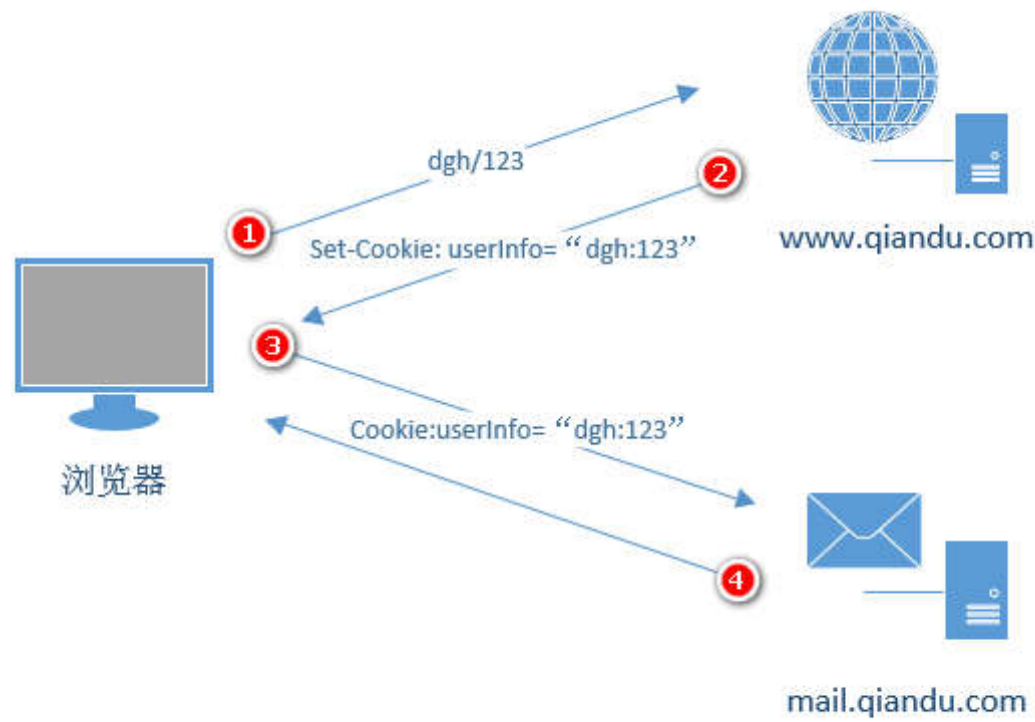


CAS单点登录原理解析（转载）

1、基于Cookie的单点登录的回顾



基于Cookie的单点登录核心原理：

公告

昵称：你明哥
园龄：2年
粉丝：4
关注：6
[+加关注](#)

<	2018年9月						>
日	一	二	三	四	五	六	
26	27	28	29	30	31	1	
2	3	4	5	6	7	8	
9	10	11	12	13	14	15	
16	17	18	19	20	21	22	
23	24	25	26	27	28	29	
30	1	2	3	4	5	6	

