

# 19 | 让我知道你是谁：HTTP的Cookie机制

2019-07-10 Chrono

《透视HTTP协议》

课程介绍 >



讲述：Chrono

时长 10:41 大小 12.24M



在之前的 [第 13 讲](#)、[第 14 讲](#) 中，我曾经说过，HTTP 是“无状态”的，这既是优点也是缺点。优点是服务器没有状态差异，可以很容易地组成集群，而缺点就是无法支持需要记录状态的事务操作。

好在 HTTP 协议是可扩展的，后来发明的 Cookie 技术，给 HTTP 增加了“记忆能力”。

## 什么是 Cookie?

不知道你有没有看过克里斯托弗·诺兰导演的一部经典电影《记忆碎片》（Memento），里面的主角患有短期失忆症，记不住最近发生的事情。

领资料





比如，电影里有个场景，某人刚跟主角说完话，大闹了一通，过了几分钟再回来，主角却是一脸茫然，完全不记得这个人是谁，刚才又做了什么，只能任人摆布。

这种情况就很像 HTTP 里“无状态”的 Web 服务器，只不过服务器的“失忆症”比他还要严重，连一分钟的记忆也保存不了，请求处理完立刻就忘得一干二净。即使这个请求会让服务器发生 500 的严重错误，下次来也会依旧“热情招待”。

如果 Web 服务器只是用来管理静态文件还好说，对方是谁并不重要，把文件从磁盘读出来发走就可以了。但随着 HTTP 应用领域的不断扩大，对“记忆能力”的需求也越来越强烈。比如网上论坛、电商购物，都需要“看客下菜”，只有记住用户的身份才能执行发帖子、下订单等一系列会话事务。

那该怎么样让原本无“记忆能力”的服务器拥有“记忆能力”呢？

看看电影里的主角是怎么做的吧。他通过纹身、贴纸条、立拍得等手段，在外界留下了各种记录，一旦失忆，只要看到这些提示信息，就能够在头脑中快速重建起之前的记忆，从而把因失忆而耽误的事情继续做下去。

HTTP 的 Cookie 机制也是一样的道理，既然服务器记不住，那就在外部想办法记住。相当于是服务器给每个客户端都贴上一张小纸条，上面写了一些只有服务器才能理解的数据，需要的时候客户端把这些信息发给服务器，服务器看到 Cookie，就能够认出对方是谁了。



## Cookie 的工作过程

那么，Cookie 这张小纸条是怎么传递的呢？

这要用到两个字段：响应头字段 **Set-Cookie** 和请求头字段 **Cookie**。

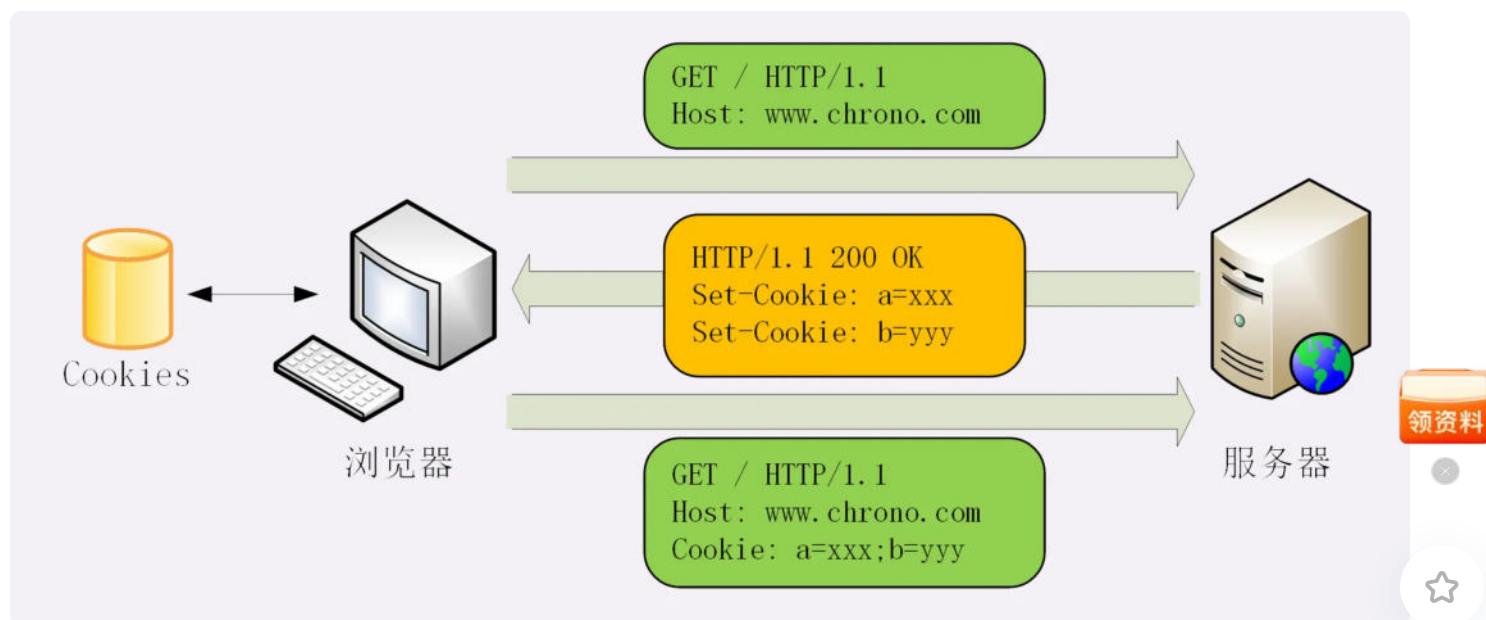
当用户通过浏览器第一次访问服务器的时候，服务器肯定是不知道他的身份的。所以，就要创建一个独特的身份标识数据，格式是“**key=value**”，然后放进 Set-Cookie 字段里，随着响应报文一同发给浏览器。

