Access Control Vulnerability Assessment Methodology & Checklist

Eyad Islam El-Taher

Executive Summary

This document provides a comprehensive methodology for identifying and exploiting Access Control vulnerabilities based on extensive research and practical experience.

1 Phase 1: Reconnaissance & Application Mapping

1.1 Application Enumeration

- **Identify all endpoints:**
  - Administrative interfaces (`/admin`, `/dashboard`)
  - User management endpoints (`/users`, `/profile`)
  - API endpoints with role-based access
  - File upload/download functionality
- **Discover hidden functionality:**
  - Analyze `robots.txt` for disallowed paths
  - Check `sitemap.xml` for application structure
  - Review JavaScript files for hidden endpoints
  - Examine HTML comments for development notes
- **User role identification:**
  - Map available user roles (admin, user, moderator, etc.)
  - Identify privilege differences between roles
  - Document role-specific functionality

2 Phase 2: Authentication & Session Analysis

2.1 Session Management Testing

1. **Session token analysis:**
   - Check for session tokens in URLs
   - Analyze cookie structure and security flags
   - Test session fixation vulnerabilities
   - Verify session expiration mechanisms
2. **Authentication bypass testing:**

```
# Parameter manipulation
?admin=true
?is_admin=1
?role=admin

# Cookie manipulation
Cookie: user_role=admin; is_authenticated=true
```

3. **Privilege parameter testing:**
   - URL parameters
   - POST data parameters
   - HTTP headers
   - JSON/XML request bodies

3 Phase 3: Vertical Privilege Escalation Testing

3.1 Administrative Function Access

- **Direct URL access:**
  - Access admin panels as regular user
  - Bypass role-based restrictions
  - Test forced browsing techniques
- **Parameter-based escalation:**

```
# Basic privilege parameters
https://site.com/user/profile?user_id=123&admin=true

# Role manipulation
POST /update_profile
user_role=administrator&user_id=123

# Cookie privilege escalation
Cookie: role=admin; permissions=all
```

- **HTTP method manipulation:**
  - Change POST to GET/PUT/DELETE
  - Test method override headers
  - Verify consistent access controls

3.2 Advanced Vertical Escalation Vectors

- **IDOR to vertical escalation:**

```
# Access admin user through IDOR
/users/profile?id=1 (regular user)
/users/profile?id=0 (admin user)

# Modify admin user data
POST /users/update?id=0
email=attacker@evil.com&password=newpass
```

- **API endpoint testing:**
  - Test API version differences
  - Check mobile vs web API privileges
  - Verify GraphQL query access controls

4 Phase 4: Horizontal Privilege Escalation & IDOR

4.1 IDOR Testing Methodology

- **Object reference testing:**
  - Sequential IDs (1, 2, 3...)
  - UUIDs/GUIDs from other sources
  - Username/email enumeration
  - Predictable hash values
- **Common IDOR endpoints:**

```
/users/[ID]/profile
/orders/[ID]/details
/files/[ID]/download
/api/v1/users/[ID]
/admin/users/[ID]/edit
```

- **Testing techniques:**
  - Increment/decrement numeric IDs
  - Change GUIDs to other users'
  - Test different object types
  - Verify response differences

4.2 Advanced IDOR Scenarios

- **Mass IDOR exploitation:**

```
# Batch user data extraction
for i in {1..100}; do
    curl "https://site.com/users/$i/profile"
done

# API mass enumeration
GET /api/users/1
GET /api/users/2
GET /api/users/3
```

- **IDOR in different HTTP methods:**
  - GET for information disclosure
  - POST/PUT for data modification
  - DELETE for object removal
  - PATCH for partial updates
- **Blind IDOR detection:**
  - Timing differences in responses
  - Error message variations
  - Response length analysis
  - Behavioral changes detection

5 Phase 5: Platform & Configuration Bypasses

5.1 HTTP Header Manipulation

- **URL override headers:**

```
# Bypass front-end controls
GET / HTTP/1.1
X-Original-URL: /admin
X-Rewrite-URL: /admin

# Method override headers
X-HTTP-Method-Override: DELETE
_method: PUT
```

