The Cozy Croissant

Security Assessment Report

[redacted] Finals

14 January 2023

TABLE OF CONTENTS

Introduction

The Firm ([redacted]) was contracted by the Client, The Cozy Croissant (TCC), to conduct a penetration test to assess the security of the company's network and identify vulnerabilities to cyber attacks. Prior to the penetration test, both parties agreed to the terms of engagement that were determined by TCC. This portion of the penetration test was performed on-site from January 13th-14th, 2023 and was a continuation of an initial security assessment that was performed on [redacted] 2022. Various methods and tools were used throughout the test to discover security vulnerabilities on the network. New security solutions had been set in place since the first security assessment, and various methods were used to attempt circumventing those new solutions. All methods and tools were used in ways to simulate realistic threat actor activities. The following goals were focused on throughout this security assessment:

- Identify methods to circumvent new security measures set in place
- Identify ways a remote threat actor could gain initial access to the network
- Determine privilege escalation paths a threat actor could take to gain higher access
- Determine the overall impact that each vulnerability would have to the TCC environment

Key Assessment Details

Scope

The scope for this penetration test was limited to the 10.0.0.0/24 subnet and the 10.0.200.0/24 subnet in the Client's network. As an additional facet to the scope, a UBesGoo safe was provided for the physical security assessment. The social engineering portion of the test was limited to one vishing call to a TCC employee, and pre-engagement OSINT. All tools that were used were open-source, free, and did not require an account to use.

Key Strengths

Four key points stood out as positions of strength during this penetration test. First, the network access controls lists (ACLs) that were set in place significantly improved the security in the TCC environment. We observed that these ACLs made it difficult to find services that were previously visible and accessible to us in the previous engagement. The ACLs successfully removed much unnecessary user access to these services, thus improving the security of these services.

Second, Employee security awareness was observed to be greatly improved from engagement one, to engagement two. Throughout the entirety of Engagement two, all interactions with TCC employees showed great interest in how they could take actions to improve the security posture of TCC. It was clear that a culture of Security was emphatically present from top to bottom. In the specific Social engineering aspect of Engagement #2, a vishing call, the employee who answered our call did not give any customer information out, and was impervious to attempts to draw additional data out.

Third, the password policy was updated to now require a minimum of twelve characters. This password policy greatly improved the security of user credentials. These longer passwords would significantly increase the difficulty of password cracking and guessing, both of which are common attack methods that threat actors take to discover user credentials.

Fourth, blocking remote desktop access for users. The ability to remote desktop into the devices on the TCC network should only be provided to authorized users. Limiting this usage has greatly reduced the amount of unnecessary user access into the system.

Main Areas of Improvement

The foremost area found in this penetration test for the Client to improve is ensuring that support and service accounts are not using weak or default credentials. There were many weak credentials that were found being used on support and service accounts in TCC's environment. These credentials pose a security vulnerability because they are easy to guess and could provide attackers with access to many critical services on this network.

Another area that we found to be in need of improvement was the disabling of default guest and admin accounts on the hosts. Having the guest account enabled allows anyone to log into that account and have any access that comes with it. The default admin account has many known vulnerabilities that come with it and should thus be disabled as well, as not to provide that as an attack vector for threat actors.

Next, every host, no matter the operating system, should have regular and consistent updates applied. The windows hosts examined within the TCC Corporation network had no updates applied to them ever. With Operating systems as old as 2016, this is of paramount importance and will immensely reduce the number of vulnerabilities. Keeping hosts running at the most up to date operating systems themselves is critical not just for security reasons but will also improve performance as well.

Finally, implementing redundancy in the form of containerization, round robin setup or other backup solutions will allow for greater business uptime for critical systems. In the testing performed by the Engagement team, some services were unintentionally taken down by seemingly innocuous actions. To prevent this and to operate with good technical hygiene, having a solution in place to ensure as little service downtime as possible is highly recommended.

Frameworks and Models

Assessment Methodology

We took a breadth-first approach to conduct this penetration test. To start, we divided the target hosts evenly and assigned them to each team member. On our assigned hosts, we used the Nmap tool to perform initial port scanning and service enumeration. After our initial assessment of the running services on each host, we started investigating the services in more detail, searching for any weaknesses and vulnerabilities that could provide us access into the network. We also referred to our report from our initial engagement and tested the vulnerabilities that had been previously discovered. From our investigations, we were able to gain access into the network. From there, we searched for and tested various methods to achieve more access, gain higher privileges, and find possible ways to move laterally through the network. Throughout our assessment, we shared information and discoveries with all team members, as sharing information and collaborating would improve our effectiveness in this penetration test. We utilized the tool called Dradis to organize and share all information and notable findings.

Severity Definitions - CVSS

Severity levels are determined by the Common Vulnerability Scoring System, or CVSS, a commonly used system in penetration tests worldwide. CVSS scores are calculated from a variety of inputs, such as the impact on Confidentiality/Integrity/Availability, attack complexity, privileges required if user interaction is required, and more. CVSS specifications and subcategories can be seen below:

Risk Ratings

The risk rating scale that we have used for this report is a scale from 0-5 with 5 being the highest risk and 0 being the lowest. These risk ratings are used to show the determined likelihood of a vulnerability occurring on the Client's network. The higher the risk score, the more likely the vulnerability will be exploited. The risk rating is determined by using our best judgment given our knowledge of the environment to figure out a likelihood of exploitation, as well as the CVSS score. If the vulnerability has a low CVSS score, but would be likely exploited in the Client's environment, then the risk rating will be higher than a high CVSS score vulnerability that barely affects the Client's environment. Below is a table indicating the risk rating scale:

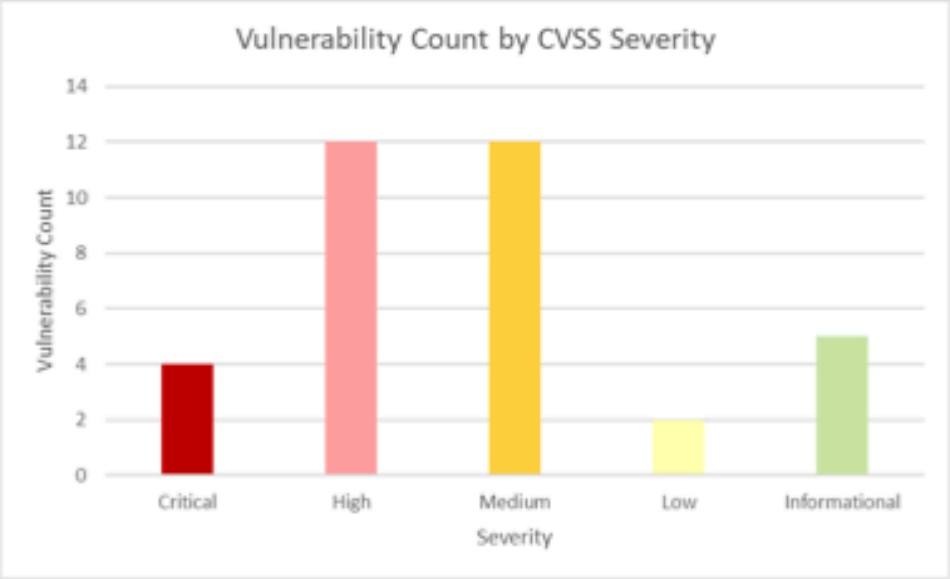
	CVSS Level				
Likelihood		Low	Medium	High	Critical
	Low	0	1	2	3
	Moderate	1	2	3	4
	High	2	3	4	5

Compliance

The Cozy Croissant stores credit card information from their customers and should handle this information in accordance with the PCI-DSS compliance requirements. To be PCI-DSS compliant, all credit card information being stored needs to be encrypted to provide a layer of confidentiality to the data. Additionally, CVV numbers should not be stored in any capacity. This helps to ensure that the credit card information stays separated as to decrease the chances of attackers gaining all the credit card information they need in order to use it. Credit card information is extremely sensitive and failing to store it in compliance with PCI-DSS could result in major fines ranging from $5,000 to $10,000 per month.

