Cross-domain	Public ICT infrastructure

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

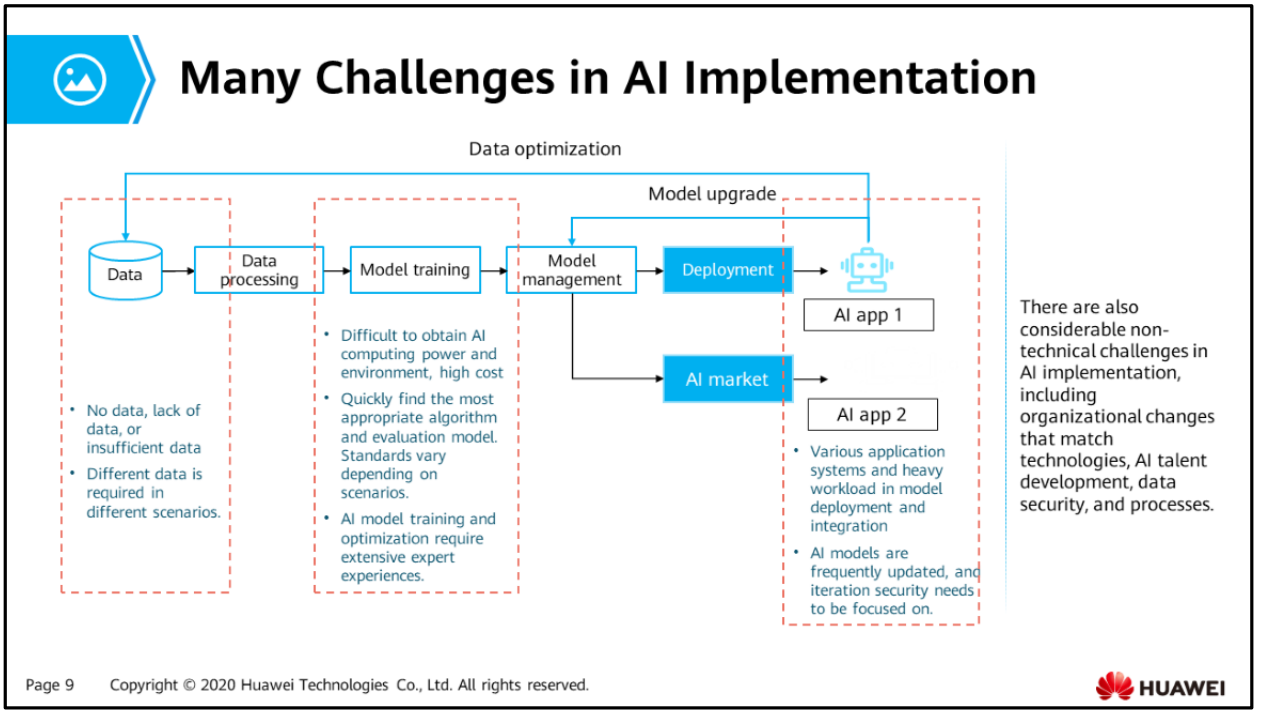


- In the telecom field, AI has many autonomous driving network cases. Telecom operators have AI cases from the access network, core network, to NOC/SOC.
- NOC: Network Operation Center
- SOC: Security Operation Center

AI Network Example: Massive MIMO Optimization



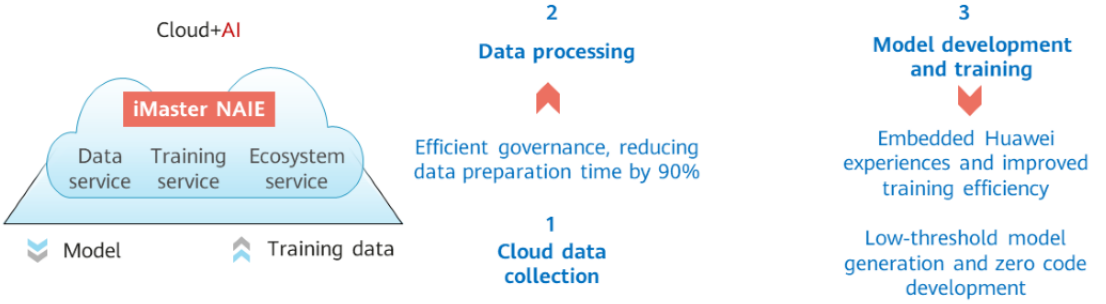
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



- Technically, AI case training requires joint development across domains (such as data, algorithm, and expert experience). Model optimization requires continuous iterative training, which has the following difficulties:
  1. The success of AI projects requires the cooperation of service experts and AI experts.
  2. It is difficult for service experts to transform into AI experts.
  3. There are many data issues, such as a few data sources and the requirement for data governance (labor-intensive).
  4. There are many algorithm engineering issues, such as conversion from paper to code and the efficiency of open-source algorithms.
  5. The computing power is difficult to obtain. The computing power is used during peak hours. (Nvida does not allow the use of G series GPUs in data centers.)
- Essentially, AI will bring about organizational transformation, from human resources to human-machine coexistence in the AI Ops phase.

Network AI Engine iMaster NAIE

- iMaster NAIE is a one-stop AI application development cloud platform that Huawei introduces AI technologies to telecom networks. Based on the public cloud mode, iMaster NAIE provides the data service, model training service, model generation service, and communication model service, covering the most complex parts of network AI app development, such as data preparation, data feature exploration, and model optimization. This helps developers quickly obtain NAIE capabilities.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

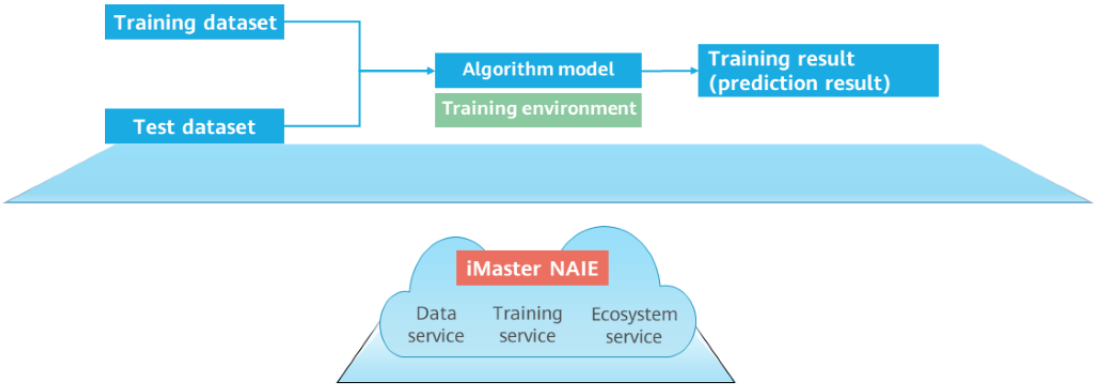
Example: Intelligent Traffic Sorting

- Requirement description:
- Data center networks carry various services, including big data, distributed storage, high-performance computing, and GPU cluster services. Traffic sorting is a process in which data center switches identify and classify traffic packets and determine the service to which each data flow or data packet belongs.
- It is required that an AI algorithm for traffic classification be implemented based on the embedded AI platform APIs (compatible with TensorFlow and Caffe) of Huawei's AI Fabric switches.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Training of the Intelligent Traffic Sorting Model

- The dataset, algorithm model, and training environment required for intelligent traffic sorting can be provided by the iMaster NAIE.



 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- The final objective of the experiment is to deploy the trained model in a real environment. Therefore, it is expected that the trained model can obtain a good prediction effect on real data. That is, it is expected that a smaller error between a prediction result of the model and the real result on real data is better. The best method is to divide real data into a training dataset and a test dataset. We can use the training dataset to train the model, and then use the error of the test dataset as the error of the final model in actual scenarios. With the test dataset, to verify the final effect of the model, we can calculate the error of the trained model only based on the test dataset. A smaller error indicates a better algorithm model.
- For detailed operations, see the following website: https://devstar.developer.huaweicloud.com/devstar/code-templates/e9078ee2d7024ffabbac3f8fd1bad806
- For more information about AI, refer to Huawei AI certification documents.

Quiz

1. (TorF) The goal of network development is to achieve full autonomous driving network of L5. ( )

   A. True

   B. False

2. What is the training process of the intelligent traffic sorting model?

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

1. A
2. The real traffic data is divided into a test dataset and a training dataset. The training dataset is used to train a model, the trained model is tested based on the test dataset, and then the model evaluation result is obtained.

Summary

- The combination of network and AI is the development trend of the network industry.
- Huawei iMaster NAIE provides network-based AI services, making network AI openness easier.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.


