DVAPI Vulnerability Assessment Report

# API8:2023 – Security Misconfiguration

## Objective

To demonstrate exploitation of the Security Misconfiguration vulnerability in DVAPI (API8:2023) using Postman and Burp Suite. This includes identifying improper JWT handling that allows privilege escalation and unauthorized access to sensitive data.

## Overview of DVAPI

DVAPI (Damn Vulnerable API) is a deliberately insecure API project used for training and assessing API security flaws. It replicates vulnerabilities listed in the OWASP API Security Top 10, allowing practical exploitation in a safe environment.

## Vulnerability Description

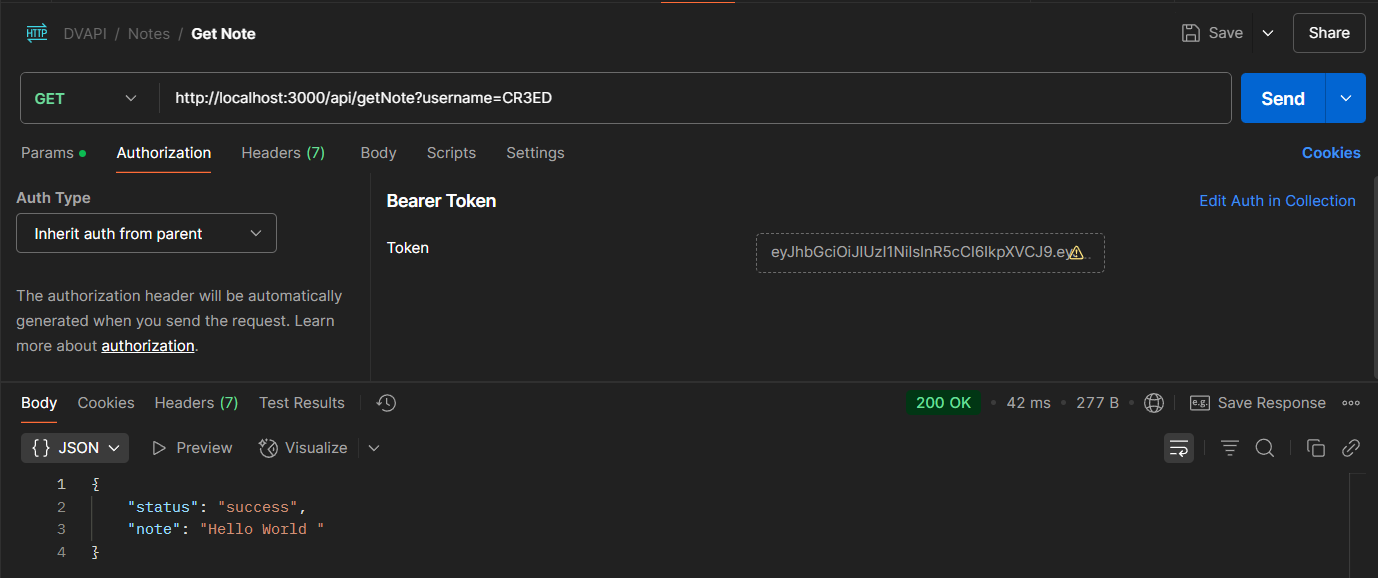
Security misconfiguration occurs when applications or APIs are insecurely configured, either by default or due to improper practices. In this case, JWT tokens were used without securely validating the signature or claims, allowing modification of user roles.

