dubbo 注解和Shiro realm 不能注入service问题解决

曾令胜 2020-06-12 11:53:47 44 收藏

版权

一、背景：

项目 采用dubbo+shiro 进行权限管理，shiro放在消费者里面，在shiro 的realm中注入用户的service。采用这种方式，

public class UserRealm extends AuthorizingRealm {

@Reference

IUserService userService;

@Override

protected AuthorizationInfo doGetAuthorizationInfo(PrincipalCollection principalCollection) {

String username = (String) principalCollection.getPrimaryPrincipal();

SimpleAuthorizationInfo authorizationInfo = new SimpleAuthorizationInfo();

//在数据库中查询用户拥有的角色/权限

/\* authorizationInfo.setRoles(userService.findRoles(username));

authorizationInfo.setStringPermissions(userService.findPermissions(username));\*/

return authorizationInfo;

}

@Override

protected AuthenticationInfo doGetAuthenticationInfo(AuthenticationToken authenticationToken) throws AuthenticationException {

String username = (String) authenticationToken.getPrincipal();

/\* IUserService userService = (IUserService)SpringUtil.getBean("userService");\*/

User user= userService.findByUsername(username);

if(user == null){

throw new UnknownAccountException(); //没找到账号

}

if(Boolean.TRUE.equals(user.getLocked())){

throw new LockedAccountException(); //账号被锁定

}

SimpleAuthenticationInfo authenticationInfo = new SimpleAuthenticationInfo(username,user.getPassWord(),getName());

return authenticationInfo;

}

}

28

用refrence 注入，service为空，在网上查找原因为，AuthorizingRealm 创建在先，dubbo给属性赋值在后。所以获取不到。

二、解决办法：

将userService获取后交给spring容器管理。使用的时候从spring容器中获取。

1.编写从spring容器中获取bean工具类

@Component

public class SpringUtil implements ApplicationContextAware {

private static ApplicationContext applicationContext;

@Override

public void setApplicationContext(ApplicationContext applicationContext) throws BeansException {

if(SpringUtil.applicationContext == null) {

SpringUtil.applicationContext = applicationContext;

}

}

//获取applicationContext

public static ApplicationContext getApplicationContext() {

return applicationContext;

}

//通过name获取 Bean.

public static Object getBean(String name){

return getApplicationContext().getBean(name);

}

//通过class获取Bean.

public static <T> T getBean(Class<T> clazz){

return getApplicationContext().getBean(clazz);

}

//通过name,以及Clazz返回指定的Bean

public static <T> T getBean(String name,Class<T> clazz){

return getApplicationContext().getBean(name, clazz);

}

}

32

2.将userService通过dubbo获取后放入spring容器

@RestController

public class UserController {

@Reference

private IUserService userService;

@Bean

public IUserService userService(){

return userService;

}

}

9

3.改造UserRealm 为

public class UserRealm extends AuthorizingRealm {

@Override

protected AuthorizationInfo doGetAuthorizationInfo(PrincipalCollection principalCollection) {

String username = (String) principalCollection.getPrimaryPrincipal();

SimpleAuthorizationInfo authorizationInfo = new SimpleAuthorizationInfo();

//在数据库中查询用户拥有的角色/权限

/\* authorizationInfo.setRoles(userService.findRoles(username));

authorizationInfo.setStringPermissions(userService.findPermissions(username));\*/

return authorizationInfo;

}

@Override

protected AuthenticationInfo doGetAuthenticationInfo(AuthenticationToken authenticationToken) throws AuthenticationException {

String username = (String) authenticationToken.getPrincipal();

//从容器中获取

IUserService userService = (IUserService) SpringUtil.getBean("userService");

User user= userService.findByUsername(username);

if(user == null){

throw new UnknownAccountException(); //没找到账号

}

if(Boolean.TRUE.equals(user.getLocked())){

throw new LockedAccountException(); //账号被锁定

}

SimpleAuthenticationInfo authenticationInfo = new SimpleAuthenticationInfo(username,user.getPassWord(),getName());

return authenticationInfo;

}

}

————————————————

版权声明：本文为CSDN博主「曾令胜」的原创文章，遵循CC 4.0 BY-SA版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/qq\_21190847/java/article/details/106712807