图解HTTP

He Yanhong

Published
with GitBook

目錄

《图解HTTP》读书笔记

了解WEB及网络基础

HTTP基础

- 1989年3月，HTTP诞生；
- HTTP属于TCP/IP的一个子集。

网络基础TCP/IP

TCP/IP是在IP协议的通信过程中使用到的协议族的统称。

应用层

决定了向用户提供应用服务时通信的活动[http];

传输层

提供处于网络连接中的两台计算机之间的数据传输[TCP/UDP];

网络层

用来处理在网络上流动的数据包[IP];

链路层

用来处理连接网络的硬件部分,包括控制操作系统、硬件的设备驱、网卡及光纤等物理可见部分。

第二章 简单的HTTP协议

HTTP基础

- 两台计算机在使用HTTP通信在一条线路上的必须是一端为客户端，一端为服务器；
- HTTP协议规定请求从客户端发出，最后服务器端响应该请求并返回；
- HTTP是不保存状态，即无状态协议，于是为了实现保持状态功能引入了Cookie技术；

HTTP方法

GET：获取资源

GET方法用来请求访问已被URI识别的资源，制定的资源经服务器端解析后返回响应内容。

POST：传输实体主体

POST方法用来传输实体的主体，虽然GET方法也可以但一般不用GET方法传输实体的主体。

PUT：传输文件

PUT方法用来传输文件。就像FTP协议的文件上传一样，要求在请求报文的主体中包含文件的内容，然后保存到请求URI指定的位置。

HEAD：获得报文首部

HEAD方法和GET方法一样，只是不返回报文主体部分，用于确认URI的有效性及资源更新的日期时间。

DELETE：删除文件

DELETE方法用来删除文件，是与PUT方法相反的方法。

OPTION：询问支持的方法

OPTION方法用来查询针对请求URI指定的资源支持的方法。

TRACE：追踪路径

TRACE方法是让WEB服务器端将之前的请求通信环回给客户端的方法。

CONNECT：要求用隧道协议连接代理

CONNECT方法要求在代理服务器通信时建立隧道，实现隧道协议进行TCP通信。

持久连接

问题：

初始的HTTP协议版本中，每进行一次HTTP通信就要断开一次TCP连接，随着HTTP的普及，文档中包含大量图片的情况多了起来以后，每次请求都会造成无谓的TCP连接建立和断开，增加通信量的开销。

解决：

使用持久连接的方法。特点是：只要任意一端没有明确的提出断开连接，则保持TCP的连接状态。

优点：

减少了TCP连接的重复建立和断开造成的额外开销，减轻了服务器的负载；另外，减少了开销的那部分时间，使HTTP请求和响应不能够更早的结束，这样WEB页面的显示速度也就相应的提高了。

使用**Cookie**的状态管理

Cookie会根据从服务器端发送的响应报文内的一个叫做Set—Cookie的首部字段信息，通知保存Cookie。当下次客户端再往该服务器发送请求时，客户端会自动在请求报文中加入Cookie值后发送出去。

服务端发现客户端发送过来的Cookie后，会去检查究竟是从哪一个客户端发来的连接请求，然后对比服务器上的记录，最后得到之前的状态信息。

第三章 HTTP报文内的HTTP信息

HTTP通信过程包括从客户端发往服务器端的请求及从服务器端返回客户端的响应。

用于HTTP协议交互的信息被称为HTTP报文。请求端（客户端）的HTTP报文叫请求报文，响应端（服务器端）的叫做响应报文。

HTTP报文大致可分为报文首部和报文主体两块，两者由最初出现的空行来划分，通常并不一定要有报文主体。

编码提升传输速率

HTTP在传输数据时可以按照数据原貌直接传输，但也可以在传输过程中通过编码提升传输速率。在传输时编码，能有效地处理大量的访问请求。

报文主体和实体主体：

报文：HTTP通信中的基本单位； 实体：作为请求或响应的有效载荷数据被传输，其内容由实体首部和实体主体组成。

HTTP报文的主体用于传输请求或者响应实体的主体。通常报文主体就等于实体主体。只有当传输中进行编码操作时，实体主体的内容发生变化，才导致它和报文主体产生差异。

压缩传输的内容编码

内容编码指明应用在石头内容上的编码格式，并保持实体信息原样压缩，内容编码后的实体由客户端接收并负责解码。

常用的内容编码有以下几种：

- gzip（GUN zip）
- compress（UNIX系统的标准压缩）
- deflate（zlib）
- identify（不进行编码）

分割发送的分块传输编码

请求的编码实体资源尚未全部传输完成之前，浏览器无法显示请求页面。在传输大容量数据时，通过吧数据分割成多块，能够让浏览器逐步显示页面。这种吧实体主体分块的功能称为分块传输编码

