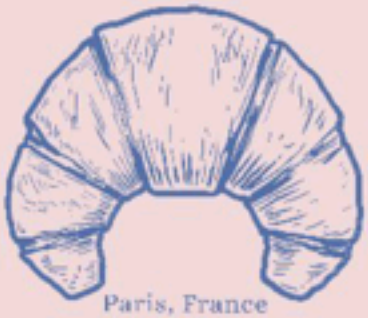

LE BONBON CROISSANT

CPTC 2021

Penetration Test Report

Conducted by:

[redacted]

January 9, 2022

Contents

1 Report Overview

1.1 Executive Summary

[redacted] was contacted by Le Bonbon Croissant (LBC) for a second penetration test in order to identify security issues within their infrastructure. This report was written initially on January 7th and submitted on January 9th at 1:00AM. This penetration test is in the interest of LBC, as part of a restrained scope second penetration test and risk assessment. The Report Overview section contains an outlined summary of [redacted]s findings, including recommendations for improving LBC's security, mitigating potential business risk, and reducing attack surface. The Technical Findings section expands upon the report overview by including each discovered vulnerability's evaluated risk, exploitation details, and recommended remediation steps.

Based upon the results of the assessment, LBC is at risk to be fined by payment providers due to severe PCI DSS violations. These fines could range from $5,000 to $100,000 per month depending on factors such as size of business [1]. Based on these issues, we suggest spending resources to become complaint. Details of all PCI DSS violations can be found in Section 4.5.5.

Similarly to the previous assessment conducted against LBC, [redacted] was able to gain full access to a SCADA system using the same default credentials as the previous engagement. This device is extremely important as it is critical to industrial systems which operate the storage, delivery, and packaging warehouse facility. Similarly, the industrial control device is not isolated in any way, could be manipulated by any user on the network. It is important to take this with extreme caution, as malicious tampering with these devices could result in loss of life. This would result in unwanted attention, could harm the companies reputation, and could cost LBCvast sums of money from both lawsuits and long term loss of business.

During this engagement, a total of **11** vulnerabilities were found in LBC's network. In terms of severity, **3** vulnerabilities are critical, **3** vulnerabilities are high, **1** vulnerability is moderate, and **4** vulnerabilities posed low risk. Many of the vulnerabilities in the environment are due to improper authentication, default credentials, or not applying principles of least privileges. More information about these vulnerabilities can be found in Section 4.

1.2 Engagement Overview

[redacted] conducted a follow up penetration test starting on Janu ry 7th, 2022 based upon the Request for Proposal (RFP) document obtained by [redacted]ocus was placed on the following goals during the engagement:

- Assessing internally developed and customized software packages.
- Assessment of Industrial controls within a storage, delivery, and packing warehouse facility.
- Assessment of APIs related to payment, transaction, billing, and inventory processing systems.
- Discovering vulnerabilities and complications which could impact the confidentiality, integrity, and availability (CIA) of LBC's information systems.
- Assisting LBC in improving their security posture.
- Evaluate LBC's security posture against the Payment Card Industry Digital Security Standard (PCI-DSS).

1.3 Scope of Engagement

The full scope of this penetration test was limited to the following CIDR range. At the request of LBC , the first day of the engagement (Jan 7 2022) excluded hosts 10.0.17.50 and 10.0.17.51 in order to ensure availability of critical infrastructure. These hosts were included on the second day (Jan 8th starting at 9:20AM) to ensure proper testing. Care was taken by [redacted] to ensure penetration test activity did not reduce the availability of industrial systems.

- 10.0.17.0/24

The penetration test was conducted with extreme care to ensure actions were contained within the defined scope. Additionally, because the engagement was within a production environment, the team ensured that no services were disrupted. [redacted] did not exfiltrate, modify, or delete any data not included in this report.

[redacted] is available upon request to improve the security, protect the employees, and customers of LBC. This includes verifying and validating implemented mitigation techniques as well as deploying security strategies to ensure LBC has several layers of defense. The team is happy to continue a partnership with LBC and excited to work along side them in securing their operations.

2 Observations

This section serves as a high level overview of the security posture of LBC. A detailed list of all discovered vulnerabilities can be found in Section 4. It is important to note that this list is by no means exhaustive and that there are most likely vulnerabilities that [redacted] did not find.

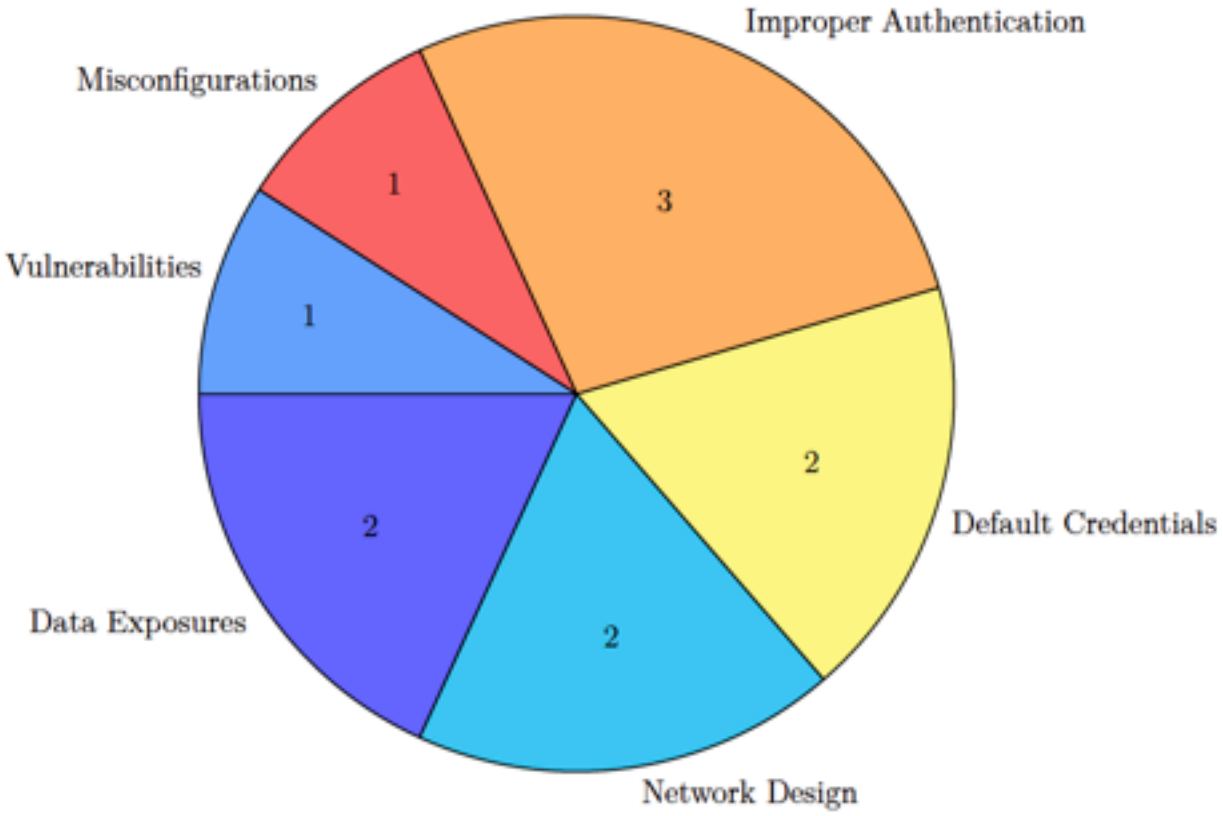

Figure 1: Summary of Issues within the Network

Default Credentials & Lack of authentication

The most immediate observation about LBC's security posture is that default, null, and passwordless authentication was discovered on multiple systems. [redacted] was able to gain access into a SCADA system using the same credentials (which were default) as the last engagement. The system in question is critical to the storage, delivery, and packaging processes within the warehouse facility. Moreover, several databases contained this same issue and were found to not be requiring password authentication. It is important to remember these credentials are for critical services and PII. These weaknesses can have an enormous impact on LBC's ability to operate if discovered by threat actors. These vulnerabilities can be remediated with low cost and have an outsized impact on the security of LBC. More details on mitigation for vulnerabilities such as these can be found in each vulnerability's remediation suggestions in Section 4.

Improper/Lack of Encryption

Another problem observed by [redacted] was that payment information was stored in cleartext, violating PCI DSS. Using proper encryption to secure cardholder data can have an immense impact on maintaining PCI DSS compliance. Additionally, passwords stored in one database were encoded in base64, which is insecure and not an industry standard. Passwords are generally hashed using algorithms such as SHA2 which are theoretically irreversible. Improper encryption of LBC's marketplace passwords resulted in their retrieval.

Passwords

[redacted] also noticed that their was not a password policy on the LBC's marketplace. This indicates that users can use weak passwords creating a risk of potential accounts that can be stolen and potentially credit card information stolen. Additionally, because of access to a database and improper use of encryption the passwords were able to be seen in cleartext. Many account credentials contained weak passwords and more information can be found in Section 4.5.2. Accounts were found to be using the same password and had contained information related to LBC.

Software Configurations

[redacted] also noticed software misconfigurations apart from default credentials. For example, default installations of many services were found to be unmodified or containing unnecessary components. Additionally, one web server was found that was nonessential to the operations of LBC. Services like these could either be removed or isolated onto different machines to help mitigate the number of potential attack vectors. Enabling encryption could also help prevent potential attacks on the network as it would be harder to discover endpoints such as the SCADA web application. However, compared to the last engagement, encryption was implemented on the majority of hosts, which is a major improvement.

