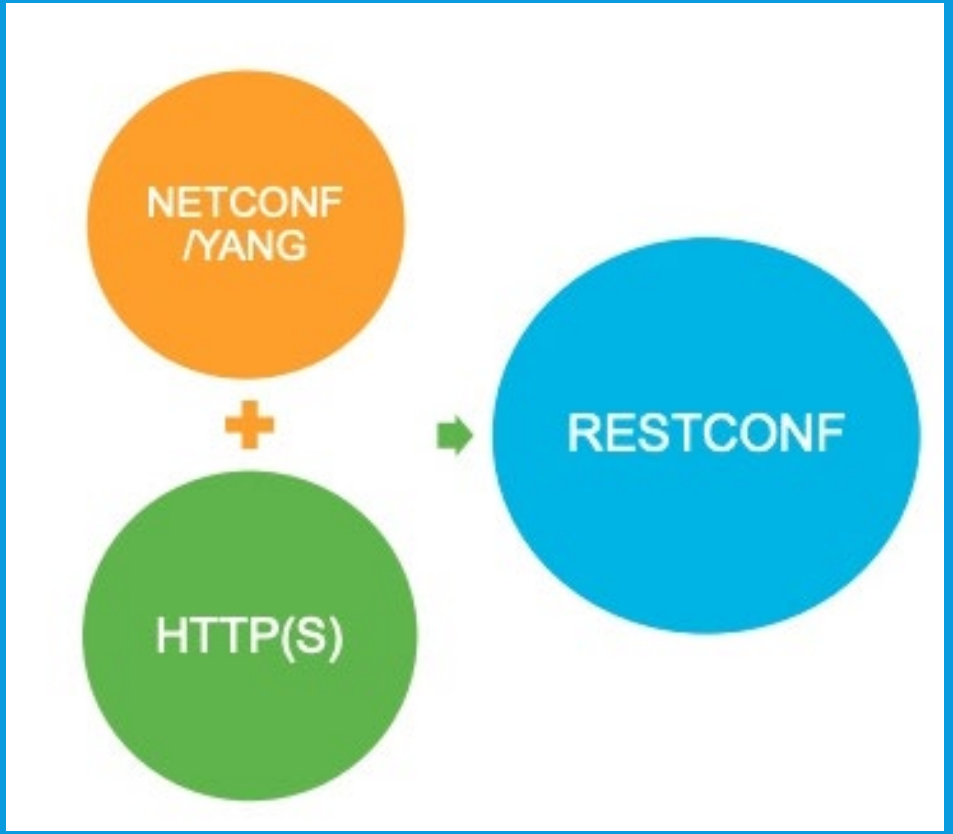
使用RESTCONF协议配置网络

学院：信息工程学院

教师：张迁

序言

- 随着网络规模的增大、复杂性的增加，自动化运维的需求日益增加。NETCONF提供基于RPC机制的应用编程接口。但是NETCONF已无法满足网络发展中对设备编程接口提出的新要求，希望能够提供支持WEB应用访问和操作网络设备的标准化接口。
- RESTCONF是在融合NETCONF和HTTP协议的基础上发展而来的。RESTCONF以HTTP协议的方法提供了NETCONF协议的核心功能，编程接口符合IT业界流行的RESTful风格，为用户提供高效开发WEB化运维工具的能力。



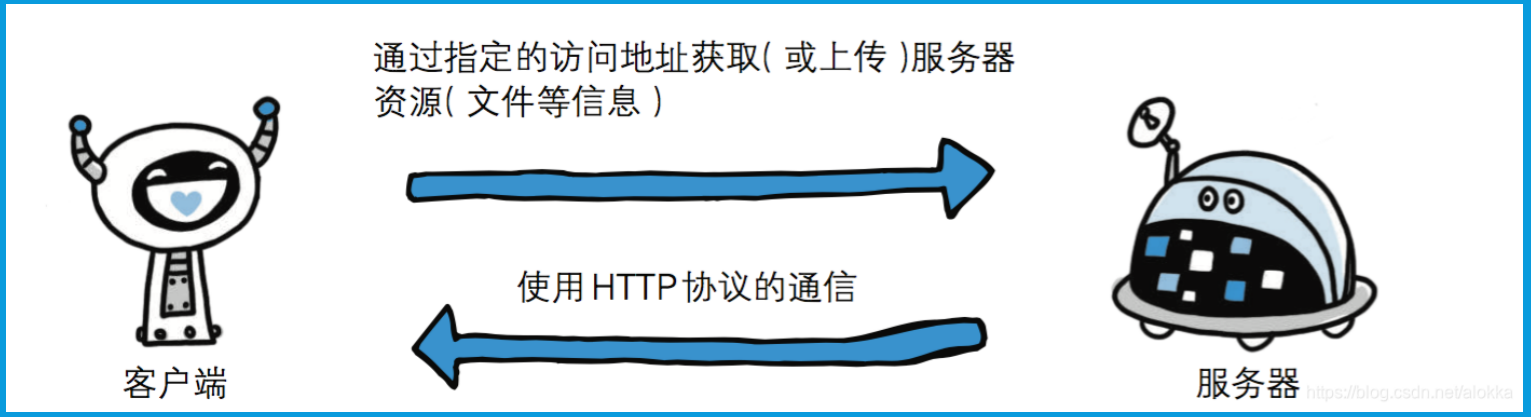
目录

1. HTTP协议

- HTTP协议概述
- HTTP客户端请求报文及示例
- HTTP服务器响应报文及示例

1.1 HTTP协议概述

- 超文本传输协议HTTP（HyperText Transfer Protocol）是一种用于分布式、协作式和超媒体信息系统的应用层协议。HTTP是万维网的数据通信的基础。
- HTTP的发展是由蒂姆·伯纳斯-李于1989年在欧洲核子研究组织（CERN）所发起。HTTP的标准制定由万维网协会（World Wide Web Consortium，W3C）和互联网工程任务组（Internet Engineering Task Force，IETF）进行协调，最终发布了一系列的RFC，其中最著名的是1999年6月公布的 RFC 2616，定义了HTTP协议中现今广泛使用的一个版本——HTTP 1.1。
- 2014年12月，互联网工程任务组（IETF）的Hypertext Transfer Protocol Bis（httpbis）工作小组将HTTP/2标准提议递交至IESG进行讨论，于2015年2月17日被批准。 HTTP/2标准于2015年5月以RFC 7540正式发表，取代HTTP 1.1成为HTTP的实现标准。



1.1 HTTP协议概述

HTTP是客户端浏览器或其他程序与Web服务器之间的应用层通信协议。在Internet上的Web服务器上存放的都是超文本信息，客户机需要通过HTTP协议传输所要访问的超文本信息。HTTP包含命令和传输信息，不仅可用于Web访问，也可以用于其他因特网/内联网应用系统之间的通信，从而实现各类应用资源超媒体访问的集成。

我们在浏览器的地址栏里输入的网站地址叫做URL（Uniform Resource Locator，统一资源定位符）。 就像每家每户都有一个门牌地址一样，每个网页也都有一个Internet地址。当你在浏览器的地址框中输入一个URL或是单击一个超级链接时，URL就确定了要浏览的地址。浏览器通过超文本传输协议(HTTP)，将Web服务器上站点的网页代码提取出来，并翻译成漂亮的网页。