## Tools Used

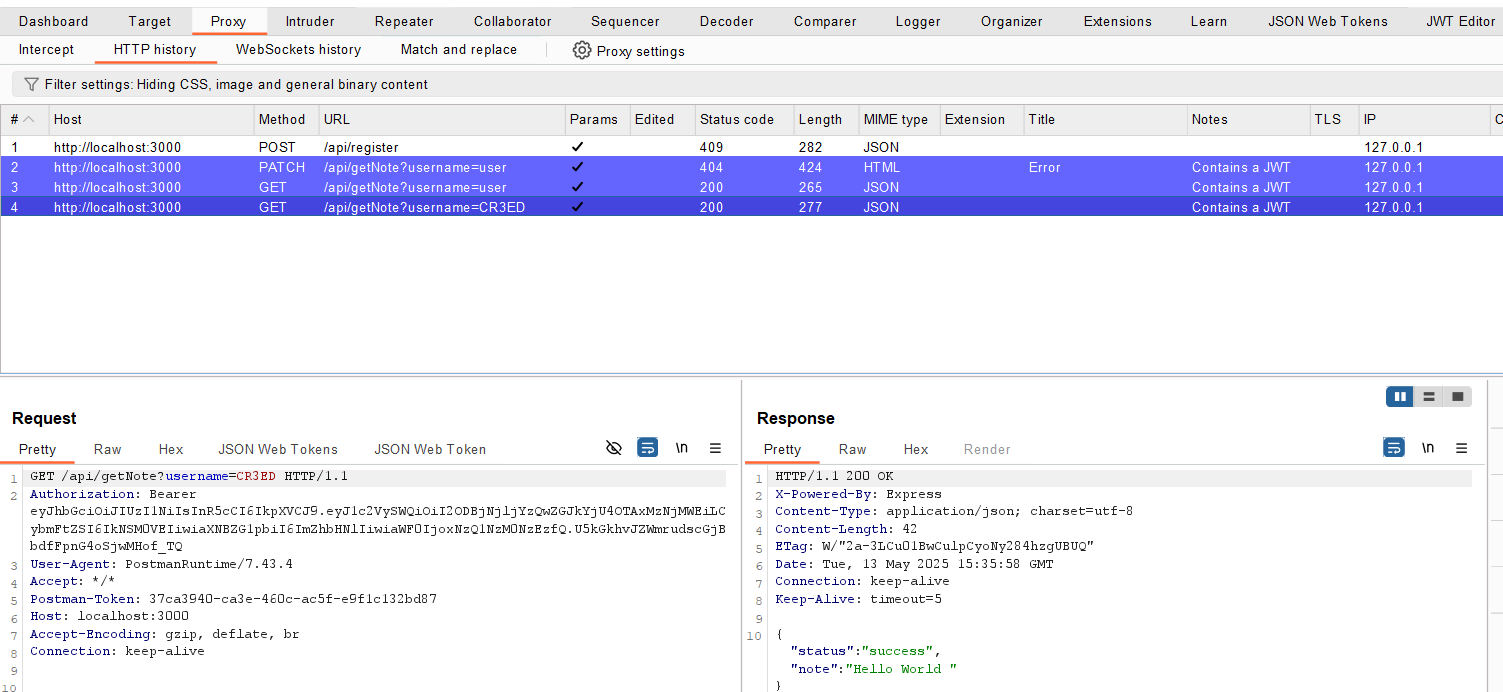
**• DVAPI:** Local instance at http://localhost:3000  
**• Postman:** Used to send and test API requests  
**• Burp Suite:** Used for intercepting, modifying, and replaying requests, including JWT token manipulation

## Steps to Reproduce

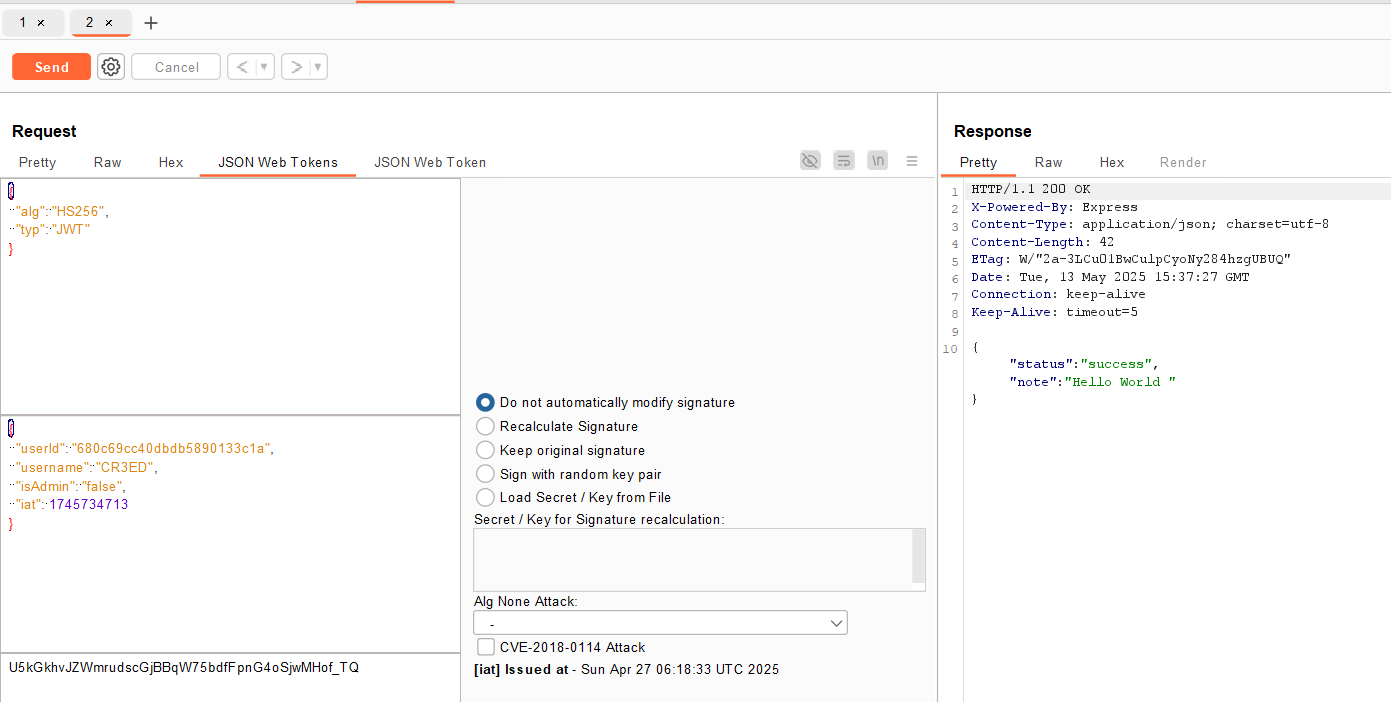
1. A request was made to the /api/getNote endpoint in Postman using a JWT token with user privileges.