搜索

找找看

谷歌搜索

常用链接

- 我的随笔
- 我的评论
- 我的参与
- 最新评论
- 我的标签

随笔分类

ajax

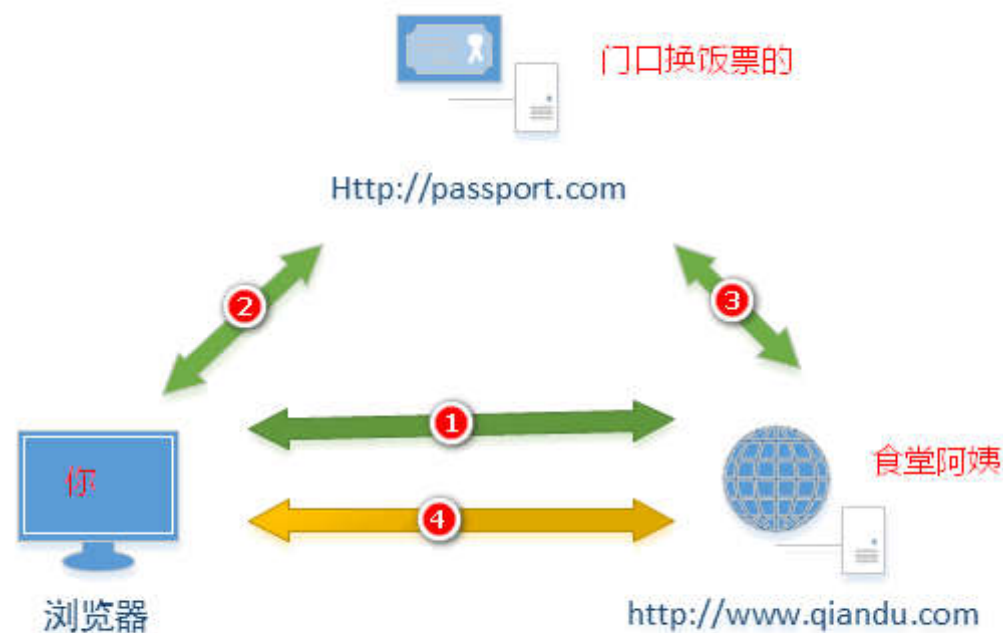
将用户名密码加密之后存于Cookie中，之后访问网站时在过滤器（filter）中校验用户权限，如果没有权限则从Cookie中取出用户名密码进行登录，让用户从某种意义上觉得只登录了一次。

该方式缺点就是多次传送用户名密码，增加被盗风险，以及不能跨域。同时www.qiandu.com与mail.qiandu.com同时拥有登录逻辑的代码，如果涉及到修改操作，则需要修改两处。

2、统一认证中心方案原理

在生活中我们也有类似的相关生活经验，例如你去食堂吃饭，食堂打饭的阿姨（www.qiandu.com）告诉你，不收现金。并且告诉你，你去门口找换票的（passport.com）换小票。于是你换完票之后，再去找食堂阿姨，食堂阿姨拿着你的票，问门口换票的，这个票是真的吗？换票的说，是真的，于是给你打饭了。

基于上述生活中的场景，我们将基于Cookie的单点登录改良以后的方案如下：



CAS 单点登录(1)

hibernate

java(12)

jquery

mybatis

mysql

nginx

oracle

redis

spring(21)

struts2(1)

线程

银联

支付宝

随笔档案

2017年4月 (1)

2017年3月 (1)

2017年2月 (1)

2016年8月 (33)

最新评论

1. Re: CAS单点登录原理解析（转载）

总结到位,感谢

--遨游边际

2. Re: Spring AOP那些学术概念—通知、增强处理
连接点 (JoinPoint)切面 (Aspect)

@傍晚的羊羔 对 jdk代理是没有办法基于继承来实现代理模式，因为是单继承，所以才通过接口的方式实现代理。

--你明哥

3. Re: Spring AOP那些学术概念—通知、增强处理
连接点 (JoinPoint)切面 (Aspect)

作者，你好。你这篇文章写得很精彩我也收获很多，但是我这边有个疑虑能请你解答下吗？你说spring的动态代理是基于接口的形式实现动态代

经过分析，Cookie单点登录认证太过于分散，每个网站都持有一份登陆认证代码。于是我们将认证统一化，形成一个独立的服务。当我们需要登录操作时，则重定向到统一认证中心<http://passport.com>。于是乎整个流程就如上图所示：

第一步：用户访问www.qiandu.com。过滤器判断用户是否登录，没有登录，则重定向（302）到网站<http://passport.com>。

第二步：重定向到passport.com，输入用户名密码。passport.com将用户登录的信息记录到服务器的session中。

第三步：passport.com给浏览器发送一个特殊的凭证，浏览器将凭证交给www.qiandu.com，www.qiandu.com则拿着浏览器交给他的凭证去passport.com验证凭证是否有效，从而判断用户是否登录成功。

第四步：登录成功，浏览器与网站之间进行正常的访问。

3、Yelu大学研发的CAS(Central Authentication Server)

下面就以耶鲁大学研发的CAS为分析依据，分析其工作原理。首先看一下最上层的项目部署图：

理的，这样的好处是松耦合，这点我不否认，但是，别人分析源码后发现，基.....

