

# 使用Telemetry实时监控CPU 和内存使用率

学院：信息工程学院

教师：张迁

# 目录

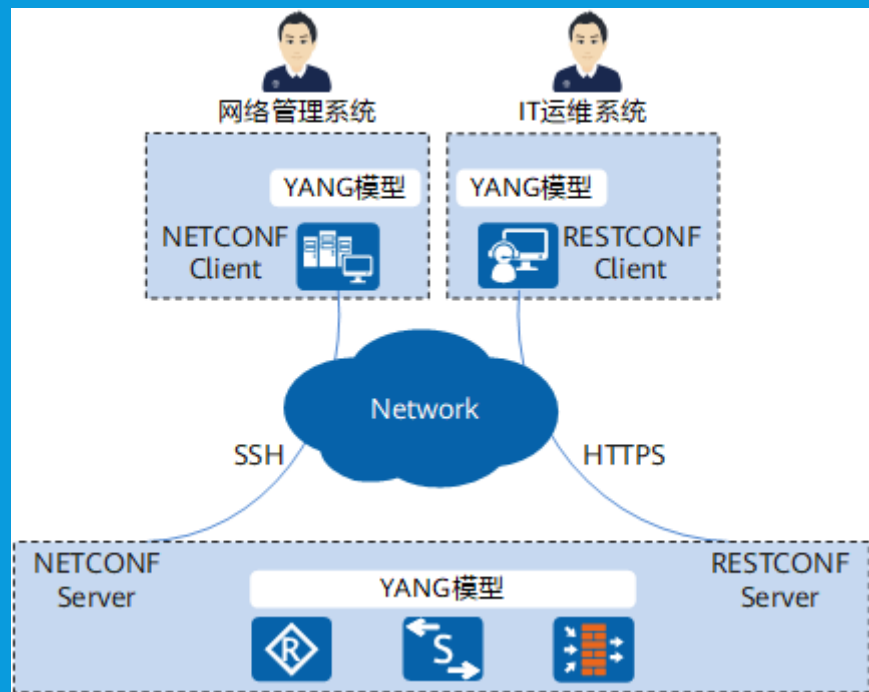
1. YANG建模语言
2. Telemetry技术
3. Proto文件
4. gRPC协议
5. 华为设备配置设备侧数据订阅指令
6. Grpcio-tools模块

# 1. YANG建模语言

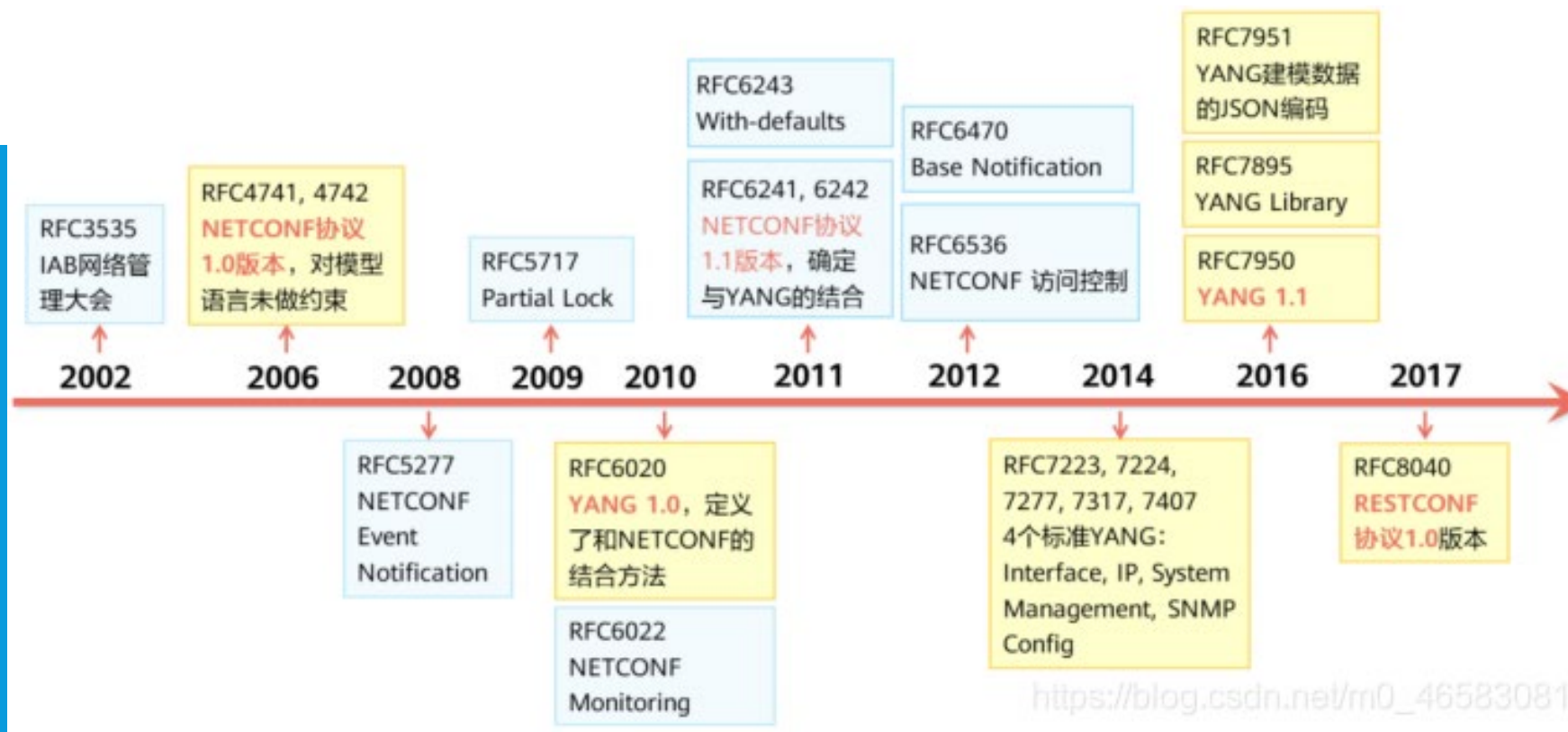
- YANG建模语言发展历程
- YANG模型主要元素
- YANG文件的树结构
- Pyang模块使用方法

# 1.1 YANG建模语言发展历程

- YANG是一种数据建模语言，YANG模型定义了数据的层次化结构，可用于基于网络配置管理协议（例如NETCONF/RESTCONF）的操作，包括配置、状态数据、远程过程调用和通知。
- 通过YANG描述数据结构、数据完整性约束、数据操作，形成了一个YANG模型（或者叫YANG文件）。



# 1.1 YANG建模语言发展历程



- 随着标准化的推行，YANG正逐渐成为业界主流的数据描述规范，标准组织、厂商、运营商、OTT纷纷定义各自的YANG模型。

# 1.1 YANG建模语言发展历程

■ YANG文件可以简单分为三类：

1、厂家私有YANG文件：以华为为例，其私有YANG模型一般以huawei开头；

2、IETF标准YANG：IETF定义的公有YANG模型，模型名称以ietf开头，少部分例外。

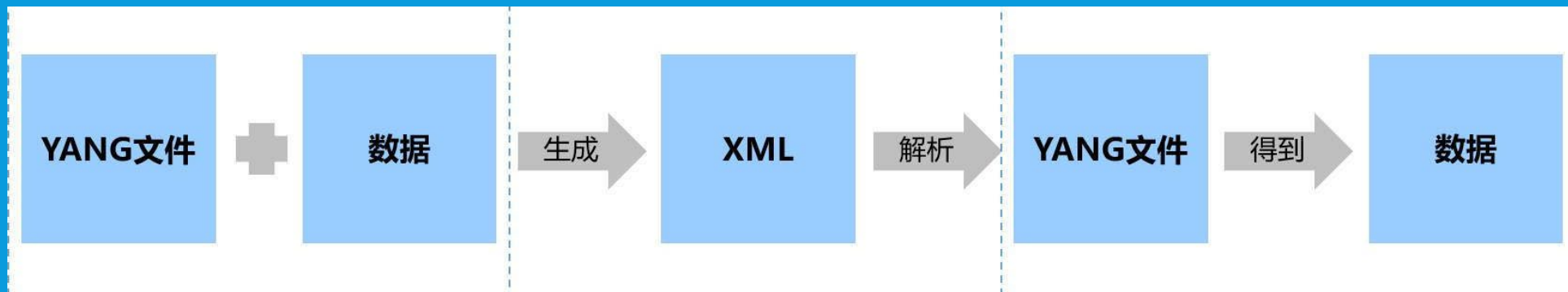
3、OpenConfig YANG：OPENCONFIG组织定义的公有YANG模型，也称为OC YANG。模型名称以openconfig开头

YANG模型的最终呈现是.yang为后缀的文件。

```
module huawei-ifm {  
  namespace "urn:huawei:yang:huawei-ifm";  
  prefix ifm;  
  import huawei-pub-type {  
    prefix pub-type;  
  }  
  organization  
    "Huawei Technologies Co., Ltd.";  
  contact  
    "Huawei Industrial Base  
    Bantian, Longgang  
    Shenzhen 518129  
    People's Republic of China  
    Website: https://www.huawei.com  
    Email: support@huawei.com";  
  description  
    "Common interface management, which includes the public configuration of interface"
```

