单点登录

题目难度：★★★

知识点标签：单点登录

学习时长：20分钟

题目描述

你谈一下单点登录吧?

解题思路

> 面试官问题可以从几个方面来回答：单点登录原理、来源、实现以及技术方案

为什么需要单点登录

1）提高用户的效率。

用户不再被多次登录困扰，也不需要记住多个 ID 和密码。另外，用户忘记密码并求助于支持人员的情况也会减少。

2）提高开发人员的效率。

SSO（Single Sign On）为开发人员提供了一个通用的身份验证框架。实际上，如果 SSO 机制是独立的，那么开发人员就完全不需要为身份验证操心。他们可以假设，只要对应用程序的请求附带一个用户名，身份验证就已经完成了。

3）简化管理。

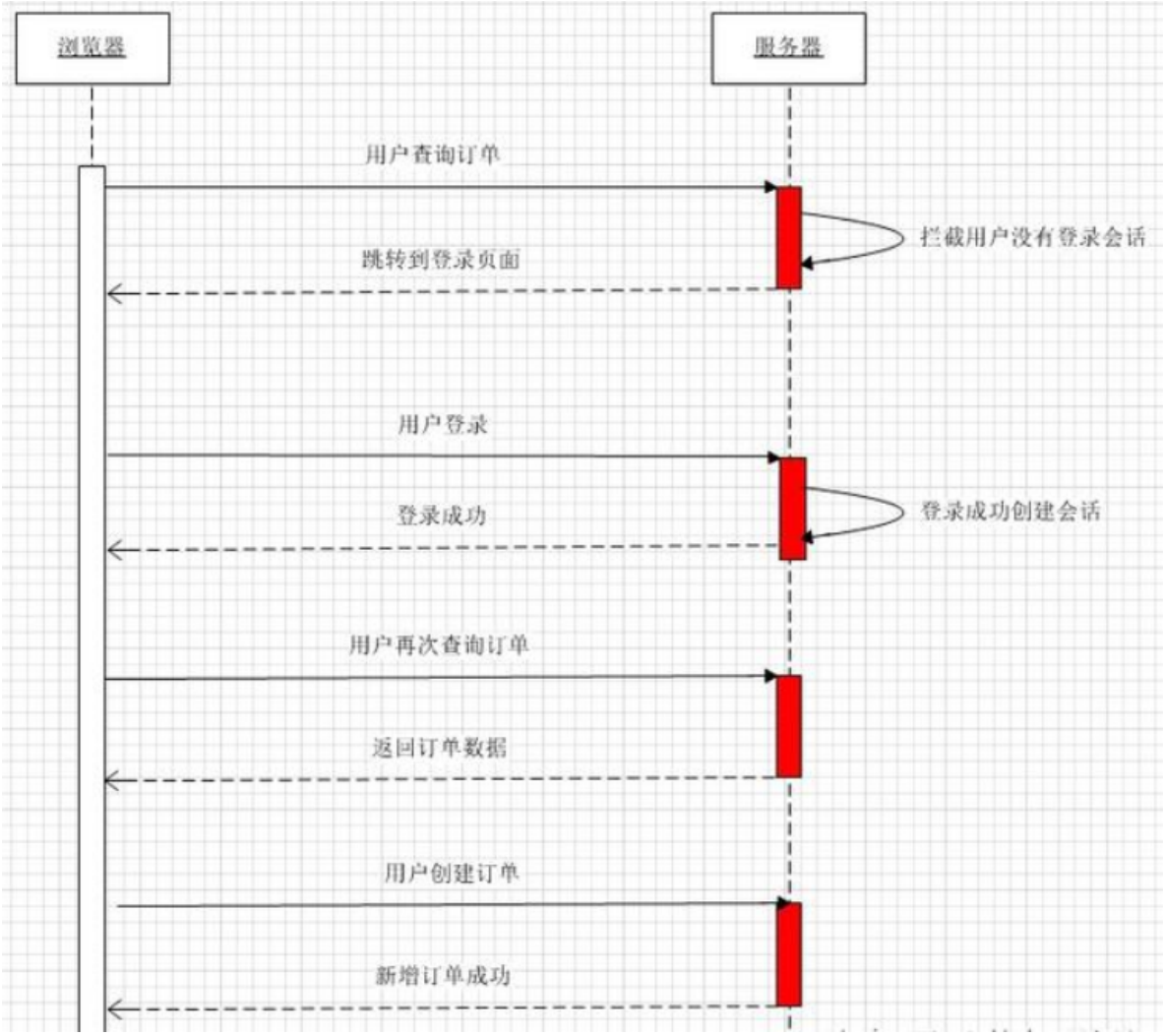
如果应用程序加入了单点登录协议，管理用户帐号的负担就会减轻。简化的程度取决于应用程序，因为 SSO 只处理身份验证。所以，应用程序可能仍然需要设置用户的属性（比如访问特权）。