--傍晚的羊羔

4. Re:Spring AOP那些学术概念—通知、增强处理连接点 (JoinPoint)切面 (Aspect)

简单易懂，非常不错的解释

--山城码农

5. Re:spring AspectJ的Execution表达式

比较反人类

--依然是等待

阅读排行榜

1. CAS单点登录原理解析（转载）(10493)
2. java.util.vector中的vector的详细用法(9055)
3. Spring AOP那些学术概念—通知、增强处理连接点 (JoinPoint)切面 (Aspect) (5480)
4. 漫谈AOP开发之初探AOP及AspectJ的用法 (5237)
5. wap站、手机APP 接入支付宝、微信、银联支付。(3878)

评论排行榜

1. Spring AOP那些学术概念—通知、增强处理连接点 (JoinPoint)切面 (Aspect) (3)
2. CAS单点登录原理解析（转载）(1)
3. spring AspectJ的Execution表达式(1)

推荐排行榜

1. CAS单点登录原理解析（转载）(4)
2. 漫谈AOP开发之初探AOP及AspectJ的用法(1)

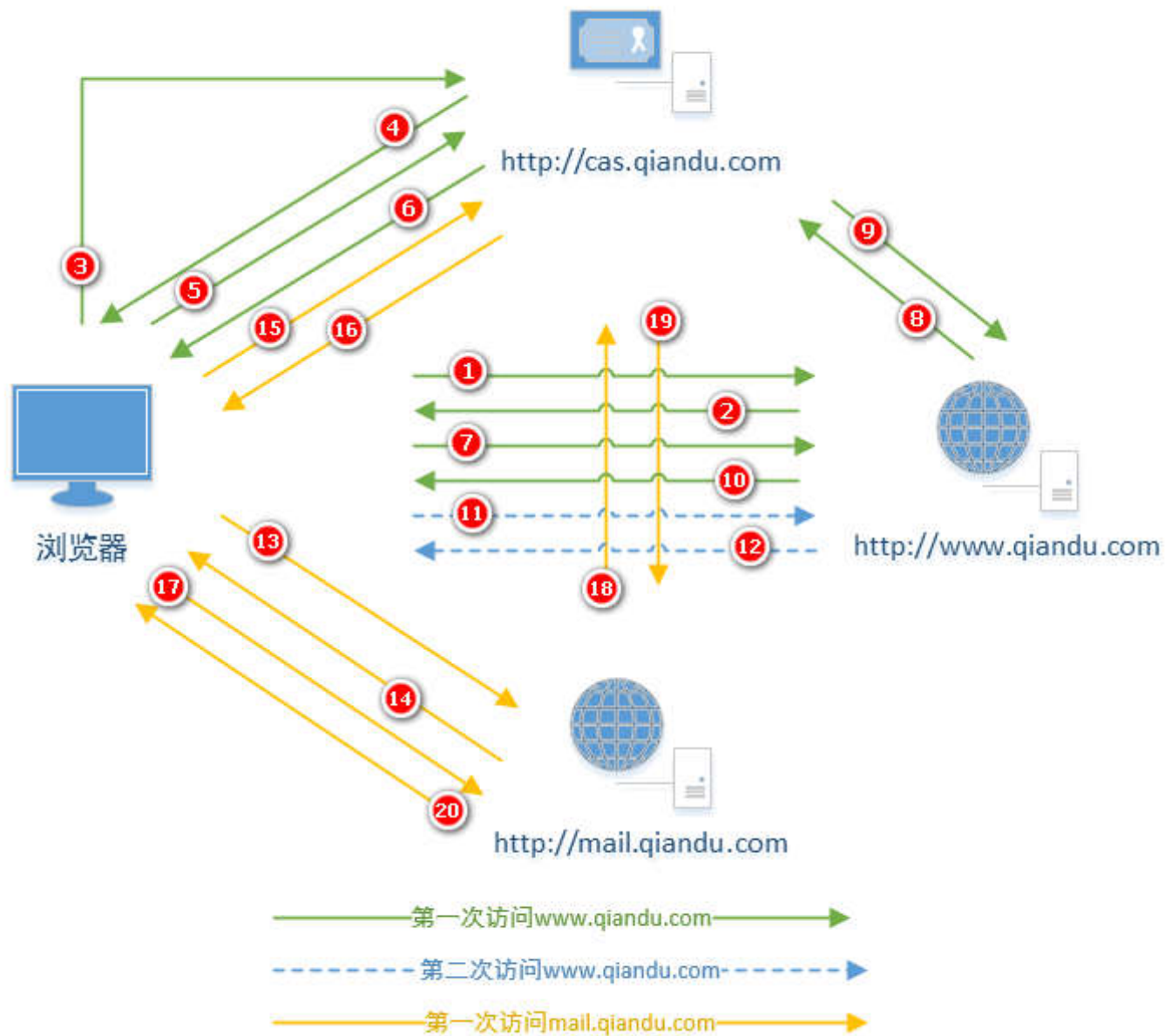


部署项目时需要部署一个独立的认证中心（cas.qiandu.com），以及其他N个用户自己的web服务。

认证中心：也就是cas.qiandu.com，即cas-server。用来提供认证服务，由CAS框架提供，用户只需要根据业务实现认证的逻辑即可。

