



External Penetration Test

**Report of Findings**

**Inlanefreight Ltd.**

June 13, 2024

Version 1.0

Table of Contents

[Statement of Confidentiality 3](#_Toc106718117)

[Engagement Contacts 4](#_Toc106718118)

[Executive Summary 5](#_Toc106718119)

[Approach 5](#_Toc106718120)

[Scope 6](#_Toc106718121)

[Assessment Overview and Recommendations 6](#_Toc106718122)

[Network Penetration Test Assessment Summary 8](#_Toc106718123)

[Summary of Findings 8](#_Toc106718124)

[Internal Network Compromise Walkthrough 9](#_Toc106718125)

[Detailed Walkthrough 9](#_Toc106718126)

[Remediation Summary 17](#_Toc106718127)

[Short Term 17](#_Toc106718128)

[Medium Term 17](#_Toc106718129)

[Long Term 17](#_Toc106718130)

[Technical Findings Details 18](#_Toc106718131)

[Appendices 32](#_Toc106718132)

[Appendix A – Finding Severities 32](#_Toc106718133)

[Appendix B – Exploited Hosts 33](#_Toc106718134)

[Appendix C – Compromised Users 34](#_Toc106718135)

[Appendix D – Changes/Host Cleanup 35](#_Toc106718136)

[Appendix E – INLANEFREIGHT.LOCAL Domain Password Review 36](#_Toc106718137)

# Statement of Confidentiality

This report contains confidential and proprietary information of Inlane Freight. The contents of this document are intended solely for the internal use of Inlane Freight and may contain privileged and sensitive information. Unauthorized distribution, copying, or disclosure of this document, in whole or in part, to any third party without the prior written consent of both Unity Solutions and Inlane Freight is strictly prohibited.

All findings, recommendations, and data contained in this report are provided exclusively for the benefit of Inlane Freight and are based on the specific scope, methodologies, and timeframes defined in the engagement agreement. The information should not be used or relied upon for any other purpose or by any other party without proper authorization.

If you have received this report in error, please notify Unity Solutions immediately and destroy all copies of the document in your possession.

Unity Solutions takes the security and confidentiality of client information very seriously and adheres to stringent protocols to ensure the protection of sensitive data. We appreciate Inlane Freight 's trust in our services and are committed to maintaining the highest standards of confidentiality and integrity.

# Engagement Contacts

|  |  |  |
| --- | --- | --- |
| Inlanefreight Contacts | | |
| **Primary Contact** | **Title** | **Primary Contact Email** |
| Name | Chief Executive Officer | Email@gmail.com |
| **Secondary Contact** | **Title** | **Secondary Contact Email** |
| Name | Chief Technical Officer | Email@gmail.com |

|  |  |  |
| --- | --- | --- |
| Assessor Contact | | |
| **Assessor Name** | **Title** | **Assessor Contact Email** |
| Moghees Ahmad | Security Consultant | Test@gmail.com |

# Executive Summary

Inlane freight Ltd. contracted Unity Solutions to perform a External Penetration Test of Inlane freight’s external facing network to identify security weaknesses, determine the impact to Inlanefreight, document all findings in a clear and repeatable manner, and provide remediation recommendations.

Approach

Unity Solutions performed testing under a “black box” approach June 1, 2024, to June 6, 2024 without credentials or any advance knowledge of Inlane freight’s externally facing environment with the goal of identifying unknown weaknesses. Testing was performed from an evasive standpoint with the goal of uncovering as many misconfigurations and vulnerabilities as possible. Testing was performed remotely via a host that was provisioned specifically for this assessment. Each weakness identified was documented and manually investigated to determine exploitation possibilities and escalation potential. Unity Solutions sought to demonstrate the full impact of every vulnerability, up to and including external facing hosts and internal domain compromise. If Unity Solutions were able to gain a foothold in the internal network, Inlane freight allowed for further testing including lateral movement and horizontal/vertical privilege escalation to demonstrate the impact of external and internal network compromise.

Scope

The scope of this assessment was one internal network range and the INLANEFREIGHT.LOCAL Active Directory domain.

In-Scope Assets

|  |  |
| --- | --- |
| Host/URL/IP Address | Description |
| 10.10.10.100 | External facing target |
| \*.inlanefrieght.local | All subdomains |

Table 1: Scope Details

Assessment Overview and Recommendations

During the external penetration test against Inlanefreight, Unity Solutions identified twelve(13) findings that threaten the confidentiality, integrity, and availability of Inlanefreight’s information systems. The findings were categorized by severity level, with seven (7) of the findings being assigned a high-risk rating, three (3) medium-risk, and one (1) low risk. There was also two (2) informational finding related to enhancing security monitoring capabilities within the internal network.

The tester found Inlanefreight’s patch and vulnerability management to be well-maintained. None of the findings in this report were related to missing operating system or third-party patches of known vulnerabilities in services and applications that could result in unauthorized access and system compromise. Each flaw discovered during testing was related to a misconfiguration or lack of hardening, with most falling under the categories of weak authentication and weak authorization.

One finding involved the FTP service allowing anonymous logins. This vulnerability can enable unauthorized users to access the FTP server, potentially leading to data exposure or modification. In most corporate environments, anonymous FTP access is unnecessary and should be disabled. During the assessment, it was noted that authenticated access could be implemented instead, ensuring only authorized users can access FTP resources. Inlanefreight should begin formulating a plan to disable anonymous FTP access and enforce strong authentication mechanisms.

Several instances of default credentials were found during the penetration test. Default credentials are commonly known and can be easily exploited by attackers to gain unauthorized access. Inlanefreight should replace all default credentials with strong, unique passwords and enforce a policy that prevents the use of default credentials in the future. An insecure file upload functionality was found that allows for remote code execution (RCE). This vulnerability enables attackers to upload malicious files that can be executed on the server, leading to full system compromise. To mitigate this risk, Inlanefreight should implement strict validation and sanitization of file uploads, restrict allowed file types, and regularly update and patch the file upload component to prevent exploitation.

The assessment identified a Local File Inclusion (LFI) vulnerability caused by an outdated WordPress plugin. This flaw can allow attackers to include and execute files from the server, potentially exposing sensitive information. It is recommended that Inlanefreight immediately update or replace the vulnerable plugin and conduct regular audits and updates of all WordPress components to ensure they are secure.

HTTP Verb Tampering was detected, allowing attackers to bypass security controls by using unexpected HTTP methods. Inlanefreight should restrict and validate HTTP methods allowed by the application server and implement security controls to handle and log unexpected HTTP verbs. The assessment uncovered an XML External Entity (XXE) injection vulnerability. XXE can be exploited to read local files, execute remote code, or perform Denial of Service (DoS) attacks. To mitigate this, Inlanefreight should disable XML external entity processing in all XML parsers and use secure libraries and frameworks that protect against XXE attacks.

An error-based SQL injection vulnerability was found, which could allow attackers to execute arbitrary SQL commands and access sensitive database information. Inlanefreight should use parameterized queries and prepared statements to prevent SQL injection and regularly review and test application code for vulnerabilities.

HTTP Verb Tampering was detected, allowing attackers to bypass security controls by using unexpected HTTP methods. Inlanefreight should restrict and validate HTTP methods allowed by the application server and implement security controls to handle and log unexpected HTTP verbs.

An Insecure Direct Object Reference (IDOR) vulnerability was found, enabling attackers to access unauthorized objects by manipulating reference values. Inlanefreight should implement proper authorization checks before granting access to objects and use indirect reference maps to prevent exposure of internal object references.

The assessment revealed a weak password policy and lack of brute force prevention measures. Inlanefreight should enforce a strong password policy with complexity requirements and implement account lockout mechanisms and rate limiting to prevent brute force attacks.

The SMTP server was found to have the VRFY command enabled for anonymous users, potentially exposing valid email addresses. Inlanefreight should disable the VRFY command in the SMTP server configuration to prevent information disclosure and ensure the SMTP server is properly secured.

RPC information disclosure was identified, which could be leveraged by attackers to gain insights into network services and users. Inlanefreight should restrict RPC services to trusted networks and authenticated users, and disable unnecessary RPC services to reduce the attack surface.

A subdomain name was discovered in a public GitLab repository, which could be used by attackers for reconnaissance and targeted attacks. Inlanefreight should regularly audit code repositories to ensure sensitive information is not exposed and remove any references to internal subdomains from public repositories.

By addressing these vulnerabilities promptly, Inlanefreight can significantly enhance its security posture and reduce the risk of potential attacks. Continuous monitoring, regular security assessments, and prompt patching of discovered vulnerabilities are essential practices to maintain a robust security framework.

# Web Server Penetration Test Assessment Summary

Unity Solutions began all testing activities from the perspective of an unauthenticated user from outside of the network. Inlanefreight did not provided the tester with any additional information.

Summary of Findings

During the course of testing, Unity Solutions uncovered findings that pose a material risk to Inlanefreight’s information systems. Unity Solutions also identified one informational finding that, if addressed, could further strengthen Inlanefreight’s overall security posture. Informational findings are observations for areas of improvement by the organization and do not represent security vulnerabilities on their own. The below table provides a summary of the findings by severity level.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | | | |
| **High** | **Medium** | **Low** | **Info** | **Total** |
| **7** | **3** | **1** | **2** | **13** |

Table 2: Severity Summary

Below is a high-level overview of each finding identified during testing. These findings are covered in depth in the [Technical Findings Details](#_Technical_Findings_Details) section of this report.

|  |  |  |
| --- | --- | --- |
| Finding # | Severity Level | Finding Name |
| 1. | **High** | FTP Anonymous Login |
| 2. | **High** | RCE through Insecure File Upload |
| 3. | **High** | Local File Inclusion (LFI) through wordpress outdated plugin |
| 4. | **High** | Login Using Default Credentials |
| 5. | **High** | XML External Entity Injection (XXE) |
| 6. | **High** | Error Based SQL Injection |
| 7. | **High** | OS Command Injection |
| 8. | **Medium** | HTTP Verb Tampering |
| 9. | **Medium** | Insecure Direct Object Reference (IDOR) |
| 10. | **Medium** | Weak Password Policy and No Brute Forcing Prevention |
| 11. | **Low** | SMTP VRFY command enabled to anonymous user |
| 12. | **Info** | RPC Information |
| 13. | **Info** | Another subdomain name found on GitLab |

Table 3: Finding List

# Web Server Compromise Walkthrough

During the course of the assessment Unity Solutions was able gain a foothold on the web server which was connected to internal network of Inlanefreight. The steps below demonstrate the steps taken to compromise and does not include all vulnerabilities and misconfigurations discovered during the course of testing. Any issues not used as part of the path to compromise are listed as separate, standalone issues in the [Technical Findings Details](#_Technical_Findings_Details_1) section, ranked by severity level. The intent of this attack chain is to demonstrate to Inlanefreight the impact of each vulnerability shown in this report and how they fit together to demonstrate the overall risk to the client environment and help to prioritize remediation efforts. While other findings shown in this report could be leveraged to gain a similar level of access, this attack chain shows the initial path of least resistance taken by the tester to achieve domain compromise.

Detailed Walkthrough

Unity Solutions performed the following to fully compromise the INLANEFREIGHT.LOCAL web server.

