HTTP协议

HTTP请求报文主要包括请求行、请求头、空行和请求数据4个部分组成。

- 请求行：请求方法、URL字段、HTTP协议版本字段，他们用空格分隔开，例如GET /index.html HTTP/1.1

  HTTTP协议的方法有:GET、POST、HEAD、PUT、DELETE、OPTIONS、TRACE、CONNECT。
- 请求头常用字段

  Accept: 浏览器接受的数据类型

  Accept-Charset: 浏览器接受的编码格式

  Accept-Encoding：浏览器接受的数据压缩格式

  Accept-Language: 浏览器接受的语言

  If-Modified-Since: 浏览器最后的缓存时间

  Host: 当前请求访问的目标地址

  Content-Type: 请求消息的类型

  Content-Length: 请求消息正文的长度

  Cookie: 浏览器缓存的Cookie信息

  Connection: 表示是否需要持久连接

  Date: 请求发出的时间

  User-Agent: 浏览器类型

  Referer: 当前请求的来源

  Cache-Control: 缓存协商
- 请求数据的三种不同方式
  - 任意类型Content-Type=application/json，告诉服务器端消息主体是序列化后的JSON字符串，服务器端为有处理JSON的函数，可以方便提交复杂的数据结构。
  - Content-Type=text/xml，这是一种使用HTTP作为传输协议，XML作为编码方式的远程调用规范，感觉类似于GRPC
  - 最常见的POST提交数据的方式，使用浏览器原生的表单Content-Type=application/x-www-form-urlencoded，多个键值对之间用&连接，键与值之前用=连接，且只能用ASCII字符，非ASCII字符需使用UrlEncode编码
- GET与POST区别，知乎链接：[!https://www.zhihu.com/question/28586791]

HTTP请求报文主要包括响应行、响应头、空行和响应数据4个部分组成。

- 响应行:HTTP协议版本字段、响应状态代码和状态代码文本描述
  - 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。

  常见状态码：

    - 200 OK：客户端请求成功。
    - 301 永久重定向。
    - 302 临时重定向
    - 304 缓存文件并未过期，还可继续使用，无需再次从服务端获取
    - 400 Bad Request：客户端请求有语法错误，不能被服务器所理解。
    - 401 Unauthorized：请求未经授权。
    - 403 Forbidden：服务器收到请求，但是拒绝提供服务。
    - 404 Not Found：请求资源不存在。
    - 500 Internal Server Error：服务器发生不可预期的错误。
    - 503 Server Unavailable：服务器当前不能处理客户端的请求，一段时间后可能恢复正常
- 响应头常见字段与请求头常见字段相似。

  Server: 服务器应用软件名称与版本

HTTP协议优化

- HTTP1.0与HTTP1.1区别
    - 缓存处理：在HTTP1.0中主要使用头部的If-Modified-Since,Expires来做缓存判断，HTTP1.1则引入了更多的缓存控制策略如Entity tag，If-Unmodified-Since, If-Match, If-None-Match
    - 带宽优化及网络连接的使用：HTTP1.1支持对象的部分请求，在请求头引入了range域，即返回码为206，充分利用带宽。
    - 错误通知管理：新增24个错误状态响应码，如409（Conflict）表示请求的资源与资源的当前状态发生冲突；410（Gone）表示服务器上的某个资源被永久性的删除。
    - Host头处理：支持Host头域，实现多个虚拟主机。
    - 长链接：HTTP1.1支持长链接与请求的流水线，默认开启Connection： keep-alive，Pipeling方法为若干个请求排队串行化单线程处理，后面的请求等待前面请求的返回才能获得执行机会，一旦有某请求超时等，后续请求只能被阻塞。
- HTTP2.0和HTTP1.x区别
    - 新的二进制格式：HTTP2.0的协议解析决定采用二进制格式，实现方便且健壮。
    - 多路复用：即连接共享，即每一个request都是是用作连接共享机制的。一个request对应一个id，这样一个连接上可以有多个request，每个连接的request可以随机的混杂在一起，接收方可以根据request的 id将request再归属到各自不同的服务端请求里面。
    - header压缩：减少头部开销
    - 服务端推送：服务器主动推送客户端需要的内容。

