Method-based access control can be circumvented(access control)

Executive Summary

This report describes a critical vulnerability discovered in the access control implementation of the target application. The application relies on HTTP method-based controls to restrict access to administrative functionality. Through method tampering, a non-privileged user was able to bypass these restrictions and escalate their privileges to administrator. This vulnerability is categorized as **high severity** due to the ease of exploitation and the resulting impact on user roles and access control.

Introduction

The assessment aimed to test the robustness of access control mechanisms protecting administrative endpoints in the application. Particular attention was paid to the request methods and their influence on access permissions. The test identified a logic flaw that allowed privilege escalation using HTTP method manipulation.

Methodology

1. Used valid administrator and non-administrator accounts to observe normal and restricted behavior.

2. Captured requests made to the admin panel using Burp Suite.

3. Intercepted and modified HTTP methods to test access control weaknesses.

4. Attempted to replay requests with alternate methods and user sessions to validate unauthorized access.

Vulnerability Finding

- **Type**: Method-based Access Control Bypass

- **Location**: `/admin-roles` endpoint

- **Severity**: High

Description

The application enforced access control partially based on the HTTP method used in requests. While `POST` requests from non-admin users to the admin role assignment endpoint are blocked as unauthorized, the system fails to enforce the same controls when the request method is switched to `GET`.

This oversight allowed an attacker to bypass access restrictions and perform unauthorized administrative actions by manipulating the HTTP method.

Proof of Concept (PoC)

1. Log in as `administrator: admin` and access the admin panel.

2. Promote `user` and send the request to Burp Repeater:

   bash

   CopyEdit

   `POST /admin-roles?username=user&action=upgrade`

3. Log in as `user` (non-admin) in a private browser window.

4. Copy the non-admin user's session cookie into the captured request in Repeater.

5. Replay the original `POST` request – observe the "Unauthorized" response.

6. Change the request method to `POSTX` – response now shows "missing parameter", indicating the route is accessible but the method is invalid.

7. Convert the request to `GET` and adjust the URL to:

   pgsql

   CopyEdit

   `GET /admin-roles?username=user&action=upgrade`

8. Replay the request – notice that the non-admin user is now successfully promoted to admin.

Impact Assessment

This vulnerability allowed any authenticated user to:

- Bypass role-based access restrictions,

- Escalate privileges to administrator,

- Gain access to sensitive administrative functions,

- Perform unauthorized user and system management operations.

This constitutes a significant security risk to the confidentiality, integrity, and availability of the system.

Recommendations

1. Enforce server-side role checks regardless of HTTP request method.

2. Eliminate reliance on HTTP methods as part of the access control decision-making.

3. Use centralized middleware or access control layers to validate user roles before processing any admin-related request.

4. Regularly test access controls using both standard and non-standard HTTP methods.

Conclusion

This assessment highlighted a critical flaw in the application's access control model. The vulnerability enabled privilege escalation by manipulating the HTTP method, a common yet serious oversight. Immediate remediation and implementation of robust, method-independent access controls are strongly recommended to protect the application from further exploitation.