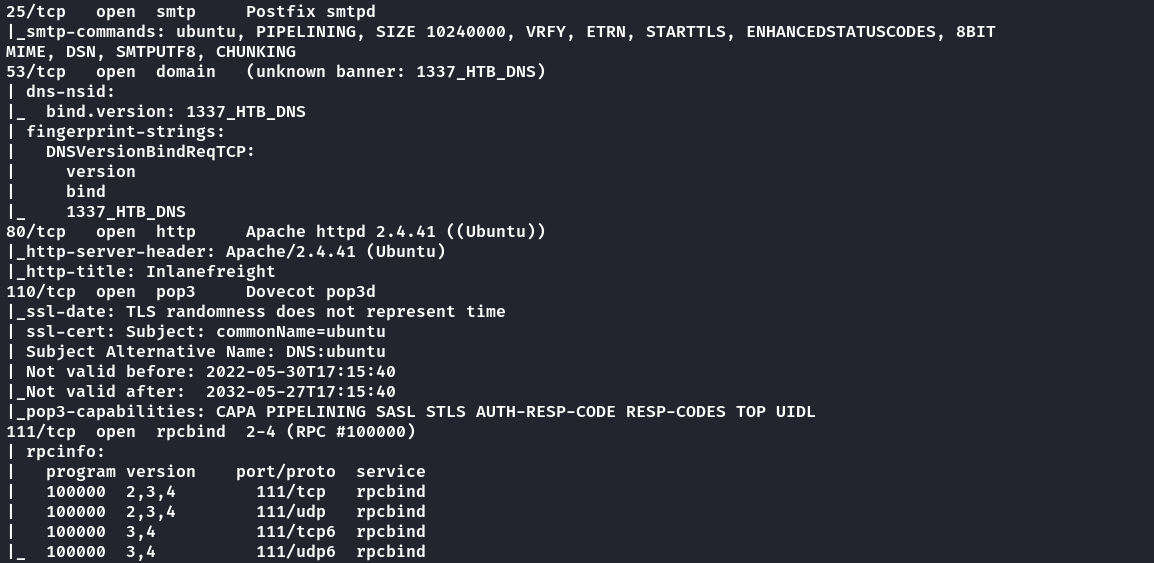
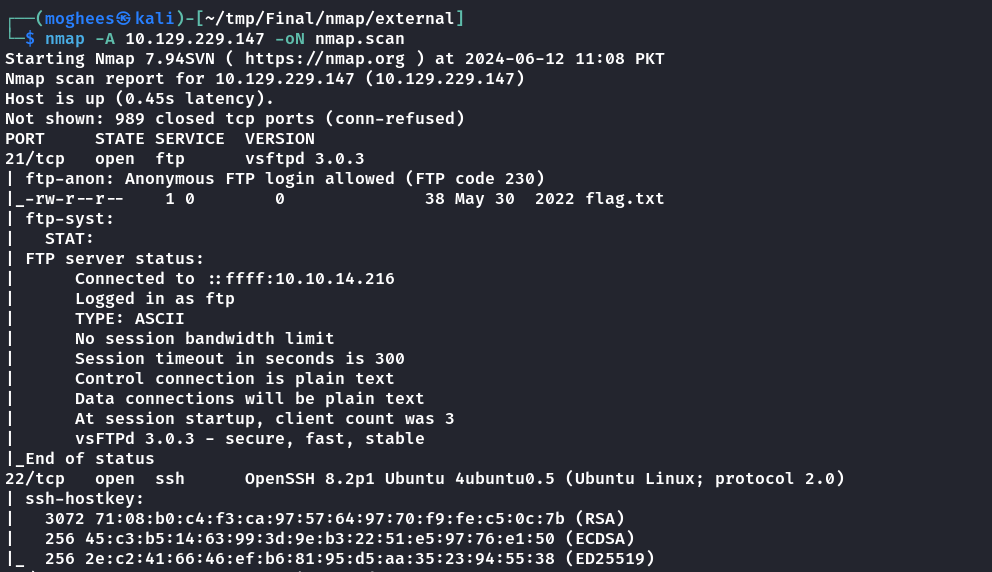
1. The tester utilized **nmap** for scanning the open ports and services running on the server.
2. Then FTP service was enumerated and attacker was able to access the private files through **anonymous** login.
3. SMTP service allowed pentester to verfiy valid users using **VRFY** command.
4. Then pentester was able to find a list of subdomains thorugh **dig tool**  and a VHOST through using a wordlist. One subdomain was found while testing gitlab.inlanefreight.local.

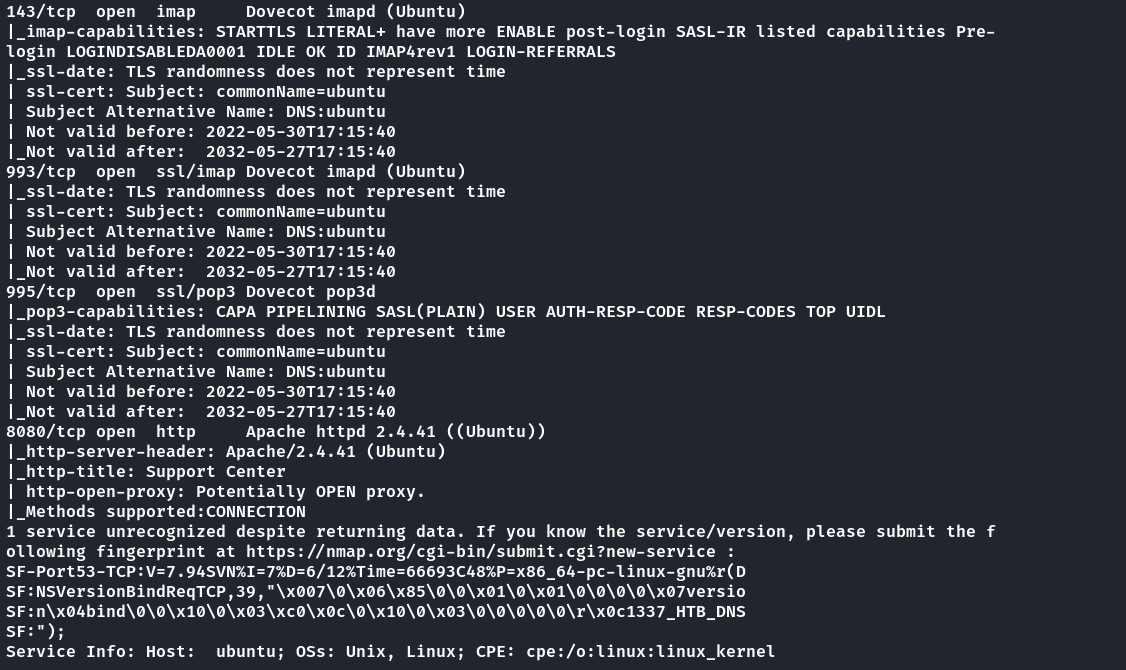
* blog.inlanefreight.local,
* careers.inlanefreight.local.
* dev.inlanefreight.local.
* gitlab.inlanefreight.local.
* Ir.inlanefreight.local.
* Shopdev2.inlanefreight.local
* status.inlanefreight.local.
* vpn.inlanefreight.local.
* Monitoring.inlanefreight.local.

1. Pentester found 13 vulnerabilities during testing all these subdomains.

**Detailed reproduction steps for this attack chain are as follows:**

Scan all the open ports on web server using nmap:

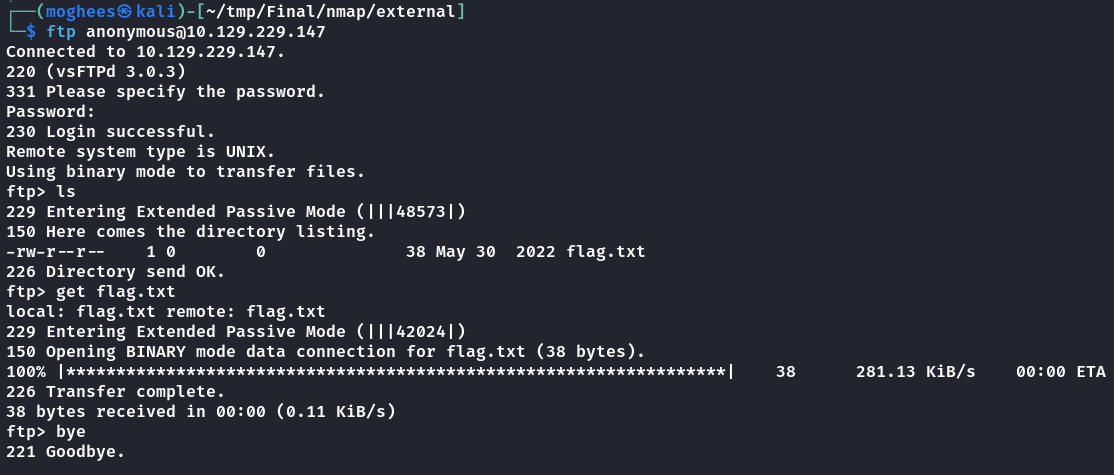




**FTP:**

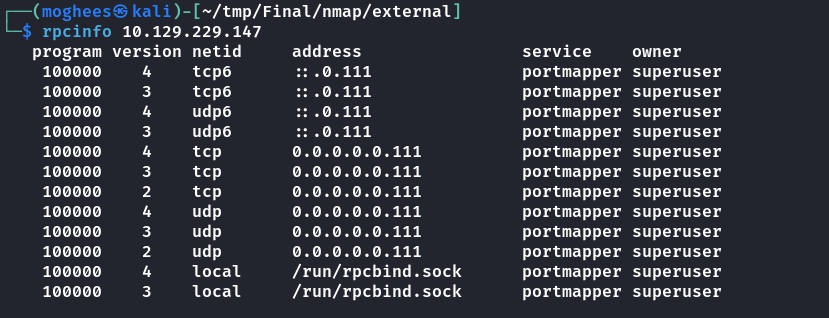
Tester was able to access the server using anonymous credentials and was able to read private files of the company.

* Username: **anonymous**
* Password: **anonymous**



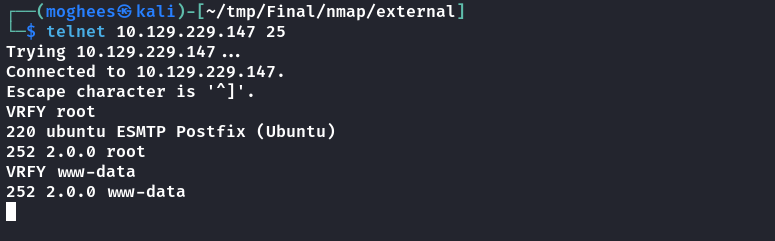
**RPC:**

Tester was also able to read rpcinfo which must not be displayed publically.



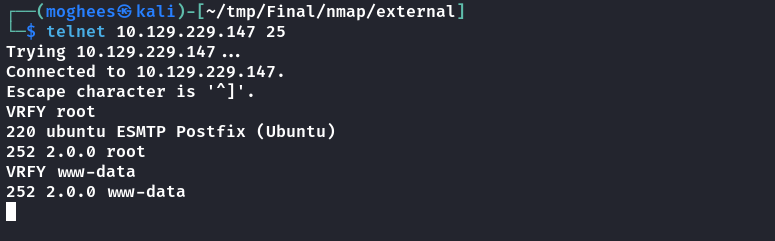
**SMTP:**

Tester was able to verify valid users on the server by logging into SMTP server without credentials and using **VRFY** command.