Technical Findings

Overview



Over the course of the engagement we observed that the Windows systems were in general more susceptible to our attacks than the Linux-based systems. In particular, the domain controllers were highly vulnerable to several easy-to-exploit CVEs, commonly known as Zerologon, PetitPotam, and noPac. Most of the vulnerabilities discovered were in the Corporate network, which we had not gained access to before the network ACLs were removed. This indicates that the ACLs do a good job of slowing attackers seeking vulnerabilities.

Machine Assessment

IP Address	Operating System	Hostname	Machine Description	Vulnerability Count
10.0.0.2	[unknown]	[unknown]	Likely router	0
10.0.0.5	Windows Server 2016	DC01	Corporate Domain Controller	9
10.0.0.6	Windows Server 2016	ADCS	Active Directory Certificate Services	6

10.0.0.7	Ubuntu 20.04	DOAPI	MongoDB server	3
10.0.0.11	Windows Server 2016	HMS	Hotel Management System	12
10.0.0.12	Ubuntu 20.04	LPS	My Rewards Web Server	13
10.0.0.20	Ubuntu 20.04	MEDIA	JellyFin Media Server	7
10.0.0.51	Windows 10	WORKSTATION01	Employee Workstation	5
10.0.0.52	Windows 10	WORKSTATION02	Employee Workstation	5
10.0.0.100	Ubuntu 20.04	LDAP	OpenLDAP Server for Customers	2
10.0.0.102	Ubuntu 20.04	PROFILER	LDAP Server web client	1
10.0.0.200	Ubuntu 20.04	PAYMENT-WEB	Web server for viewing payment info	6
10.0.0.210	Ubuntu 20.04	PAYMENT-DB	Database server for storing payment info	5
10.0.0.254	[unknown]	[unknown]	Likely router	0
10.0.200.2	[unknown]	[unknown]	Likely router	0
10.0.200.101	Windows Server 2016	KIOSK01	Kiosk	
10.0.200.102	Windows Server 2016	KIOSK02	Kiosk	
10.0.200.103	Windows Server 2016	KIOSK03	Kiosk	
10.0.200.104	Windows Server 2016	KIOSK04	Kiosk	
10.0.200.254	[unknown]	[unknown]	Likely router	0

Domain Vulnerable to Zerologon

Severity	Vector String	CVSS Score
Critical	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H	**10.0**
Risk Rating	**5**	Likelihood: **HIGH**
Affected Scope	10.0.0.5	

Details:
Zerologon (CVE-2020-1472) is a vulnerability in the NETLOGON protocol that allows an attacker with network access to the domain controller to get domain administrator credentials with just a few simple commands. The attack replaces the domain controller's password in Active Directory with an empty string, allowing an attacker to access the domain controller without issues.

Impact:
Due to the minimal complexity of this attack, and the proliferation of tutorials on the internet on how to perform the attack, the impact is quite high. Privileged access to the domain controller gives an attacker full control over the entire domain, allowing them to create/add any other machines, users, policies, or anything else they want to do on the domain.

This could not only lead to the addition of objects into the domain, but also the deletion of objects. Thus, with minimal effort an attacker could remove all objects in the domain. The impact of such actions would require many hours to fix, and depending on the attacker's actions, many thousands, if not millions of dollars in lost revenue. The attacker could entirely wipe the Cozy Croissant's payment manager, user database, rewards database, and remove kiosks so no one can check in.

Remediation:
There are a few steps that can be undertaken to minimize the risk of facing a compromise using this attack. First, Microsoft released a patch that can be put in place to prevent this attack. That patch can be found at the following link:
https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2020-1472

Second, Domain-joined machines that are not Windows machines are unable to be patched. Search for any domain-joined non-Windows machines that have elevated privileges (such as domain replication privileges) and, if possible, remove these privileges. They can be used to launch Zerologon attacks even against patched domain controllers.

Third, if patching/removing is not possible, the main thing to do is to enable multi-factor authentication on all accounts, as this will help prevent even a compromised account from accessing the domain. In addition, monitor for this sort of attack, particularly for anyone attempting to change the domain controller's Active Directory password.

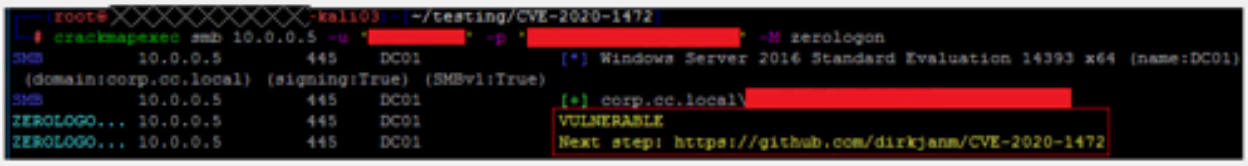
Steps for Reproduction:
The main thing an attacker needs to exploit this vulnerability is network access to the domain controller. With that access, all you, or an attacker, need to do to test the efficacy of the exploit (if they have credentials for at least one user in the domain) is:

Run the tool crackmapexec with specific flags to search for this vulnerability:

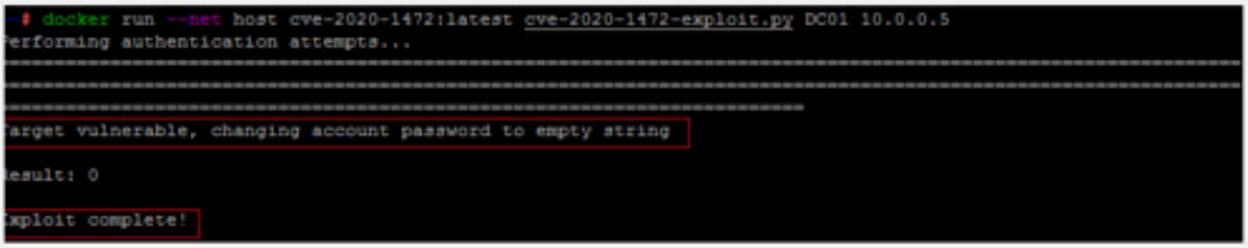
```
crackmapexec smb <DOMAIN_CONTROLLER_IP> -u '<USERNAME>' -p '<PASSWORD>' -M zerologon
```

To perform the exploit, which does not require any credentials, you can use the details found at the following link: https://www.sprocketsecurity.com/resources/how-to-exploit-zerologon.