单点登陆的来源

1.早期的单机部署：web单系统应用

早期我们开发web应用都是所有的包放在一起打成一个war包放入tomcat容器来运行的,所有的功能,所有的业务,后台管理,门户界面,都是由这一个war来支持的,这样的单应用,也称之为巨石应用,因为十分不好扩展和拆分。

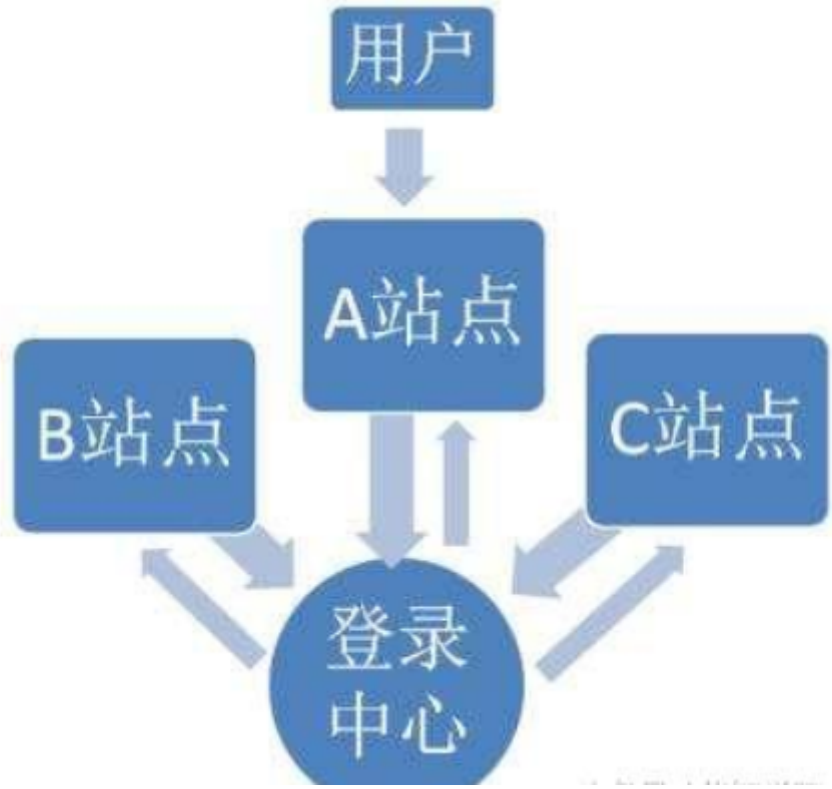
在巨石应用下,用户的登录以及权限就显得十分简单,用户登录成功后,把相关信息放入会话中,HTTP维护这个会话,再每次用户请求服务器的时候来验证这个会话即可,大致可以用下图来表示:



验证登录的这个会话就是session,维护了用户状态,也就是所谓的HTTP有状态协议,我们经常可以在浏览器中看到JSESSIONID,这个就是用来维持这个关系的key。

2.分布式集群部署

由于网站的访问量越来也大，单机部署已经是巨大瓶颈，所以才有了后来的分布式集群部署。例如：如果引入集群的概念,1单应用可能重新部署在3台tomcat以上服务器,使用nginx来实现反向代理, 此时,这个session就无法在这3台tomcat上共享,用户信息会丢失，所以不得不考虑多服务器之间的session同步问题，这就是单点登陆的来源。



