Internal Penetration Test Report

Private and Confidential

PREPARED BY	Date	VERSION
[redacted]	01/09/2022	2.0

IMPORTANT: The information contained in this document may be privileged, business sensitive, proprietary and/or copyright, protected from disclosure and/or be subject to US export control. If you are not the intended recipient, you are hereby notified that any dissemination, distribution, or copying of this communication is strictly prohibited.

Disclosure Statement

This document contains confidential information related to the network environment, practices, and vulnerabilities that were found in LBC's infrastructure. Information in this document is intended only for the person or organization to which it is disclosed. Any attempt to access, use, or redistribute this document must be approved by LeBonBonCroissant and [redacted] This document follows the terms and conditions of the non-disclosure agreement between [redacted] and LeBonBonCroissant.

Document Property

Client	Le Bonbon Croissant
File Name	2022_[redacted]_LBC_Report.pdf
Version	2.0
Date	01/09/2022
Point of Contact	Jim Joseph
Contact	http://lebonboncroissant.com/

Document History

Version	Date	Description
0.1	10/6/2021	Initial document template created
0.5	10/23/2021	Report draft updated with 1st engagement findings
1.0	10/23/2021	Report released to the client
1.1	01/09/2022	Report updated with findings from 2nd engagement
2.0	01/09/2022	Report released to the client

Team 1 Contact Information

Team Lead	[redacted] Principal Security Engineer
Address	[redacted]
Email	[redacted]cptc.org
Phone	555-555-5555

Table of Contents

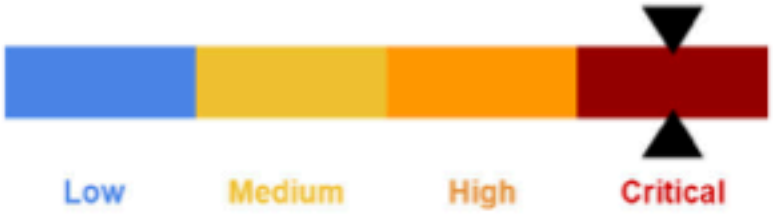
1. Executive Summary

[redacted] was contracted by Le BonBon Croissant (hereafter referred to as LBC) to conduct their network penetration test between January 7th, 2022 and January 8th, 2022. This is the second internal penetration test of LBC's network that [redacted] conducted. The assessment consisted of four main goals: (1) Re-test and validate whether the findings from the previous engagement were mitigated (2) Identify new vulnerabilities and assess their risk to LBC's infrastructure and business operations (3) Evaluate and assess security posture related to Payment Card Industry Data Security Standard (PCI/DSS), and (4) Outline key remediation steps to harden digital and critical infrastructure security as part of LBC's plan to secure their network.

[redacted] identified that LBC has fully or partially remediated about 72% of vulnerabilities from the previous engagement, indicating significant dedication by LBC towards security. [redacted] commends such improvement, which will serve well to reduce LBC's exposure to threat actors. After the second test, [redacted] concluded that LBC's current infrastructure is vulnerable to several internal threats. [redacted] identified **0 critical** severity vulnerabilities, **7 high** severity vulnerabilities, **9 medium** severity vulnerabilities, **2 low** severity vulnerabilities and **2 informational** findings. Unauthenticated access to databases containing customer data and lack of access control, particularly on industrial control systems, are major vulnerabilities. Adversaries can exploit these vulnerabilities to extract customer data at scale and render LBC's physical warehouse operations inoperable. This could lead to significant disruption of LBC's core business and cause reputational loss.

Several findings deviated from the security guidelines outlined in the PCI-DSS standard. Findings related to unauthenticated access to LBC assets violate both GDPR and PCI-DSS regulations. Given LBC's business interests in France, deviations from GDPR on customer data storage could also expose LBC to regulatory risk.

Although none of the vulnerabilities were classified as critical from a technical perspective, [redacted] considers the overall risk to LBC's business operations to be **Critical** due to the business impact of the findings.



[redacted] recommends enforcing strong authentication on databases, ensuring segmentation of critical warehouse PLCs, and employing thorough source code review on internal applications. Applying these recommendations and other mitigations in this report will significantly secure LBC's network and ensure regulatory compliance with PCI-DSS and GDPR.

2. Engagement Overview

After the initial engagement with LBC on 10/23/2021, [redacted] was contracted to perform another internal penetration test on LBC's network. Based on the shared RFP (Request for proposal) document and feedback from LBC after the initial engagement, the team focused on the following goals during the engagement.

1. Find gaps and vulnerabilities within digital security, security management, protective measures, mitigation, and recovery within the infrastructure of LBC.

2. Aid LBC with improving critical infrastructure security.

3. Evaluate LBC security posture against the Payment Card Industry Data Security Standard (PCI DSS) and GDPR standards.

4. Check and validate if findings found during the first engagement have been remediated.

2.1. Scope

The scope of the engagement is shown in the table below. During the engagement, the critical infrastructure PLCs (10.0.17.50 and 10.0.17.51) were initially out of scope but later added to LBC's request after creating a comprehensive test plan with [redacted]

IP Range (CIDR)	Name	Description
10.0.17.0/24	Paris Warehouse	LBC warehouse critical infrastructure (PLCs), new LBC e-commerce platform, Rewards system & other misc. Web apps and databases.

2.1.1. Topology

The overall topology [redacted] discovered and tested during the engagement is illustrated below.

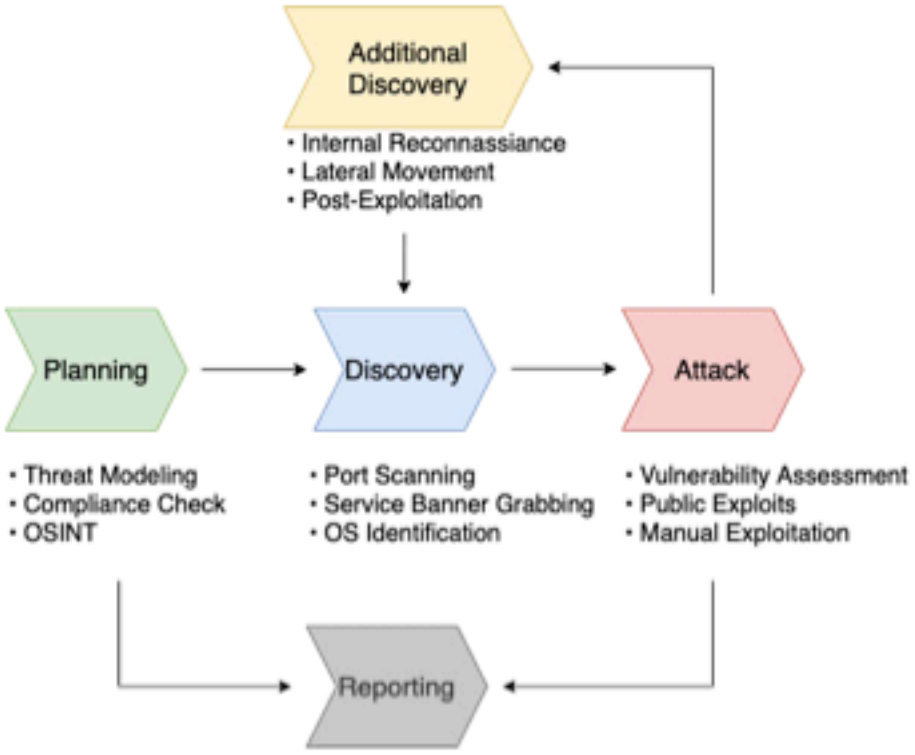
The scope of the engagement was a flat network 10.0.17.0/24 hosting ten machines. All hosts were accessible directly through the assigned testing VDI infrastructure.

2.2. Methodology

[redacted] utilizes the National Institute of Standards and Technology (NIST) Special Publication 800-115 as the overall penetration testing methodology. NIST is an organization under the U.S Department of Commerce that guides the organizations to lead the industry by providing various standards and guides. Of those, NIST 800-115 is a "Technical Guide to Information Security Testing and Assessment".

NIST 800-115 provides a general methodology of a security assessment. The methodology covers assessment overview, documentation, target identification, target vulnerability validation, assessment planning, assessment execution, and post-test activities.

For penetration testing specifically, NIST 800-115 presents five steps: Planning, Discovery, Attack, Additional Discovery, and Reporting. The following diagram explains [redacted]s implementation of NIST 800-115 penetration testing phases.



2.3. Technical Impact Metric

For the technical assessment of a vulnerability, [redacted] uses the Common Vulnerability Scoring System version 3.1 (CVSS v3.1). CVSS is a universally accepted and open standard created by the National Institute of Standards and Technology (NIST) Computer Security Resource Center (CSRC). CVSS measures a vulnerability's complexity, accessibility, and impact on the confidentiality, integrity, and availability of a system. For the calculation of CVSS, [redacted] utilizes the National Vulnerability Database (NVD)'s CVSS v3.1 calculator.

The scores represented in the report are based on the collective experience of [redacted] and are tailored specifically to LBC. These scores are not representative of the scoring assigned officially in the NVD and should not be interpreted as such. In each vulnerability or finding table, the CVSS string is included along with the raw score to give further context to LBC's technical staff. Further reading and information about the scoring system can be found on the Forum for Incident Response and Security Teams (FIRST) website (www.first.org).

CVSS SCORING	
SEVERITY	**BASE SCORE RATING**
Critical	**9.0-10.0**
High	**7-8.9**
Medium	**4-6.9**
Low	**0.1-3.9**
Info	**0**

2.4. Business Impact Metric

While the CVSS score provides technical insight about a vulnerability, vulnerabilities are often tied with real-world business impact and likelihood. To consider these contexts, [redacted] also uses a Risk-Matrix. The table below provides some context into the overall risk, given the business impact and likelihood.

RISK MATRIX		THREAT IMPACT			
		LOW	MEDIUM	HIGH	CRITICAL
LIKELIHOOD	RARE	Low	Low	Medium	Medium
	UNLIKELY	Low	Medium	High	High
	LIKELY	Low	Medium	High	Critical
	VERY LIKELY	Low	Medium	Critical	Critical

2.5. Mitigation Prioritization Metric

Mitigation Prioritization metrics are designed to assist the clients with prioritizing their mitigation strategies for vulnerabilities discovered on their network. The technical and business impact metrics are used as a standard to find a vulnerability's risk. [redacted]s in-house tier system is the mitigation prioritization metric used to prioritize findings remediation.

Mitigation Priority	Description
Category - 4 (CAT-4)	Finding has a critical business impact, likelihood, and risk. It damages the operation of the client. Finding causes a direct violation of regulation, law, or compliance that applies to the client. Finding leaks Personal Identifiable Information (PII), Sensitive Information (SI), or any other information that can lead to further access to sensitive data. Finding is related to previous indicators of compromise and suggests the occurrence of past cyberattacks.

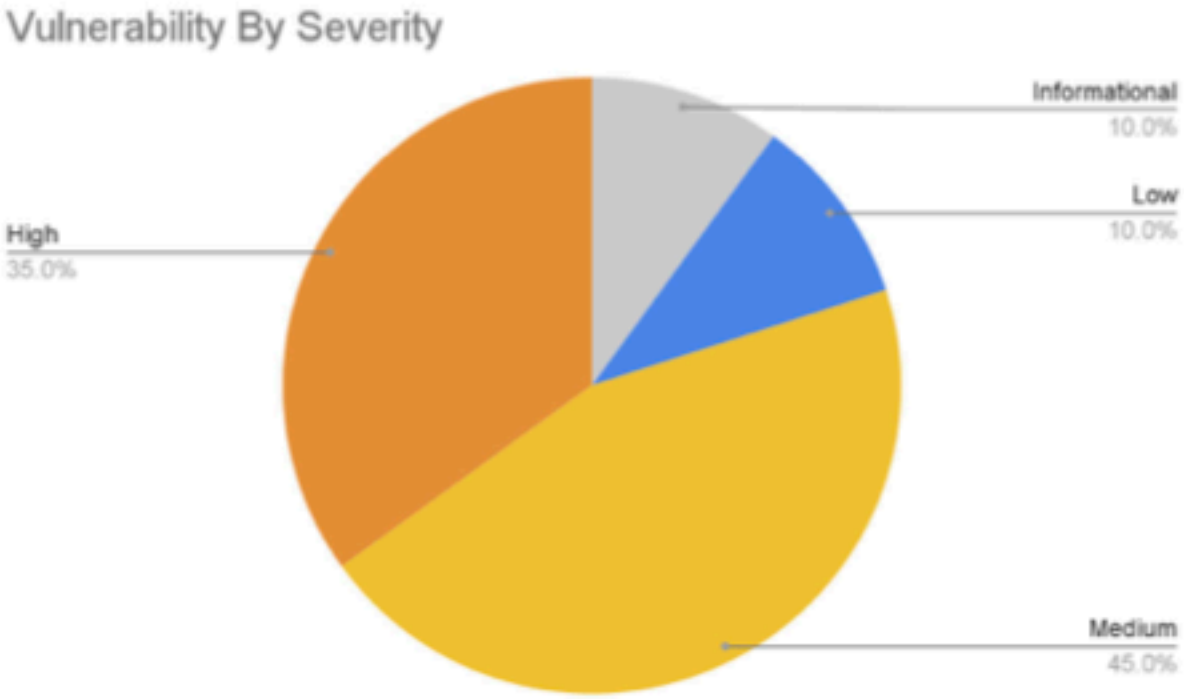
Category - 3 (CAT-3)	Finding has a high business impact, likelihood, and risk. It partially damages the client's operation and has the potential for further exploitation. Finding gives attackers direct access to a system or a service. Finding allows the attackers to violate the Confidentiality, Integrity, Availability of a system.
Category - 2 (CAT-2)	Finding has a medium business impact, likelihood, and risk. Finding is related to security misconfigurations which can lead to further potential attacks. The finding allows attackers to partially violate the Confidentiality, Integrity, and availability of a system.
Category - 1 (CAT-1)	Finding has a low business impact, likelihood, and risk. Finding is not following the best security practices. Finding is a bug or an unintentional mistake with little to no security implication.

3. Assessment Summary

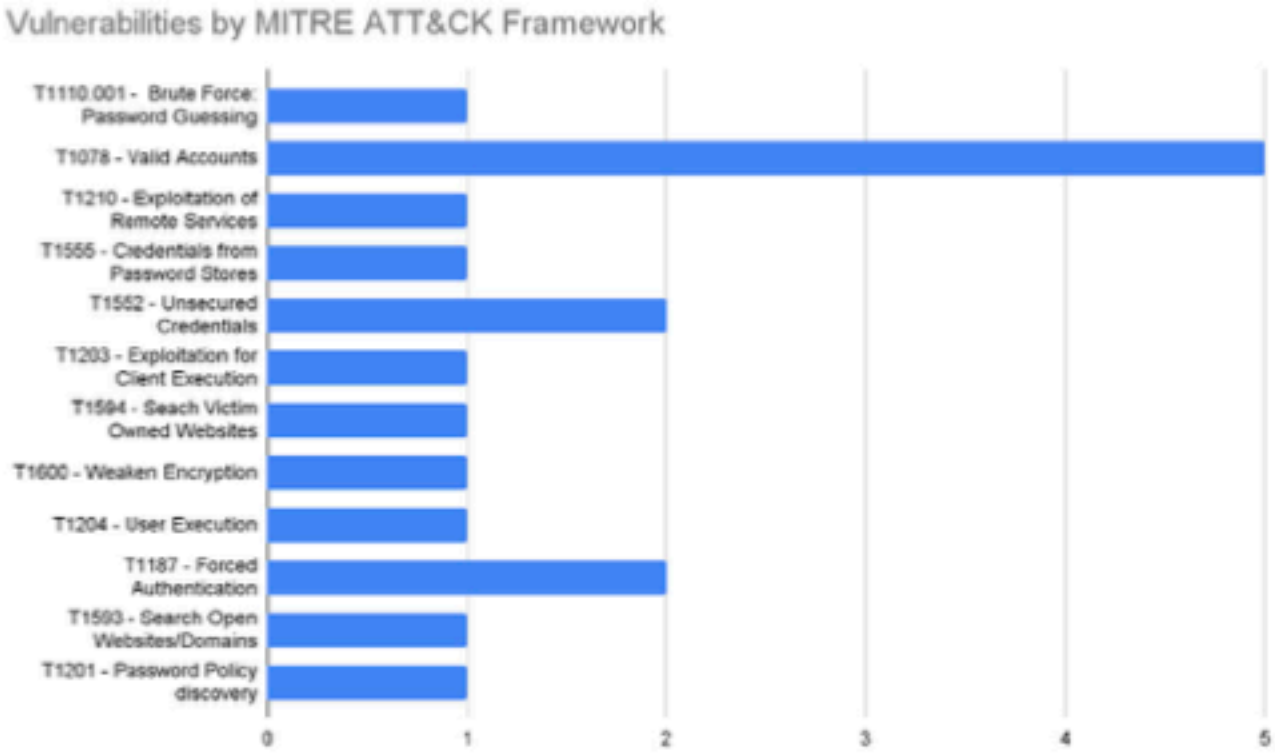
The following section breaks down some of the important statistics, key findings, and key remediations found during the engagement.

3.1. Statistics

The pie chart below summarizes all of the vulnerabilities found in LBC's infrastructure during the second engagement. The categorization of the vulnerabilities is done using CVSS v3.1, as mentioned in the technical metrics section. In total, [redacted] was able to find 0 critical, 7 high, 9 medium, 2 low, and 2 informational vulnerabilities on the infrastructure.



The histogram below provides a visual representation, mapping the discovered findings to Tactics, Techniques, and Procedures (TTP) of the MITRE ATT&CK Framework. These TTP's can assist in developing strategies and mitigations of vulnerabilities at a higher level than just the findings themselves.