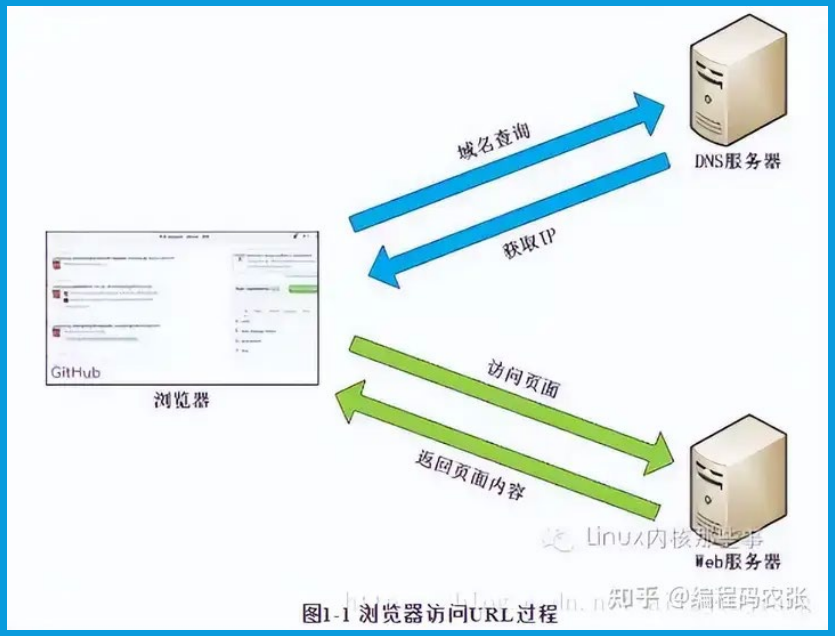


图1-1 浏览器访问URL过程

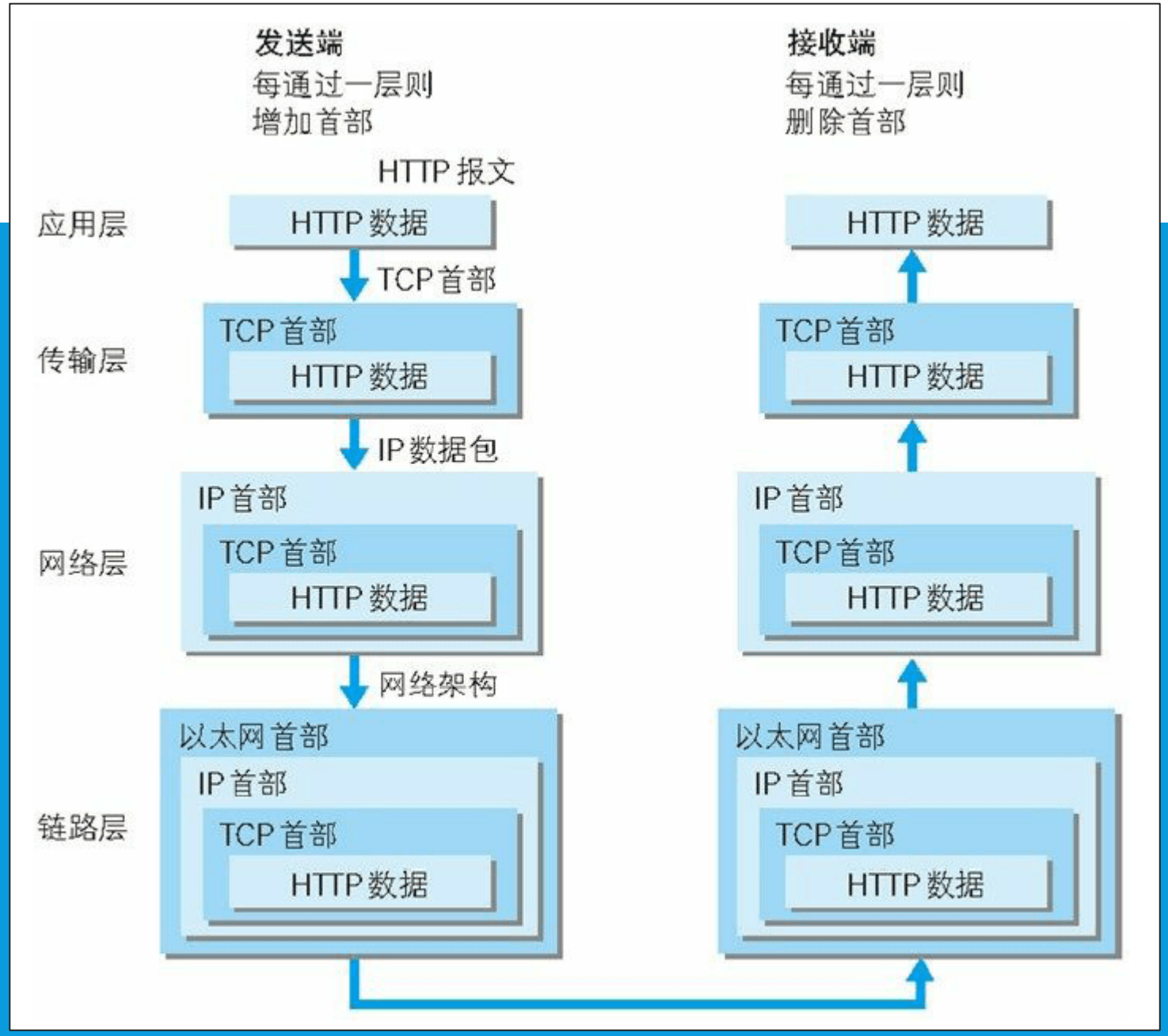
1.1 HTTP协议概述

- HTTP是一种请求/响应协议，基于TCP/IP协议来传递数据，默认工作在TCP的80端口。

OSI七层模型	TCP/IP概念层模型	功能	TCP/IP协议族
应用层	应用层	文件传输，电子邮件，文件服务，虚拟终端	TFTP，HTTP，SNMP，FTP，SMTP，DNS，Telnet
表示层		数据格式化，代码转换，数据加密	没有协议
会话层		解除或建立与别的接点的联系	没有协议
传输层	传输层	提供端对端的接口	TCP，UDP
网络层	网络层	为数据包选择路由	IP，ICMP，RIP，OSPF，BGP，IGMP
数据链路层	链路层	传输有地址的帧以及错误检测功能	SLIP，CSLIP，PPP，ARP，RARP，MTU
物理层		以二进制数据形式在物理媒体上传输数据	ISO2110，IEEE802，IEEE802.2

1.1 HTTP 协议概述

- 利用 TCP/IP 协议族进行网络通信时，会通过分层顺序与对方进行通信。发送端从应用层往下走，接收端则从链路层往上走。



1.1 HTTP协议概述

- HTTP基于客户端/服务端（C/S）的架构模型。HTTP请求及响应有如下五个步骤：

① 客户端与服务器建立TCP连接。
② 客户端发送HTTP请求。请求报文由请求行、请求头部、空行和请求数据四部分组成。
③ 服务器接受请求并返回HTTP响应。响应报文由状态行、响应头部、空行和响应正文四部分组成。
④ 释放TCP连接。
⑤ 客户端浏览器解析响应报文并显示。客户端浏览器依次解析状态行、响应头部、响应正文并显示。如正文数据为HTML，客户端根据HTML的语法对其进行格式化，并在浏览器窗口中显示。

- HTTP客户端通常是浏览器，Web服务器可以是Apache服务器，IIS服务器（Internet Information Services）等

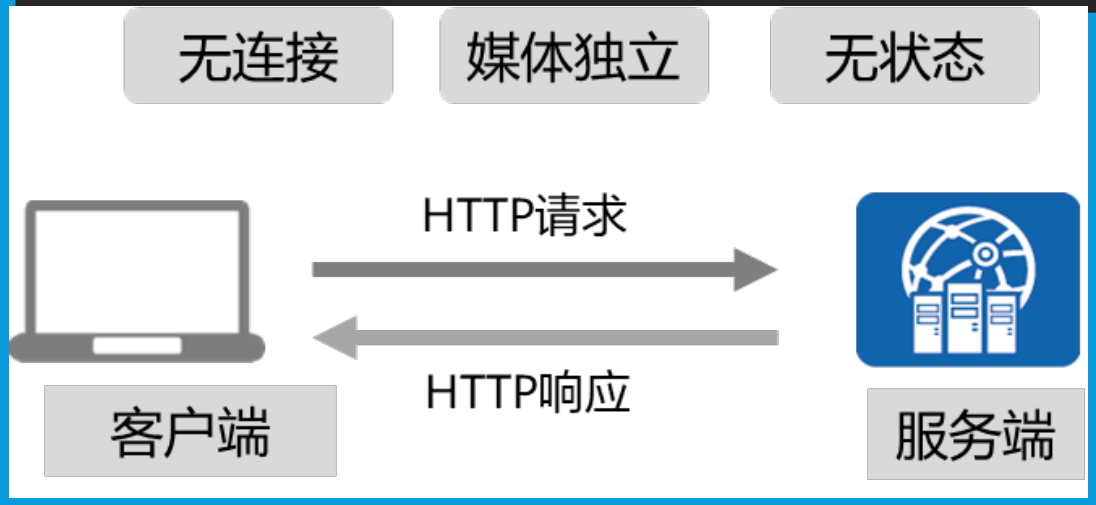
1.1 HTTP协议概述

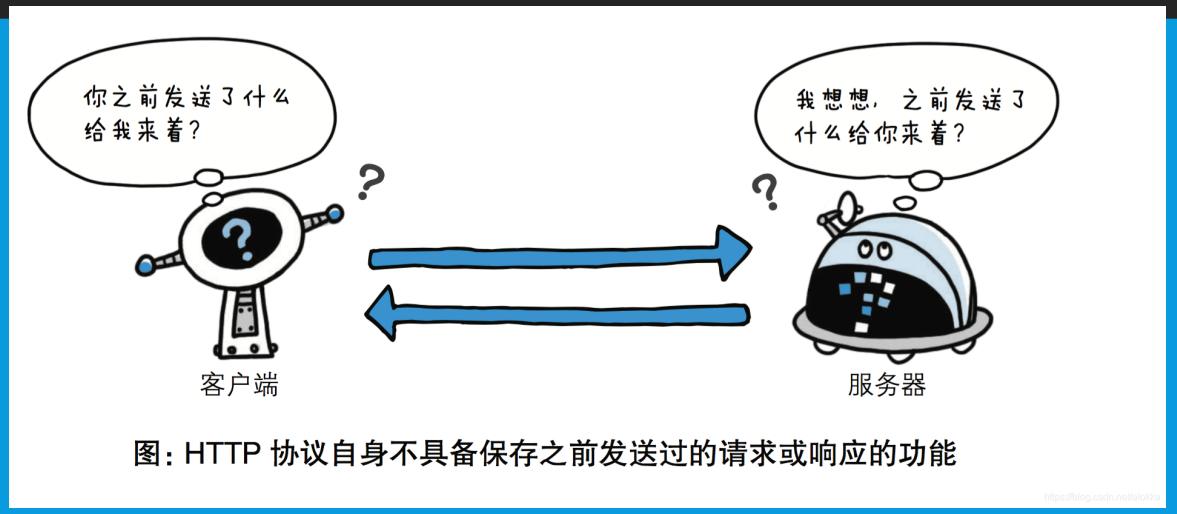
- HTTP具有如下特点：

① 无连接：无连接的含义是限制每次连接只处理一个请求。服务器处理完客户的请求后就断开连接。

② 媒体独立：这意味着，只要客户端和服务器知道如何处理的数据内容，任何类型的数据都可以通过HTTP发送。客户端以及服务器通过头部字段指定适合的MIME Type内容类型。