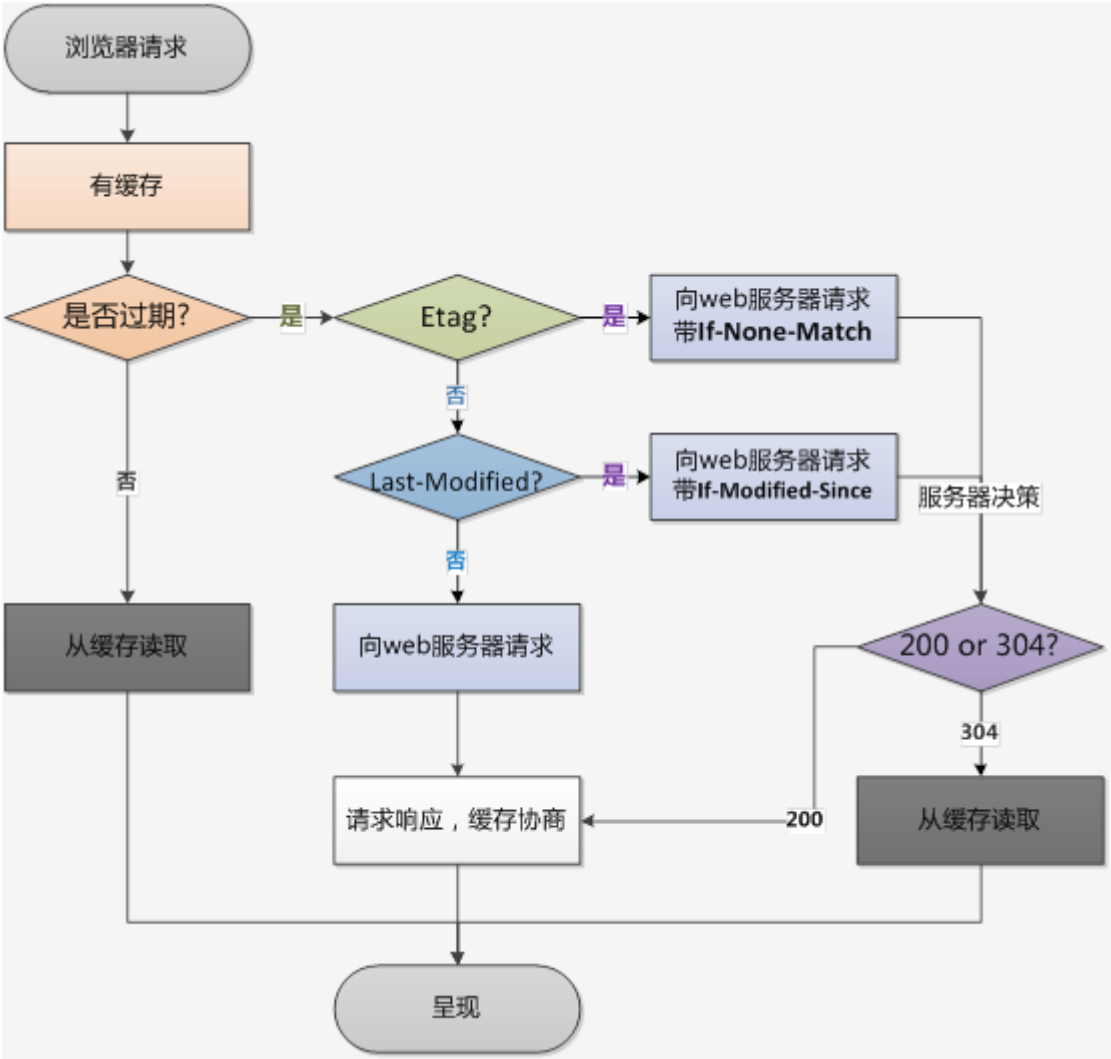
HTTP缓存机制

HTTP缓存有多种规则，根据是否需要重新向服务器发送请求分为强制缓存与对比缓存。强制缓存的优先级高于对比缓存，而Etag优先级高于Last-Modified。

强制缓存使用Cache-Control字段来控制：

- private: 客户端可以缓存；
- public: 客户端和代理服务器都可缓存；
- max-age=xxx: 缓存的内容将在 xxx 秒后失效；
- no-cache: 需要使用对比缓存来验证缓存数据；
- no-store: 所有内容都不会缓存，强制缓存，对比缓存都不会触发