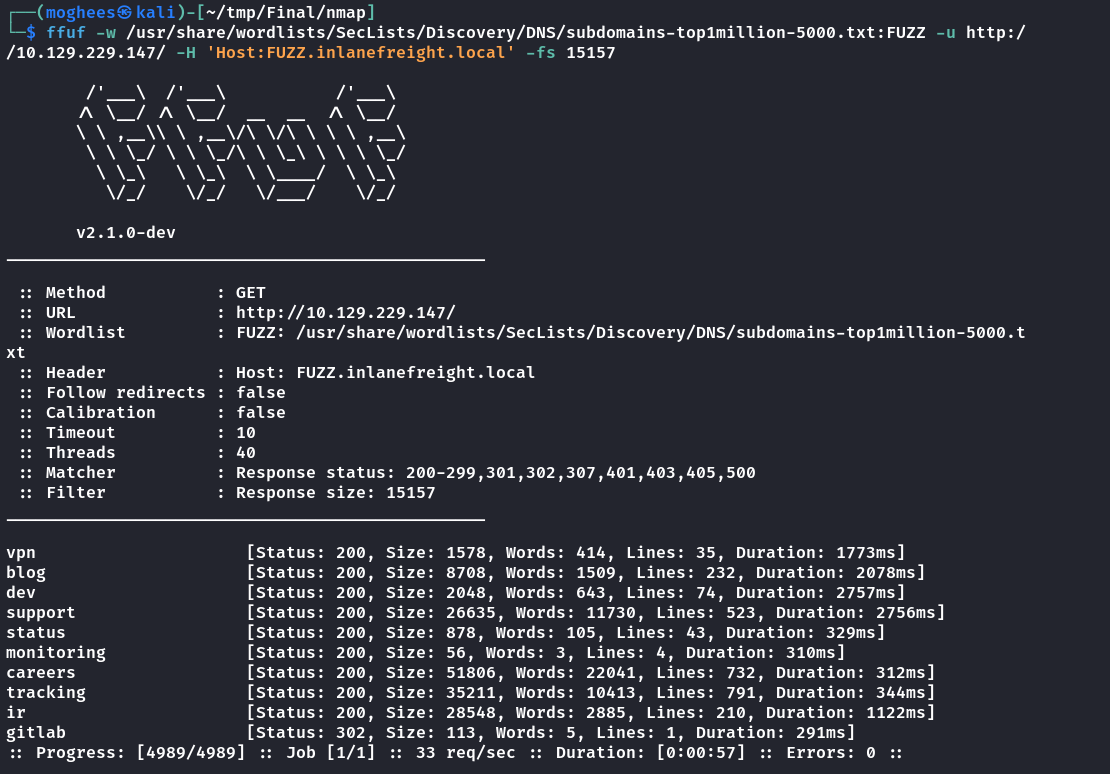


**Web Server:**

Tesster performed subdomain enumeration using **dig**  tool.



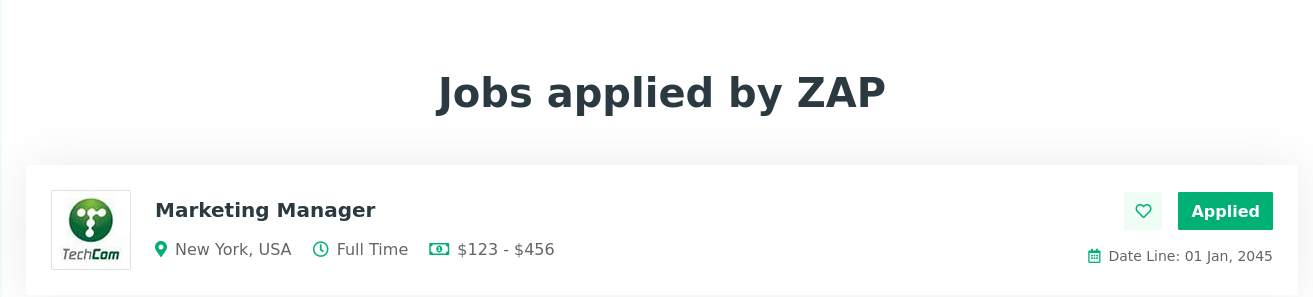
Tester then performed Vhost enumeration and found one more subdomain.



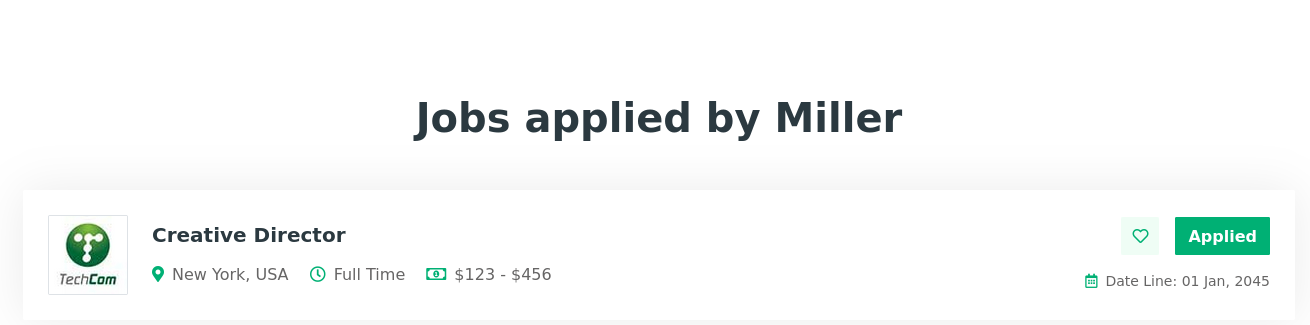
**careers.inlanefreight.local**

While testing this subdomain, tester was able to read private data of other users by exploitinig **IDOR** vulnerability in **id** parameter.

<http://careers.inlanefreight.local/profile?id=9>

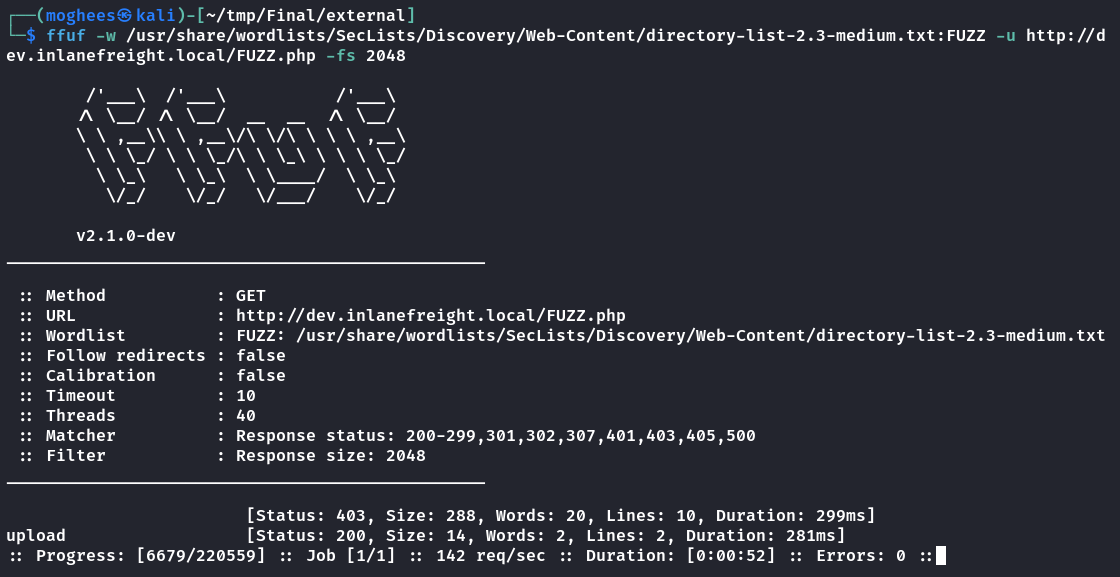


<http://careers.inlanefreight.local/profile?id=8>

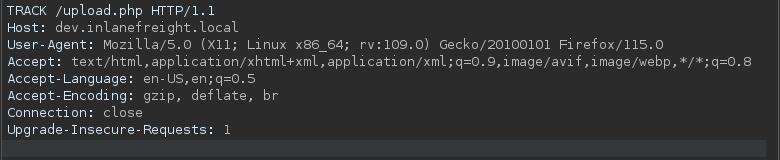


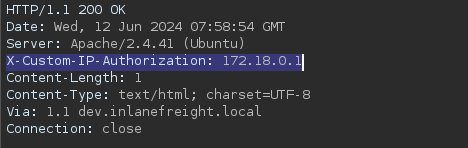
**dev.inlanefreight.local**

The tester found **upload.php** while testing this subdomain using **ffuf** and a wordlist of common wmon web directories.

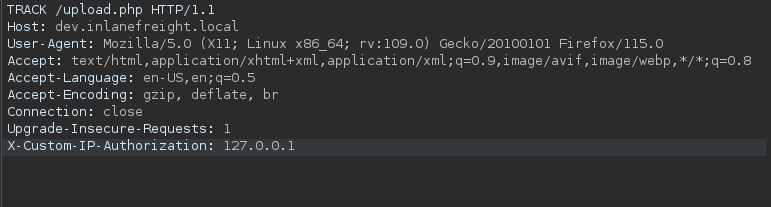


When tester tried to access it, **403** status code was returned which means Unauthorized. But tester was able to find an additional HTTP request header (X-Custom-IP-Authorization) on trying **HTTP Verb Tampering.**

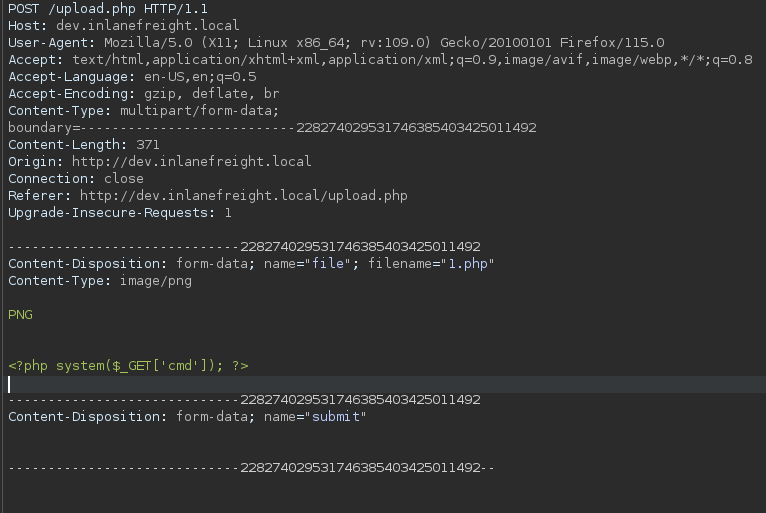
Response:



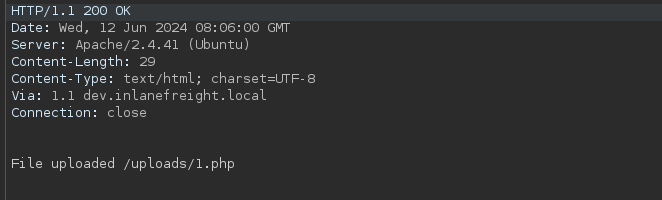
Tester then added **X-Custom-IP-Authorization** with IP of localhost, to the request. This way tester was able to get access to the page.



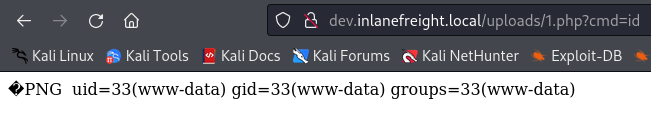
Then tester found a file upload option on the page. There was no proper input sanitization implemented, and tester was able to upload php files. Attacker exploited this vulnerability to get **Remote Code Execution** on the server.



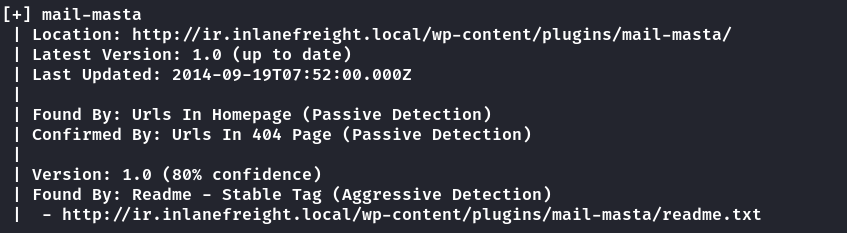
Response:



Exploitation:

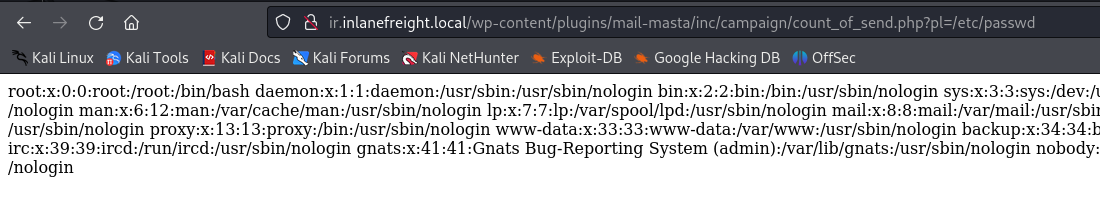


**ir.inlanefreight.local**

While testing this subdomain, tester used **wpscan** to identify any vulnerabilities in the website and found out there was an outdated plugin **mail-masta** used in the website.