浏览器收到响应报文，看到里面有 Set-Cookie，知道这是服务器给的身份标识，于是就保存起来，下次再请求的时候就自动把这个值放进 Cookie 字段里发给服务器。

因为第二次请求里面有了 Cookie 字段，服务器就知道这个用户不是新人，之前来过，就可以拿出 Cookie 里的值，识别出用户的身份，然后提供个性化的服务。

不过因为服务器的“记忆能力”实在是太差，一张小纸条经常不够用。所以，服务器有时会在响应头里添加多个 Set-Cookie，存储多个“key=value”。但浏览器这边发送时不需要用多个 Cookie 字段，只要在一行里用“;”隔开就行。

我画了一张图来描述这个过程，你看过就能理解了。

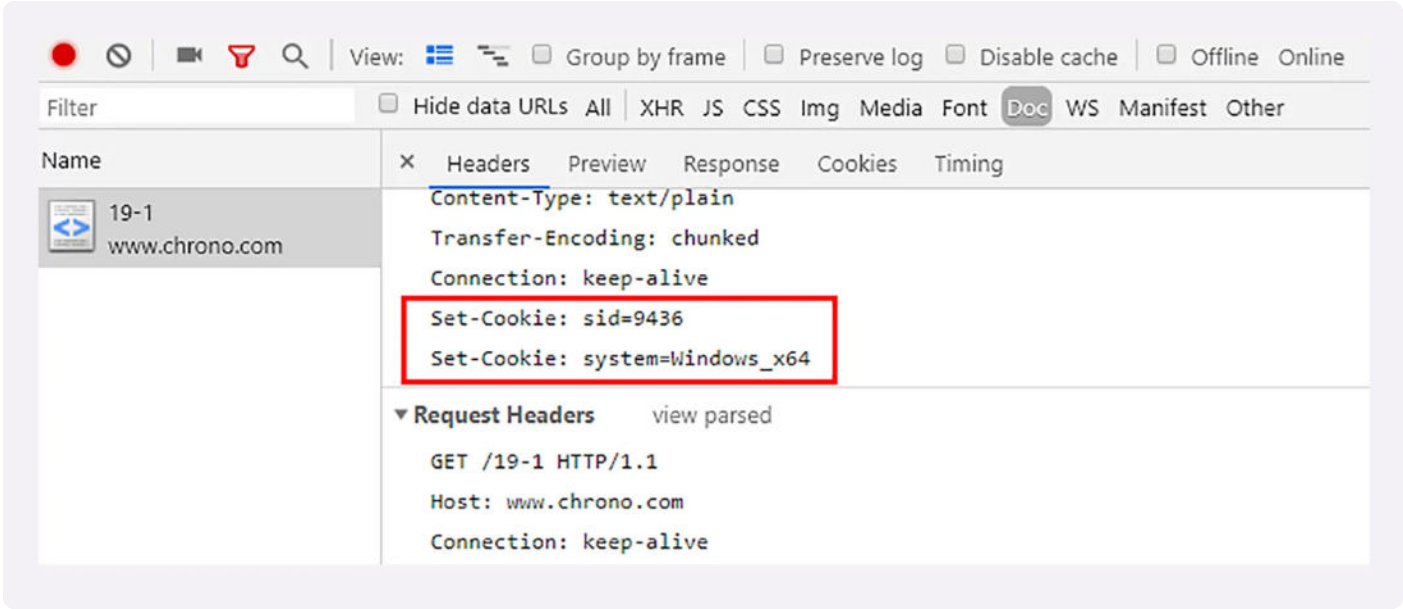


从这张图中我们也能够看到，Cookie 是由浏览器负责存储的，而不是操作系统。所以，它是“浏览器绑定”的，只能在本浏览器内生效。

如果你换个浏览器或者换台电脑，新的浏览器里没有服务器对应的 Cookie，就好像是脱掉了贴着纸条的衣服，“健忘”的服务器也就认不出来了，只能再走一遍 Set-Cookie 流程。