③ 无状态：无状态是指协议对于事务处理没有记忆能力，这样做利于更快地处理大量事务，确保协议的可伸缩性。。

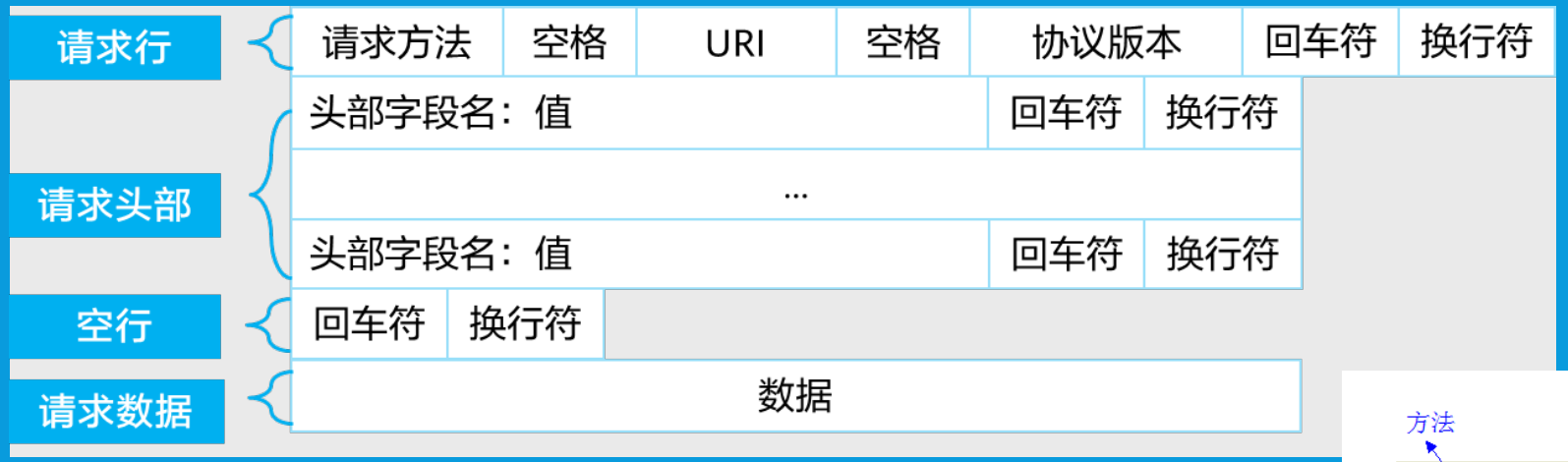


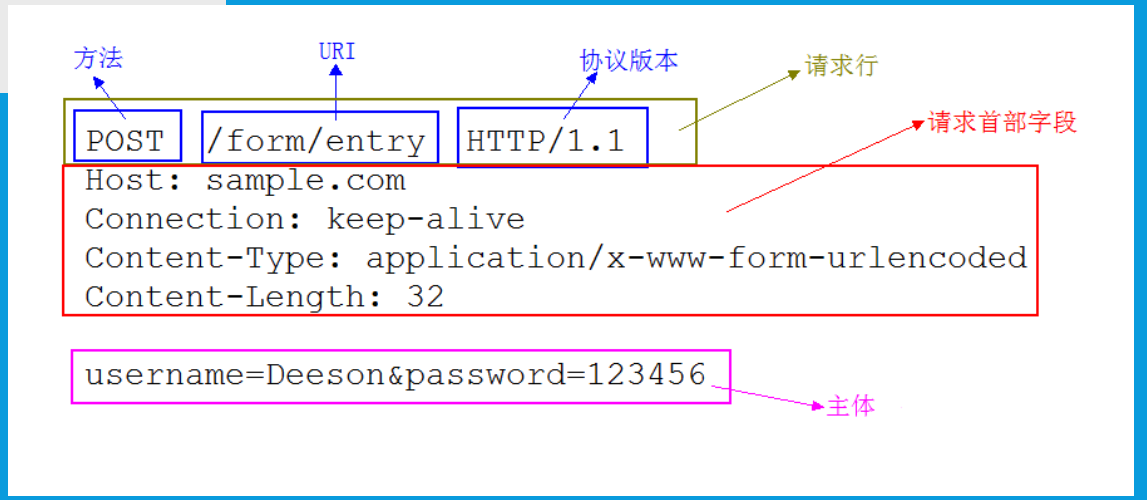


图：HTTP 协议自身不具备保存之前发送过的请求或响应的功能

1.2 HTTP客户端请求报文及示例

- 客户端发送的HTTP请求消息包括由请求行、请求头部、空行和请求数据四个部分组成，下图给出了请求报文的一般格式。





1.2 HTTP客户端请求报文及示例

- 请求行由请求方法字段、URI字段和HTTP协议版本字段3个字段组成。

① 请求方法：HTTP使用的请求方法，比如常见的GET/POST等。HTTP客户程序(例如浏览器)，向服务器发送请求的时候必须指明请求类型。

② URI：URI是一个统一资源标识符，它标识了请求所针对的资源。

③ 协议版本：协议版本旨在允许发送方指示消息的格式和理解后续HTTP通信的能力。

请求行示例：GET http://www.w3.org/pub/WWW/TheProject.html HTTP/1.1

1.2 HTTP客户端请求报文及示例

- 根据HTTP标准，HTTP请求可以使用多种请求方法。HTTP1.0定义了三种请求方法：GET、POST和HEAD方法。
- HTTP1.1新增了六种请求方法：OPTIONS、PUT、PATCH、DELETE、TRACE和CONNECT方法。

方法	描述
GET	请求指定的页面信息，服务端将返回具体内容数据。
POST	提交数据，例如提交表单。
HEAD	类似于GET请求，但是返回的响应中没有具体的内容，用于获取报头。
PUT	更新和修改数据。
DELETE	请求删除指定的页面。
CONNECT	用于HTTP代理。
OPTIONS	允许客户端查看服务器的性能。
TRACE	回显服务器收到的请求，主要用于测试或诊断。
PATCH	用来对已知资源进行局部更新。

1.2 HTTP客户端请求报文及示例

- 请求头部允许客户端向服务器传递关于请求的附加信息。这些字段充当请求修饰符，其语义相当于编程语言方法调用中的参数。

请求头部示例：

Accept: text/html

Accept-Encoding: gzip, deflate, br

Connection: keep-alive

1.2 HTTP客户端请求报文及示例

- HTTP/1.0对每个连接都只能传送一个请求和响应，而从HTTP/1.1开始，在同一个连接中可以传送多个请求和响应，多个请求可以同时进行。
- www.authenticate是一种简单、有效的用户身份认证技术，在RFC HTTP/1.1标准中有详细解释。

请求头字段	含义
Accept	客户端可接受的MIME类型。
Accept-Encoding	客户端能够进行解码的数据编码方式，比如gzip。
Accept-Language	客户端所希望的语言种类，当服务器能够提供一种以上的语言版本时需要。
Authorization	授权信息，通常出现在对服务器发送的WWW-Authenticate头的应答中。
Content-Type	表示数据属于什么MIME类型。默认为text/plain纯文本格式。
Content-Length	表示请求消息正文的长度。
Host	客户端要访问的主机名。
Referer	客户端通过该字段表示自己从哪个资源来访问服务器的，该字段包含一个URL，表示从该URL代表的页面出发访问当前请求的页面。
User-Agent	该字段包含发出请求的用户信息，客户端类型。
Cookie	用于客户端保存服务器返回的数据，通常保存用户身份信息。
Connection	表示处理完该次请求后是否断开连接。

1.2 HTTP客户端请求报文及示例

MIME 的组成结构非常简单，由类型与子类型两个字符串中间用 / 分隔而组成，不允许有空格。type 表示可以被分多个子类的独立类别，subtype 表示细分后的每个类型。

类型	描述	典型示例
text	表明文件是普通文本，理论上是人类可读	text/plain, text/html, text/css, text/javascript
image	表明是某种图像。不包括视频，但是动态图（比如动态gif）也使用image类型	image/gif, image/png, image/jpeg, image/bmp, image/webp, image/x-icon, image/vnd.microsoft.icon
audio	表明是某种音频文件	audio/midi, audio/mpeg, audio/webm, audio/ogg, audio/wav
video	表明是某种视频文件	video/webm, video/ogg
application	表明是某种二进制数据	application/octet-stream, application/pkcs12, application/vnd.mspowerpoint, application/xhtml+xml, application/xml, application/pdf

1.2 HTTP客户端请求报文及示例

▪Cookie用于在客户端和服务器之间存储会话信息或跟踪用户状态，以便在用户访问同一网站时保持持久性和状态。Cookie通常包含了一些键值对的数据，以及一些关于Cookie的属性：
▪名称（Name）：Cookie的名称，用于唯一标识一个Cookie。
▪值（Value）：Cookie的值，存储在客户端的数据。
▪域（Domain）：指定了Cookie所属的域名。默认情况下，Cookie的域为创建它的服务器的域名，但也可以通过设置Domain属性来指定其他域名。
▪路径（Path）：指定了Cookie的可见路径。只有在指定路径下的页面才能访问到这个Cookie，
▪过期时间（Expires/Max-Age）：指定了Cookie的过期时间。过期时间可以是一个具体的日期时间，也可以是从当前时间开始的秒数。
▪安全标志（Secure）：指示浏览器仅在通过加密协议（如HTTPS）发送请求时才发送Cookie到服务器。这样可以确保Cookie在传输过程中不被窃取或篡改。
▪HttpOnly标志（HttpOnly）：指示浏览器禁止JavaScript访问Cookie，这样可以防止某些类型的跨站点脚本攻击

1.2 HTTP客户端请求报文及示例

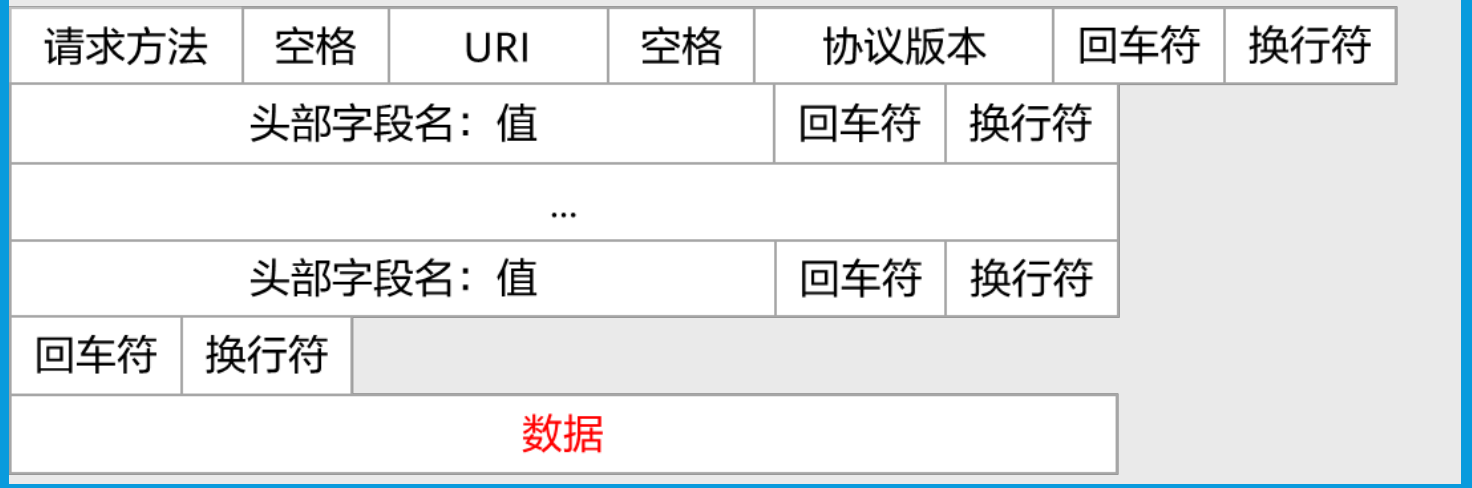
Content-Type是HTTP头部中的一个重要字段，用于指示HTTP消息（如请求体或响应体）中的实体内容的类型和字符集。其通常由一个主类型和一个子类型组成。以下是Content-Type头部的一些常见取值及其含义：text/plain、text/html、application/json、application/xml、image/jpeg、image/png、image/gif、application/octet-stream。

1.2 HTTP客户端请求报文及示例

- 空行：它的作用是通过一个空行，告诉服务器请求头部到此为止。
- 请求数据：若方法字段是GET，则此项为空，没有数据。若方法字段是POST，则通常来说此处放置的是要提交的数据。例如提交登陆数据：

