HTTP协议背后的事

一灯学堂 www.yidengxuetang.com

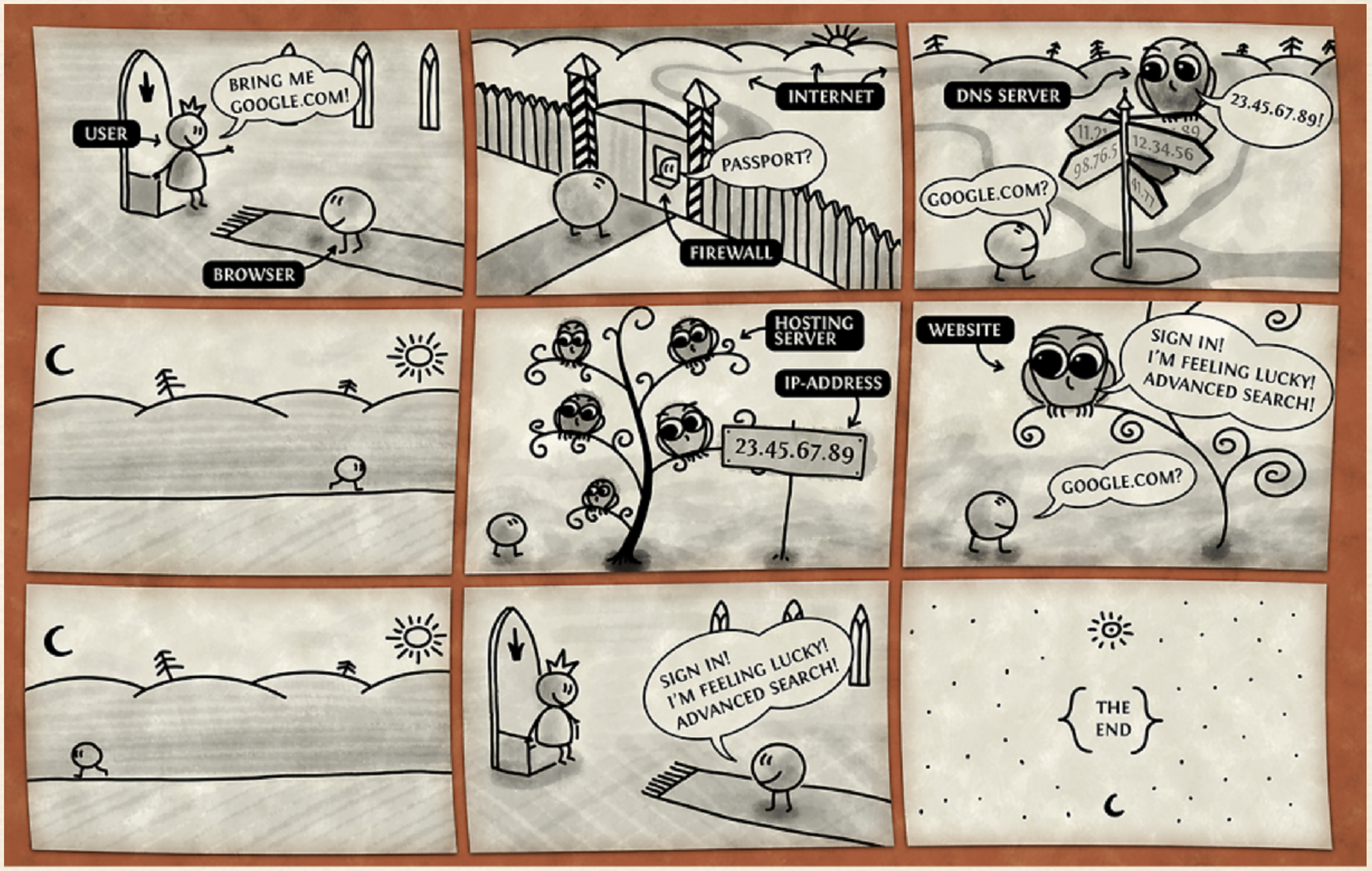
你要学到的内容

- 当我们输入网址后发生了什么
- HTTP协议详解
- HTTP协议安全
- 后台服务与HTTP
- 反向代理与WEB服务

HTTP请求模型

浏览器行为与HTTP协议

- 处理流程：
- 1、输入网址并回车
- 2、解析域名
- 3、浏览器发送HTTP请求
- 4、服务器处理请求
- 5、服务器返回HTML响应
- 6、浏览器处理HTML页面
- 7、继续请求其他资源



什么是HTTP协议

- HTTP是超文本传输协议，从www浏览器传输到本地浏览器的一种传输协议，网站是基于HTTP协议的，例如网站的图片、CSS、JS等都是基于HTTP协议进行传输的。
- HTTP协议是由从客户机到服务器的请求(Request)和从服务器到客户机的响应(response)进行约束和规范。

了解TCP/IP协议栈

1. 应用层

- 为用户提供所需要的各种服务，例如：HTTP、FTP、DNS、SMTP等.