在实验环境里，你可以用 Chrome 访问 URI“/19-1”，实地看一下 Cookie 工作过程。

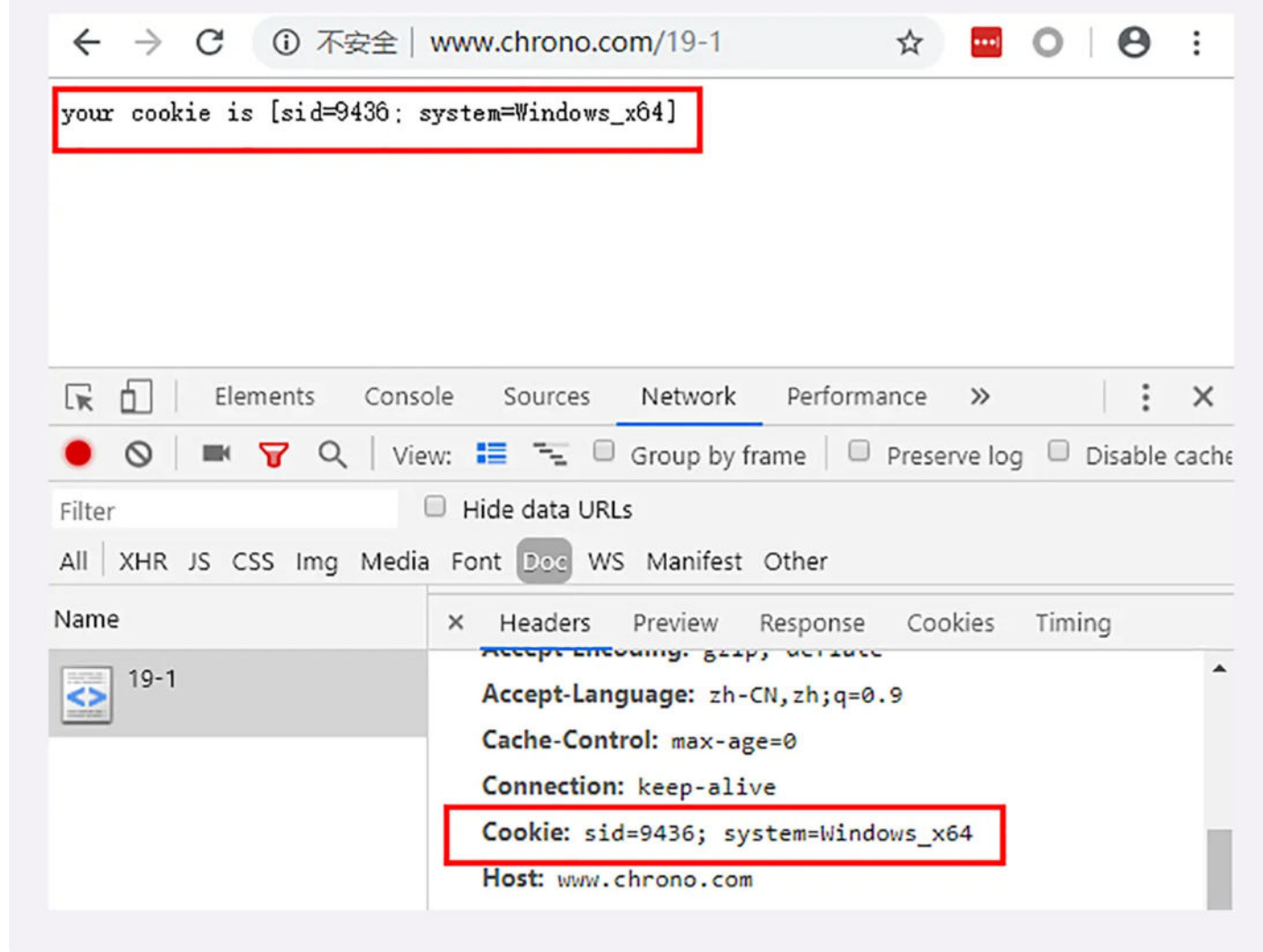
首次访问时服务器会设置两个 Cookie。



然后刷新这个页面，浏览器就会在请求头里自动送出 Cookie，服务器就能认出你了。







如果换成 Firefox 等其他浏览器，因为 Cookie 是存在 Chrome 里的，所以服务器就又“蒙圈”了，不知道你是谁，就会给 Firefox 再贴上小纸条。

