

# **COMP – 357**

# **MITIGATION REPORT**

Submitted by: Sreehari Jiji

Student id: 10316483

# Exercise 1 — OWASP Juice Shop

## Mitigation Recommendations

### Mitigation for Attack 1: DOM-Based XSS

#### -Summary of Attack

A DOM-based XSS condition was demonstrated through the search feature, where client-side handling of user input allowed injected script to execute in the browser context.

#### -Root Cause

Client-side input was processed and rendered without adequate sanitization and output encoding.

#### -Recommended Controls

- Apply output encoding for any user-supplied content rendered into the DOM.
- Enforce strict input validation on both client and server.
- Implement a strong Content Security Policy (CSP) to reduce script execution risk.
- Avoid unsafe DOM patterns.
- Use security headers consistently across the application.

#### -Expected Outcome

In a production-equivalent system with these controls, injected script content would be treated as text or rejected, preventing browser execution.

## **Mitigation for Attack 2: Broken Access Control**

### **-Summary of Attack**

An IDOR-style access control weakness was demonstrated by manipulating the client-exposed basket identifier (bid) to change the basket context beyond intended authorization.

### **-Root Cause**

Insufficient server-side authorization enforcement on object access tied to authenticated users.

### **-Recommended Controls**

- Enforce server-side ownership checks on every request involving user-scoped objects (basket, orders, profile).
- Bind resources to the authenticated identity and validate ownership before read/write operations.
- Use non-guessable identifiers instead of sequential numeric IDs.
- Implement access control tests as part of CI/CD.
- Add logging and alerting for abnormal access patterns.

### **-Expected Outcome**

Changing client-visible identifiers would not grant access to other users' resources, and unauthorized access attempts would be denied and logged.

## **Mitigation for Attack 3: Broken Authentication**

### **-Summary of Attack**

A login bypass was demonstrated using injection-style input in the email field, resulting in unauthorized administrative-level access in the training context.

### **-Root Cause**

Authentication logic tolerated crafted input, indicating insufficient validation or unsafe query construction.

### **-Recommended Controls**

- Use parameterized queries / prepared statements for all authentication-related database operations.
- Apply strict server-side input validation for identity fields.
- Implement rate limiting and account lockout policies for repeated failed attempts.
- Require multi-factor authentication (MFA) for privileged roles in production.
- Centralize authentication logs and monitor for anomalous login patterns.

### **-Expected Outcome**

Injection-style input would fail authentication, and privileged access would require legitimate credentials and additional verification.

# Exercise 2 — Kerberoasting

## Mitigation Recommendations

### Mitigation for Attack: Kerberoasting – Service Account Credential Recovery

#### Summary of Attack

A Kerberoasting demonstration successfully recovered the lab service account password for LAB\svcweb by requesting an SPN-associated service ticket for HTTP/webapp.lab.local and performing controlled offline password cracking. This confirms that SPN-linked service accounts become high-risk targets when weak or long-lived credentials are used.

#### Mitigation Options

Kerberoasting risk can be reduced through layered identity, configuration, and monitoring controls. In production environments, recommended mitigations include:

1. Strong, long, and regularly rotated service account passwords  
Use high-entropy passwords (preferably random, long length) to make offline cracking impractical.
2. Use Group Managed Service Accounts (gMSA) where possible  
gMSAs provide automatic password management and reduce the risk of weak or static service account credentials.
3. Least privilege for service accounts  
Ensure service accounts do not hold unnecessary permissions or membership in privileged groups.
4. Limit service account sprawl and enforce governance  
Maintain an accurate inventory of SPNs and service accounts, remove unused SPNs, and enforce ownership and review cycles.
5. Monitor Kerberos and service ticket activity  
Detect abnormal spikes in TGS requests or unusual patterns from non-administrative users.

6. Separate high-risk services and restrict lateral pathways  
Segment sensitive resources and reduce the blast radius of any single credential compromise.

### **Mitigation Demonstrated in This Lab**

While multiple mitigations are applicable, this exercise demonstrated one practical control for validation:

resetting the service account password to a strong, complex value.

The SPN configuration (HTTP/webapp.lab.local) was intentionally left unchanged so the validation specifically measured the impact of credential strength on Kerberoasting feasibility.

### **Validation Method**

After the new service account password was applied, a fresh service ticket for the SPN-associated account was requested and extracted. The updated ticket hash was tested using the same offline cracking approach used in the baseline demonstration to confirm whether the attack remained practical.

### **Validation Result**

After the new password was applied, John the Ripper did not recover the updated service account password within the lab test scope. This demonstrates that improving service account password strength significantly reduces Kerberoasting effectiveness in real-world equivalents.