2. 传输层

- 为应用层实体提供端到端的通信功能，保证数据包的顺序传送及数据的完整性。该层定义了两个主要的协议：传输控制协议（TCP）和用户数据报协议（UDP).

3. 网络层

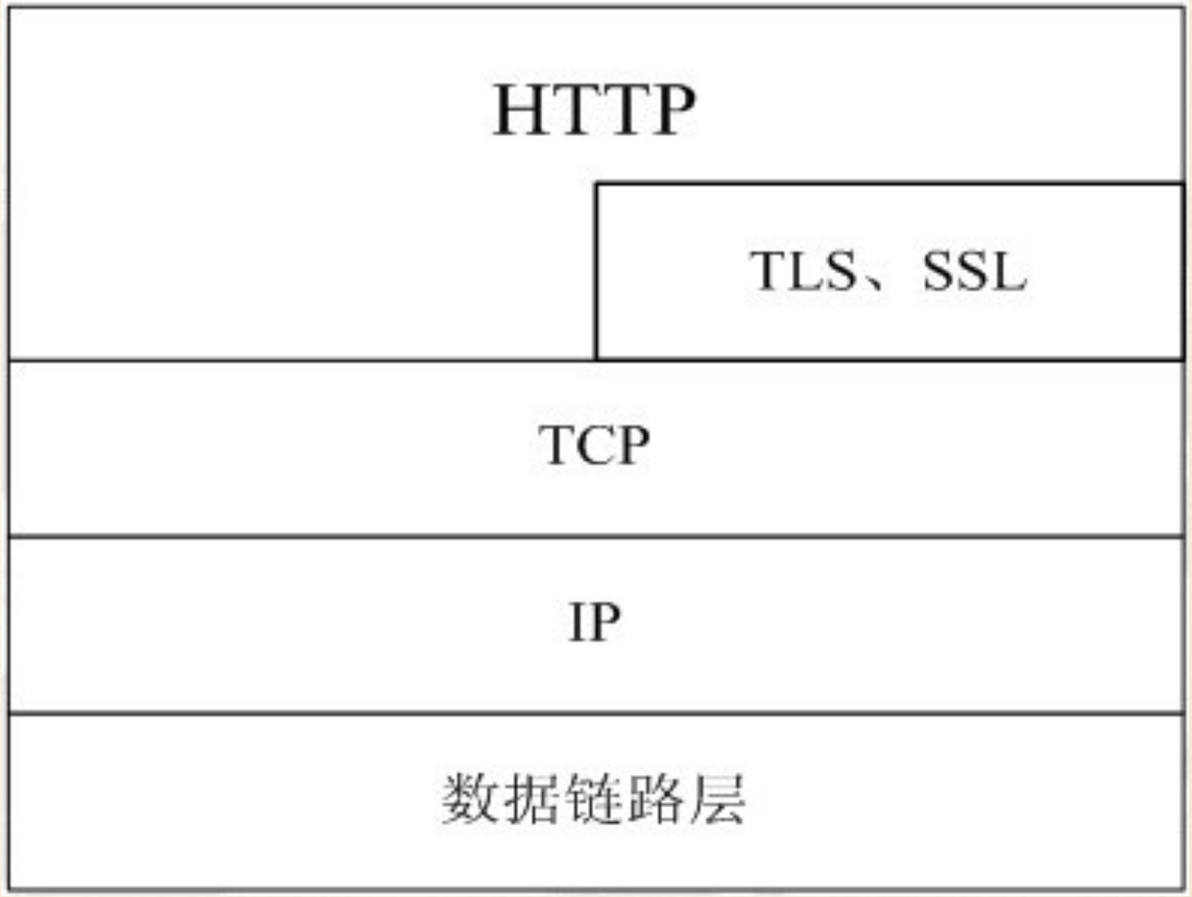
- 主要解决主机到主机的通信问题。IP协议是网际互联层最重要的协议。

4. 网络接口层

- 负责监视数据在主机和网络之间的交换。

在TCP/IP协议栈中的位置

- 目前应用版本HTTP 1.1
- HTTP默认端口号为80
- HTTPS默认端口号为443



HTTP的工作过程

一次HTTP操作称为一个事务，其工作过程可分为四步：

- 1)首先客户机与服务器需要建立连接。只要单击某个超级链接，HTTP的工作开始。
- 2)建立连接后，客户机发送一个请求给服务器，请求方式的格式为：统一资源标识符(URL)、协议版本号，后边是MIME信息包括请求修饰符、客户机信息和可能的内容。
- 3)服务器接到请求后，给予相应的响应信息，其格式为一个状态行，包括信息的协议版本号、一个成功或错误的代码，后边是MIME信息包括服务器信息、实体信息和可能的内容。
- 4)客户端接收服务器所返回的信息通过浏览器显示在用户的显示屏上，然后客户机与服务器断开连接。