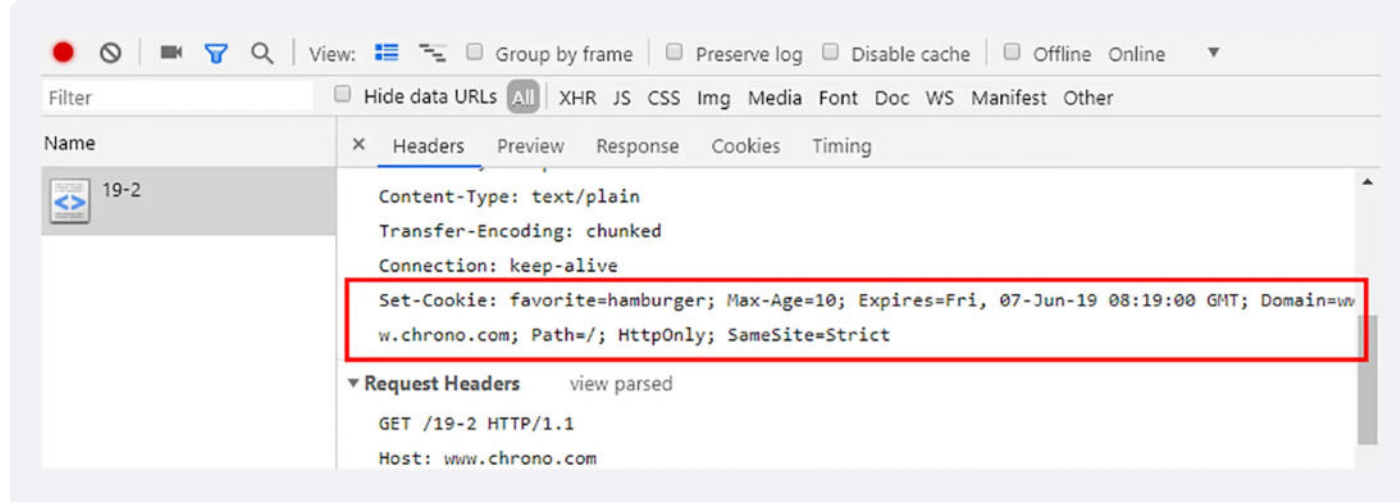
## Cookie 的属性

说到这里，你应该知道了，Cookie 就是服务器委托浏览器存储在客户端里的一些数据，而这些数据通常都会记录用户的关键识别信息。所以，就需要在“key=value”外再用一些手段来保护，防止外泄或窃取，这些手段就是 Cookie 的属性。

下面这个截图是实验环境“/19-2”的响应头，我来对着这个实际案例讲一下都有哪些常见的 Cookie 属性。

领资料





首先，我们应该设置 **Cookie 的生存周期**，也就是它的有效期，让它只能在一段时间内可用，就像是食品的“保鲜期”，一旦超过这个期限浏览器就认为是 Cookie 失效，在存储里删除，也不会发送给服务器。

Cookie 的有效期可以使用 Expires 和 Max-Age 两个属性来设置。

“**Expires**”俗称“过期时间”，用的是绝对时间点，可以理解为“截止日期”（deadline）。“**Max-Age**”用的是相对时间，单位是秒，浏览器用收到报文的时间点再加上 Max-Age，就可以得到失效的绝对时间。

Expires 和 Max-Age 可以同时出现，两者的失效时间可以一致，也可以不一致，但浏览器会优先采用 Max-Age 计算失效期。

比如在这个例子里，Expires 标记的过期时间是“GMT 2019 年 6 月 7 号 8 点 19 分”，而 Max-Age 则只有 10 秒，如果现在是 6 月 6 号零点，那么 Cookie 的实际有效期就是“6 月 6 号零点过 10 秒”。

其次，我们需要设置 **Cookie 的作用域**，让浏览器仅发送给特定的服务器和 URI，避免被其他网站盗用。

作用域的设置比较简单，“**Domain**”和“**Path**”指定了 Cookie 所属的域名和路径，浏览器在发送 Cookie 前会从 URI 中提取出 host 和 path 部分，对比 Cookie 的属性。如果不满足条件，就不会在请求头里发送 Cookie。

领资料



使用这两个属性可以为不同的域名和路径分别设置各自的 Cookie，比如“/19-1”用一个 Cookie，“/19-2”再用另外一个 Cookie，两者互不干扰。不过现实中为了省事，通常 Path 就用一个“/”或者直接省略，表示域名下的任意路径都允许使用 Cookie，让服务器自己去挑。