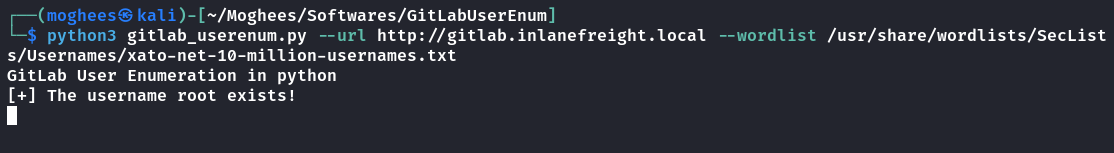
It was vulnerable to **LFI**, public PoC can be found here: <https://www.exploit-db.com/exploits/40290>

Here is the PoC of exploitation.

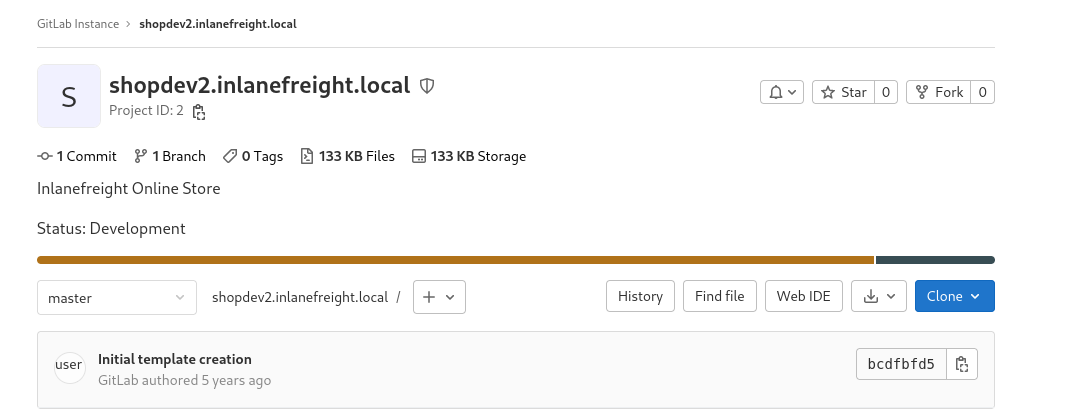


**gitlab.inlanefrieght.local**

Tester found a valid user on this website using **gitlab\_userenum.py** and a wordlist of common names.



Furthermore tester created a new account and selected role as **System Administrator.** Tester found a new subdomain and its code.

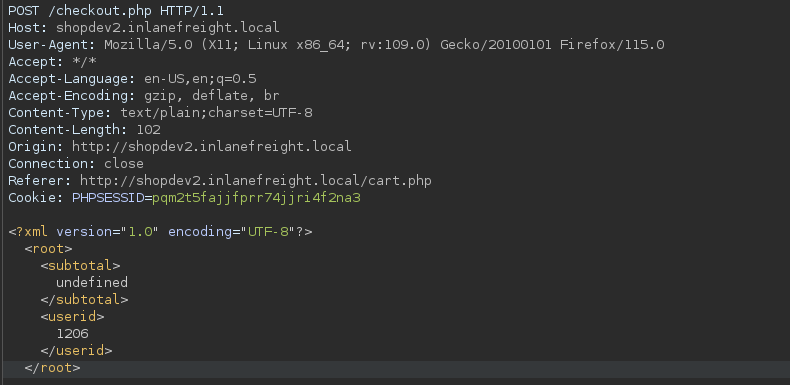


**shopdev2.inlanefreight.local**

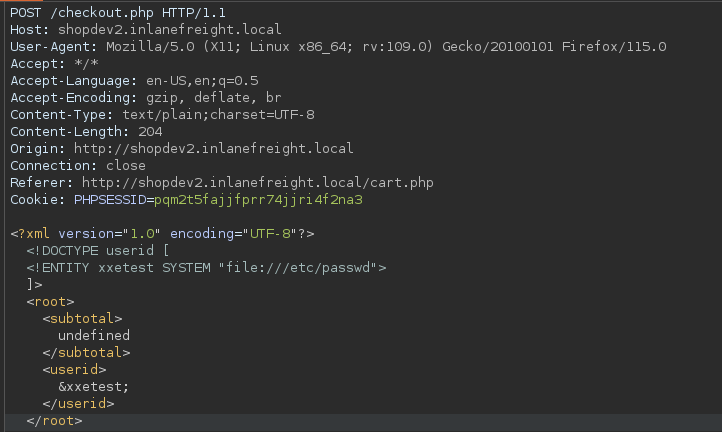
Tester was able to login to the website using default credentials:

* Username: **admin**
* Password: **admin**

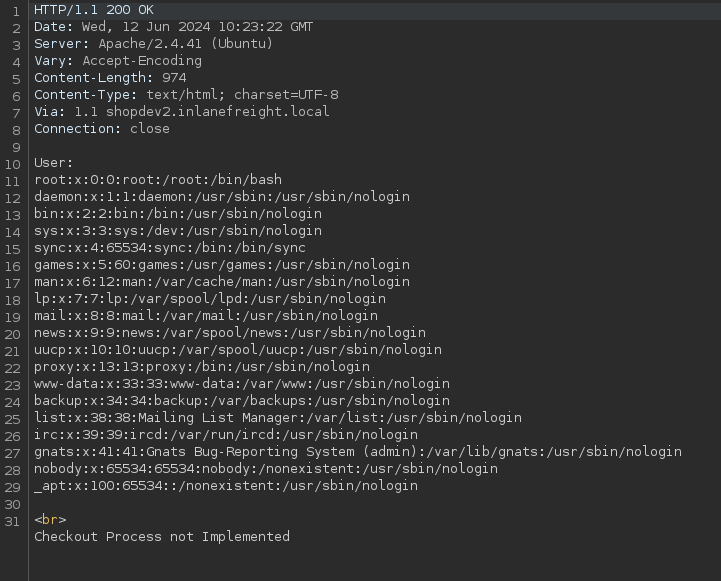
On further enumeration, tester found a page names **checkout**, on clicking the checkout button a request is sent containing userid and subtotal but it was sent in a form of **xml**.



Due to improper input validation, tester was able to perform XML External Entity Injection.

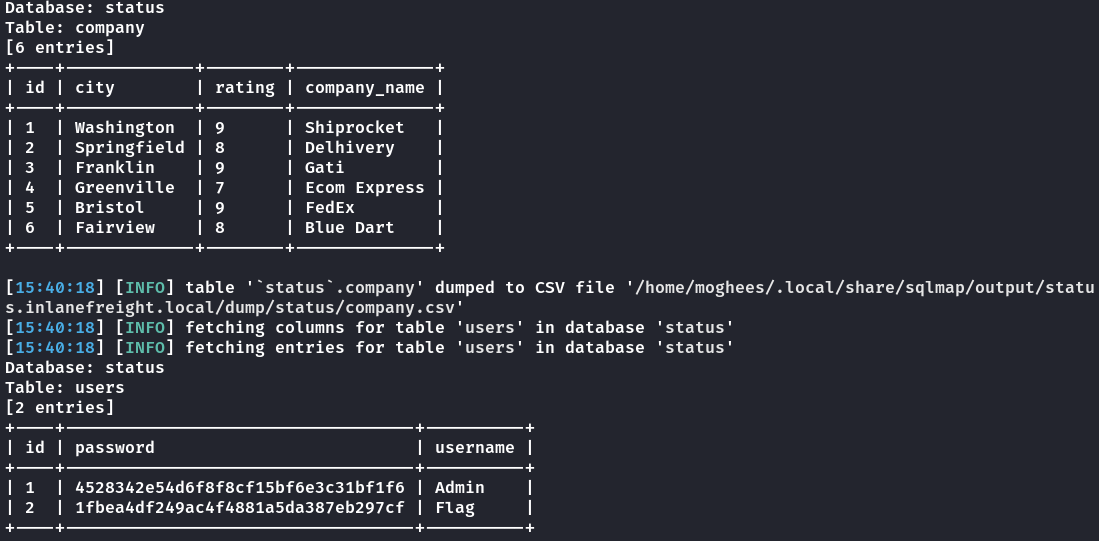
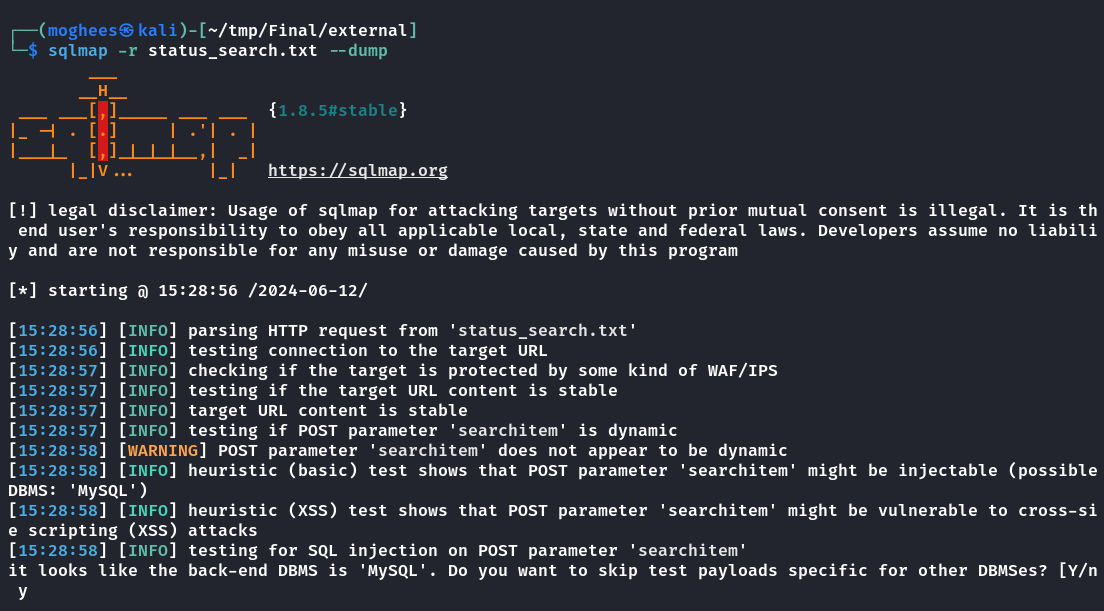


Response:



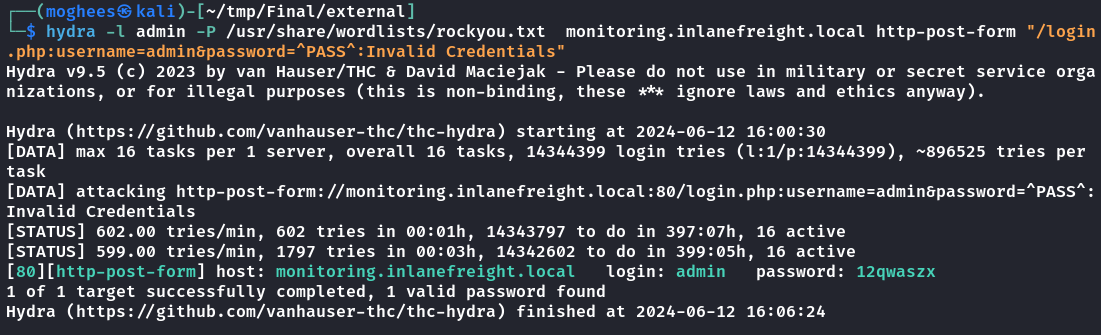
**Status.inlanefreight.local**

There is a search functionality in this subdomain. On testing using **sqlmap** tester was able to find and exploit E**rror Based Sql Injection** vulnerability in the website and dump all data in database.