发送多种数据的多部分对象集合

获取部分内容的范围请求

内容协商返回最合适的内容

内容协商制是指客户端和服务器端就响应的资源进行交涉，然后提供给客户最为合适的资源，内容协商会以响应资源的语言、字符集、编码方式等作为判断 基准。

内容协商技术的三种类型：

- 服务器驱动协商
- 客户端驱动协商
- 透明协商（上面两种的结合）

第四章 返回结果的HTTP状态码

HTTP状态码负责表示客户端HTTP请求的返回结果、标记服务器端的处理是否正常、通知出现的错误等。

状态码类型：

	类别	原因短语
1XX	Informational(信息状态码)	接受的请求正在处理
2XX	Succ(成功状态吗)	请求正常处理完毕
3XX	Redirection(重定向状态码)	需要进行附加操作以完成请求
4XX	Client Error(客户端错误状态码)	服务器无法处理请求
5XX	Server Error(服务器错误状态码)	服务器处理请求出错

第五章 与HTTP协作的Web服务器

用单台虚拟主机实现多个域名

HTTP/1.1规范允许一台HTTP服务器搭建多个Web站点。即使物理层面只有一台服务器，但只要使用虚拟主机的功能，则可以假想已具有多台服务器。

通信数据转发程序

代理

接收客户端的请求并转发给服务器，同时也接收服务器的响应返回给客户端。

使用代理服务器的理由：

利用缓存技术减少带宽的流量，组织内部针对网站的控制，以获取访问日志为主要目的等。

代理按照使用方法的分类：

- 缓存代理：代理转发响应式，缓存代理会先将资源的副本保存在代理服务器上。
- 透明代理：转发请求或响应时，不对报文做任何加工的代理类型称为透明代理，反之则称为非透明代理。

网关

接收客户端发过来的请求并对其进行处理。

网关的工作机制和代理十分相似。而且网关能使通信线路上的服务器提供非HTTP协议服务。利用网关还能提高通信的安全性，因为在客户端与网关之间的通信线路上加密以确保连接的安全。

隧道

在相隔甚远的客户端和服务器两者之间进行中转，并保持双方通信连接的应用程序。

隧道可按照要求建立起一条与其他服务器的通信线路，届时使用SSL等加密手段进行通信。隧道的目的是确保客户端能与服务器进行安全的通信。隧道本身并不会去解析HTTP请求，隧道会在通信双方断开连接时结束。

第六章　HTTP首部

HTTP报文首部

HTTP请求报文：

在请求中，HTTP报文由方法，URL，HTTP版本和HTTP首部字段等构成；

HTTP响应报文：

在响应中，HTTP报文由HTTP版本，状态码，HTTP首部三个部分组成。

HTTP首部字段

在客户端和服务器之间以HTTP协议进行通信的过程中，无论是请求还是响应都会使用到首部字段，它能起到传递额外重要信息的作用。

四种HTTP首部字段类型：

- 通用首部字段（**General Header Fields**）：请求报文和响应报文两方都会使用的首部；
- 请求首部字段（**Request Header Fields**）：从客户端向服务器发送请求报文时使用的首部。补充了请求的附加内容，客户端的信息，响应内容相关的优先级等信息。
- 响应首部字段（**Response Header Fields**）：从服务器向客户端返回响应报文时使用的首部。补充了响应的附加内容，也会要求客户端附加额外的内容信息。
- 实体首部字段（**Entity Header Fields**）：针对请求报文和响应报文的实体部分使用的首部。补充了资源内容更新时间等与实体有关的信息。