用户web项目：只需要在web.xml中配置几个过滤器，用来保护资源，过滤器也是CAS框架提供了，即cas-client，基本不需要改动可以直接使用。

4、CAS的详细登录流程



上图是3个登录场景，分别为：第一次访问www.qiandu.com、第二次访问、以及登录状态下第一次访问mail.qiandu.com。

下面就详细说明上图中每个数字标号做了什么，以及相关的请求内容，响应内容。

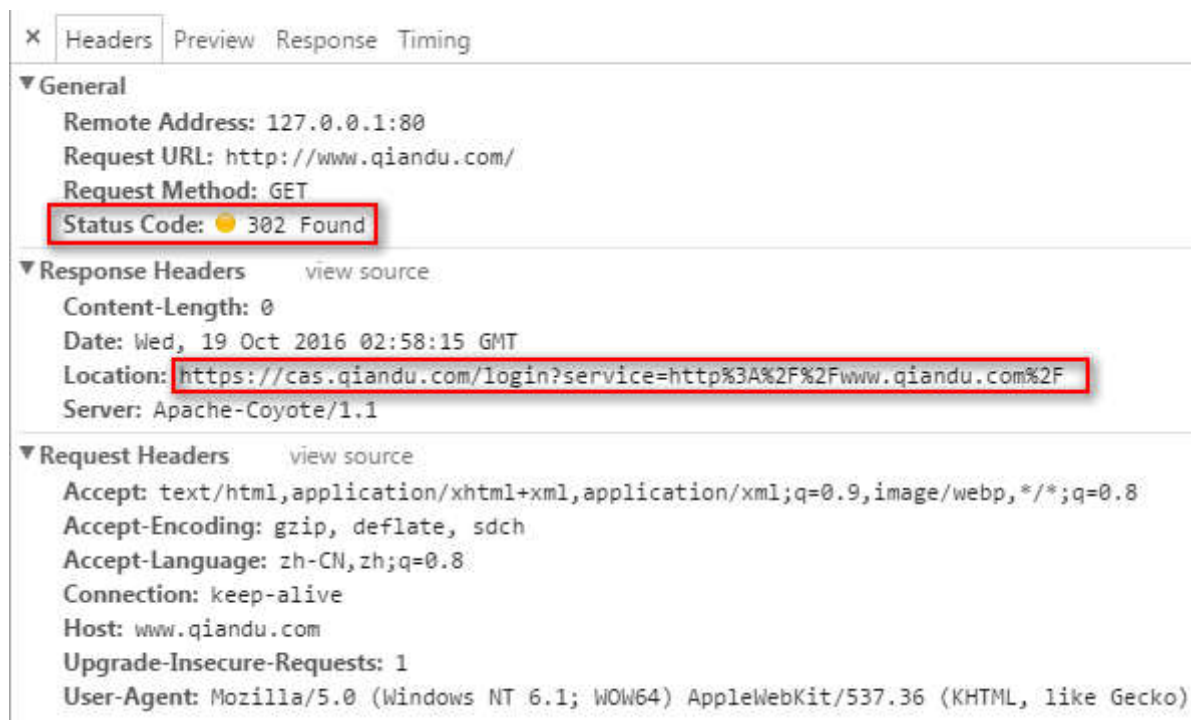
4.1、第一次访问www.qiandu.com

标号1：用户访问http://www.qiandu.com，经过他的第一个过滤器（cas提供，在web.xml中配置）AuthenticationFilter。

过滤器全称：org.jasig.cas.client.authentication.AuthenticationFilter

主要作用：判断是否登录，如果没有登录则重定向到认证中心。

标号2：www.qiandu.com发现用户没有登录，则返回浏览器重定向地址。

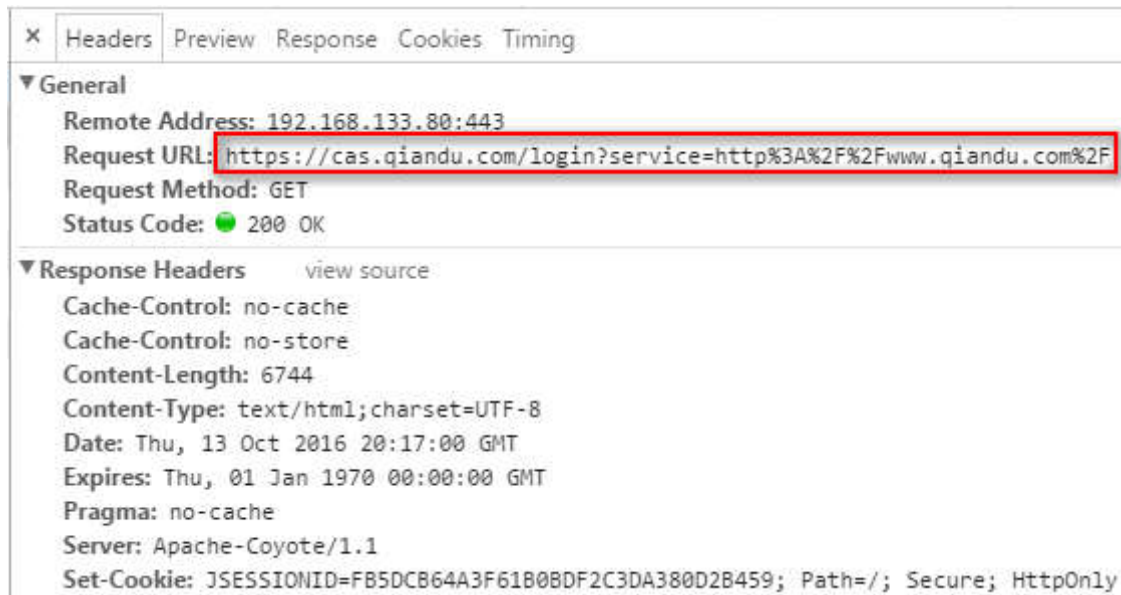


首先可以看到我们请求www.qiandu.com，之后浏览器返回状态码302，然后让浏览器重定向到cas.qiandu.com并且通过get的方式添加参数service，该参数目的是登录成功之后会要重定向回来，因此

需要该参数。并且你会发现，其实server的值就是编码之后的我们请求www.qiandu.com的地址。

标号3：浏览器接收到重定向之后发起重定向，请求cas.qiandu.com。

标号4：认证中心cas.qiandu.com接收到登录请求，返回登陆页面。



上图就是标号3的请求，以及标号4的响应。请求的URL是标号2返回的URL。之后认证中心就展示登录的页面，等待用户输入用户名密码。

标号5：用户在cas.qiandu.com的login页面输入用户名密码，提交。

标号6：服务器接收到用户名密码，则验证是否有效，验证逻辑可以使用cas-server提供现成的，也可以自己实现。

× Headers Preview Response Cookies Timing

▼ Response Headers view source

Cache-Control: no-cache
Cache-Control: no-store
Content-Length: 0
Date: Thu, 13 Oct 2016 20:33:58 GMT
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Location: <http://www.qiandu.com/?ticket=ST-2-7fDGT5YEUCUUSTLohwXa-cas01.example.org>
Pragma: no-cache
Server: Apache-Coyote/1.1
Set-Cookie: CASPRIVACY=""; Expires=Thu, 01-Jan-1970 00:00:10 GMT; Path=/
Set-Cookie: CASTGC=TGT-2-uJXdsyPcqM72Vg3lLCRr0mSMv5UP77C1wMiOUCqvxTLkr0nEY-cas01.example.org; Path=