如果在以上过程中的某一步出现错误，那么产生错误的信息将返回到客户端，有显示屏输出。对于用户来说，这些过程是由HTTP自己完成的，用户只要用鼠标点击，等待信息显示就可以了。

请求与响应

- HTTP请求组成：请求行、消息报头、请求正文。
- HTTP响应组成：状态行、消息报头、响应正文。
- 请求行组成：以一个方法符号开头，后面跟着请求的URI和协议的版本。
- 状态行组成：服务器HTTP协议的版本，服务器发回的响应状态代码和状态代码的文本描述。

请求报文

▼ Request Headers

:host: www.taobao.com
:method: GET
:path: /
:scheme: https
:version: HTTP/1.1
accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
accept-encoding: gzip, deflate, sdch, br
accept-language: zh-CN,zh;q=0.8,zh-TW;q=0.6
cookie: cna=v37bDcEJN3YCAW/A9PxN9K/t; miid=6267479964539816470; _med=dw:1280&dh:800&pw:2560&ph:1600&ist:0; x=e%3D1%26p%3D*%26s%3D0%26
D0%26g%3D0%26t%3D0%26__ll%3D-1%26_ato%3D0; whl=-1%260%260%261442056538235; thw=cn; uc2=wuf=http%3A%2F%2Fmail.yidengxuetang.com%2Fal
nezing_session=ZvVIHgNIVc6xhUCjiAsFAAKq_1466160149267bQyr_24; v=0; _tb_token_=t6V6SDOFph7uQTA; uc3=sg2=W5%2FRyiwzsgYZIaZK%2BjAkG5nZ
YGzgfh4%3D&nk2=DlGh1DcWwEABa7Y%3D&id2=UU22w94C72c%3D&vt3=F8dASmXQz62gqaW7Joc%3D&lg2=UtASsssmOIJ0bQ%3D%3D; existShop=MTQ2NjY3Mjg2Mg%3
H%2F4bLHxsyRenBkYOCDtStmuSL8Ttgt15Gkkjb5htKI3W98yVoKm6vinnw%3D%3D; lgc=magician000; tracknick=magician000; cookie2=1d29fc8056b96e3c9
65ed; sg=01f; mt=np=&ci=60_1&cyk=0_0; cookie1=BYjdFWafjel0fC%2FFHuVMf0b5EAHZECAGSSL9hgZTylQ%3D; unb=25482001; skt=943f06935fec85e8;
17d92ca5c210ab51700e7fe; publishItemObj=Ng%3D%3D; _cc_=VT5L2FSpdA%3D%3D; tg=0; _l_g_=Ug%3D%3D; _nk_=magician000; cookie17=UU22w94C72
ookie14=UoWxN%2FXeO2JORg%3D%3D&existShop=false&cookie16=WqG3DMC9UpAPBHGz5QBErFxlCA%3D%3D&cookie21=WqG3DMC9FxUx&tag=7&cookie15=WqG3DM
D%3D&pas=0; l=AiAgnaWV/Lb-l-jE1mlLwNbZcCTy1QT/
dnt: 1
upgrade-insecure-requests: 1
user-agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/50.0.2661.102 Safari/537.36

响应报文

▼ General

Request URL: https://www.taobao.com/
Request Method: GET
Status Code: 200 OK
Remote Address: 119.167.195.253:443

▼ Response Headers

age: 62
cache-control: max-age=0, s-maxage=100
content-encoding: gzip
content-type: text/html; charset=utf-8
date: Sat, 25 Jun 2016 03:31:01 GMT
eagleid: 77a7c3e614668254615463243e
server: nginx
status: 200 OK
strict-transport-security: max-age=31536000
timing-allow-origin: *
vary: Ali-Detector-Type, X-CIP-PT
version: HTTP/1.1
via: cache12.l2nu16[194,200-0,C], cache47.l2nu16[185,0], cache10.cn106[0,200-0,H], cache4.cn106[0,0]
x-cache: HIT TCP_MEM_HIT dirn:-2:-2
x-swift-cachetime: 100
x-swift-savetime: Sat, 25 Jun 2016 03:29:59 GMT

