# 构建web应用(195)

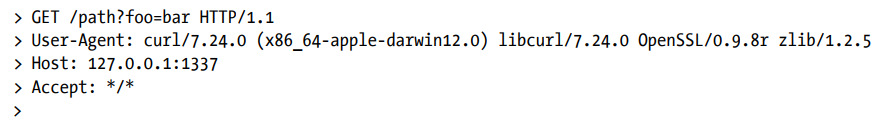
简单的服务器返回[helloworld](helloWorld.js)例子。

在具体业务中，可能还有如下需求：

1. 请求方法的判断
2. URL路径解析
3. URL查询字符串解析
4. Cookie解析
5. Basic认证
6. 表单数据解析
7. 任意格式文件上传
8. sesstion会话

8.1.1 请求

Web中常见的请求方法是GET、POST，除此之外还有HEAD、DELETE、PUT、CONNECT。



解析时，将报文头抽取出来，设为req.method。通常只有GET和POST。

PUT代表新建一个资源，POST代表更新一个资源，GET代表查看一个资源。

[method](method.js)

8.1.2 路径解析

最常见的路径的判断，存在报文第一行第二部分。

http://user:pass@host.com:8080/p/a/t/h?query=string#hash

在这个地址里，hash会被丢弃，其他的解析为req.url

8.1.3 查询字符串

?foo=bar&baz=val 报文首行的第二部分

Node提供了querystring模块来处理

8.1.4 Cookie

浏览器处理cookie：

1. 服务器想浏览器发送cookie
2. 浏览器保存cookie
3. 浏览器每次都会向服务器发送cookie

HTTP\_Parser会将所有的报文解析到req.headers，cookie就是req.headers.cookie，表现为字符串形式

告知客户端cookie的方式是，Set-Cookie API

Set-CookieSet-Cookie: name=value; Path=/; Expires=Sun, 23-Apr-23 09:01:35 GMT; Domain=.domain.com;

Path表示cookie影响的路径，当路径不匹配时，浏览器不会向服务器发送cookie。

Expires和Max-Age用来告知浏览器这个cookie什么时候过期，如果不设置此项浏览器关闭时清除这个cookie。Expires值为一个标准的UTC格式时间，Max-Age则是多久过期。Expires一般不存在问题，但是服务器时间和客户端时间不匹配的时候会出错。

HttpOnly告知浏览器不允许通过document来访问这个值，但在http请求中会带上这个值。

Secure为true时，Http中无效，只有在Https中才有效。

8.1.5 session

Cookie会增大请求体积，前端可以篡改。

Session数据只保留在服务器，且无需每次都被传递。

第一种，将session的key值保存在cookie里，这个值是唯一不重复的。如果不存在则产生一个新的session，过期了也重新产生一个session。

第二种，将session保存在url里面。

Session直接保存在sessions的变量里，占用内存。一方面可能用户过多，内存大小有限制，会引起垃圾回收的频繁扫描，带来性能问题。另一方面，可能启动多核CPU可能启动多个进程，进程间数据不共享，用户session可能错乱。

常用的方案是将分散的session转移到集中的数据储存中。常用工具有Redis、Memcached。采用第三方储存会多发起一个网络请求，不过有以下几点使我们还是采用第三方储存：

1. node与缓存服务保持长连接，握手的影响只在初始化的时候有。
2. 高速缓存区直接在内存中进行读写。
3. 通常缓存服务器与node服务器在同一机房，网络造成影响小。

因此，采用第三方储存时，获取session将变成异步形式。

Session安全

即使session只存在后台，是安全的，但是客户端也可以利用session口令间接利用服务器。这里的安全指让session口令更安全。

一种做法是将口令加密。

var sign = function (val, secret) {

return val + '.' + crypto

.createHmac('sha256', secret)

.update(val)

.digest('base64')

.replace(/\=+$/, '');

};

该方法被Connect框架中间件使用。

攻击者通过某种方式获取口令和签名，还是可以欺骗服务器进行登录。一种方案是将一些独有信息与原值进行组合，包括用户ip和user-agent。

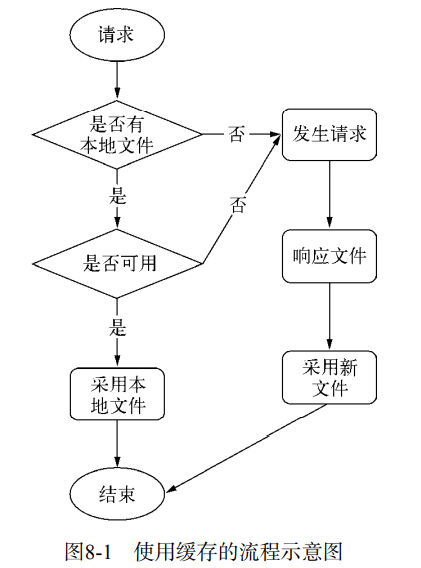
XSS漏洞

缓存

大部分html，css，js都不需要重传，造成不必要的带宽浪费。

1. 在报文中添加Expires或Cache-Control到报文中。
2. 配置ETags。
3. 让Ajax可缓存。

让浏览器缓存文件，需要前端和后台一起完成。大多数缓存只应用与GET请求。



第二次请求，浏览器会本地文件进行检查，如果不能确定这份文件是否可用，会发起一起请求。条件就在GET请求的报文中，附带If-Modified-Since。

If-Modified-Since: Sun, 03 Feb 2013 06:01:12 GMT

它将访问服务器端是否有新版本，若没有则相应304，否则将新内容发送给客户端。[Cache](cache.js)