▼ Request Headers view source

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.8
Cache-Control: max-age=0
Connection: keep-alive
Content-Length: 125
Content-Type: application/x-www-form-urlencoded
Cookie: JSESSIONID=DA70C38DD2C6E0712A40E82D273C1519
Host: cas.qiandu.com
Origin: https://cas.qiandu.com
Referer: https://cas.qiandu.com/login?service=http%3A%2F%2Fwww.qiandu.com%2F
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/4

▼ Query String Parameters view source view URL encoded

service: http://www.qiandu.com/

▼ Form Data view source view URL encoded

username: admin
password: admin

上图就是标号5的请求，以及标号6的响应了。当cas.qiandu.com即csa-server认证通过之后，会返回给浏览器302，重定向的地址就是Referer中的service参数对应的值。后边并通过get的方式挟带了一个ticket令牌，这个ticket就是ST（数字3处）。同时会在Cookie中设置一个CASTGC，该cookie是网站cas.qiandu.com的cookie，只有访问这个网站才会携带这个cookie过去。

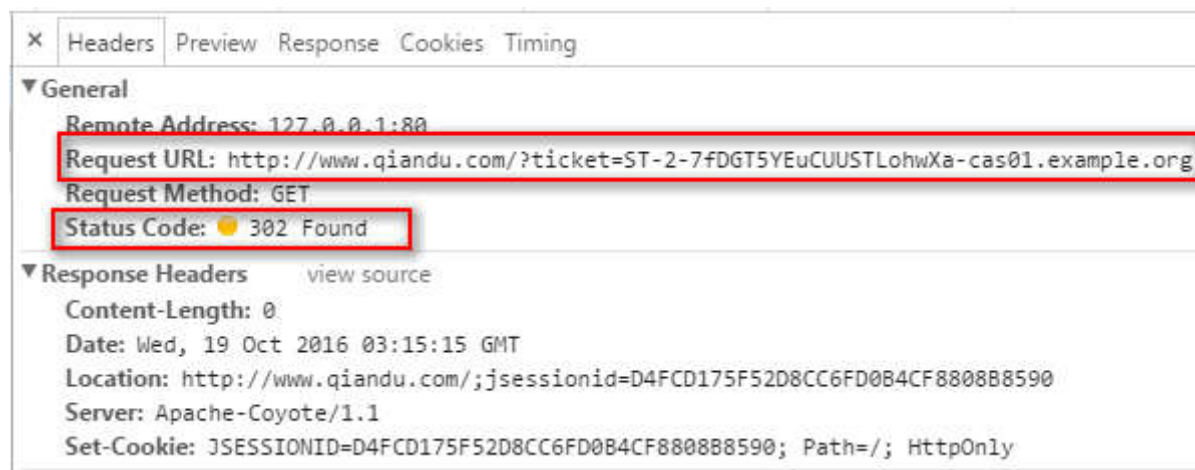
Cookie中的CASTGC：向cookie中添加该值的目的是当下次访问cas.qiandu.com时，浏览器将Cookie中的TGC携带到服务器，服务器根据这个TGC，查找与之对应的TGT。从而判断用户是否登录过了，是否需要展示登录页面。TGT与TGC的关系就像SESSION与Cookie中SESSIONID的关系。