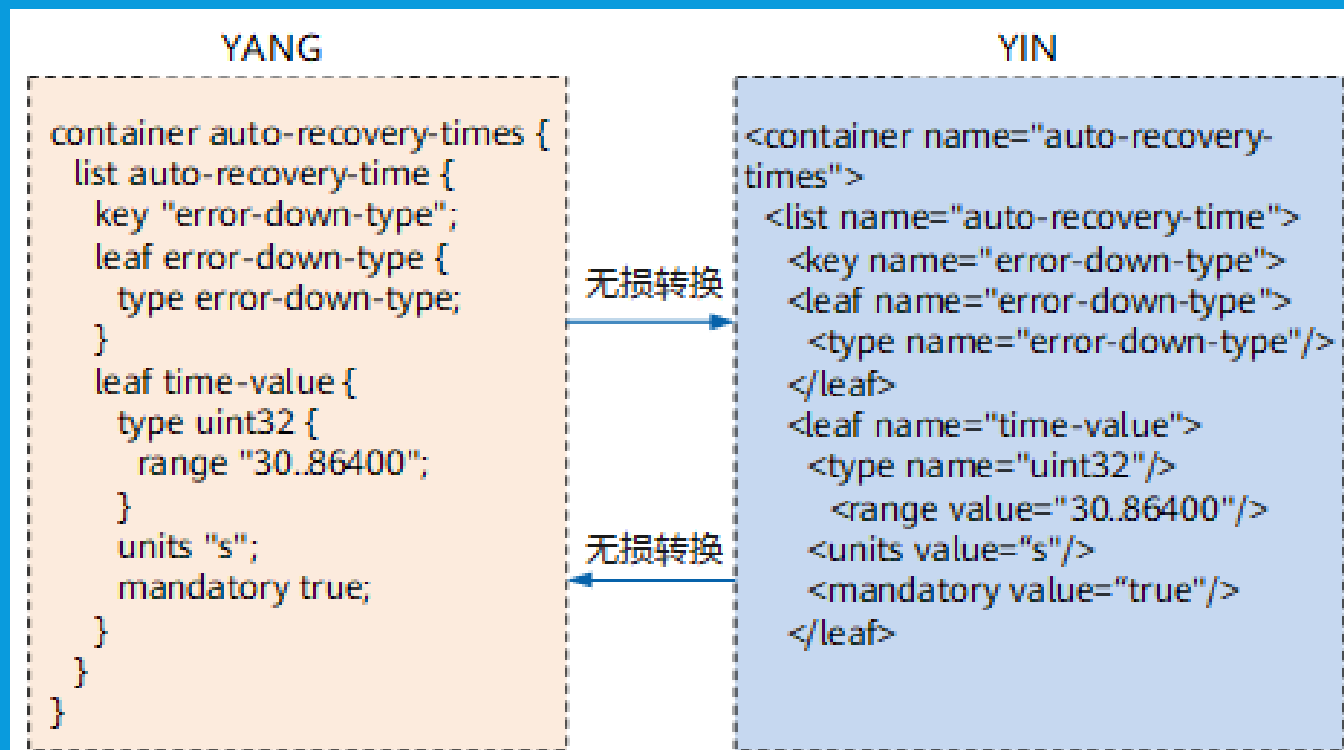
# 1.1 YANG建模语言发展历程

- YANG模型定义了数据的层次化结构，其有以下特点：
  - 基于层次化的树状结构建模。
  - 数据模型以模块和子模块呈现。
  - 可以和基于XML的语法的YIN（YANG Independent Notation）模型无损转换。
  - 定义内置的数据类型和允许可扩展类型。



# 1.1 YANG建模语言发展历程

- 设备解析模型时用YIN模型文件。
- 用YIN是为了利用各编程语言中现有的XML解析器等工具，这些工具可以用来进行数据过滤和验证，自动生成代码和文件或者其他任务，这样可以提升设备解析YANG模型的效率。





## 1.2 YANG模型主要元素

元素	描述
module	<p>YANG将数据模型构建为模块，模块名与YANG文件名一致。</p> <p>一个模块可以从其他模块中导入数据，也可以从子模块中引用数据。</p> <p>外部模块的引入（import &amp; include）：“include”声明允许模块或者子模块引用子模块中的材料，“import”声明允许引用其他模块中定义的材料。</p>
namespace	模块的名字空间，是全球唯一的URI。在对数据的XML编码过程中会使用到名字空间。
prefix	namespace的简写，简写唯一。
organization	YANG归属组织名。
contact	YANG模块的联系信息。
description	YANG模块的功能描述。
revision	YANG模块的版本信息，提供了模块的编辑版本历史。
container	容器节点，用来描述若干相关节点的集合。
list	列表节点，定义了列表条目序列，每个条目就像一个结构体或者一个记录实例，由其关键叶节点的值（key值）唯一识别。
leaf	叶节点。一个叶节点包含简单的数据，如整形数据或字符串。

- YANG模型是由一个模型和无数的叶子节点、节点列表、叶子列表、容器节点组成的描述整个设备的一颗树。

# 1.2 YANG 模型主要元素

取huawei-ifm.yang的部分内容做一个简单的YANG模型组成介绍：

模型开头部分是描述huawei-ifm.yang的基本信息。

Container，list，leaf则是YANG的定义的一些模型节点类型。

```
module huawei-ifm {
  namespace "urn:huawei:yang:huawei-ifm";
  prefix ifm;
  import huawei-pub-type {
    prefix pub-type;
  }
  organization
    "Huawei Technologies Co., Ltd.";
  contact
    "Huawei Industrial Base
    Bantian, Longgang
    Shenzhen 518129
    People's Republic of China
    Website: https://www.huawei.com
    Email: support@huawei.com";
  description
    "Common interface management, which includes the public configuration of interface";
  revision 2020-06-10 {
    description
      "Add units attribute.";
    reference
      "Huawei private.";
  }
  container auto-recovery-times {
    description
      "List of automatic recovery time configuration.";
    list auto-recovery-time {
      key "error-down-type";
      description
        "Configure automatic recovery time.";
      leaf error-down-type {
        type error-down-type;
        description
          "Cause of the error-down event.";
      }
      leaf time-value {
        type uint32 {
          range "30..86400";
        }
        units "s";
        mandatory true;
        description
          "Delay for the status transition from down to up.";
      }
    }
  }
}
```

## 1.2 YANG模型主要元素-模块

- 一个YANG文件通常可以定义为一个模块（module）或者子模块（submodule）。
- 一个模块包含5个类型的语句：
  - 1、head语句(yang-version/namespace/prefix)
  - 2、连接语句(import/include)
  - 3、元信息(organization/contract)
  - 4、revision语句
  - 5、定义语句(container/leaf/leaf-list/grouping等等)。

## 1.2 YANG模型主要元素-模块

- YANG语句写法如下:

Module Huawei-ifm{

# YANG文件所有内容都在这里

namespace "xxx" ;

Prefix xxx;

Import xxx;

organization xxx;

...

}

```
module huawei-ifm {  
  namespace "urn:huawei:yang:huawei-ifm";  
  prefix ifm;  
  import huawei-pub-type {  
    prefix pub-type;  
  }  
  organization  
    "Huawei Technologies Co., Ltd.";  
  contact  
    "Huawei Industrial Base  
    Bantian, Longgang  
    Shenzhen 518129  
    People's Republic of China  
    Website: https://www.huawei.com  
    Email: support@huawei.com";  
  description  
    "Common interface management, which includes the public configuration of interfaces.";  
  revision 2020-06-10 {  
    description  
      "Add units attribute.";  
    reference  
      "Huawei private.";  
  }  
  container auto-recovery-times {  
    description  
      "List of automatic recovery time configuration."
```

## 1.2 YANG模型主要元素-节点

YANG定义了数据建模的四种主要类型的数据节点。

- 叶节点(Leaf Nodes)
- 叶列表节点(Leaf-List Nodes)
- 容器节点(Container Nodes)
- 列表节点(List Nodes)

## 1.2 YANG模型主要元素-节点

**Leaf Node:** 用于定义一个简单指定类型的变量，使用关键字“leaf”申明。其中“type”表示取值的类型，而“description”为描述。其支持config、description、mandatory、reference和units等定义。

YANG Example:

```
leaf host-name {  
    type string;  
    mandatory true;  
    config true;  
    description "Host name"  
}
```

- A leaf has
  - one value
  - no children
  - one instance

NETCONF XML Encoding:

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

## 1.2 YANG模型主要元素-节点

**Leaf list:** Leaf List用于定义一个数组类型变量，使用“leaf-list”关键字申明。下面例子中“domain-search”内含有两个子声明：“type”和“description”。

YANG Example:

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

- A leaf-list has:
  - one value
  - no children
  - multiple instances

NETCONF XML Encoding:

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

## 1.2 YANG模型主要元素-节点

**List node:** List节点用于定义一个更高层次的数据节点。一个List节点使用” key “唯一标识，可以包含多个leaf节点

### The "list" Statement

YANG Example:

```
list user {  
  key name;  
  leaf name {  
    type string;  
  }  
  leaf uid {  
    type uint32;  
  }  
  leaf full-name {  
    type string;  
  }  
  leaf class {  
    type string;  
    default viewer;  
  }  
}
```

- A list is
  - uniquely identified by key(s)
  - holds related children
  - no value
  - multiple instances

NETCONF XML Encoding:

```
<user>  
  <name>glocks</name>  
  <full-name>Goldie</full-name>  
  <class>intruder</class>  
</user>  
<user>  
  <name>snowey</name>  
  <full-name>Snow</full-name>  
  <class>free-loader</class>  
</user>  
<user>  
  <name>rzull</name>  
  <full-name>Repun</full-name>  
</user>
```