请求方法

- GET：　请求获取Request-URI所标识的资源
- POST：　在Request-URI所标识的资源后附加新的数据
- HEAD：　请求获取由Request-URI所标识的资源的响应消息报头
- PUT：　请求服务器存储一个资源，并用Request-URI作为其标识
- DELETE：请求服务器删除Request-URI所标识的资源
- TRACE：　请求服务器回送收到的请求信息，主要用于测试或诊断
- CONNECT：保留将来使用
- OPTIONS：请求查询服务器的性能，或者查询与资源相关的选项和需求

HTTP状态码

- 状态代码有三位数字组成，第一个数字定义了响应的类别，且有五种可能取值：
- 1xx：指示信息--表示请求已接收，继续处理
- 2xx：成功--表示请求已被成功接收、理解、接受
- 3xx：重定向--要完成请求必须进行更进一步的操作
- 4xx：客户端错误--请求有语法错误或请求无法实现
- 5xx：服务器端错误--服务器未能实现合法的请求

常用的请求报头

- Accept请求报头域用于指定客户端接受哪些类型的信息。eg：Accept：image/gif，Accept：text/htmlAccept-Charset请求报头域用于指定客户端接受的字符集。Accept-Encoding：Accept-Encoding请求报头域类似于Accept，但是它是用于指定可接受的内容编码。
- Accept-Language请求报头域类似于Accept，但是它是用于指定一种自然语言。
- Authorization请求报头域主要用于证明客户端有权查看某个资源。当浏览器访问一个页面时，如果收到服务器的响应代码为401（未授权），可以发送一个包含Authorization请求报头域的请求，要求服务器对其进行验证。
- Host请求报头域主要用于指定被请求资源的Internet主机和端口号，它通常从HTTP URL中提取出来的，发送请求时，该报头域是必需的。
- User-Agent请求报头域允许客户端将它的操作系统、浏览器和其它属性告诉服务器。

常用的响应报头

- Location响应报头域用于重定向接受者到一个新的位置。Location响应报头域常用在更换域名的时候。
- Server响应报头域包含了服务器用来处理请求的软件信息。与User-Agent请求报头域是相对应的。
- WWW-Authenticate响应报头域必须被包含在401（未授权的）响应消息中，客户端收到401响应消息时候，并发送Authorization报头域请求服务器对其进行验证时，服务端响应报头就包含该报头域。

实体报头

- 请求和响应消息都可以传送一个实体。一个实体由实体报头域和实体正文组成，但并不是说实体报头域和实体正文要在一起发送，可以只发送实体报头域。实体报头定义了关于实体正文（eg：有无实体正文）和请求所标识的资源的元信息。

常用的实体报头

- Content-Encoding实体报头域被用作媒体类型的修饰符，它的值指示了已经被应用到实体正文的附加内容的编码，因而要获得Content-Type报头域中所引用的媒体类型，必须采用相应的解码机制。
- Content-Language实体报头域描述了资源所用的自然语言。
- Content-Length实体报头域用于指明实体正文的长度，以字节方式存储的十进制数字来表示。
- Content-Type实体报头域用语指明发送给接收者的实体正文的媒体类型。
- Last-Modified实体报头域用于指示资源的最后修改日期和时间。
- Expires实体报头域给出响应过期的日期和时间。

cookies与session

- Cookies是保存在客户端的小段文本，随客户端点每一个请求发送该url下的所有cookies到服务器端。
- Session则保存在服务器端，通过唯一的值sessionID来区别每一个用户。SessionID随每个连接请求发送到服务器，服务器根据sessionID来识别客户端，再通过session 的key获取session值。

Cookie使用

与Cookie相关的HTTP扩展头

- 1)Cookie：客户端将服务器设置的Cookie返回到服务器;
- 2)Set-Cookie：服务器向客户端设置Cookie;

服务器在响应消息中用Set-Cookie头将Cookie的内容回送给客户端，客户端在新的请求中将相同的内容携带在Cookie头中发送给服务器。从而实现会话的保持。