HTTP首部字段一览表

通用首部字段

首部字段名	说明
Cache-Control	控制缓存的行为
Connection	逐跳首部，连接的管理
Date	创建报文的日期时间
Pragna	报文指令
Trailer	报文末端的首部一览
Transfer-Encoding	指定报文主体的传输编码方式
Upgrade	升级为其他协议
Via	代理服务器的相关信息
Warning	错误通知

请求首部字段

首部字段名	说明
Accept	用户代理可处理的媒体类型
Accept—Charset	优先的字符集
Accept-Encoding	优先的内容编码
Accept-Language	优先的语言（自然语言）
Authorization	Web认证信息
Expect	期待服务器的指定行为
From	用户的电子邮箱地址
Host	请求资源所在服务器
if-Match	比较实体标记（ETag）
if-Modified-Since	比较资源的更新时间
if-None-Match	比较实体标记（与if-Match相反）
if-Range	资源为更新时发送实体Byte的范围请求
if-Unmodified-Since	比较资源的更新时间（与if-Modified-Since相反）
Max-Forwards	最大传输逐跳数
Proxy-Authorization	代理服务器要求客户端的认证信息
Range	实体字节范围请求
Referer	对请求中的URL的原始获取方法
TE	传输编码的优先级
User-Agent	HTTP客户端程序的信息

响应首部字段

首部字段名	说明
Accept-Ranges	是否接受字节范围请求
Age	推算资源创建经过时间
ETag	资源的匹配信息
Location	令客户端重定向至指定的URL
Proxy-Authenticate	代理服务器对客户端的认证信息
Rety-After	对再次发起请求的时机要求
Server	HTTP服务器的安装信息
Vary	代理服务器缓存的管理信息
WWW-Authenticate	服务器对客户端的认证信息

实体首部字段

首部字段名	说明
Allow	资源科支持的HTTP方法
Content-Encoding	实体主体适用的编码方式
Content-Language	实体主体的自然语言
Content-Length	实体主体的大小（单位：字节）
Content-Location	替代对资源的URL
Content-MD5	实体主体的报文摘要
Content-Range	实体主体的位置范围
Content-Type	实体主体的媒体类型
Expires	实体主体过期的日期时间
Last-Modified	资源的最后修改日期时间

为Cookie服务的首部字段

首部字段名	说明	首部类型
Set-Cookie	开始状态管理所有的Cookie信息	响应首部字段
Cookie	服务器接收到的Cookie信息	请求首部字段

Set—Cookie字段的属性

属性	说明
NAME=VALUE	赋予Cookie的名称和其值
expires=DATE	Cookie的有效期（若不mingque指定则默认为浏览器关闭前为止）
path=PATH	将服务器上的文件目录作为Cookie的适用对象（若不指定则默认为文档所在的目录）
domain=域名	作为Cookie适用对象的域名（若不指定则默认为创建Cookie的服务器的域名）
Scure	仅在HTTPS安全通信时才会发送Cookie
HttpOnly	加以限制，使Cookie不能被JavaScript脚本访问

第七章 确保Web安全的 HTTPS

在HTTP协议中有可能存在信息窃听或身份伪装等安全问题，使用HTTPS通信机制可以有效地防止这些问题。

HTTP的缺点

- 通信使用明文（不加密），内容可能会被窃听；
- 不验证通信方的身份，因此可能遭遇伪装；
- 无法证明报文的完整性，所以可能已遭篡改。

HTTP + 加密 + 认证 + 完整性保护 = HTTPS

HTTPS并非是应用层的一种新协议。只是普通HTTP通信接口部分用SSL和TLS协议替代而已。

SSL是独立于HTTP的协议，所以不光是HTTP协议，其他运行在应用层的SMTP和Telnet等协议均可配合SSL协议使用。所以说SSL是当今世界上应用最为广泛的网络安全技术。

- 由于HTTPS需要做服务器、客户端双方加密及解密处理，因此会消耗CPU和内存等硬件资源；
- 和HTTP相比，SSL通信部分消耗网络资源。而SSL通信部分，又因为要对通信进行处理，所以时间上有延迟了；
- 和HTTP相比，网络负载和速度上会变慢2~100倍。

为什么不一直使用HTTPS？

- 一个原因与纯文本通信相比，加密通信会消耗更多的CPU及内存资，能够处理的请求数量必定会减少；
- 另一个原因，想要节约购买证书的开销也是原因之一（通常一年的授权需要数万日元[一万日元600左右人民币]）。