## 1.2 YANG模型主要元素-节点

**Container:** 主要定义一个schema树的内部节点，它本身没有任何值和意义，只是作为一系列子节点的父亲存在，只有一个实例。

YANG Example:

```
container system {  
  container services {  
    container ssh {  
      presence "Enables SSH";  
      description "SSH service specific configuration";  
      // more leafs, containers and stuff here...  
    }  
  }  
}
```

NETCONF XML Encoding:

```
<system>  
  <services>  
    <ssh/>  
  </services>  
</system>
```

■ A container has

- no value
- holds related children
- one instance

May have specific meaning (presence)

Or may simply contain other nodes

## 1.2 YANG模型主要元素-Grouping

**Grouping:** 一个grouping定义一个可以重复使用的节点集合，使用时通过use语句，并可通过refine语句进行改进。下面例子中target定义了leaf address和port。Container peer中声明use target，表示复用此leaf模型。

### The "grouping" Statement

#### YANG Example

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

- Defines a reusable collection of nodes
- Use multiple times
  - A modules may use groupings imported from other modules
- Refinement
- Use as structure, record, or object

#### NETCONF XML Encoding:

```
<peer>  
  <destination>  
    <address>192.0.2.1</address>  
    <port>22</port>  
  </destination>  
</peer>
```

## 1.2 YANG模型主要元素-派生类型

**Typedef:** YANG模型允许开发者自定义数据类型，typedef声明从基本类型定义派生新类型。派生类型可以作为参数的类型声明。下面代码定义了percent类型及其取值范围，同时定义了叶子节点completed使用该类型。

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

## 1.3 YANG文件的树结构

1、**配置数据和状态数据**：通过添加一个接口容器来说明配置数据和状态数据。其中：

配置数据：可读可写的配置字段，可以是接口名称、IP地址、子网掩码和管理员启用/禁用接口的配置命令等。

状态数据：只读的操作数据字段，包含数据包计数器和接口的物理状态暨UP/DOWN等。

## 1.3 YANG文件的树结构

2、**添加配置数据**：通过向容器中添加一个列表，用于定义接口配置数据。

```
container interfaces{
  list interface{
    key "name";
    leaf name{
      type string;
      mandatory "true";
      description
        "Interface name. Examl value: GigabitEthernet 0/0/0";
    }
    leaf address{
      type string;
      mandatory "true";
      description
        "Interface IP address. Examl value: 10.10.10.1";
    }
    leaf subet-mask{
      type string;
      mandatory "true";
      description
        "Interface subnet mask. Examl value: 255.255.255.0";
    }
    leaf enabled{
      type boolean;
      default "false";
      description
        "Enable or disable the interface. Examl value: true";
    }
  }
}
```

## 1.3 YANG文件的树结构

3、**添加状态数据**：在interfaces容器的配置数据后面添加状态数据，设置config为false，表示属于列表的子节点是只读的，不允许修改。

```
list interface-state{
  config false;
  key "name";
  leaf name{
    type string;
    description
      "Interface name. Example value: GigabitEthernet 0/0/0";
  }
  leaf oper-status{
    type enumeration {
      enum up;
      enum down;
    }
    mandatory "true";
    description
      "Describes whether the interface is physically up or down";
  }
}
```

## 1.4 pyang 使用方法

PYANG 是个 github 上的开源项目。作者是 Martin Bjorklund。NETCONF、YANG的RFC都是这位作者的作品。网络运维中使用pyang完成操作如下：

- 1、验证YANG模块。
- 2、生成YANG tree，以便于YANG的快速可视化。
- 3、将用 XML 编码的实例文档的模式感知转换为 JSON。

安装pyang模块：由于其不是Python标准库，因此在使用前需要进行安装：

`pip install pyang`

```
(base) C:\Users\USSTz>conda activate ensp_py1
(ensp_py1) C:\Users\USSTz>pip install pyang
Collecting pyang
  WARNING: Retrying (Retry(total=4, connect=None, read=None, redirect=None, status=None)) after connection error: ConnectionError(<pip._vendor.urllib3.connection.HTTPSConnection object at 0x0000000000000000>: /packages/29/f6343c83/pyang-2.6.1-py2.py3-none-any.whl.metadata)
  Downloading pyang-2.6.1-py2.py3-none-any.whl.metadata (821 byte)
  Requirement already satisfied: lxml in c:\users\usstz\anaconda3\envs\ensp_py1\lib\site-packages (2.4.2)
  Downloading pyang-2.6.1-py2.py3-none-any.whl (594 kB)
    594.7/594.7 kB 32.4 k
Installing collected packages: pyang
Successfully installed pyang-2.6.1
(ensp_py1) C:\Users\USSTz>
```

# 1.4 pyang 使用方法

安装完成后，可以通过以下命令查看pyang的使用说明：

**pyang -help**

普通用户模式转换主要关注如下几个即可：

-f 输出格式，这里支持的格式包括tree、yang、yin等，用户可以根据需求灵活选择；

-o 输出文件名；

-p 输出路径；

```
(ensp_py1) C:\Users\USSTz>pyang --help
Usage: pyang [options] [<filename>...]

Validates the YANG module in <filename> (or stdin), and all its dependencies.

Options:
  -h, --help                Show this help message and exit
  -v, --version              Show version number and exit
  -V, --verbose              Print a listing of all error and warning codes and
                             exit.
  -e, --list-errors          On errors, print the error code instead of the error
                             message.
  --print-error-code         On errors, print the basename of files of the error
                             message.
  --print-error-basename    Template used to display error messages. This is a
                             python new-style format string used to format the
                             message information with keys file, line, code, type
                             and msg. Example: --msg-template='{file} || {line} ||
                             {code} || {type} || {level} || {msg}'
  --msg-template=MSG_TEMPLATE
  -W WARNING                 If WARNING is 'error', treat all warnings as errors,
```



## 1.4 pyang 使用方法

一个yang文件至少有一个模块，而且模块名称与文件名相同，以config-interfaces.yang为例，我们使用pyang验证其Tree视图：

```
pyang -f tree config-interfaces.yang
```

```
(ensp_py1) C:\Users\USSTz\Desktop\网络运维文件包\拓扑图\63711-网络自动化运维教程-代码\项目1>pyang -f tree config-interfaces.yang
```

```
module: config-interfaces
```

```
  +--rw interfaces
```

```
    +--rw interface* [name]
```

```
      | +--rw name      string
```

```
      | +--rw address   string
```

```
      | +--rw subet-mask string
```

```
      | +--rw enabled?  boolean
```

```
  +--ro interface-state* [name]
```

```
    +--ro name      string
```

```
    +--ro oper-status enumeration
```

进入文件所在路径

## 1.4 pyang 使用方法

使用pyang将YANG文件转换为yin文件:

```
pyang -f yin -o config-interfaces.yin config-interfaces.yang
```

```
(ensp_py1) C:\Users\USSTz\Desktop\网络运维文件包\拓扑图\63711-网络自动化运维教程-代码\项目1>pyang -f yin -o config-interfaces.yin config-interfaces.yang
```

```
(ensp_py1) C:\Users\USSTz\Desktop\网络运维文件包\拓扑图\63711-网络自动化运维教程-代码\项目1>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<module name="config-interfaces"
  xmlns="urn:ietf:params:xml:ns:yang:yin:1"
  xmlns:ifm="urn:jsou:yang:jsou-ifm">
  <namespace uri="urn:jsou:yang:jsou-ifm"/>
  <prefix value="ifm"/>
  <organization>
    <text>jsou Technologies Co., Ltd.</text>
  </organization>
  <contact>
    <text>jsou Industrial Base
    jianye
    Nanjing 210036
    People's Republic of China
    Website: https://www.jsou.com
    Email: zhangqian@jsou.edu.cn</text>
```