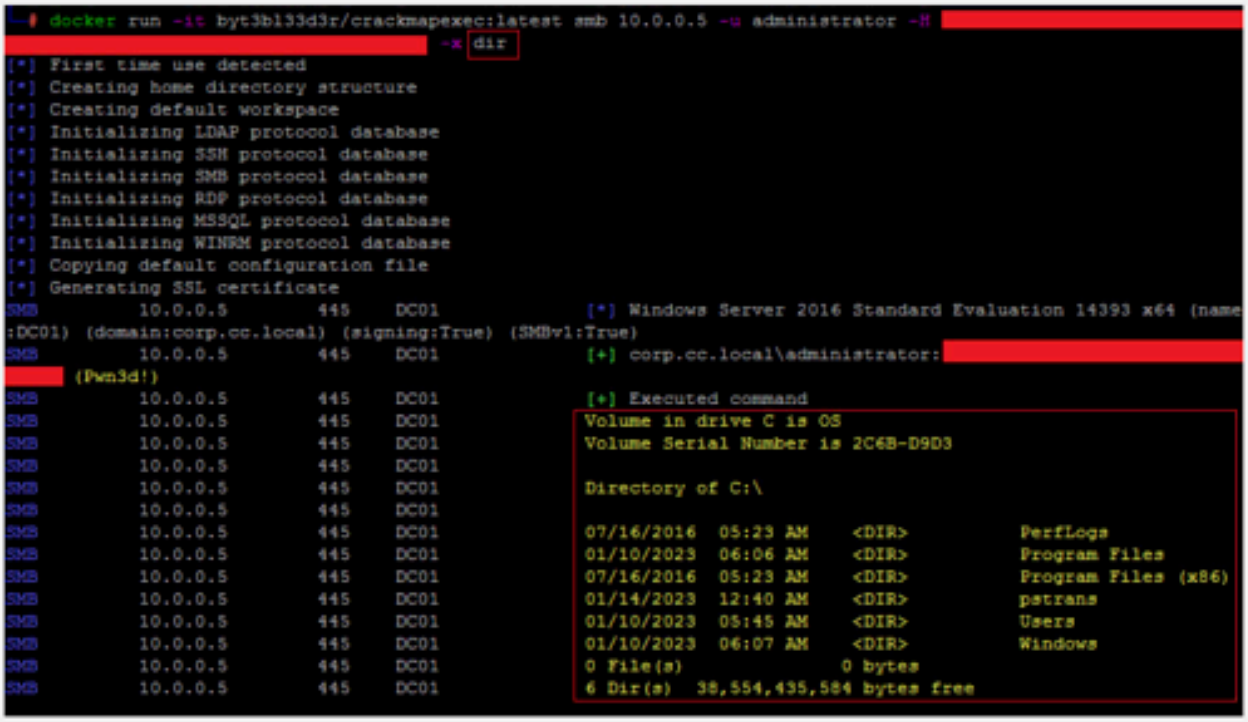
Relevant Screenshots



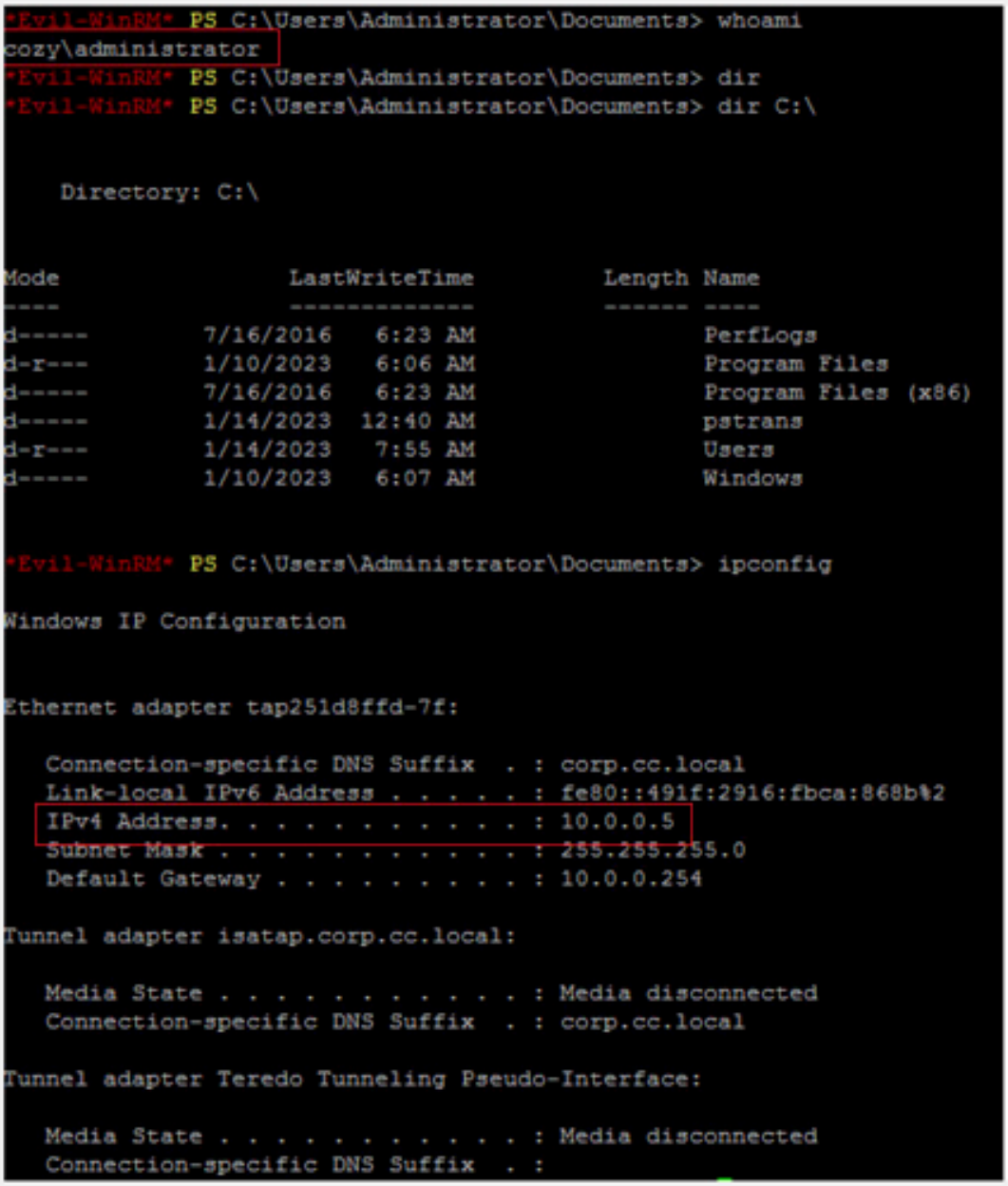
Checking whether the domain controller is vulnerable to Zerologon



Successfully exploiting the Zerologon vulnerability



Using crackmapexec to run arbitrary commands on the domain controller



Connecting to the domain controller via Evil-WinRM using the Administrator user's hash

Weak Credentials for Wordpress Site

Severity	Vector String	CVSS Score
Critical	CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H	**9.4**
Risk Rating	**5**	Likelihood: **HIGH**
Affected Scope	10.0.0.11	

Details:
Using the BurpSuite tool, we were able to send POST requests to the Hotel Management System. In these POST requests, we were able to modify the username and password that were being used. In the response of our requests, we were able to tell when the username and password were correct based on the `isAdmin` and `url` parameters that were returned. We were able to easily guess the admin credentials using this method due to the credentials being weak.