最后要考虑的就是 **Cookie 的安全性**了，尽量不要让服务器以外的人看到。

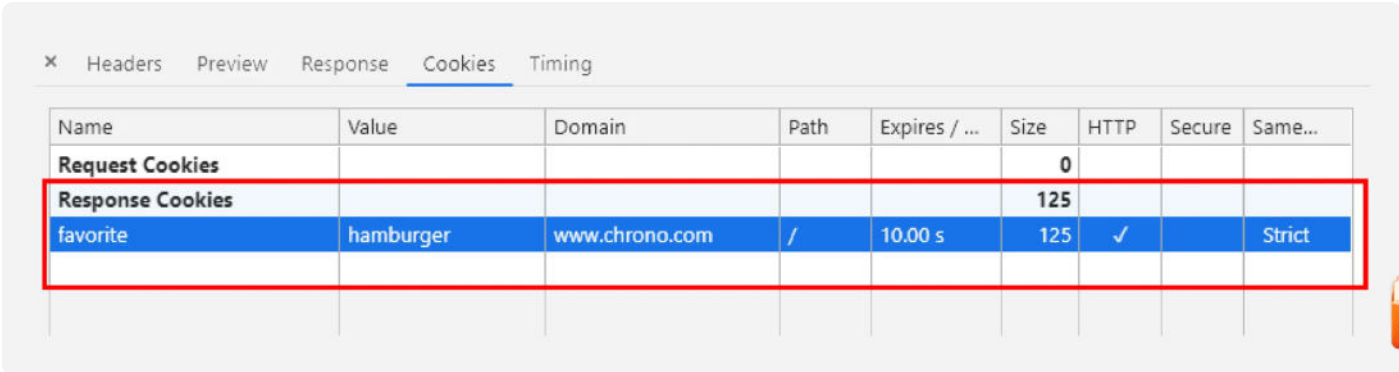
写过前端的同学一定知道，在 JS 脚本里可以用 document.cookie 来读写 Cookie 数据，这就带来了安全隐患，有可能会导致“跨站脚本”（XSS）攻击窃取数据。

属性“**HttpOnly**”会告诉浏览器，此 Cookie 只能通过浏览器 HTTP 协议传输，禁止其他方式访问，浏览器的 JS 引擎就会禁用 document.cookie 等一切相关的 API，脚本攻击也就无从谈起了。

另一个属性“**SameSite**”可以防范“跨站请求伪造”（XSRF）攻击，设置成“SameSite=Strict”可以严格限定 Cookie 不能随着跳转链接跨站发送，而“SameSite=Lax”则略宽松一点，允许 GET/HEAD 等安全方法，但禁止 POST 跨站发送。

还有一个属性叫“**Secure**”，表示这个 Cookie 仅能用 HTTPS 协议加密传输，明文的 HTTP 协议会禁止发送。但 Cookie 本身不是加密的，浏览器里还是以明文的形式存在。

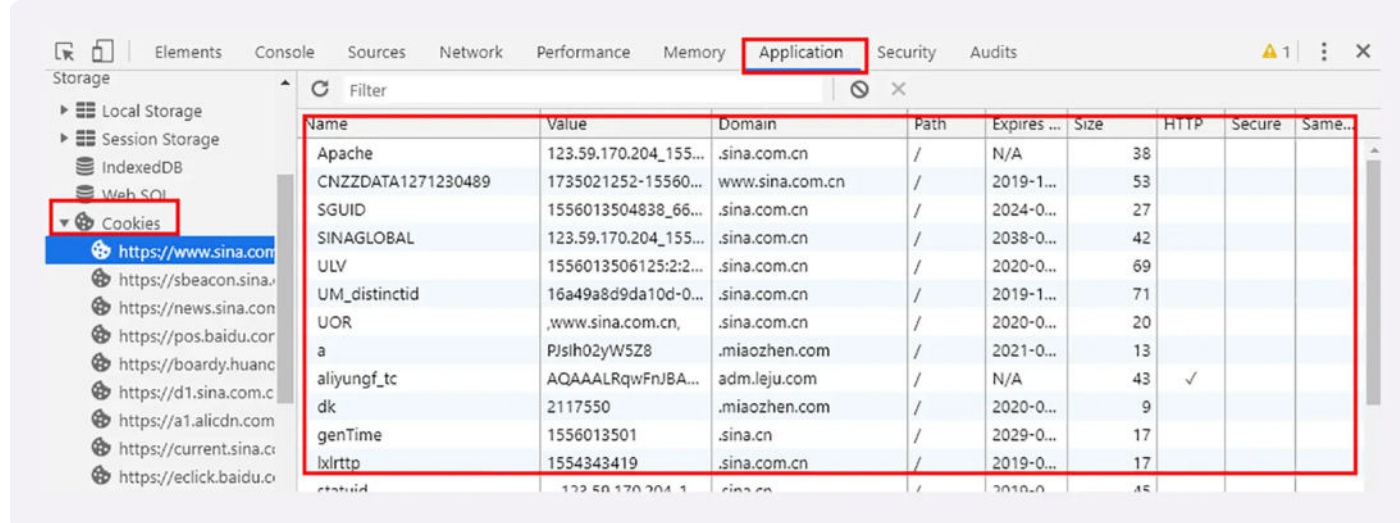
Chrome 开发者工具是查看 Cookie 的有力工具，在“Network-Cookies”里可以看到单个页面 Cookie 的各种属性，另一个“Application”面板里则能够方便地看到全站的所有 Cookie。