TGT: Ticket Granted Ticket（俗称大令牌，或者说票根，他可以签发ST）

TGC: Ticket Granted Cookie（cookie中的value），存在Cookie中，根据他可以找到TGT。

ST: Service Ticket（小令牌），是TGT生成的，默认是用一次就生效了。也就是上面数字3处的ticket值。

标号7：浏览器从cas.qiandu.com哪里拿到ticket之后，就根据指示重定向到www.qiandu.com，请求的url就是上面返回的url。



标号8：www.qiandu.com在过滤器中会取到ticket的值，然后通过http方式调用cas.qiandu.com验证该ticket是否是有效的。

标号9：cas.qiandu.com接收到ticket之后，验证，验证通过返回结果告诉www.qiandu.com该ticket有效。

标号10：www.qiandu.com接收到cas-server的返回，知道了用户合法，展示相关资源到用户浏览器上。



至此，第一次访问的整个流程结束，其中标号8与标号9的环节是通过代码调用的，并不是浏览器发起，所以没有截取到报文。

4.2、第二次访问www.qiandu.com

上面以及访问过一次了，当第二次访问的时候发生了什么呢？

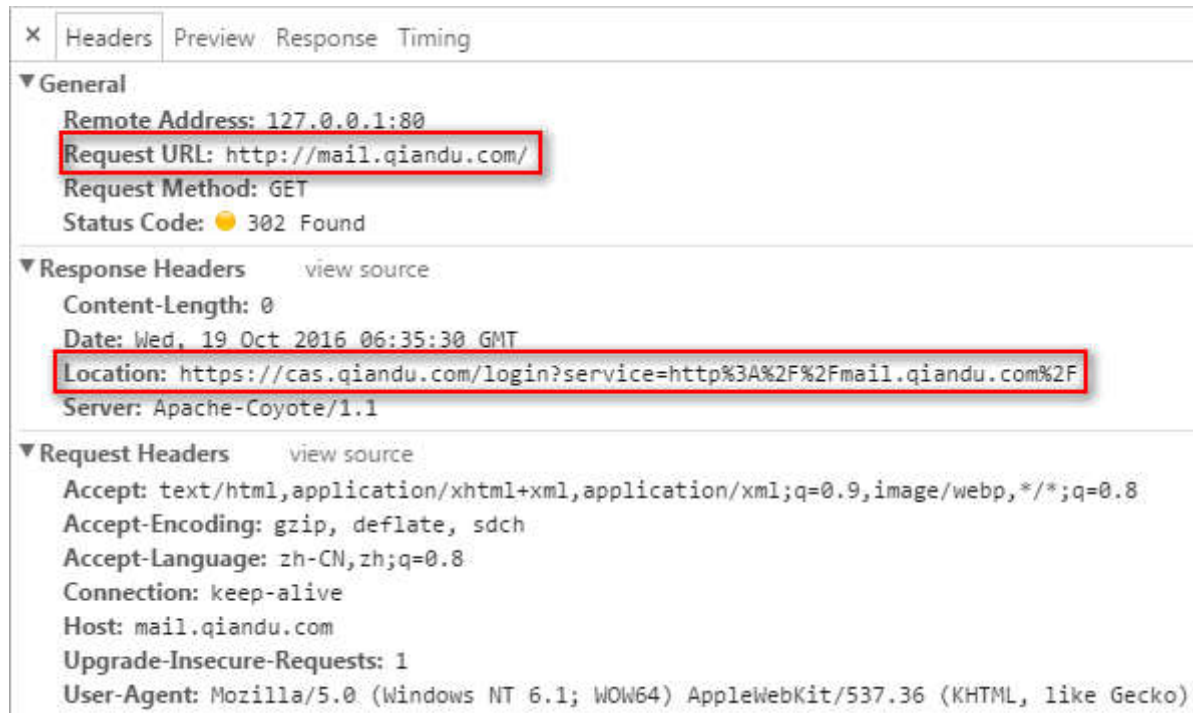
标号11：用户发起请求，访问www.qiandu.com。会经过cas-client，也就是过滤器，因为第一次访问成功之后www.qiandu.com中会在session中记录用户信息，因此这里直接就通过了，不用验证了。