Impact:
Using weak credentials for admin accounts can greatly impact the TCC business. It can provide an easy way for attackers to obtain admin credentials and administrative access to critical systems, such as the Hotel Management System. In addition, when paired with the second vulnerability of a vulnerable theme, remote code access is achieved on the local machine.

Remediation:
This vulnerability can be remediated by changing the password to be long and complex. Long and complex passwords ensure that attackers are not able to easily guess them.

Steps for Reproduction:
1. Open the BurpSuite tool and, with the intercept on, navigate to the 10.0.0.11's HMS system in the BurpSuite browser (http://10.0.0.11)
2. Send the intercepted request to the Repeater
3. In the repeater, place the request found in the screenshot below with various username and password combinations.
4. Continue modifying the username and password parameters until the response returns `isAdmin` and `url`. When those parameters are returned, you know you have the correct username and password combination.

Relevant Screenshots

Request

Pretty Raw Hex

```
1 POST /xmlrpc.php HTTP/1.1
2 Host: 127.0.0.1
3 Connection: close
4 Content-Length: 168
5
6 <methodCall>
7   <methodName>
      wp.getUsersBlogs
    </methodName>
8   <params>
9     <param>
      <value>
        A[redacted]
      </value>
      </param>
10    <param>
      <value>
        [redacted]
      </value>
      </param>
11  </params>
12 </methodCall>
```

Response

Pretty Raw Hex Render

```
1 HTTP/1.1 200 OK
2 Date: Fri, 13 Jan 2023 20:55:56 GMT
3 Server: Apache/2.4.54 (Win64) OpenSSL/1.1.1p PHP/7.4.33
4 X-Powered-By: PHP/7.4.33
5 Set-Cookie: wp_sr_session_[redacted]
  ; expires=Sun, 15-Jan-2023 20:55:57 GMT; Max-Age=172800; path=/
6 Connection: close
7 Content-Length: 644
8 Content-Type: text/xml; charset=UTF-8
9
10 <?xml version="1.0" encoding="UTF-8"?>
11 <methodResponse>
12   <params>
13     <param>
14       <value>
15       <array><data>
16   <value><struct>
17   <member><name>isAdmin</name><value><boolean>1</boolean></value>
18   <member><name>url</name><value><string>http://127.0.0.1/</strin
19   <member><name>blogid</name><value><string>1</string></value></m
20   <member><name>blogName</name><value><string>The Cozy Croissant<
21   <member><name>xmlrpc</name><value><string>http://127.0.0.1/xmlr
```

The two values censored in the request are the username and password values. Since parameter values like `isAdmin` and `url` are returned, this means the username/password combination is correct.

Writable WordPress Plugin Directory

Severity	Vector String	CVSS Score
Critical	CVSS:3.1/AV:N/AC:L/PR:H/UI:N/S:C/C:H/I:H/A:H	**9.1**
Risk Rating	**5**	Likelihood: **HIGH**
Affected Scope	10.0.0.11	

Details:
The plugin directory on the webserver hosting the HMS WordPress website is writable by the service account, allowing an administrator on the portal to upload PHP code and execute.

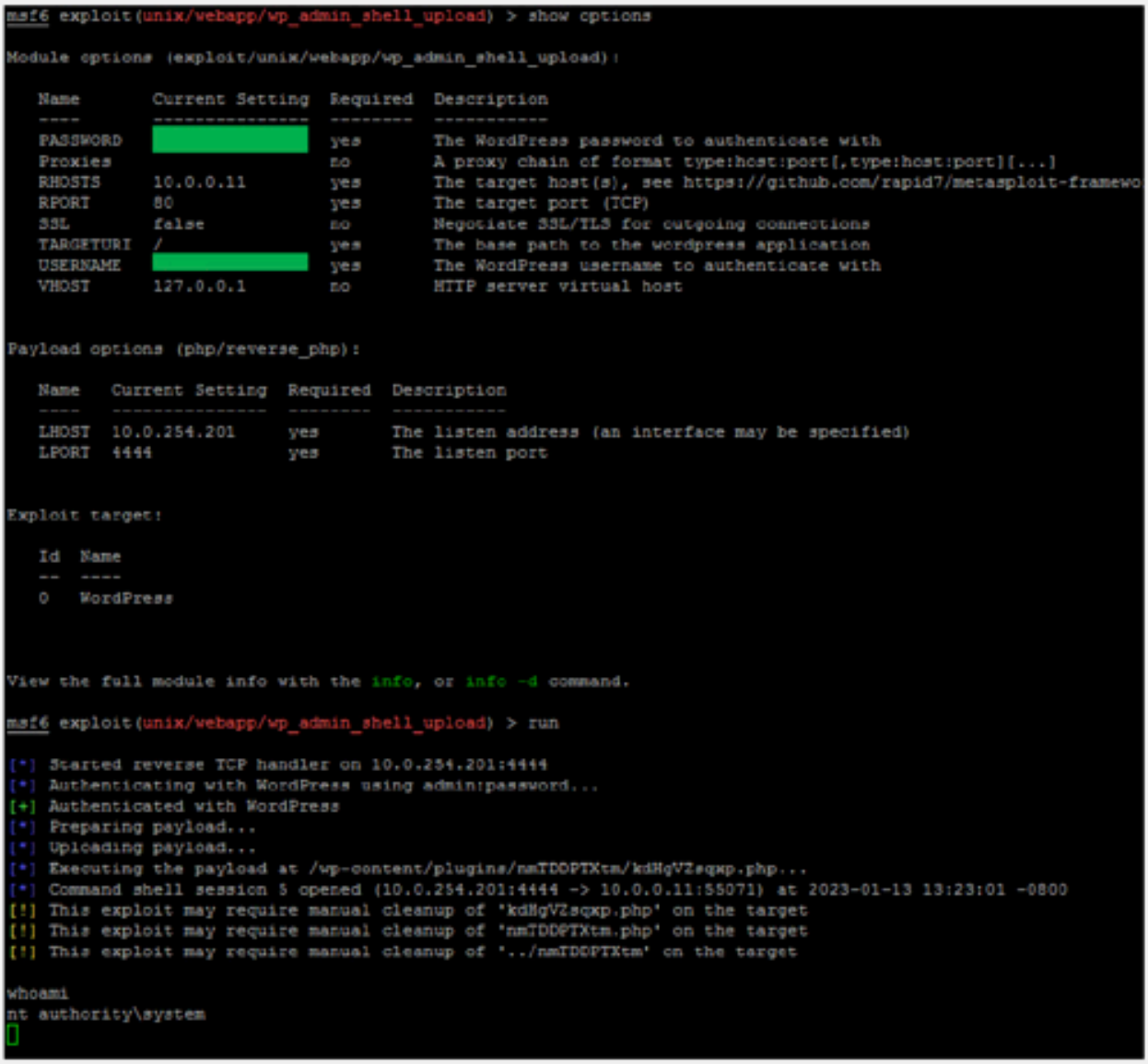
Impact:
This allows someone with only knowledge of the administrator password for the WordPress site to escalate their privileges to run arbitrary commands on the server. This increased attack surface may allow them to pivot to other accounts or machines and gain a stronger foothold in the network.