2、单点登陆的来源

单点登录的实现方案，一般就包含：Cookies，Session同步，分布式Session方式，目前的大型网站都是采用分布式Session的方式。我先从cookie的实现谈起，你就能很清楚的知道为什么需要分布式session方式实现单点登录。

1.基于Cookie的单点登录

最简单的单点登录实现方式，是使用cookie作为媒介，存放用户凭证。 用户登录父应用之后，应用返回一个加密的cookie，当用户访问子应用的时候，携带上这个cookie，授权应用解密cookie并进行校验，校验通过则登录当前用户。

不难发现以上方式把信任存储在客户端的Cookie中，这种方式很容易令人质疑：

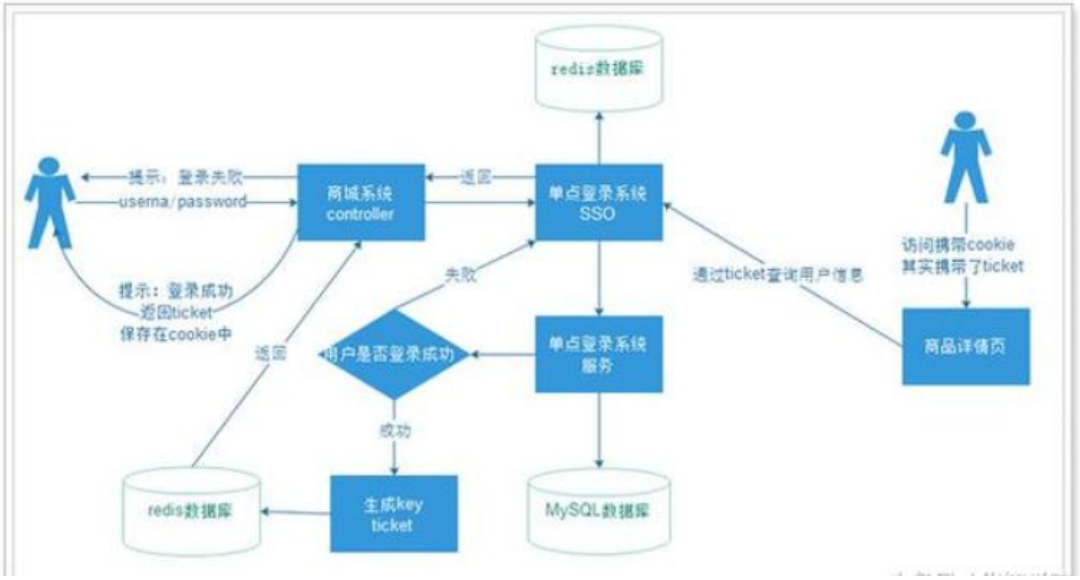
- Cookie不安全
- 不能跨域实现免登

对于第一个问题，通过加密Cookie可以保证安全性，当然这是在源代码不泄露的前提下。如果Cookie的加密算法泄露，攻击者通过伪造Cookie则可以伪造特定用户身份，这是很危险的。 对于第二个问题，不能跨域实现免登更是硬伤。所以，才有了以下的分布式session方案。

2.分布式session方式实现单点登录

例如阿里有很多系统分割为多个子系统，独立部署后，不可避免的会遇到会话管理的问题，类似这样的电商网站一般采用分布式Session实现。

再进一步可以根据分布式Session，建立完善的单点登录或账户管理系统。



流程运行：

（1） 用户第一次登录时，将会话信息（用户Id和用户信息），比如以用户Id为Key，写入分布式Session；

（2） 用户再次登录时，获取分布式Session，是否有会话信息，如果没有则调到登录页；

（3） 一般采用Cache中间件实现，建议使用Redis，因此它有持久化功能，方便分布式Session宕机后，可以从持久化存储中加载会话信息；

（4） 存入会话时，可以设置会话保持的时间，比如15分钟，超过后自动超时；

结合Cache中间件，实现的分布式Session，可以很好的模拟Session会话。

总结

以上分别从单点登录的原理、来源、实现机制来完整的解读单点登录。

实现单点登录说到底就是要解决如何产生和存储那个信任，再就是其他系统如何验证这个信任的有效性，因此要点也就以下两个：

- 存储信任
- 验证信任

存储信任建议可以采用分布式文件存储redis来实现。