Network Design & Firewalls

Based upon our assessment, the network environment does not abide by best practices. Communications between ICS devices are not restricted to specific hosts and [redacted] was able to connect to this device. Additionally, a music player server was found to be running on the same network segment as critical infrastructure, which, can pose an unnecessary risk. If the media server were compromised, it could result in easier compromise of the industrial controls critical to the function of LBC. [redacted] recommends that LBC implement a Demilitarized Zone (DMZ) to allow for isolation and improve the security of operations. Many of the attacks on LBC's systems could be prevented by enabling firewalls as many machines did not have one enabled.

2.1 Summary of Recommendations

The following is an overview of recommendations which should be implemented:

- Resolve any PCI DSS compliance violations in Section 2.3.1 and ensure that LBC is currently meeting all documentation requirements set by PCI DSS.
- Implement both ingress & egress filtering to reduce attack surface on hosts.
- Ensure all hosts use least privilege principals to reduce attack surface.
- Ensure proper encryption is used for confidential data (e.g. passwords and card holder data).
- Implementation of a strong password policy (see Section 4.5.3 for more information).
- Implement Multi-Factor authentication to provide defense in depth in addition to passwords.
- Uses centralized logging to be able to respond to potential incidents faster.
- Ensure null or password-less authentication is not allowed.
- Ensure only necessary services are running within The Paris Warehouse subnet.

2.2 Positive Security Measures

As the engagement progressed, [redacted] was impeded by the security safeguards LBC had in place. A number of basic security best practices were observed that limited [redacted]s ability to move through the network. Some instances of aforementioned security practices implemented by LBC include:

- The usage of TLS on websites to protect information.
- The usage of Cross-Origin Resource Sharing (CORS) headers prevented specific attacks.
- The marketplace & music player required authentication.
- Some APIs required authentication in order to query sensitive information (although not all).

These controls should be continuously monitored and regulated to maintain the company's security posture.

2.3 Compliance

2.3.1 PCI DSS Violations

The following table details violations of the PCI Data Security Standard discovered by [redacted] over the scope of this engagement.

Table 1: PCI DSS Compliance Violations.

Regulation	Reason	Reference
PCI DSS 1.1.4	A firewall is not implemented at every internet connection.	Section 4.5.4
PCI DSS 1.2	The PostgreSQL server that stores cardholder data does not have a firewall restricting connections from untrusted networks.	Section 4.1.3
PCI DSS 1.3	The PostgreSQL server is not segmented through the use of a DMZ.	Section 4.1.3
PCI DSS 2.1	Default accounts were not removed from all systems on the network.	Sections 4.1.3 & 4.1.1
PCI DSS 2.2	No evidence found to indicate that LBC has configuration standards.	Section 4.5.5
PCI DSS 2.2.1	The server "charley" had more than one primary function implemented: MariaDB and PostgreSQL.	Sections 4.1.3 & 4.2.2
PCI DSS 2.2.2	No evidence of documentation for enabled insecure services, daemons, or protocols.	N/A
PCI DSS 2.2.4	There were insecure security parameters present on system configurations. No found evidence of documented common system security parameters and no evidence of system configuration standards.	Sections 4.1.3 & 4.1.1
PCI DSS 2.4	An inventory of what was considered to be in scope for PCI DSS was not provided to penetration testers nor found during the engagement.	Section 4.5.5
PCI DSS 2.5	No evidence of security policies and operational procedures was found during the engagement	Section 4.5.5

Table 2: PCI DSS Compliance Violations Continued.

Regulation	Reason	Reference
PCI DSS 3.2.2	The card verification codes for credit cards were stored after authorization in a PostgreSQL database.	4.1.3
PCI DSS 3.5	No encryption is used to secure stored cardholder data.	Section 4.1.3
PCI DSS 3.6	No evidence of a key-management process nor documentation of such was found. Additionally, no cyrptographic keys were used for any encryption of cardholder data.	Section 4.5.5
PCI DSS 3.7	No evidence of security policies and operational procedures for protecting stored cardholder data was found.	Section 4.5.5
PCI DSS 5	No evidence of protection against malware or anti-virus was found to be present on the network.	N/A
PCI DSS 6.2	No evidence of policies or procedures for regular application of security patches was found.	Section 4.5.5
PCI DSS 6.3	LBC's staff informed [redacted] that the web based store found on the "scrumdiddlyumptious" host was both in early development and in production use, thus not following industry standards and best practices for software development.	N/A
PCI DSS 6.4.1	Similar to 6.3, the development and production environments for the web store found on the "scrumdiddlyumptious" host were not separate.	N/A
PCI DSS 6.4.2	The client informed [redacted] that the Oompas and Loompas at LBC do not always stay consistent within their separation of development and production environment duties.	N/A
PCI DSS 6.4.5	No evidence was found that showed documentation of change-control procedures for implementing security patches and software modifications.	Section 4.5.5
PCI DSS 6.5	No evidence was found that showed documentation of software-development processes or procedures which address coding vulnerabilities.	Section 4.5.5

Table 3: PCI DSS Compliance Violations Continued.

Regulation	Reason	Reference
PCI DSS 6.7	No evidence was found that showed documentation of security policies and procedures for development and maintenance of systems and applications	Section 4.5.5
PCI DSS 7	Access to cardholder data was not restricted by business need to know as there was not a password required to access the "billing" database on PostgreSQL, thus anyone could access cardholder data.	Section 4.1.3
PCI DSS 8.1	LBC does not currently satisfy any of the sub-requirements of PCI DSS 8.1. Notably, 8.1.2 because the database servers both have password-less authentication.	Sections 4.1.3 & 4.2.2
PCI DSS 8.2.1	Strong cryptography is not utilized in password storage in the "wmci" database on the "charley" host, as the passwords are only encoded with base64.	Section 4.5.2
PCI DSS 8.2.3	No password policies were found to be implemented within the environment that require seven character passwords that consist of both numeric and alphabetic values.	Section 4.5.3
PCI DSS 8.2.4	No password policies were found to be implemented within the environment that require password resets after 90 days or less.	Section 4.5.3
PCI DSS 8.2.5	No password policies were found to be implemented within the environment that require password changes to be different from the four most recent previous passwords.	Section 4.5.3
PCI DSS 8.2.6	No password policies were found to be implemented within the environment that outline the procedure for passwords on new generated users.	Section 4.5.3
PCI DSS 8.3	No evidence of multi-factor authentication was found anywhere on the network.	N/A
PCI DSS 8.4	No policies or procedures were found during the engagement that indicated guidance for strong authentication credentials.	Section 4.5.3
PCI DSS 8.5	Generic and default accounts were found to be present on the network, including the PostgreSQL server that stores cardholder data.	Sections 4.1.3 & 4.1.1

Table 4: PCI DSS Compliance Violations Continued.

Regulation	**Reason**	**Reference**
PCI DSS 8.7	Access to the PostgreSQL database was not restricted by any means.	Section 4.1.3
PCI DSS 8.8	No policies or procedures for identification and authentication were found to be documented.	Section 4.5.5
PCI DSS 10	No evidence of tracking or monitoring of access to cardholder data was found in the environment.	N/A
PCI DSS 11.3.3*	In order to maintain compliance following this penetration test, all exploitable vulnerabilities detailed in this report must be resolved. It should be noted that since [redacted]s last engagement with LBC, some previously reported vulnerabilities have still remained unmodified.	Section 4
PCI DSS 11.4	No evidence of intrusion-detection nor intrusion-prevention systems were found anywhere on the network.	Section 4.5.4
PCI DSS 11.5	No evidence of change-detection mechanisms were found anywhere on the network	N/A
PCI DSS 11.6	No evidence of policy and procedure documentation for security monitoring and testing was found.	Section 4.5.5

* This compliance violation is only applicable if the exploitable vulnerabilities detailed herein are not resolved.

3 Testing Methodology

3.1 Penetration Testing Execution Standard

Throughout the engagement [redacted] references the Penetration Testing Execution Standard (PTES) when conducting security assessments [2].

Figure 2: PTES Methodology

3.2 MITRE ATT&CK Framework

MITRE ATT&CK is a knowledge base of Tactics, Techniques, and Procedures (TTPs) based upon real-world observations from security professionals. ATT&CK is a curated knowledge base for cyber adversary behavior, reflecting the attack lifecycle and platforms known to target. [redacted] uses ATT&CK to aide in understanding TTPs that can be used to conduct an attack against LBC that could be conduct by real world adversaries [3].

3.3 OWASP Top 10

Referenced in this report is the Open Web Application Security Project (OWASP) Top 10 when applications are found within the applicable scope [4]. OWASP Top 10 focuses vulnerabilities focus on common vulnerabilities that pose security risks to web applications:

Table 5: OWASP Top 10

1. Broken Access Controls	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

3.4 PCI DSS Auditing

One of the requests of LBC was for experience in PCI DSS from the RFP. Throughout the engagement, [redacted] audited PCI DSS compliance in accordance with the proper Self-Assessment Questionnaire (SAQ). By doing this, [redacted] can ensure which PCI DSS security requirements are met in accordance with the proper SAQ for the type(s) of transactions being performed by LBC. Each PCI DSS compliance failure can be found in Section 2.3.1.

3.5 NIST SP 800-53

NIST SP 800-53 is the National Institute of Standards and Technology Special Publication 800-53, Security and Privacy Controls for Federal Information Systems and Organization. NIST 800-53 is a security compliance standard that offers guidance for how organizations should select then maintain security and privacy controls for information systems. NIST 800-53 is mandatory for all federal agencies however, its guidelines can be adopted by any organization operating information systems with sensitive or regulated data. This standard provides a catalog of privacy and security controls for protecting against various threats.

Table 6 provides security and privacy control methodology which are organized into 20 families. These control families are referenced throughout the document and are used to constitute common terminology. Additionally, referenced in NIST 800-53 is control families enhancements to help provide guidance to aide in securing LBC's information systems [5].

Table 6: NIST 800-53 Security and Privacy Control Families for Compliance.