Thank You
www.huawei.com



NCE Service Openness and Programmability

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Foreword

- As digital network transformation moves forward, new services and technologies are burgeoning, and network requirements become more complex. Traditional closed networks cannot meet the requirements for rapid service innovation. It is an inevitable trend to improve network openness using existing assets. More agile planning of new services and network service orchestration based on customized customer intents are core capabilities.
- NCE service openness and programmability are based on the open architecture driven by the YANG model. Huawei enables openness and programmability at the NE and network layers in the form of Specific NE Driver (SND) and Specific Service Plugin (SSP) packages, implementing quick interconnection between new devices and fast rollout of new network services.
- This course introduces the background of service openness and programmability and Huawei's practices.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Objectives

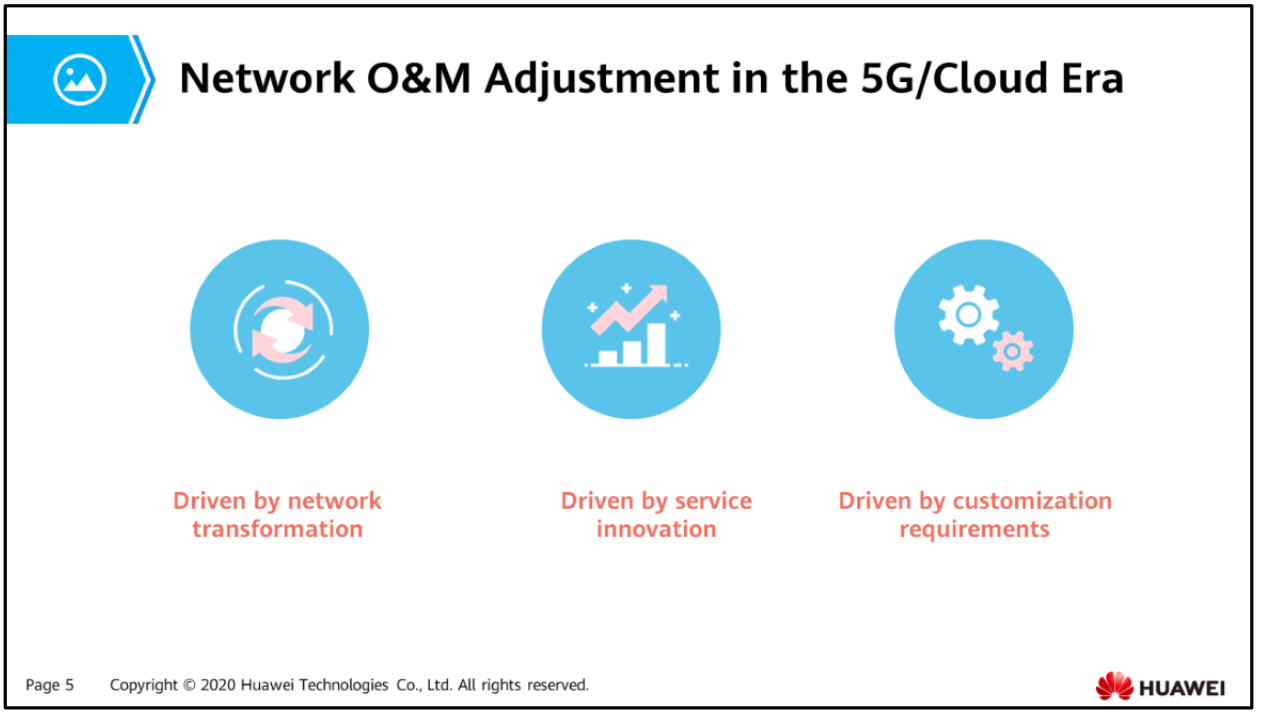
- On completion of this course, you will be able to:
  - Describe the basic concepts of Huawei NCE service openness and programmability.
  - Describe the open and programmable architecture and working principles of NCE services.
  - Describe the application scenarios of NCE service openness and programmability.
  - Use the iMaster NCE open programmability system (OPS) to interconnect new devices and roll out new services.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

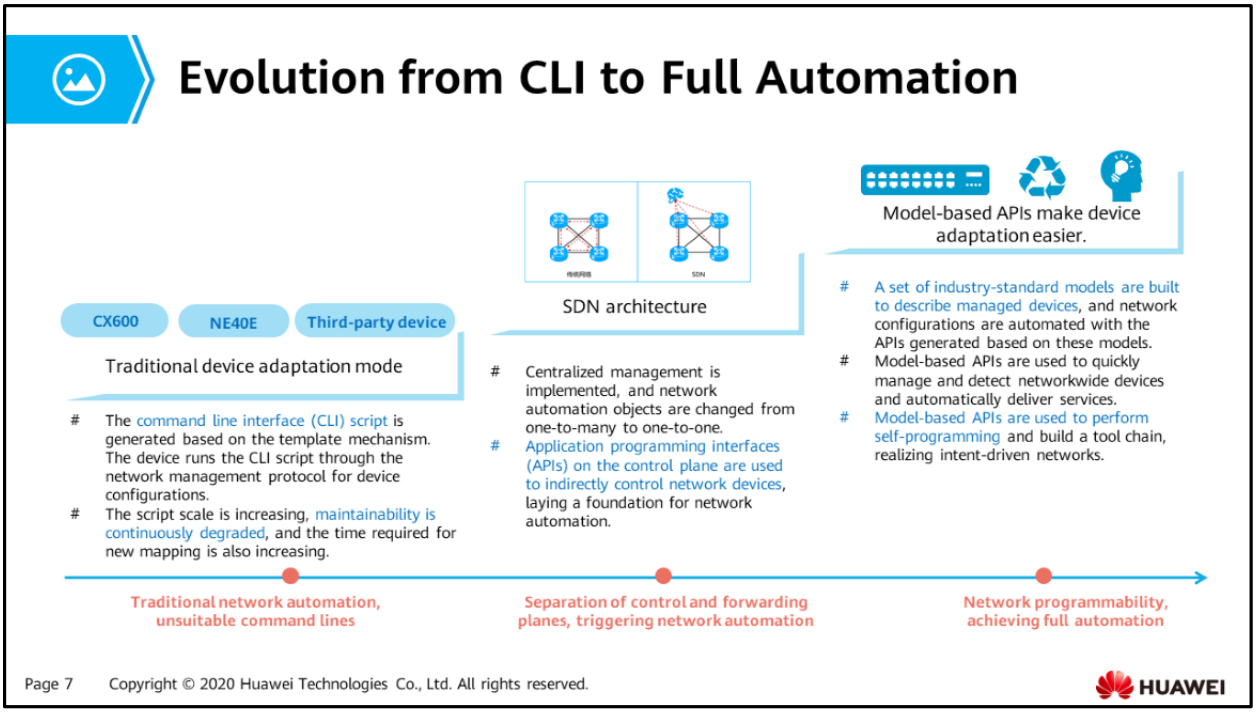
Contents

1. **Background**
2. Introduction to NCE Service Openness and Programmability
3. Key Capabilities
4. Related Concepts
5. Practice Cases

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.



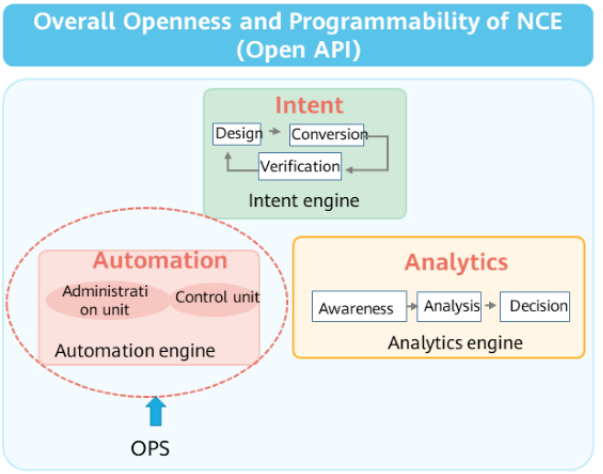
- First, let's take carriers as an example. Globally, most carriers face the problems of revenue decrease and OPEX increase. Moreover, as OTT providers continue to preempt market shares, more and more carriers take OTT providers as their competitors. These factors drive carriers to transform their networks. In this case, carriers are faced with the following problems: how to implement multi-network convergence, multi-vendor collaboration, and fast and efficient management of converged networks.
- In the 5G era, everyone predicts that 5G will lead to new businesses and services. However, carriers raise requirements for the rollout of new services, and device vendors implement the requirements. The rollout period is half a year or several years. It takes only a few months for OTT providers to launch new services, which makes it impossible for carriers and OTT providers to compete equally. There are many reasons for slow service rollout. One of the reasons is that there is a gap between carriers and vendors. That is, carriers do not understand devices, and vendors do not understand carrier services. It is an urgent issue to eliminate the impact of this gap and enable carriers and vendors to play their roles in the fields they are familiar with and quickly provision new services.
- Finally, the products provided by vendors are universal, that is, they are applicable to most operators. Carriers want systems to match their service requirements and enterprise cultures. Therefore, they have customization requirements. For example, a carrier writes the customization capability into its bidding document or customizes enterprise specifications. From the perspective of vendors, customization requirements of customers generate high costs. Therefore, the best solution is to provide the customization capability and let customers complete customization by themselves.



- On traditional networks, network automation refers to the process of generating command line scripts based on the template mechanism and enabling devices to run the received command line scripts through the network management protocol. It does not change the way it interacts with network devices. During device adaptation, network management engineers use Python or Perl to compile a specific function with a narrow application scope to implement a series of automatic operations, or use automation tools such as Ansible and Puppet to implement more complex automation tasks. Network management engineers need to adapt to network devices to be supported one by one, regardless of whether they write scripts or use automation tools. As the script scale becomes larger and larger, script maintainability decreases continuously, and the time required for adding a new version increases accordingly. With the advent of the Internet of Everything (IoE) era, the time to market (TTM) of new services has become a core indicator for enterprises to survive.
- With the great success of the commercialization of cloud computing, the concept of software-defined networking (SDN, sometimes referred to as “software-driven network”), which was popular only in the academic circle, has begun to flourish. On an SDN network, the separation between the control and forwarding planes is highly recommended. In an ideal SDN network, a centralized controller becomes an indispensable basis. As the brain of the entire network, it collects information about the network topology, calculates an optimal path globally based on service requirements, and notifies devices along the path. When receiving a service packet, these devices forward the packet according to a path determined by the centralized controller.

Huawei's NCE Service Openness and Programmability Practice

- iMaster NCE (NCE for short) is an innovative network cloudification engine of Huawei. The overall openness and programmability of NCE include automation, analytics, and intent. The goal is to build an open programmable architecture throughout the lifecycle to satisfy customers' requirements for openness and programmability.
- **Huawei open programmability system (OPS)**, a closed-loop component of the **automation engine**, are crucial to the entire open programming system of iMaster NCE.



Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

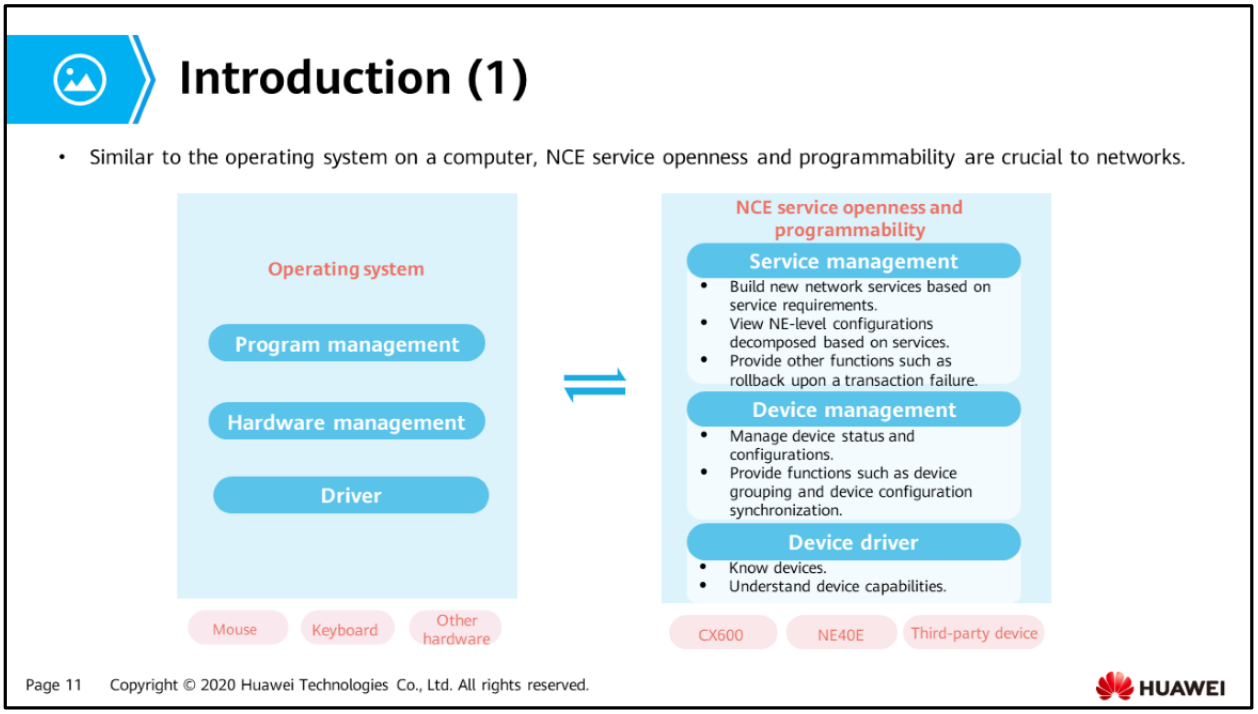
- iMaster NCE is an innovative network cloudification engine developed by Huawei. Positioned as the brain of future cloud-based networks, NCE integrates functions such as network management, service control, and network analysis. It is the core enablement system for network resource pooling, network connection automation, and O&M automation. NCE aims to build an intent-driven network (IDN) that is first automated, then self-adapting, and finally autonomous.
- The overall openness and programmability of NCE include automation, analytics, and intent. The goal is to build a full-lifecycle open and programmable architecture to satisfy customer needs. The OPS, as a part of the automation engine, are crucial for the entire open programming system of NCE to form a closed loop. Equivalent to the limbs of the human body, the OPS is an executor, which needs to be flexible to support the automatic closed-loop capability driven by the brain of an intent-driven network.

Contents

1. Background
2. **Introduction to NCE Service Openness and Programmability**
3. Key Capabilities
4. Related Concepts
5. Practice Cases

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

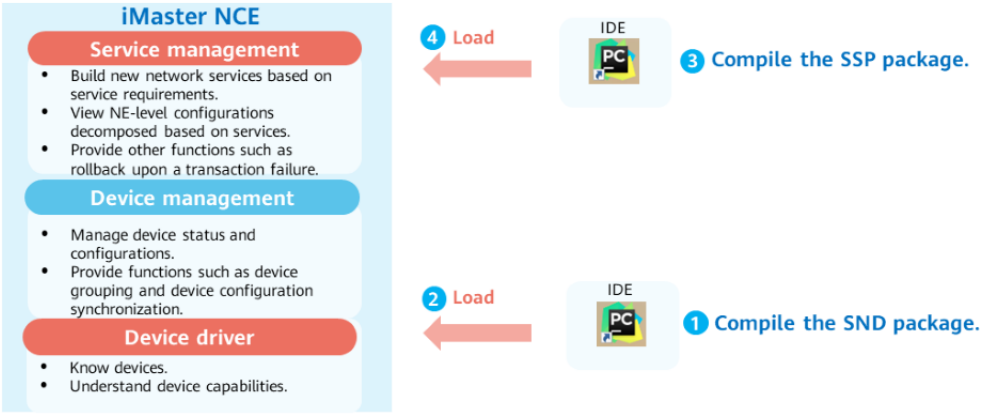
HUAWEI



- The open architectures of different industries are similar. Similar to the operating system on a computer, NCE service openness and programmability are crucial to networks.
- To connect the operating system to managed hardware, such as the mouse and keyboard, you need to install corresponding drivers. The drivers enable the operating system to recognize the hardware. NCE service openness and programmability have similar functions. The difference is that switches and routers are managed in the datacom industry. First, we need to understand and manage these switches and routers. That is, load device drivers first, and then add and understand the specific capabilities of the devices.
- At the upper layer, the operating system implements hardware management, specifically, managing the status of the mouse and keyboard. NCE service openness and programmability implement device management, such as managing device status and configurations.
- At the top layer, the operating system provides program management to manage various applications, such as Word and Excel. Note that the mouse and keyboard capabilities are required for using these programs. NCE service openness and programmability implement service management at the top layer, that is, building network service capabilities based on application scenarios. In addition, NCE provides capabilities such as rollback up on a transaction failure and automatic detection of device configuration changes to improve O&M.

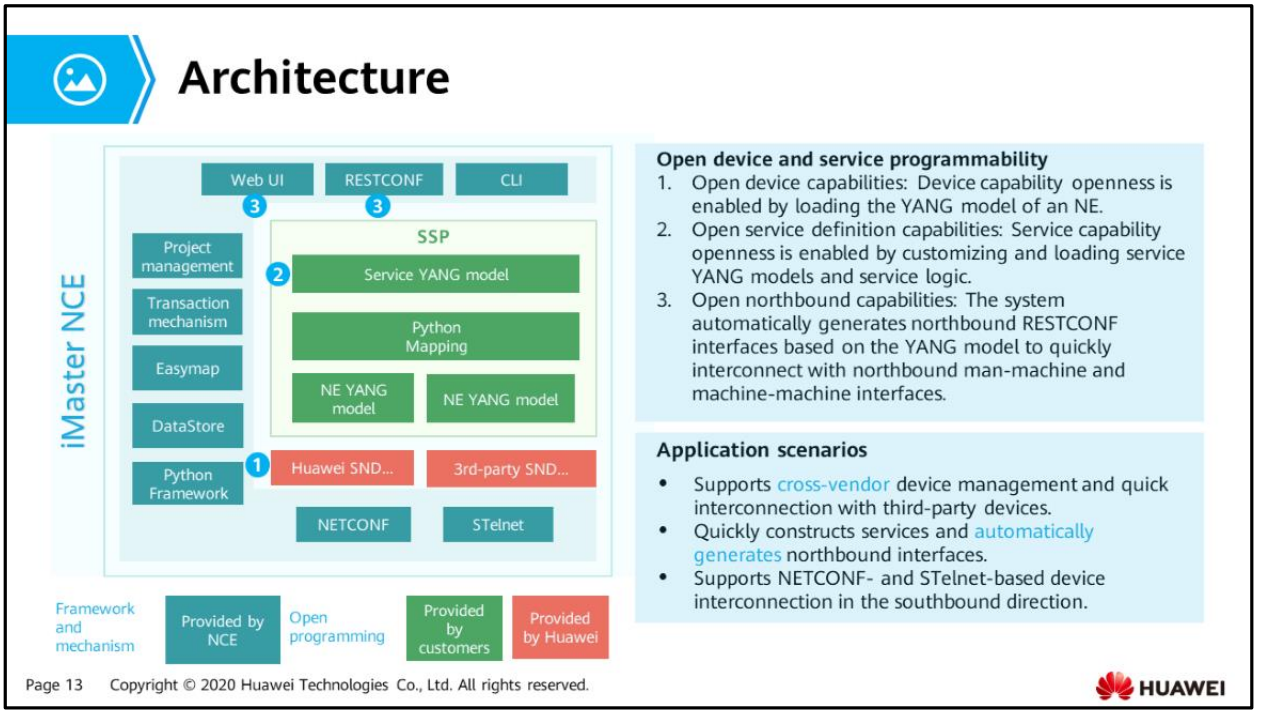
Introduction (2)

- NCE service openness and programmability are implemented based on two key software packages: Specific NE Driver (SND) and Specific Service Plugin (SSP). By compiling and loading two driver packages, you can quickly interconnect new devices and build new services.



 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

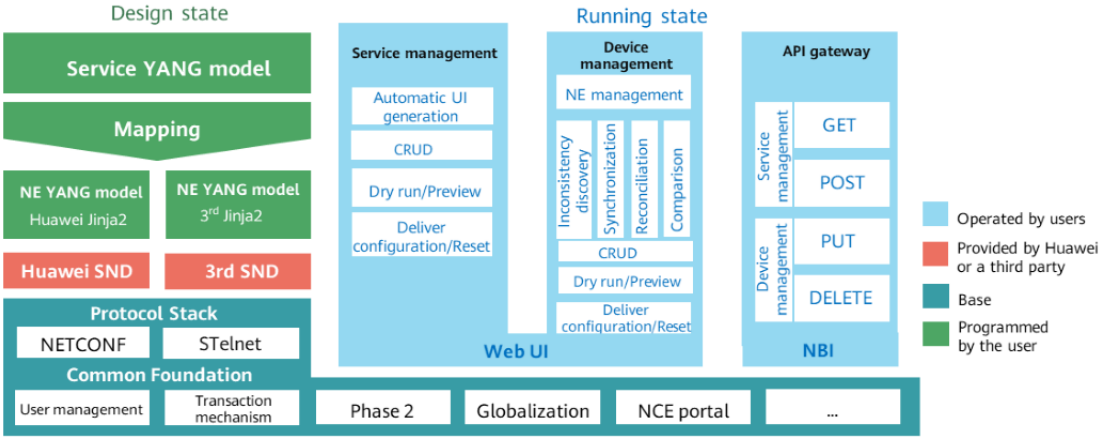
- NCE service openness and programmability depend on two software packages: SND and SSP.
  - Specific NE Driver (SND): provides a data model for the iMaster NCE OPS to interact with NEs.
  - Specific Service Plugin (SSP): defines a data model for completing network service configuration.
- Engineers compile SND packages and load them to iMaster NCE to quickly interconnect with new devices. Then, engineers compile SSP packages and load them to iMaster NCE to quickly construct new services.