# 实训8-（一）：使用pyang模块检查YANG文件树结构

【任务目标】掌握pyang模块检查YANG文件树结构的方法

（一）XML 语法规则及操作（请按要求填写命令，粘贴结果图）↵

1、XML 文档必须有一个根元素吗？↵

A. · 不需要↵

B. · 必须有↵

正确答案：B↵

2、XML 文档中的元素标签是否区分大小写？↵

## 2. Telemetry 技术

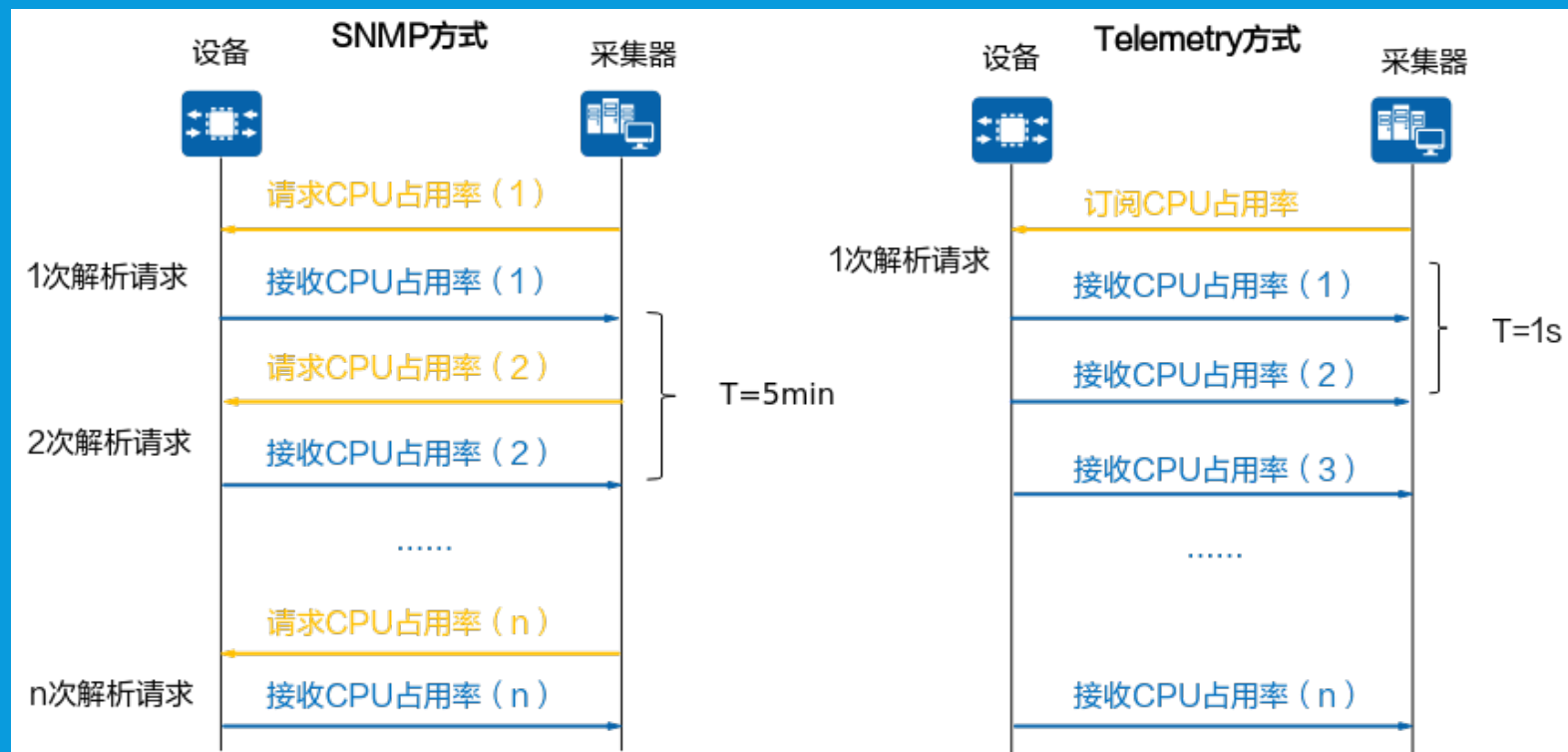
- Telemetry 技术原理及优势
- Telemetry 数据订阅
- Telemetry 采样数据集编码格式

## 2.1 Telemetry 技术原理及优势

- Telemetry也叫Network Telemetry（网络遥测技术），设备通过“推模式（Push Mode）”周期性地主动向采集器上送设备信息，提供更实时、更高速、更精确的网络监控功能。
- 具体来说，Telemetry 按照 YANG 模型组织数据，利用 GPB（Google Protocol Buffer）格式或JSON格式编码，并通过 gRPC（Google Remote Procedure Call Protocol）协议或UDP协议传输数据。

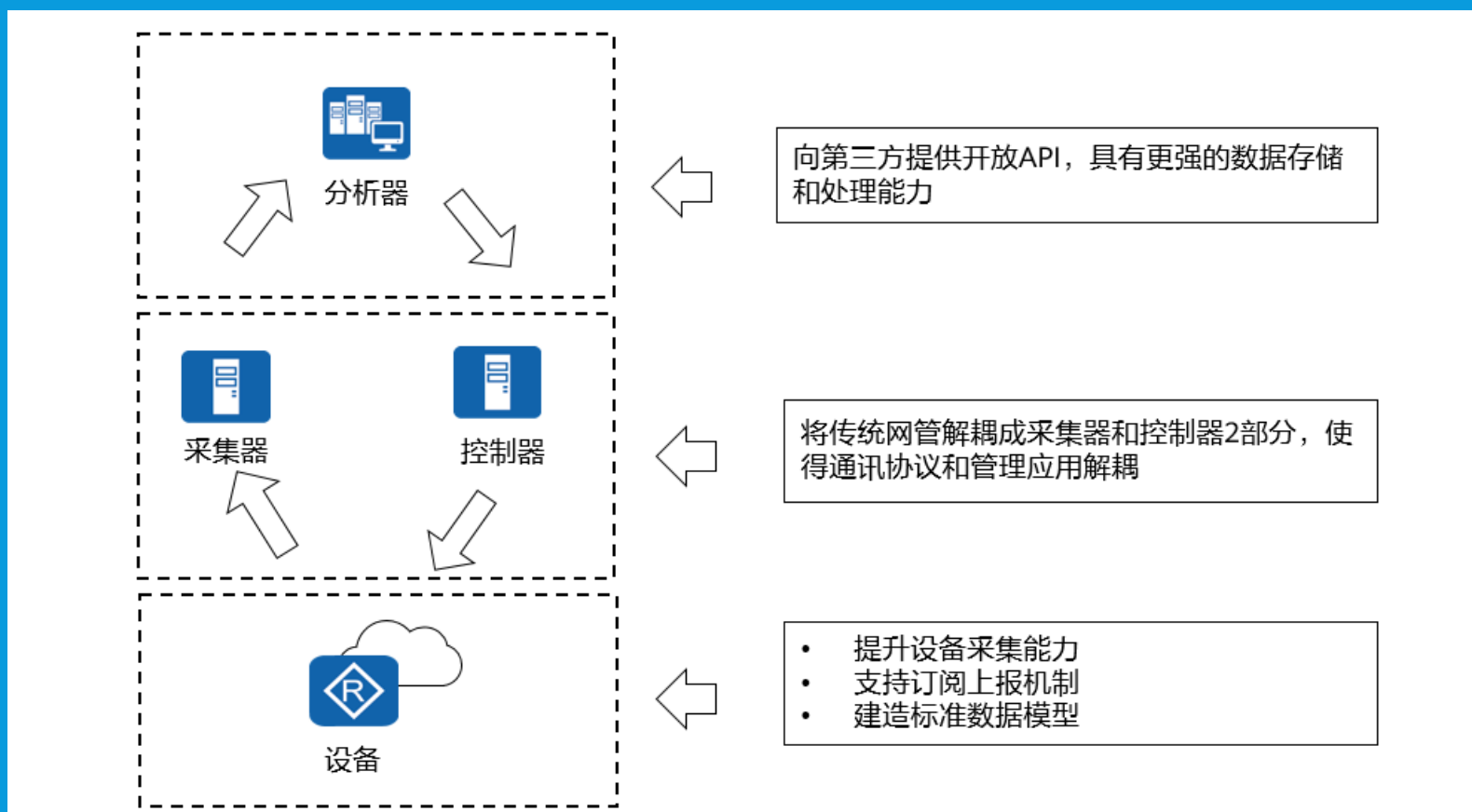
## 2.1 Telemetry 技术原理及优势

- SNMP是小型简单网络的主流数据采集技术；
- Telemetry在大型数据网络中表现出许多优势。
- SNMP 采用“拉模式”，Telemetry采用“推模式”。



## 2.1 Telemetry 技术原理及优势

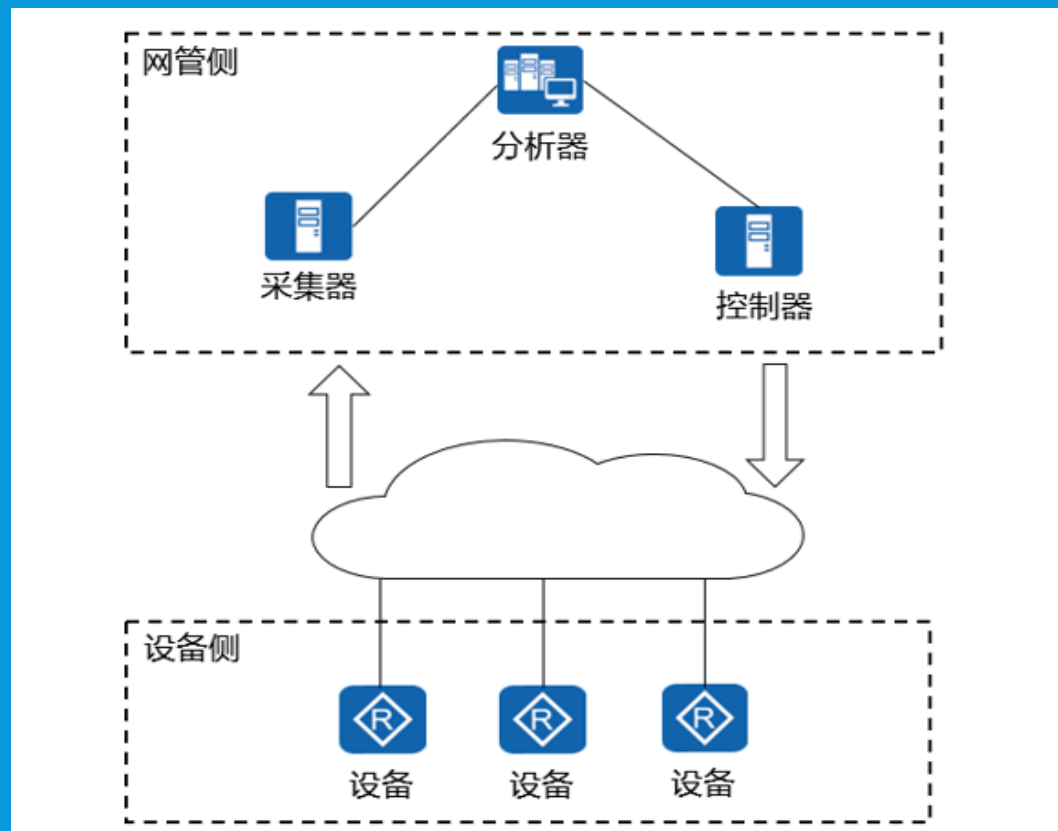
### ■ Telemetry 技术特点:



## 2.1 Telemetry 技术原理及优势

- Telemetry 网络模型分广义和狭义两种：

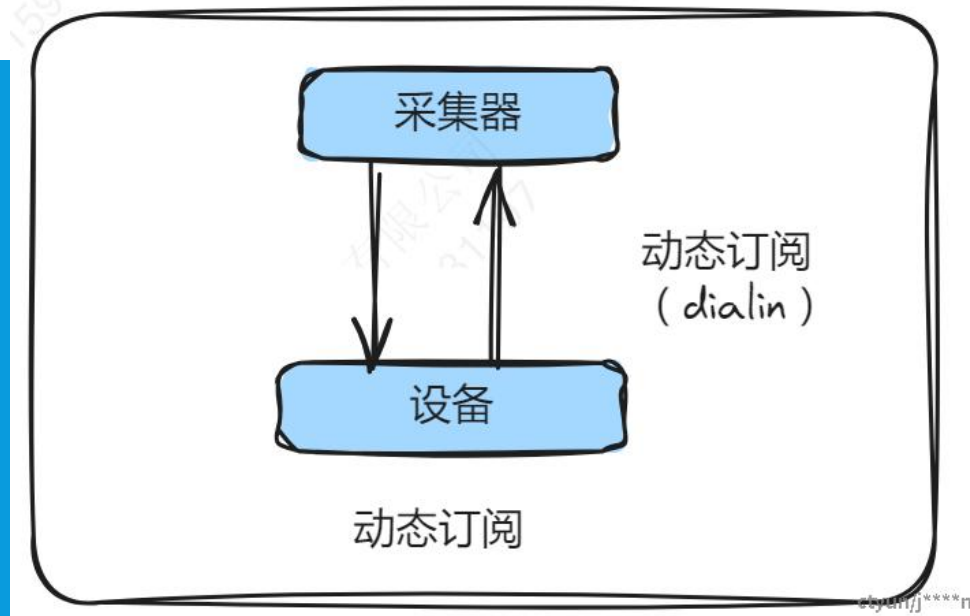
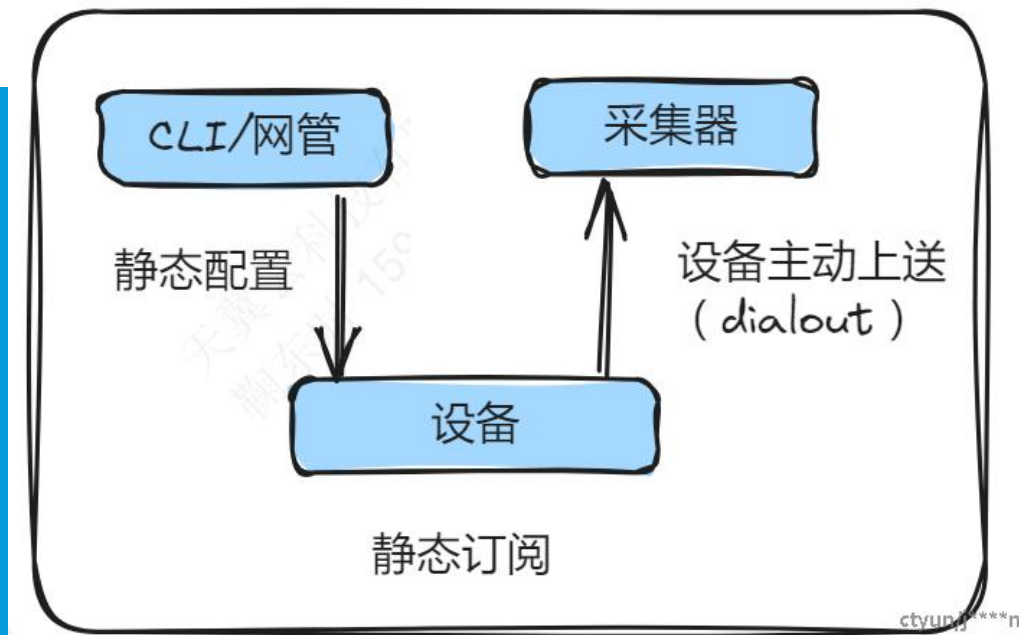
- 广义 Telemetry：包括采集器、分析器、控制器和设备共同构成的一个自闭环系统。
- 狭义 Telemetry：指设备采样数据上送给采集器的功能。



广义Telemetry网络模型



## 2.2 Telemetry 数据订阅



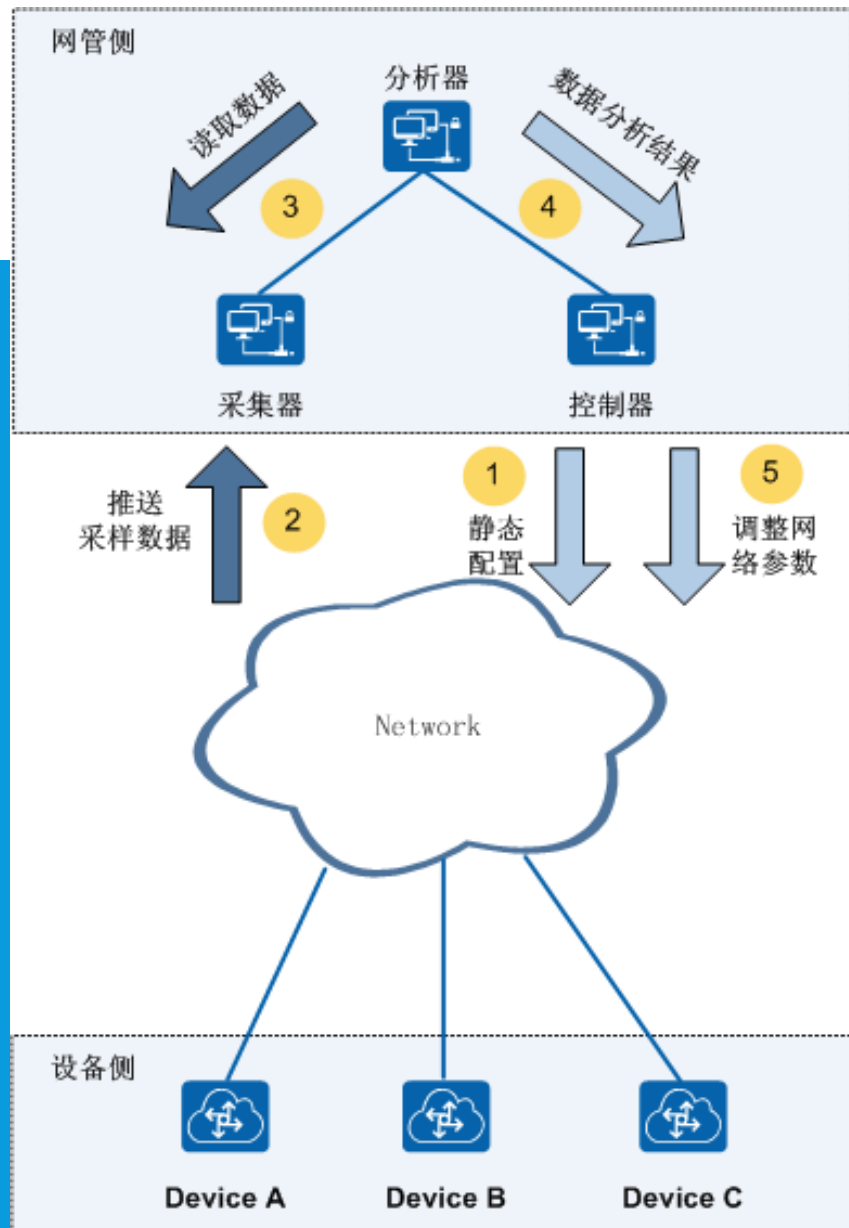
静态订阅是指设备作为客户端，采集器作为服务端。由设备主动发起到采集器的连接，进行数据采集上送。多用于长期巡检。