请求数据示例：

user=admin&password=123456

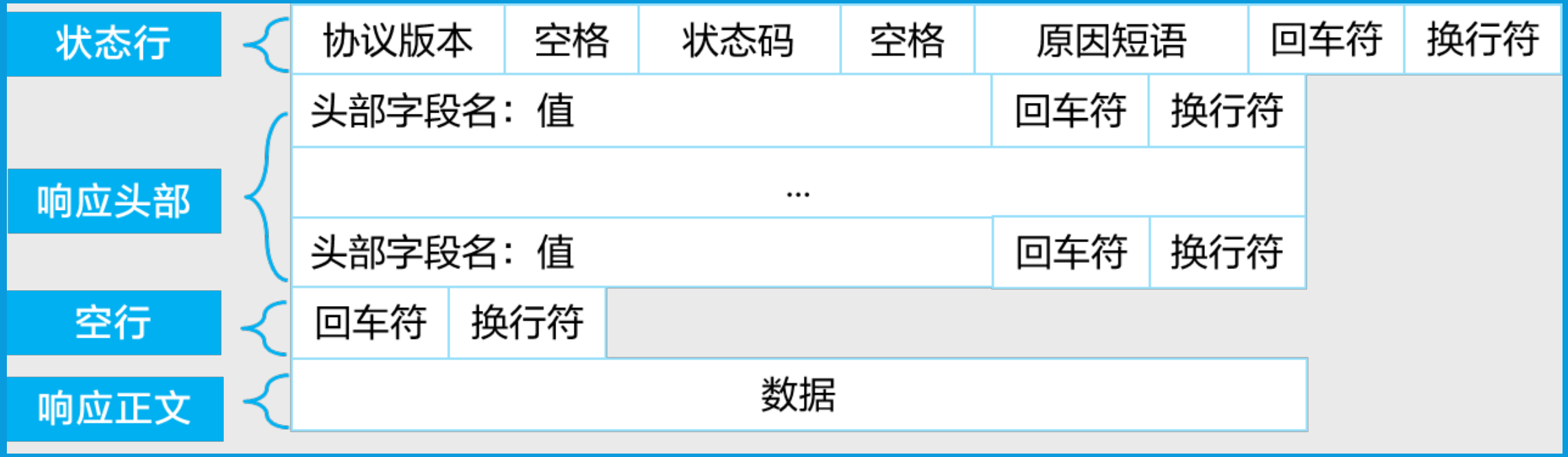


1.2 HTTP客户端请求报文及示例

- 客户端向服务器发送含有用户名密码的请求消息，进行登录认证。

1.3 HTTP服务器响应报文及示例

- 客户端向服务器发送含有用户名密码的请求消息，进行登录认证。



1.3 HTTP服务器响应报文及示例

- 响应消息的第一行是状态行，由协议版本、状态码和原因短语组成，每个元素由空格字符分隔。

① 协议版本：协议版本旨在允许发送方指示消息的格式及其理解后续HTTP通信的能力。

② 状态码：一个3位整数结果码，用于向客户端返回操作结果。

③ 原因短语：旨在对状态码进行简短的文本描述，帮助理解。

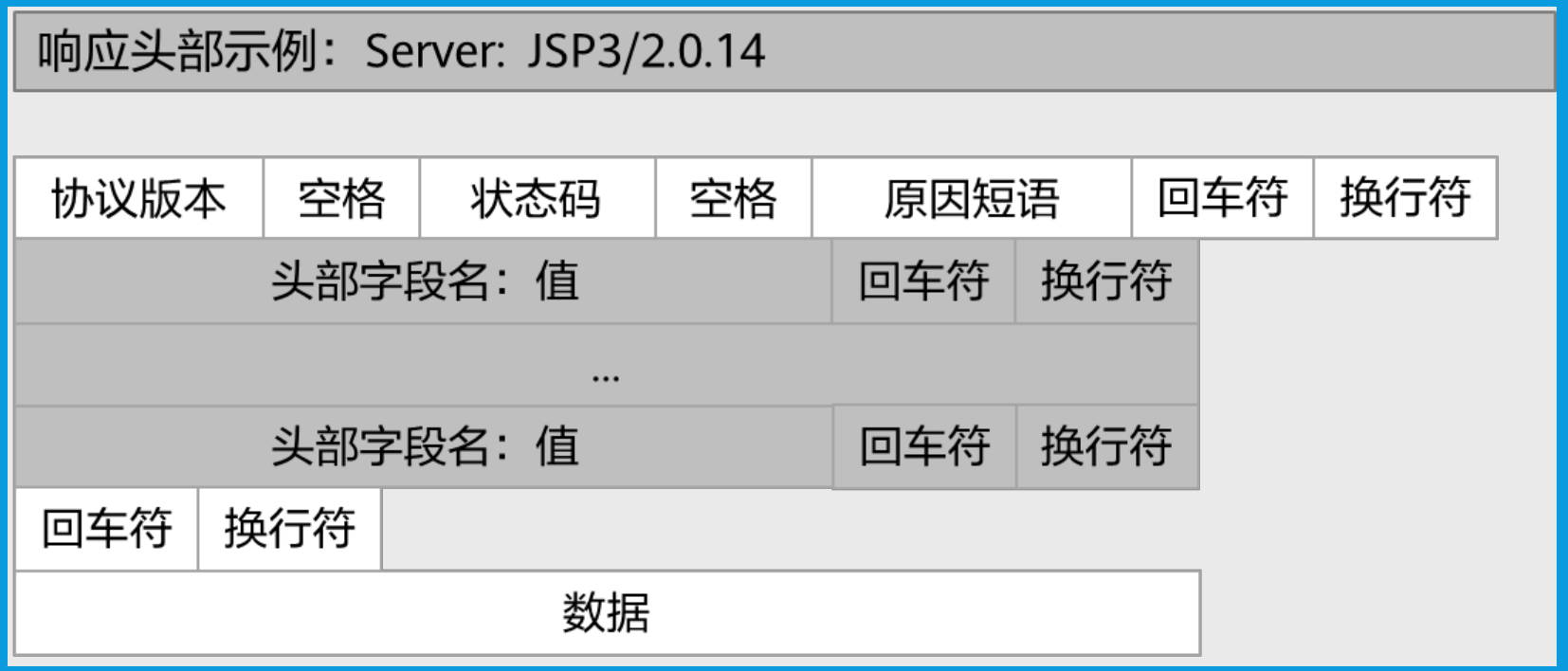
1.3 HTTP服务器响应报文及示例

- HTTP状态码（HTTP Status Code）是服务器响应状态的3位数字码，用于向客户端返回操作结果。

状态码		状态码说明	
1XX	(Informational)请求被接收	100 Continue	请求被接收，继续执行。
2XX	(Successful)请求成功	200 OK	成功，有响应消息体。
		201 Created	资源创建成功。
		204 No Content	成功，无响应消息体。
3XX	(Redirection)进一步操作需要被执行	301 Moved Permanently	目标资源被分配了新的URI，且未来的资源都会关联到新的URI。
4XX	(Client Error)请求错误	400 Bad Request	请求消息体错误，消息体中携带有错误描述。
		401 Unauthorized	授权失败，例如证书不匹配。
		403 Forbidden	禁止访问，可能的原因是：用户试图执行权限之外的操作或登录用户名密码错误。
		404 Not Found	找不到请求的资源。
5XX	(Server Error)服务端错误	500 Internal Server Error	服务器内部错误，不能执行此请求，用户需要稍后重新发送请求。
		501 Not Implemented	功能未实现。

1.3 HTTP服务器响应报文及示例

- 响应头部允许服务器传递关于响应的附加信息，这些头部字段提供了关于服务器的相关信息以及URI所标识资源的信息。



1.3 HTTP服务器响应报文及示例

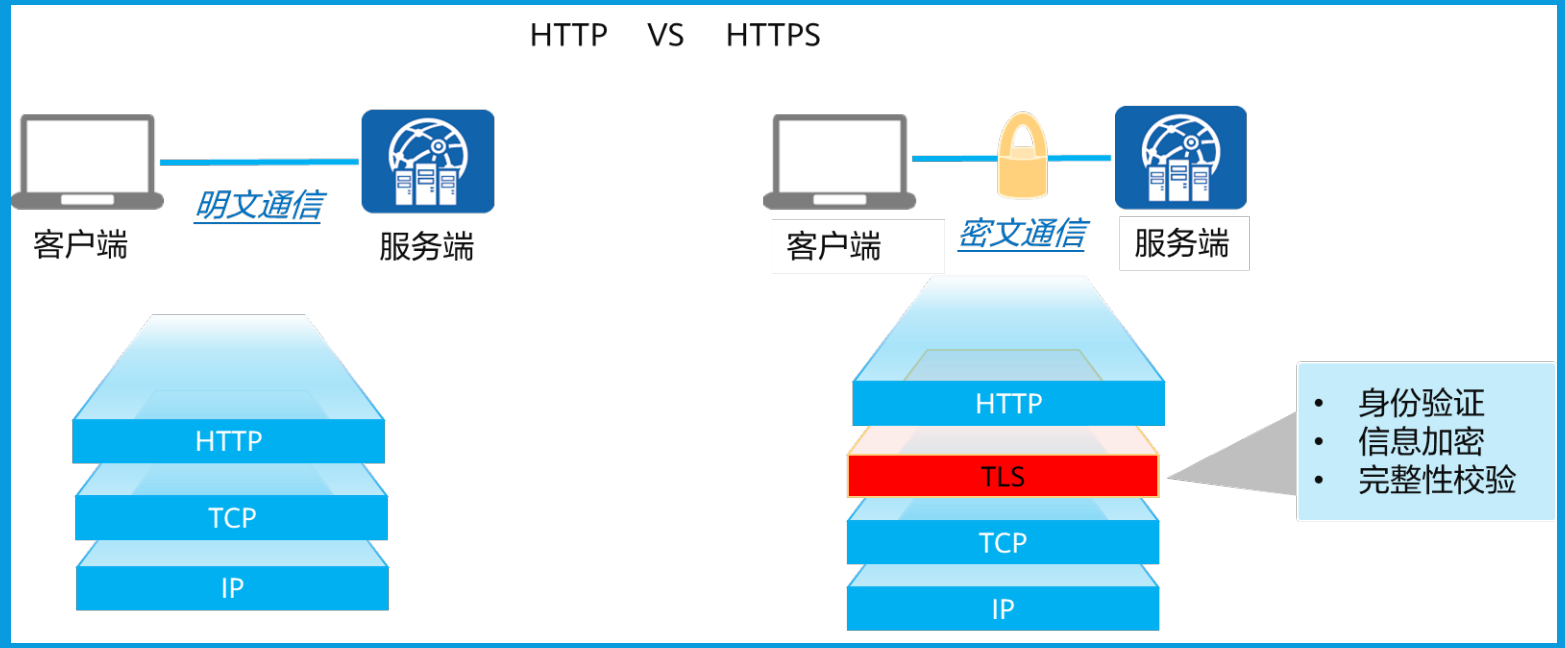
响应头字段	含义
Allow	服务器支持哪些请求方法(如GET、POST等)。
Content-Encoding	文档的编码(Encode)方法。只有在解码之后才可以得到Content-Type头指定的内容类型。
Content-Length	表示内容长度。只有当客户端使用持久HTTP连接时才需要该字段。
Content-Type	表示数据属于什么MIME类型。默认为text/plain纯文本格式。
Date	当前的GMT时间。
Location	该字段配合302状态码使用，用于重定向到一个新URI地址。表示客户端应当去寻找并提取资源的位置。
Server	表示服务器的类型。
Set-Cookie	设置和页面关联的Cookie。
Transfer-Encoding	数据的传送格式。
WWW-Authenticate	表示客户应该在Authorization头中提供什么类型的授权信息，在包含401(Unauthorized)状态行的应答中这个字段是必需的。

1.3 HTTP服务器响应报文及示例

- 服务器向客户端返回响应消息，认证成功。

拓展知识-HTTPS

- HTTP协议采用明文传输信息，存在信息窃听、信息篡改和信息劫持的风险。HTTPS具有身份验证、信息加密和完整性校验的功能，可以避免此类问题发生。
- HTTPS (Hyper Text Transfer Protocol over SecureSocket Layer，安全超文本传输协议），是以安全为目标的HTTP通道。HTTPS在HTTP的基础下加入SSL/TLS层，是使用SSL/TLS加密的HTTP协议。



实训8-（一）：HTTP协议基础及报文格式

【任务目标】掌握HTTP协议基本概念及报文操作方法

（一）HTTP 协议基础及报文格式（请按要求填写命令，粘贴结果图）

1、HTTP 代表什么？

A. HyperText Transfer Protocol

B. Hypertext Terminal Protocol

C. High Transfer Protocol

D. Hyper Transfer Protocol

正确答案：A

2、HTTP 工作在哪一层？

A. 应用层

B. 传输层

2. RESTCONF 协议

- RESTCONF 协议概述
- 描述性状态转移（REST）概述
- RESTCONF网络基本架构
- RESTCONF与NETCONF的比较
- RESTCONF协议报文格式
- 华为设备开启RESTCONF服务

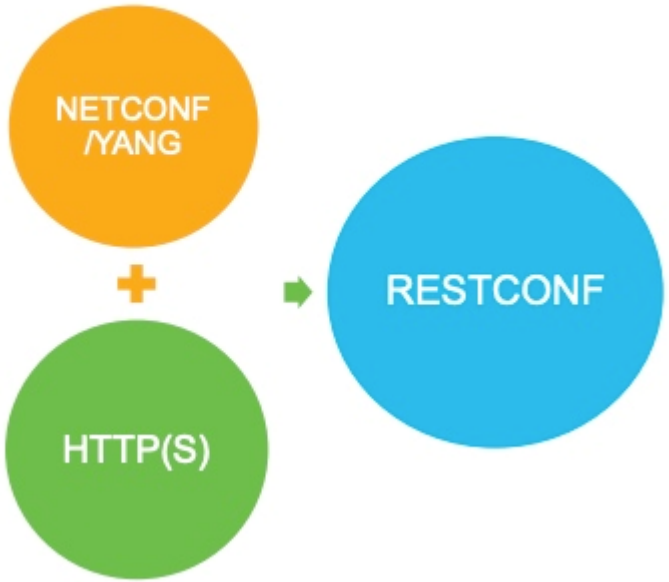
2.1 RESTCONF 协议概述

- RESTCONF是在融合NETCONF和HTTP协议的基础上发展而来的。
- 广泛用于云计算平台、物联网及企业网络运维等。
- RESTful接口风靡全球，被越来越多公司接受。

RESTCONF

What is it?