因此，如果是非敏感信息则使用HTTP通信，只有在包含个人信息等敏感数据时，才利用HTTPS加密通信。

第八章 确认访问用户身份的认证

某些Web页面只想让特定的人浏览，或者干脆仅让本人可见，为了达到这个目的，必不可少的就是认证功能。

何为认证

- 密码： 只有本人才会知道的字符串信息；
- 动态令牌： 仅限本人持有的设备内显示的一次性密码；
- 数字证书： 仅限本人终端持有的信息；
- 生物认证： 指纹和虹膜等本人的生理信息；
- **IC**卡等： 仅限本人持有的信息。

HTTP/1.1使用的认证方式

- **BASIC**认证：基本认证
- **DIGEST**认证：摘要认证
- **SSL**客户端认证：
- **FormBase**认证：基于表单认证

由于使用的便利性和安全性的问题，HTTP协议标准提供的BASIC认证和DIGEST认证几乎不怎么使用，另外SSL客户端认证虽然具有高度的安全等级，但因为导入及维护费用等问题，还尚未普及，认证多为基于表单认证。

Session管理及**Cookie**应用

第九章 HTTP的功能追加协议

虽然HTTP协议既简单有快捷，但是随着时代的发展，其功能使用上捉襟见肘的疲惫状态已经凸显。

消除HTTP瓶颈的SPDY

Google在2010年发布了SPDY，其开发目标旨在解决HTTP的性能瓶颈，缩进Web页面的加载时间（%50）。

HTTP的瓶颈

- 一条连接上只能发送一个请求；
- 请求只能从客户端开始，客户端不可以接收除响应以外的指令；
- 请求/响应首部未经压缩就发送，首部信息越多延迟越大；
- 发送冗长的首部，每次互相发送相同的首部造成的浪费较多；
- 可任意选择数据压缩格式，非强制压缩发送。

Ajax的解决办法：

有效利用JavaScript和DOM的操作，以达到局部Web页面替换加载的异步通信手段。由于只更新一部分页面，响应中传输的数据量因此而减少。

Comet的解决办法：

通常服务器端接收到请求，在处理完毕后就会立即返回响应，但是为了实现推送功能，Comet会先将响应置于挂起状态，当服务器有内容更新时，再返回该响应。因此，服务器端一旦有更新，就可以立即反馈给客户端。

SPDY的设计与功能

SPDY没有完全改写HTTP协议，而是在TCP/IP的应用层与传输层之间通过新加会话层的形式运作。同时，考虑到安全问题，SDPY规定通信中使用SSL。

- 多路复用流；
- 赋予请求优先级；
- 压缩HTTP首部；
- 推送功能；
- 服务器提示功能。

使用浏览器进行全双工通信的WebSocket

WebSocket技术主要是为了解决Ajax和Comet里XMLHttpRequest附带的缺陷所引起的问题。

WebSocket协议

一旦Web服务器与客户端之间建立起WebSocket协议的通信连接，之后所有的通信都依靠这个专用协议进行。只要通信一确立，任意一方都可直接向对方发送报文。

Web服务器管理文件的WebDAV

WebDAV（基于万维网的分布式创作和版本控制）是一个可对Web服务器上的内容直接进行文件复制、编辑等操作的分布式文件系统。除了创建、删除文件的基本功能，它还具备文件创建者管理、文件编辑过程中禁止其他用户内容覆盖的枷锁功能，以及对文件内容修改的版本控制功能。

第十章 构建Web内容的技等术

- HTML/CSS
- JavaScript/DOM
- WEB应用等

第十一章 Web的攻击技术

针对Web站点的攻击技术

简单的HTTP协议本身并不存在安全性问题，因此协议本身几乎不会成为攻击的对象。

- HTTP不具备必要的安全功能
- 在客户端即可篡改请求
- 以服务器为目的的主动攻击：SQL注入、OS命令注入
- 以服务器为目的的被动攻击：跨站脚本攻击、跨站点请求伪造
- 利用用户的身份攻击企业内部网络
- HTTP首部注入攻击
- 邮件首部注入攻击
- 目录遍历攻击
- 远程文件包含漏洞