动态订阅是指设备作为服务端，采集器作为客户端发起到设备的连接。由设备进行数据采集上送。多用于短期监控。

## 2.2 Telemetry 数据订阅

完整的Telemetry系统静态订阅分为5个过程。

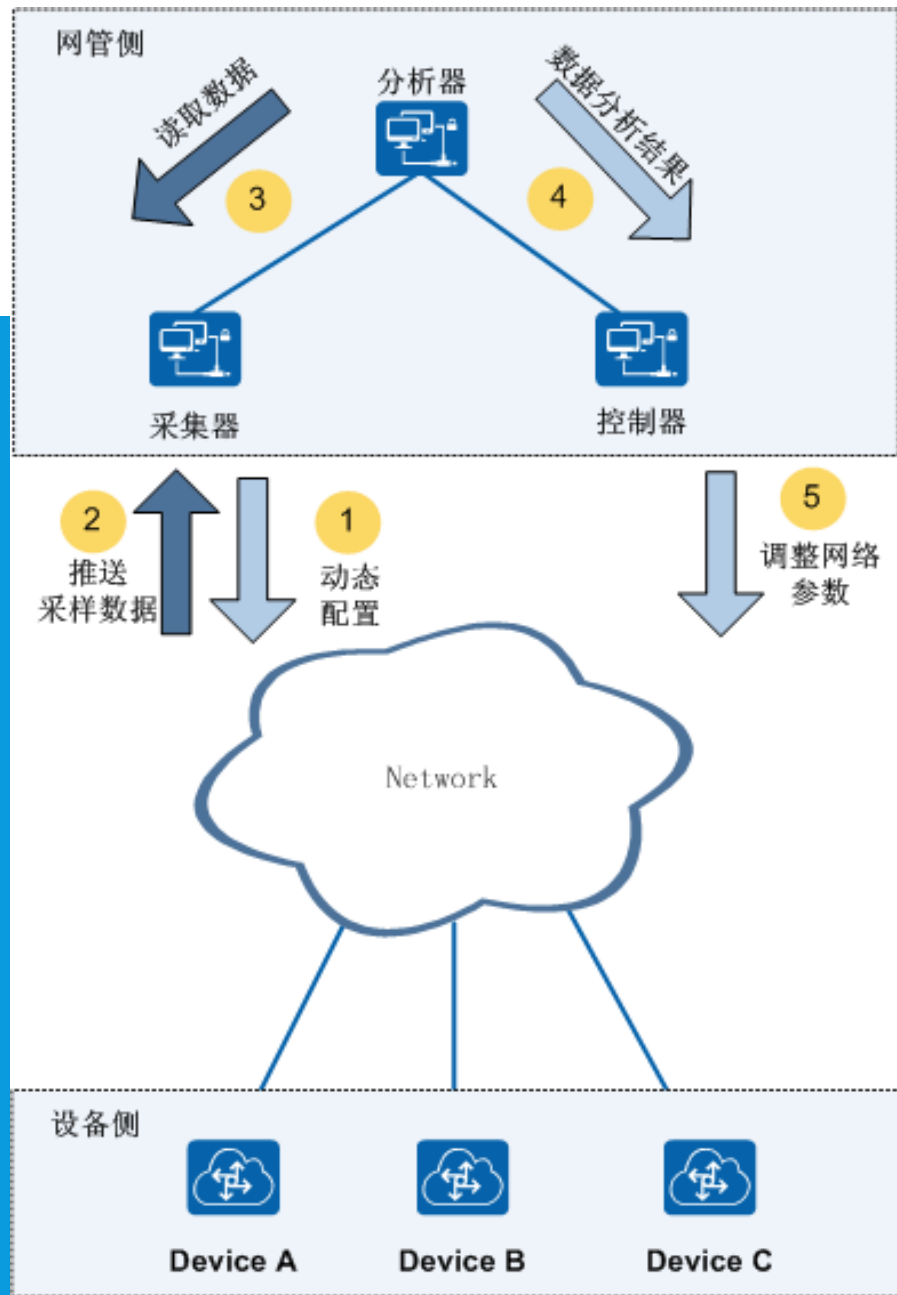
- 静态配置：控制器通过命令行配置支持Telemetry的设备，订阅数据源，完成数据采集。
- 推送采集数据，设备依据订阅数据方式，将采集完成的数据，上报给采集器进行接收和存储。
- 读取数据，分析器读取采集器存储的采集数据。
- 分析数据，分析器分析读取到的采集数据，并将分析结果发给控制器，便于控制器对网络进行配置管理，及时调优网络。
- 调整网络参数，控制器将网络需要调整的配置下发给设备，配置下发生效后，新的采集数据又会上报到采集器，此时分析器可以分析调优后的网络效果是否符合预期，直到调优完成后，整个业务流程形成闭环。



## 2.2 Telemetry 数据订阅

完整的Telemetry系统动态订阅也分为5个过程。

- 动态配置：支持Telemetry的设备在完成gRPC服务的相关配置后，由采集器下发动态配置到设备，完成数据采集
- 推送采集数据，设备依据订阅数据方式，将采集完成的数据，上报给采集器进行接收和存储。
- 读取数据，分析器读取采集器存储的采集数据。
- 分析数据，分析器分析读取到的采集数据，并将分析结果发给控制器，便于控制器对网络进行配置管理，及时调优网络。
- 调整网络参数，控制器将网络需要调整的配置下发给设备，配置下发生效后，新的采集数据又会上报到采集器，此时分析器可以分析调优后的网络效果是否符合预期，直到调优完成后，整个业务流程形成闭环。



## 2.3 Telemetry 采样数据集编码格式

- Telemetry的采样数据主要包括如下3个方面：

**原始数据：**Telemetry采样的原始数据可来自网络设备的转发面、控制面和管理面，目前支持采集设备的接口流量统计、CPU或内存数据等信息。

**数据模型：**Telemetry基于YANG模型组织采集数据。YANG是一种数据建模语言，用于设计可以作为各种传输协议操作的配置数据模型、状态数据模型、远程调用模型和通知机制等。

**性能指标：**Telemetry技术目前支持特定的采样传感器路径采集指定的数据信息。

## 2.3 Telemetry 采样数据集编码格式

- 用户通过采样路径来描述自己需要采样的数据。设备上的数据已经通过YANG模型描述说明，基于YANG模型和它的子树路径可以构成采样路径。

## 2.3 Telemetry 采样数据集编码格式

- 采样周期，是指周期性的主动向采集器上送设备的接口流量统计、CPU或内存数据等信息。采样周期受到采样实例数、采样数据源的周期和CPU占用率等因素影响。
- 在设备和采集器之间传输数据时，需要对数据进行编码，当前支持如下两种编码格式：
  - GPB (Google Protocol Buffer) 编码格式
  - JSON编码格式

## 2.3 Telemetry 采样数据集编码格式

- GPB编码格式，是一种与语言无关、平台无关、扩展性好的用于通信协议数据存储的序列化结构数据格式。它是一种二进制编码。
- gRPC协议用GPB编码格式（文件名后缀为.Proto）承载数据。
- 目前，GPB包括v2和v3两个版本，大多数华为设备当前支持的GPB版本是v3。

GPB编码解析前	GPB编码解析后
<pre>{   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"         }}       }     }   }}   8:1515727243419   9:10000   10:"OK"   11:"CE16800"   12:0 }</pre>	<pre>{   "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"           }}         }       }     }}   },   "collection_end_time":"2018/1/12 11:20:43.419",   "current_period":10000,   "except_desc":"OK",   "product_name":"CE16800",   "encoding":Encoding_GPB }</pre>

## 2.3 Telemetry 采样数据集编码格式

JSON编码解析前	JSON编码解析后
<pre>{   1: "HUAWEI"   2: "s4"   3: "huawei-ifm: ifm/interfaces/interface"   4: 46   5: 1515727243419   6: 1515727243514   8: 1515727243419   9: 10000   10: "OK"   11: "CE16800"   12: 1   14: {     "row": [{       "timestamp": "2018/1/12 11:20:43.419",       "content": {         "interfaces": {           "interface": [{             "ifAdminStatus": 1,             "ifIndex": 2,             "ifName": "Eth-Trunk1"           }]         }       }     }]   } }</pre>	<pre>{   "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",   "collection_end_time": "2018/1/12 11:20:43.419",   "current_period": 10000,   "except_desc": "OK",   "product_name": "CE16800",   "encoding": "Encoding_JSON",   "data_str": {     "row": [{       "timestamp": "2018/1/12 11:20:43.419",       "content": {         "interfaces": {           "interface": [{             "ifAdminStatus": 1,             "ifIndex": 2,             "ifName": "Eth-Trunk1"           }]         }       }     }]   } }</pre>

Telemetry支持两种风格的JSON编码格式：

- 纯JSON编码格式：Telemetry层和业务数据层均为JSON编码格式。
- 混合JSON编码格式：Telemetry层为GPB编码格式，业务数据层为JSON编码格式。



### 3. Proto 文件

- Telemetry支持的三个公共Proto文件
- Telemetry采集数据时使用的业务数据Proto文件
- 华为设备的Proto文件结构

## 3.1 Telemetry支持的三个公共Proto文件

[https://support.huawei.com/enterprise/zh/doc/EDOC1100216375/ad4096aa#ZH-CN\\_TOPIC\\_0303965542](https://support.huawei.com/enterprise/zh/doc/EDOC1100216375/ad4096aa#ZH-CN_TOPIC_0303965542)

Proto文件用于定义GPB编码的编码规则，包含公共Proto文件和业务数据Proto文件。

Telemetry提供3个公共的Proto文件，支持数据上送和订阅功能：

1. huawei-grpc-dialout.Proto文件是RPC头文件。设备作为客户端对外推送数据时（即为静态订阅），该文件定义了RPC接口。
2. huawei-grpc-dialin.Proto文件是RPC头文件。设备作为服务端对外推送数据时（即为动态订阅），该文件定义了RPC接口。
3. Telemetry头定义文件huawei-telemetry.Proto，定义了Telemetry采样数据上送时的数据头，包括采样路径，采样时间戳等重要信息。

## 3.2 Telemetry 采集数据时使用的业务数据Proto文件

[https://support.huawei.com/enterprise/zh/doc/EDOC1100216375/ad4096aa#ZH-CN\\_TOPIC\\_0303965542](https://support.huawei.com/enterprise/zh/doc/EDOC1100216375/ad4096aa#ZH-CN_TOPIC_0303965542)