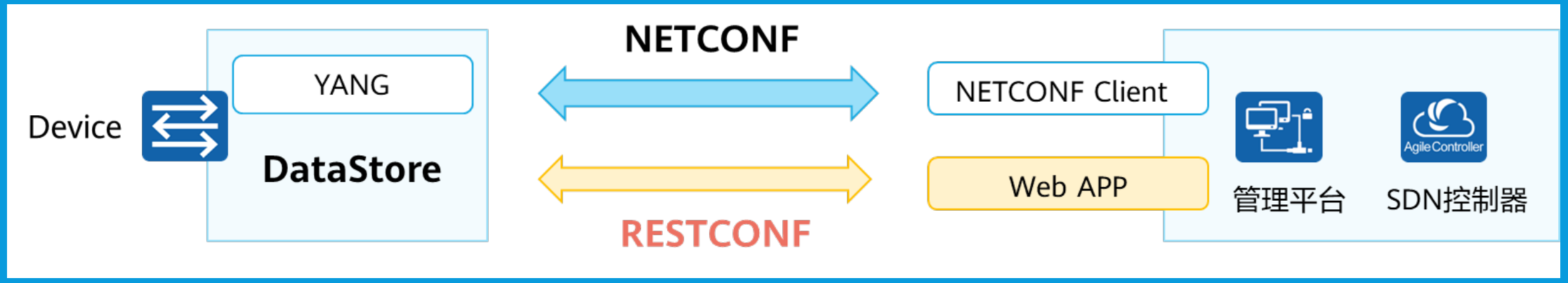
- It's an implementation of a REST API
- Model-driven API
- Functional sub-set of NETCONF
- Exposes YANG models via a REST API (URL)
- Uses HTTP(S) as transport
- Uses XML or JSON for encoding
- Developed to use HTTP tools and programming libraries



2.1 RESTCONF 协议概述

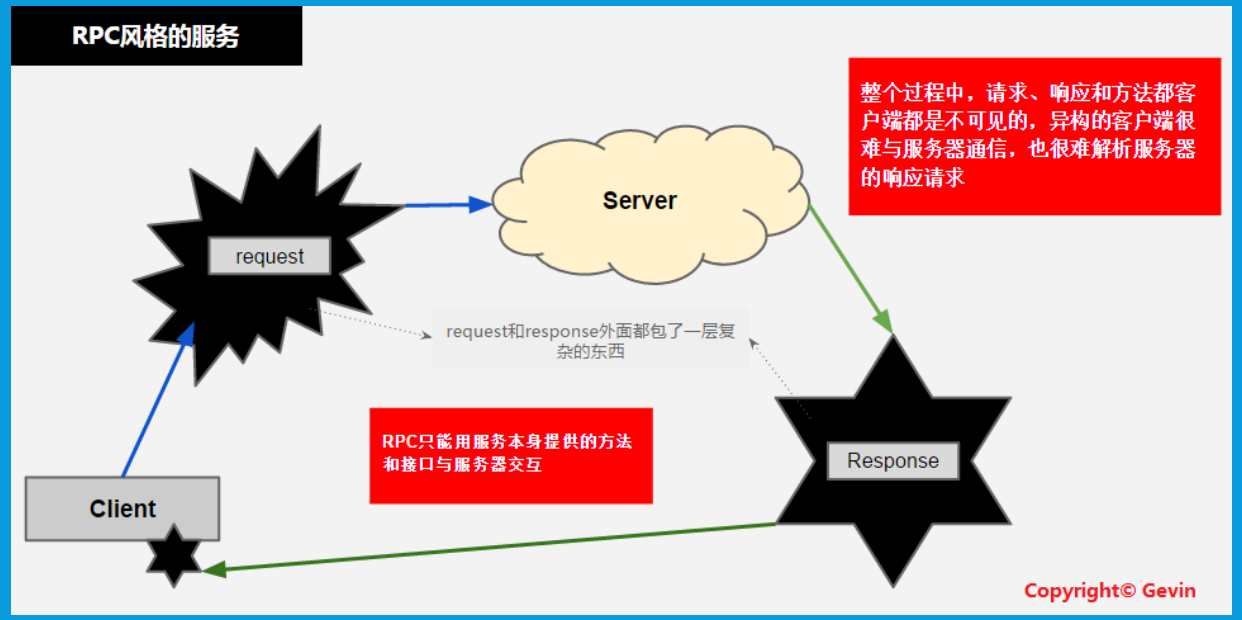
- RESTCONF允许Web应用以一种模块化、可扩展的方式访问网络设备的配置数据、状态数据和事件通知。而NETCONG使用的NETCONF Client。

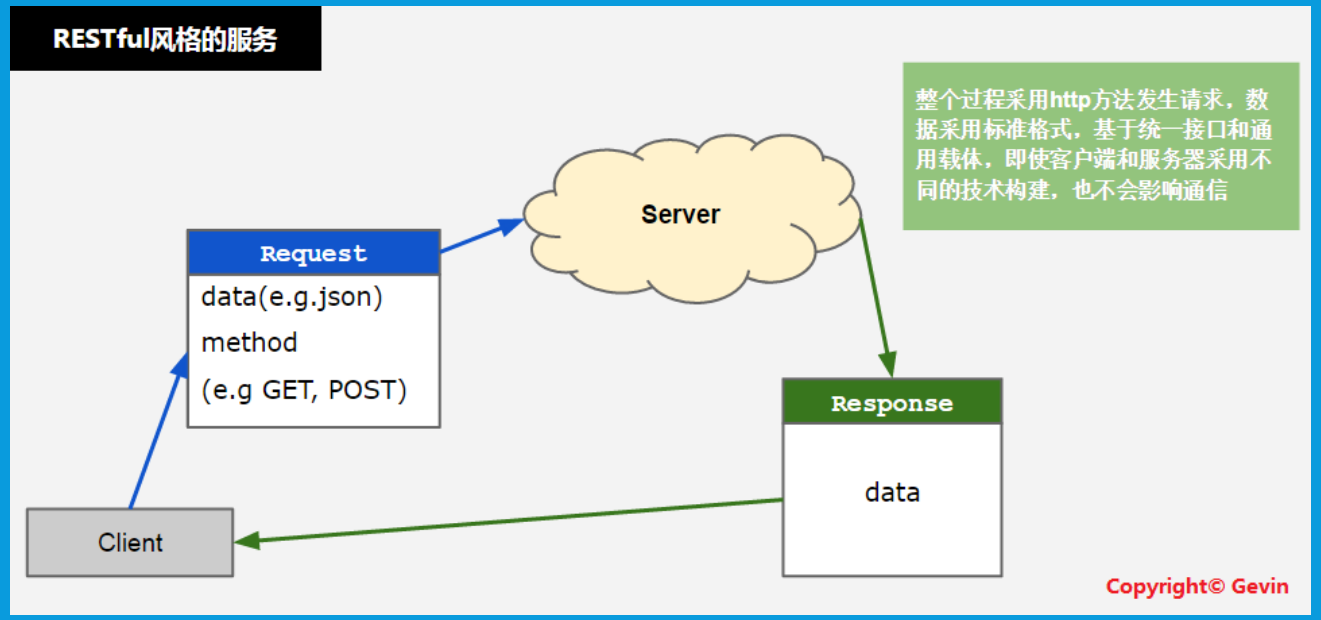
  - RESTCONF使用HTTP的方法对设备YANG定义的数据进行操作（增删改查）。
  - 设备NETCONF和RESTCONF可以共享的YANG文件。
  - 数据编码格式支持XML或者JSON 。



2.2描述性状态转移（REST）概述

- REST (Representational State Transfer) API 是一种设计分布式系统和Web服务的架构风格，它遵循一组特定的原则和约束，以确保简洁、一致和可伸缩的交互。REST具有资源（URI）、表现层和状态转化等三个概念。任何满足REST约束条件和原则的架构，成为RESTful架构。



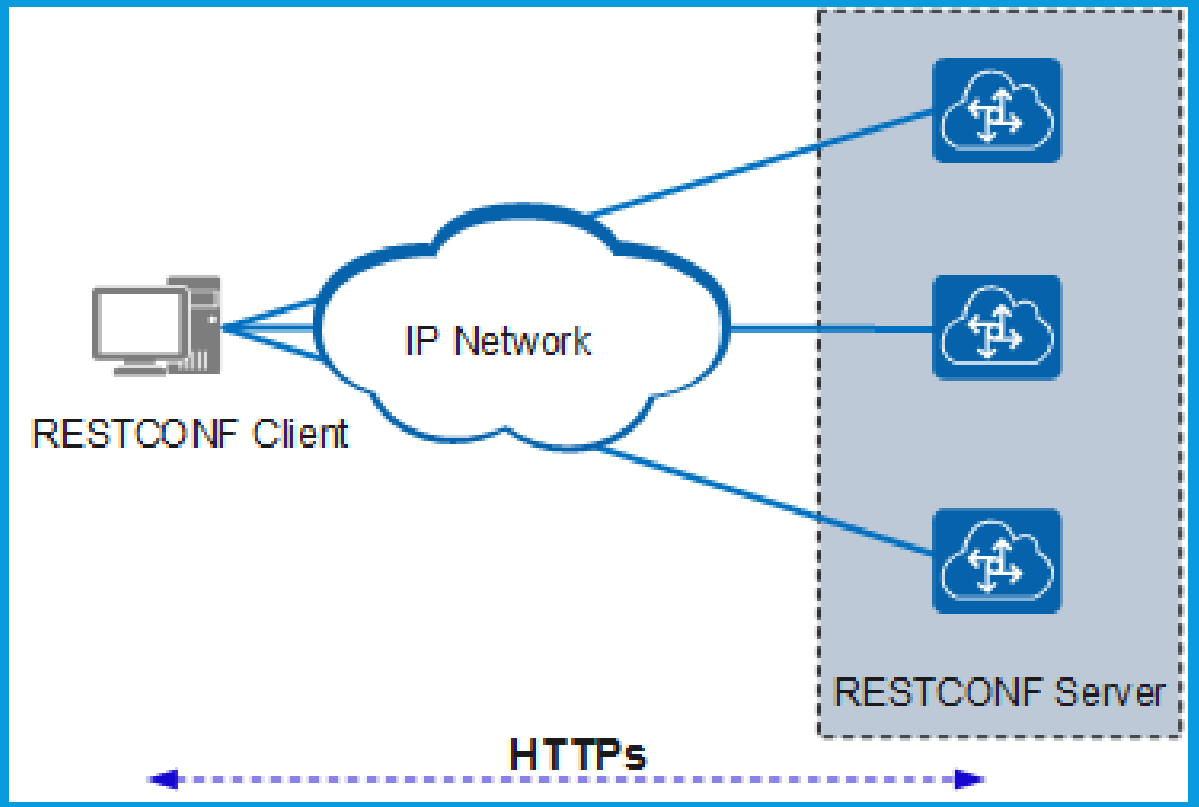


2.3 RESTCONF网络基本架构

- RESTCONF基本网络架构如下图所示。RESTCONF基本网络架构中主要元素：

- RESTCONF Client：
  - 客户端利用RESTCONF协议对网络设备进行系统管理。客户端向服务器发送请求，可以实现创建、删除、修改或查询一个或多个数据。