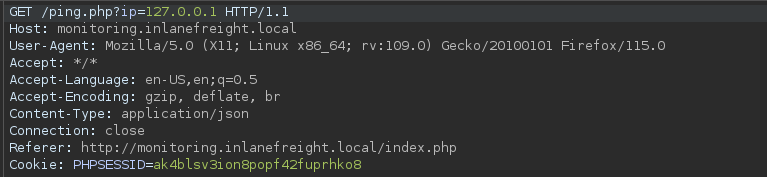


**monitoring.inlanefreight.local**

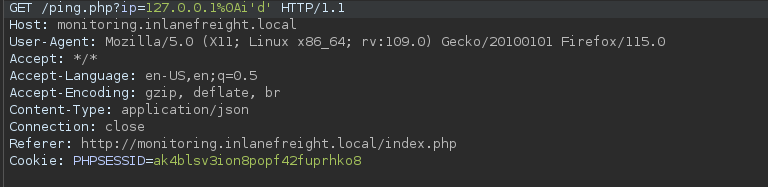
After enumeration, tester found out that there is no **fail2ban** implemented on the website so tester attempted a dictionary attack on the login form using username **admin** and a wordlist of most common passwords and was able to login successfully.



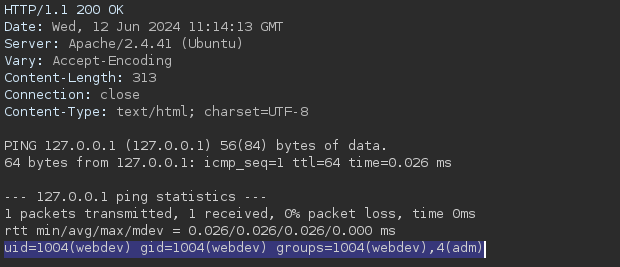
After logging in, tester found a request which was generated when **connection\_test** command was executed.



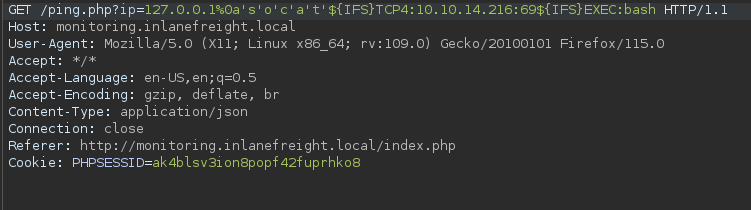
On testing, tester was able to exploit command injection vulnerability due to improper input sanitization and got Remote Code Execution (RCE).

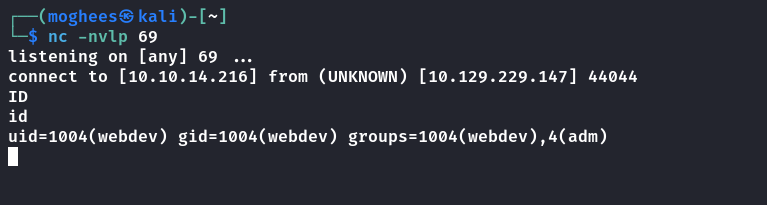


Response:



Tester was able to get foothold on the webserver exploiting this vulnerability.





These were the complete reproduction steps of attack chain that got tester foothold on the webserver owned by InlaneFreight.

# Remediation Summary

As a result of this assessment there are several opportunities for Inlanefreight to strengthen its external facing web server. Remediation efforts are prioritized below starting with those that will likely take the least amount of time and effort to complete. Inlanefreight should ensure that all remediation steps and mitigating controls are carefully planned and tested to prevent any service disruptions or loss of data.

Short Term

* [**Finding 1**] – Disable anonymous login on FTP.
* [**Finding 3**] – Update Wordpress plugins
* [**Finding 4**] – Change the default admin credentials **shopdev2.inlanefreight.local**
* [**Finding 11**] – Disable VRFY command on SMTP on NULL session.

Medium Term

* [**Finding 1**] – Implement proper authorization mechanisms on **careers.inlanefreight.local**
* [**Finding 2**] – Implement proper input validation on upload functionality on **dev.inlanefreight.local**
* [**Finding 3**] – Implement proper authorization on gitlab projects.
* [**Finding 4**] – Disable DTDs (External Entity) completely in **shopdev2.inlanefreight.local**
* [**Finding 4**] – Use prepared statement in search function on **status.inlanefreight.local**
* [**Finding 5**] – Implement strong password policy on **monitoring.inlanefreight.local** and improve input validation. Also implement brute force protections like fail2ban.

Long Term

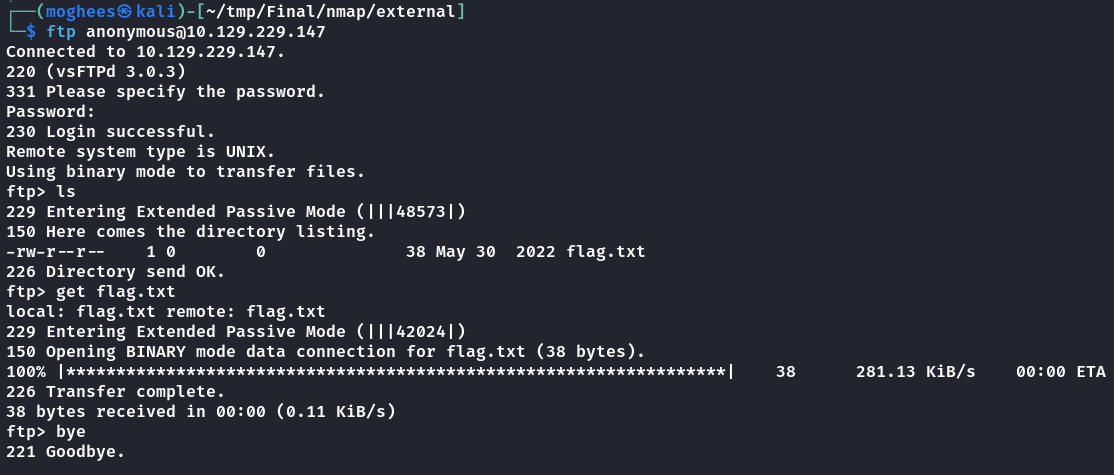
* Implement a login functionality with strong password policy and access controls on **dev.inlanefreight.local**
* Upgrade firewalls in order to avoid easy exploitation of vulnerabilities.

# Technical Findings Details

1. FTP Anonymous Login - High

|  |  |
| --- | --- |
| CWE | CWE-284 |
| CVSS 3.1 Score | NULL |
| Description (Incl. Root Cause) | Anonymous FTP is enabled. |
| Security Impact | Attacker is able to read private data. |
| Affected Domain | 10.10.10.100 |
| Remediation | Anonymous FTP is an unsecured protocol for Internet facing systems and should only be used on a limited basis to provide a specific functional requirement, otherwise disabled. The software should be patched and configured properly. |
| External References | https://attack.mitre.org/techniques/T1071/002/ |

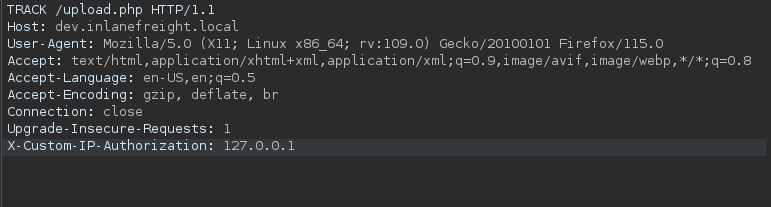
Finding Evidence:

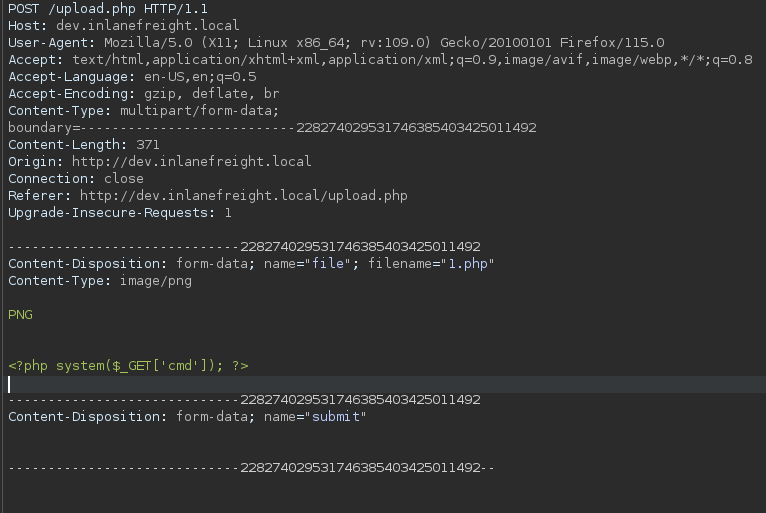


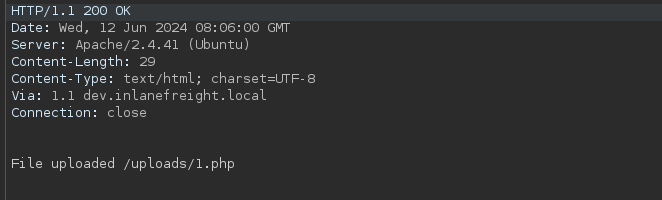
1. RCE Through Insecure File Upload - High

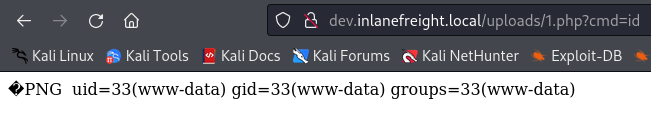
|  |  |
| --- | --- |
| CWE | CWE-434 |
| CVSS 3.1 Score | 8.2 |
| Description | Remote Code Execution (RCE) through insecure file upload is a critical vulnerability that allows attackers to execute arbitrary code on the server hosting the vulnerable application. This type of vulnerability is particularly severe because it can lead to complete system compromise, allowing attackers to gain control over the server, access sensitive information, and disrupt services. |
| Security Impact | This can result in significant service disruptions, data breaches, and loss of customer trust. Additionally, the organization may face substantial financial losses due to incident response costs, legal liabilities, and regulatory fines for non-compliance with data protection laws. The overall damage to the company's reputation and brand image can be profound, making it crucial attack. |
| Affected Domain | * DEV.INLANEFREIGHT.LOCAL |
| Remediation | * Implement stringent file validation * Access controls * Regular security audits * Robust incident response plans to mitigate such risks. |
| External References | https://attack.mitre.org/techniques/ T1203 |

Finding Evidence:





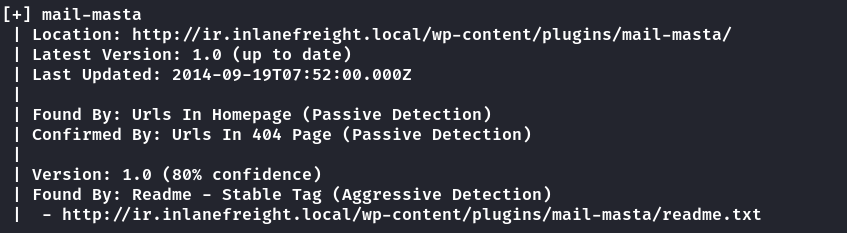


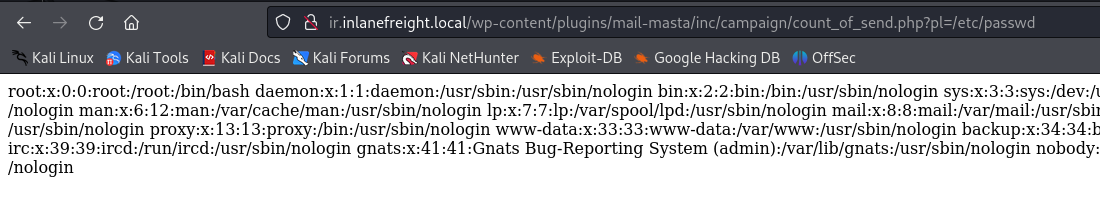


1. Local File Inclusion on Wordpress - High

|  |  |
| --- | --- |
| CWE | CWE-98 |
| CVSS 3.1 Score | 7.5 |
| Description (Incl. Root Cause) | Local File Inclusion (LFI) is a vulnerability that occurs when a WordPress application includes files without proper validation, allowing an attacker to manipulate the file path. This can result in the inclusion of unintended files, such as sensitive configuration files or other local files containing sensitive data or executable code. In WordPress, this often stems from outdated or poorly coded plugins and themes that do not properly sanitize user input. |
| Security Impact | Exploiting an LFI vulnerability can lead to information disclosure, code execution, or even full system compromise. Attackers can read sensitive files such as /etc/passwd or WordPress configuration files (wp-config.php), gain access to sensitive information, and potentially execute arbitrary code if they can upload malicious scripts to the server. |
| Affected Domain | * IR.INLANEFREIGHT.LOCAL |
| Remediation | To remediate LFI vulnerabilities, update all WordPress components, including core files, plugins, and themes, to the latest versions. Implement stringent input validation and sanitization to ensure user inputs are correctly handled. Use WordPress security plugins to monitor and protect against such vulnerabilities and ensure your server is configured to prevent file execution in upload directories. |
| External References | https://attack.mitre.org/techniques/T1203 |

Finding Evidence:





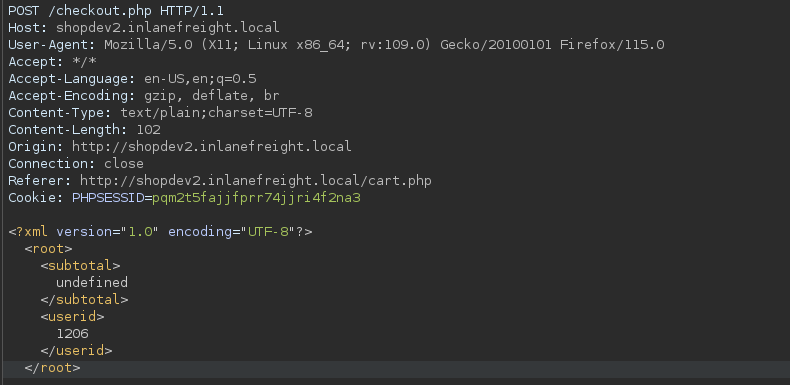
1. Login Using Default Credentials - High

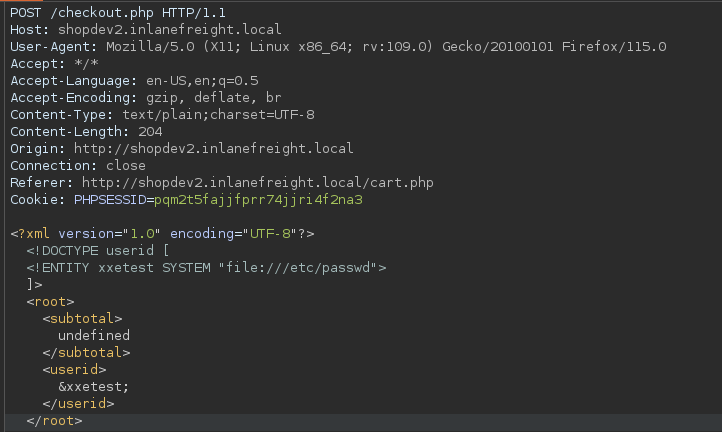
|  |  |
| --- | --- |
| CWE | CWE-798 |
| CVSS 3.1 Score | 7.8 |
| Description (Incl. Root Cause) | Login using default credentials occurs when a WordPress site or its associated services (e.g., database, admin panels) have not changed default usernames and passwords from their initial configurations. Attackers can easily exploit this oversight by using publicly available default credentials to gain unauthorized access. This vulnerability is often found in new installations or configurations where administrators neglect to change default settings. |
| Security Impact | Exploiting default credentials can lead to unauthorized access and compromise of sensitive data, manipulation of system configurations, and potential disruption of services. Attackers can escalate privileges, install malicious software, and perform unauthorized actions with elevated privileges associated with default accounts. |
| Affected Domain | * SHOPDEV2.INLANEFREIGHT.LOCAL   *See Appendix E – INLANEFREIGHT.LOCAL Domain Password Review for a detailed domain password analysis.* |
| Remediation | To mitigate this risk, administrators should change all default credentials immediately after deploying systems or applications. Use strong, unique passwords and implement password policies that enforce regular changes and complexity requirements. Conduct regular audits to identify and replace any remaining default credentials. Employ multi-factor authentication (MFA) where possible to enhance security against credential-based attacks. |
| External References | https://attack.mitre.org/mitigations/T1078 |

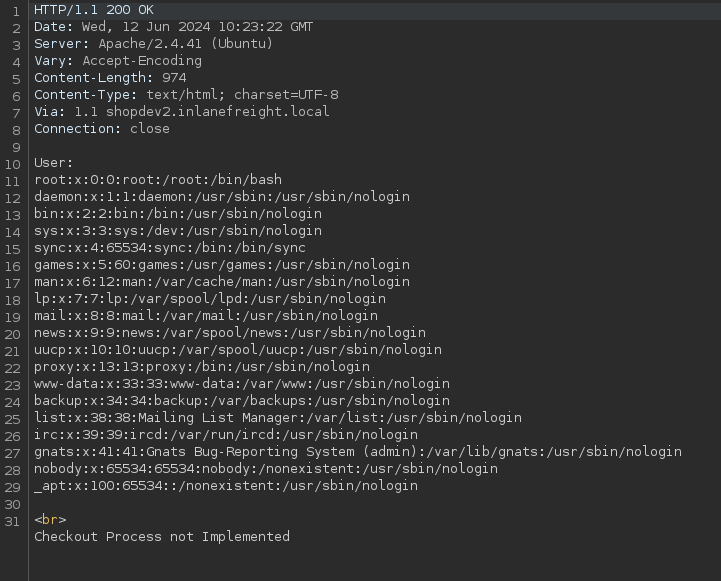
1. XXE - High

|  |  |
| --- | --- |
| CWE | CWE-611 |
| CVSS 3.1 Score | 7.1 |
| Description (Incl. Root Cause) | XML External Entity (XXE) injection is a vulnerability that occurs when an application parses XML input from untrusted sources without disabling external entities. Attackers can exploit this vulnerability to retrieve sensitive data, execute remote code, or perform denial of service attacks by including malicious XML content in their input. This issue is common in applications that process XML data without proper validation and sanitization. |
| Security Impact | Exploiting XXE vulnerabilities can lead to unauthorized access to sensitive data stored on the server, such as configuration files, credentials, and internal network resources. Attackers can also escalate their attack to execute arbitrary code on the server, leading to full system compromise and disruption of services. |
| Affected Host(s) | * SHOPDEV2.INLANEFREIGHT.LOCAL |
| Remediation | To mitigate this risk, developers should disable external entity references in XML parsers or use safer alternatives that do not support external entities. Implement strict input validation and sanitize XML input to remove or block external entity declarations. Regularly update XML parsers and libraries to the latest versions that mitigate XXE vulnerabilities. Perform security assessments and penetration testing to identify and remediate XXE vulnerabilities in applications. |
| External References | https://attack.mitre.org/techniques/T1203 |

Finding Evidence:



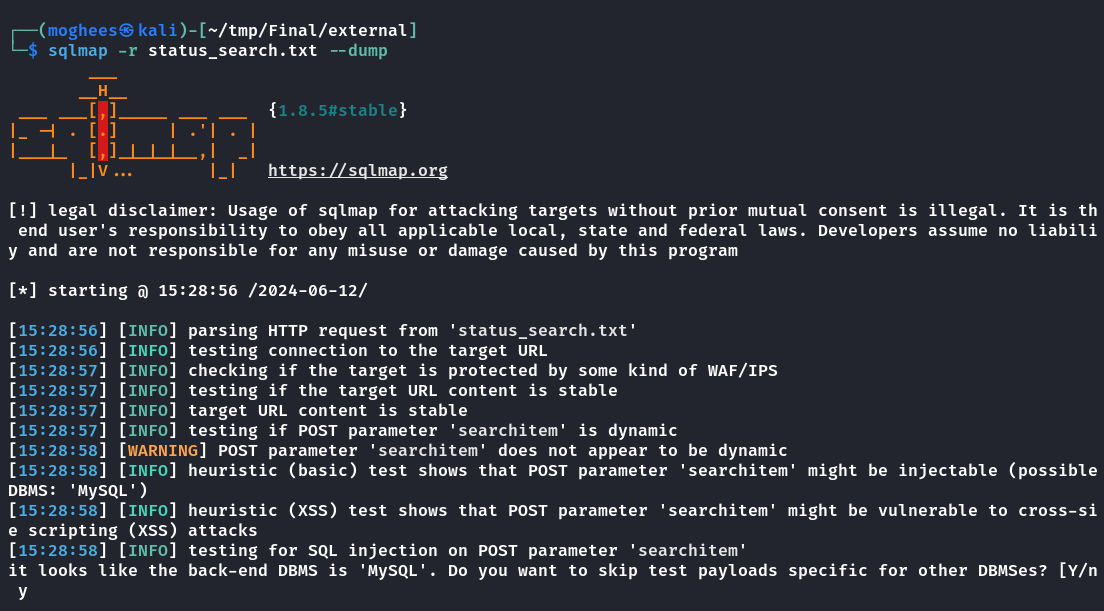


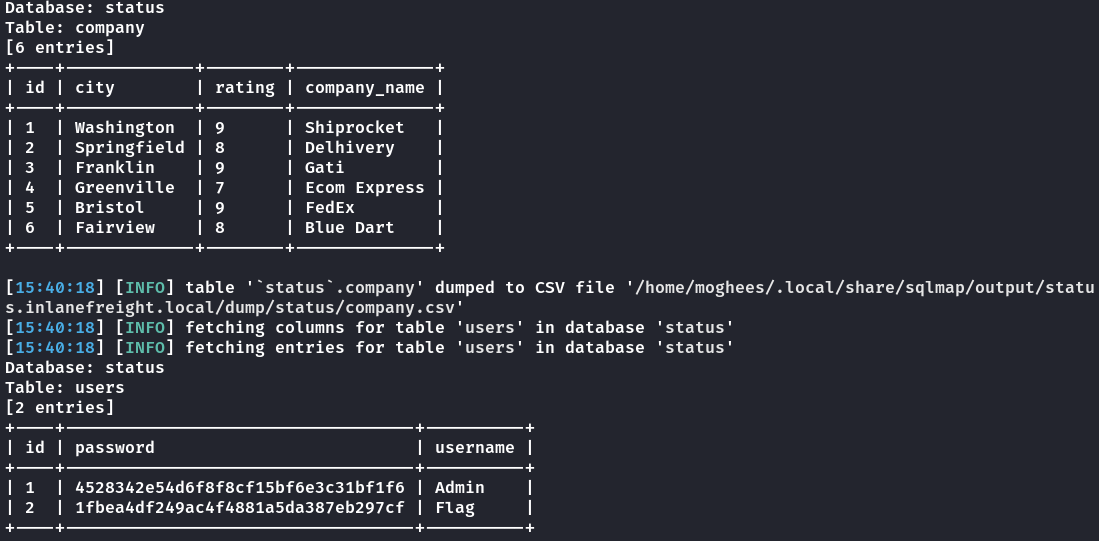


1. Error Based SQL Injection - High

|  |  |
| --- | --- |
| CWE | CWE-89 |
| CVSS 3.1 Score | 9.8 |
| Description (Incl. Root Cause) | SQL Injection (SQLi) is a vulnerability that occurs when untrusted data is inserted into SQL queries without proper validation or sanitization. Attackers exploit this vulnerability by injecting malicious SQL code into input fields of an application, manipulating the query to perform unintended operations. Error-Based SQL Injection specifically leverages error messages returned by the database server to gather information about the database structure or execute arbitrary commands. |
| Security Impact | Exploiting SQLi vulnerabilities can lead to unauthorized access to sensitive data stored in the database, such as usernames, passwords, and financial information. Attackers can modify or delete data, escalate privileges, and execute administrative commands on the database server. This can result in data breaches, service disruptions, and financial losses for the affected organization. |
| Affected Host(s) | * STATUS.INLANEFREIGHT.LOCAL |
| Remediation | To mitigate this risk, developers should implement parameterized queries or prepared statements to separate SQL code from data inputs effectively. Use ORM frameworks that automatically handle SQL queries and prevent injection attacks. Perform thorough input validation and sanitize user inputs to block malicious SQL commands. Regularly update database server software and monitor SQL error logs for signs of attempted SQLi attacks. |
| External References | https://attack.mitre.org/techniques/T1212 |