ID	Family	ID	Family
AC	Access Control	**PE**	Physical and Environmental Protection
AT	Awareness and Training	**PL**	Planning
AU	Audit and Accountability	**PM**	Program Management
CA	Assessment, Authorization, Monitoring	**PS**	Personnel Security
CM	Configuration Management	**PT**	PII Processing and Transparency
CP	Contingency Planning	**RA**	Risk Assessment
IA	Identification and Authentication	**SA**	System & Services Acquisition
IR	Incident Response	**SC**	System & Communications Protection
MA	Maintenance	**SI**	System & Information Integrity
MP	Media Protection	**SR**	Supply Chain Risk Management

4 Technical Findings

This table shows the total number of vulnerabilities found during the penetration test engagement. The vulnerabilities are categorized based on the risk level. The risk levels were calculated using the Common Vulnerability Scoring System (CVSS) [6].

Risk Level and Total Number of Discovered Vulnerabilities

Severity	Low (0.1-3.9)	Moderate (4.0-6.9)	High (7.0-8.9)	Critical (9.0-10.0)
Vulnerability Count	4	1	3	3

The following table breaks down the discovered vulnerabilities by overall risk score, impact, and exploitability. The scores were calculated using NIST's CVSS v3.1 calculator [7].

Summary of Vulnerabilities by Base Score

Risk Summary	Overall Risk Score	Impact	Exploitability
Default SCADA System Credentials	10	10	10
SCADA Controller Remote Code Execution	9	10	8
Lack of PostgreSQL Authentication	9	8	10
Lack of PLC Authentication	8	10	6
Lack of MariaDB Authentication	7.5	5	10
PostgreSQL Remote Code Execution	7	8	6
Payment Transaction Enumeration	4.5	4	5
Music Player Daemon (MPD) Directory Traversal	2	2	2
ScadaBR Reflected XSS (Username)	1.25	1	1.5
ScadaBR Reflected XSS (URL)	1	1	1
Memcache Null Authentication	0.75	0.5	1

4.1 Critical Risk

4.1.1 ScadaBR Default Credentials

Threat Level: Critical (10)

Description:

An application running the SCADA system known as "ScadaBR" was found to be running on a Tomcat web server. The system was using default credentials that gave full access to the administrative interface of the web application to [redacted]

Potential Business Impact:

Any user on LBC's network could authenticate to the SCADA HMI and view the layout of LBC's manufacturing systems. A threat actor could make modifications to operations or potentially stop all activity. This would include changing belt speeds, turning off devices, and more. If an attacker stopped these devices, it would result in a halt of LBC's operations. Additionally, improper modification of PLC devices could result in destruction or potential loss of life. Therefore, the significance of the ScadaBR controller using default credentials cannot taken lightly.

Affected Host:

Crunch (10.0.17.50)

Exploitation Details:

By browsing to `http://10.0.17.50:9090/ScadaBR` a user could enter the username: "admin" and password: "admin" to successfully authenticate to the ScadaBR system.

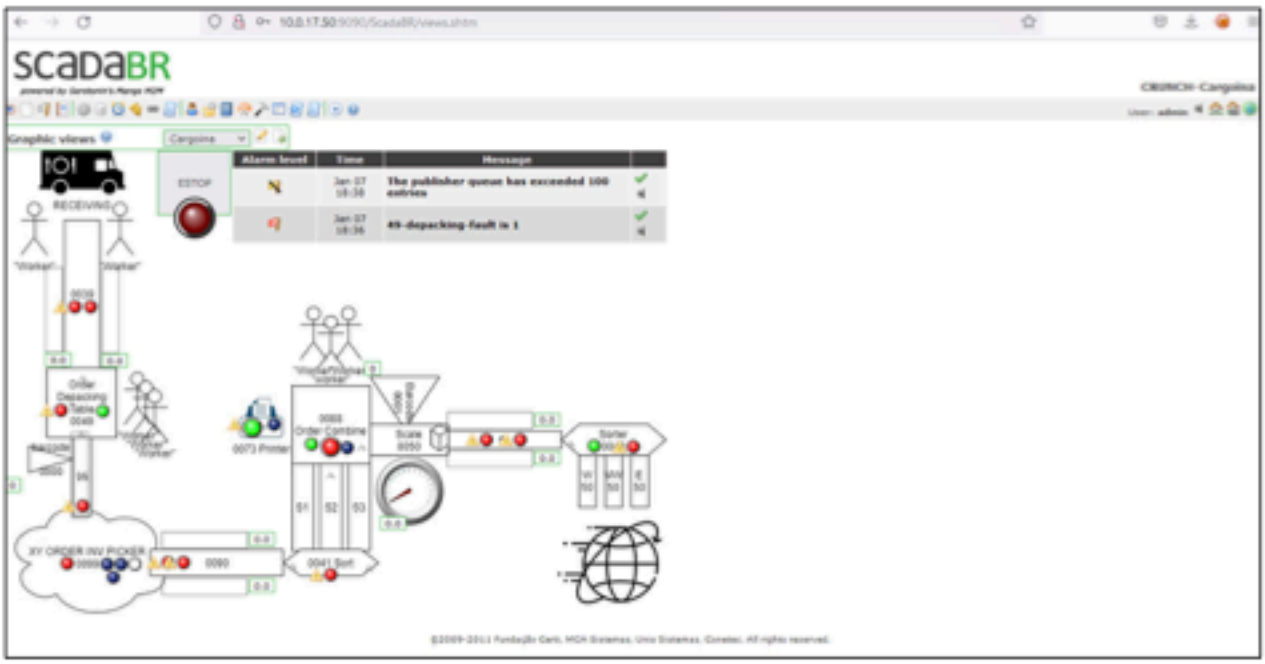

Figure 3: Authenticated as Admin on ScadaBR.

Recommended Remediation:

Change default credentials to use a strong password. An example of a strong password policy can be found in the NIST password recommendations. A suggestion for a sufficient password policy is the following:

- Do not uses passwords related to LBC in ANY manner.
- Have at least one uppercase character.
- Have at least one lowercase character.
- Have at least one symbol character.
- Have at least one numeric character.
- Passwords should be at least 12 characters long.

If requested, [redacted] can aide in providing more guidance for a strong password policy.

Additionally, ICS systems should be isolated from the corporate network with a DMZ and proper network segmentation should be implemented. Firewall rules should be created using utilities such as iptables to mimize potential risk related to the SCADA system.

References:

`https://auth0.com/blog/dont-pass-on-the-new-nist-password-guidelines/`

`https://www.scadabr.com.br/`

4.1.2 ScadaBR Remote Code Execution

Threat Level: Critical (9)

Description:

An authenticated remote code execution vulnerability present in ScadaBR lead to root-level access on the SCADA controller.

Potential Business Impact:

With valid credentials to the ScadaBR web interface, an individual could execute arbitrary code and have unrestricted access to the underlying system. The protocol used by the SCADA controller to communicate with the PLC was present on the system in the form of php scripts. An adversary with access to this information could directly manipulate values on the PLC without needing to go through the SCADA controller interface.

Affected Hosts:

Crunch (10.0.17.50)

Crunchserial (10.0.17.51).

Exploitation Details:

SCADABR allows an authenticated user to upload an arbitrary file as a "Background image" with the *view_edit.shtml* page, and subsequently execute said file by visiting */uploads/<filename>*. Due to the underlying Tomcat web server's ability to execute Java, a JSP file containing a malicious payload could be utilized to gain an interactive shell.

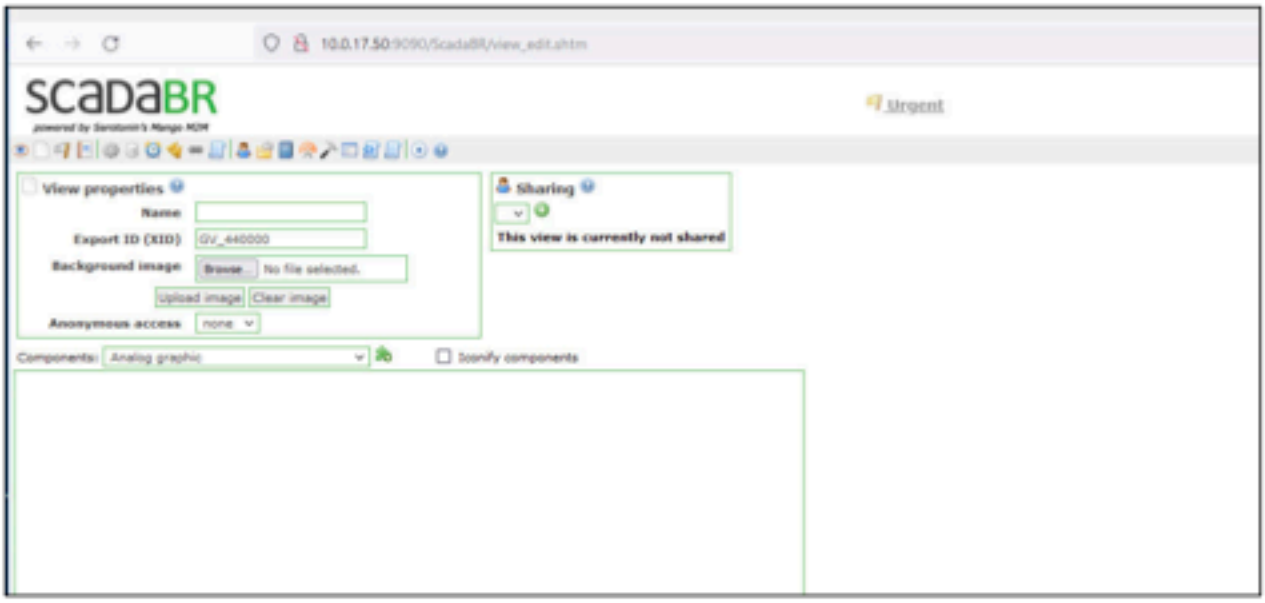

Figure 4: SCADA Arbitrary File Upload

Figure 4 shows the web page allowing the user to upload an arbitrary file, and Figure 5 shows

the web shell payload used by [redacted] to achieve code execution. After triggering a reverse shell, [redacted] had interactive shell access as user *root*, as shown in Figure 6.