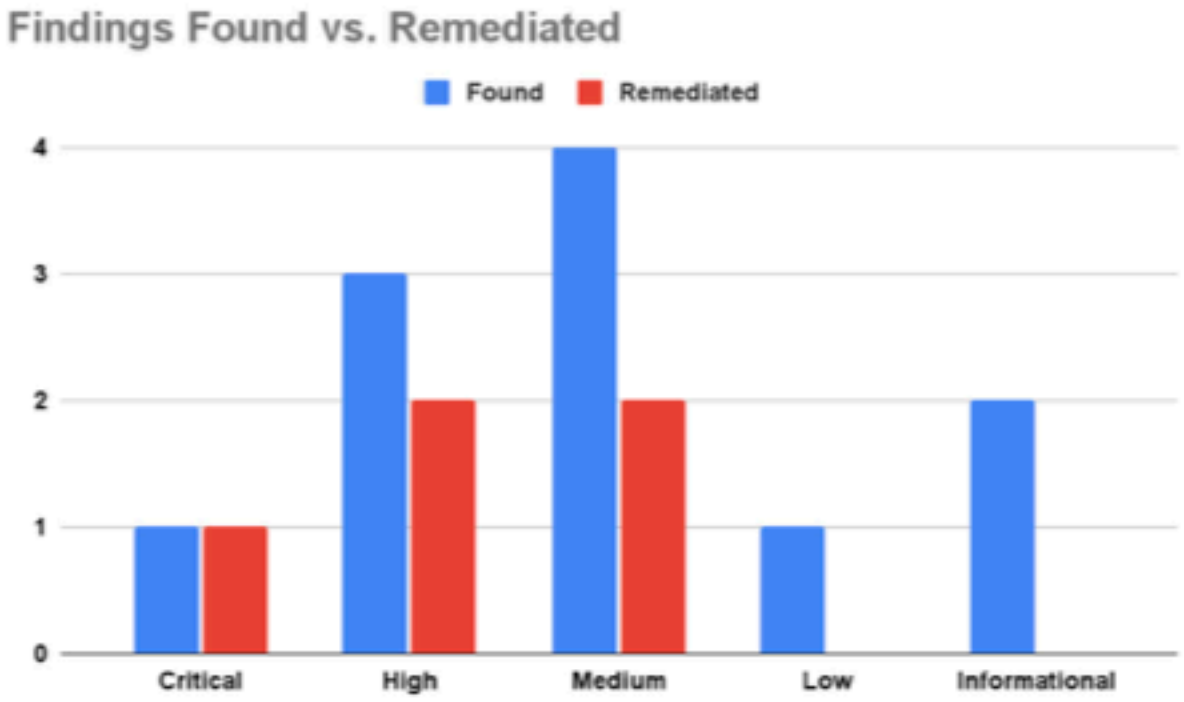


3.2. Vulnerabilities Remediated

This engagement was the second engagement [redacted] has conducted for LBC. [redacted] has re-validated all of the vulnerabilities found during the first engagement to deliver thorough testing. The following table shows vulnerabilities that have, and have not, been remediated since the first engagement. For detail, please refer to the Findings section and Remediations section.

In overview, the LBC IT/Security team worked hard at remediating many of the High and Medium level findings discovered in the previous report. Remediations must be done quickly, but equally as important that they're done well; LBC's IT/Security team has excelled in both areas. Additionally, [redacted] believes that LBC's security team could benefit from additional funding dedicated to hiring and budgeting for in-house security tools.

[redacted] recognizes and appreciates LBC's security team's commitment to security. The column graph below shows the total number of findings encountered during the first engagement compared to findings that have been validated to be remediated. However, it should be noted that there are still some findings unpatched.



Vulnerability	Status
SMB v1 Enabled	Remediated
Poorly Authenticated VNC	Remediated
Outdated Software and Operating System	Remediated
SMB Signing Disabled	Remediated
Unauthenticated Rewards API Access	Remediated
Missing SSL/TLS Implementation	Partially Remediated
HTTP Header Information Disclosure	Partially Remediated
Missing HTTP Security Headers	Partially Remediated
Unauthenticated PostgreSQL Server	Not Remediated
OSINT	Not Remediated
Exposed Rewards API Documentation	Not Remediated

3.3. Key findings

3.3.1. Lack of Authentication

[redacted] noticed a lack of authentication on many services within the LBC network. The team could access multiple sensitive LBC assets, the most severe of which was a MySQL core database that did not require authentication for the root user. The database contained large amounts of sensitive information, including PII, customer invoices, passwords, and login tokens. Access to the database also allowed for the creation and tampering of data.

A vulnerability like this is a major risk for the company's integrity. Exposed PII can negatively affect customers and violate GDPR's Security of Processing standard. Attackers could also use exposed data to conduct phishing attacks on LBC employees and customers.

3.3.2. Lack of Segmentation

With consent from LBC, [redacted] was able to interact with two (2) industrial control systems (ICS) within the network scope. These legacy systems lacked any form of authentication and, despite their sensitive nature, were not segmented from the main subnet. Exposed ICS is a risk for LBC because they can easily be tampered with by an attacker or even crashed with mundane network traffic.

While [redacted] could not confirm the use of these ICS, similar control systems are typically used to interact with large, business-critical machines. These systems could easily be a risk for company security and business revenue.

3.3.3. Poor Web Application Architecture

Many of the web application vulnerabilities that [redacted] identified were introduced due to poor web application architecture choices. Many of these vulnerabilities unnecessarily expose user credentials or PII. This includes using unencrypted network protocols, hard-coded access tokens, credentials encoded in access tokens, credentials being returned by the application, and more.

Architectural security flaws allow attackers to compromise an otherwise functional system and use it to access sensitive data or escalate within an environment. They also incentivize attackers to spend more time analyzing the application rather than moving on to "lower hanging fruit". Poor application design is a risk for both LBC and its customers.

3.4. Key Remediations

The key remediations focus on mitigating vulnerabilities that [redacted] believes require attention to address the potential business impact and regulatory compliance.

3.4.1. Implement Authentication

Ensuring stringent authentication through strong password policies and multi-factor authentication can go a long way in enhancing customer security. [redacted] recommends implementing proper authentication on every server within LBC's internal network but, most importantly, the MySQL database server, which stores LBC customers' PII, payment information, and passwords. Authentication also plays a critical role in regulatory frameworks that LBC might be subject to, especially GDPR and PCI-DSS.

3.4.2. ICS Segmentation

[redacted] recommends that LBC harden access to ICS systems. Specifically, segment critical infrastructure on a separate subnet, implement a proper intermediate authentication, and enforce access control software for accessing ICS systems. Moreover, since the Modbus protocol has no authentication and traffic encryption, long-term mitigation to improve security posture would be to migrate to newer protocols that were developed with security in mind (Ex. Secure Modbus). [redacted] also understands that newer protocols will require more operational overhead and most legacy systems will not support the protocols, so as a short-term solution, implementing the aforementioned remediations will greatly minimize the attack surface.

3.4.3. Secure Code Review

[redacted] found multiple vulnerabilities in web applications introduced to those applications due to poor architectural design and code. [redacted] recommends LBC hire a third party to perform extensive source code review, provide feedback and training to employees on building secure applications. Employing secure applications would help LBC mitigate the vulnerabilities and prevent an attacker from exploiting the web applications in the future. It would also save LBC from suffering monetary loss and negative customer experience.

4. Regulations and Compliance Assessment

4.1. PCI-DSS

Payment Card Industry Data Security Standard (PCI DSS) is a security standard geared towards protecting systems that handle and store payment processing information. It applies to merchants that accept payment information (e.g., credit, debit, or prepaid cards) from customers for their e-commerce requirements.

Based on the Request for Proposal, [redacted] assessed LBC's infrastructure and discovered some deviations in LBC's current security posture from PCI-DSS standards. The table below is the overview of the variations, and the following subsections explain each occurrence in detail.

Req	PCI-DSS Domain	Deviation
R1	Build and Maintain a Secure Network	1. 8.2.2. Tomcat Missing SSL/TLS Implementation 2. 8.3.5. Unrestricted Cross-Origin Resource Sharing (CORS) 3. 8.3.6 Sensitive Info in JWT
R2	Protect Cardholder Data	1. 8.1.1. Unauthenticated MySQL DB 2. 8.2.3. Plain-Text (Base64) Password Storage
R4	Implement Strong Access Control Measures	1. 8.1.2. Unauthenticated Memcached Server 2. 8.2.1. Unauthenticated PostgreSQL Server 3. 8.3.4. Weak Password Policy

4.1.1 - Build and Maintain a Secure Network

PCI-DSS Section	**Build and Maintain a Secure Network** "2.2 Develop configuration standards for all system components. Assure that these standards address all known security vulnerabilities and are consistent with industry-accepted system hardening standards."
Description of the Deviation	Insecure configurations of various services such as overly permissive CORS & lack of TLS certificates was observed.

Mitigation	Such security configurations should be tested against industry-standard hardening guidelines, and correct mitigations should be applied accordingly. Please refer to each finding's mitigation section for specific mitigation strategies.

4.1.2 - Protect Cardholder Data

PCI-DSS Section	**Build and Maintain a Secure Network** "Identify all locations where cardholder data is transmitted or received over open, public networks. Examine documented standards and compare to system configurations to verify the use of security protocols and strong cryptography for all locations."
Description of the Deviation	Unauthenticated access to customer data on the MySQL database and using base64 encoding to store customer passwords.
Mitigation	Access to the databases storing customer data should be protected using strong authentication schemas, and customer passwords should be stored in an irreversible format using hashing and salting.

4.1.3 - Implement Strong Access Control Measures

PCI-DSS Section	**Implement Strong Access Control Measures** "8.2.3 Passwords/phrases must meet the following: Require a minimum length of at least seven characters. Contain both numeric and alphabetic characters. Alternatively, the passwords/phrases must have complexity and strength at least equivalent to the parameters specified above."
Description of the Deviation	Multiple instances of no or weak passwords were seen in LBC's environment, especially in key business impact avenues such as web applications and database servers.
Mitigation	A strong password mechanism should be developed and enforced at all facets of applications within LBC's network.

4.2. GDPR

General Data Protection Regulation (GDPR) is the world's toughest privacy protection law. GDPR is a privacy and security law that provides guidelines for collecting and processing personal information from individuals who live in the European Union (EU). GDPR applies to any organization that collects or processes the personal data of EU citizens.

Based on the Request for Proposal, [redacted] assessed LBC's infrastructure and found some deviations in LBC's current security posture from GDPR standards. The table below shows the overview of the variations, and the following subsections explain each occurrence in detail.

Req	GDPR Domain	Deviation
Article 32(1.b), Article 32 (1.a), Recital 83	Security of Processing	1. 8.1.1 Unauthenticated MySQL DB 2. 8.1.2 Unauthenticated Memcached Server 3. 8.2.1. Unauthenticated PostgreSQL Server 4. 8.2.2 Plain-Text (Base64) Password Storage 5. 8.3.3 Lack of Two Factor Authentication 6. 8.3.4 Weak Password Policy

4.2.1 - Security of Processing

Article 32 (1.b)	States the controller and processor should ensure the ongoing confidentiality, integrity, availability, and resilience of processing systems and services.
Description of the Deviation	During the engagement, various services were found to have minimum or no authentication, which would harm the confidentiality of the data stored if an attacker gets to them. The attacker can also damage the availability of these systems and processes. [redacted] found an unauthenticated database server that had customer PII stored in it. [redacted] also found multiple instances of sensitive information being accessed or transmitted in an insecure manner.
Mitigation	Implement strong authentication mechanisms on the processing systems and services such as strong password policy, account lockout policy and multi-factor authentication

Article 32 (1.a)	States the controller and the processor shall implement appropriate technical and organizational measures to ensure a level of security appropriate to the risk, including inter alias as appropriate the pseudonymization and encryption of personal data.
Description of the Deviation	The customer PII [redacted] found passwords in the database that were stored in plaintext and in such a way that if the attacker gets their hands on it can negatively affect the customers.
Mitigation	Pseudo anonymize the customer PII so that a malicious actor cannot differentiate between the customers even if they get their hands on some customer PII.

Recital 83	States that personal data should be protected during transit, processing, and at rest.
Description of the Deviation	[redacted] found passwords stored with Base64 encoding in the MySQL database. Base64 encoding is very easy to decode, making it easier for a malicious actor to obtain plaintext passwords and use them to gain further access to LBC's network.
Mitigation	Install GDPR compliant data encryption software on processing systems and services, and implement stringent access control with access to personal data storing services on a business-need basis.

5. Response Plan

Based on the Prioritization Metric stated above, [redacted] suggests prioritizing mitigations by following the response plan. The response plan categorizes which vulnerabilities need to be addressed first, based on their technical and business criticality. The response and mitigation prioritization is [redacted]s opinion and suggestion, based on [redacted]s experience in technical assessments and not a definitive solution.

Mitigation Prioritization	Vulnerability
Category - 4 (CAT-4)	N/A
Category - 3 (CAT-3)	• Unauthenticated MySQL Server • Tomcat Missing SSL/TLS implementation • Unsegmented ICS Systems • Hard Coded API Key • Unauthenticated Memcached Server • Unauthenticated PostgreSQL Server • Plain-Text (Base64) Password Storage
Category - 2 (CAT-2)	• Unauthenticated Information Disclosure • Sensitive Information in JWT • Password Returned in Server Response • Directory Indexing • Support for Insecure Ciphers • Unauthenticated Account Information Disclosed • Lack of Two Factor Authentication • Weak Password Policy • Unrestricted Cross-Origin Resource Sharing (CORS)
Category - 1 (CAT-1)	• Missing Security Headers • SSH Misconfiguration • OSINT • Exposed API Documentation

6. Attack Narrative

Friday - 08/01/2021

On 08/01/2021 at 9:45 AM EST, [redacted] was given access to LBC's network through a VDI infrastructure consisting of a Windows and a Kali-Linux host per tester. [redacted] began preparing the VDI infrastructure with the necessary tooling (as detailed in Appendix A) and documentation framework to capture various findings. The targets in scope were hosts within the 10.0.17.0/24 subnet. By 10:00 AM, [redacted] initiated enumeration of the 10.0.17.0/24 subnet using host discovery and service enumeration, while isolating the 10.0.17.50 and 10.0.17.51 hosts (which were out of scope at this point). After initial enumeration, [redacted] started re-validation of findings found in the previous report.

By 11:17 AM, [redacted] gained access to the memcached server on 10.0.17.15. Further enumeration of web applications revealed directory listing enabled on the 10.0.17.12 host. On discovering the unauthenticated MySQL database and analyzing stored data, [redacted] immediately informed LBC at 12:45 PM and gained approval to continue testing. [redacted] also began working on a plan to test the out-of-scope PLC hosts safely to ensure the uptime of LBC's critical infrastructure. At 1:30 PM, [redacted] received test credentials from LBC to test the new e-commerce platform scrumdiddlyumptious and analyze the potential vectors for abuse and revenue loss on it. [redacted] generated a document for LBC's review that outlined the same. Around 5.30 PM, [redacted] was informed of an issue with LBC's public-facing website through their third-party hosting provider.

Saturday- 09/01/2021

At 9:15 AM, [redacted] was granted access to the VDI testing infrastructure for the second day of testing LBC's network. [redacted] began with a rescan of the 10.0.17.0/24 network to check for any new hosts or network changes. [redacted] performed a Root Cause Analysis of the incident with LBC's public-facing website and provided the RCA to LBC for further review. On receiving permission from LBC to engage with the PLC hosts on 10.0.17.50 and 10.0.17.51, [redacted] began to enumerate the services on the hosts carefully. At this point, unauthenticated access to the PLCs was discovered. At 11.42 AM, [redacted] began checking hosts for password reuse (using passwords extracted from the MySQL DB). At 1:03 PM, a hardcoded JWT token was discovered. On discovering inoperational API endpoints, [redacted] contacted LBC and requested the Loompa team investigate it. Around 5 PM, [redacted] delivered a presentation to Wilma Wonka about potential insider threats to LBC.

By 5.30 PM, [redacted] began removing any artifacts uploaded to LBC hosts as part of the engagement wrap-up.

7. Timeline

The timeline is expected to aid LBC's system administrators and security analysts to monitor the penetration tester's activities during and after the engagement. The timeline below is a record of [redacted]s activity during the engagement. If the engagement occurred over multiple days, each day is noted appropriately.

TIME (EST)	ACTIVITY
01/07/2022 - Friday	
10:00:36	All testers given permission to access the infrastructure
10:15:36	Initial enumeration on all network in-scope begins
10:31:36	Started preparing a detailed plan to scan and test the ICS applications
10:45:00	Engaged with the LBC ICS Team for Q&A to prepare for the plan
11:09:00	Found Directory listing on 10.0.17.12
11:17:00	Discovered and gained access to the memcached server on 10.0.17.15
12:01:05	Gained access to the MySQL database server on 10.0.17.14
12:45:15	Sent a ticket to the point of contact regarding the customer PII found in the MySQL database server
12:20:15	Found Base64 encoded passwords stored in the MySQL database server and decoded them to obtain plaintext passwords
13:18:21	Performed Directory Enumeration for Web applications
13:30:00	Received test credentials for scrumdiddlyumptious from the point of contact to perform a test on the store
14:09:00	Received Permission to perform privilege escalation attempt on scrumdiddlyumptious
15:00:00	Submitted E-Commerce Site Launch Use Case Analysis detailing three scenarios where users can mess with the store
16:45:00	Cleaned up the artifacts dropped in the system during the engagement
17:30:00	[redacted] informed of incident with LBC's public facing site
17:45:00	End of testing for first day
01/08/2022 - Saturday	
9:15:00	All testers given permission to access the infrastructure
9:31:58	Received permission to test ICS applications and scan started to find out

	the ICS applications
9:45:00	Root Cause Analysis provided to LBC over the incident.
10:40:11	Vulnerabilities found in the ICS applications during the test
10:58:51	Started Password Spraying the Web Services using the password found in the MySQL database server along with some common passwords
11:35:00	Wrote an email advising Wilma Wonka on the steps to take to recover from the ransomware attack she suffered.
11:42:18	Started Password Spraying SSH using the password found in the MySQL database server along with some common passwords
13:03:03	Discovered Hardcoded JWT Token
13:33:26	Lost access to the Kali VDI
14:34:00	Gained access to the Kali boxes again
15:12:31	Sent a ticket to the point of contact regarding the working of LBC store
17:05:03	Gave a quick presentation to Wilma Wonka on Insider Threat and steps LBC should consider taking to minimise it.
17:45:00	Hands off keyboard

8. Findings

This section provides detailed information for each vulnerability found during the penetration test. All of the findings are sorted by their risk level.