Session的使用

- 使用Cookie来实现
- 使用URL回显来实现

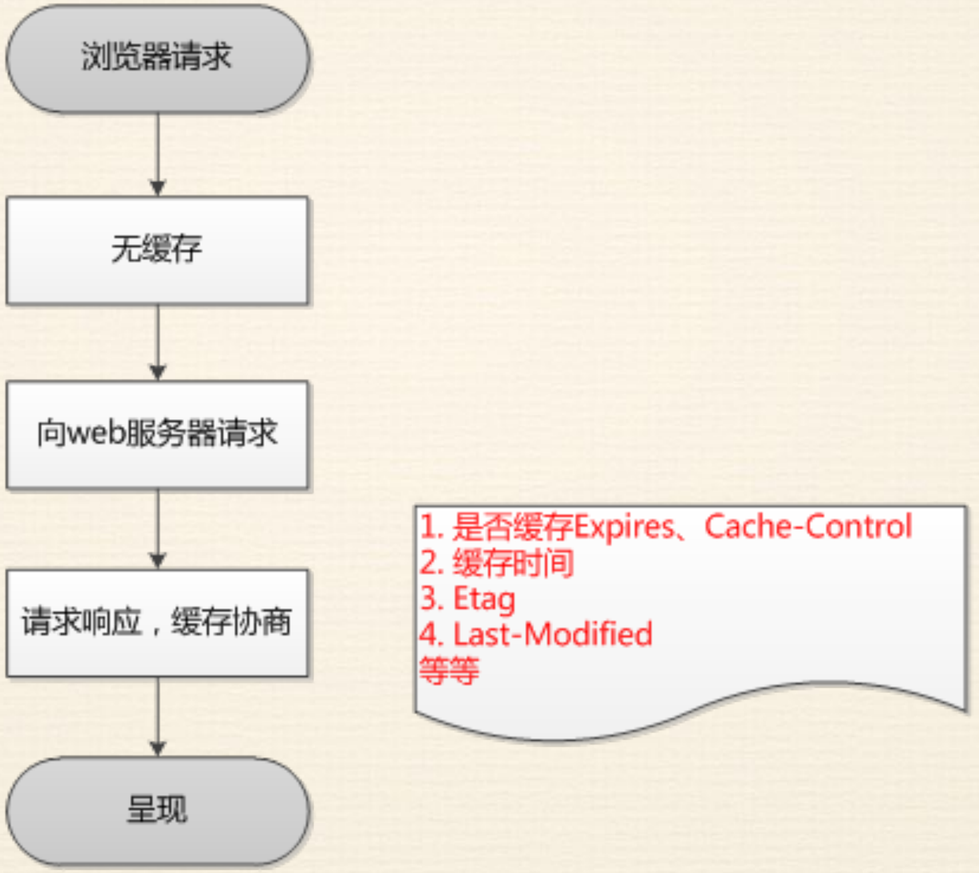
缓存机制

缓存会根据请求保存输出内容的副本，例如html页面，图片，文件，当下一个请求来到的时候：如果是相同的URL，缓存直接使用副本响应访问请求，而不是向源服务器再次发送请求。