- ◆设备提供多个业务数据Proto文件，用于定义具体业务数据的GPB编码，采集器侧需要根据实际要监控的业务选择对应Proto文件。
- ◆网管通过Telemetry采样设备数据时，首先要根据业务确定需要从设备上采样的数据范围，从而确定采样路径，再从采样路径获取相应的Proto文件。
- ◆例如，当需要采样接口流量数据时，意味着采样路径是：

huawei-ifm:ifm/interfaces/interface/mib-statistic

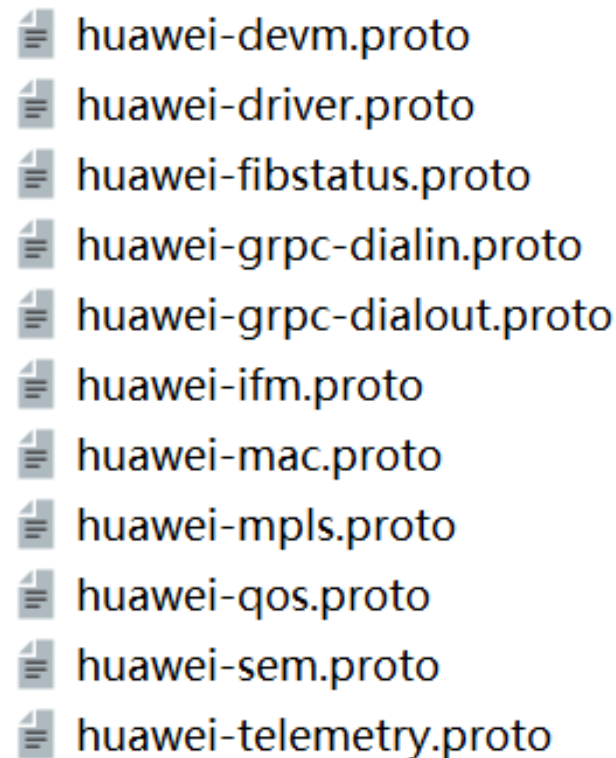
## 3.3 华为设备的Proto文件结构

### ◆访问华为企业用户技术支持网站

(<http://support.huawei.com/enterprise>) 或运营商用户技术支持网站

(<http://support.huawei.com/carrier>) , 搜索相应的设备型号及版本。

### ◆进入软件下载页面获取相应版本的Proto文件。



A list of 11 Proto files, each preceded by a small icon of a document with three horizontal lines. The files are listed vertically in a white box with a blue background.

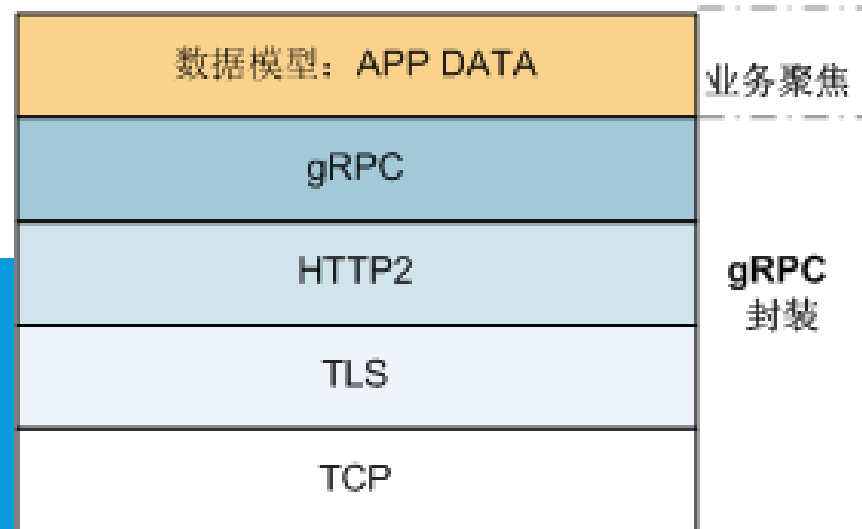
- huawei-devm.proto
- huawei-driver.proto
- huawei-fibstatus.proto
- huawei-grpc-dialin.proto
- huawei-grpc-dialout.proto
- huawei-ifm.proto
- huawei-mac.proto
- huawei-mpls.proto
- huawei-qos.proto
- huawei-sem.proto
- huawei-telemetry.proto

# 4.gRPC 协议

- 协议栈
- 网络架构
- 服务模式

## 4.1 协议栈

gRPC协议（google Remote Procedure Call Protocol）是谷歌发布的一个基于HTTP2协议承载的高性能、通用的RPC开源软件框架，共有五层结构。



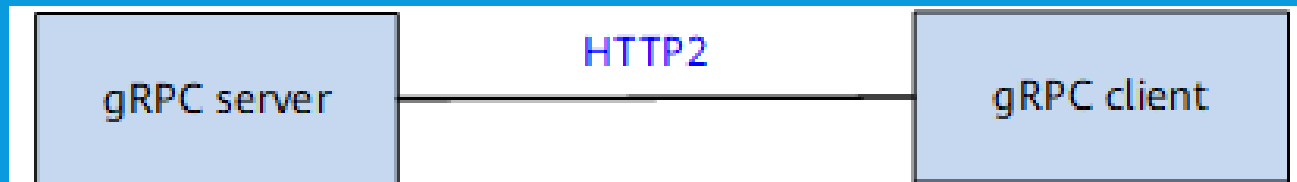
字段	说明
TCP层	底层通信协议，基于TCP连接。
TLS层	该层是可选的，基于TLS加密通道。
HTTP2层	gRPC承载在HTTP2协议上，利用了HTTP2的双向流、流控、头部压缩、单连接上的多路复用请求等特性。
gRPC层	远程过程调用，定义了远程过程调用的协议交互格式。
数据模型层	通信双方需要了解彼此的数据模型，才能正确交互。

## 4.2 网络架构

gRPC采用客户端和服务端模型，工作机制如下：

- 服务器通过监测指定服务端口来等待客户端的连接请求。
- 用户通过执行客户端程序登录到服务器。
- 客户端调用.Proto文件提供的gRPC方法发送请求消息。
- 服务器回复应答消息。

支持静态订阅和动态订阅场景。



## 4.3 服务模式

RPC服务是通过参数和返回值类型来指定可以远程调用的方法。gRPC使用Proto Buffers语言来定义服务的方法、参数和返回值，这些方法定义在“.Proto”文件中，定义格式为：

```
service 服务名称 {  
  rpc 方法名称 (stream 参数名称) returns (stream 返回值) {};  
}
```

例如下面的具体定义：

```
service gRPCDataService {  
  rpc dataPublish (stream serviceArgs) returns (stream serviceArgs) {};  
}
```



## 4.3 服务模式

根据stream关键字的所在位置，gRPC方法可以有如下几种服务模式

工作模式	说明和示例
简单模式	一问一答的简单交互。 示例: <code>rpc Cancel(CancelArgs) returns(CancelReply) {};</code>
服务端流模式	客户端发送一个请求，服务端不断返回数据给客户端。 示例: <code>rpc Subscribe(SubsArgs) returns(<b>stream</b> SubsReply) {};</code>
客户端流模式	客户端不断向服务端推送数据，并等待服务端返回应答。 示例: <code>rpc LotsOfGreetings(<b>stream</b> HelloRequest) returns (HelloResponse) {};</code>
双向流模式	客户端和服务端双方都可以分别通过一个读写数据流来发送一系列消息。两个数据流相互独立，通信双方可以根据数据流内容进一步控制交互，拥有最大的灵活性。 示例: <code>rpc dataPublish(<b>stream</b> serviceArgs) returns(<b>stream</b> serviceArgs) {};</code>

## 5. 华为设备配置设备侧数据订阅指令

- 配置静态订阅
- 配置动态订阅

## 5.1 配置静态订阅

### ■配置采样数据要推送的目标采集器

#### 操作步骤

1. 进入系统视图。

```
system-view
```

2. 进入Telemetry视图。

```
telemetry
```

3. 创建采样数据目标采集器所在的目标组，并进入Destination-group视图。

```
destination-group destination-group-name
```

4. 配置目标采集器的IP地址、端口号、推送协议和加密方式。

- 目标采集器的IP地址类型为IPv4时，进行如下配置：

```
ipv4-address ip-address-ip4 port port-value [ vpn-instance vpn-name ] [ protocol { grpc [ no-tls ] } ]
```

- 目标采集器的IP地址类型为IPv6时，进行如下配置：

```
ipv6-address ip-address-ipv6 port port-value [ vpn-instance vpn-name ] [ protocol { grpc [ no-tls ] } ]
```

### ■检查配置：

**执行命令**`display telemetry destination [ dest-name ]`，查看推送目标组信息

## 5.1 配置静态订阅

### ■ 配置采样路径和过滤条件

system-view # 进入系统视图。

telemetry # 进入Telemetry视图。

sensor-group sensor-name # 创建采样传感器组名称，并进入Sensor-group视图。

sensor-path path # 配置该采样传感器组的非自定义事件，主要包括采样路径和过滤条件等。配置非自定义事件的采样路径。

depth depth-value # （可选）配置非自定义事件的数据采样深度。

filter filter-name # 创建采样路径的条件过滤器。创建采样路径的条件过滤器名称，并进入过滤器视图。

op-field field op-type { eq | gt | ge | lt | le } op-value value # 配置过滤条件。

condition-relation { and | or } #配置多个过滤条件间的逻辑运算关系。

# 5.1 配置静态订阅

## ■ 创建订阅

### 操作步骤

#### 1. 创建订阅。

##### a. 进入系统视图。

```
system-view
```

##### b. 进入Telemetry视图。

```
telemetry
```

##### c. 创建订阅用于关联目标采集器所在的目标组和采样传感器组，并进入Subscription视图。

```
subscription subscription-name
```

##### d. 关联目标采集器所在的目标组。

```
destination-group destination-name
```

##### e. 关联采样传感器组，并可配置该采样传感器组的采样周期、冗余抑制和心跳间隔。

```
sensor-group sensor-name [ sample-interval sample-interval { [ suppress-redundant ] | [ heartbeat-interval heartbeat-interval ] } * ]
```