Remediation:
The plugin directory should be set to read-only for all users and write-only for root (`chmod -R 774 plugins/`).

Steps for Reproduction:

1. Open up Metasploit and use the module `unix/webapp/wp_admin_shell_upload` with the options set as shown in the screenshot below.
2. Execute the command run, and once you see the output `This exploit may require manual cleanup…`, then type any command such as `whoami`.

Relevant Screenshots



The combination of known admin credentials plus a vulnerable Wordpress theme allows an attacker to run arbitrary commands using a Metasploit module.

MySQL Default Credentials

Severity	Vector String	CVSS Score
High	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	**8.8**
Risk Rating	**4**	Likelihood: **HIGH**
Affected Scope	10.0.0.12	

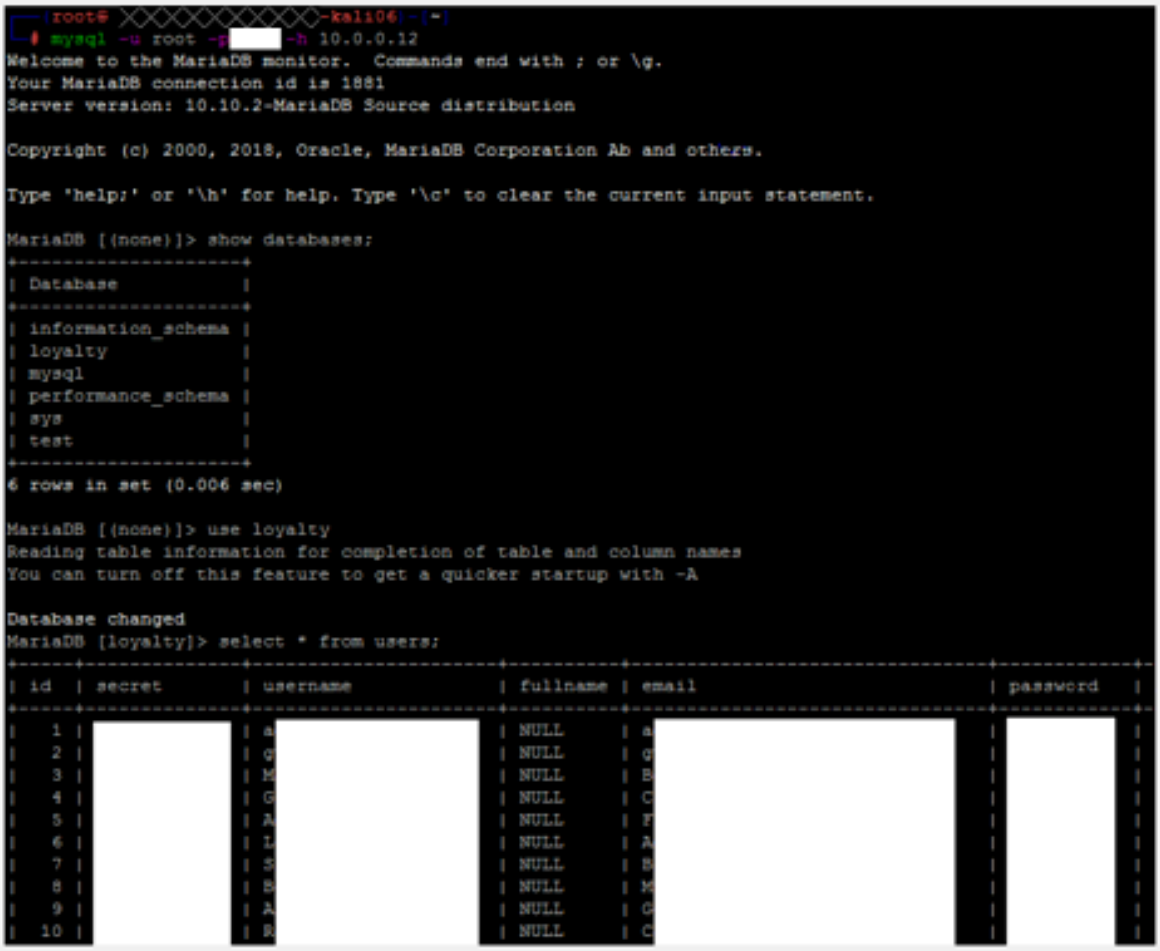
Details:
The host 10.0.0.12 (rewards.corp.cc.local) has a MariaDB instance running that is exposed on port 3306 that can be accessed by any host on the local network. We found that this MariaDB instance is configured to be using a set of default credentials for the root login. Logging in as this account gives access to the entire database and all of the information stored inside of it. Upon logging in with the default credentials, we discovered a table in one of the databases that includes the usernames, passwords, and emails in plaintext of the My Rewards system users.

Impact:
This vulnerability opens up the possibility for a bad actor to affect the integrity, confidentiality, and availability of the data in the rewards database.This is a risk to the company because anyone on the network can login using the root credentials, have access to all information in the databases, change or remove data, and potentially change the account passwords, effectively locking legitimate users out of the database.

Remediation:
The only step needed to remediate this vulnerability is to change the password for the root account of the MariaDB server to be a new password that is in accordance with the company's strong password policy. This will prevent unwanted parties from gaining access to the server through the use of default credentials.

Steps for Reproduction:
Connect to the MariaDB database: `$ mariadb -u root -p`
Switch to the mysql database: `> use mysql;`
Update the user password: `> ALTER USER 'root'@'localhost' IDENTIFIED BY '<new_password>';`
Flush privileges to clear stored credentials: `> FLUSH PRIVILEGES;`
Exit the database: `> exit`

Relevant Screenshots



Default credentials were used to login to the MariaDB server and show all data in the loyalty database

noPac

Severity	Vector String	CVSS Score
High	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	**8.8**
Risk Rating	**4**	Likelihood: **HIGH**
Affected Scope	10.0.0.5	

Details:
noPac is the combination of two separate CVEs (CVE-2021-42278 and CVE-2021-42287). CVE-2021-42278 results from an attacker improperly naming a computer in an Active Directory environment. To exploit this CVE, all an attacker needs to do is add a computer to the domain with the same name as the domain controller without the $ at the end. CVE-2021-42287 arises from the KDC in an Active Directory environment mishandling the scenario when it is trying to grant a ticket to a computer whose name does not exist.