- NE YANG model: YANG files generated by abstracting atomic capabilities (such as creating sub-interfaces) at the device layer. They are provided by device vendors.
- Service YANG model: YANG files generated by abstracting service models can be used to generate northbound interfaces and configuration GUIs.
- Easymap: a mapping logic algorithm that decomposes network-layer services into NE-layer services.

Principles

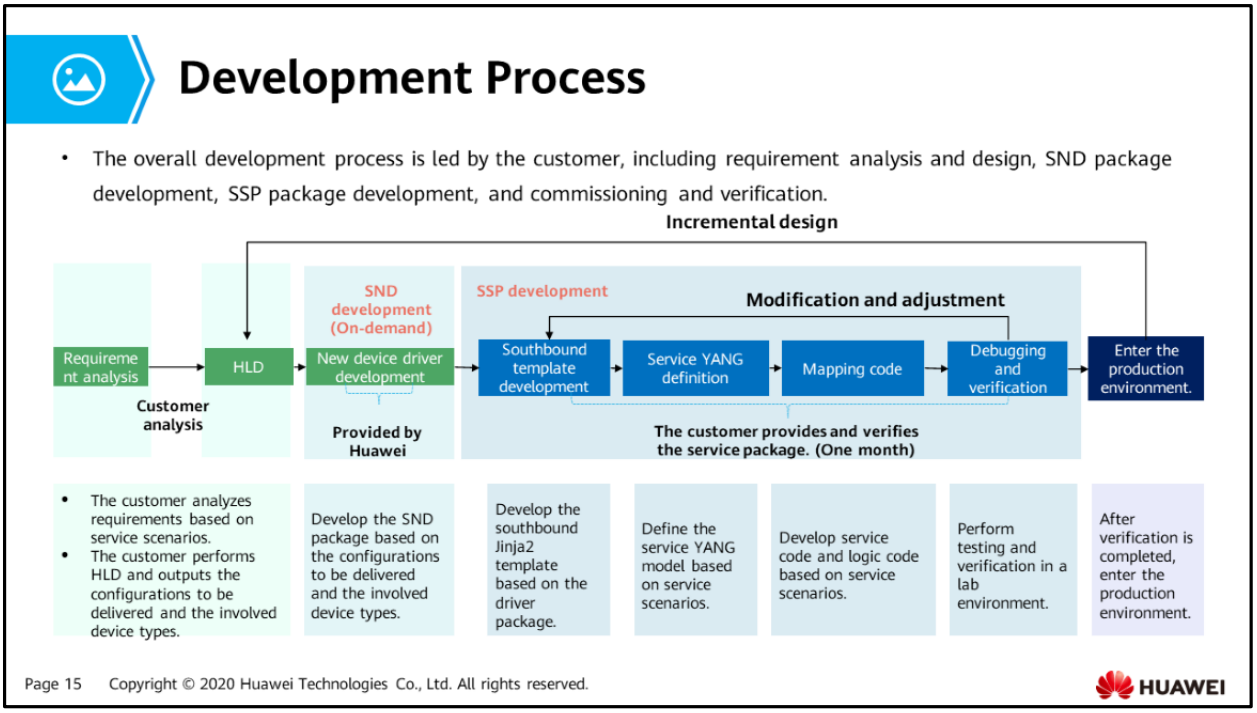
- NCE service openness and programmability consist of the design state and running state. The design state is used to design and develop service logic. The running state automatically generates service and device management GUIs and northbound interfaces based on the model defined in the design state. Users can configure devices and deliver services through the GUIs and northbound interfaces.



 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- The design state is used to establish the mapping between the service YANG model and NE YANG model. The system provides the mapping logic algorithm to decompose network-layer services into NE-layer services. Currently, the NCE service openness and programmability framework supports two layers of mapping logic: 1. Mapping from the service model to the device model, which is processed by the SSP package. 2. Mapping from the device model to protocol packets, which is processed by the SND package.
- The running state uses the mappings established in the design state to manage devices and provision services. Specifically:
  - Service management automatically generates a service management GUI based on the service YANG model to add, delete, modify, and query services.
  - Device management automatically generates an NE management GUI based on the NE YANG model to add, delete, modify, and query NE resources, achieving functions such as difference comparison, data synchronization, and configuration reconciliation.
  - API gateway automatically generates northbound RESTCONF interfaces based on the service and device YANG models and works with the mapping between the two models to add, delete, modify, and query services and NE resources.
- The running state provides the dryRun function to help users preview the results of the current operation and the modification of related device configurations.
- Jinja2 is a Python template engine. NCE service openness and programmability use

Jinja2 to quickly complete the template-based processing of SSP packages.



- The development process of NCE service openness and programmability is as follows:
  - First, analyze requirements based on service scenarios and output the high level design (HLD). In this phase, analyze the configuration commands to be delivered and the involved device types, and then start the development of a Specific NE Driver (SND) package. The SND package is developed as required. If the SND package of a device exists and the SND package to be delivered is supported, you do not need to develop the SND package again.
  - Then, develop a Specific Service Plugin (SSP) package. Step 1: Develop the southbound Jinja2 template. The southbound Jinja2 template can be considered as the tailoring of the open interfaces of the device. There are many open capabilities of devices. However, we only need to use some of them. Therefore, find and select the required ones. Step 2: Define the service YANG model and determine northbound input parameters. Step 3: Develop the service logic. This step is optional. If the service layer can directly map and use the southbound template, skip this step.
  - Finally, perform commissioning and verification. After the commissioning and verification are completed, the use in the production environment is formally started. If there are incremental requirements, follow the incremental design process and perform incremental development with reference to the preceding steps.

Contents

1. Background
2. Introduction to NCE Service Openness and Programmability
3. **Key Capabilities**
4. Related Concepts
5. Practice Cases

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

Overview of Key Capabilities

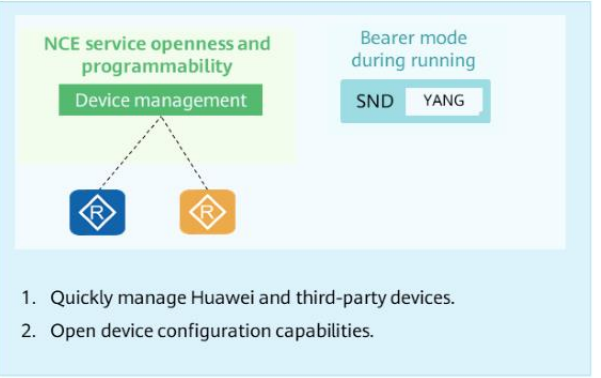
Key Capabilities

- Device capability openness
- Service capability openness
- Shielding of device differences for services
- Multi-protocol interconnection
- Batch configuration
- Transaction mechanism
- Service conflict detection
- Security audit

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Device Capability Openness

- Open atomic capabilities of devices. Driven by the device YANG model, the iMaster NCE OPS automatically generates northbound interfaces and configuration GUIs to quickly manage Huawei and third-party devices. It supports device data consistency comparison, configuration reconciliation, and configuration synchronization.

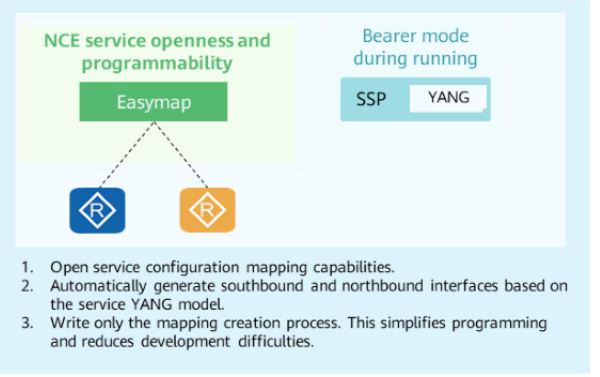


1. Quickly manage Huawei and third-party devices.
2. Open device configuration capabilities.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Service Capability Openness

- Open service configuration mapping capabilities. Driven by the service YANG model, the iMaster NCE OPS automatically generates northbound and southbound interfaces to quickly build new services. The Easymap algorithm in the mechanism is used. Users only need to write the creation process. The update and deletion processes are calculated by the algorithm, thereby simplifying programming and reducing development difficulties.

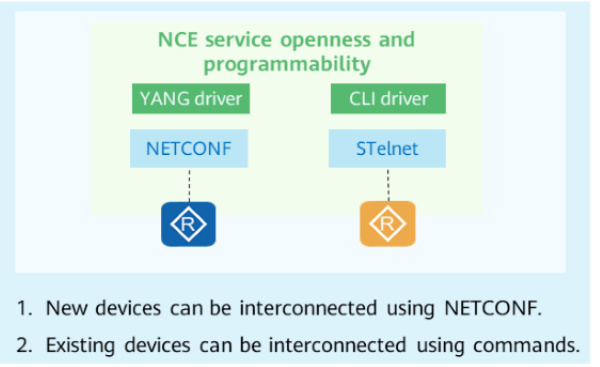


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Easymap: a mapping logic algorithm that decomposes network-layer services into NE-layer services. Currently, the NCE service openness and programmability framework supports two layers of mapping logic: 1. Mapping from the service model to the device model, which is processed by the SSP package. 2. Mapping from the device model to protocol packets, which is processed by the SND package.