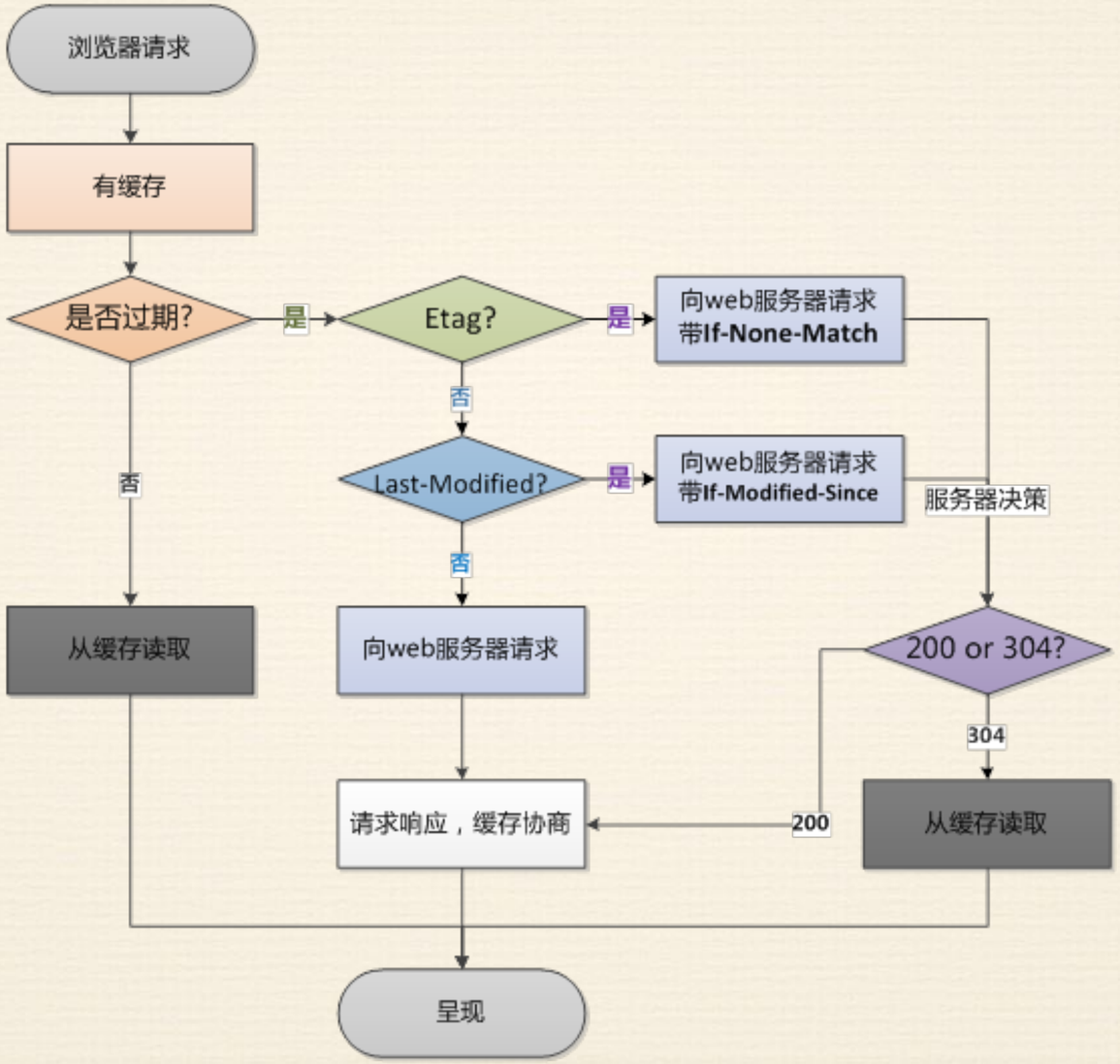
缓存的优点：

- 减少相应延迟
- 减少网络带宽消耗

浏览器缓存机制- 浏览器第一次请求



浏览器缓存机制- 浏览器再次请求



Etag/If-None-Match策略

- Etag：web服务器响应请求时，告诉浏览器当前资源在服务器的唯一标识（生成规则由服务器决定）
- If-None-Match：当资源过期时（使用Cache-Control标识的max-age），发现资源具有Etage声明，则再次向web服务器请求时带上头If-None-Match （Etag的值）。web服务器收到请求后发现有头If-None-Match 则与被请求资源的相应校验串进行比对，决定返回200或304。

Last-Modified/If-Modified-Since策略

- Last-Modified：标示这个响应资源的最后修改时间。web服务器在响应请求时，告诉浏览器资源的最后修改时间。
- If-Modified-Since：当资源过期时（使用Cache-Control标识的max-age），发现资源具有Last-Modified声明，则再次向web服务器请求时带上头 If-Modified-Since，表示请求时间。web服务器收到请求后发现有头If-Modified-Since 则与被请求资源的最后修改时间进行比对。若最后修改时间较新，说明资源又被改动过，则响应整片资源内容（写在响应消息包体内），HTTP 200；若最后修改时间较旧，说明资源无新修改，则响应HTTP 304 (无需包体，节省浏览)，告知浏览器继续使用所保存的cache。