Taken together, these vulnerabilities allow an attacker to create a computer with a name identical to the domain controller, request a ticket from the KDC, and receive a ticket with the privileges of the domain controller, due to the similarities of the names.

Impact:
The technical impact of the exploitation of this vulnerability is large. An attacker with the privileges of a domain admin can do whatever he or she wants in order to establish persistence, add and remove users or computers, change policies and settings, and more. A competent attacker would be able to lock out the defensive team and prevent them from responding to the attack.

The business impact is also very large. A common use of these privileges resulting from this attack vector is the deployment of ransomware throughout a domain, leading to the loss of critical systems and heavy potential financial losses.

Remediation:
Microsoft released patches for the two CVEs that should be applied to prevent the exploitation of the CVEs. These can be found at https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-42278 and https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-42287. In addition, monitoring for computer name resolution failures should be put in place.

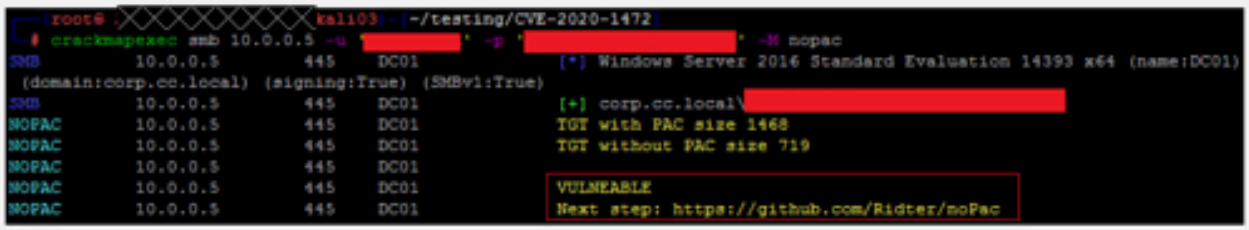
Steps for Reproduction:
The main things an attacker needs to exploit this vulnerability is network access to the domain controller and access to a user in the domain. With that access, all you, or an attacker, need to do to test the efficacy of the exploit (if they have credentials for at least one user in the domain) is:

Run the tool crackmapexec with specific flags to search for this vulnerability:

```
crackmapexec smb <DOMAIN_CONTROLLER_IP> -u '<USERNAME>' -p '<PASSWORD>' -M nopac
```

With the vulnerability confirmed, there is a tool that can be used to escalate privileges from standard domain account to domain admin, found at https://github.com/Ridter/noPac. Using the provided README you can find examples for getting a shell with elevated privileges, dumping hashes of domain users, and more.

Relevant Screenshots



Using crackmapexec to show the domain controller is vulnerable to the noPac exploit

Gaining shell access to the domain controller using noPac and showing privilege level

Dumping hashes using noPac and basic domain credentials

Payment Portal Log Injection

Severity	Vector String	CVSS Score
High	CVSS:3.1/AV:N/AC:H/PR:N/UI:R/S:C/C:H/I:H/A:L	**8.2**
Risk Rating	**2**	Likelihood: **LOW**
Affected Scope	10.0.0.200	

Details:
The data passed to the endpoint is placed directly into a python f-string, without going through any sort of validation. This allows a malicious actor to perform log injection into the logs on the server.

Impact:
This would allow an attacker to execute arbitrary code on the target machine, potentially leading to information disclosure and loss of file and system integrity.

Remediation:
Instead of using python f-strings to take input from the user directly, sanitization of the input should take place first, so that arbitrary code cannot be run upon opening a log.

Steps for Reproduction:
Navigate to http://10.0.0.200:8000/doc/#/admin/rooms/ in a browser and put some standard SQL injection payloads into the URL after the final slash. You can also go to the source code for that endpoint and see the direct use of the `room_id` variable in a SQL statement.

Relevant Screenshots

```python
@api.route('/admin/rooms/<string:room_id>', methods=['GET'])
def return_room_details(room_id):
    log_message = f"GET /admin/rooms/{room_id}; "
    room_details_query = f"SELECT * FROM wp_sr_room_types where id = {room_id}"
    log_message += f"Query: {room_details_query}; "
    try:
        res = query_db_hotel(room_details_query, 'select')
        log_message += f"Status: 200; Message: Information for {room_id} queried; Data: {jsonify(res)}"
        AppLog.info(log_message)
        return jsonify(res)
    except Exception as error:
        log_message += f"Status: 500; Message: {str(error)}"
        AppLog.error(log_message)
        return jsonify({"error":str(error)}), 500
```

No checks are made in the `room_id` user-supplied variable, meaning newlines could be inserted and arbitrary log values could be created.

Insecure Direct Object Reference in API Endpoints

Severity	Vector String	CVSS Score
High	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N	**8.1**
Risk Rating	**4**	Likelihood: **HIGH**
Affected Scope	10.0.0.200	

Details:
The Payment Portal on 10.0.0.200 allows customers to add, modify, and delete payment methods and invoices. While authentication is required, no checks are made server-side to ensure that the invoices or payment methods modified belong or are related to the user making the request.

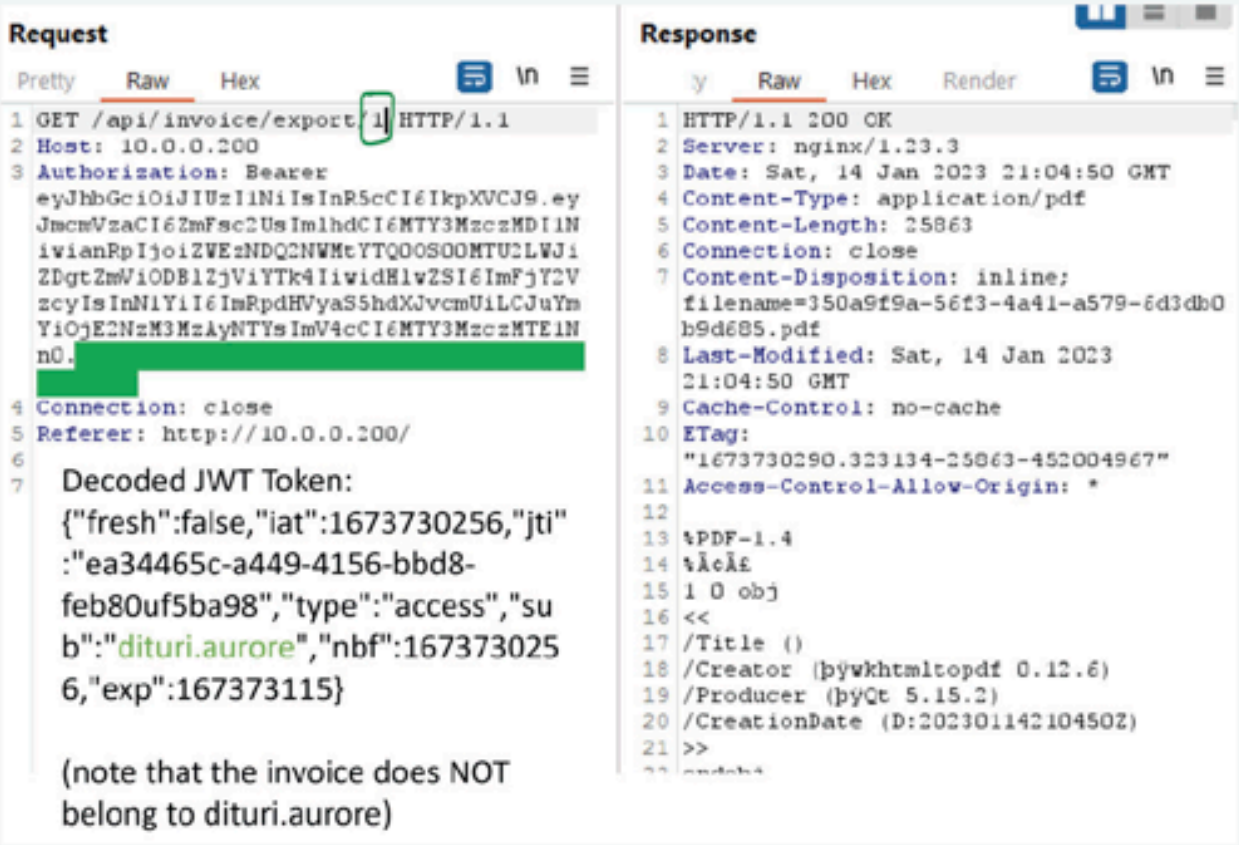
Impact:
This would allow users to modify and see the data for anyone else, potentially resulting in financial compromise for clients.