标号12：用户通过权限验证，浏览器返回正常资源。

4.3、访问mail.qiandu.com

标号13：用户在www.qiandu.com正常上网，突然想访问mail.qiandu.com，于是发起访问mail.qiandu.com的请求。

标号14：mail.qiandu.com接收到请求，发现第一次访问，于是给他一个重定向的地址，让他去找认证中心登录。



上图可以看到，用户请求mail.qiandu.com，然后返回给他一个网址，状态302重定向，service参数就是回来的地址。

标号15：浏览器根据14返回的地址，发起重定向，因为之前访问过一次了，因此这次会携带上次返回的Cookie：TGC到认证中心。

标号16：认证中心收到请求，发现TGC对应了一个TGT，于是用TGT签发一个ST，并且返回给浏览器，让他重定向到mail.qiandu.com

× Headers Preview Response Cookies Timing

▼ General

Remote Address: 192.168.133.80:443

Request URL: https://cas.qiandu.com/login?service=http%3A%2F%2Fmail.qiandu.com%2F

Request Method: GET

Status Code: 302 Found

▼ Response Headers view source

Cache-Control: no-cache

Cache-Control: no-store

Content-Length: 0

Date: Thu, 13 Oct 2016 23:54:18 GMT

Expires: Thu, 01 Jan 1970 00:00:00 GMT

Location: http://mail.qiandu.com/?ticket=ST-7-GfpdKdFwtPlcrNNc1lze-cas01.example.org

Pragma: no-cache

Server: Apache-Coyote/1.1

▼ Request Headers view source

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8

Accept-Encoding: gzip, deflate, sdch

Accept-Language: zh-CN,zh;q=0.8

Connection: keep-alive

Cookie: JSESSIONID=BD5F5484C587DA071AA3479C35CBD078; CASTGC=TGT-5-rgnT3ak0FLeKxu5w0rHxxlotWcolb32aGdT9Jcd034cFdyMYdb-cas01.example.org

Host: cas.qiandu.com

Upgrade-Insecure-Requests: 1

User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/46.0.2490.86 Safari/537.36

可以发现请求的时候是携带Cookie: CASTGC的, 响应的就是一个地址加上TGT签发的ST也就是ticket。

标号17: 浏览器根据16返回的网址发起重定向。

标号18: mail.qiandu.com获取ticket去认证中心验证是否有效。

标号19: 认证成功, 返回在mail.qiandu.com的session中设置登录状态, 下次就直接登录。

标号20: 认证成功之后就反正用想要访问的资源了。



Hello ! mail.qiandu.com

5、总结

至此，CAS登录的整个过程就完毕了，以后有时间总结下如何使用CAS，并运用到项目中。

分类: [CAS 单点登录](#)



 [你明哥](#)
[关注 - 6](#)
[粉丝 - 4](#)

[+加关注](#)

« 上一篇: [请教Hibernate和JPA什么区别?](#)

» 下一篇: [wap站、手机APP 接入支付宝、微信、银联支付。](#)

4

0

posted @ 2017-03-03 10:52 你明哥 阅读(10495) 评论(1) 编辑 收藏

评论列表

#1楼 2018-06-29 11:15 遨游边际

总结到位,感谢

[支持\(0\)](#) [反对\(0\)](#)

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

【推荐】超50万VC++源码：大型组态工控、电力仿真CAD与GIS源码库！

【免费】要想入门学习Linux系统技术，你应该先选择一本适合自己的书籍

【前端】SpreadJS表格控件，可嵌入应用开发的在线Excel

【直播】如何快速接入微信支付功能



最新IT新闻：

- Windows 7迎累积更新KB4463376：只为IE11
 - 扎克伯格卷入Facebook投资者诉讼：11月将出庭作证
 - 英语流利说公布赴美IPO发行价区间：11.5-13.5美元
 - Twitter进一步突出直播功能：将直播放在时间线顶端
 - 谷歌云联合Unity开源Open Match玩家配对方案
- » 更多新闻...



最新知识库文章：

- 为什么说 Java 程序员必须掌握 Spring Boot ？
- 在学习中，有一个比掌握知识更重要的能力
- 如何招到一个靠谱的程序员
- 一个故事看懂“区块链”
- 被踢出去的用户

2018/9/17

CAS单点登录原理解析（转载） - 你明哥 - 博客园

» [更多知识库文章...](#)

Copyright ©2018 你明哥