8.1. Critical

N/A

8.2. High

8.2.1. Unauthenticated MySQL DB

Unauthenticated MySQL DB		CVSS	Prioritization
Risk	**Critical**	**8.8** High	**CAT-4**
Impact	**Critical**		
Likelihood	**Likely**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H		
MITRE ATT&CK	T1110.001 - Brute Force: Password Guessing		
Hosts	10.0.17.14 (3306/tcp)		
History	2.0 - Vulnerability found		

Impact
LBC wouldn't just face monetary loss due to unavailable services but would also suffer from reputational damage and risks of regulatory fines due to the leaked customer data. This vulnerability also provides an attack vector for insider threats. A disgruntled employee could abuse their access and exfiltrate customer data to cause business loss.

Details

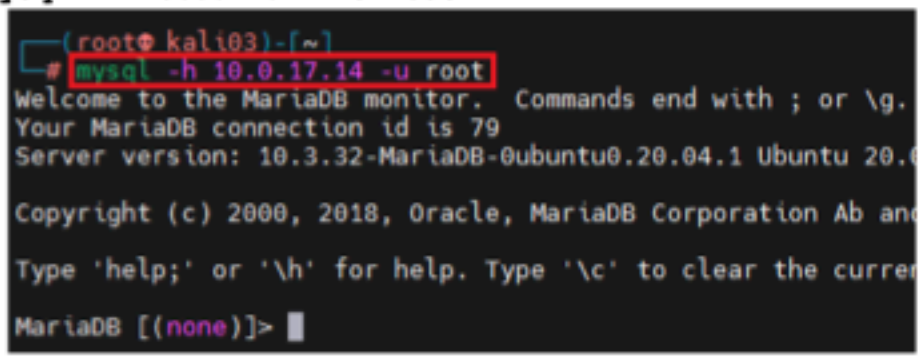
[redacted] discovered that the MySQL database was not secured and did not require a user "root" password. This allowed [redacted] to log in to the MySQL server and arbitrarily read and modify the information at will.

Absence of proper authentication while connecting to the MySQL databases would allow the attacker to access the database with root privilege, which means the attacker can read the sensitive customer data stored in the database. They can also modify or delete the database, resulting in the integrity of data being compromised or, in the worst-case scenarios, the systems connecting to the database crash and through an error.

[redacted] ascertained that given the ease of gaining access to the server with admin privileges, the likelihood is very high. The impact is critical since this server is a core database server of LBC, which stores extremely sensitive customer information.

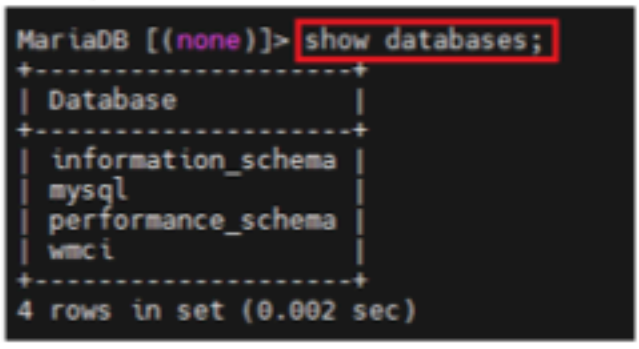
Replication

1. Connect to the MySQL database using the user "root".
   # mysql -h 10.0.17.14 -u root



MySQL Server Command Line Client

2. List all the databases the MySQL user has read permission.
   # show databases;



Reading Databases Present In The MySQL Server

3. Reading sensitive information is possible with the following command.

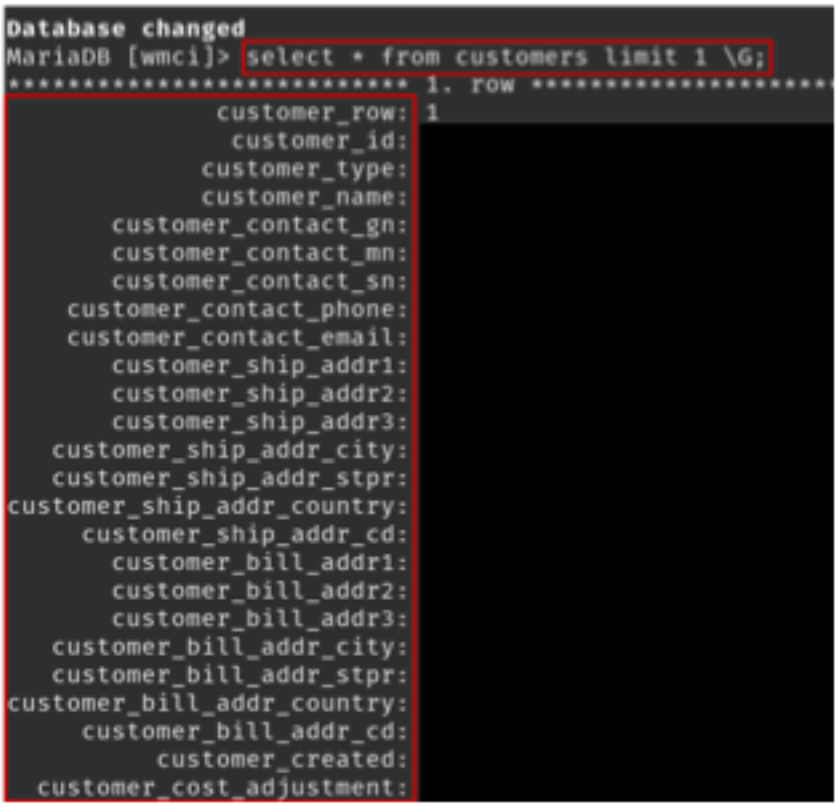
```
# use [database];
# show tables;
# select * from [table];
```

```
MariaDB [information_schema]> use wmci;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [wmci]> show tables;
+---------------------+
| Tables_in_wmci      |
+---------------------+
| customer_types      |
| customers           |
| invoice_items       |
| invoice_payments    |
| invoice_statuses    |
| invoices            |
| item_category_types |
| items               |
| login_role_types    |
| logins              |
| payment_statuses    |
| payment_types       |
| payments            |
| tokens              |
| unit_types          |
+---------------------+
15 rows in set (0.001 sec)

MariaDB [wmci]>
```

Accessing tables stored in the database



Accessing Customer PII

Mitigation
1. [redacted] recommends creating a strong password longer than 15 characters for the "root" user to remediate this vulnerability. 2. Furthermore, a stricter access control list should be implemented for the database user as it is a superuser. 3. Only allow certain IP addresses to connect to the database.

References
1. https://dev.mysql.com/doc/refman/8.0/en/resetting-permissions.html

8.2.2. Tomcat Missing SSL/TLS Certificate

Tomcat Missing SSL/TLS Certificate		CVSS	Prioritization
Risk	High	8.5	CAT-3
Impact	High		

Likelihood	Likely	High	
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:N		
MITRE ATT&CK	T1210 - Exploitation of Remote Services		
Host	10.0.17.50 (9090/tcp)		
History	1.0 - Vulnerability found 2.0 - Vulnerability not remediated		

Impact
Anyone on LBC's network could obtain login credentials and other sensitive information by capturing network traffic. This could lead to compromise of LBC infrastructure, causing significant business impact. As this host has an authentication prompt and is also a critical infrastructure, HTTPS must be enabled.

Details
During the previous engagement [redacted] found that HTTPS is not configured or forced for any of the applications on the 10.0.17.0/24 subnet. The HTTP protocol sends all data across the network in plain text, meaning that any login credentials or data from the application could be sniffed using a packet analyzer by an attacker. An attacker must be suitably positioned to eavesdrop on the victim's network traffic to exploit this vulnerability. This scenario typically occurs when a client communicates with the server over an insecure connection such as public Wi-Fi or a corporate or home network shared with a compromised computer. HTTPS uses SSL/TLS to encrypt all communications sent to the server, reducing the likelihood of the vulnerabilities mentioned above being exploited. [redacted] classifies this finding as High risk, as lack of in-transit network encryption exposes these applications to traffic sniffing.

Replication
1. Browse to ***http://10.0.17.50:9090***

Tomcat Landing Page Missing SSL Cert

Mitigation
1. As an immediate short-term fix, [redacted] suggests deploying self-signed certificates on the webserver and enabling HTTPS using industry-standard cipher suites at layer 7 such as TLS 1.2 or TLS 1.3. 2. [redacted] recommends that LBC set up its own internal Root CA and generate SSL certificates for all HTTP web servers on the LBC internal network as a long-term solution. 3. After HTTPS is implemented, force all HTTP requests to redirect to HTTPS. This can be accomplished via the 'HSTS' HTTP header to ensure that only HTTPS is being used or per specific web server documentation.

References
1. https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/hh831740(v=ws.11) 2. https://nginx.org/en/docs/http/configuring_https_servers.html

8.2.3. Unsegmented ICS Systems

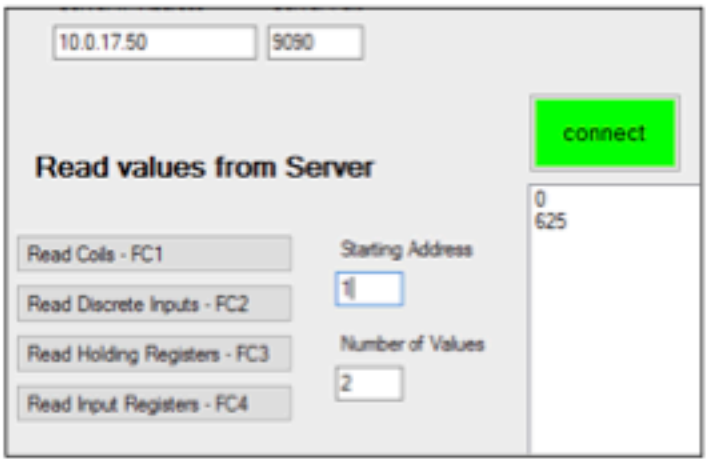
Unsegmented ICS Systems		CVSS	Prioritization
Risk	Critical	8.3 High	CAT-4
Impact	Critical		
Likelihood	Very Likely		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:L/I:H/A:H		
MITRE ATT&CK	N/A		
Hosts	10.0.17.50 (9090/tcp), 10.0.17.51 (2021/tcp)		
History	2.0 - Vulnerability found		

Impact

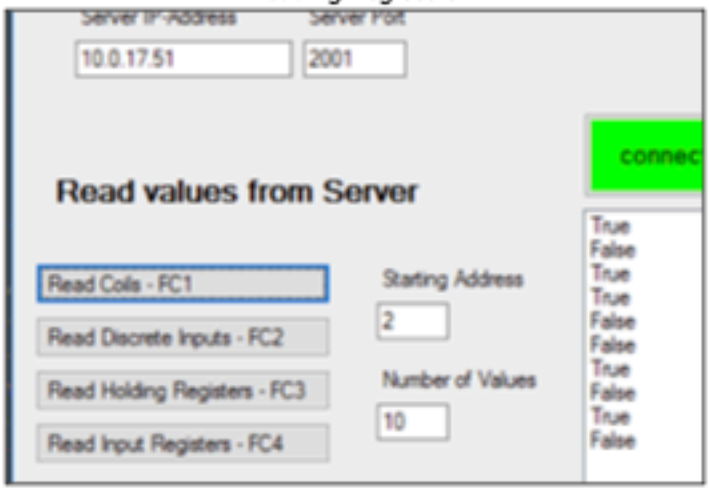
[redacted] connected to LBC's ICS's systems and subsequently read data from the coils, potentially replicating the device state. This would further allow an adversary to research and craft complex attacks. Also, these devices are often legacy systems. An attacker could cause a denial of service by sending random network traffic. Considering the above scenarios, the business impact to LBC due to this vulnerability is considered Critical.

Details
After working with the LBC team to develop a testing plan, [redacted] received authorization to test the ICS systems. The testing was kept to be as minimally intensive as possible to avoid any potential downtime. [redacted] identified that any authentication layer did not protect the ICS devices, and we were able to connect to the coils and read data from them, as shown below. The attack complexity is very low, and open-source tools are available, due to which the likelihood of this vulnerability being exploited is Likely. Considering the impact detailed above and the likelihood the overall risk evaluates to Critical.

Replication
1. Enter the following telnet command in kali-linux to connect to the devices: **# pymodbus.console tcp --host 10.0.17.50 --port 9090**      *Successful Connection To ICS Device* 2. Next, [redacted] used the *easymodbusTCP* application to read data from the coils as shown below.



Reading Registers



Reading Coils

Mitigation
1. [redacted] recommends that all the ICS systems be segmented into a separate subnet protected by an authentication layer. 2. Implementing a comprehensive firewall to block any traffic that could crash the devices and avoid a potential denial of service is also recommended.

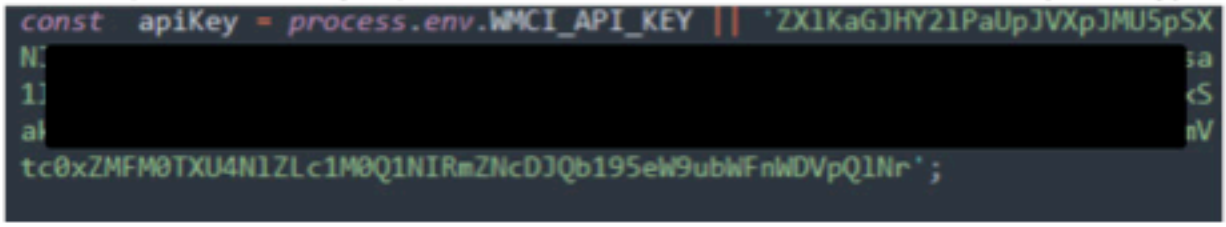
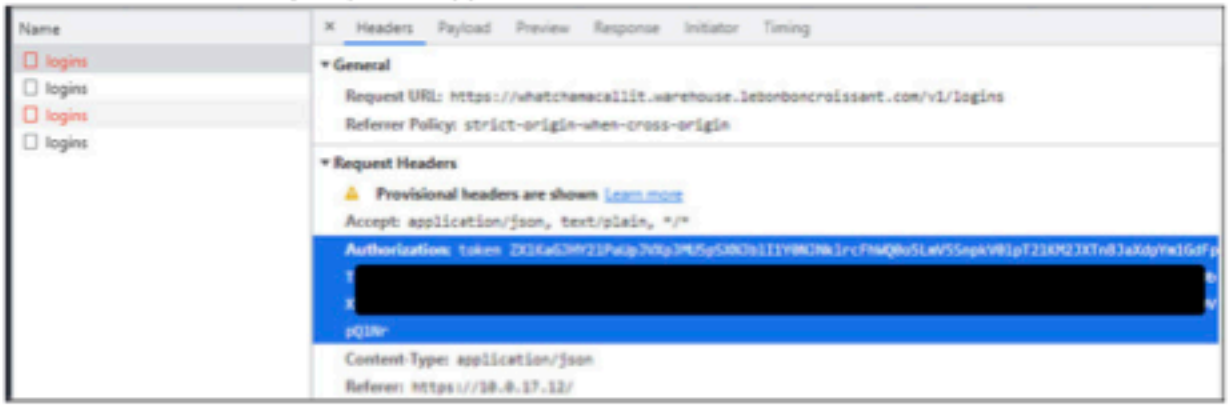
References
1. https://spectrum.ieee.org/telecom/security/the-real-story-of-stuxnet 2. https://spectrum.ieee.org/telecom/security/the-real-story-of-stuxnet 3. https://github.com/riptideio/pymodbus 4. https://github.com/rossmann-engineering/EasyModbusTCP.NET

8.2.4. Hard Coded API Key

Hard Coded API Key		CVSS	Prioritization
Risk	**High**	**8.1 High**	**CAT-3**
Impact	**High**		
Likelihood	**Likely**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N		
MITRE ATT&CK	T1552 - Unsecured Credentials		
Hosts	10.0.17.12 (443/tcp)		
History	2.0 - Vulnerability found		

Impact
This finding may result in an unauthorized threat actor being able to submit authorized API calls to the whatchamacallit (10.0.17.13) host. Whatchamacallit has functions that return customer PII that an attacker could steal. This could severely impact the reputation of LBC, damage customer loyalty, and even lead to fines due to GDPR. It is important to note that [redacted] could not successfully show the unauthorized use of the /v1/customer API endpoint to obtain customer PII due to an internal server error with the whatchamacallit host. However, [redacted] strongly believes that this threat is still very likely should the API resume normal behavior.

Details
[redacted] discovered a hard-coded API key within the application on 10.0.17.12 (scrumdiddlyumptious) that is used to authorize API calls on the 10.0.17.13 (whatchamacallit) host. Hard-coded credentials typically create a significant hole that allows an attacker to bypass the authentication that the software administrator has configured. Scrumdiddlyumptious is what the OAuth2 Spec refers to as a public client. Public clients are clients that are "incapable of maintaining the confidentiality of their credentials (e.g., clients executing on the device used by the resource owner, such as an installed native application or a web browser-based application), and incapable of secure client authentication via any other means." Because scrumdiddlyumptious cannot maintain the confidentiality of the API key, it is trivial for an attacker to obtain this key and use it to submit authorized API calls to the whatchamacallit host.

Replication
The hard coded key can be found in two places: 1. The API key can be found on line one in Config.js within the source code of ***https://scrumdiddlyumptious.warehouse.lebonboncroissant.com/static/js/Config.js***    *Hard Coded Token in Config.js Source Code* 2. The API key is used in the Authorization header sent when a request is generated from the scrumdiddlyumptious application.    *Hard Coded Token in Authorization Header*

Mitigation
1. As an immediate band-aid solution, steps should be taken to remove the hard-coded key from the client application. This can be accomplished by following recommendations outlined in CWE-798, "store passwords, keys, and other credentials outside of the code in a strongly-protected, encrypted configuration file or database that is protected from access by all outsiders, including other local users on the same system. Properly protect the key (CWE-320). If you cannot use encryption to protect the file, then make sure that the permissions are as restrictive as possible [REF-7]." 2. Another option, [redacted] recommends that the scrumdiddlyumptious application and the whatchamacallit back-end be re-architected to use OAuth2 Access Tokens since JWT's are not commonly used for API keys. 3. However, the most secure solution is to remove the API key entirely and restructure the

application such that all authorization is stored server-side via techniques such as sessions.