### **Evidence**

- SPN verification after new password — output confirming HTTP/webapp.lab.local remains registered to LAB\svcweb.

```

Administrator: Command Prompt
C:\>Restart-Service DNS
'Restart-Service' is not recognized as an internal or external command,
operable program or batch file.

C:\>setspn -L LAB\svcweb
Registered ServicePrincipalNames for CN=Sevice web,OU=LAB USERS,DC=lab,DC=local:
    HTTP/webapp.lab.local

C:\>_

```

- Updated service ticket hash after new password — extraction output showing a new TGS hash for the hardened account.

```

Session Actions Edit View Help
(kali㉿kali)-[~]
$ GetUserSPNs.py lab.local/user1:P@ssw0rd -dc-ip 192.168.45.160 -request
Impacket v0.13.0 - Copyright Fortra, LLC and its affiliated companies

ServicePrincipalName      Name      MemberOf      PasswordLastSet      LastLogon      Deleg
HTTP/webapp.lab.local    svcweb          2025-12-07 06:14:48.248948  2025-12-07 05:05:31.317653

[-] CCache file is not found. Skipping ...
$krb5tgs$23$*svcweb$LAB.LOCAL$lab.local/svcweb*$a899c1cb86b1c9ba19676541f64a5ffe$686c7bc786b9725ea972c
1e108ee041851b88bccbea1731e4c51ec5e630d551724d973fc87b9a5642dc61f3fd4946b024fffb2fd69fe1b5fe2c6145fd2
734278d39b15247d494d3655b20275c250a269a9613076e876a7eb73eb901f3a695c2ed208fc253d4c59750be49822ea2826d2
279a21c2d8252a3e9ff23f0967b6bad668786b3264f3e41159b3fe35ca569f19df5c231f27ac64ffd7fa87f7b425bcdfa14e4
0807217a760de0a22ec2521b29b14f3d8ef245a74cf7022675d97014d1bae992a8e541421db26b21c937c330b2478ba7b1e576
2cae1f0a45a280cd507bc33e41d08064e25279dafcb9fae9b84ba1cc9bd6f5233fbf68bcba1164c8144ad87f7a7762e0de6156
09d3c9edea921beb2592e729517fa5f7756e8c38d79554f951a77d80e8a1286bef9e572662e8a18a513fdc792d04a7442b581
d4ef03207ab62be428beb8415e6dd658f1a93709da95206481fd3777a490ff41b4cc3f137f4298fedcf743664e484696a7a506
4cb77cb2914831430927fc9f2bf8a23d6e4466992e6582874cae83abfc55dc862780e51e3ee368518f82d1c9f8131a668a4834
4e0f5c24b1876c1b83bd712b75b1ba8a7f2b5d

(kali㉿kali)-[~]
$ nano newhash.txt

```

- John the Ripper output after new password — confirms the updated password was not recovered during the validation attempt.

```
kali@kali: ~
Session Actions Edit View Help
[(kali㉿kali)-[~]] $ GetUserSPNs.py lab.local/user1:P@ssw0rd -dc-ip 192.168.45.160 -request
Impacket v0.13.0 - Copyright Fortra, LLC and its affiliated companies

ServicePrincipalName      Name      MemberOf      PasswordLastSet      LastLogon      Deleg
HTTP/webapp.lab.local    svcweb          2025-12-07 06:14:48.248948  2025-12-07 05:05:31.317653

[-] CCache file is not found. Skipping...
$krb5tgs$23$*svcweb$LAB.LOCAL$lab.local/svcweb*a899c1cb86b1c9ba19676541f64a5ffe$686c7bc786b9725ea972c
1e108ee041851b88bccbea1731e4c51ec5e630d551724d973fc87b9a5642dc61f3fd4946b024ffb2fd69fe1b5fe2c6145fd2
734278d39b15247d494d3655b20275c250a269a9613076e876a7eb73eb901f3a695c2ed208fc253d4c59750be49822ea2826d2
279a21c2d8252a3e9ff23f0967b6bad68786b3264f3e41159b3fe35ca569f19df5c231f27ac64ffd7fa87f7b425bcdfa14e4
0807217a760de0a22ec2521b29b14f3d8ef245a74cf7022675d97014d1ba6992a8e541421db26b21c937c330b2478ba7b1e576
2cae1f0a45a280cd507bc33e41d08064e25279dafcb9fae9b84ba1cc9bd6f5233fb68bcb1164c8144ad87f7a7762e0de6156
09d3c9edea921beb2592e729517fa5f7756e8c38d79554f951a77d80e8a1286bef9e572662e8a18a513fdc792d04a7442b581
d4ef03207ab62be428beb8415e6dd658f1a93709da95206481fd3777a490ff41b4cc3f137f4298fedcf743664e484696a7a506
4cb77cb2914831430927fc9f2bf8a23d6e4466992e6582874cae83abfc55dc862780e51e3ee368518f82d1c9f8131a668a4834
4e0f5c24b1876c1b83bd712b75b1ba8a7f2b5d

[(kali㉿kali)-[~]] $ nano newhash.txt

[(kali㉿kali)-[~]] $ john newhash.txt --wordlist=/usr/share/wordlists/rockyou.txt --format=krb5tgs
Using default input encoding: UTF-8
Loaded 1 password hash (Kerberos 5 TGS etype 23 [MD4 HMAC-MD5 RC4])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:12 DONE (2025-12-07 06:16) 0g/s 1172Kp/s 1172Kc/s 1172KC/s !! 12Honey ..*7;Vamos!
Session completed.

[(kali㉿kali)-[~]] $
```