- RESTCONF Server：
  - 设备作为服务器端，服务器用于维护被管理设备的信息数据并响应客户端的请求，把数据返回给发送请求的客户端。服务器收到客户端的请求后会进行解析并处理请求，然后给客户端



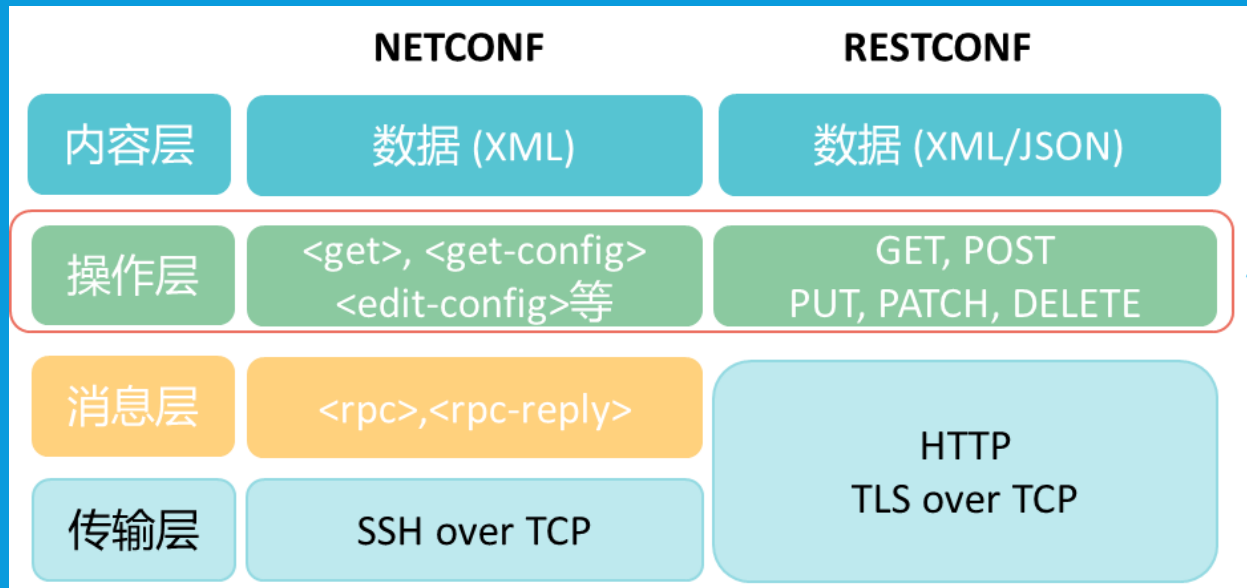
2.4 RESTCONF与NETCONF的比较

- RESTCONF较于NETCONF，使用了不同的操作方法和数据编码。

RESTCONF与NETCONF比较		
比较项目	NETCONF+YANG	RESTCONF+YANG
传输通道（协议）	NETCONF传输层首选推荐SSH（Secure Shell）协议，XML信息通过SSH协议承载。	RESTCONF是基于HTTP协议访问设备资源。RESTCONF提供的编程接口符合IT业界流行的RESTful风格。
报文格式	采用XML编码。	采用XML或JSON编码。
操作特点	NETCONF的操作复杂，例如： ● NETCONF支持增、删、改、查，支持多个配置数据库，也支持回滚等。 ● NETCONF需要两阶段提交（即先提交参数，再commit参数）。	RESTCONF的操作简单，例如： ● RESTCONF支持增、删、改、查操作，仅支持<running/>配置数据库。 ● RESTCONF操作方法无需两阶段提交，操作直接生效。

2.4 RESTCONF与NETCONF的比较

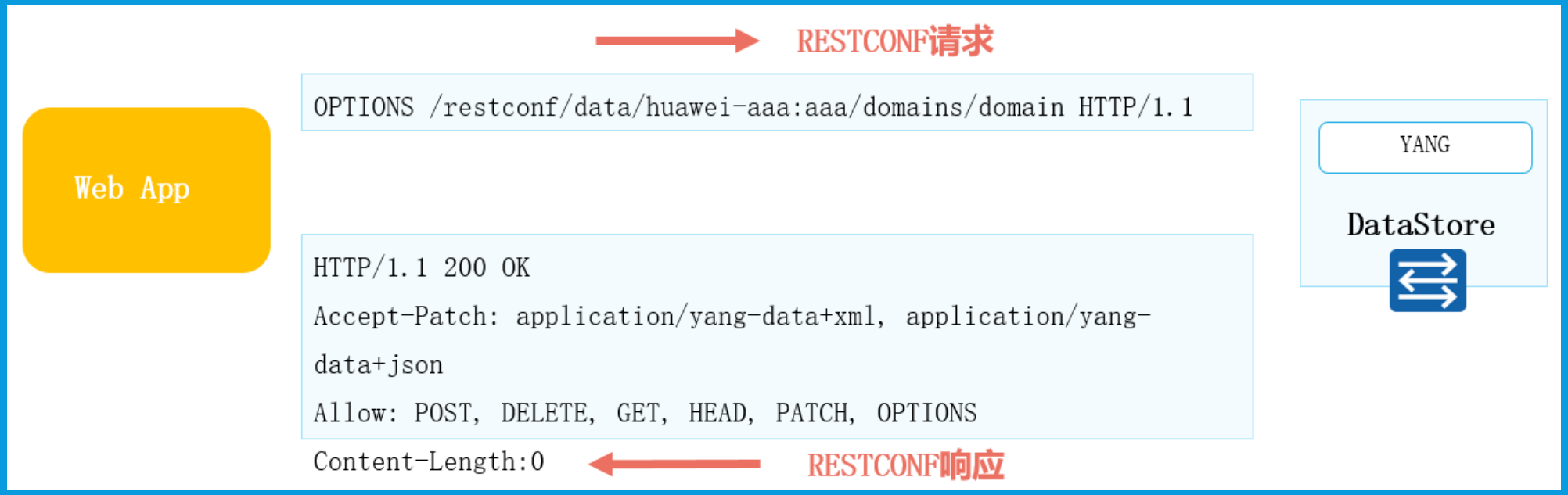
- 对于RESTCONF较于NETCONF定义了配置数据库和增、删、改、查操作，这些操作可以用来访问配置数据库。NETCONF使用YANG语言定义了数据库内容、配置数据、状态数据、RPC操作等的语法语意。
- RESTCONF协议通过HTTP方法可以识别NETCONF中定义的增删改查操作，用于访问YANG定义的数据。



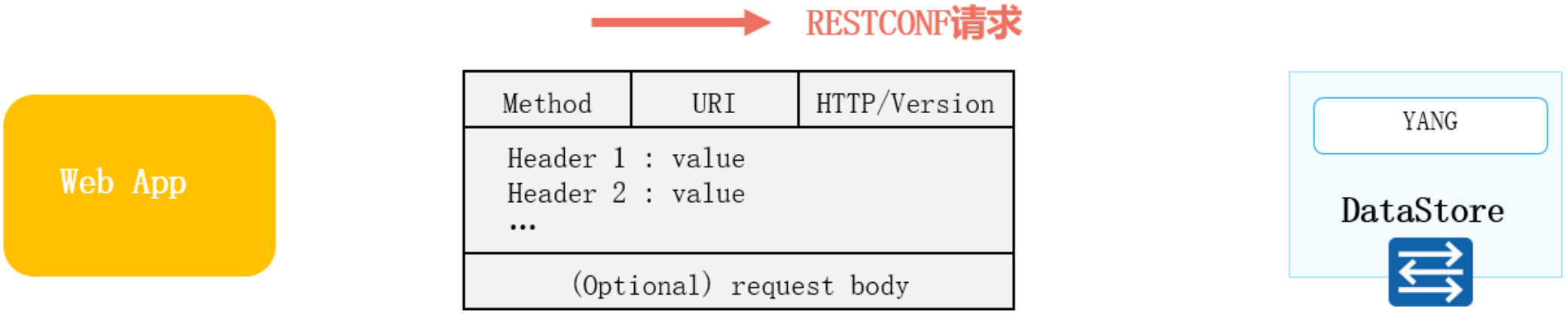
RESTCONF	对应的NETCONF操作
GET	<get-config>, <get>
POST	<edit-config> (nc:operation="create")
	invoke an RPC operation
PUT	<edit-config> (nc:operation="create/replace")
	<copy-config> (PUT on datastore)
PATCH	<edit-config> (nc:operation depends on PATCH content)
DELETE	<edit-config> (nc:operation="delete")

2.5 RESTCONF协议报文格式

- 一次完整的RESTCONF交互包含请求和响应。
- 本例客户端通过OPTIONS方法获取设备支持的操作方法。
- 设备回复支持操作有：POST, DELETE, GET, HEAD, PATCH, OPTIONS



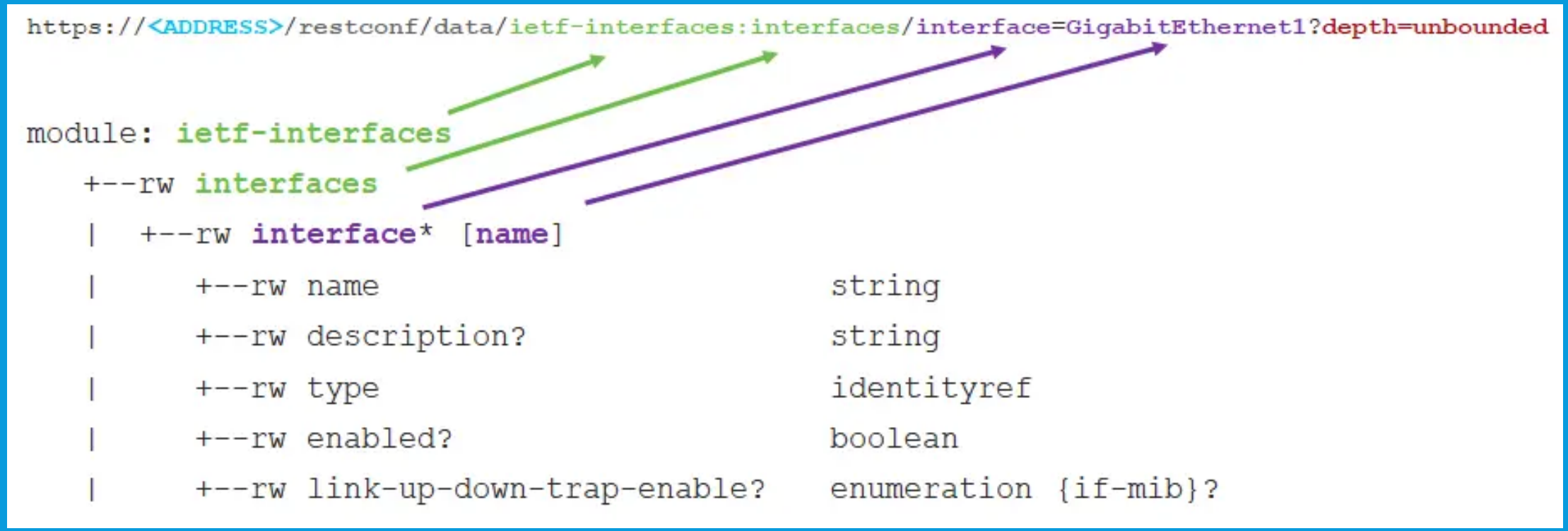
2.5 RESTCONF协议报文格式-请求报文



字段说明	
Method	HTTP操作方法，作用于URI中指定的目标资源。
URI	统一资源标识
HTTP/Version	HTTP协议版本。
Header : value	请求报文的头部，有特定字段要求。格式为header字段和值。
Request body	（可选）请求消息体。有些方法不携带body信息