■ 执行命令display telemetry subscription [ subscription-name ]，查看订阅信息。

■ 执行命令display telemetry destination [ dest-name ]，查看上送目标组信息。

## 5.2 配置动态订阅

- Telemetry动态订阅是指设备作为服务端，采集器作为客户端发起到设备的连接，由设备进行数据采集上送。

1. 配置gRPC服务器相关信息，并使能gRPC服务。

a. 进入系统视图。

```
system-view
```

b. 进入gRPC视图。

```
grpc
```

c. 进入服务器视图。

- 当网络是IPv4网络时，进入gRPC服务器视图。

```
grpc server
```

- 网络是IPv6网络时，进入gRPC IPv6服务器视图。

```
grpc server ipv6
```

d. 配置动态订阅时侦听的源IPv4地址或源IPv6地址。

```
source-ip ip-address [ vpn-instance vpn-instance-name ]
```

e. 配置动态订阅时侦听的端口号。

```
server-port port-number
```

缺省情况下，配置动态订阅时侦听的端口号为57400。

- 执行命令display telemetry destination [ dest-name ]，查看推送目标组信息

## 6. Grpcio-tools模 块

- 模块安装
- 演示使用gprc模块从交换机采集CPU的状态信息

## 6.1 模块安装

grpc 是一个 google 开源的 rpc 库，支持多种语言，以下是 python 版本的 grpc。

安装：

`pip install grpcio`

`pip install grpcio-tools`

```
(ensp_py1) C:\Users\USSTz>pip install grpcio
Collecting grpcio
  Downloading grpcio-1.66.2-cp310-cp310-win_amd64.whl.metadata (4.0 kB)
  Downloading grpcio-1.66.2-cp310-cp310-win_amd64.whl (4.3 MB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.3/4.3 MB 57.9 kB/s eta 0:00:00
Installing collected packages: grpcio
Successfully installed grpcio-1.66.2

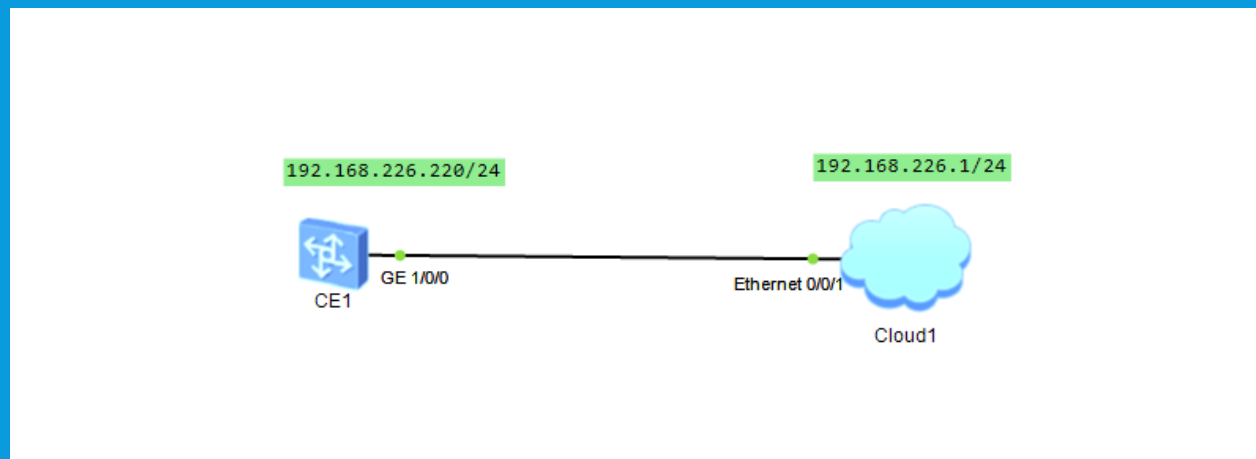
(ensp_py1) C:\Users\USSTz>pip install grpcio-tools
Collecting grpcio-tools
  Downloading grpcio_tools-1.66.2-cp310-cp310-win_amd64.whl.metadata (5.5 kB)
Collecting protobuf<6.0dev,>=5.26.1 (from grpcio-tools)
  Downloading protobuf-5.28.2-cp310-abi3-win_amd64.whl.metadata (592 bytes)
Requirement already satisfied: grpcio>=1.66.2 in c:\users\usstz\anaconda3\envs\ensp_py1\packages (from grpcio-tools) (1.66.2)
Requirement already satisfied: setuptools in c:\users\usstz\anaconda3\envs\ensp_py1\packages (from grpcio-tools) (72.1.0)
  Downloading grpcio_tools-1.66.2-cp310-cp310-win_amd64.whl (1.1 MB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.1/1.1 MB 11.9 kB/s eta 0:00:00
  Downloading protobuf-5.28.2-cp310-abi3-win_amd64.whl (431 kB)
Installing collected packages: protobuf, grpcio-tools
Successfully installed grpcio-tools-1.66.2 protobuf-5.28.2
```



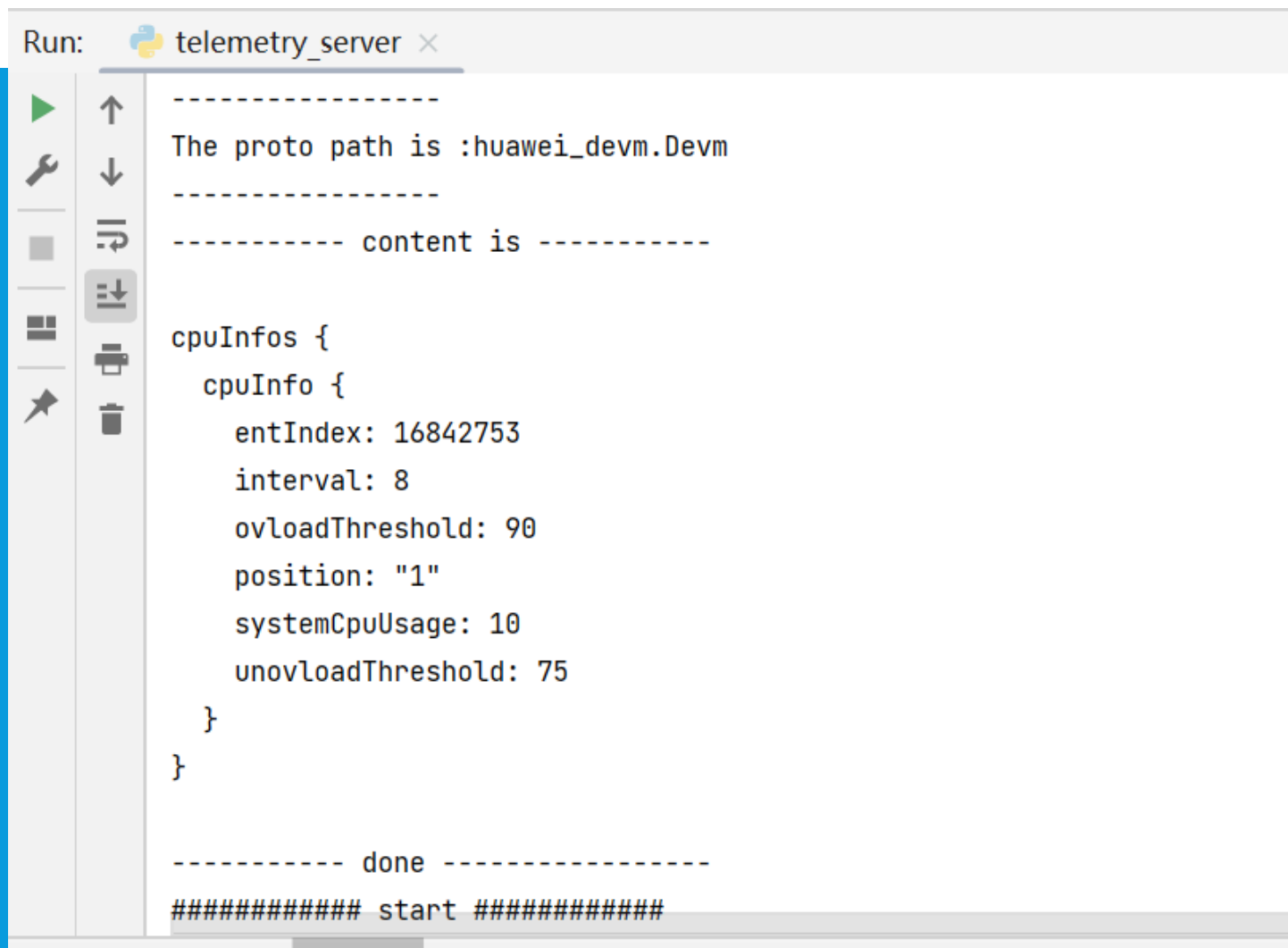
## 6.2 演示使用gprc模块从CE12800采集CPU的状态信息

【任务目标】指导教师给出网络拓扑图截图，配置局域网，并通过telemetry静态订阅方式监控CE12800设备的CPU使用率。需要完成的任务如下。

- (1) 创建网络拓扑图并配置参数。
- (2) 配置CE12800设备SSH服务及Telemetry静态订阅服务。
- (3) 编写Python脚本。
- (4) 运行Python脚本。



## 6.2 演示使用gprc模块从CE12800采集CPU的状态信息



```
Run: telemetry_server x
-----
The proto path is :huawei_devvm.Devm
-----
----- content is -----
cpuInfos {
  cpuInfo {
    entIndex: 16842753
    interval: 8
    overloadThreshold: 90
    position: "1"
    systemCpuUsage: 10
    unloadThreshold: 75
  }
}
----- done -----
##### start #####
```