Multi-Protocol Interconnection

- Support NETCONF- and STelnet-based device interconnection in the southbound direction. New devices can be interconnected using NETCONF, and existing devices can be interconnected using commands.



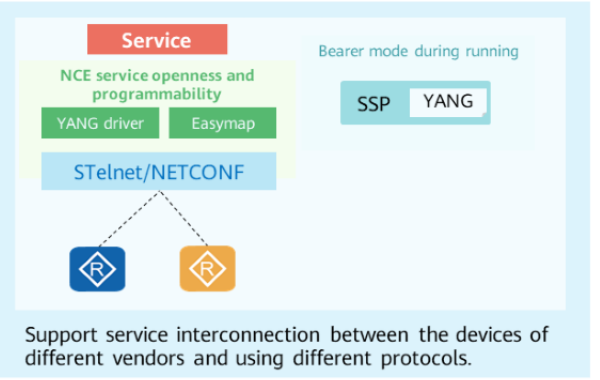
1. New devices can be interconnected using NETCONF.
2. Existing devices can be interconnected using commands.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- The SND package of the CLI driver is also a YANG file.

Shielding of Differences Between Underlying Devices at the Service Layer

- Only specific services need to be focused on at the service layer. Devices of different vendors and using different protocols do not need to be interconnected separately. The OPS supports service interconnection between the devices of different vendors and using different protocols.

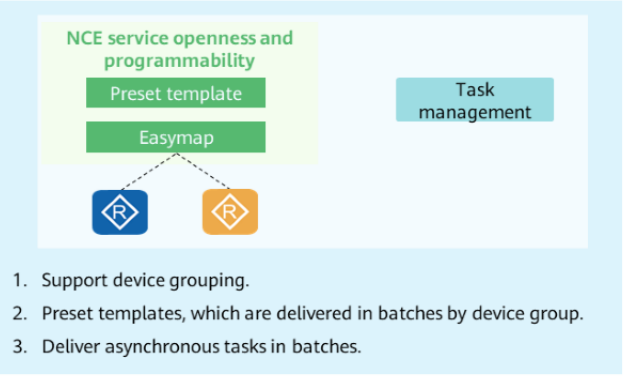


Support service interconnection between the devices of different vendors and using different protocols.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Batch Configuration

- Provide batch configuration capabilities. Devices with the same configuration can be grouped. A template is preset in the system first, and then configurations are delivered in batches.
- Templates are applied in batches by device group to improve efficiency.

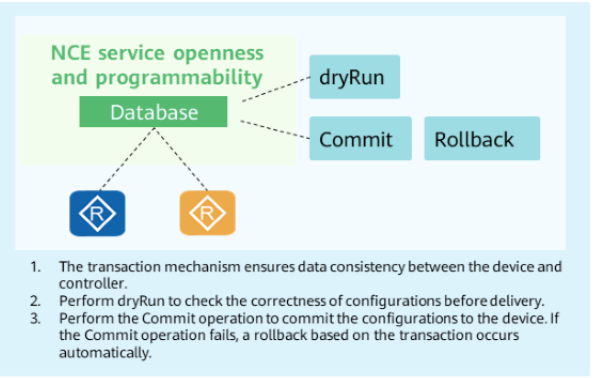


1. Support device grouping.
2. Preset templates, which are delivered in batches by device group.
3. Deliver asynchronous tasks in batches.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Transaction Mechanism

- Provide a transaction mechanism to ensure data consistency between the device and controller in both success and failure scenarios. The dryRun function enables the customer to check the correctness of configurations before delivery. The configurations can be repeatedly modified, and then the dryRun function is implemented until the configurations are correct. Then, the Commit operation can be started to commit the configurations to the device.

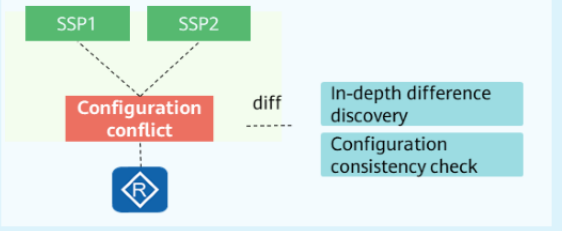


1. The transaction mechanism ensures data consistency between the device and controller.
2. Perform dryRun to check the correctness of configurations before delivery.
3. Perform the Commit operation to commit the configurations to the device. If the Commit operation fails, a rollback based on the transaction occurs automatically.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Service Conflict Detection

- Provide service conflict detection capabilities. The OPS supports source tracing of configured data sources. Configuration conflicts between different services can be detected before delivery. If the configurations of a device are modified by a third party, the device can automatically detect the modifications and re-deploy the configurations to restore the services interrupted by the modifications by the third party.
- Implement in-depth difference discovery. The controller directly compares the NE-level data decomposed by services with device data to check data consistency.

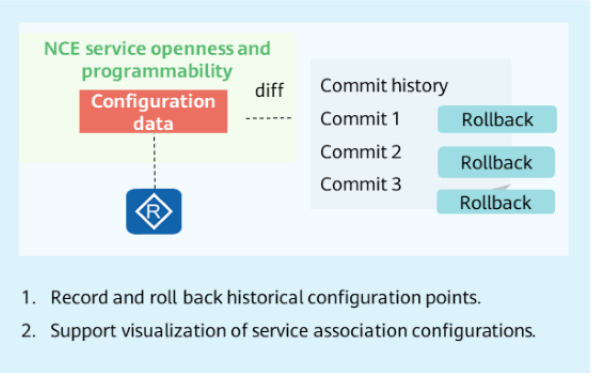


1. Perform data source tracing and conflict detection.
2. Support service and device data consistency check.
3. Automatically detect device configuration modifications by a third party and restore services.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Security Audit

- Provide security audit capabilities. Historical configuration points can be recorded and rolled back, and service association configurations can be visualized.



1. Record and roll back historical configuration points.
2. Support visualization of service association configurations.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Contents

1. Background
2. Introduction to NCE Service Openness and Programmability
3. Key Capabilities
4. **Related Concepts**
5. Practice Cases

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Function Package

Specific NE Driver (SND)

A type of software package that provides a data model for interaction between the OPS and NEs.

The data model usually contains a **.py** file and a data model (YANG) of several features. Specifically:

- The former is used to define information related to an NE, such as device type, vendor, and connection information.
- The latter describes the data structure of NE-related features.

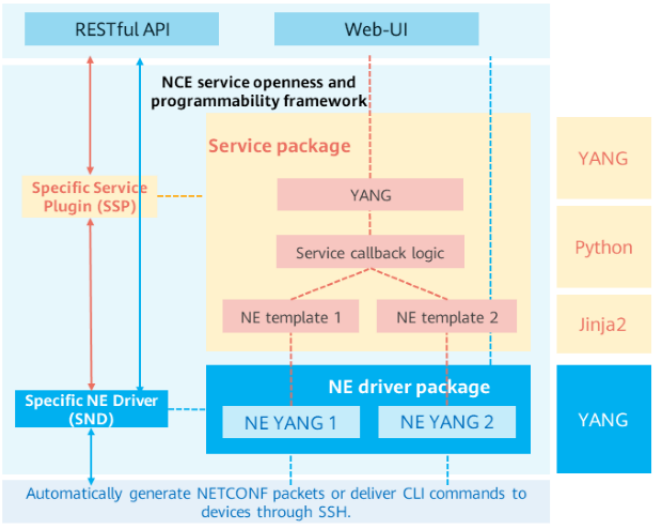
Specific Service Plugin (SSP)

A type of software package that defines a data model for completing network-level service configurations. The data model usually contains a Jinja2 template file, a Python mapping script, and a service YANG model. Specifically:

- The Jinja2 template describes the data structure of services and uses the Jinja2 syntax to perform operations such as interpolation, condition judgment, and recursion.
- The Python mapping script describes how to fill the data submitted by users into the template and map the data to the NE data structure.
- The service YANG model describes service parameters and is constructed based on service input.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Package Mapping Mechanism



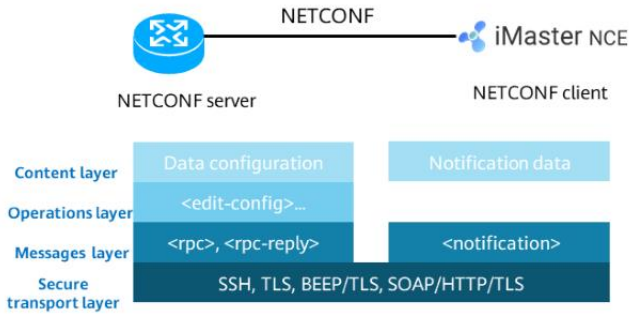
Packet mapping supports two layers of mapping logic: During the mapping from the service model to the device model, the logic is processed by the SSP package. During the mapping from the device model to protocol packets, the logic is processed by the SND package.

- NCE automatically generates northbound interfaces or configuration GUIs based on service models.
- Users send configuration requests to the service logic compiled by users through the interface provided by the service model.
- The service processing consists of two parts:
  - Python code processing, which processes service logic. For example, allocate resources and read NE or service information.
  - Jinja2 template processing, which processes vendor-related logic. A template is actually the data delivered to a device model. Devices from different vendors have different templates.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- Currently, the NCE service openness and programmability framework supports two layers of mapping logic: 1. Mapping from the service model to the device model, which is processed by the SSP package. 2. Mapping from the device model to protocol packets, which is processed by the SND package.
- For SND package processing, if the device is a NETCONF device, NCE service openness and programmability automatically convert the model data into NETCONF packets.