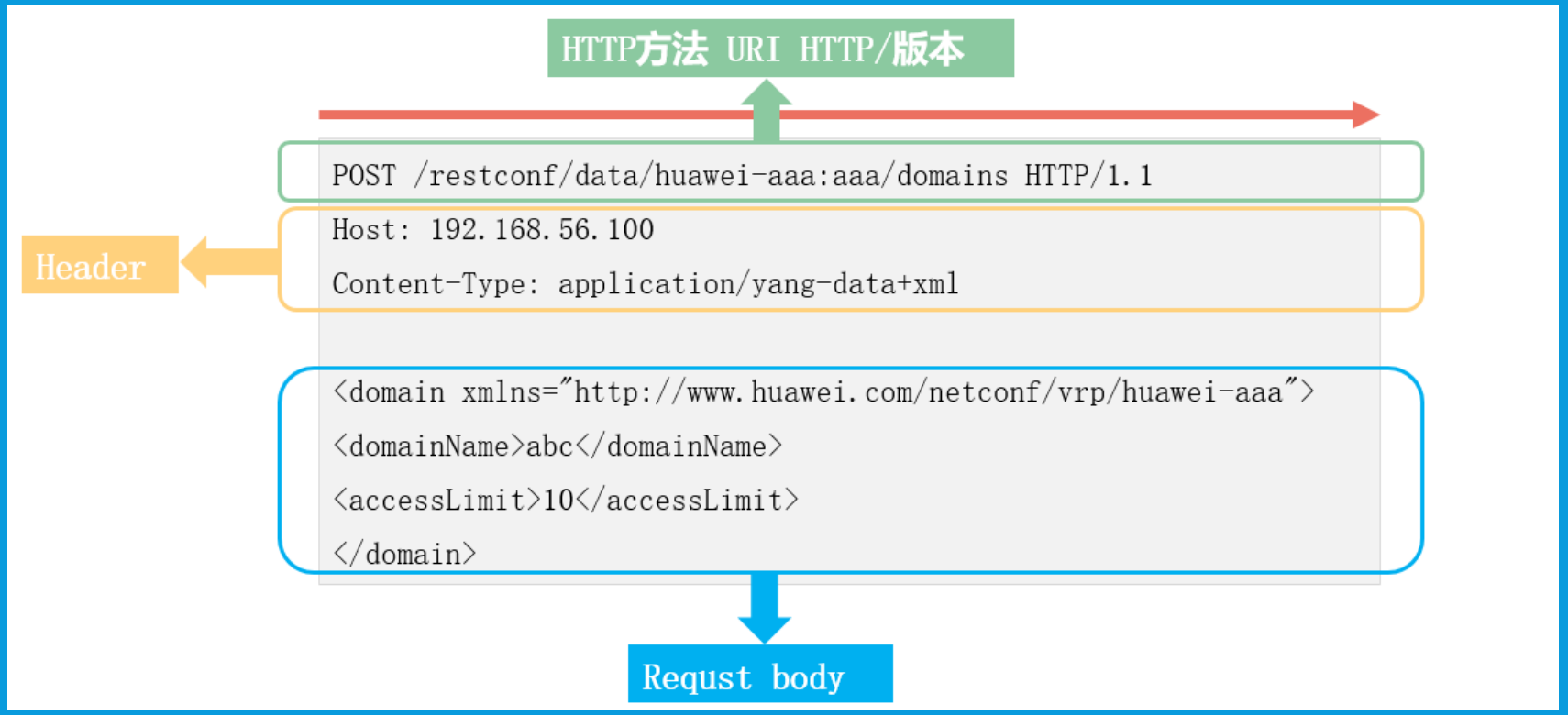
2.5 RESTCONF协议报文格式-URL

- 基于HTTP协议的URI格式为：
- https://<ADDRESS>/<ROOT>/data/<[YANG MODULE:]CONTAINER>/<LEAF>[?<OPTIONS>]

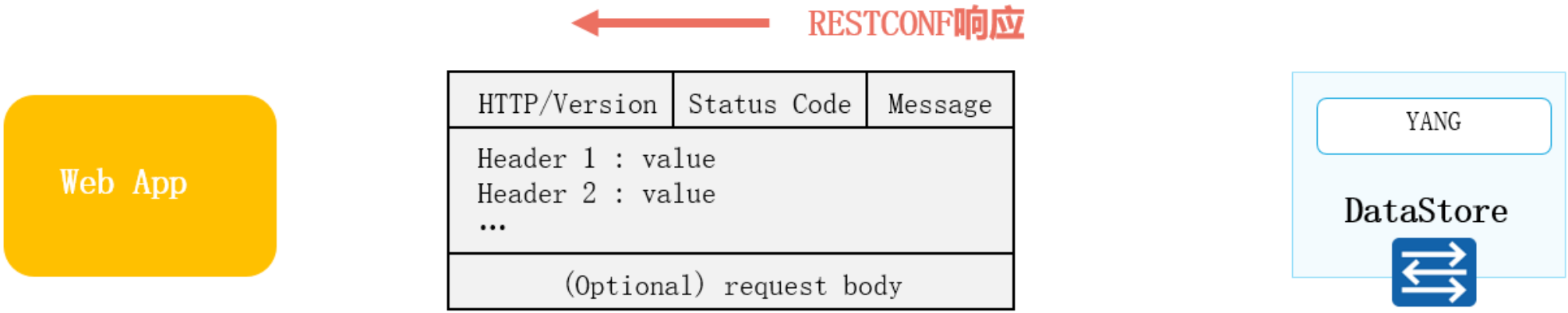


2.5 RESTCONF协议报文格式

- RESTCONF请求对象地址为192.168.56.100。修改数据配置数据domainName为abc，修改accessLimit值为10。



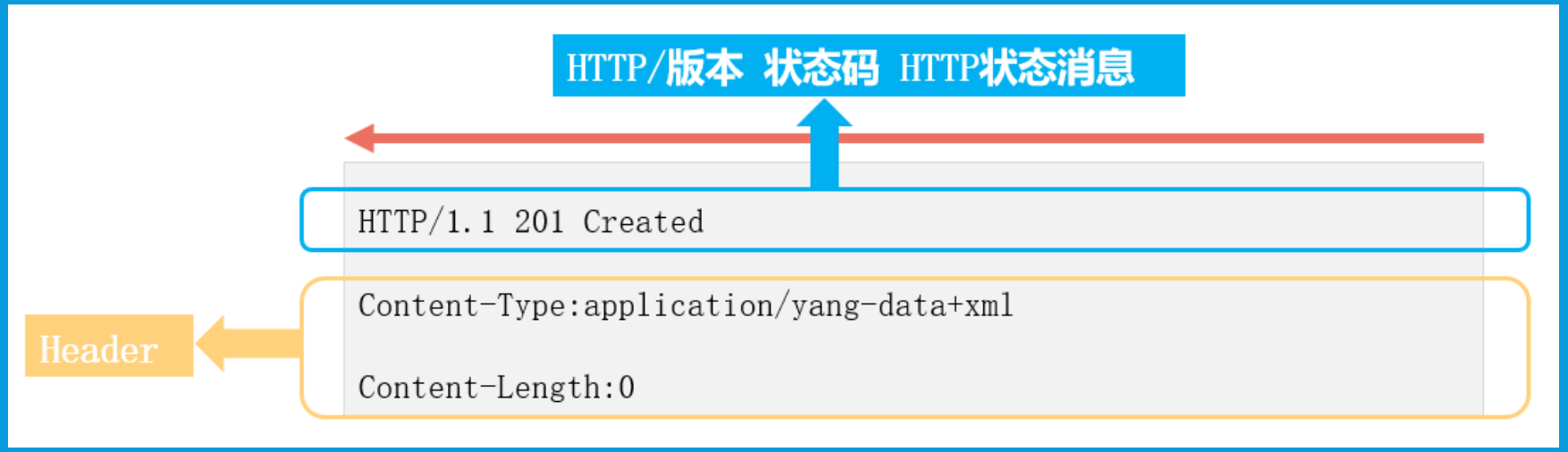
2.5 RESTCONF协议报文格式-响应报文



字段说明	
HTTP/Version	HTTP操作方法，作用于URI中指定的目标资源。
Status code	HTTP状态码
Message	HTTP状态消息。
Header : value	响应报文的头部，有特定字段要求。格式为header字段和值。
Request body	（可选）响应消息体。有些方法不携带body信息

2.5 RESTCONF协议报文格式

- RESTCONF响应报文：
  - ✓ 返回状态码201，表示资源创建成功。
  - ✓ 头部信息Content-Type和Content-Length，描述Body信息。Body数据类型为XML，内容长度为0。



2.6华为设备开启RESTCONF服务

- 执行命令system-view，进入系统视图。
- 执行命令aaa，进入AAA视图。
- 执行命令local-user user-name password [cipher password | irreversible-cipher irreversible-cipher-password]，配置本地用户名和密码。
- 执行命令local-user user-name service-type service-type，配置本地用户的接入类型。
- 执行命令local-user user-name level level，配置本地用户级别。

```
return
<Huawei>sys
Enter system view, return user view with return command.
[~Huawei]aaa
[~Huawei-aaa]local-user python password cipher Huawei@123
Error: Failed to set the password for administrator python, because it
used among the last 10 used passwords.
[~Huawei-aaa]
[~Huawei-aaa]local-user python service-type http
Warning: The service type configured contains unsecured protocol(http)
commended to configure the secure service type only.
[*Huawei-aaa]
[*Huawei-aaa]commit
[~Huawei-aaa]
[~Huawei-aaa]quit
[~Huawei]
```