× Headers Preview Response Cookies Timing								
Name	Value	Domain	Path	Expires / ...	Size	HTTP	Secure	Same...
Request Cookies								
Response Cookies								
favorite	hamburger	www.chrono.com	/	10.00 s	125	✓		Strict

领资料





## Cookie 的应用

现在回到我们最开始的话题，有了 Cookie，服务器就有了“记忆能力”，能够保存“状态”，那么应该如何使用 Cookie 呢？

Cookie 最基本的一个用途就是**身份识别**，保存用户的登录信息，实现会话事务。

比如，你用账号和密码登录某电商，登录成功后网站服务器就会发给浏览器一个 Cookie，内容大概是“name=yourid”，这样就成功地把身份标签贴在了你身上。

之后你在网站里随便访问哪件商品的页面，浏览器都会自动把身份 Cookie 发给服务器，所以服务器总会知道你的身份，一方面免去了重复登录的麻烦，另一方面也能够自动记录你的浏览记录和购物下单（在后台数据库或者也用 Cookie），实现了“状态保持”。

Cookie 的另一个常见用途是**广告跟踪**。

你上网的时候肯定看过很多的广告图片，这些图片背后都是广告商网站（例如 Google），它会“偷偷地”给你贴上 Cookie 小纸条，这样你上其他的网站，别的广告就能用 Cookie 读出你的身份，然后做行为分析，再推给你广告。

这种 Cookie 不是由访问的主站存储的，所以又叫“第三方 Cookie”（third-party cookie）。如果广告商势力很大，广告到处都是，那么就比较“恐怖”了，无论你走到哪里它都会通过 Cookie 认出你来，实现广告“精准打击”。





为了防止滥用 Cookie 搜集用户隐私，互联网组织相继提出了 DNT (Do Not Track) 和 P3P (Platform for Privacy Preferences Project) ，但实际作用不大。

## 小结

今天我们学习了 HTTP 里的 Cookie 知识。虽然现在已经出现了多种 Local Web Storage 技术，能够比 Cookie 存储更多的数据，但 Cookie 仍然是最通用、兼容性最强的客户端数据存储手段。

简单小结一下今天的内容：

1. Cookie 是服务器委托浏览器存储的一些数据，让服务器有了“记忆能力”；
2. 响应报文使用 Set-Cookie 字段发送“key=value”形式的 Cookie 值；
3. 请求报文里用 Cookie 字段发送多个 Cookie 值；
4. 为了保护 Cookie，还要给它设置有效期、作用域等属性，常用的有 Max-Age、Expires、Domain、HttpOnly 等；
5. Cookie 最基本的用途是身份识别，实现有状态的会话事务。

还要提醒你一点，因为 Cookie 并不属于 HTTP 标准（RFC6265，而不是 RFC2616/7230），所以语法上与其他字段不太一致，使用的分隔符是“;”，与 Accept 等字段的“,”不同，小心不要弄错了。

## 课下作业

1. 如果 Cookie 的 Max-Age 属性设置为 0，会有什么效果呢？
2. Cookie 的好处已经很清楚了，你觉得它有什么缺点呢？

欢迎你把自己的学习体会写在留言区，与我和其他同学一起讨论。如果你觉得有所收获，也欢迎把文章分享给你的朋友。

领资料



## 课外小贴士

- 01 Cookie 这个词来源于计算机编程里的术语“Magic Cookie”，意思是不透明的数据，并不是“小甜饼”的含义（虽然字面意如此）。
- 02 早期 Cookie 直接就是磁盘上的一些小文本文件，现在基本上都是以数据库记录的形式存放的（通常使用的是 Sqlite）。浏览器对 Cookie 的数量和大小也都有限制，不允许无限存储，一般总大小不能超过 4K。
- 03 如果不指定 Expires 或 Max-Age 属性，那么 Cookie 仅在浏览器运行时有效，一旦浏览器关闭就会失效，这被称为会话 Cookie (session cookie) 或内存 Cookie (in-memory cookie)，在 Chrome 里过期时间会显示为“Session”或“N/A”。
- 04 历史上还有“Set-Cookie2”和“Cookie2”这样的字段，但现在已经不再使用。

# 透视 HTTP 协议

深入理解 HTTP 协议本质与应用

罗剑锋

奇虎360技术专家


Nginx/OpenResty 开源项目贡献者



新版升级：点击「 请朋友读」，20位好友免费读，邀请订阅更有**现金**奖励。

分享给需要的人，Ta订阅超级会员，你将得 **50** 元

Ta单独购买本课程，你将得 **20** 元

 生成海报并分享

 赞 21  提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 18 | 四通八达：HTTP的重定向和跳转