NETCONF

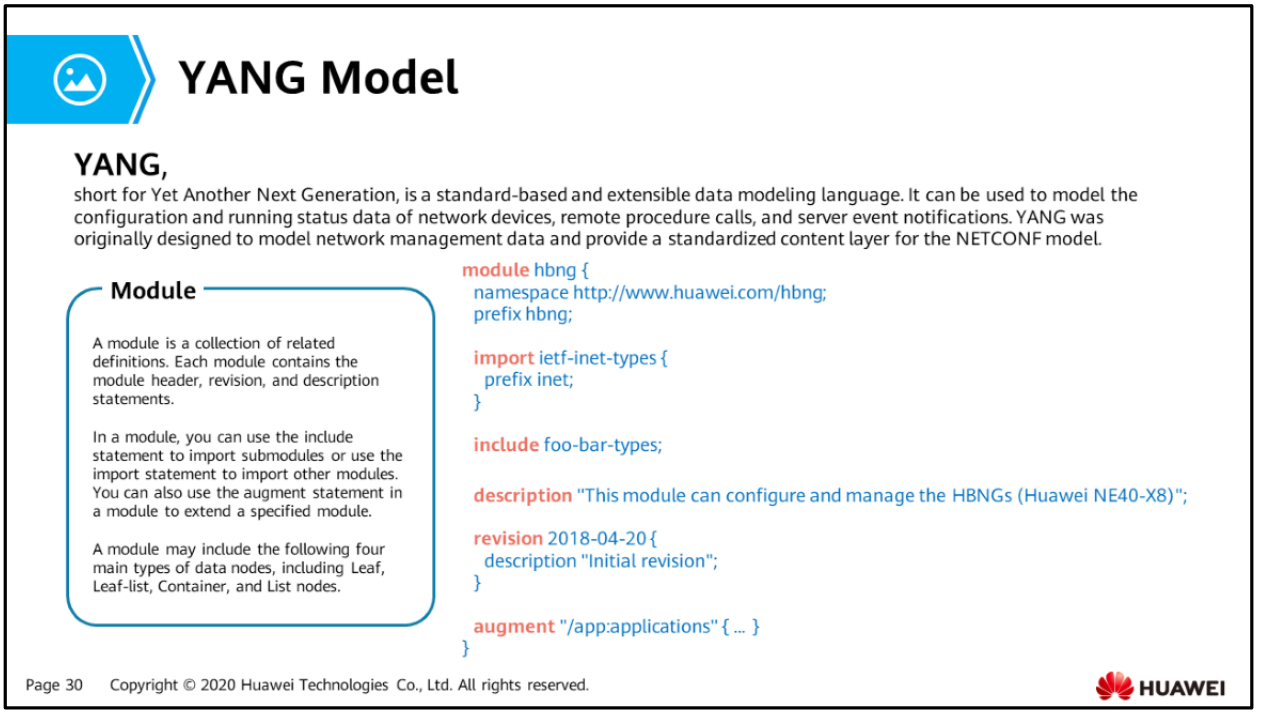


NETCONF is a protocol defined by the IETF for installing, maintaining, and deleting configuration data on NEs. All NETCONF operations are implemented using XML-encoded RPCs. NCE service openness and programmability use NETCONF to communicate with NEs.

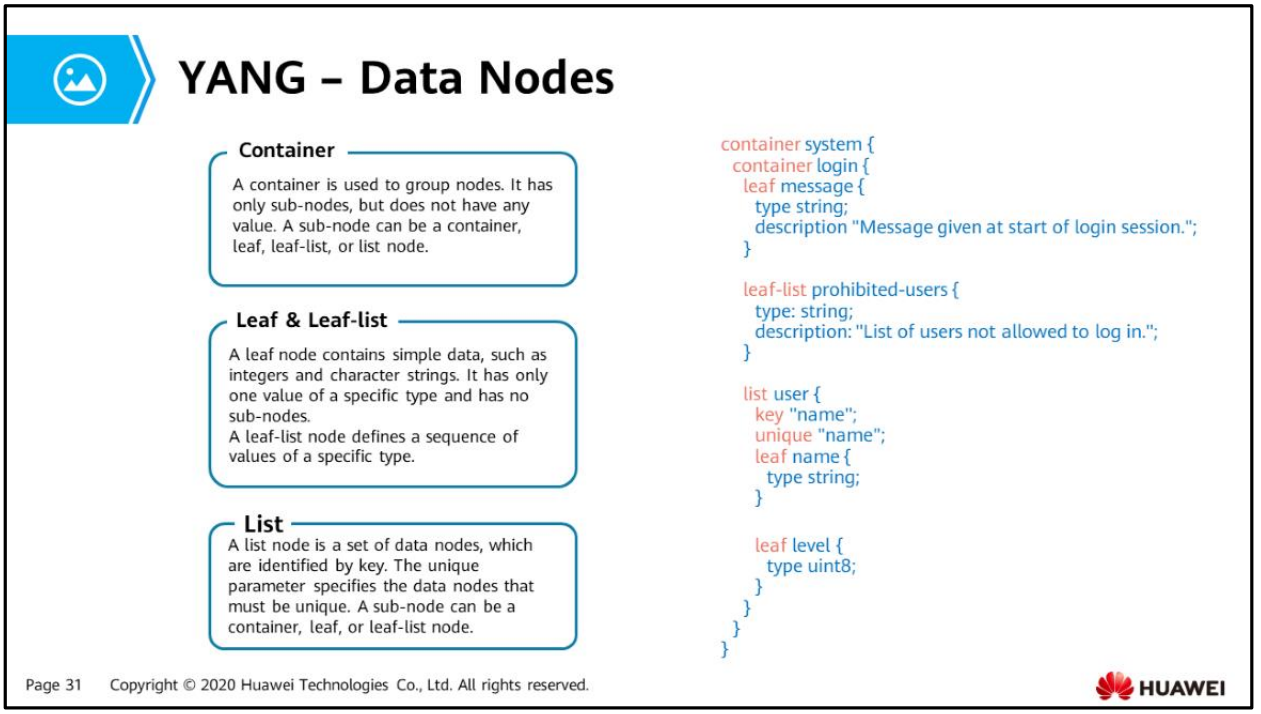
- Content layer: It needs to transmit configuration data and notification data.
- Operations layer: It defines an XML-encoded operation set for implementing RPC operations.
- Messages layer: It implements a simple framing mechanism based on the requirements of the transport protocol to encapsulate RPC messages or notification messages.
- Secure transport layer: It provides a communication path between the client and server. The NETCONF message hierarchy mechanism is implemented by using the communication paths that meet related conditions.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

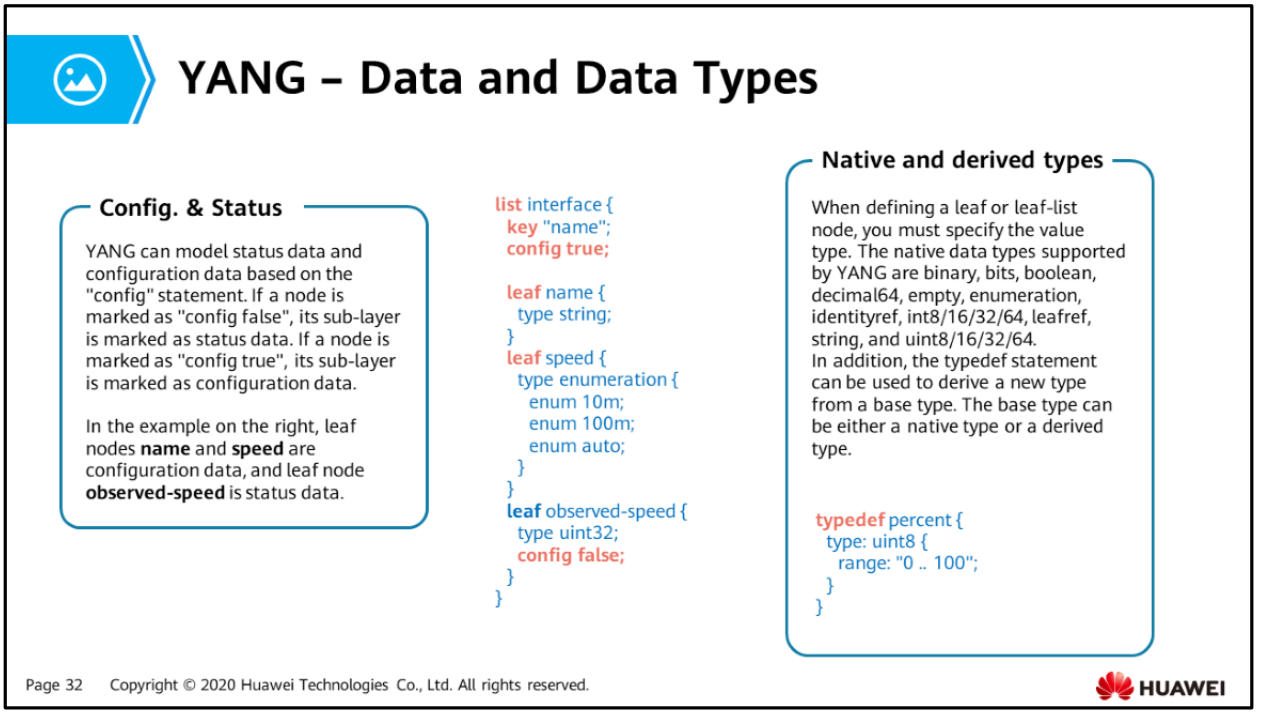
- For more information about NETCONF, see *NETCONF/YANG Principles and Practices*.



- In this example, the service YANG module **hbng** is customized.
- **description** describes the functions of the **hbng** module.
- **revision** is 2018-04-20, indicating the initial version of the **hbng** module.
- **import** and **include** introduce two modules for subsequent node definition.
- **augment "/app:applications" { ... }** indicates that the current module **hbng** is extended to the **/app:applications** directory of the **app** module.



- In this example, a container node named **system** is created, including the **login** container sub-node for recording login information.
- The container sub-node **login** contains the following:
  - A leaf node named **message**, which records the login prompt information.
  - A leaf-list node named **prohibited-users**, which records the blacklist of users who are not allowed to log in to the system.
  - A list node named **user**. In the list node, the unique key is defined as **name** and its type is character string; **level** is defined as user level and its type is number.



- In this example, the list interface is defined. **config true** indicates that the list is configuration data, and **config false** in **observed-speed** indicates that this leaf is status data.
- The leaf node **name** is a character string. The leaf node **speed** provides three options. **type enumeration** indicates that the enumerated values are **10m**, **100m**, and **auto**. The leaf node **observed-speed** is a positive integer of the **uint32** type.