Figure 5: SCADA Arbitrary Code Execution

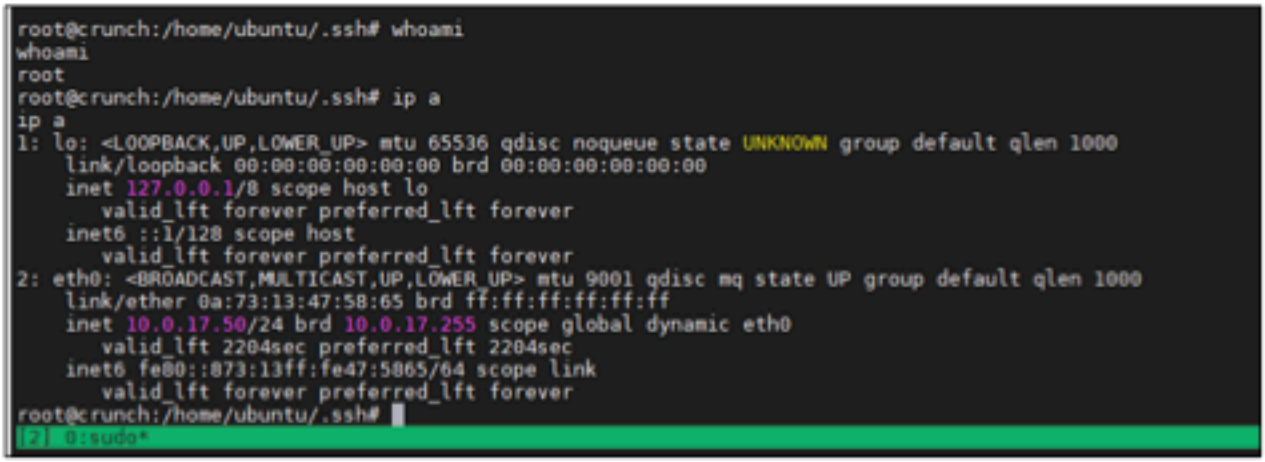

Figure 6: SCADA Root Shell

Recommended Remediation:

If possible patch ScadaBR to a version not vulnerable to this RCE. Additionally, ensure that access to the ScadaBR web application is secure. Ensure proper logging of users and modifications of the PLC. Moreover, by implement progress ingress and egress filtering can make it potentially harder for an attacker to gain a proper connection from the RCE. **References**:

`https://github.com/h3v0x/CVE-2021-26828_ScadaBR_RCE`

4.1.3 Lack Of PostgreSQL Authentication

Threat Level: Critical (9)

Description:

The host Charley on the network did not require password authentication for the postgres user in PostgreSQL. As a result, attackers can access all databases on charley and enumerate data found. The postgres user has full control over the database within the host.

```
┌──(root㉿kali01)-[/tmp]
└─# 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.

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 7: User postgres does not require a password to authenticate.

Potential Business Impact:

The data stored within this database contained unencrypted database information which is a direction violation of PCI DSS more information information can be found in Section 2.3.1. Failures of PCI DSS can result in fines and other punishments. Each security incidents and breaches can result of a $500,000 fine [8]. Figure 8 shows that credit card information was stored encrypted.

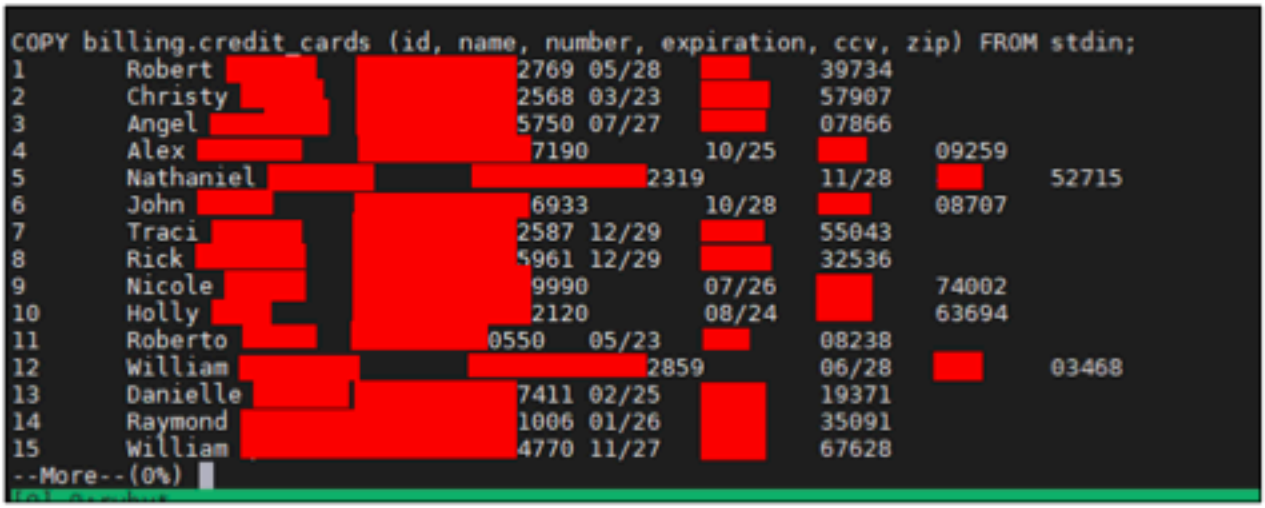

Figure 8: PostgreSQL Billing Table stored in the clear.

Affected Host:

Eggdicator (10.0.17.10)

Scrumdiddlyumptious (10.0.17.12)

Charley (10.0.17.14)

Exploitation Details:

A user who can connect to 10.0.17.14 can connect to the postgresql server by running the following command:

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

Recommended Remediation:

Harden the PostgreSQL server to require password authentication. Additionally, having firewall access controls to restrict what respective IP addresses can access the database would provide an additional layer of security.

```
ALTER USER postgres PASSWORD 'B3tt3rP@ssw0rd';
```

Additionally, the PostgreSQL instance could be further hardened by making rules in the *pg_hba.conf* file to only allow for authentication from certain hosts. More information about this configuration file can be found in the references for this section.

References:

`https://www.postgresql.org/docs/13/auth-password.html`

`https://www.postgresql.org/docs/9.2/auth-pg-hba-conf.html`

4.2 High Risk

4.2.1 Lack of PLC Authentication

Threat Level: High (8)

Description:

No access controls were in place to prevent unauthorized manipulation of the programmable logic controller (PLC) running on the warehouse network. As a result, anyone would be able to read or write to stored values which represent configurations for industrial machines in LBC warehouses.

Potential Business Impact:

The ramifications of this issue are twofold. A malicious actor could set operational parameters of industrial equipment to unsafe levels, potentially damaging machinery or leading to loss of life. They could also set these values such that the machinery remains idle or inoperable, denying service to the factory and leading to lost revenue.

Affected Hosts:

Crunchserial (10.0.17.51)

Exploitation Details:

The protocol for communicating with the PLC was determined by examining PHP source code and Nginx request logs found on *crunch*. Reading values from the PLC is accomplished by sending a command of the form "G<key1>,<key2>" over a TCP socket on port 2001. Similarly, writing values can be done by sending "S<key1>,<key2>,<value>". [redacted] used *netcat* to send TCP traffic for testing these commands.

```
┌──(root㉿kali02)-[~/scans]
└─# echo "G0039,0092" | nc 10.0.17.51 2001
0

┌──(root㉿kali02)-[~/scans]
└─# echo "S0039,0092,1" | nc 10.0.17.51 2001
OK

┌──(root㉿kali02)-[~/scans]
└─# echo "G0039,0092" | nc 10.0.17.51 2001
1

┌──(root㉿kali02)-[~/scans]
└─# echo "S0039,0092,0" | nc 10.0.17.51 2001
OK

┌──(root㉿kali02)-[~/scans]
└─# echo "G0039,0092" | nc 10.0.17.51 2001
0
```

Figure 9: PLC Value Manipulation

Figure 10 shows some of keys used to access specific values on the PLC, along with what devices they refer to.