References
1. https://cwe.mitre.org/data/definitions/798.html 2. https://datatracker.ietf.org/doc/html/rfc6749#section-2.1 3. https://medium.com/@robert.broeckelmann/oauth2-access-tokens-vs-api-keys-using-jwts-651f97df9e19

8.2.5. Unauthenticated Memcached Server

Unauthenticated Memcached Server		CVSS	Prioritization
Risk	**High**	**8.1** **High**	**CAT-3**
Impact	**High**		
Likelihood	**Likely**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:H		
MITRE ATT&CK	T1078 - Valid Accounts		
Hosts	10.0.17.15 (11211/tcp)		
History	2.0 - Vulnerability found		

Impact
Memcached is open-source software, and the service is used to reduce the database query time by storing information in its cache. This information could be highly critical to an organization depending on the use case. [redacted] did not observe any sensitive information cached within the server during the engagement. However, depending on the use case, it could contain highly critical information to LBC such as customer payments, physical addresses, PII, NPI, etc. It could also contain credentials or post-authenticated tokens that are used by the various PLC devices. If these devices were to be compromised by an adversary, it might cause downtime, directly impacting LBC's bottom line.

Details

[redacted] identified that the host 10.0.17.15 was running a Memcached server on port 11211 without any authentication layer or a firewall. [redacted] was able to connect to the server without any credentials using telnet. Once connected, we extracted critical information such as the service version, uptime, current slab-stats, max-connections, etc.

The attack complexity is low as it requires telnet. There are various open-source documentation/articles on exploiting and extracting information from Memcached servers, making exploiting this vulnerability likely. Considering the high business impact, which is further described in the below section, the overall risk can be evaluated to be high.

Replication

1. [redacted] ran the "memcached-info" nmap script to pull information about the machine and determine the authentication status.

```
# nmap -n -sV -script memcached
```

```
┌──(root💀kali06)-[~]
└─# nmap -n -sV --script memcached-info -p 11211 10.0.17.15
Starting Nmap 7.92 ( https://nmap.org ) at 2022-01-07 12:28 EST
Nmap scan report for 10.0.17.15
Host is up (0.0038s latency).

PORT      STATE SERVICE   VERSION
11211/tcp open  memcached Memcached 1.5.6 (uptime 35606 seconds; Ubuntu)
| memcached-info:
|   Process ID: 8767
|   Uptime: 35606 seconds
|   Server time: 2022-01-07T17:28:36
|   Architecture: 64 bit
|   Used CPU (user): 2.857721
|   Used CPU (system): 2.822019
|   Current connections: 1
|   Total connections: 12
|   Maximum connections: 1024
|   TCP Port: 11211
|   UDP Port: 0
|_  Authentication: no
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Nmap Memcached-info Script Output

2. Enter the following telnet command to connect to the memcached server.

```
# telnet 10.0.17.15 11211
```

```
┌──(root💀kali03)-[~]
└─# telnet 10.0.17.15 11211
Trying 10.0.17.15...
Connected to 10.0.17.15.
Escape character is '^]'.
```

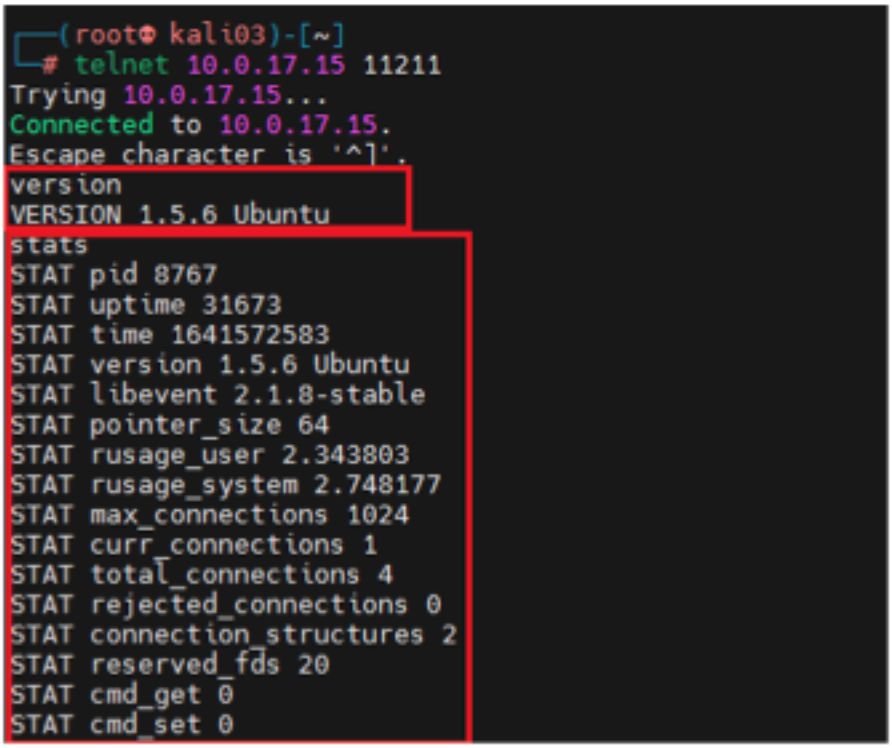
Connecting To The Memcache Server Using Telnet

3. Once connected, enter the command below to display version information.

```
# version
```

4. Enter the command below to display the server's current statistics information.

```
# stats
```



Results For Version & Stats Commands

Mitigation
1. [redacted] recommends that the server be placed behind string authentication and a load-balancing layer to prevent unauthorized access and potential DoS attacks. 2. It is advised to constantly update the server to the latest stable release and stay up to date on security issues.

References
1. https://cwe.mitre.org/data/definitions/306.html 2. https://www.hackingarticles.in/penetration-testing-on-memcached-server/ 3. https://www.digitalocean.com/community/tutorials/how-to-secure-memcached-by-reducing-exposure

8.2.6. Unauthenticated PostgreSQL Server

Unauthenticated PostgreSQL Server		CVSS	Prioritization
Risk	**High**	**7.6** **High**	**CAT-3**
Impact	**High**		
Likelihood	**Likely**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:L/A:L		
MITRE ATT&CK	T1078 - Valid Accounts		
Hosts	10.0.17.14 (5432/TCP)		
History	1.0 - Vulnerability found 2.0 - Vulnerability not remediated		

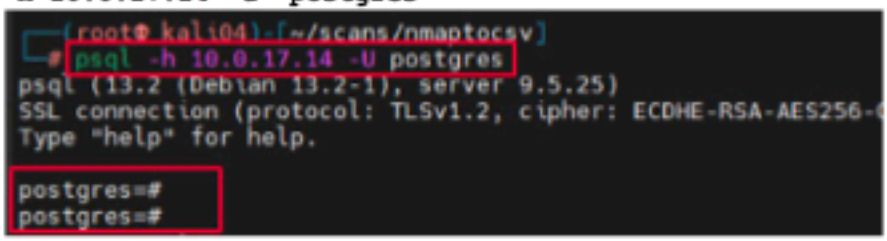
Impact
Although this finding did not yield any customer data, it allows a malicious actor a staging area to further probe LBC's network, bypassing potential firewall rules that would have otherwise protected the internal LBC network. This could also have potential PCI-DSS compliance implications, exposing LBC to regulatory risk.

Details
[redacted] was able to login to the PostgreSQL service running on host 10.0.17.14 with the default username and no password. [redacted] found that the database was empty on connecting to the server, but it would have been a serious breach if it contained any data. Also, It is possible to execute shell commands on the database host via pg_execute_server_program privilege and COPY external command. [redacted] was then able to extract server information, enumerate users, check and modify database configuration files, and more An attacker could deploy malware, a command-and-control service, or use the host for more attacks. The complexity of the attack is characterized as low, as a readily available Metasploit module can be used to exploit this, rendering the likelihood of exploitation as likely. [redacted] ascertained the likelihood that this will be exploited as high since there is no authentication. As such, [redacted] characterizes the impact and risk of this vulnerability to be high.

Replication

1. The tester used *psql* to authenticate to the server using default credentials .

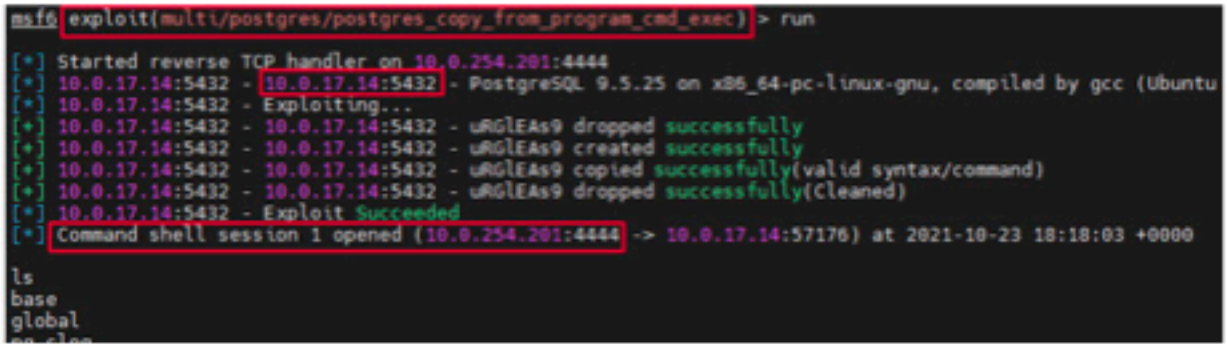
```
# psql -h 10.0.17.14 -u "postgres"
```



PostgreSQL Server Command Line Client

2. [redacted] used a Metasploit module to exploit this vulnerability

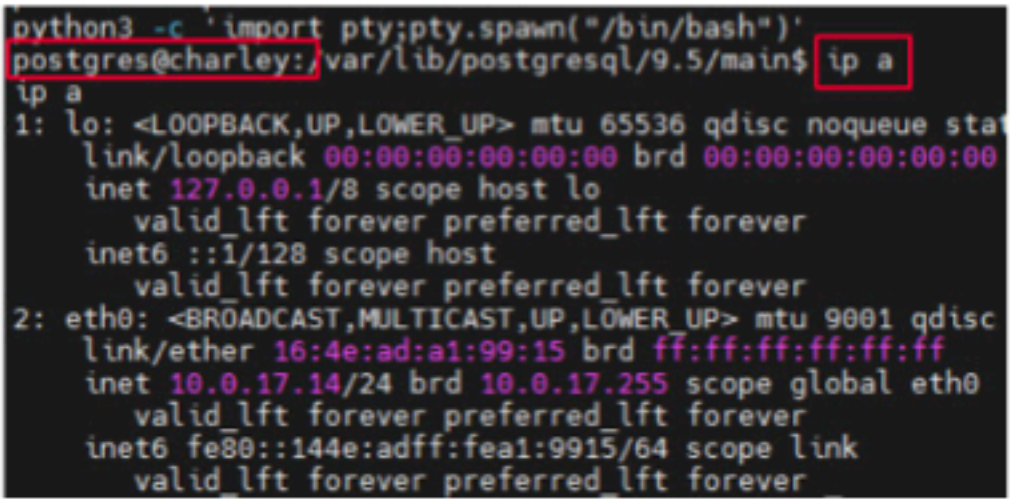
```
# msfconsole
# use multi/postgres/postgres_copy_from_program_cmd_exec
# set RHOSTS 10.0.17.14
# run
```



Metasploit Module Used to Execute Shell Commands on The Database Host

3. Python was used to upgrade the shell to include PTY.

```
# python3 -c 'import pty;pty.spawn("/bin/bash")'
```



Demonstrating Command Execution

Mitigation
1. Enable password-based authentication mechanisms and also change the default credentials.

References
1. https://www.postgresql.org/docs/10/auth-methods.html

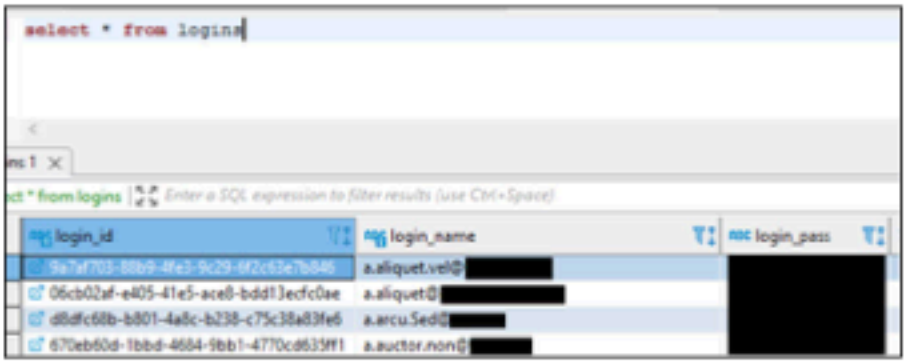
8.2.7. Plain-Text (Base64) Password Storage

Plain-Text (Base64) Password Storage		CVSS	Prioritization
Risk	**High**	**7.3 High**	**CAT-3**
Impact	**High**		
Likelihood	**Likely**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N		
MITRE ATT&CK	T1552 - Unsecured Credentials		
Hosts	10.0.1.14 (3306/tcp)		
History	2.0 - Vulnerability found		

Impact
This vulnerability exposes customers' passwords in plaintext, putting them at significant risk of reusing passwords. Not only does this expose LBC to reputational loss, but it also exposes LBC to regulatory risk in terms of PCI-DSS compliance.

Details
On further analysis of the database found in host 10.0.1.14, [redacted] found that the customer passwords stored in the logins table within the wmci database are stored in plaintext, albeit Base64 encoded. This form of storage is merely an encoding format and can be easily decoded, as demonstrated in the screenshots below. A potential attacker could extract real-world passwords of customers, which could be very dangerous if they are reusing the same passwords across other non-LBC services. As Base64 decoding is trivial, combined with the unauthenticated MySQL database finding, it is likely that this vulnerability will be exploited. Given these factors, [redacted] characterizes the risk of this finding to be High.

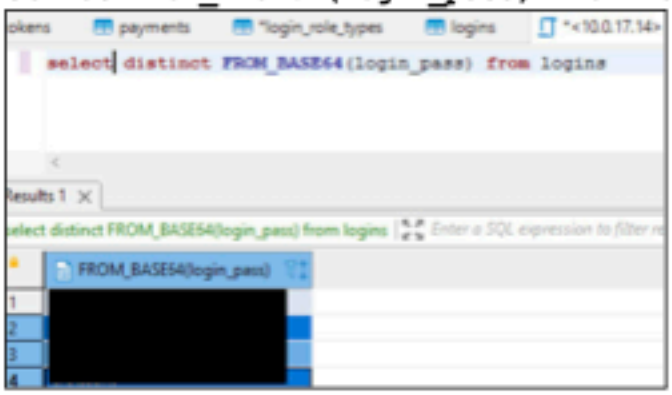
Replication
1. Connect to the MySQL database as described in *8.1.1 Unauthenticated MySQL DB* and execute the following query. `# select * from logins`



Customer Login Information

2. The passwords can be decoded using any base64 decoder. In this example, the MySQL *from_Base64* function is used.