- **Referer-based bypass:**

```
GET /admin/deleteUser HTTP/1.1
Referer: https://site.com/admin

# Forged Referer for access
Referer: https://site.com/approved-page
```

5.2 Web Server Misconfigurations

- **Directory traversal testing:**

```
/admin/../admin
/admin/./admin
/admin//admin
/admin/%2e%2e/admin
```

- **HTTP verb tampering:**
  - Test all HTTP methods on endpoints
  - Check for HEAD/OPTIONS information leaks
  - Verify TRACE/CONNECT accessibility
- **Case sensitivity bypass:**

```
/Admin
/ADMIN
/aDmIn
/admin/
/admin/.
```

6 Phase 6: Multi-Step Process Testing

6.1 Workflow Bypass Assessment

- **Step skipping detection:**
  - Access final steps directly
  - Modify step sequence parameters
  - Bypass validation steps
- **State parameter manipulation:**

```
# Payment process bypass
Step 1: /checkout/cart (controlled)
Step 2: /checkout/shipping (controlled)
Step 3: /checkout/payment (controlled)
Step 4: /checkout/confirm (UNPROTECTED!)

# Direct access to final step
POST /checkout/confirm
order_id=123&payment_confirmed=true
```

6.2 Session State Testing

- **Session variable manipulation:**
  - Modify session storage values
  - Tamper with session cookies
  - Manipulate local storage data
- **CSRF token bypass:**
  - Remove CSRF tokens from requests
  - Use valid tokens from other sessions
  - Predict CSRF token generation

7 Phase 7: Business Logic & Context Testing

7.1 Business Logic Flaws

- **Price manipulation:**

```
# Hidden price parameters
POST /checkout
product_id=123&quantity=1&price=0.01

# Discount abuse
coupon_code=100PERCENTOFF
discount_amount=9999
```

- **Quantity manipulation:**
  - Negative quantities
  - Extremely large quantities
  - Decimal quantity values

7.2 Context-Based Access Testing

- **Time-based restrictions:**
  - Access expired content
  - Modify time-limited offers
  - Bypass maintenance mode
- **Geographic restrictions:**
  - IP spoofing headers (X-Forwarded-For, CF-Connecting-IP)
  - VPN/proxy testing
  - Language/country parameter manipulation
- **User state testing:**
  - Access features for unverified users
  - Bypass email verification
  - Access premium features as free user

8 Phase 8: Automated Testing & Tooling

8.1 Automated Scanning

- **Burp Suite extensions:**
  - Autorize - automatic privilege escalation
  - Authz - authorization testing
  - Access Control Tester
- **Custom testing scripts:**

```
# IDOR testing with Python
import requests
for user_id in range(1, 100):
    r = requests.get(f'https://site.com/users/{user_id}')
    if r.status_code == 200:
        print(f"Found accessible user: {user_id}")
```

8.2 Manual Testing Checklist

- **Privilege matrix testing:**
  - Test all user roles against all endpoints
  - Verify consistent access controls
  - Check for privilege creep
- **Parameter fuzzing:**
  - Role/privilege parameters
  - User ID/Object ID parameters
  - Status/state parameters
- **Response analysis:**
  - Compare responses between roles
  - Check for information leakage
  - Analyze error messages

9 Phase 9: Impact Assessment & Exploitation

9.1 Risk Classification

- **Critical impact:**
  - Full administrative access
  - Database compromise
  - Financial fraud capability
  - User data mass extraction
- **High impact:**
  - Other user account takeover
  - Sensitive data access
  - Privilege escalation to mid-level roles
- **Medium impact:**
  - Limited information disclosure
  - Partial privilege escalation
  - Business logic bypass

9.2 Proof of Concept Development

- **Reproducible test cases:**
  - Step-by-step exploitation guide
  - Required user roles and permissions
  - Expected vs actual results
- **Evidence collection:**
  - Screenshots of unauthorized access
  - HTTP request/response logs
  - Database extracts (if authorized)

Conclusion

This methodology provides a systematic approach to access control vulnerability assessment, covering from basic reconnaissance to advanced exploitation techniques. The structured approach ensures comprehensive testing while helping identify complex vulnerability chains and privilege escalation paths.