```
//
////////////////////// SORTERS ////////////////////////////
"0041" => [
    "0032" => 0, /// 41-sort2order-sensepkgin
    "0099" => 0, /// 41-sort2order-outputln
    "0021" => 0, /// 41-sort2order-resetln
    "0055" => 0, /// 41-sort2order-fault
    "0043" => 0, /// 41-sort2order-estop
    "0058" => 0, /// 41-sort2order-sensepkgout
],
"0049" => [
    "0032" => 0, /// 49-depacking-sensepkgin
    "0099" => 0, /// 49-depacking-outputln
    "0021" => 0, /// 49-depacking-resetln
    "0055" => 0, /// 49-depacking-fault
    "0043" => 0, /// 49-depacking-estop
    "0058" => 0, /// 49-depacking-sensepkgout
],
"0088" => [
    "0032" => 0, /// 88-ordercomb-sensepkgin
    "0099" => 0, /// 88-ordercomb-outputln
    "0021" => 0, /// 88-ordercomb-resetln
    "0055" => 0, /// 88-ordercomb-fault
    "0043" => 0, /// 0039
    "0058" => 0, /// 88-ordercomb-sensepkgout
],
```

Figure 10: PLC Key Examples

Recommended Remediation:

Ideally, an authentication method should be implemented on the PLC to prevent unauthorized access to the system. If this is not possible due to limitations of the device, the PLC should be placed on a segmented network behind a firewall such that it can only be reached by the ScadaBR host (Cruch) and the attached industrial machinery.

4.2.2 Lack of MariaDB Authentication

Threat Level: High (7.5)

Description:

Unauthenticated access to a MySQL database permits access/modification to sensitive datasets, including the following:

- Customer Accounts and passwords (Base64 encoded).
- Customer PII - includes phone numbers, address, and payments (including amounts).
- Invoices and payments.
- Creation of administrator accounts for LBC's croissant marketplace.
- Insertion, deletion, and modification of all data within the database.

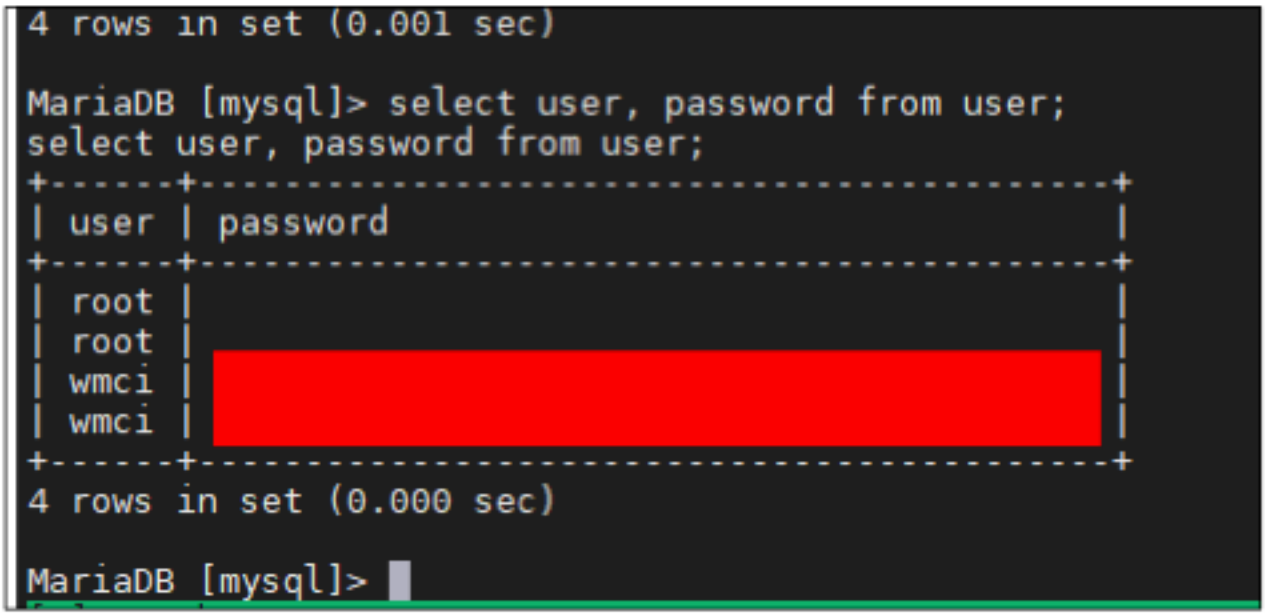

Figure 11: Passwordless Root MariaDB Access

Potential Business Impact:

This host (Charley) can severely impact the Confidentiality, Integrity, and Availability (CIA) of transactions within the warehouse management systems. Improper encoding schemes result in an environment in which all of the LBC store website (Scrumdiddlyumptious) users' passwords can be decoded from base64.

All accounts can also be modified to granted administrator level access on the LBC store. As data is parsed through the root user, transactions, account details, items (for sale), and other information which is integral to LBC's ability to sell its products online may be subject to unauthorized modification. This data is not directly accessed by the web store host (Scrumdiddlyumptious), but rather through an API endpoint host (Whatchamacallit).

Affected Hosts:

Charley (10.0.17.14)

Scrumdiddlyumptious (10.0.17.12)

Whatchamacallit (10.0.17.13)

Exploitation Details:

[redacted] used the MySQL command line application to query the unauthenticated database as the root user. All customer information could be obtained without modifying the database schema or contents. This user also had write permissions, meaning a malicious actor could arbitrarily modify database contents.

Figure 12 shows the connection from the PostgreSQL server and initial shell commands.

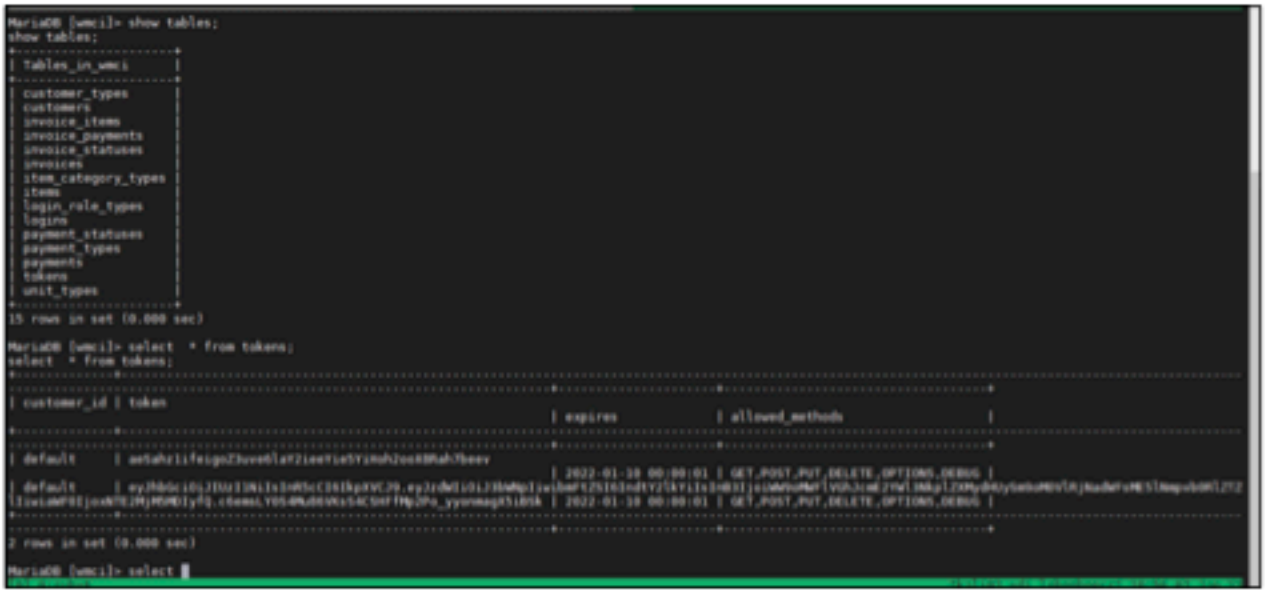

Figure 12: MariaDB tokens Table Dump

Recommended Remediation:

A minimal number of hosts should be able to interact on the network with the MySQL database. By limiting network access to only those required hosts, potential attacks against the MySQL service are minimized. The following iptables commands can be used to enable access controls and limit which hosts can reach MySQL:

```
iptables -A INPUT -p tcp --dport 3306 -s <IP> -j ACCEPT
iptables -A INPUT -p tcp --dport 8000 -j DROP
```

If a firewall is not an option due to constraints within the environment (we recommend using a firewall), another option is to use ACLs within MySQL. This can be done by altering users to only be accessible via certain IPs or domains. This can be done using the following commands:

```
CREATE USER 'user'@'localhost' IDENTIFIED BY 'password';
CREATE USER 'user'@'10.0.17.13' IDENTIFIED BY 'password';
```

References:

https://dev.mysql.com/doc/mysql-security-excerpt/8.0/en/general-security-issues.html

https://dev.mysql.com/doc/mysql-security-excerpt/8.0/en/access-control.html

https://www.digitalocean.com/community/tutorials/iptables-essentials-common-firewall-rules-and-commands

4.2.3 PostgreSQL 9.5.25 Remote Code Execution

Threat Level: High (7)

Description:

Postgres users with elevated permissions can exploit a vulnerability in PostgreSQL 9.5.25 which allows interactive shell access as user *postgres* on the underlying system.

Potential Business Impact:

A threat actor exploiting this vulnerability would have access to the underlying Ubuntu server running the PostgreSQL database. They may be able to alter the PostgreSQL server configuration and could deny service to the database.

Affected Hosts:

Charley (10.0.17.14)

Exploitation Details:

Users with "pg_read_server_files' can run commands in instances where they have permissions with the "pg_execute_server_program'. Within the host, the postgresql user had no password and had proper permissions to utilize these two functionalities. Therefore, [redacted] used the *postgres_copy_from_program_cmd_exec* metasploit module to exploit this vulnerability. The module executed a reverse shell which connected to [redacted]s host, allowing for an interactive shell as *PostgreSQL*. Figure 13 shows the connection from the PostgreSQL server and initial shell commands.

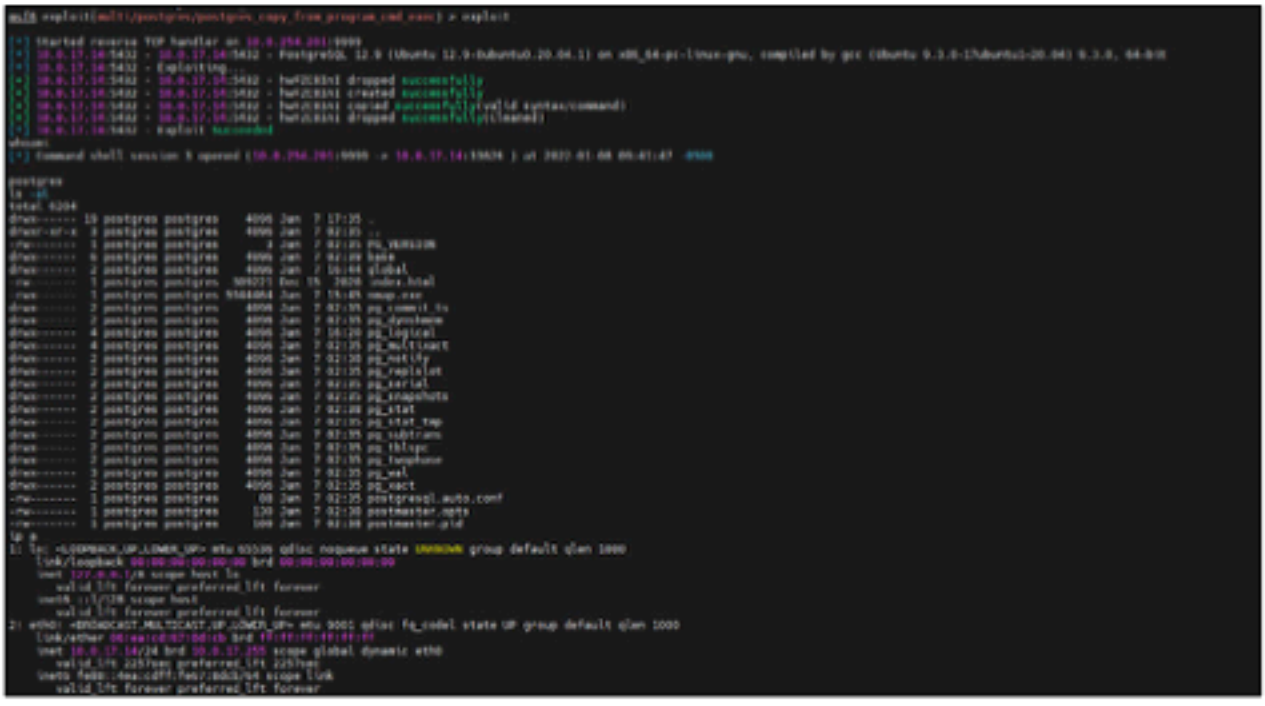

Figure 13: PostgreSQL Remote Code Execution

Recommended Remediation:

Users in "pg_read_server_files' can run commands in only a certain instance. This instance is when it is a superuser or users with "pg_execute_server_program" permission are required in order to be able to exploit this vulnerability. We highly recommend having no users with this permission as this can mitigate the vulnerability. Also, because any superuser in the postgresql database can still take advantage of the vulnerability, we recommend making it so only localhost can connect to the superuser. This also means that any host that uses the administrator account should be changed to a low level user with least privileges depending on the level of access required for the functionality of the database.

In addition, implementing firewall egress filtering can make remote code execution much more difficult.

References:

`https://meterpreter.org/cve-2019-9193-postgresql-arbitrary-code-execution/`

`https://nvd.nist.gov/vuln/detail/CVE-2019-9193#vulnCurrentDescriptionTitle`

4.3 Moderate Risk

4.3.1 Payment Transaction Enumeration

Threat Level: Moderate (4.5)

Description:

The Jawbreaker portal on the eggdicator host allows users to enter transaction IDs and returns transaction information including the amount, customer_id, and status. All customer transactions can be enumerated without any authentication (see Figure 14).

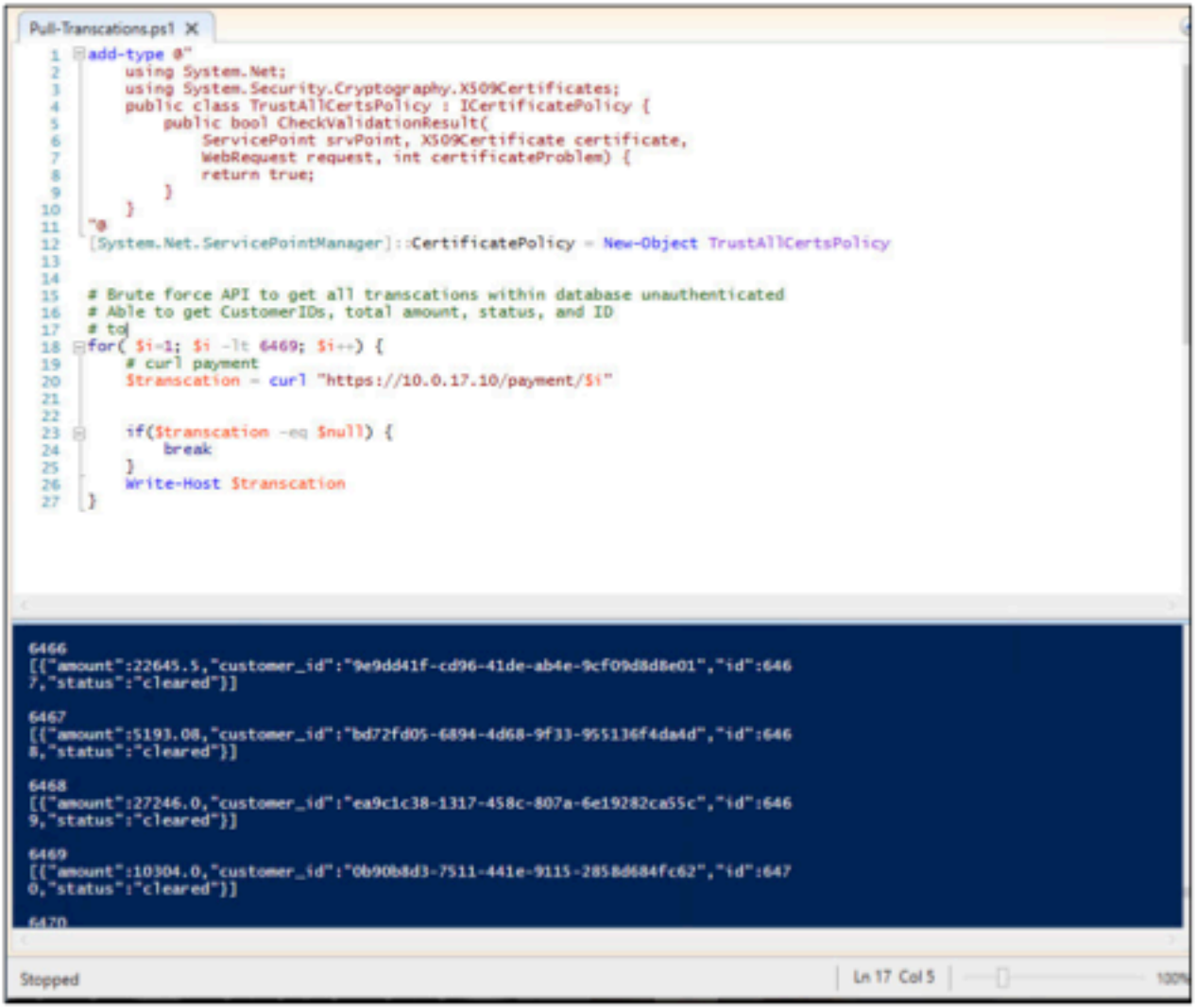

Figure 14: Powershell Script to pull all Customer transactions.

Potential Business Impact:

Potential attackers could extract all transactions regarding the revenue of LBC. Although theses transactions only refer to the customer as a UUID, the value is unique to that customer and could be used to associate multiple transactions to specific purchasers.

Affected Hosts:

Eggdicator (10.0.17.10)

Exploitation Details:

The powershell script shown above can be used by navigating to *https://10.0.17.10/payment/$i* and replacing "$i" with a numeric value. This would return JSON data regarding a transaction if one exists with the given ID. Based on results obtained from the script, a total of 6469 transactions were present and extractable.

Recommended Remediation:

Using a UUID instead of an id for each transaction would mitigate sequential enumeration and would greatly increase the time needed for a brute force attack. Additionally, rate limiting hosts to only a specific number of requests per minute (more information can be found in references) would further mitigate the attack. Moreover, implementing authentication on the API would limit enumeration to only authorized users. Access tokens or basic http authentication are both options which would help reduce the risk introduced by this vulnerability.

References:

`https://www.nginx.com/blog/rate-limiting-nginx/` `https://docs.nginx.com/nginx/admin-guide/security-controls/configuring-http-basic-authentication/`

4.4 Low Risk

4.4.1 Music Player Daemon (MPD) Directory Traversal

Threat Level: Low (2)

Description:

A directory traversal vulnerability in Music Player Daemon (MPD) allowed listing of directory contents in any location on the host machine, including the *root* directory.

```
Last-Modified: 2019-12-03T21:50:49Z
directory: misc
Last-Modified: 2020-04-15T11:09:51Z
directory: apport
Last-Modified: 2022-01-07T07:22:52Z
directory: sudo
Last-Modified: 2022-01-04T23:11:34Z
OK
listfiles ../../../home
ACK [52@0] {listfiles} Failed to open /var/lib/mpd/music/../../../home: No such file or directory
listfiles ../../../../home/
directory: ubuntu
Last-Modified: 2022-01-07T07:22:59Z
OK
```

Figure 15: Music Player Daemon Directory Traversal

Potential Business Impact:

The impact of this vulnerability is minimal, as only metadata about files on the device can be viewed, not their contents. Any sensitive information found in filenames would be exposed, however, including usernames.

Affected Host:

Rockbox (10.0.17.87)

Exploitation Details:

After connecting to *Rockbox* using a command-line network utility such as *telnet*, a relative path can be given to the *listfiles* command to view any directory on the machine. For example, the following commands would list the contents of the *root* directory.