YANG – Reuse & Selection

Reusable node group (grouping)

If a group of nodes can be reused in different modules or nodes, you can define this group of nodes as a reusable node group and use the **uses** keyword to reference modules or nodes.

Choices

YANG allows data models to use the choice and case statements to define a set of pattern nodes that cannot appear together.

```
grouping ip-port {
  leaf ip {
    type inet:ip-address;
    description "ip address";
  }

  leaf port {
    type inet:port-number;
    description "port number";
  }
}

container quadruple {
  container source {
    uses ip-port;
  }

  container destination {
    uses ip-port;
  }
}
```

```
container transfer-protocol {
  choice name {
    case a {
      leaf udp {
        type empty;
      }
    }

    case b {
      leaf tcp {
        type empty;
      }
    }
  }
}
```


Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- In this example, a group node named **ip-port** is defined, including two leaf sub-nodes: **ip** and **port**.
- The container **quadruple** contains the source and destination information containers, both of which use the IP address and port information. The group node **ip-port** is reused.
- The container **transfer-protocol** is used to indicate the transmission protocol. The UDP and TCP protocols are provided. Either of them can be selected using the **choice** function. **case a** indicates that the UDP protocol is used, and **case b** indicates that the TCP protocol is used.

YANG – Operations

RPCs & Actions

YANG allows the following operation definitions:

- Operations at the top layer of the module are defined using RPC statements. An RPC node usually consists of an input node and an output node. The input and output nodes can contain container, list, leaf, and leaf-list nodes.
- Operations in a container or list node are defined by action statements. Generally, an action node contains an input node and an output node.
- The difference between RPC and action is that RPC does not directly operate data, but action is directly bound to the node in the data storage.

```
module server-farm {
  yang-version: 1.1;
  namespace: "urn:example:server-farm";
  prefix: "sfarm";

  import ietf-yang-types {
    prefix: yang;
  }

  rpc reset-specified-servers {
    input {
      leaf-list servers {
        type: "inet:ip-address";
      }
    }
  }

  list servers {
    key: "name";
    leaf name {
      type: string;
    }
    leaf address {
      type: inet;
    }

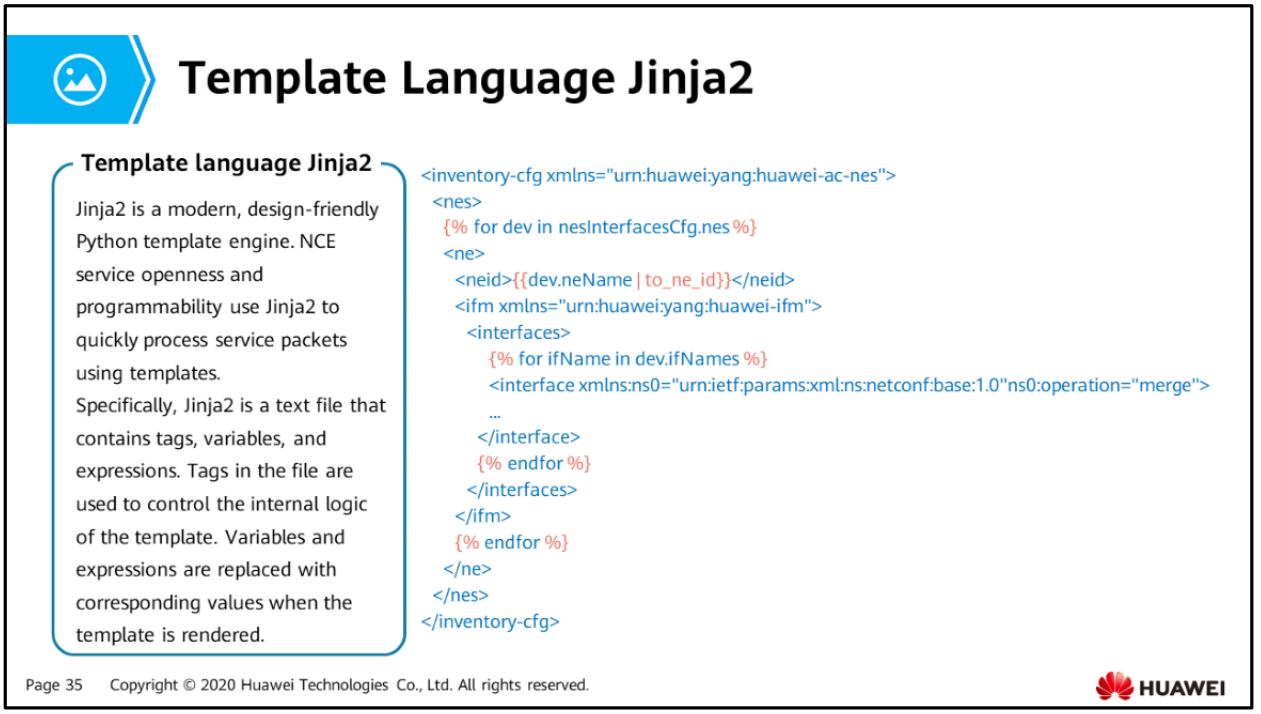
    leaf location {
      type: string;
    }

    action reset {
      input {
        leaf reset-at {
          type: yang:date-and-time;
        }
      }

      output {
        leaf complete-at {
          type: yang:date-and-time;
        }
      }
    }
```

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- In this example, an RPC interface named **reset-specified-servers** is defined for resetting services. **input** indicates that the input parameter is the IP address of the server to be restarted. If **output** is not defined, the HTTP status is used to determine the returned result.
- The **servers** list node defines **action reset** to restart the corresponding service. **Input** defines the leaf node **reset-at**, which indicates that the input parameter is the restart time. **Output** defines the leaf node **complete-at**, which indicates that the returned result is the restart completion time.



- The Jinja2 template is only a text file, which can be based on any text format (HTML, XML, CSV, etc.). In this example, the XML format is used.
- A template contains variables and expressions. The variables and expressions are converted to corresponding values when the template is used. It has the following common syntaxes:
  - {% ... **%}** contains Control Structures. In this example, **{% for dev in nesInterfacesCfg.nes %}** indicates that the for loop starts, and **{% endfor %}** indicates that the loop ends.
  - **{{...}}** contains an expression, which can be a constant, variable, mathematical formula, or logical statement.
  - {# ... **#}** indicates the comment.
- The variables in **{{...}}** can be modified using filters. Filters and variables are separated by vertical bars (|). For example, **{{ 'abc' | capitalize }}** indicates that the first letter is capitalized and the filtering result is **Abc**. In this example, **{{dev.neName | to_ne_id}}: to_ne_id** is a user-defined filter, indicating that the variable device name **dev.neName** is converted to the device ID.
- For more information, see the *Template Designer Document* at https://jinja.palletsprojects.com/en/2.11.x/.

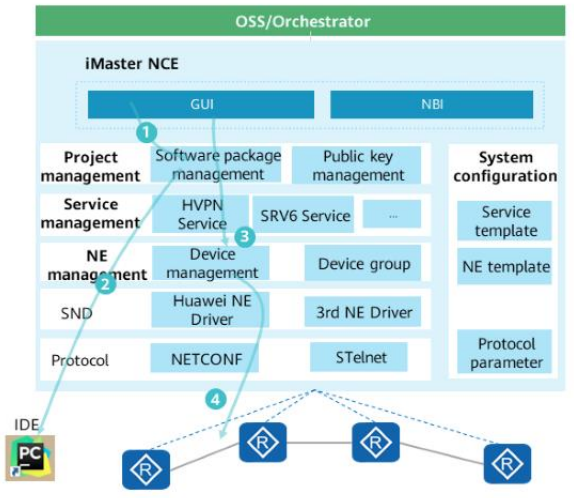
Contents

1. Background
2. Introduction to NCE Service Openness and Programmability
3. Key Capabilities
4. Related Concepts
5. **Practice Cases**

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

HUAWEI

Case 1: Fast Interconnection with New Devices and Automatic Generation of NBIs



Implementation process

1. **Generating the SND template**
   - In the OPS, choose **Project Management** > **Software Package Management** and generate an SND template.
2. **Developing the SND package**
   - Download the generated SND template and import it to the local IDE tool.
   - Develop the SND package based on the device type.
3. **Loading the SND package**
   - Load the developed SND package to NCE.
   - Add a new device. Device information is the same as that in the imported SND package.
   - Automatically generate configuration GUIs and NBIs for new devices.
4. **Delivering basic device configurations**
   - Apply basic device configurations through the GUIs or NBIs.