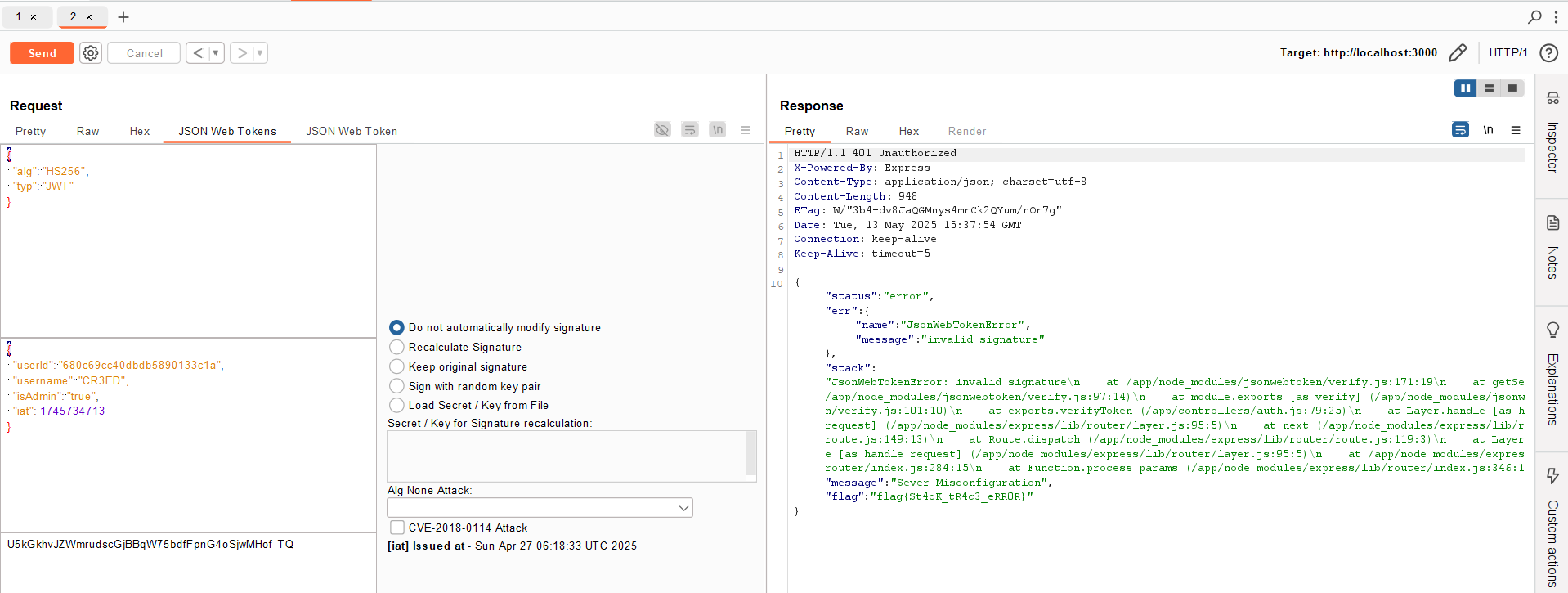
2. This request was intercepted in Burp Suite (HTTP History) and forwarded to the Repeater tab.



3. In Burp Suite’s JWT Editor, the payload of the token was inspected. The 'isAdmin' claim was originally set to false.



4. The 'isAdmin' claim was modified to true and the request was resent. The API responded with a flag, confirming unauthorized access due to misconfiguration.



## Impact

An attacker can escalate privileges by manipulating insecure JWT tokens, potentially gaining admin-level access. This could expose sensitive data, functionality, or full control of the application.

## Mitigation Strategy

• Validate JWT signatures using secure and secret keys.  
• Do not trust client-side JWT claims like isAdmin without verification.  
• Use signed and encrypted tokens when handling sensitive claims.  
• Regularly audit application configuration for default or weak setups.  
• Apply role-based access control (RBAC) checks server-side.