Finding Evidence:

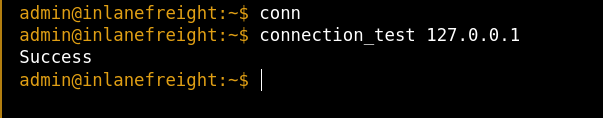


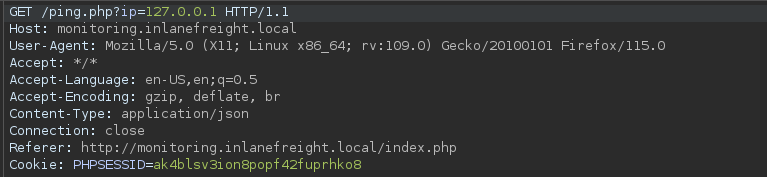


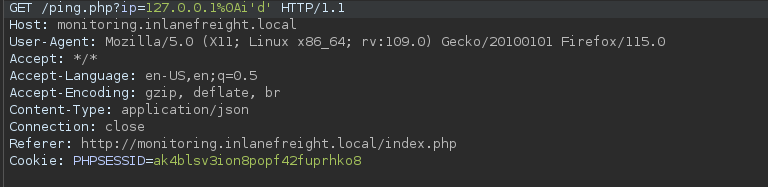
1. OS Command Injection - High

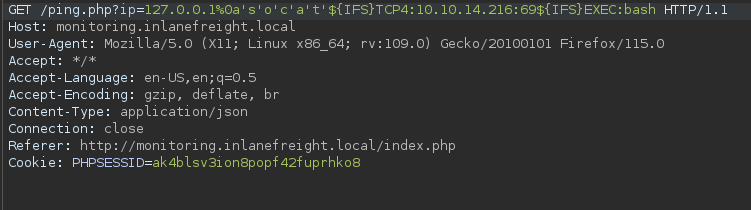
|  |  |
| --- | --- |
| CWE | CWE-78 |
| CVSS 3.1 Score | 9.0 |
| Description (Incl. Root Cause) | OS Command Injection is a vulnerability that occurs when unvalidated user inputs are passed to a system shell or command interpreter. Attackers exploit this vulnerability by injecting malicious commands into input fields or parameters used by the application to execute operating system commands. This allows them to execute arbitrary commands on the underlying operating system with the privileges of the vulnerable application. |
| Security Impact | Exploiting OS Command Injection vulnerabilities can lead to unauthorized access to sensitive data, manipulation of system configurations, and execution of malicious commands on the host operating system. Attackers can install backdoors, escalate privileges, and compromise the entire system, leading to service disruptions, data breaches, and significant financial losses. |
| Affected Host(s) | * MONITORING.INLANEFREIGHT.LOCAL |
| Remediation | To mitigate this risk, developers should implement secure coding practices such as input validation, command sanitization, and using parameterized commands or APIs that do not rely on shell execution. Avoid passing user-supplied data directly to system commands and use application-level controls to limit command execution to necessary and trusted operations. Regularly update system components, libraries, and frameworks to patch known vulnerabilities that could be exploited for OS Command Injection. |
| External References | https://attack.mitre.org/techniques/T1059 |

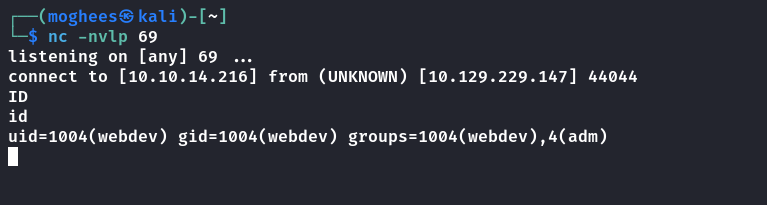
Finding Evidence:







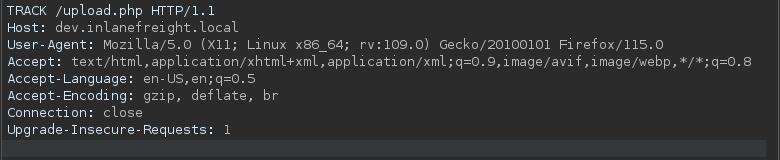




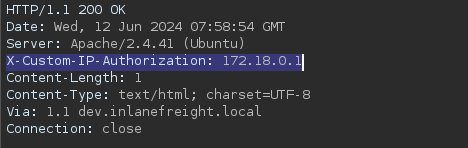
1. HTTP Verb Tampering - Medium

|  |  |
| --- | --- |
| CWE | CWE-287 |
| CVSS 3.1 Score | 6.5 |
| Description (Incl. Root Cause) | HTTP Verb Tampering is a vulnerability that occurs when a web application incorrectly processes unexpected HTTP methods (verbs) such as GET, POST, PUT, DELETE, etc. Attackers exploit this vulnerability by using unsupported or rarely used HTTP methods to bypass security controls, gain unauthorized access, or manipulate data. This can happen when access controls are implemented based only on HTTP methods, leading to inconsistent enforcement of security policies. |
| Security Impact | Exploiting HTTP Verb Tampering can allow attackers to bypass authentication and authorization mechanisms, access sensitive information, and perform unauthorized actions. This can lead to data breaches, modification or deletion of critical data, and unauthorized operations on the server. The overall security posture of the application can be significantly compromised, resulting in potential financial and reputational damage to the organization. |
| Affected Domain | * IDEV.NLANEFREIGHT.LOCAL |
| Remediation | To mitigate this risk, developers should enforce strict validation and handling of HTTP methods used by the web application. Implement comprehensive access control policies that are consistent across all HTTP methods and ensure that all endpoints are protected regardless of the HTTP method used. Use secure coding practices to validate and sanitize all inputs and HTTP requests. Regularly conduct security assessments and penetration testing to identify and remediate HTTP Verb Tampering vulnerabilities. |
| External References | https://attack.mitre.org/techniques/T1078 |

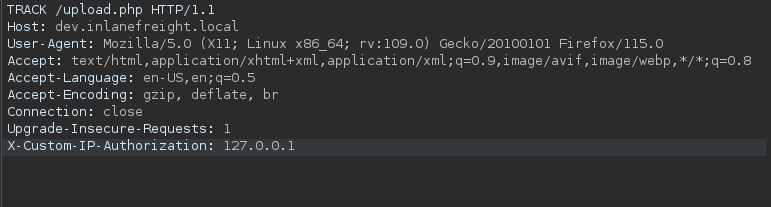
Finding Evidence:



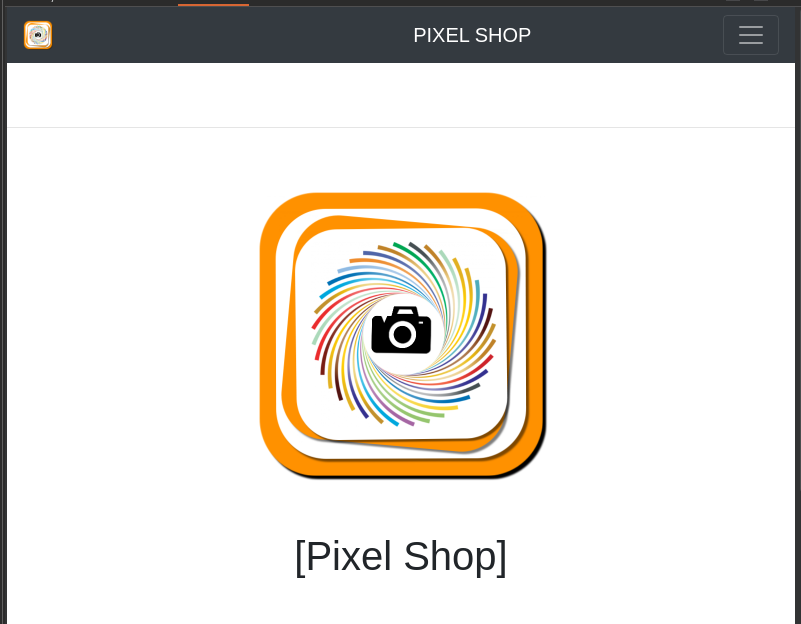
Found an extra header in response



Used this in request and got access to the page:



Got access:

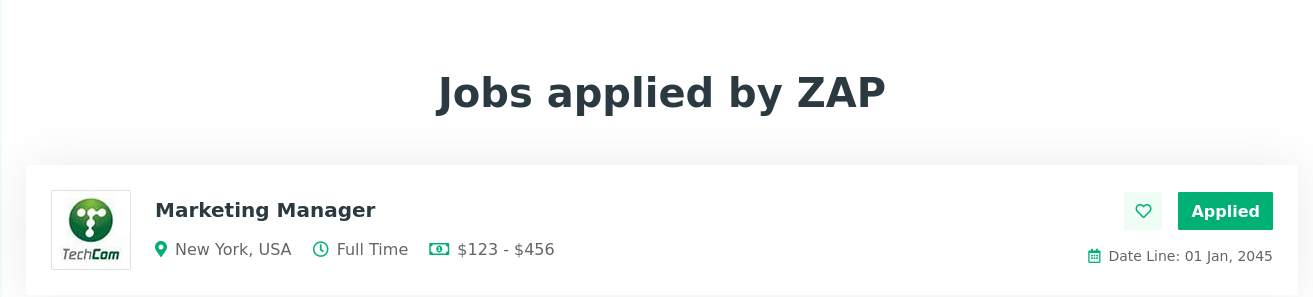


1. Insecure Direct Object Reference (IDOR) - Medium

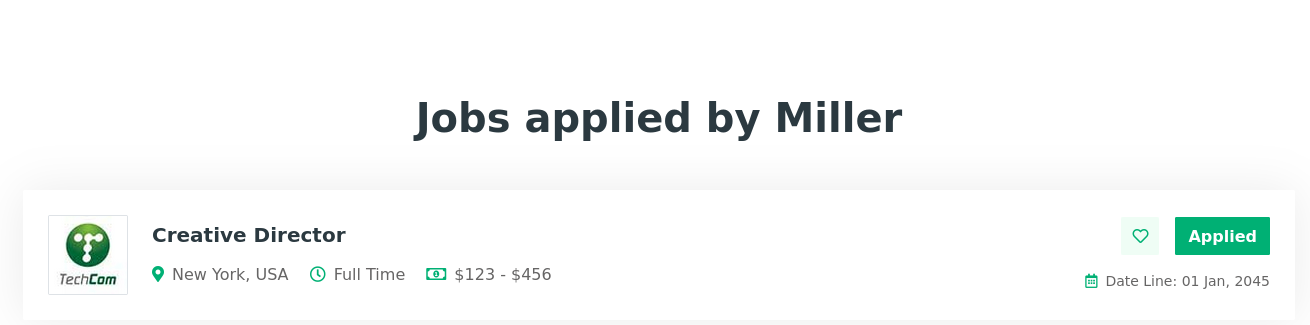
|  |  |
| --- | --- |
| CWE | CWE-639 |
| CVSS 3.1 Score | 6.9 |
| Description (Incl. Root Cause) | Insecure Direct Object Reference (IDOR) is a vulnerability that occurs when an application exposes internal object references to users without proper authorization checks. This allows attackers to manipulate these references to access objects they are not authorized to, such as database records, files, or other resources. The vulnerability arises from insufficient validation and access control checks on user inputs that specify which object to access. |
| Security Impact | Exploiting an IDOR vulnerability can lead to unauthorized access to sensitive information, including personal data, financial records, and confidential business information. Attackers can also perform unauthorized actions, such as modifying or deleting data. This can result in significant data breaches, loss of data integrity, and potentially severe financial and reputational damage to the affected organization. |
| Affected Domain | * CAREERS.INLANEFREIGHT.LOCAL |
| Remediation | Exploiting an IDOR vulnerability can lead to unauthorized access to sensitive information, including personal data, financial records, and confidential business information. Attackers can also perform unauthorized actions, such as modifying or deleting data. This can result in significant data breaches, loss of data integrity, and potentially severe financial and reputational damage to the affected organization. |
| External References | https://attack.mitre.org/techniques/T1078 |