对比缓存，需要进行对比判断是否可以使用缓存，浏览器第一次请求数据时，服务器将缓存标识与数据一起返回给客户端，客户端缓存二者至数据库。再次请求数据时，客户端将备份的标识发送给服务器，服务器根据缓存标识进行判断，通过返回304来表示对比成功，不成功返回200。



HTTP协议攻击

- 跨站脚本攻击(XSS)

  XSS 的原理是恶意攻击者往 Web 页面里插入恶意可执行网页脚本代码，当用户浏览该页之时，嵌入其中 Web 里面的脚本代码会被执行，从而可以达到攻击者盗取用户信息或其他侵犯用户安全隐私的目的。

- 跨站请求伪造攻击(CSRF)

  必要条件

  - 用户已经登录了站点 A，并在本地记录了 cookie
  - 在用户没有登出站点 A 的情况下（也就是 cookie 生效的情况下），访问了恶意攻击者提供的引诱危险站点 B (B 站点要求访问站点A)。
  - 站点 A 没有做任何 CSRF 防御

  预防措施

  - 正确使用 GET，POST 请求和 cookie
  - 在非 GET 请求中增加 token
- SQL注入

程序没有有效的转义过滤用户的输入，使攻击者成功的向服务器提交恶意的 SQL 查询代码，程序在接收后错误的将攻击者的输入作为查询语句的一部分执行，导致原始的查询逻辑被改变，额外的执行了攻击者精心构造的恶意代码。

防御措施

- 严格限制Web应用的数据库的操作权限，给此用户提供仅仅能够满足其工作的最低权限，从而最大限度的减少注入攻击对数据库的危害
- 后端代码检查输入的数据是否符合预期，严格限制变量的类型
- 对进入数据库的特殊字符（'，"，\，<，>，&，*，;等）进行转义处理
- 所有的查询语句建议使用数据库提供的参数化查询接口

- 命令行注入（与SQL注入类似）
- DDoS攻击（分布式拒绝服务）其原理就是利用大量的请求造成资源过载，导致服务不可用。
  - 网络层DDoS包括SYN Flood、ACK Flood、UDP Flood、ICMP Flood等。
    - SYN Flood主要理由三次握手过程中的bug，对服务器的SYN+ACK不响应，导致服务器短时间内资源和服务不可用。
    - ACK Flood
    - UDP Flood攻击者可以伪造大量的源 IP 地址去发送 UDP 包，此种攻击属于大流量攻击。正常应用情况下，UDP 包双向流量会基本相等，因此发起这种攻击的攻击者在消耗对方资源的时候也在消耗自己的资源。
  - 应用层DDoS
    - cc攻击
    - DNS Flood
    - HTTP慢速连接攻击

Cookie与Seesion

Session是在服务端保存的一个数据结构，用来跟踪用户的状态，这个数据可以保存在集群、数据库、文件中；

Cookie是客户端保存用户信息的一种机制，用来记录用户的一些信息，也是实现Session的一种最常用的方式，但并不是唯一的方法，禁用cookie后还有其他方法存储。

若Cookie禁用了，可用使用URL重写的方式，在后附加上sid=xxxxxxx，服务端据此来识别用户。

本来 session 是一个抽象概念，开发者为了实现中断和继续等操作，将 user agent 和 server 之间一对一的交互，抽象为“会话”，进而衍生出“会话状态”，也就是 session 的概念。 而 cookie 是一个实际存在的东西，http 协议中定义在 header 中的字段。可以认为是 session 的一种后端无状态实现。而我们今天常说的 “session”，是为了绕开 cookie 的各种限制，通常借助 cookie 本身和后端存储实现的，一种更高级的会话状态实现。

输入一条URL的过程