HTTP的链路安全

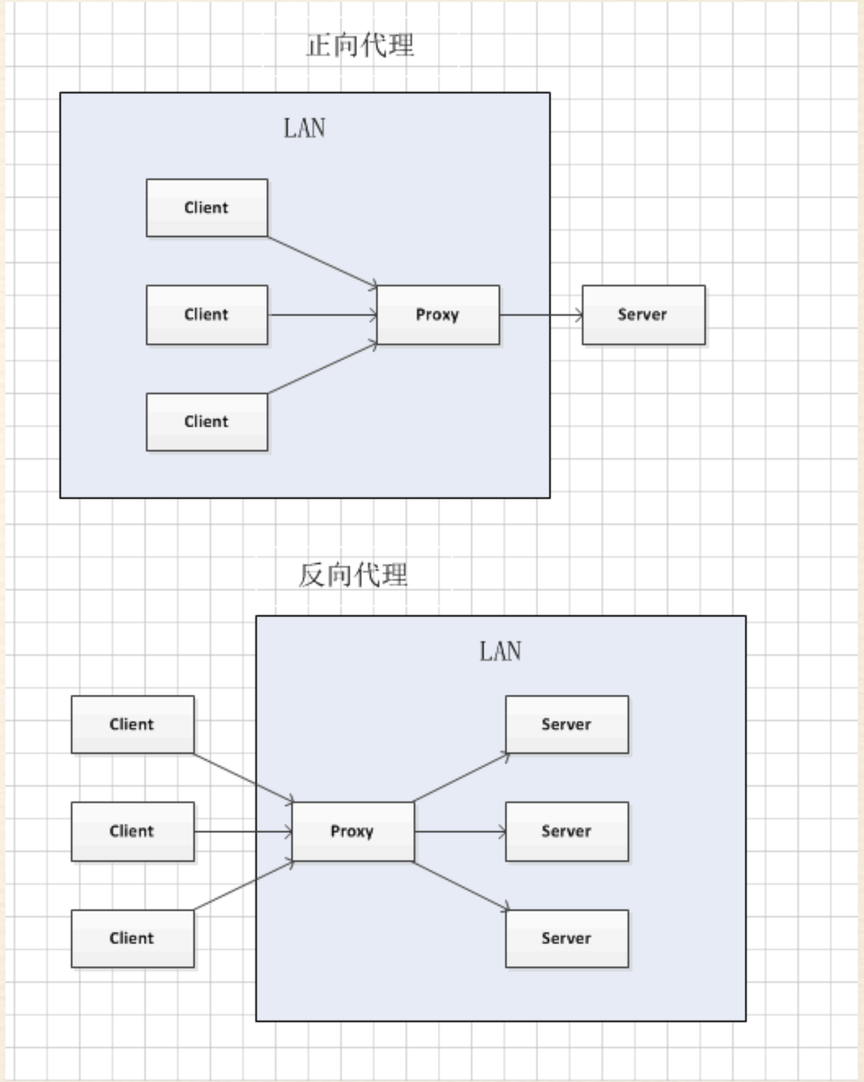
- 1、加密重要数据
- 2、对非重要数据签名
- 3、使用安全连接HTTPS协议

下一代标准:HTTP2

- 使用二进制格式传输，更高效、更紧凑。
- 对报头压缩，降低开销。
- 多路复用，一个网络连接实现并行请求。
- 服务器主动推送，减少请求的延迟

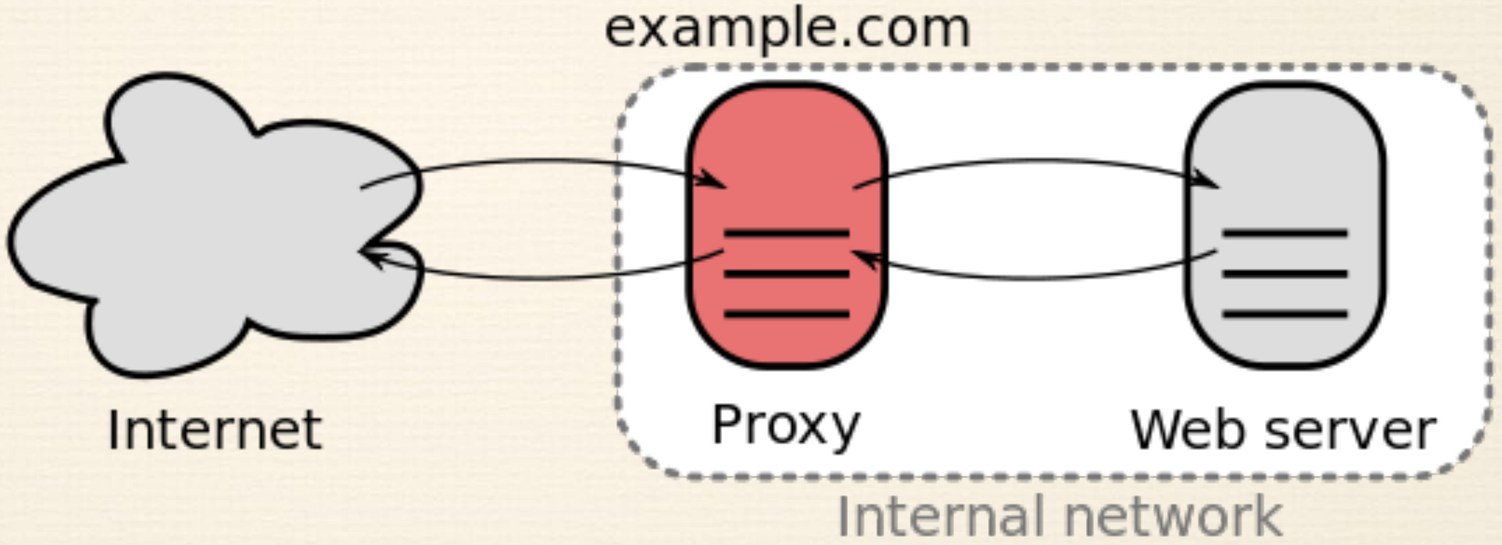
HTTP与反向代理

- 什么是代理，什么又是反向代理?
- 为什么要使用反向代理?
- 都有哪些反向代理服务器?