Finding Evidence:

<http://careers.inlanefreight.local/profile?id=9>



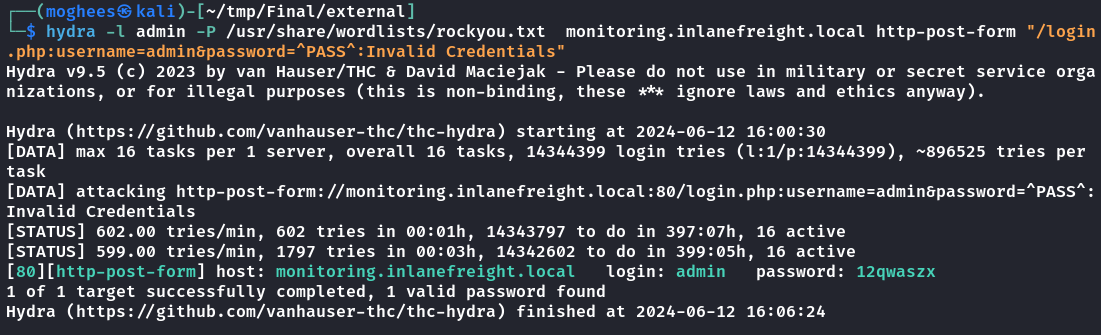
<http://careers.inlanefreight.local/profile?id=8>



1. Weak Password Policy and No Brute Forcing Prevention - Medium

|  |  |
| --- | --- |
| CWE | CWE-307 |
| CVSS 3.1 Score | 6.9 |
| Description (Incl. Root Cause) | Weak Password Policies and the absence of brute forcing prevention mechanisms occur when systems allow users to set weak passwords and do not implement controls to prevent automated attacks on login endpoints. This vulnerability enables attackers to use automated tools to repeatedly attempt to guess passwords, exploiting weak, commonly used, or default passwords. The lack of rate limiting or account lockout mechanisms exacerbates this issue, making it possible for attackers to gain unauthorized access through brute force attacks. |
| Security Impact | Exploiting weak password policies and lacking brute forcing prevention can lead to unauthorized access to user accounts and systems. Attackers can compromise sensitive data, escalate privileges, and perform malicious actions, including data theft, data manipulation, and service disruption. The ease of gaining access through brute force attacks can lead to significant financial and reputational damage, as well as potential legal and regulatory repercussions. |
| Affected Domain | * MONITORING.INLANEFREIGHT.LOCAL |
| Remediation | To mitigate this risk, organizations should enforce strong password policies that require complex, unique passwords with a sufficient length. Implement account lockout mechanisms, CAPTCHA, or rate limiting on login attempts to thwart automated brute force attacks. Use multi-factor authentication (MFA) to enhance security and reduce reliance on passwords alone. Regularly review and update security policies, and perform security testing to identify and address vulnerabilities related to authentication and access controls. |
| External References | https://attack.mitre.org/techniques/T1110 |

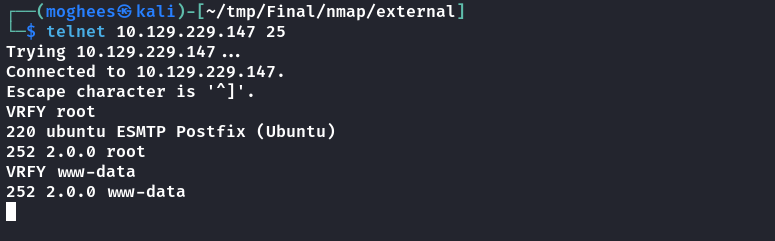
Finding Evidence:



1. SMTP VRFY Command Enabled- Low

|  |  |
| --- | --- |
| CWE | CWE-269 |
| CVSS 3.1 Score | 4.5 |
| Description (Incl. Root Cause) | The SMTP VRFY (verify) command is used to verify the existence of email addresses on a mail server. When enabled, attackers can use this command to enumerate valid email addresses, which can then be used for targeted phishing attacks, spamming, and other malicious activities. This vulnerability arises from the improper configuration of the mail server, where the VRFY command is left enabled, providing a valuable reconnaissance tool for attackers. |
| Security Impact | Exploiting the SMTP VRFY command can lead to information disclosure, as attackers can compile a list of valid email addresses on the target server. This information can be used to launch subsequent attacks, such as targeted phishing campaigns, social engineering attacks, and brute force attempts on email accounts. The compromised email addresses can also be sold or used for spamming, leading to reputational damage and potential legal consequences for the affected organization. |
| Affected Host(s) | * 10.10.10.100:25 |
| Remediation | To mitigate this risk, administrators should disable the VRFY command on the SMTP server. This can typically be done by modifying the mail server's configuration settings. Additionally, administrators should implement email security best practices, such as enforcing strong authentication mechanisms, using spam filters, and monitoring email server logs for suspicious activity. Regular security assessments and audits of the mail server configuration should be conducted to ensure that insecure commands and configurations are identified and remediated promptly. |
| External References | https://attack.mitre.org/techniques/T1087 |

Finding Evidence:



1. RPC Information Disclosure - Info

|  |  |
| --- | --- |
| CWE | CWE-200 |
| Description (Incl. Root Cause) | RPC (Remote Procedure Call) information disclosure occurs when an RPC service is improperly configured, allowing unauthorized users to retrieve sensitive information about the network, services, and users. This can happen if the service is not secured with proper access controls, leading to the exposure of details such as available RPC services, user accounts, and other network resources. Attackers can exploit this vulnerability to gather intelligence for planning further attacks, such as lateral movement or privilege escalation. |
| Security Impact | Exploiting an RPC information disclosure vulnerability can lead to unauthorized access to critical network information, which can be used to facilitate a range of attacks, including unauthorized access, privilege escalation, and lateral movement within the network. This exposure can significantly compromise the security of the network, leading to data breaches, service disruptions, and other malicious activities. |
| Remediation | To mitigate this risk, administrators should secure RPC services by implementing strong access controls and authentication mechanisms. Ensure that only authorized users and systems can access RPC services, and restrict RPC traffic to trusted networks. Regularly audit and update RPC configurations to ensure they follow best security practices. Additionally, monitor network traffic for unusual or unauthorized RPC activity, and conduct regular security assessments to identify and remediate potential vulnerabilities. |
| External References | https://attack.mitre.org/tactics/TA1018 |

1. Sensitive Data Exposure on GitLab - Info

|  |  |
| --- | --- |
| CWE | CWE-200 |
| Description (Incl. Root Cause) | Sensitive data exposure on GitLab occurs when repositories, issues, or other GitLab resources unintentionally reveal confidential information such as API keys, passwords, personal data, or proprietary code. This can happen if access controls are misconfigured, sensitive information is hardcoded or improperly stored within the repository, or public repositories are used inappropriately. Attackers can exploit this vulnerability to gain unauthorized access to critical systems, steal sensitive data, or use exposed information for further attacks. |
| Security Impact | Exploiting sensitive data exposure on GitLab can lead to unauthorized access to systems and data, resulting in data breaches, identity theft, financial loss, and reputational damage. Attackers can use exposed credentials to infiltrate other systems, escalate privileges, and move laterally within an organization’s network. The leaked sensitive information can also be used to launch targeted attacks, such as phishing campaigns or credential stuffing. |
| Remediation | To mitigate this risk, organizations should enforce strict access controls and permissions on GitLab repositories, ensuring that only authorized users have access to sensitive data. Implement secret management solutions to handle sensitive information securely and avoid hardcoding credentials in repositories. Regularly review and audit repositories for exposed sensitive data and rotate exposed credentials immediately. Educate developers on secure coding practices and the importance of keeping sensitive information out of code repositories. |
| External References | https://attack.mitre.org/tactics/TA1005 |

# Appendices

Appendix A – Finding Severities

Each finding has been assigned a severity rating of high, medium, or low. The rating is based off of an assessment of the priority with which each finding should be viewed and the potential impact each has on the confidentiality, integrity, and availability of Inlanefreight’s data.

|  |  |
| --- | --- |
| Rating | Severity Rating Definition |
| **High** | Exploitation of the technical or procedural vulnerability will cause substantial harm. Significant political, financial, and/or legal damage is likely to result. The threat exposure is high, thereby increasing the likelihood of occurrence. Security controls are not effectively implemented to reduce the severity of impact if the vulnerability were exploited. |
| **Medium** | Exploitation of the technical or procedural vulnerability will significantly impact the confidentiality, integrity, and/or availability of the system, application, or data. Exploitation of the vulnerability may cause moderate financial loss or public embarrassment. The threat exposure is moderate-to-high, thereby increasing the likelihood of occurrence. Security controls are in place to contain the severity of impact if the vulnerability were exploited, such that further political, financial, or legal damage will not occur.  - OR -  The vulnerability is such that it would otherwise be considered High Risk, but the threat exposure is so limited that the likelihood of occurrence is minimal. |
| Low | Exploitation of the technical or procedural vulnerability will cause minimal impact to operations. The Confidentiality, Integrity and Availability (CIA) of sensitive information are not at risk of compromise. Exploitation of the vulnerability may cause slight financial loss or public embarrassment. The threat exposure is moderate-to-low. Security controls are in place to contain the severity of impact if the vulnerability were exploited, such that further political, financial, or legal damage will not occur.  - OR -  The vulnerability is such that it would otherwise be considered Medium Risk, but the threat exposure is so limited that the likelihood of occurrence is minimal. |

Table 4: Severity Definitions

Appendix B – Exploited Hosts

|  |  |  |  |
| --- | --- | --- | --- |
| Host | Scope | Method | Notes |
| 10.10.10.100 | External | OS Command Injection | Foothold |

Table 5: Exploitation Attempt Details

Appendix C – Compromised Users

|  |  |  |  |
| --- | --- | --- | --- |
| Username | Type | Method | Notes |
| www-data | Local User | RCE through insecure file upload | dev.inlanefreight.local |
| admin | Web User | Default credentials | shopdev2.inlanefreight.local |
| admin | Web User | Dictionary Attack | monitoring.inlanefreight.local |

Table 6: User Accounts Compromised

Appendix D – Changes/Host Cleanup

|  |  |  |
| --- | --- | --- |
| Host | Scope | Change/Cleanup needed |
| Dev.inlanefreight.local | External | Delete upload.php file from web server |

Table 7: Assessment Artifacts

Appendix E – INLANEFREIGHT.LOCAL Domain Password Review

Password Statistics

|  |  |
| --- | --- |
| Metric | # |
| Total Password Found | 2 |

Table 8: Password Statistics