Remediation:
Checks should be made server-side to ensure the identity of the authorized user matches a field in the database associated with the data. This way, if a user accesses an invoice or payment method that is not linked to them, they will be denied and the information will not be returned, modified, or deleted.

Steps for Reproduction:

1. Sign into the Payment Portal at http://10.0.0.200, and access the URL http://10.0.0.200/api/invoice/export/x, where x is any number.
2. Notice that the invoice is returned, regardless of who it belongs to.

Relevant Screenshots



A valid JWT token for the user dituri.aurore is used to request the invoice with the id 1. The PDF of the invoice is returned, even though the invoice does not belong to this user.

Plaintext Credentials in Windows Kiosk

Severity	Vector String	CVSS Score
High	CVSS:3.1/AV:L/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:L	**8.0**
Risk Rating	**4**	Likelihood: **HIGH**
Affected Scope	10.0.200.103	

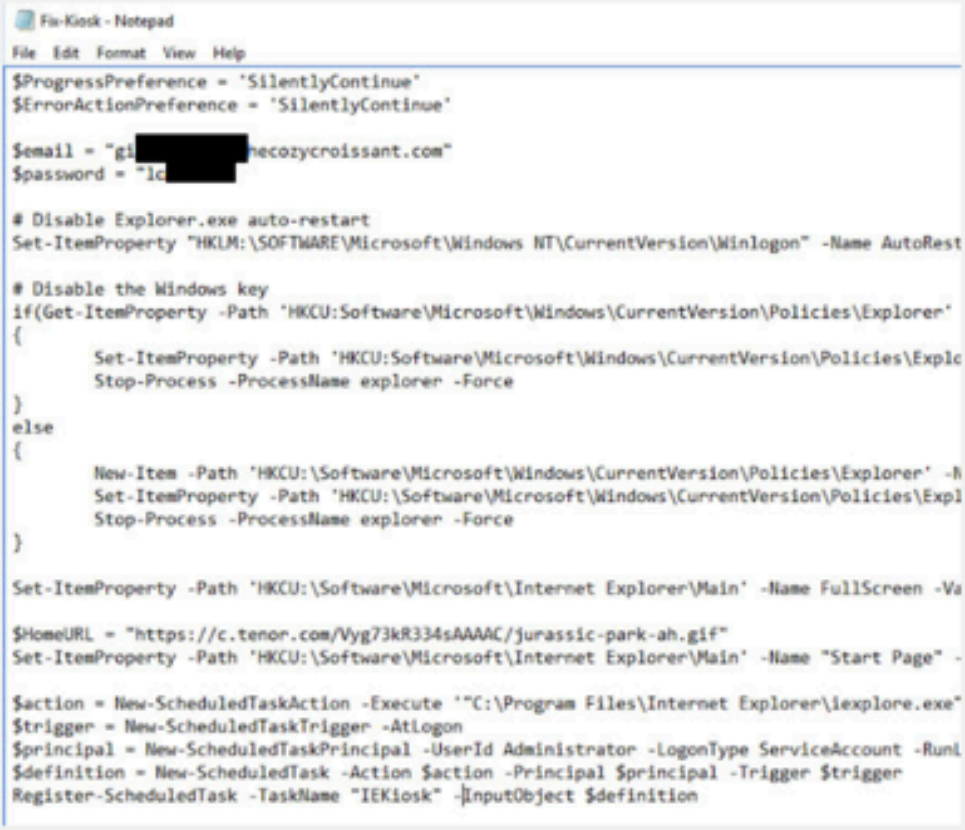
Details:
A patch file for a kiosk error was found on the desktop of one of the guest kiosks. This patch file included an email and password in plaintext for a TCC employee. Any unprivileged user on the machine could see the employee's credentials, which were valid for other machines and a separate domain.

Impact:
The guest kiosks do not require authentication for the administrator account, meaning an unauthenticated actor could get a set of valid domain credentials almost instantly. However, even if the kiosks did require authentication, an arbitrary read vulnerability or any unprivileged but authenticated user could escalate their privileges to that of this TCC employee.

Remediation:
Having a username and password in plaintext is a security vulnerability by itself, and they should be removed. However, if this is not possible, the credentials should be encrypted or placed into a file that's only readable by local administrators.

Steps to Reproduce:
In the local file system, navigate to `C:\Users\Public\Public Desktop\`, and you'll find `Fix-Kiosk.ps1` in that directory.

Relevant Screenshots


Fix-Kiosk - Notepad

File Edit Format View Help

```
$ProgressPreference = 'SilentlyContinue'
$ErrorActionPreference = 'SilentlyContinue'

$email = "gi[redacted]hecozycroissant.com"
$password = "lc[redacted]

# Disable Explorer.exe auto-restart
Set-ItemProperty "HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon" -Name AutoRest

# Disable the Windows key
if(Get-ItemProperty -Path 'HKCU:Software\Microsoft\Windows\CurrentVersion\Policies\Explorer'
{
        Set-ItemProperty -Path 'HKCU:Software\Microsoft\Windows\CurrentVersion\Policies\Explo
        Stop-Process -ProcessName explorer -Force
}
else
{
        New-Item -Path 'HKCU:\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer' -F
        Set-ItemProperty -Path 'HKCU:\Software\Microsoft\Windows\CurrentVersion\Policies\Expl
        Stop-Process -ProcessName explorer -Force
}

Set-ItemProperty -Path 'HKCU:\Software\Microsoft\Internet Explorer\Main' -Name FullScreen -Va

$HomeURL = "https://c.tenor.com/Vyg73kR334sAAAAC/jurassic-park-ah.gif"
Set-ItemProperty -Path 'HKCU:\Software\Microsoft\Internet Explorer\Main' -Name "Start Page" -

$action = New-ScheduledTaskAction -Execute '"C:\Program Files\Internet Explorer\iexplore.exe"
$trigger = New-ScheduledTaskTrigger -AtLogon
$principal = New-ScheduledTaskPrincipal -UserId Administrator -LogonType ServiceAccount -RunL
$definition = New-ScheduledTask -Action $action -Principal $principal -Trigger $trigger
Register-ScheduledTask -TaskName "IEKiosk" -InputObject $definition
```

The kiosk script that includes plaintext credentials

Outdated Linux Kernel

Severity	Vector String	CVSS Score
High	CVSS:3.0/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	**7.8**
Risk Rating	**3**	Likelihood: **Moderate**
Affected Scope	10.0.0.210	

Details:
An Ubuntu machine had an outdated kernel version. This flaw allows a local user with lower privileges to crash the system and escalate privileges.