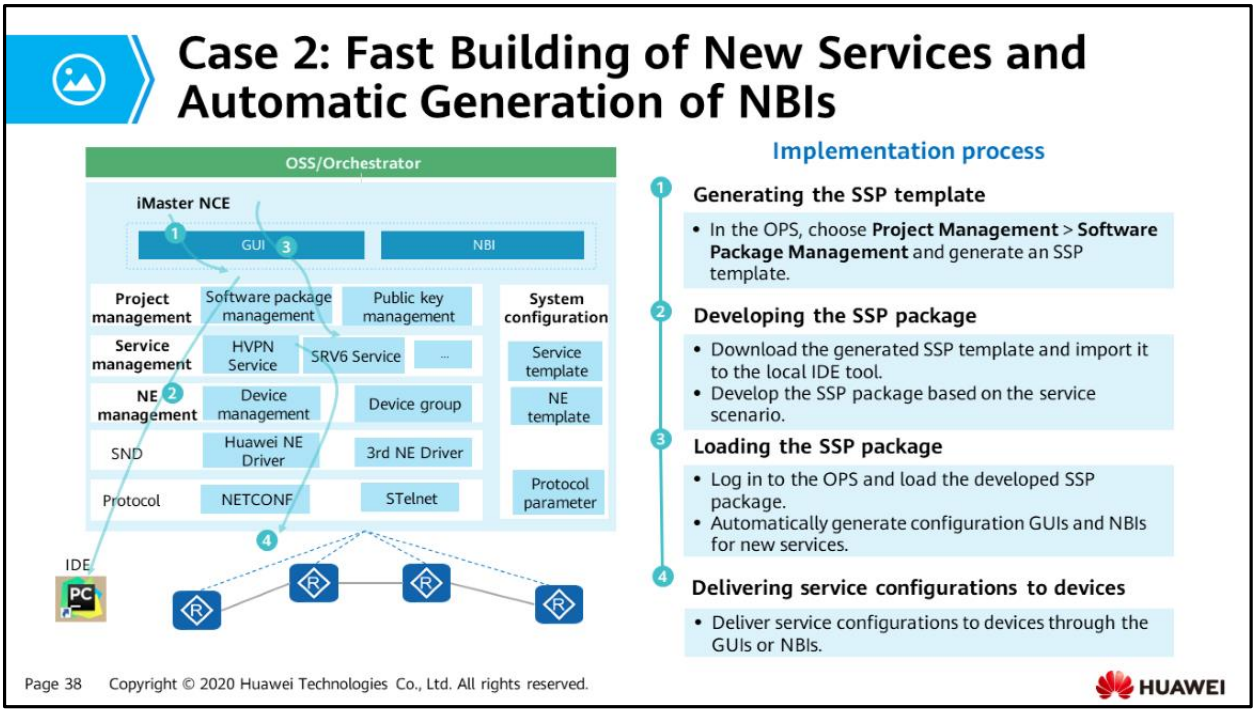
Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

- NCE uses the Specific NE Driver (SND) package to quickly interconnect and manage Huawei and third-party devices and open device configuration capabilities. To manage third-party devices, you need to obtain the YANG file of the device from the vendor's website. If third-party devices support only command lines and do not support NETCONF interconnection, Huawei can customize interconnection capabilities.
- Key capabilities:

  1. Quickly manage Huawei and third-party devices.

  2. Open device configuration capabilities.

  3. Automatically generate the configuration GUIs and northbound interfaces of new devices.



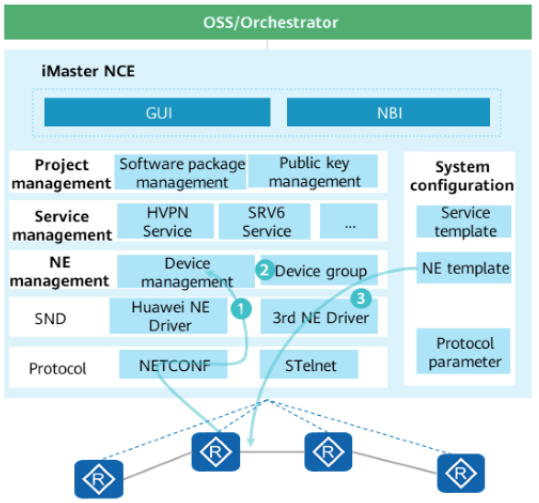
- In this example, the service openness capability is used. Similar to the device atomic capability openness, the system is developed based on the standard NETCONF protocol. The internal data model uses the YANG modeling language to automatically generate configuration GUIs and northbound interfaces based on the YANG model of services. In addition, the Easymap algorithm is provided for customers to write only the creation process, and the update and deletion are calculated by comparing algorithms. This simplifies customer programming.
- The service layer shields differences between devices, supports interconnection with different device types, and delivers configurations through different protocols. The maintenance personnel or upper-layer system only needs to view corresponding services. They do not need to know the specific vendor and protocol of the device. This feature improves interconnection efficiency and reduces the pressure on maintenance personnel.
- Key capabilities:

    1. Open service capabilities.

    2. Shield differences between underlying devices at the service layer.

    3. Automatically generate southbound and northbound interfaces based on the YANG model.

Case 3: Batch Configuration Delivery



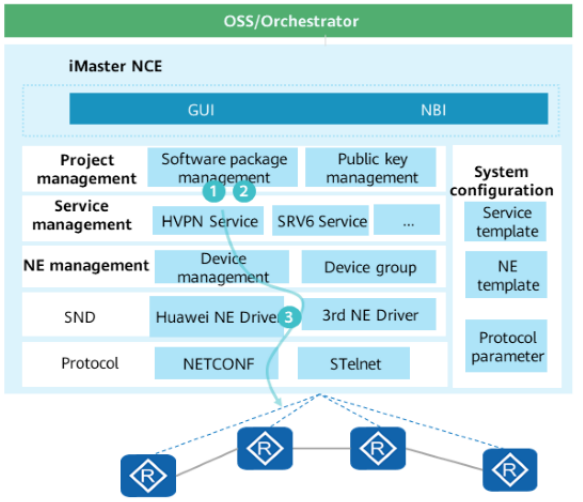
Implementation process

1. **Managing devices**
   - Choose NE Management > Device Management, add a device, and match the SND package based on the device NE type, software version, and vendor.
   - Synchronize device configurations to the OPS.
2. **Grouping devices on demand**
   - To apply one template to multiple devices, you can add these devices to the same device group.
3. **Delivering configurations in batches by device group**
   - Some templates can be preset for common configurations (such as underlay), and device groups can be selected for batch delivery.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

- Devices with the same configuration can be grouped. A preset template can be applied to the system for batch configuration delivery. Currently, more than 60 templates are preset in the enterprise DCN for users to apply.

Case 4: Configuration Correctness Pre-check Using the dryRun Function



Implementation process

1. **Configuring services**

   Configure services on the **Service Management** page as required.
   - Create a service.
   - Select an existing service and modify the configuration parameters.
   - Select an existing service and delete it.

2. **Checking before service configuration delivery**
   - Perform dryRun and check whether the generated configurations are correct.

3. **Delivering service configurations to devices**
   - Perform the Commit operation to deliver configurations to the device.
   - View the delivered configurations through the associated data.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.
HUAWEI

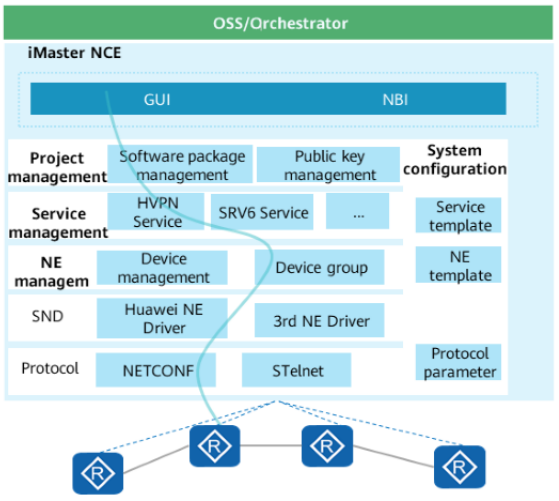
- Before service configurations are delivered, the OPS provides the dryRun function to check the correctness of delivered configurations in advance. If an error occurs, modify the dryRun function. After the configurations are correct, commit the configurations again. The system provides a transaction mechanism to ensure data consistency between the device and controller. If the data fails to be synchronized, the system automatically rolls back the data to ensure that no residual data exists. For a service that is successfully delivered, you can view the delivered configurations of the associated device. In addition, you can view the delivered configurations in historical records. You can roll back the configurations based on the rollback point.
- Key capabilities:

  1. Use the dryRun function to check whether the delivered configurations are correct in advance.

  2. Provide a transaction mechanism to ensure data consistency between the device and controller. If a failure occurs, automatic rollback will be performed.

  3. Provide the visualized display of service association data and historical configurations.

Case 5: Automatic Detection of Service Configuration Conflicts



Implementation process

1 Automatic detection of service configuration conflicts

Before delivering configurations, the OPS automatically checks whether the configurations delivered to devices conflict with each other. For example, if service 1 delivers the description configuration of VLAN 2 to device 1 and service 2 also delivers the description configuration of VLAN 2 to device 1, this system checks whether the two values are the same. If they are different, a message indicating that the configurations conflict will be displayed.

On the **Service Management** page, deliver configurations. The system automatically checks whether service configurations conflict.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Section Summary

- This chapter describes the practice cases of Huawei NCE service openness and programmability, including compiling NE driver packages to interconnect with new devices, compiling service packages to build new services, delivering configurations in batches, using the dryRun function to check configuration correctness, and automatically detecting service configuration conflicts.
- For details, see the HCIP-Datacom-NCE Service Openness and Programmability Lab Guide.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Quiz

1. What are the core function packages of Huawei iMaster NCE service openness and programmability?
2. (Multiple) Which of the following are included in a Specific Service Plugin (SSP) package? ( )

   A. Service YANG model

   B. Device YANG model

   C. Jinaji2 template

   D. Python mapping script

 Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

1. Specific NE Driver (SND) and Specific Service Plugin (SSP) packages.
2. ACD

Summary

- Network service openness and programmability pave the way for development of the network industry. Enabling the capabilities of network service openness and programmability is the cornerstone of building an open network ecosystem.
- Huawei OPS is a practice offered by Huawei in this field. Users can compile SND and SSP packages to quickly interconnect new devices and construct new services.
- To learn this chapter, you need to have a good understanding of Python, NETCONF, YANG, and Jinja2.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

More Information

- Log in to the HUAWEI CLOUD developer community and click the Datacom Network Openness and Programmability tab to obtain related tools and documents.

  https://developer.huaweicloud.com/resource/network.html#AOC

- Key documents:
  - Datacom Network Openness and Programmability Development Guide: This document describes the development process and methods of related driver packages, and provides guidance for developers to develop SND and SSP packages, load the packages to the system, and automatically generate configuration GUIs and NBIs.
  - Datacom Network Openness and Programmability User Guide: This document describes operations related to the configuration GUIs and NBIs, and provides guidance for O&M personnel to deliver device and service configurations.

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.

Recommendations

- For more information about Jinja2, visit the following website:

  https://jinja.palletsprojects.com/en/2.11.x/

Copyright © 2020 Huawei Technologies Co., Ltd. All rights reserved.


Thank You
www.huawei.com