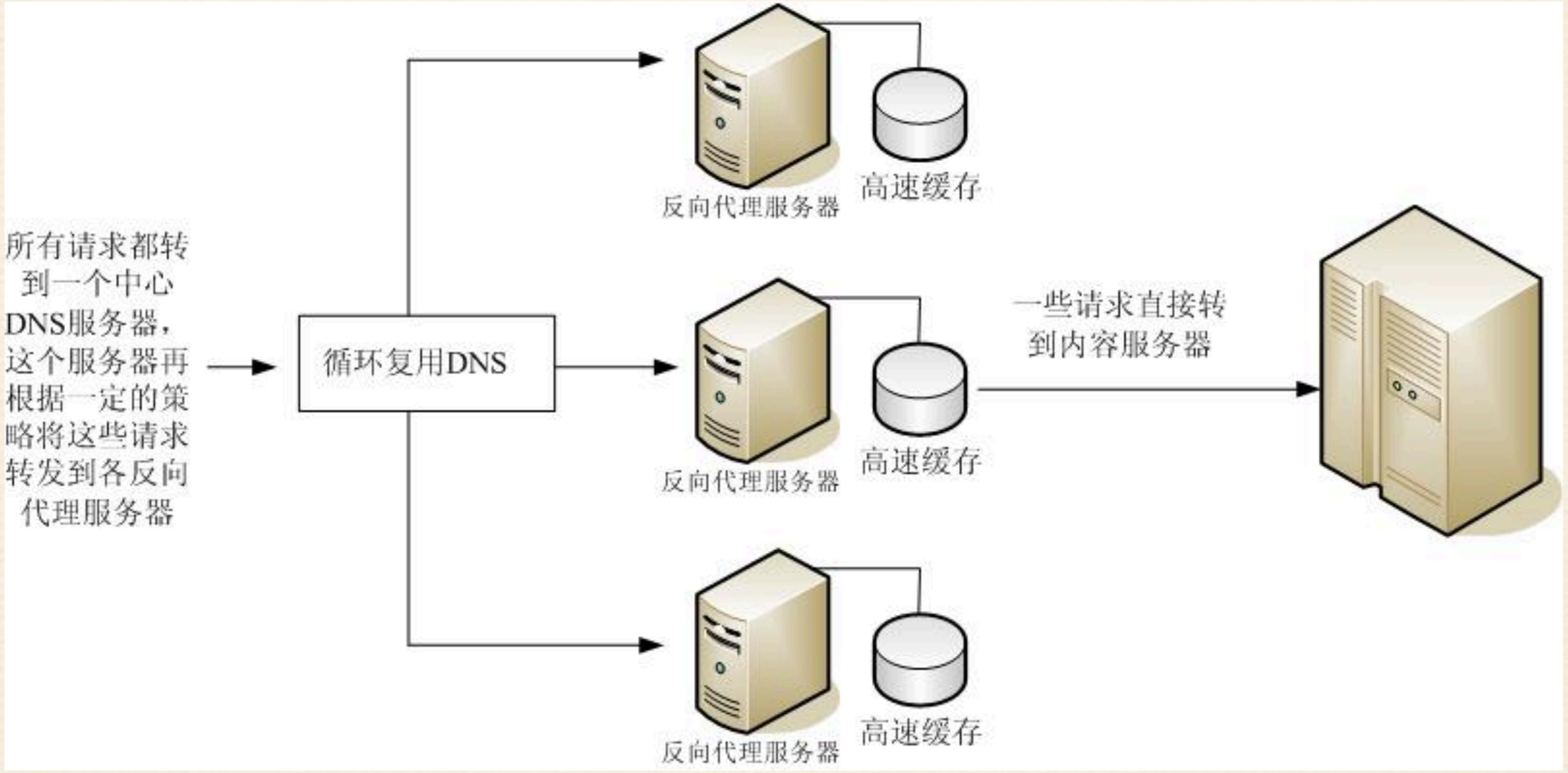


反向代理的用途

- 加密和SSL加速
- 负载均衡
- 缓存静态内容
- 压缩
- 减速上传
- 安全
- 外网发布



反向代理做负载均衡


所有请求都转到一个中心DNS服务器，这个服务器再根据一定的策略将这些请求转发到各反向代理服务器
循环复用DNS
反向代理服务器
高速缓存
反向代理服务器
高速缓存
反向代理服务器
高速缓存
一些请求直接转到内容服务器

让nginx跑起来

- 准备环境：Linux服务器、gcc编译器、nginx源代码
- 获取nginx源码：http://nginx.org
- 编译安装nginx源码
- 配置规则