```
# select distinct FROM_BASE64(login_pass) from logins
```



Plaintext Customer Passwords In Database

Mitigation
1. Immediately remove the cleartext passwords and force the affected accounts to change their password. 2. If there is an indication of compromise, inform all affected parties immediately. 3. Implement a GPU-resistant hashing algorithm such as Argon2, PBKDF2, Scrypt, or Bcrypt. Avoid any "fast" hashing algorithms such as MD5 or any of the SHA-family of hash functions. Each hash should have a unique salt, further increasing the computing power needed to crack a majority of the hashes. These measures ensure that in the event of a breach, the risk of each user's password hash being cracked is limited, allowing more time for affected users to change their passwords.

References
1. https://cwe.mitre.org/data/definitions/256.html

2. https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html

8.3. Medium

8.3.1. Unauthenticated Information Disclosure

Unauthenticated Information Disclosure		CVSS	Prioritization
Risk	**High**	**6.5 Medium**	**CAT-3**
Impact	**High**		
Likelihood	**Likely**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N		
MITRE ATT&CK	T1078 - Valid Accounts		
Hosts	10.0.17.10 (443/tcp)		
History	2.0 - Vulnerability found		

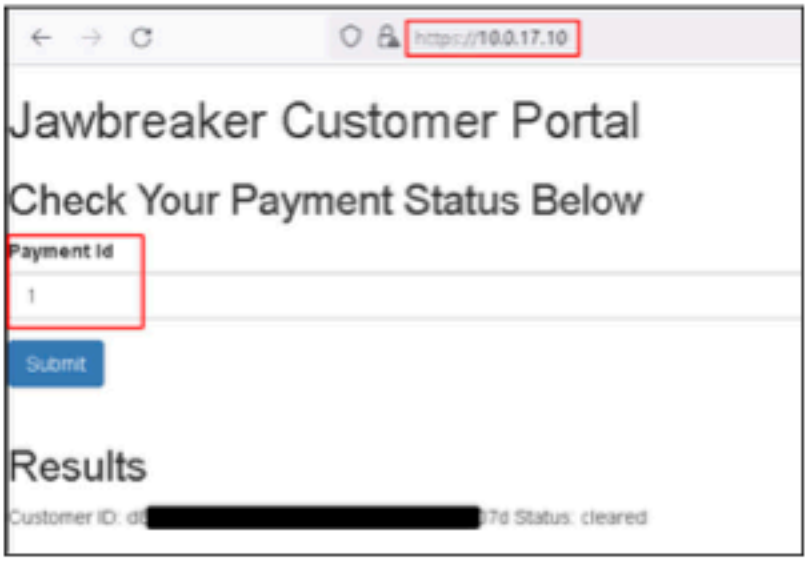
Impact
This finding identifies information disclosure that can lead to customer PII being leaked to an attacker. Such a scenario could severely impact the reputation of LBC, damage customer loyalty, and even lead to fines due to GDPR regulation. It is important to note that [redacted] could not successfully show the unauthorized use of the /v1/customer API endpoint to obtain customer PII due to an internal server error with the whatchamacallit host. However, [redacted] strongly believes that this threat is still very likely should the API resume normal behavior.

Details
[redacted] discovered an unauthenticated internal tool, Jawbreaker Customer Portal, that disclosed information about a customer. Some of this information could obtain sensitive customer PII, specifically using the Customer ID. Customer IDs obtained from the Jawbreaker tool can be passed as arguments to the /v1/customer API endpoint of the whatchamacallit (10.0.17.13) using the hard-coded API key outlined in 8.2.4 or by modifying a request legitimately sent from scrumdiddlyumptious (10.0.17.12). This API function returns customer PII that an attacker could steal. In addition, the amount of a payment object seems to be inadvertently leaked in the HTTP response of the Jawbreaker tool. An attacker could use this to enumerate which Customer IDs

to target by identifying which customers were completing large transactions. Due to the simplicity of exploitation and the severity of possible exposed PII, the overall risk of vulnerability is high.

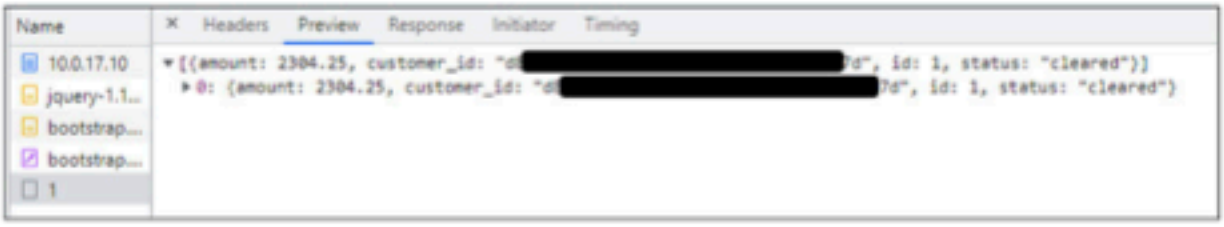
Replication

1. Access the jawbreaker application and submit any arbitrary payment ID. These can be known, guessed or brute forced.



Jawbreaker Customer Portal

2. Examine the response data in Developer Tools



Information Leak of Payment Amount

3. Submit the returned Customer ID to the /v1/customer endpoint. The 504 server error represents the PII that [redacted] expects to observe should the system resume normal behaviour. The API key must be included in the Authorization header in order to successfully call the API.

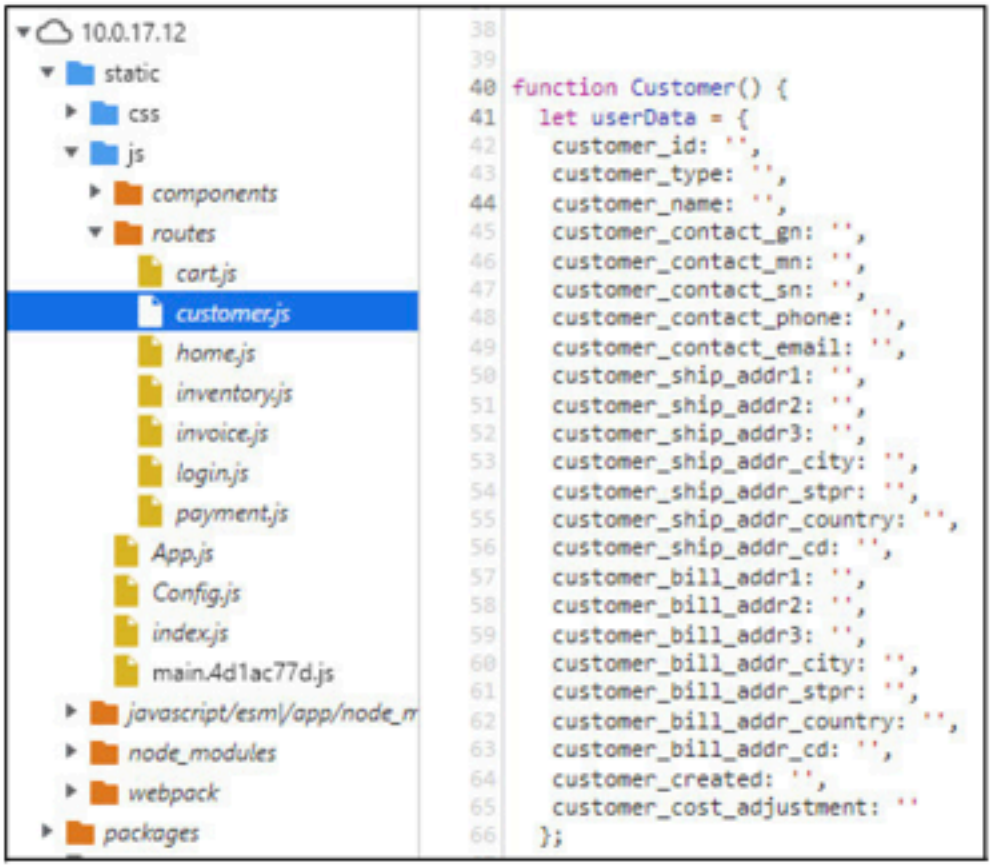
```
# curl -k 'https://10.0.17.13/v1/customer/{customer_id}' \
#  -header 'Authorization: token {API key}' \
#  -header 'Content-Type: application/json'
```

```
curl -k 'https://10.0.17.12/v1/customer/f03d22d2-01e1-43cb-b133-65cc6e8473
91' --header 'Content-Type: application/json' --header 'Authorization: token ZXlKaGJHY2lPaUpJVXpJMU5pSXNJbll1Y0NJNklrcFhWQ0o5L
m%                                                                                                    VmxSak5hZFd
Gc                                                                                                    Nr'
<html>
<head><title>504 Gateway Time-out</title></head>
<body>
<center><h1>504 Gateway Time-out</h1></center>
<hr><center>nginx</center>
</body>
</html>
```

Predicted API Call to Return PII

```
curl -k 'https://10.0.17.12/v1/customer/f03d22d2-01e1-43cb-b133-65cc6e8473
91' --header 'Content-Type: application/json'
{"code":403,"msg":"missing authentication"}
```

API Response with Missing Authentication



The Expected PII Output From tThe /v1/customer Endpoint

Mitigation
1. Since Customer IDs can easily be used to obtain PII, they must be treated as sensitive information. Authentication should be implemented on the internal Jawbreaker tool, and the principle of least privilege should be followed by only allowing access to employees who need it.

2. In addition, unnecessary information should not be included in server responses, even if it is not directly outputted to the user.
3. While the impact of this finding may be drastically reduced by security controls implemented by the API endpoint, it is still important to use a layered approach to improve security posture.

References
1. https://cwe.mitre.org/data/definitions/798.html 2. https://datatracker.ietf.org/doc/html/rfc6749#section-2.1 3. https://medium.com/@robert.broeckelmann/oauth2-access-tokens-vs-api-keys-using-jwts-651f97df9e19

8.3.2. Sensitive Information in JWT

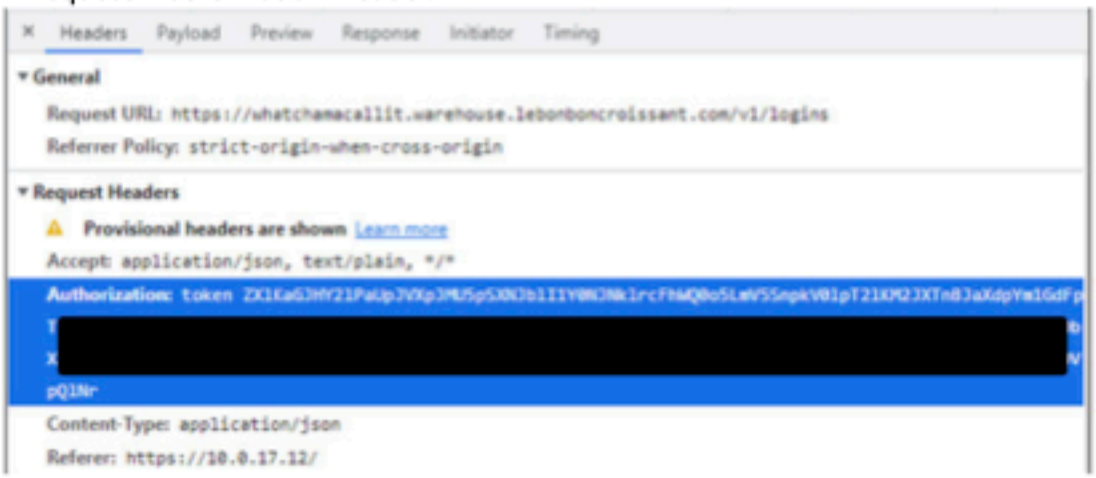
Sensitive Information in JWT		CVSS	Prioritization
Risk	**Medium**	**6.5 Medium**	**CAT-2**
Impact	**Medium**		
Likelihood	**Unlikely**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N		
MITRE ATT&CK	T1078 - Valid Accounts		
Hosts	10.0.17.12 (443/TCP)		
History	2.0 - Vulnerability found		

Impact
JSON Web Tokens (JWTs) are often cryptographically signed, but they are stored in plain text. If an attacker is able to recover a JWT, whether or not it is still valid, they will be able to read the credentials stored within it. They can then use those credentials to access the account, gain permissions in the environment, and view other sensitive data.

Details
JWTs used by the web application contained a "name" and "pw" field in their payload. These fields can be interpreted as a username and password associated with the token. [redacted] did not witness the "pw" field being used within the application and thus should be stored server-side hashed inside a database. NOTE: [redacted] could not confirm the validity of these credentials within the testing environment.

Replication

1. Visit ***https://10.0.17.12/customers/*** in a web browser.
2. Login to the customer portal.
3. Inspect the page and view the network tab. The base64-encoded JWT will be in the request "Authorization" header.



Base64-encoded JWT in the Authorization header

4. Decode the JWT into UTF-8.
5. Decode the JWT again to view the username and password within the payload field.

Plaintext credentials in payload field of the JWT

Mitigation
1. Modify the backend web application logic to no longer store credentials within the JWT payload. 2. For security purposes, change any credentials that may have been exposed as a result of this vulnerability.

References
1. https://www.ibm.com/docs/en/order-management-sw/10.0?topic=features-jwt-authentication

8.3.3. Password Returned in Server Response

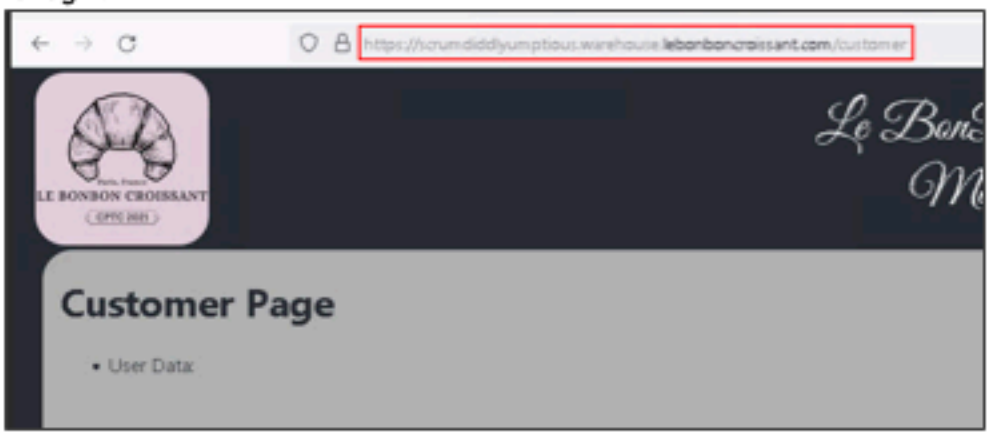
Password Returned in Server Response		CVSS	Prioritization
Risk	**High**	**5.8 Medium**	**CAT-3**
Impact	**High**		
Likelihood	**Unlikely**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:L/UI:R/S:U/C:H/I:L/A:N		
MITRE ATT&CK	T1555 - Credentials from Password Stores		
Hosts	10.0.17.12 (443/tcp)		
History	2.0 - Vulnerability found		

Impact
This finding may lead to the disclosure of users' passwords to unauthorized threat actors, resulting in a full compromise of a user's account. This would allow a malicious actor to impersonate a user and access any of the user's saved data. In addition, this type of compromise would be quite difficult to investigate since audit trails are obscured. It may also be likely that a user has reused the compromised password elsewhere, so the impact of this vulnerability could severely affect LBC's users outside the scope of the scrumdiddlyumptious web application. This could cause large reputational damage to LBC's brand and customer loyalty.

Details
[redacted] discovered base64 encoded passwords returned in server responses upon successful login attempts to the /v1/logins endpoint on the 10.0.17.13 host. This behavior increases the risk that an attacker will capture users' passwords. Base64 encoding is trivial to decode to plaintext and is not a substitute for encryption or hashing. Since this behavior was only noticed on the /v1/logins endpoint, which requires user interaction to perform a successful login, it is unlikely that this vulnerability would be exploited. However, since the password is essentially transmitted in plaintext, the impact should the vulnerability be exploited would be high. Thus, the overall risk is high.

Replication

1. Visit the customer page on ***scrumdiddlyumptious.warehouse.lebonboncroissant.com*** and login.



Customer Page After Login

2. Examine the data in the HTTP response using the built in browser console.

Login Password Sent In Base64

3. Decode the base64 encoded password to identify the plaintext password.

```
# echo "<password>" | base64 -d
```

Mitigation
A user's password should not be included in the HTTP response upon successful login. Even if the password is hashed rather than encoded, weak hashes may still be cracked, and there is usually no reason to include the credential in server responses.

References
1. https://portswigger.net/web-security/information-disclosure 2. https://cwe.mitre.org/data/definitions/204.html 3. https://capec.mitre.org/data/definitions/37.html

8.3.4. Directory Indexing

Directory Indexing		CVSS	Prioritization
Risk	Low	5.3 Medium	CAT-2
Impact	Low		
Likelihood	Likely		
CVSS String	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N		
MITRE ATT&CK	T1594 - Search Victim-Owned Websites		
Host	10.0.17.12 (443/TCP)		
History	2.0 - Vulnerability found		

Impact
Directory indexing can aid an attacker by quickly identifying the resources at a given path and proceeding directly to analyzing and attacking those resources. It increases the exposure of sensitive files within the directory that are not intended to be accessible to users, such as temporary files and crash dumps.

Details
Web servers can be configured to automatically list the contents of directories that do not have an index page present. Directory listings themselves do not necessarily constitute a security vulnerability. Any sensitive resources within the web root should have proper access controls and only be accessible by authenticated users.

Replication

1. Visit the /static and /inventory endpoints at https://10.0.17.12

Files Listed In The Inventory Directory

Files Listed In The Static Directory

Mitigation

1. Configure the affected webserver to prevent directory listings for all paths beneath the web-root. In Nginx this is the "autoindex" module.

References

1. https://portswigger.net/kb/issues/00600100_directory-listing

8.3.5. Support for Insecure Ciphers

Support for Insecure Ciphers		CVSS	Prioritization
Risk	**Low**	**4.8 Medium**	**CAT-1**
Impact	**Low**		
Likelihood	**Unlikely**		
CVSS String	CVSS:3.1/AV:A/AC:H/PR:N/UI:R/S:U/C:H/I:N/A:N		
MITRE ATT&CK	T1600 - Weaken Encryption		
Hosts	10.0.17.10 (22,443/TCP), 10.0.17.11 (22,443/TCP), 10.0.17.12 (22,443/TCP), 10.0.17.13 (22,443/TCP), 10.0.17.14 (22/TCP), 10.0.17.15 (22/TCP), 10.0.17.16 (22/TCP), 10.0.17.87 (22/TCP)		
History	2.0 - Vulnerability found		

Impact
Attacks can abuse insecure ciphers to gather credentials and other sensitive data from otherwise secure network traffic. These can be used to perform account take-overs, escalate privileges, or access new systems.

Details
Old versions of TLS (TLSv1.0 and TLSv1.1) are end-of-life and considered insecure. In addition, some supported ciphers are also no longer considered secure. These ciphers include: • ECDH_SHA2_NISTP • TLS_RSA_WITH_AES_<128/256>_CBC_SHA • TLS_RSA_WITH_CAMELLIA_<128/256>_CBC_SHA • TLS_ECDH_RSA_WITH_AES_<128/256>_CBC_SHA Services that support insecure ciphers may be susceptible to "downgrade attacks." In this kind of attack, a malicious actor sabotages a client's connection to use a weak encryption algorithm. This makes it easier for the malicious actor to break the encryption and sniff the connection. In addition, any RSA-based algorithms may be susceptible to ROBOT attacks.

Replication

1. Web servers were analyzed with SSlyze

```
#  sslyze <host_ip:host_port>
```

```
* TLS 1.0 Cipher Suites:
    Attempted to connect using 80 cipher suites.