下一篇 20 | 生鲜速递：HTTP的缓存控制

领资料



# JVM + NIO + Spring

各大厂面试题及知识点详解

限时免费



## 精选留言 (67)

写留言



徐海浪

2019-07-10

1. 如果 Cookie 的 Max-Age 属性设置为 0，会有什么效果呢？  
设置为0，服务器0秒就让Cookie失效，即立即失效，服务器不存Cookie。
2. Cookie 的好处已经清楚了，你觉得它有什么缺点呢？  
好处：方便了市民  
缺点：方便了黑客:)

作者回复: √

共 2 条评论 >

41



放开那个猴子

2019-07-10

广告追踪没看明白呀，能否详细讲讲

作者回复: 是这样的，网站的页面里会嵌入很多广告代码，里面就会访问广告商，在浏览器里存储广告商的cookie。

你换到其他网站，上面也有这个广告商的广告代码，因为都是一个广告商网站，自然就能够读取之前设置的cookie，也就获得了你的信息。

领资料



**Fstar**

2019-07-11

1. (我修改 Lua 文件测试了一下) 如果 Max-Age 设置为0, 浏览器中该 Cookie 失效, 即便这个 Cookie 已存在于浏览器中, 且尚未过期。另外 Web 应用开发中, 可以通过这种方式消除掉用户的登陆状态, 此外记得在服务器的 session 中移除该 cookie 和其对应的用户信息。

2. Cookie 的缺点:

(1) 不安全。如果被中间人获取到 Cookie, 完全将它作为用户凭证冒充用户。解决方案是使用 https 进行加密。

(2) 有数量和大小限制。另外 Cookie 太大也不好, 传输的数据会变大。

(3) 客户端可能不会保存 Cookie。比如用 telnet 收发数据, 用户禁用浏览器 Cookie 保存功能的情况。

作者回复: good。



👍 21

**Geek\_66666**

2019-08-15

既然max-age=0会立即失效, 那不就等于无记忆了? 那干嘛还用cookie?

作者回复: max-age=0是指不能缓存, 但在会话期间是可用的, 浏览器会话关闭之前可以用cookie记录用户的信息。



👍 18

**饭饭**

2019-07-10

Max-age: -1 的时候会永久有效吧?

作者回复: rfc里有说明, 如果max-age <=0, 统一按0算, 立即过期。



👍 13

[领资料](#)**WL**

2019-07-10

对于XSS和XSRF一直不是很理解希望老师帮忙解答一下:

1. XSS攻击是指第三方的JS代码读取到浏览器A网站的Cookie然后冒充我去访问A网站吗?





2.XSRF是指浏览器从A网站跳转到B网站是会带上A网站的Cookie吗？这个不是由Domain和Path已经限定了吗？

作者回复: 1, 是的。

2.你理解的反了，应该是带上B网站的cookie。

<https://developer.mozilla.org/zh-CN/docs/Web/HTTP/Cookies>

这个链接里举的例子也许能够帮助你理解。

共 3 条评论 >

👍 12



业余草

2019-07-10

属性“HttpOnly”、“Secure”、“SameSite”很少见，老师可以给几个配套例子，后面答疑篇，可以来个攻防实战！

作者回复: 其实并不少见，上几个大站，用开发者工具看看就能看到。

共 2 条评论 >

👍 9



rongyefeng

2020-05-28

还有一个属性叫“Secure”，表示这个 Cookie 仅能用 HTTPS 协议加密传输，明文的 HTTP 协议会禁止发送。

但 Cookie 本身不是加密的，浏览器里还是以明文的形式存在。

这里的“Cookie 本身”怎么理解？

作者回复: 意思是cookie在传输过程是被https加密的，不能用明文的http传输，但在本地，它还是明文，没有被加密，还是能够被任意查看。



👍 6

Not Online.  
Phone Me  
When  
Necessary.

大小兵

2019-07-10

要是能把session和token也说一下就好了

作者回复: 这个不在http范围之内，而且篇幅有限，还望见谅。

共 2 条评论 >

👍 6

领资料





kissingers

2019-12-10

哈哈，就是九品芝麻官里的半个烧饼：你后人凭这个来找我

作者回复: nice。



👍 4



chengzise

2019-07-10

老师好，我理解的广告跟踪那个原理是：用户访问A网站，A会给用户设置cookies（T），用户再次访问网站B时，浏览器会带上cookies（T），B网站就能够识别到被A标记的用户。这里A给用户设置的cookies里面是不是把domain设置为B网站？不然凭什么访问B网站的时候，浏览器会带上A设置的cookies。

作者回复: 在这个场景里B是“第三方cookie”，不是A设置的，所以只要在页面里嵌入了B相关的代码，就会带上B的cookie。

完全弄清楚可能还需要一点html的知识。



👍 4



cp3yqng

2019-07-10

域名+路径的方式存储cookie，感觉像只有一台业务服务器，那后台如何过分布式系统呢，用户中心是一个系统，核心业务是其他的系统，这里cookie肯定要共享，应该有一级域名和二级域名等等的概念吧，麻烦老师在解释解释。我本人是做移动端开发的，都是自己把token写在网络底层的请求头中，其实核心思想是一样的，但是缺点就是所有的域名里面都带token，这样也不好，好像还有优化的空间。