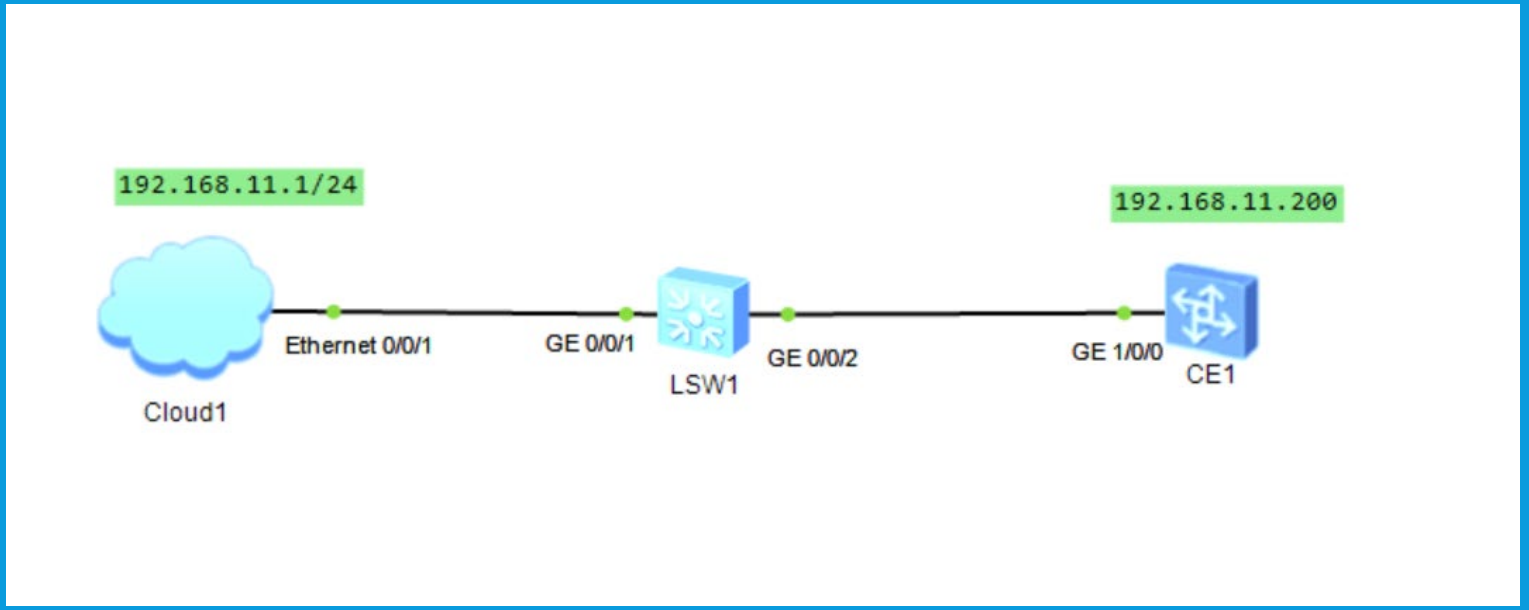
2.6华为设备开启RESTCONF服务

- 执行命令quit，退出AAA视图。
- 执行命令http，进入HTTP视图。
- 执行命令service restconf，创建并进入Service-Restconf视图。
- 根据需要选择HTTP是安全服务还是普通服务，本文仅介绍HTTP普通服务。
- 如果使能普通HTTP服务，执行以下步骤：
- 执行命令server enable，使能HTTP侦听服务。
- （可选）执行命令server port port-number，配置HTTP服务侦听端口号。缺省情况下，HTTP服务侦听端口号是80。
- 执行命令commit，提交配置。

```
[~Huawei]http
[~Huawei-http]
[~Huawei-http]service restconf
[~Huawei-http-service-restconf]
[~Huawei-http-service-restconf]server enable
[~Huawei-http-service-restconf]
[~Huawei-http-service-restconf]server port 8080
[~Huawei-http-service-restconf]
[~Huawei-http-service-restconf]commit
[~Huawei-http-service-restconf]
[~Huawei-http-service-restconf]quit
[~Huawei-http]quit
[~Huawei]
```

实训9-（二）：配置交换机CE12800开启RESTCONF服务

【任务目标】根据指导教师给出网络拓扑图，配置CE12800设备开启RESTCONF服务，选择HTTP普通服务。其中用户名为Python，密码为Huawei@123，HTTP端口为8080。



3. Request模 块

- Request模块安装
- Request模块常用方法

3.1 Request模块安装

- Requests是一个常用的HTTP请求模块，其宗旨是让HTTP服务于人类。由于它是第三方库，因此需要进行安装：

  pip install requests

- 安装后，可以通过以下命令验证：

  python

  import requests as rq

  rq.__name__

```
(ensp_py1) C:\Users\USSTz>pip install requests
Collecting requests
  Using cached requests-2.32.3-py3-none-any.whl.metadata (4.6 kB)
Collecting charset-normalizer<4,>=2 (from requests)
  Using cached charset_normalizer-3.3.2-cp310-cp310-win_amd64.whl.m
Collecting idna<4,>=2.5 (from requests)
  Downloading idna-3.10-py3-none-any.whl.metadata (10 kB)
Collecting urllib3<3,>=1.21.1 (from requests)
  Downloading urllib3-2.2.3-py3-none-any.whl.metadata (6.5 kB)
Collecting certifi>=2017.4.17 (from requests)
  Downloading certifi-2024.8.30-py3-none-any.whl.metadata (2.2 kB)
Using cached requests-2.32.3-py3-none-any.whl (64 kB)
Downloading certifi-2024.8.30-py3-none-any.whl (167 kB)
Using cached charset_normalizer-3.3.2-cp310-cp310-win_amd64.whl (10
Downloading idna-3.10-py3-none-any.whl (70 kB)
Downloading urllib3-2.2.3-py3-none-any.whl (126 kB)
Installing collected packages: urllib3, idna, charset-normalizer, c
Successfully installed certifi-2024.8.30 charset-normalizer-3.3.2 i
b3-2.2.3
```

```
(ensp_py1) C:\Users\USSTz>python
Python 3.10.14 | packaged by Anaconda, Inc. | (main, May  6 2
MD64)] on win32
Type "help", "copyright", "credits" or "license" for more inf
>>> import requests as rq
>>> rq.__name__
'requests'
>>>
```

3.2 Request模块常用方法

方法	描述
delete(*url*, *args*)	发送 DELETE 请求到指定 url
get(*url*, *params*, *args*)	发送 GET 请求到指定 url
head(*url*, *args*)	发送 HEAD 请求到指定 url
patch(*url*, *data*, *args*)	发送 PATCH 请求到指定 url
post(*url*, *data*, *json*, *args*)	发送 POST 请求到指定 url
put(*url*, *data*, *args*)	发送 PUT 请求到指定 url
request(*method*, *url*, *args*)	向指定的 url 发送指定的请求方法

■表中常用参数说明：

1、method表示请求方法，包括get、post、delete等。

2、url表示请求URL，根据接口文档的标注获取。

3、params指代查询字符串参数，可以是python字典、列表等对象。

4、data是发送给指定URL的字典、列表等对象。

5、JSON是发送给URL的JSON对象。

方法	说明
requests.request()	构造一个请求，支撑以下各方法的基础方法
requests.get()	获取HTML网页的主要方法，对应于HTTP的GET
requests.head()	获取HTML网页头信息的方法，对应于HTTP的HEAD
requests.post()	向HTML网页提交POST请求的方法，对应于HTTP的POST
requests.put()	向HTML网页提交PUT请求的方法，对应于HTTP的PUT
requests.patch()	向HTML网页提交局部修改请求，对应于HTTP的PATCH
requests.delete()	向HTML页面提交删除请求，对应于HTTP的DELETE

```python
import requests
from requests.auth import HTTPBasicAuth
import json

ret = requests.get("http://www.baidu.com")
print(ret.status_code)
print(ret.text)
print(ret.encoding)
print(ret.apparent_encoding)
print(ret.content)
```

Run: get_test

```
C:\Users\USSTz\anaconda3\envs\ensp_env\python.exe C:\Users\USSTz\Desktop
200
<!DOCTYPE html>
<!--STATUS OK--><html> <head><meta http-equiv=conten

ISO-8859-1
utf-8
b'<!DOCTYPE html>\r\n<!--STATUS OK--><html> <head><m
```

Response对象的属性

属性	说明
r.status_code	HTTP请求的返回状态，200表示连接成功，404表示失败
r.text	HTTP响应内容的字符串形式，即，url对应的页面内容
r.encoding	从HTTP header中猜测的响应内容编码方式
r.apparent_encoding	从内容中分析出的响应内容编码方式（备选编码方式）
r.content	HTTP响应内容的二进制形式

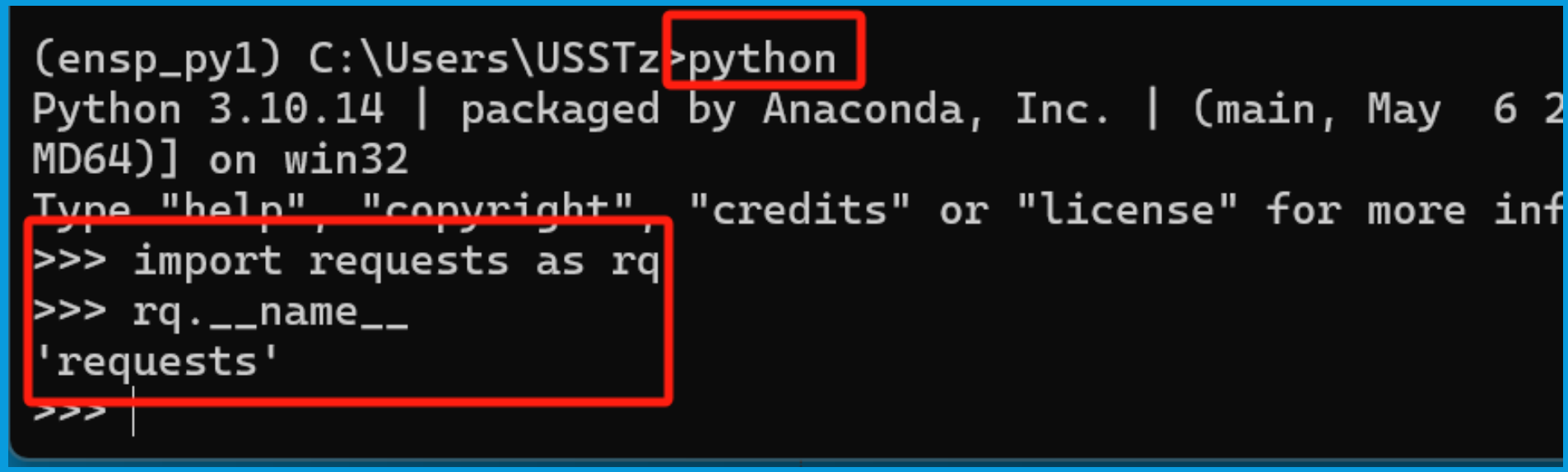
从http中猜测出来的编码方式

区别

从url上获取一个图片，可以通过r.content获取这个图片

实训9-（三）：安装requests并验证

【任务目标】通过Anaconda Prompt在虚拟环境ensp_py下安装requests包，并验证安装是否正确



实训9-（四）：使用requests模块登入CE12800设备

【任务目标】参考实验指导说明书，基于指导教师给的网络拓扑图，配置CE12800开启RESTCONF服务，并通过requests模块登陆该设备。需要完成的任务如下。

（1）在网络拓扑图中配置CE12800设备的RESTCONF服务及本地账户,用户名：python,密码：Huawei@123。

（2）通过浏览器输入URL登陆CE12800设备。

（3）编写Python脚本实现自动登入CE12800设备。

（4）运行Python脚本。