    The server accepted the following 6 cipher suites:
       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA                  256
       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA                  128
       TLS_RSA_WITH_AES_256_CBC_SHA                       256
       TLS_RSA_WITH_AES_128_CBC_SHA                       128
       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA                 256       ECDH: prime256v1 (256 bits)
       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA                 128       ECDH: prime256v1 (256 bits)
```

Sslyze Output

2. SSH servers were analyzed with ssh-audit

```
# ssh-audit <host_ip>
```

```
# algorithm recommendations (for OpenSSH 8.2)
(rec) -ecdh-sha2-nistp256                    -- kex algorithm to remove
(rec) -ecdh-sha2-nistp384                    -- kex algorithm to remove
(rec) -ecdh-sha2-nistp521                    -- kex algorithm to remove
(rec) -ecdsa-sha2-nistp256                   -- key algorithm to remove
(rec) -ssh-rsa                               -- key algorithm to remove
(rec) -hmac-sha1                             -- mac algorithm to remove
(rec) -hmac-sha1-etm@openssh.com             -- mac algorithm to remove
(rec) -hmac-sha2-256                         -- mac algorithm to remove
(rec) -hmac-sha2-512                         -- mac algorithm to remove
(rec) -umac-128@openssh.com                  -- mac algorithm to remove
(rec) -umac-64-etm@openssh.com               -- mac algorithm to remove
(rec) -umac-64@openssh.com                   -- mac algorithm to remove
```

Insecure Ciphers Support By The OpenSSH Server

Mitigation

1. Configure the web servers to no longer supportTLSv1.0/1.1. Analyze the supported ciphers for both web servers and OpenSSH servers and disable any insecure ciphers.

References

1. https://www.linuxjournal.com/content/cipher-security-how-harden-tls-and-ssh

8.3.6. Unauthenticated Account Information Disclosed

Unauthenticated Account Information Disclosure		CVSS	Prioritization
Risk	**Medium**	**4.3 Medium**	**CAT-2**
Impact	**Medium**		
Likelihood	**Unlikely**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N		
MITRE ATT&CK	T1078 - Valid Accounts		
Hosts	10.0.17.11 (443/tcp)		
History	2.0 - Vulnerability found		

Impact
This vulnerability could allow an adversary on the internal network to gain insight into which user IDs have the most money and which IDs have admin privileges. This could quickly give an attacker knowledge of which accounts to go after.

Details
[redacted] discovered a rewards API endpoint, and with the aid of the accompanying swagger docs, we discovered that any unauthenticated user could request the account information for all associated users. In our attempt at retrieving the information, there appeared to only be a single account registered with the application. While this did not expose sensitive information such as usernames and passwords, other metadata, including account types, balances, and ID numbers, were visible.

Replication
1. Browse to https://10.0.17.11/accounts/

Observe All Stored Account Information

Mitigation
1. Secure the endpoint by requiring authentication.

References
1. https://portswigger.net/web-security/information-disclosure

8.3.7. Lack of Two Factor Authentication

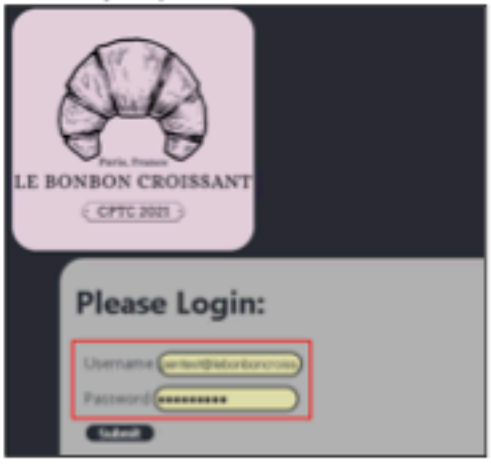
Lack of Two Factor Authentication		CVSS	Prioritization
Risk	**Medium**	**4.3 Medium**	**CAT-2**
Impact	**Medium**		
Likelihood	**Unlikely**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N		
MITRE ATT&CK	T1110 - Brute Force		
Hosts	10.0.17.12 (443/tcp)		
History	2.0 - Vulnerability found		

Impact
Without 2-factor-authentication, you rely on a single point of failure, whose security strength is dictated by the end-user. Traditionally users gravitate towards a single simple password used across multiple accounts. Having one employee's account compromised could be enough to compromise an entire organization.

Details
2-factor authentication (2FA) is an extra security measure taken to further identify and eventually authenticate someone. A password is traditionally seen as something you know which can be easily defeated with password reuse or guessing. 2FA introduces a second factor: either something you have (credit card or smartphone) or something you are (fingerprint, iris, or voice scans).

Replication

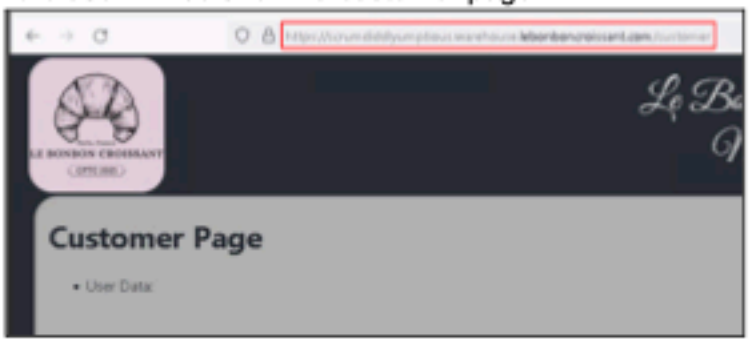
1. Browse to ***https://scrumdiddlyumptious.warehouse.lebonboncroissant.com/customer***



Login For The Application

2. Enter your credentials into the login prompt.

3. Once logged in the user will be shown a customer page.



Successfully Authenticated

Mitigation

1. Implement 2FA on every possible authentication endpoint.
2. Avoid SMS-based 2FA, since it is prone to SIM-swapping attacks.
3. Choose from one of the more secure 2FA protocols below:
    - TOTP
    - Push-based (Duo)
    - FIDO
    - FIDO2 WebAuthn authenticators

References
1. https://cheatsheetseries.owasp.org/cheatsheets/Multifactor_Authentication_Cheat_Sheet.html

8.3.8. Weak Password Policy

Weak Password Policy		CVSS	Prioritization
Risk	**High**	**4.3 Medium**	**CAT-2**
Impact	**High**		
Likelihood	**Unlikely**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:N/I:L/A:N		
MITRE ATT&CK	T1187 - Forced Authentication		
Hosts	10.0.17.12 (443/tcp)		
History	2.0 - Vulnerability found		

Impact
Weak password requirements increase the risk of an attacker gaining unauthorized access to otherwise restricted services and information. Weak passwords on admin accounts increase the risk of an attacker making malicious modifications to web application functionality.

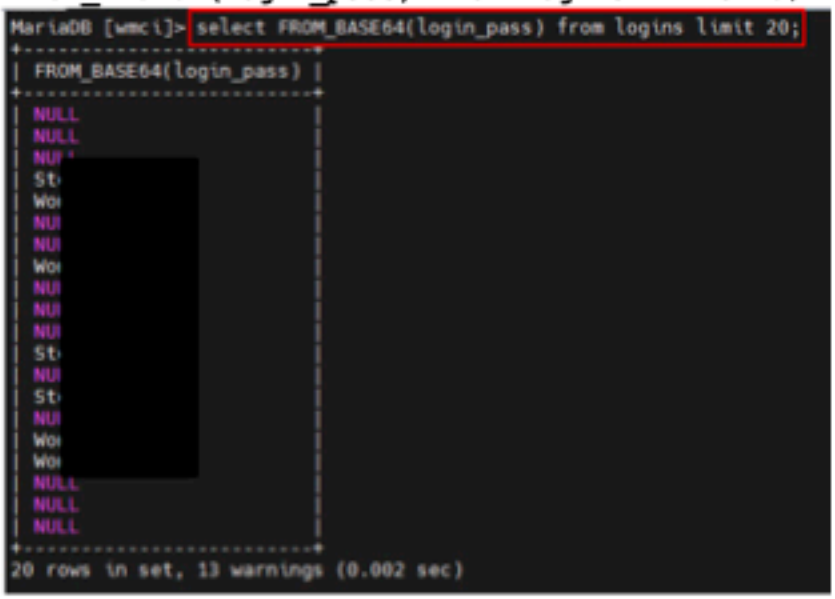
Details
Passwords are considered weak if they are common or easily guessable. Weak passwords were discovered within the environment. [redacted]s initially found these passwords after connecting to the unauthenticated MySQL server. Some of the passwords discovered were only nine characters in length, consisting of lowercase letters. This goes against best practice and can open LBC customers to credential stuffing attacks or password reuse.

Replication

1. Connect to the mysql database as described in *8.1.1 Unauthenticated MySQL DB*.

2. Select the *wnci* database
   ```
   # Use wmci;
   ```

3. Decode the base64 encoded passwords back into plain text
   ```
   # Select FROM_BASE64(login_pass) from logins limit 20;
   ```



Decoded Passwords

Mitigation

1. Create a strong password policy that ensures passwords/pass-phrases with higher entropy greater than 15 characters.
2. Ensure these password requirements are enforced equally throughout the corporate environment.
3. To prevent password cracking and drastically lower the chances of password guessing consider implementing, on sign up, a client-side API call to Have-I-Been-Pwned, which will cross-reference 613 million real-world passwords previously exposed in data breaches. This is a safe and secure way of checking if a password has been exposed due to the use of a privacy-enhancing technology called k-Anonymity. This API is open-source and free for anyone to use.

References

1. https://cwe.mitre.org/data/definitions/521.html

8.3.9. Unrestricted Cross-Origin Resource Sharing (CORS)

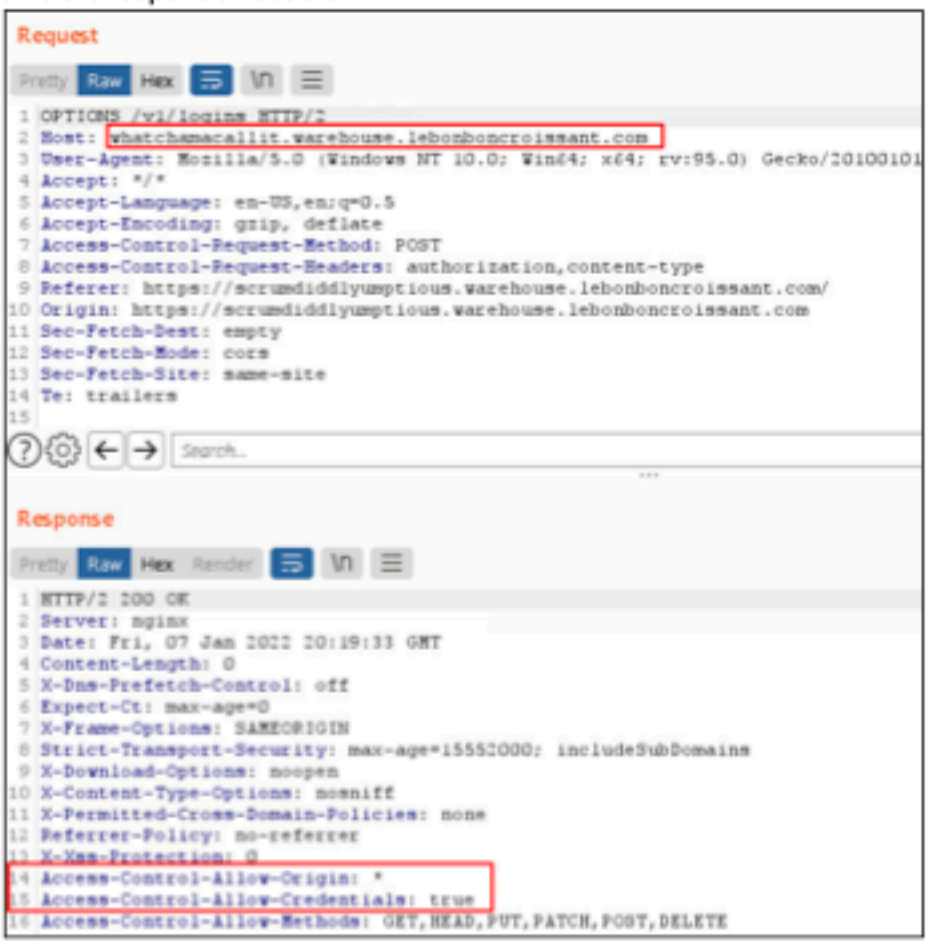
Unrestricted Cross-Origin Resource Sharing (CORS)		CVSS	Prioritization
Risk	**Medium**	**4.3 Medium**	**CAT-2**
Impact	**Medium**		
Likelihood	**Unlikely**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N		
MITRE ATT&CK	T1203 - Exploitation for Client Execution		
Hosts	10.0.17.13 (443/tcp)		
History	2.0 - Vulnerability found		

Impact
This misconfiguration allows any javascript on any website on the internet to issue requests and receive responses from ***whatchamacallit.warehouse.lebonboncroissant.com***.

Details
The same-origin policy (SOP) allows people to safely conduct business on the internet, by default only allowing access to resources within the same "origin." Two URLs are said to have the same origin if the host, protocol, and port match. In many cases, SOP is too restrictive and needs to be bypassed for certain application functionality. Cross-Origin Resource Sharing (CORS) was introduced to allow controlled violations of the SOP. This applications CORS policy was set using the "Access-Control-Allow-Origin: *" header to allow all domains to request resources within the site. Additionally, the header "Access-Control-Allow-Credentials: true" was returned to permit the browser to send credentials along with the cross-origin request to the server. However, according to the specification, the browser is explicitly prohibited from sending credentials when a wildcard accepts any origin. This precaution is done to prevent developers from unknowingly opening up their servers to attack.

Replication

1. Generate a HTTP request to ***whatchamacallit.warehouse.lebonboncroissant.com*** and observe the response headers.



Overly Permissive CORS Policy Returned

Mitigation

1. Validate the requesting origin header against a whitelist of verified domains which are allowed to be communicating with the webserver.

References

1. https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS
2. https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/11-Client-side_Testing/07-Testing_Cross_Origin_Resource_Sharing

8.4. Low

8.4.1 Missing Security Headers

Missing Security Headers		CVSS	Prioritization
Risk	**Low**	**3.7 Low**	**CAT-1**
Impact	**Low**		
Likelihood	**Low**		
CVSS String	CVSS:3.1/AV:A/AC:H/PR:N/UI:R/S:U/C:L/I:L/A:N		
MITRE ATT&CK	T1204 - User Execution		
Hosts	10.0.17.10 (443/TCP) 10.0.17.11 (443/TCP) 10.0.17.12 (443/TCP) 10.0.17.13 (443/TCP) 10.0.17.87 (443/TCP)		
History	1.0 - Vulnerability found 2.0 - Some hosts remediated, but other still vulnerable		

Impact
Lack of HTTP security headers would allow any cross-site scripting vulnerabilities to execute in client browsers. An attacker could use this as part of a phishing campaign to perform account take-overs and/or leak sensitive information.

Details
HTTP security headers, such as Content-Security-Policy and X-XSS-Protection, protect clients against cross-site scripting (XSS) attacks. [redacted] identified several web servers which do not respond with HTTP security headers. Some servers did return an X-XSS-Protection header, but with a value of "0" (disabled). Not including these headers does not make a website vulnerable to XSS, but including them provides the client with more protection if an XSS vulnerability can be exploited.

Replication
1. Visit the hosts in a web browser. 2. View the website response headers by inspecting the page and viewing the "Network" tab.

Security Headers Missing From Server Response

Mitigation
1. Configure the affected web servers to include HTTP security headers. On Nginx systems this can be done using the "add_header" directive, or the "ngx_security_headers" module.

References
1. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers#security

8.4.2. SSH Misconfiguration

SSH Misconfiguration		CVSS	Prioritization
Risk	**Medium**	**3.1 Low**	**CAT-1**
Impact	**High**		
Likelihood	**Rare**		
CVSS String	CVSS:3.1/AV:A/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N		
MITRE ATT&CK	T1021- Remote Services		
Host	10.0.17.14 (22/TCP)		
History	2.0 - Vulnerability Found		

Impact
Allowing root login and password authentication within an ssh configuration exposes the host to potential password brute force attacks or an adversary reusing credentials to pivot with escalated privileges throughout the network.

Details
Root login is enabled on the affected host. This is typically considered a misconfiguration as it exposes the root account to a potential attack by an attacker on the network. In addition, the root account supports password authentication. This exposes the root account to various password-guessing attacks.

Replication
1. First TEAM-7 followed the replication instructions from *8.2.1. Unauthenticated PostgreSQL Server* to gain a shell on the 10.0.17.14 host. 2. We proceeded to view the sshd_config file `# cd /etc/ssh && less sshd_config`

```
postgres@charley:/etc/ssh$ less sshd_config
less sshd_config
WARNING: terminal is not fully functional
sshd_config  (press RETURN)
#       $OpenBSD: sshd_config,v 1.103 2018/04/09

# This is the sshd server system-wide configurat
# sshd_config(5) for more information.
```

Opening The SSHd Config

3. Within the configuration file two configurations were discovered which raised concerns.

```
# Authentication:
:


:
#LoginGraceTime 2m
:
PermitRootLogin yes
:
```

PermitRootLogin Enabled

```
:
# To disable tunneled clear text passwo
:
PasswordAuthentication yes
:
#PermitEmptyPasswords no
:
```

Password Authentication Enabled

Mitigation
1. Consider using more secure forms of authentication, such as SSH keys, for remote login. 2. Disable root login and password authentication by toggling the above two configurations in /etc/ssh/sshd_config

References
1. https://portswigger.net/kb/issues/00600100_directory-listing

8.5. Informational

8.5.1. OSINT

OSINT		CVSS	Prioritization
Risk	**N/A**	N/A	CAT-1
Impact	**N/A**		
Likelihood	**N/A**		
CVSS String	N/A		
MITRE ATT&CK	T1593 - Search Open Websites/Domains		
Hosts	n/a		
History	1.0 - Vulnerability found 2.0 - Vulnerability not remediated		

Details
[redacted] discovered publicly available information through OSINT (Open-source intelligence), the links to which have been provided in the below "Replication" section. The artifact #1 is a github repo with the source of the "OpenCart" application. Although critical vulnerabilities were not identified, this indicates that LBC is possibly using this application and it is possible that in the future attackers might analyze the source-code and discover zero-day vulnerabilities. The artifact #2 is a post on the stackoverflow.com website, where an employee while looking for a potential solution to a problem had posted a piece of code which seems to be used on the production servers. The description in the post indicated the presence of potential security related bugs in the code but none of it could be used to compromise LBC systems. [redacted] discovered an LBC employee's email address within the HTML source code of http://lebonboncroissant.com/. This information may be used to reference past data-breaches, potentially discovering reused credentials or spear phishing. LBC Social Media Accounts Discovered: • Twitter https://twitter.com/BonbonCroissant • Linkedin https://www.linkedin.com/company/le-bonbon-croissant/

- Instagram https://www.instagram.com/lebonboncroissant/

Replication
1. https://stackoverflow.com/questions/69502434/swagger-file-security-scheme-defined-but-not-in-use 2. https://github.com/lebonboncroissant/opencart

Mitigation
1. To avoid potential threats [redacted] recommends that the above information be taken down from their respective sources or ensure that the profile is private and not accessible to unauthorized users. 2. [redacted] understands the need for providing a contact email address on the webpage, but we recommend that a generic support email be provided instead of an employee's email address. 3. As a long-term mitigation, it is recommended that LBC implement "Data Loss Prevention" systems and train their employees in best security practices.

8.5.2. Exposed API Documentation

Exposed API Documentation		CVSS	Prioritization
Risk	**N/A**	N/A	CAT-1
Impact	**N/A**		
Likelihood	**N/A**		
CVSS String	N/A		
MITRE ATT&CK	N/A		
Hosts	10.0.17.10 (443/TCP), 10.0.17.11(443/TCP)		
History	1.0 - Vulnerability found 2.0 - Vulnerability not remediated		

Details

[redacted] ran a gobuster scan against the host to discover accessible endpoints. The tester noted the endpoint "docs/" in the scan results, which returned a 200 OK response when browsed to it. API documentation could aid an attacker inside post-compromise to quickly understand the inner workings of the backend.

Replication

1. Run a content discovery scan of the host using gobuster.
   `# gobuster dir -u https://10.0.17.10 -w /usr/share/seclists/Discovery/Web-Content/raft-medium-directories.txt -x txt -k`

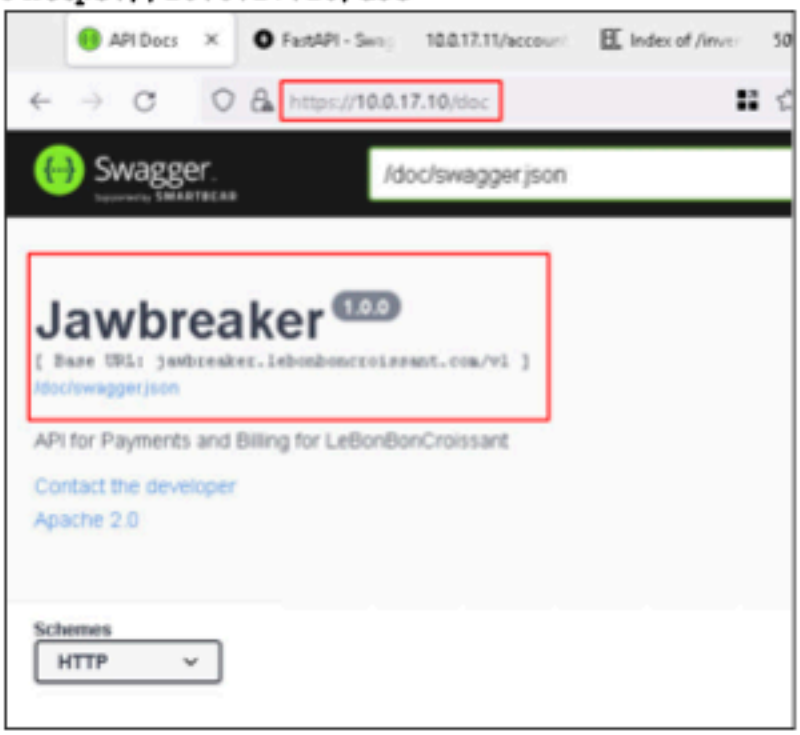
```
┌──(root💀kali06)-[~]
└─# gobuster dir -u https://10.0.17.10 -w /usr/share/seclists/Discovery/Web-Content/raft-medium-directories.txt -x txt -k
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     https://10.0.17.10
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/seclists/Discovery/Web-Content/raft-medium-directories.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Extensions:              txt
[+] Timeout:                 10s
===============================================================
2022/01/07 11:41:48 Starting gobuster in directory enumeration mode
===============================================================
/admin                (Status: 308) [Size: 242] [--> https://app/admin/]
/doc                  (Status: 200) [Size: 1394]
/payment              (Status: 405) [Size: 178]
/status               (Status: 200) [Size: 2]
```

Gobuster Results From Directory Scan

2. Browse to **`https://10.0.17.11/docs#`**

Swagger Docs For Reward API

3. Browse to `https://10.0.17.10/doc`



Jawbreaker Swagger API Docs

Mitigation
1. Restrict IP addresses able to request documentation. 2. Disable

References
n/a

9. Remediations

The following findings are ones which were found during the previous engagement, but have been validated to be remediated during this engagement.

9.1.1. SMB v1 Enabled

SMB v1 Enabled		CVSS	Prioritization
Risk	**Medium**	**3.1 Low**	**CAT-3**
Impact	**Medium**		
Likelihood	**Likely**		
CVSS String	CVSS:3.0/AV:A/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N		
MITRE ATT&CK	T1210 - Exploitation of Remote Services		
Hosts	10.0.7.200, 10.0.17.201 (445/TCP)		
History	1.0 - Vulnerability found 2.0 - Vulnerability remediated & CVSS score adjusted		

Impact
Previous instances have shown ransomwares exploiting SMBv1 to bring the businesses down. If LBC becomes a victim of such a ransomware attack it would not only lead to business loss for LBC but also monetary loss, damage to reputation along with regulatory fines.

Details
During the previous engagement [redacted] found that the above-mentioned hosts were running Server Message Block (SMB) Version-1, which is a very vulnerable version of the service. Exploiting SMB generally requires credentials but SMBv1 can be exploited without any credentials. SMBv1 has several vulnerabilities (MITRE CVE's for SMBv1) and should be avoided if possible. Even if the host is on the internal network, users could be targets of phishing attacks which combined with the vulnerable SMB version, targeted attacks are highly likely to succeed. Malwares such as Petya, NotPetya, WannaCry have exploited this service using MS017_010 vulnerability to disrupt critical services and also their source code is publicly available.

[redacted] decided not to test the EternalBlue exploit for the MS017_010 vulnerability as it might have resulted in downtime. Other methods of assessment did not reveal any sensitive information from this service. Considering its potential to be exploited, this vulnerability's impact on business is High. The likelihood of this being exploited is likely as the attack complexity is low and automated tools and resources are available for the same, which makes the risk as High.

Previous Replication

1. The commands scan the IP address to gather information about the Operating System present on it
2. Run the following nmap commands

```
# nmap -A 10.0.17.200
```

```
Host script results:
|_clock-skew: mean: -1s, deviation: 0s, median: -1s
| smb-security-mode:
|   authentication_level: user
|   challenge_response: supported
```

Nmap Results

References

1. Detect and Disable SMB v1
2. MITRE CVE's for SMB v1
3. Security Update for Microsoft Windows SMB Server

Remediated

[redacted] scanned for the ports associated with the SMB and based on our scans we were not able to interact with the hots.. Hence [redacted] concludes that the finding was remediated.

```
┌──(root㉿kali03)-[~]
└─# nmap -Pn -p 139,445 10.0.17.200
Starting Nmap 7.92 ( https://nmap.org ) at 2022-01-08 09:36 EST
Nmap scan report for ip-10-0-17-200.ec2.internal (10.0.17.200)
Host is up.

PORT    STATE    SERVICE
139/tcp filtered netbios-ssn
445/tcp filtered microsoft-ds

Nmap done: 1 IP address (1 host up) scanned in 3.07 seconds
```

SMB port show up as filtered for 10.0.17.200

9.1.2. Missing SSL/TLS Implementation

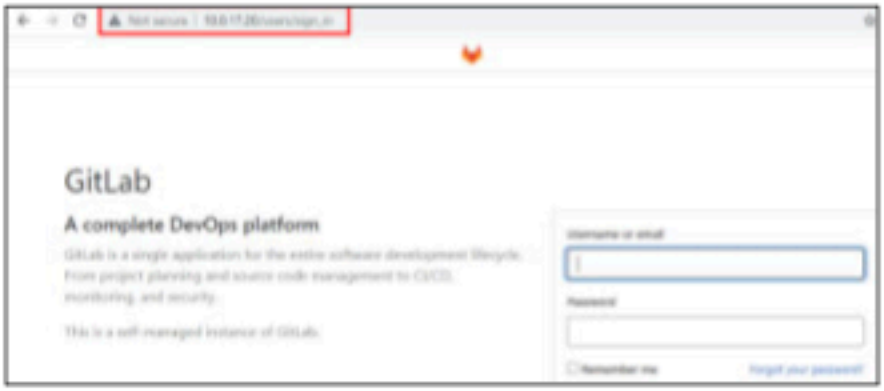
Missing SSL/TLS Implementation		CVSS	Prioritization
Risk	**High**	**8.5 High**	**CAT-3**
Impact	**High**		
Likelihood	**Likely**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:N		
MITRE ATT&CK	T1210 - Exploitation of Remote Services		
Hosts	10.0.17.10 (80/tcp), 10.0.17.11 (80/tcp), 10.0.17.20 (80/tcp), 10.0.17.178 (80/tcp), 10.0.17.50 (9090/tcp)		
History	1.0 - Vulnerability found 2.0 - Vulnerability partially remediated		

Impact
Login credentials and other sensitive information could be obtained by anyone on LBC's network by capturing network traffic. This could lead to compromise of both customer accounts and/or LBC infrastructure causing significant business impact.

Details
During the previous engagement [redacted] found that HTTPS is not configured or forced for any of the applications on the 10.0.17.0/24 subnet. The HTTP protocol sends all data across the network in plain text, meaning that any login credentials or data from the application could be sniffed using a packet analyzer by an attacker. An attacker must be suitably positioned to eavesdrop on the victim's network traffic to exploit this vulnerability. This scenario typically occurs when a client communicates with the server over an insecure connection such as public Wi-Fi or a corporate or home network shared with a compromised computer. HTTPS uses SSL/TLS to encrypt all communications sent to the server, reducing the likelihood of the vulnerabilities mentioned above being exploited. [redacted] classifies this finding as High risk, as lack of in-transit network encryption exposes these applications to traffic sniffing.

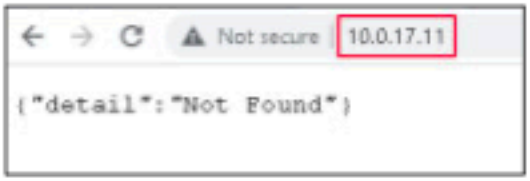
Previous Replication

1. Applications are able to be reached via HTTP

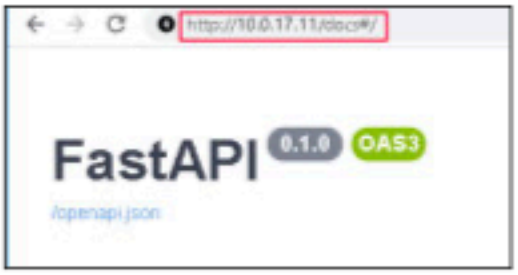


GitLab Homepage

Tomcat Server Running Over HTTP



Linux API

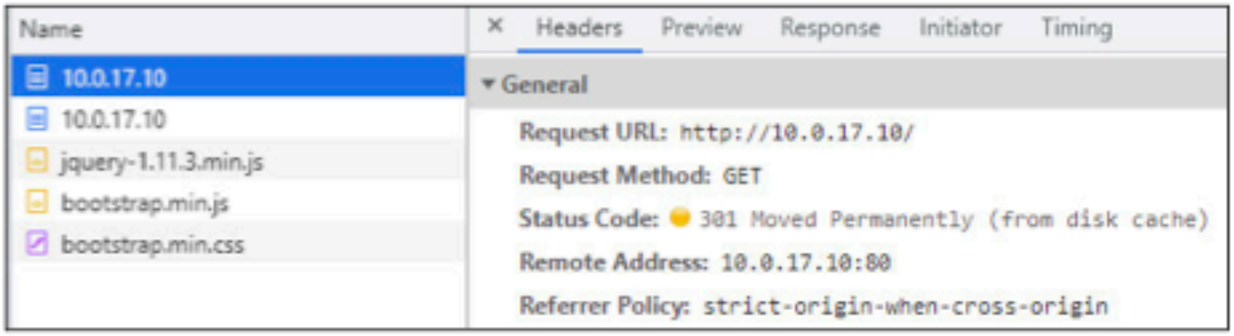


FASTAPI Swagger Docs

Ubuntu Web Server

References
1. Active Directory Certificate Services Overview 2. Configuring HTTPS servers

Remediated
LBC mostly remediated this vulnerability on all hosts except one, 10.0.17.50 (9090/tcp). LBC also forced HTTP requests to redirect to HTTPS which shows an excellent commitment to security.



HTTP requests are redirected to secure HTTPS

9.1.3. Poorly Authenticated VNC

Poorly Authenticated VNC		CVSS	Prioritization
Risk	**Medium**	**4.9** **Medium**	**CAT-2**
Impact	**Medium**		
Likelihood	**Likely**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:L/UI:R/S:U/C:L/I:L/A:L		
MITRE ATT&CK	T1078 - Valid Accounts		
Hosts	10.0.17.200 (5900/tcp)		

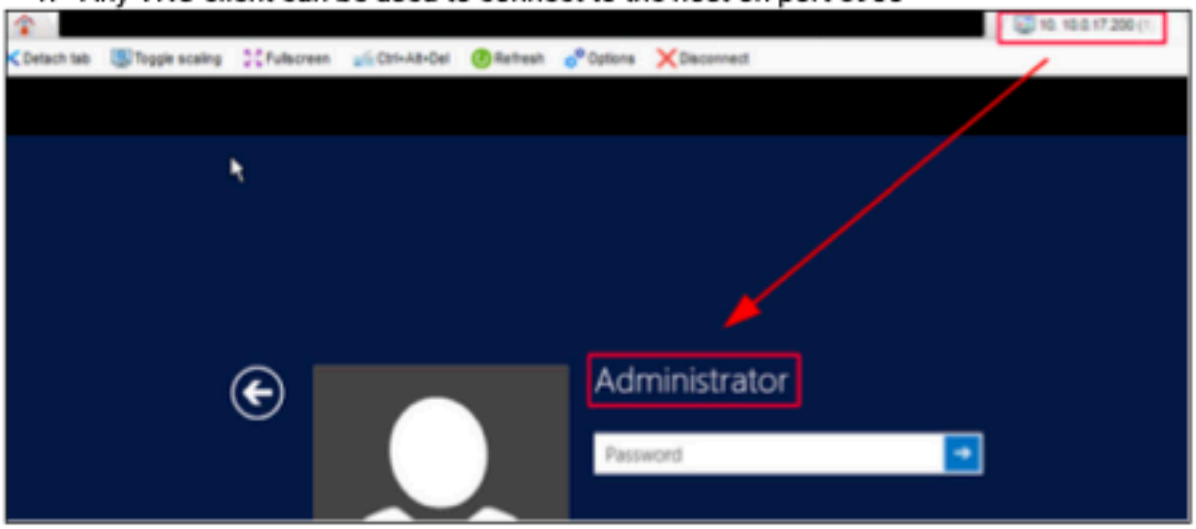
History	1.0 - Vulnerability found 2.0 - Vulnerability remediated

Impact
An attacker could potentially access the underlying host had the previous user left the host unlocked. [redacted] assesses that this poor authentication will likely be exploited given the ease of access to the VNC service. [redacted] calculates the risk of this vulnerability as medium

Details
A Windows host 10.0.17.200 running a ThinVNC service was detected by [redacted] This VNC service accepted any random password as a valid password and gave the underlying Windows host GUI console access. [redacted] was unable to bypass the Windows authentication prompt seen after the initial VNC access

Previous Replication

1. Any VNC client can be used to connect to the host on port 5900



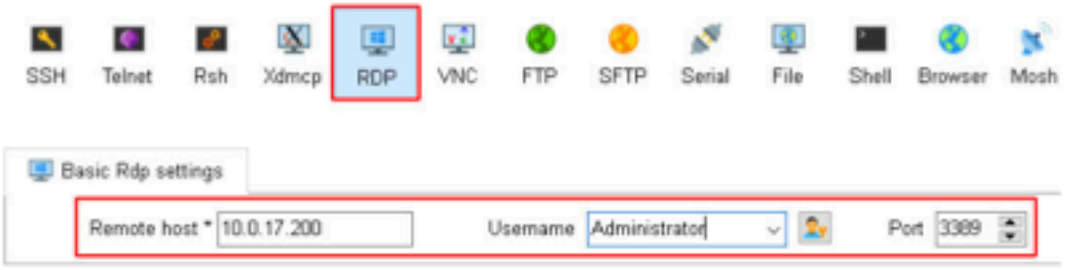
MobaXterm VNC Client Connected to 10.0.17.200 With an Arbitrary Password

References

1. Configuring VNC

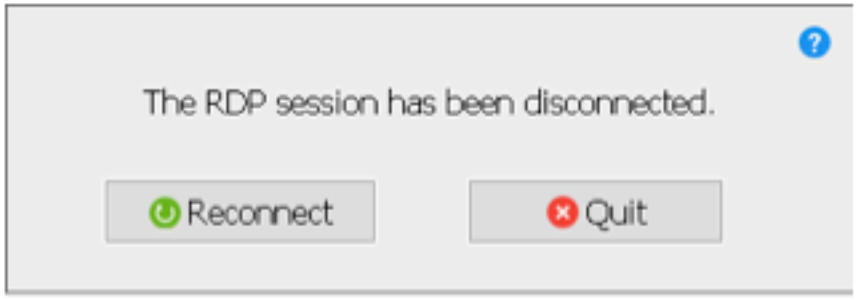
Remediated

1. [redacted] attempted the same steps initially taken to connect to the machine over VNC.



VNC Configuration For 10.0.17.200

2.



VNC Connection Failed

3. [redacted] scanned the host to determine which ports were open, and realized that the host was offline. [redacted] looked but could not find another location where it was moved to on the network.

```
# nmap 10.0.17.200
```

```
┌──(root💀kali06)-[~]
└─# nmap 10.0.17.200
Starting Nmap 7.92 ( https://nmap.org ) at 2022-01-08 09:59 EST
Note: Host seems down. If it is really up, but blocking our ping
Nmap done: 1 IP address (0 hosts up) scanned in 3.07 seconds
```

Nmap Scan Showing 10.0.17.200 Down

9.1.4. Outdated Software and Operating System

Outdated Software and Operating System		CVSS	Prioritization
Risk	**Medium**	**7.5 High**	**CAT-2**
Impact	**Medium**		
Likelihood	**Likely**		
CVSS String	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H		
MITRE ATT&CK	T1592 - Gather Victim Host Information		
Hosts	10.0.17.11(80/tcp), 10.0.17.50 (9090/tcp), 10.0.17.178 (80/tcp), 10.0.17.14		
History	1.0 - Vulnerability found 2.0 - Vulnerability partially remediated		

Details
During the assessment, [redacted] identified legacy Operating Systems and Software Applications. Several new vulnerabilities are discovered daily, and software vendors release new versions of the software. [redacted] identified that two web applications were running outdated software. 10.0.17.178 was running Nginx 1.18.0, which is known to be vulnerable to HTTP request smuggling attacks and other exploits. 10.0.17.50 was running Apache Tomcat version 6.0.53, which the Apache Tomcat Team no longer supports, according to tomcat.apache.org documentation. The host 10.0.17.11 was running FastAPI version 0.1, which is vulnerable to Cross-Site Request Forgery (CSRF) attacks. Since there was no authentication on the API, there are no session cookies that may cause a CSRF vulnerability to be High impact. However, there may be other publicly known vulnerabilities that exist in deprecated software packages. [redacted] also found that the 10.0.17.14 host was running Ubuntu 16.04.7 LTS which is known to be vulnerable to many kernel exploits. [redacted] did not attempt these risky exploits due to the high possibility that the availability of the service would likely be affected by a failed exploit. An adversary is likely to use these risky exploits and disrupt critical services. If exploited, this could cause a denial-of-service, escalate to root privileges or gain remote code execution. HTTP smuggling could also be used to poison caches or bypass security controls. However, since [redacted] found no sensitive data on the 10.0.17.14 host, no high impact via CSRF, and both web applications seemed to be default pages with low business impact, [redacted] ranked this finding as Medium risk.

Previous Replication

1. Outdated Software Discovered

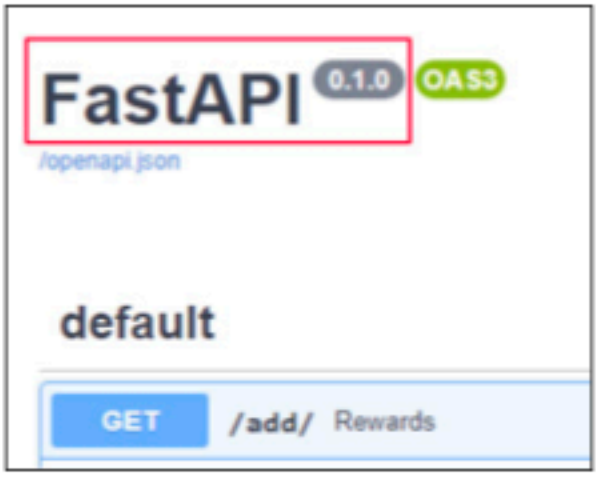
```
Nmap scan report for 10.0.17.178
Host is up (0.00047s latency).