```
telnet 10.0.17.15 11211
listfiles ../../../../../root
```

Recommended Remediation:

Ensure proper configuration of the MPD server to restrict directory access to only those needed for proper function of the music player. Additionally, run the daemon as a user other than *root*, to prevent listing the contents of */root*.

4.4.2 ScadaBR Reflected XSS (Username)

Threat Level: Low (1.25)

Description:

A cross-site scripting (XSS) vulnerability was found in the login portal of ScadaBR. A malicious actor could supply malicious Javascript to redirect users and steal cookies.

Figure 16: XSS Vulnerability in username input field

Potential Business Impact:

Because this vulnerability uses reflected XSS instead of stored XSS, phishing would most likely be required for a successful exploit. However, an employee of LBCfalling for a phishing campaign could lead to drive-by downloads and cookie theft, resulting in the potential compromise of machines on the network.

Affected Host:

Crunch (10.0.17.50)

Crunchserial (10.0.17.51)

Exploitation Details:

Results can be reproduced by navigating to *http://10.0.17.50:9090/ScadaBR/* (without having been authenticated beforehand) and typing the following in the username field:

```
admin"><script>alert('XSS')</script>
```

This will result in a reflected cross-site vulnerability displaying an alert on the page with the text "XSS". While the alert text is only an example, a threat actor could include arbitrary JavaScript in the payload.

Recommended Remediation:

Validate & sanitize all form fields to prevent XSS attacks. Use a Web application firewall (WAF) to block the execution of malicious scripts. Additionally, convert all alphanumeric

characters to HTML character entities before displaying user input. To ensure that cookies cannot be stolen, it is recommended to include in the headers "Secure" and "HttpOnly" so that cookies are not accessible to unintended parties and are sent over HTTPS. At the moment, the web traffic on the ScadaBR server is not encrypted. The Secure header can only be implemented once TLS is implemented and enabled on the host.

References:

`https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies`

CEHv11 Ethical Hacking and Countermeasures - Volume 2

4.4.3 ScadaBR Reflected XSS (URL)

Threat Level: Low (1)

Description:

A crafted link directing a user to the ScadaBR web server can cause the execution of arbitrary JavaScript. An attacker can also initiate unauthorized data transmission on behalf of the victim.

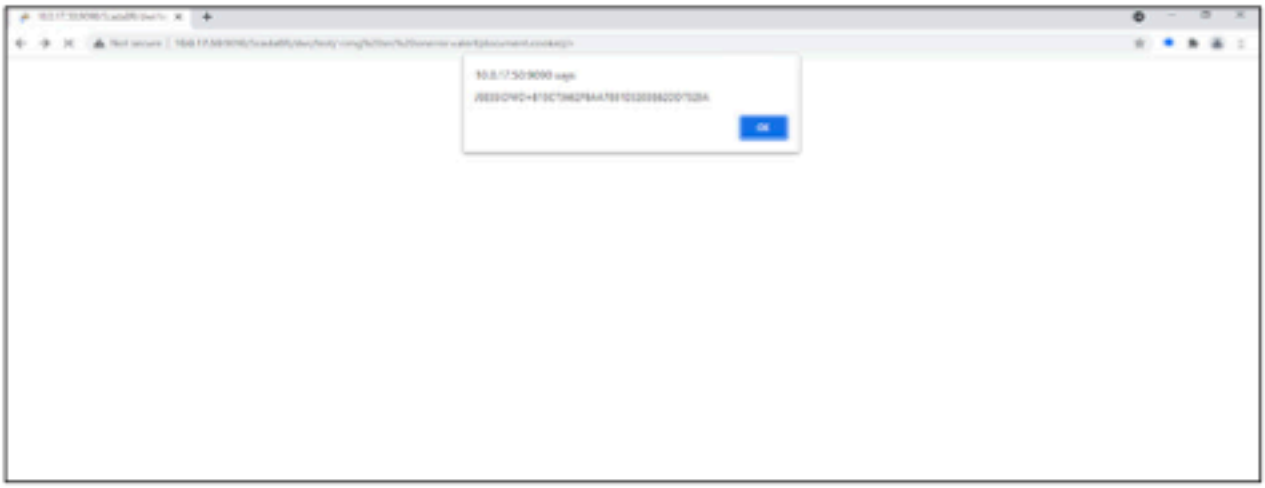

Figure 17: Relfected XSS Vulnerability

Potential Business Impact:

This reflected XSS attack requires victims to execute or click a link. For this reason, the attack is considered somewhat difficult to execute (see Figure 17). If a victim did click on this link, attackers would be able to steal user cookies. This could result in the compromise of more machines and would remove integrity within these systems.

Affected Host:

Crunch (10.0.17.50)

Exploitation Details:

Results could be reproduced by navigating to the following url:

```
http://10.0.17.50:9090/ScadaBR/dwr/test/<img%20src%20onerror=alert(
document.cookie)/>
```

This would result in a reflected cross-site scripting vulnerability and would broadcast the users' JSESSIONID back to their machine.

Recommended Remediation:

See Section 4.4.2, as this vulnerability can be remediated in the same fashion. It is recommended blocking this directory as it is a default directory served in the ScadaBR directory which can be accessed by unauthenticated users. Within tomcat, a system administrator can define directories that can not be accessed on the website. The configuration file found at "WEB-INF/web.xml" within the ScadaBR directory can be modified to accomplish this. More information can be found in the references.

References:

`https://stackoverflow.com/questions/13815997/restrict-file-access-in-tomcat`

`https://jelastic.com/blog/restrict-access-tomcat-web-application-hosting/`

4.4.4 Memcache Server Null Authentication

Threat Level: Low (0.75)

Description:

A null session on Memcached 1.5.6 can be used to enumerate information within the service. Memcached caches data, and provides information on memory usage and distribution of information through its slab allocation.

```
└─# nc 10.0.17.15 11211                                    1 ×
version
VERSION 1.5.6 Ubuntu
stats
STAT pid 8767
STAT uptime 36450
STAT time 1641577353
STAT version 1.5.6 Ubuntu
STAT libevent 2.1.8-stable
STAT pointer_size 64
STAT rusage_user 3.312714
STAT rusage_system 2.968853
STAT max_connections 1024
STAT curr_connections 1
STAT total_connections 14
STAT rejected_connections 0
STAT connection_structures 5
STAT reserved_fds 20
STAT cmd_get 1
STAT cmd_set 0
STAT cmd_flush 0
STAT cmd_touch 0
STAT get_hits 0
STAT get_misses 1
STAT get_expired 0
STAT get_flushed 0
STAT delete_misses 0
STAT delete_hits 0
STAT incr_misses 0
STAT incr_hits 0
STAT decr_misses 0
STAT decr_hits 0
STAT cas_misses 0
STAT cas_hits 0
STAT cas_badval 0
STAT touch_hits 0
STAT touch_misses 0
STAT auth_cmds 0
STAT auth_errors 0
STAT bytes_read 365
STAT bytes_written 18780
STAT limit_maxbytes 67108864
STAT accepting_conns 1
STAT listen_disabled_num 0
STAT time_in_listen_disabled_us 0
STAT threads 4
STAT conn_yields 0
STAT hash_power_level 16
STAT hash_bytes 524288
STAT hash_is_expanding 0
STAT slab_reassign_rescues 0
STAT slab_reassign_chunk_rescues 0
STAT slab_reassign_evictions_nomem 0
STAT slab_reassign_inline_reclaim 0
STAT slab_reassign_busy_items 0
```

Figure 18: Listing memcached statistics

Potential Business Impact:

Memcached could potentially save database query results from applications. If an application was querying databases with PII information, an attacker could steal the data because no password would be required to connect to the service.

Affected Host:

Bucket (10.0.17.15)

Exploitation Details:

An attacker with the network could connect to the memcache server by using the command:

```
nc 10.0.17.15 11211
```

Additionally, the following command could be used to print general statistics about the memcache instance:

```
stats
```

Recommended Remediation:

Implement authentication on the device with a strong password. Steps to enable authentication on memcached can be found in *references*.

References:

`https://memcached.org/`
`https://docs.ovh.com/us/en/dedicated/securing-server-with-memcached-service/`

4.5 Informational

4.5.1 Website API Stability

Description:

During the engagement, their were issues with the functionality of LBC's croissant marketplace because an API went down and was not functioning properly. The API's downtime resulted in functionality of the marketplace being unusable (e.g. Authentication was not possible).

Potential Business Impact:

When the API goes down it can result in loss of profits because customers cannot properly interact with the marketplace meaning they cannot place orders and use other functionality. The reason for this is this API is used for authentication, buying, and more.

Affected Host:

scrumdiddlyumptious (10.0.17.12)

whatchamacalit (10.0.17.13)

Recommended Remediation:

To help mitigate the potential downtime of the marketplace's used API it is recommended to use a load balancer to enable reduced downtime. By using a load balancer it can ensure that if one of the servers were to go down it would not hinder operations as their can be alternative server(s) that are utilized. Some potential service options for load balancing include:

- NGINX HTTP Load Balancing
- IBM's Load Balancer
- AWS' Elastic Load Balancing (ELB)

References:

`https://docs.nginx.com/nginx/admin-guide/load-balancer/http-load-balancer/`

`https://www.ibm.com/cloud/load-balancer`

`https://aws.amazon.com/elasticloadbalancing/`

4.5.2 Improper Data Encryption

Description:

PostgreSQL server stores customer credit card information in clear text. See section 4.1.3 for more information.

MariaDB server stored login email and password information in base64 encoding. This data is located in the 'logins' table.

Potential Business Impact:

Base64 is not an encryption, but a type of encoding that is easily reversible. The passwords obtained can be used to log into users accounts on the Le Bonbon Croissant Marketplace.