作者回复: 只要cookie设置了domain和path属性，浏览器在访问uri时就会根据这两个属性有选择地发送cookie。

需要根据自己的业务需求，恰当设置cookie的作用域，太大太小都不好。



👍 4

领资料



业余爱好者

2019-07-10

1.Max-Age: 是永久有效的意思。

2.cookie在浏览器端禁用，还有就是安全性，因为在本地是明文存储的

作者回复: 1不对, max-age=0, 就是立即过期, 不允许缓存, 只能在浏览器运行期间有效。



4



或豪

2019-07-21

话说现在还有用cookie的吗? 感觉似乎不是太安全吧, 毕竟各种明文传输, 我司是这么做的: 浏览器登录成功, 服务端会设置一个密文的cookie, 然后浏览器再用带着这个cookie值的请求头字段去请求用户信息的接口来获取用户信息, 当然了还有其他的头字段, 然后登陆之类的接口中的相应头也看不到set-cookie字段, 但是浏览器的cookie中却能被设置上一个cookie键值对, 以及是密文的

作者回复: 有些公司为了兼容性还是用cookie的。

共 5 条评论 >

3



院长。

2019-07-11

老师我是那个回复URI会跳转的那个, 其他的测试案例也都会跳转。  
您说的实验环境openresty需要我配置什么嘛? 我应该是顺着看您的文章的, 您专栏里哪一章有介绍我漏看的吗?  
如果是我漏看的麻烦老师说下在哪一节, 谢谢老师啦。

作者回复: 感觉是dns域名解析被缓存了, 走了其他真实网站, 没有使用本地的hosts文件, 可能要清理系统的域名缓存, 具体方法可以搜一下。



3



quaeast

2020-02-15

老师您好, 我只搞不懂cookie和session的区别

作者回复: Cookie是存在客户端的, session是存在服务器的, 都是用来保存用户的会话信息。

Cookie属于http协议的规定, 而session不是协议规定, 只是服务器的一种实现方式。

共 2 条评论 >

2

领资料



GitHubGanKai

老师，有几个问题我不是很明白1：如果用户没有登录，浏览器向服务器发送请求的时候，是否还会产生cookie呢？如果会产生，那么用来干嘛呢？2：有关session，他是保存在服务器上的，如果用户没有登录，服务器是否还会产生session吗？我的理解是：应该不会产生吧！毕竟服务器保存session需要额外的空间！而且用户又没有登录，保存它做什么呢😅？3：假设现在有三个浏览器，一个在深圳，一个在北京，一个在上海，它们都是没有登录的，同时访问同一台服务器，假设没有session的存在，会出现服务器返回数据的时候，出现无法一一对应的情况吗？就是，本来这个响应是返回给深圳那个浏览器的，结果错乱的发送给北京的那台浏览器去了！

作者回复：

1.cookie并不是需要登录才会产生，而是用来记录网站的信息，比如你不登录，它也可以记录访问时间、历史记录等，可以理解成是当做一个匿名的guest用户。

2.同理，session在服务器端保存的是用户的信息，只要你登录网站，就可以认为是一个匿名用户，就可以产生session。当然网站也可以不记录，决定权在它自己。

3.每次http请求都是一个来回，是在一个tcp上发送的，所以肯定不会乱。但因为http是无状态的，如果不用cookie，就无法区分每次请求是否为同一个用户。所以，通常的做法是在cookie里把你当做一个匿名用户。

共 2 条评论 >

👍 2



許敲敲

2019-12-26

还有个名词，session 这个适合cookie放在一起的，这个老师能简单解释下嘛？

作者回复: session是指服务器保持与客户端的会话过程，利用了cookie来存储会话信息，它是服务器上的概念。

session本身与http关系不大，是在http的应用过程中出现的，基于cookie，是cookie的一个具体应用实例。



👍 2

领资料



陈坚泓

2021-10-11

罗老师好

看到您回答广告追踪的问题说

网站的页面里会嵌入很多广告代码



这个嵌入广告代码是指网站引入广告方的js代码嘛

作者回复: 我对前端不是太了解，这里说的广告代码，应该不止是js、还会有其他的一些HTML代码，比如用个div图层、link标签等。



👍 1



**Maske**

2020-06-14

samesite的加入是为了在一定程度上防范CSRF，比如a.com页面中有一个恶意表单，提交url为b.com的域名为开头且为银行转账接口地址，且为post请求，用户误操作后会将本地cookie（domain为b.com）发送，导致账户损失，设置samesite为Lax后可避免该cookie的发送，防止该情况出现。

作者回复: good



👍 1

领资料

