链滴

java sso 单点登录 （方式之一 比较简单的方式）

作者：Jacker

原文链接：https://ld246.com/article/1504579268618

来源网站：链滴

关于单点登录（Single Sign On）

概念性的东西，谷歌百度上很多资料，我这里就不详细讲了，[附上百科链接](https://baike.baidu.com/item/SSO/3451380)，SSO的实现

网上也能找到一大堆资料。起初我是为了实现系统的保持登录状态，而了解到这个sso，单点登录应会是很多系统都必须用到的一个模块，

对于每一个程序员来说，这个是必须会的技能。单点登录也有很多层次的实现，我这里就简单讲一下个基于cookie的在同一个顶级域名下

子系统之间的sso实现原理。

关于cookie

cookie的定义，功能我在这里不在叙述，也可以在谷歌百度上找到相关资料。在这里我主要想提到cookie的存储，维基百科上提到：按在客

户端中的存储位置，可分为内存Cookie和硬盘Cookie。内存Cookie由浏览器维护，保存在内存中，览器关闭后就消失了，其存在时间是短

暂的。硬盘Cookie保存在硬盘里，有一个过期时间，除非用户手工清理或到了过期时间，硬盘Cooki不会被删除，其存在时间是长期的。所以，

按存在时间，可分为非持久Cookie和持久Cookie。

在本文提到的sso实现方式中，cookie是这个实现的桥梁，是唯一的验证token。所以cookie只要拿手，就等于有了权限，为了防止别人伪造，

这个有权限的cookie是需要加密的。而加密解密都需要在后台完成。

实现步骤

1、 用户登录，完成权限认证后，产生一个唯一的ssokey，String ssokey = des.encrypt(RANDOM.extInt(999999) + "|" + user_id + "|" + RANDOM.nextInt(999999));

参数解析：这个user_id 可以是其他，但是最好可以跟你的当前登录用户建立起联系。这个ssokey须是唯一的。

2、 把ssokey写入cookie，CookieUtil.set(response, domainName, cookieName, ssokey, keepLgin ? ssoCookieMaxage : -1, false);

参数解析：

domainName：共享cookie的域名，一般用顶级域名，比如我的 .huiyanxian.cn ,那么cookie在域名底下是共享的。

cookieName：cookie的名字，可以自定义，最好跟cookie的用途相关来命名

ssokey：唯一的token，校验的证书

keepLogin：顾名思义，是否保持登录，是的话，写入cookie存在时间ssoCookieMaxage，如果保持登录，则写入-1，cookie将在浏览器退出后删除。

false：是否加密，因为我在产生ssokey的时候已经加密过了，所以这里就不加密了 。

cookie每个人的封装习惯不一样，我只是按照我的个人封装来讲解，如果你需要封装，可以按照自的业务，或者代码习惯来。

3、 再一次登录，或者子系统sso登录，首先查看是否有登录的session，如果没有，在看看是否有ssoey 的cookie，如果没有，跳转去sso系统登录。

登录之后，ssokey的cookie就存在了，携带ssokey请求sso服务器api，sso服务器通过这个cookie来的ssokey，解密，拿到里面存在的user_id，

然后查询数据库取得登录的用户信息返回给子系统。 贴上子系统获取登录用户信息代码：

```
public static Object getLoginUser(HttpServletRequest request) {
    User user = null;
    try {
        String ssokey = CookieUtil.get(request, cookieName);
        if (StringUtils.isEmpty(ssokey)) {
            ssokey = (String) request.getAttribute("ssokey");
        }
        user = (User) request.getSession().getAttribute("loginUser");
        if (user != null) {
            return user;
        }
        if (!StringUtils.isEmpty(ssokey)) {
            String url = ssoRemoteUrl + "?ssokey=" + ssokey;
            String result = HttpUtils.get(url, "UTF-8");
            if (!StringUtils.isEmpty(result)) {
                ObjectMapper objectMapper = new ObjectMapper();
                user = objectMapper.readValue(result, User.class);
            }
        }
    } catch (Exception e) {
        log.error(e.getMessage(), e);
    }
    return user;
}
```

sso服务端获取登录用户信息代码：

```
@RequestMapping(value = "/sso/login/getLoginUser")
public @ResponseBody Object getLoginUser(HttpServletRequest request, HttpServletResponse response, @RequestParam String ssokey) {
    User user = null;
    try {
        if (!StringUtils.isEmpty(ssokey)) {
            int user_id = getUserId(ssokey);
            user = iUserService.selectById(user_id);
            if (user != null) {
                return user;
```

```
                }
            }
        } catch (Exception e) {
            log.error(e.getMessage(), e);
        }
        return user;
    }
```

sso登录完成。

总结

1、本应该附上一张流程图，无奈画图功底实在有限，就不丢出来了。本人文字表述或者实现方法可有缺陷，还请大家指出，实现方法不止一种，这只是本人

在实现后一个小总结，如若有不正确的地方，还望大家指正，共同学习交流，如果有建议，或者交流的同学，可以加我QQ：892662026，指出建议交流。

大家也可以使用第三方的开源sso插件，比如cas，比如kisso等等。

相关参考链接

[承一个人主页](http://www.cnblogs.com/ywlaker/p/6113927.html)