PORT   STATE SERVICE VERSION
80/tcp open  http    nginx 1.18.0 (Ubuntu)
|_http-server-header: nginx/1.18.0 (Ubuntu)
```

Nginx 1.18.0

```
postgres@charley:/var/lib/postgresql/9.5/main$ cat /etc/issue
cat /etc/issue
Ubuntu 16.04.7 LTS \n \l
```

Ubuntu 16.04.7



FastAPI 0.1.0

References

1. NSA Outdated Software and Protocols
2. End of life for Apache Tomcat 6.0.x
3. Nginx Security Vulnerabilities
4. Fastapi Vulnerabilities

Remediated

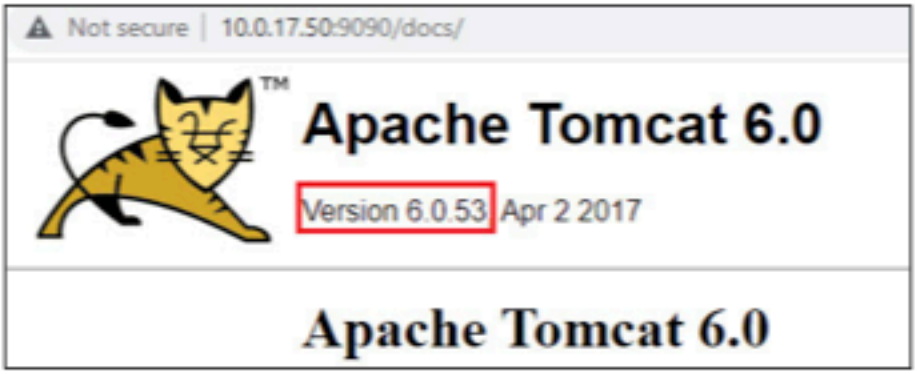
1. [redacted] identified that the host 10.0.17.14 has been updated to the latest operating system as shown in the below image.

```
es@charley:/var/lib/postgresql/12/main$ uname -a
-a
charley.warehouse.lebonboncroissant.com 5.11.0-1023-aws #24~20.04.1-Ubuntu SMP Fri Dec 3 13:46:41 UTC 2021
4 GNU/Linux
```

2. [redacted] identified that the Fast API on host 10.0.17.11 is now secured with an authentication layer.

3. The host 10.0.17.11 and 10.0.17.178 were un-accessible during the test due to which [redacted] could not verify if the vulnerability was fixed or not.

4. [redacted] identified that the vulnerability regarding the Tomcat version on host 10.0.17.50 was fixed or not as shown below.



9.1.5. Unauthenticated Rewards API Access

Unauthenticated Rewards API Access		CVSS	Prioritization
Risk	High	8.1 High	CAT-4
Impact	High		
Likelihood	Likely		
CVSS String	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N		
MITRE ATT&CK	T1078 - Valid Accounts		
Hosts	10.0.17.11 (80/TCP)		
History	1.0 - Vulnerability found 2.0 - Vulnerability remediated		

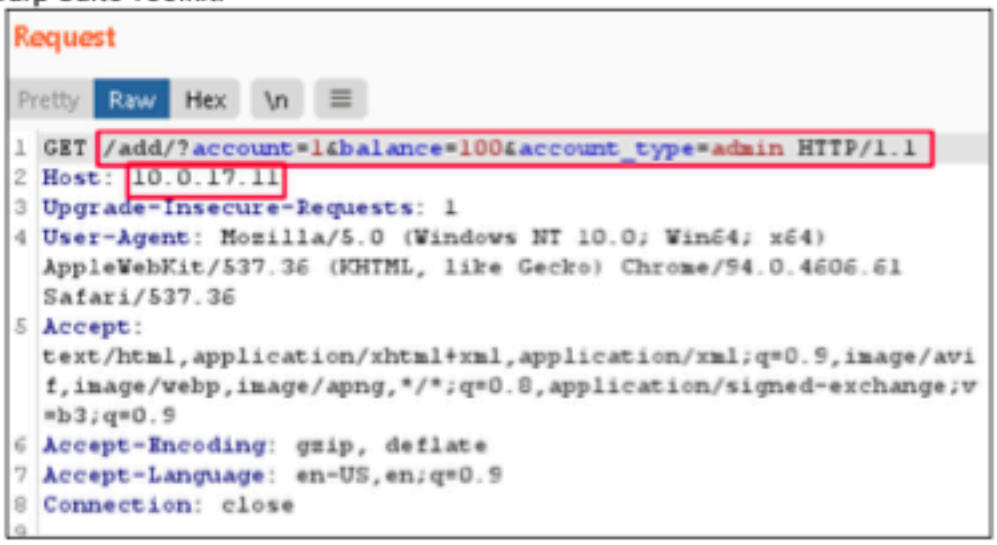
Impact

Any unauthorized user could add reward points to an account, which could cause large financial loss for LBC. Given that the likelihood of this finding being exploited is Likely, [redacted] classifies the risk of the finding to be High.

Details
Using the Swagger documentation previously discovered, [redacted] attempted to add credits to an account using the Rewards API. An API request was made not using any cookies or other session information. The GET request assigned 100 points to account number 1 with the account type "admin." The server successfully processed the request and returned a 200 OK response. This vulnerability allows anyone with access to the API to easily modify the balance amounts of different users since sequential numbers represent accounts.

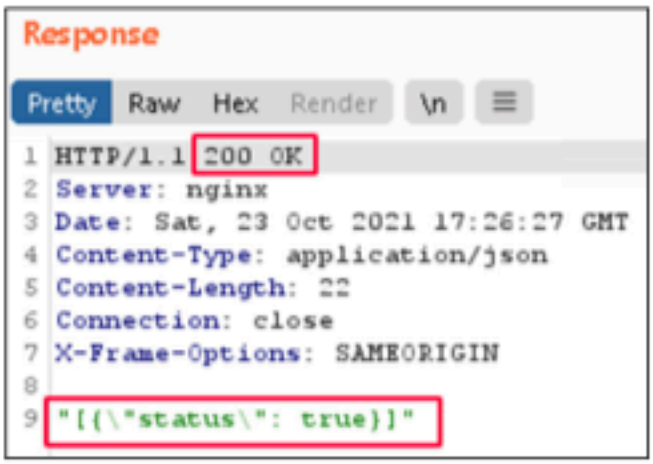
Previous Replication

1. The GET request was modified as shown below using the Repeater extension in the Burp Suite Toolkit.



Request Sent Adding 100 Credits to Account 1

2. The below response was received which confirms the request was successful, according to the Swagger documentation.



200 OK Response From Server

References
OWASP API Security Top 10 Using HTTP Cookies Secure Cookies

Remediated
FINAL-7 re-tested the endpoint to determine if the vulnerability persists, the server responded back with a "Internal Server Error".    *Server Failing To Add A Balance* While we were unable to exploit the vulnerability it seems that the changes performed by LBC to remediate the issue lead to the server crashing. It is best practice to handle all errors gracefully and it is recommended that LBC respond with a 403 Forbidden or a 404 Not Found.

9.1.6. SMB Signing Disabled

SMB Signing Disabled		CVSS	Prioritization
Risk	High	4.3	CAT-3
Impact	High		

Likelihood	Likely	Medium	
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N		
MITRE ATT&CK	T1557 - Adversary-in-the-middle		
Hosts	10.0.17.200-201(445/tcp)		
History	1.0 - Vulnerability found 2.0 - Vulnerability remediated		

Impact
An attacker could deploy a relay to capture valid credentials which the attacker can then send to the server as their own in an attempt to gain unauthorized access to critical resources with the goal of Data-Exfiltration, Denial-of-Service, Pivoting to internal networks, etc. which will have a direct impact on the LBC's business.

Details
During the previous assessment [redacted] found in the nmap scan that a guest account on the machine had smb1 message-signing disabled. Message-signing allows the SMB communications to be digitally "signed" at the "packet-level". The mechanism allows the recipient to verify the authenticity of the source. Although it is a default setting it is still dangerous since the attacker can set up a relay between the server and the client. This man-in-the-middle attack can help attackers gain the accounts present on the machine and their respective NTLM hash. The attacker can then crack offline to gather passwords or use these hashes to gain access into other machines. The risk of this vulnerability is high. Since carrying out the attack does not require much sophistication or extensive knowledge, the likelihood is very likely, which makes the overall impact high.

Previous Replication
1. The commands scan the IP address to gather information about the Operating System present 2. Run the following nmap commands `# nmap -A 10.0.17.200`

```
Host script results:
|_clock-skew: mean: -1s, deviation: 0s, median: -1s
| smb-security-mode:
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
| smb2-security-mode:
|   2.02:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2021-10-23T19:14:21
|_  start_date: 2021-10-23T15:05:45
```

Nmap SMB Service Enumeration

References
Message Signing Block

Remediated
[redacted] scanned for the ports associated with the SMB service and they came up "filtered" which means that they were firewalled off. Hence [redacted] concluded that the finding was remediated.

```
┌──(root㉿kali03)-[~]
└─# nmap -Pn -p 139,445 10.0.17.201
Starting Nmap 7.92 ( https://nmap.org ) at 2022-01-08 09:35 EST
Nmap scan report for ip-10-0-17-201.ec2.internal (10.0.17.201)
Host is up.

PORT    STATE    SERVICE
139/tcp filtered netbios-ssn
445/tcp filtered microsoft-ds

Nmap done: 1 IP address (1 host up) scanned in 3.07 seconds
```

SMB port show up as filtered for 10.0.17.201

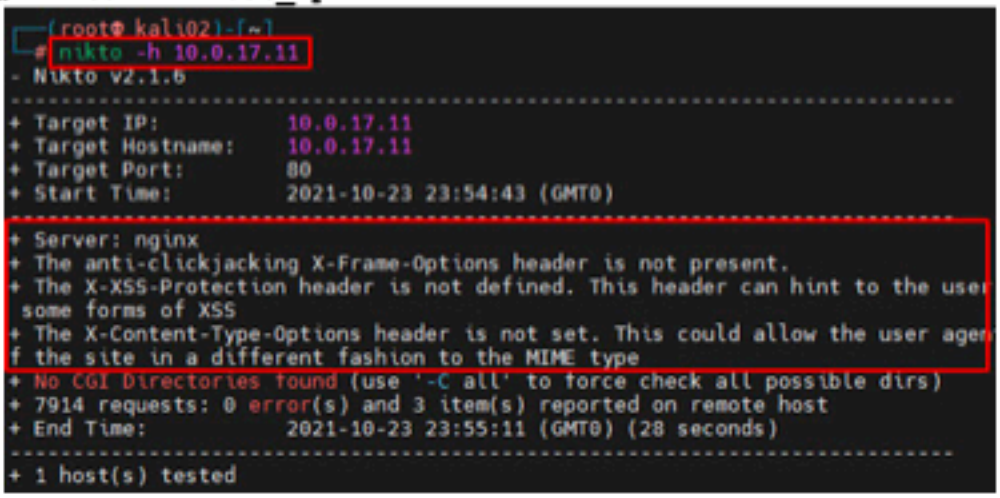
9.1.7. Missing HTTP Security Headers

Missing HTTP Security Headers	CVSS	Prioritization

Risk	**Low**	**3.7** **Low**	**CAT-1**
Impact	**Low**		
Likelihood	**Likely**		
CVSS String	CVSS:3.1/AV:A/AC:H/PR:N/UI:R/S:U/C:L/I:L/A:N		
MITRE ATT&CK	T1189 - Drive-by Compromise		
Hosts	10.0.17.10-11 (80/tcp), 10.0.17.178 (80/tcp), 10.0.17.50 (9090/tcp)		
History	1.0 - Vulnerability Found 2.0 - Vulnerability Partially Remediated		

Impact
An attacker could potentially have access to similar devices. The attacker can connect to LBC's ICS's systems and read data from the coils potentially replicating the device state. This would further allow them to research and craft complex attacks. Also, these devices are often legacy systems, an attacker could cause a denial-of-service using simple tools such as ping. Considering the above scenarios the business impact to LBC due to this vulnerability is considered as Critical.

Details
During the assessment [redacted] found HTTP Security Headers anti-clickjacking X-Frame-Options Header and X-XSS Protection Header missing on the several web applications hosted on 10.0.17.0/24 subnet. HTTP Security Headers are used by the servers to inform the clients machine of the security settings for the communication. Presence of these security headers make the web servers and web applications more resilient to attacks such as clickjacking and cross-site scripting to name a few. As the attack just requires running a simple nikto tool against the port that hosts the web-server the likelihood becomes very likely but since the attacker does not gain much critical information the impact on business is low which makes the overall risk also low.

Previous Replication
1. In the terminal type the command **# nikto -h <host_ip>**  *Figure : Nikto scan showing missing security headers*

Mitigation
[redacted] recommends adding a few security headers which protects against common web vulnerabilities like Cross Site Scripting (XSS) and other code injection attacks. 1. Add content security policy header `'Content-Security-Policy: script-src 'self' https://www.google-analytics.com'` 2. X-Frame-Options header a. In Apache `'header always set X-Frame-Options "SAMEORIGIN;" '` b. In Nginx `'add_header X-Frame-Options "SAMEORIGIN" always;'` 3. X-XSS Protection Header a. In Apache `'header always set X-XSS-Protection "1; mode=block"'`. b. In Nginx `'add_header X-XSS-Protection "1; mode=block" always;'` 4. X-Content-Type-Options a. In Apache `'header always set X-Content-Type-Options "nosniff"'` b. In Nginx `'add_header X-Content-Type-Options "nosniff" always"'` 5. Feature-Policy a. In Apache `'header always set Feature-Policy "autoplay 'none'; camera 'none'"'` b. In Nginx `'add_header Feature-Policy "autoplay 'none'; camera 'none'" always'`

References
1. Hardening HTTP Security Headers

Remediation
1. [redacted] identified that LBC had implemented security headers as mentioned in the previous engagement to protect form web-based attacks 2. [redacted] also identified that the NginX version was being transmitted as part of the Headers as shown in the below image. Although this does not directly lead to a vulnerability, an attacker could combine this information along with others to craft payloads specific to LBC's which might lead to a more severe vulnerability causing potential business impact. Hence this vulnerability is considered as partially remediated.

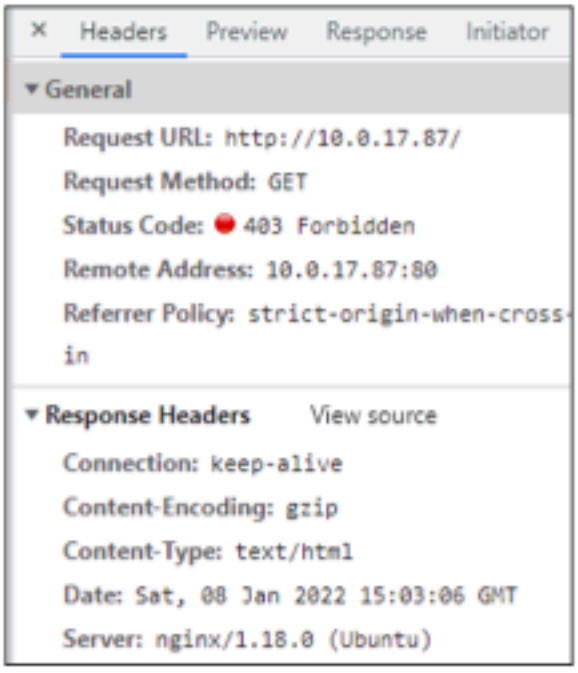


Figure : Version revealed in response headers

10. Appendix A - Tools

Tool Name	Purpose	Description
Nmap + Link	Enumeration	Port scanning, host enumeration
PyModbus	Enumeration	Interact with PLC systems
EasyModbusTCP	Enumeration	Extract data from PLC coils
Gobuster	Enumeration	Web application directory brute-forcing
Metasploit	Exploit, Post-Exploit	General penetration testing framework
Linuxprivchecker	Post-Exploit	Local privilege escalation tool for Linux
Linux-exploit-suggester	Post-Exploit	Local privilege escalation tool for Linux
nmaptocsv	Administrative	Converts Nmap output to csv for excel importing

11. Appendix B - Assessment Artifacts

System Artifacts

These artifacts consist of scripts / reverse shells that were deployed for the penetration testing engagement. They were all removed before the engagement

Artifact Name	Host	Description
LinPEAS.sh	10.0.17.14	Post-Exploit enumeration
cptc_tester.txt	10.0.17.14	Temporary file to analyze command output
payload.py	10.0.17.14	Privilege Escalation Script

Postgres Database Artifacts

These artifacts consist of tables that were created in the database systems for testing purposes.

Database System	Database	Table
PostgreSQL	postgres	pentlab
PostgreSQL	postgres	pentlab1
PostgreSQL	postgres	Demo