```
MariaDB [wmci]> select * from logins limit 10;
select * from logins limit 10;
+--------------------------------------+----------------------------------------------+----------------+------------+
| login_id                             | login_name                                   | login_pass     | login_role |
+--------------------------------------+----------------------------------------------+----------------+------------+
| 9a7af703-88b9-4fe3-9c29-6f2c63e7b846 | [redacted]                                   | [redacted]     |          1 |
| 06cb02af-e405-41e5-ace8-bdd13ecfc0ae |                                              |                |          1 |
| d8dfc68b-b801-4a8c-b238-c75c38a83fe6 |                                              |                |          1 |
| 670eb60d-1bbd-4684-9bb1-4770cd635ff1 |                                              |                |          1 |
| fc19d7e8-8d2b-46c8-b96f-0ebd0b4ae9bf |                                              |                |          1 |
| 6a85a68c-3e05-40ec-8a69-c72e05e36238 |                                              |                |          1 |
| 4c8265d2-a22b-4d49-8856-00c92ed4974b |                                              |                |          1 |
| b4affb7e-ee4a-4731-b508-4879dcf74eb4 |                                              |                |          1 |
| 92e421b1-5e74-4301-bfa6-83a6cd872ac3 |                                              |                |          1 |
| 608393fa-a651-4e08-9814-ca14aabb15fa |                                              |                |          1 |
+--------------------------------------+----------------------------------------------+----------------+------------+
10 rows in set (0.000 sec)

MariaDB [wmci]>
```

Figure 19: Table of usernames and base64 encoded passwords.

Affected Host:

Charley (10.0.17.14)

Recommended Remediation:

It is recommended that the passwords are hashed (e.g. SHA-256) before being stored in a database. Hashes can't be reversed, allowing for it to be safely stored.

References:

`https://auth0.com/blog/hashing-passwords-one-way-road-to-security/`

4.5.3 No Password Policies

Description:

[redacted] was unable to locate any evidence of password policies throughout the network which could have prevented some of the vulnerabilities detailed within this report.

Potential Business Impact:

In the event LBC does not implement password policies, they could be subject to fines as a result of PCI DSS violations. Specifically, having no password policies results in violation of requirements 8.2.3, 8.2.4, 8.2.5, and 8.2.6.

Recommended Remediation:

The minimum action required to be compliant with PCI DSS' password policy requirements would be to implement the following password policy:

- Seven character minimum
- At least one numeric character
- At least one alphabetic character
- 90 day password expiration
- New password cannot match previous four passwords for a user

References:

`https://www.pcisecuritystandards.org/documents/SAQ_D_v3_Merchant.pdf`

4.5.4 Insufficient Firewalls

Description:

During the engagement, many of the services and hosts interacted displayed insufficient Firewall Rules.

Potential Business Impact: This could save potential incidents of breach as by enabling firewalls with strong rules could thwart attackers from gaining unauthorized access to systems that potentially contain security vulnerabilities such as remote code execution.

Recommended Remediation: On hosts setup up proper firewalls using iptables, ufw, or other software. Examples of these can be found in Section 4.2.2. PCI DSS requires firewall zone-based controls between trusted and untrusted zones. Some best practices for firewalls are to:

- Block traffic by default
- Set Explicit Firewall Rules First
- Establish firewall configuration change plan
- Optimize firewall rules
- Update Firewall Software Regularly

Additionally, it may also be beneficial for LBC to implement intrusion-detection and/or intrusion-prevention systems on the network to help with detecting and preventing future exploitation of the network.

References:

`https://backbox.com/7-firewall-best-practices-for-securing-your-network/`
`https://www.checkpoint.com/cyber-hub/network-security/what-is-firewall/8-firewall-best-practices-for-securing-the-network/`

4.5.5 Lack of PCI DSS Compliance Required Documentation

Description:

No evidence was found that displayed an effort to retain the documentation which is required to maintain PCI DSS compliance. Additionally, when [redacted] inquired with LBC about this lack of documentation, LBC staff stated that no physical paper copies of such documentation exist, thus if such documentation did exist it would be present on their systems.

Potential Business Impact:

This lack of documentation per PCI DSS requirements results in numerous compliance violations which were previously detailed in Section 2.3.1. As a result, LBC could be liable for fines that could potentially be levied by their credit card processor.

Recommended Remediation:

[redacted] recommends that LBC take action as soon as possible to create the documentation necessary to satisfy the requirements for PCI DSS. Specifically the documentation requirements outlined in their SAQ D Version 3 which is designed for merchant compliance.

To assist, [redacted] has compiled the following list of requirements which are documentation related for which LBC is required to comply with:

- PCI DSS 2.2
- PCI DSS 2.4
- PCI DSS 2.5
- PCI DSS 3.6
- PCI DSS 3.7
- PCI DSS 6.2
- PCI DSS 6.4.5
- PCI DSS 6.5
- PCI DSS 6.7
- PCI DSS 8.8
- PCI DSS 11.6

References:

`https://www.pcisecuritystandards.org/documents/SAQ_D_v3_Merchant.pdf`

5 Conclusion

LBC provides the community with delicious confections. The stability and integrity of the system that provides pastries is necessary to maintaining the luxuries of modern life. It is for this reason that the threats outlined in this report should be taken seriously and immediately remedied. The gravity of the vulnerabilities described above cannot be understated. An attacker with the same level of access as what [redacted] was granted for this engagement could gain direct control of the industrial control systems of LBC's factory. If this were to happen in an uncontrolled setting, the vulnerabilities would be so grave that LBC might be liable for negligence in the result of loss of life and externalities as a result of such an attack. Additionally, LBC may incur penalties for violations of PCI DSS requirements.

[redacted] hopes that this relationship with LBC will continue in the future. We look forward to working together to ensure the vulnerabilities discussed in this report are properly remediated and to help continually improve LBC's security posture.

References

[1] *PCI Compliance Fees, Fines, and Penalties: What Happens After a Breach?* Apr. 2020. URL: https://www.lbmc.com/blog/pci-compliance-fees-fines-penalties/.

[2] *Main Page*. URL: http://www.pentest-standard.org/index.php/Main_Page.

[3] *Mitre ATT&CK®*. URL: https://attack.mitre.org/.

[4] *Introduction*. URL: https://owasp.org/Top10/.

[5] Joint Task Force. *Security and Privacy Controls for Information Systems and Organizations*. Dec. 2020. URL: https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/final.

[6] Forum of Incident Response and Inc Security Teams. *CVSS v3.1 Specification Document*. https://www.first.org/cvss/v3.1/specification-document. 2019.

[7] National Institute of Standards and Technology. *Common Vulnerability Scoring System Calculator*. https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator.

[8] *Financial Affairs*. URL: https://financial.ucsc.edu/pages/security_penalties.aspx.

Appendices

A Network Topology

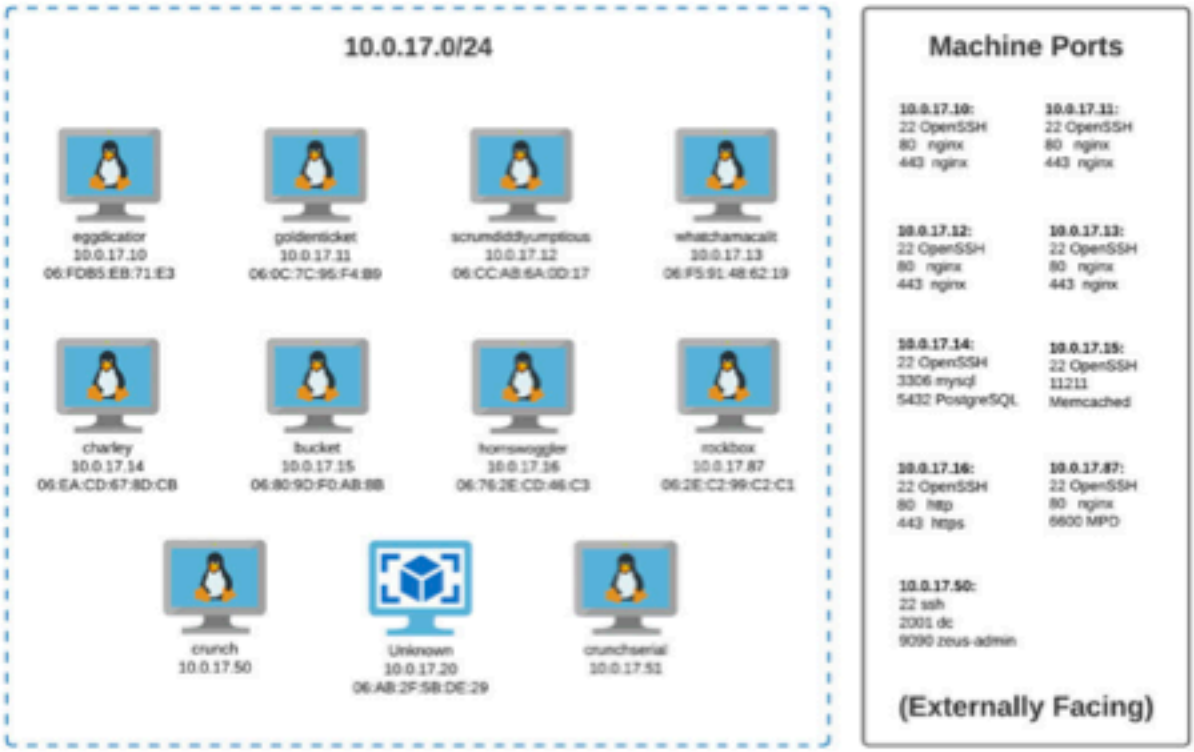

Figure 20: Network Topology

B Tools

Name	Description	Link
Nmap	Network and vulnerability scanner	https://nmap.org/
Metasploit	Exploitation framework	https://github.com/rapid7/metasploit-framework
DIRB	Directory Brute Force Tool	https://github.com/v0re/dirb
Gobuster	Directory Brute Force Tool	https://github.com/OJ/gobuster
Meterpreter	Reverse Shell	https://github.com/rapid7/meterpreter
Crowbar	Brute forcing tool	https://github.com/galkan/crowbar
netcat	Network utility	https://github.com/diegocr/netcat
hydra	Brute Forcing tool	https://github.com/vanhauser-thc/thc-hydra
Wireshark	Network traffic analyzer	https://www.wireshark.org/
Portswigger Burp Suite	Web traffic analysis tool	https://portswigger.net/burp
psql	PostgreSQL interactive terminal	https://www.postgresql.org/docs/13/app-psql.html