Impact:
This issue has a very high impact on the confidentiality and integrity of systems. A local user with limited permissions has the ability to escalate to full access. This means that if an attacker is able to login with lower access, they can quickly escalate to root and access sensitive information.

Remediation:
This issue can be solved by simply running a few updates. The kernel can be updated by running these two commands:

```
sudo apt-get update
sudo apt-get dist-upgrade
```

Steps for Reproduction:
Go step by step on how to confirm this vulnerability

Relevant Screenshots

```
g.whatson@corp.cc.local@payment-db:/tmp$ su user
Password:
# id
uid=0(user) gid=0(root) groups=0(root)
#
g.whatson@corp.cc.local@payment-db:/tmp$ head -n4 /etc/passwd
user:$1$user$[redacted]:0:0:/root/root:/bin/bash
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
g.whatson@corp.cc.local@payment-db:/tmp$ uname -r
5.4.0-113-generic
g.whatson@corp.cc.local@payment-db:/tmp$ cat /etc/os-release
NAME="Ubuntu"
VERSION="20.04.5 LTS (Focal Fossa)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 20.04.5 LTS"
VERSION_ID="20.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=focal
UBUNTU_CODENAME=focal
g.whatson@corp.cc.local@payment-db:/tmp$
```

The linux version is 5.4.0-113-generic, which is old enough that it's vulnerable to CVE-2022-2588, a privilege escalation vulnerability.

Weak MySQL Root Password on HMS

Severity	Vector String	CVSS Score
High	CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H	**7.7**
Risk Rating	**4**	Likelihood: **HIGH**
Affected Scope	10.0.0.11	

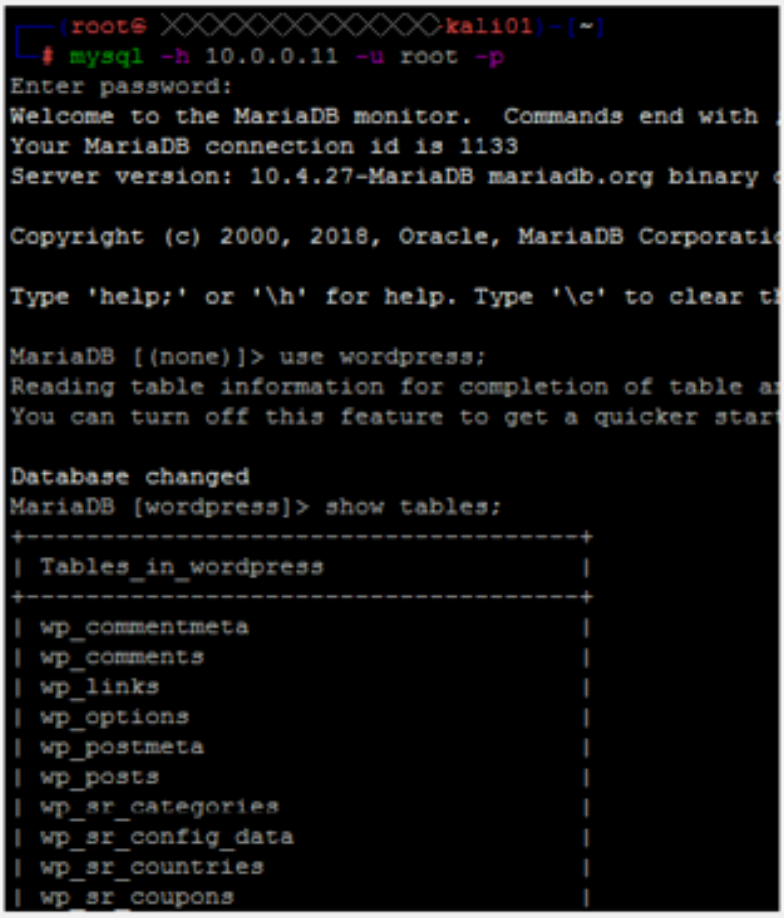
Details:
The root password to the MySQL root account on 10.0.0.11 was easily brute forced. The weak password was used for this root account and allows for attackers to brute force it, thus easily gaining administrative access to the MySQL databases on the Hotel Management System.

Impact:
If an attacker gains root access to the database, they will be able to view and modify the data in the database, thus compromising the integrity and confidentiality of the data stored there.

Remediation:
This can be remediated by changing the root password to be longer and more complex. The password should be something that will not be easily brute forced or guessed.

Steps for Reproduction:
1. Use the rockyou.txt password list to brute force the password using a tool such as Hydra, with the username set to root
2. Soon enough, the correct password will display, allowing you full access to the database.

Relevant Screenshots



A weak root password allows an attacker to brute force and gain access to the entire Wordpress MySQL database instance.

Insecure Loyalty Point Mechanism

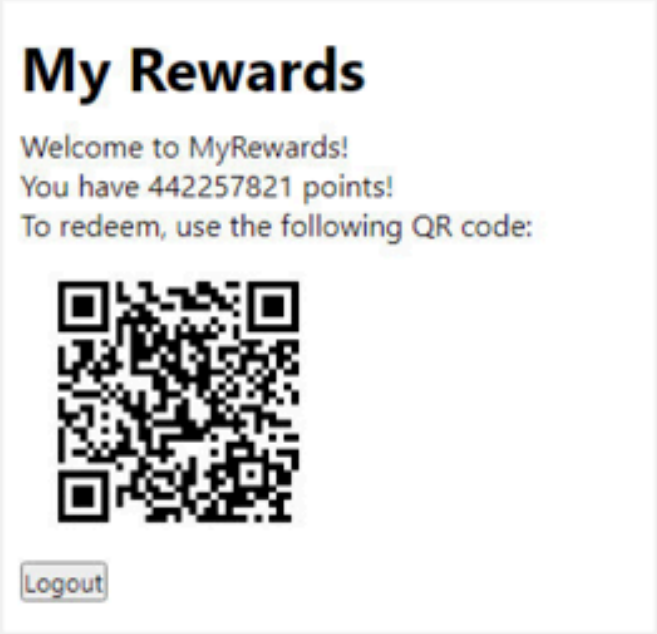
Severity	Vector String	CVSS Score
High	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:H/A:N	**7.5**
Risk Rating	**4**	Likelihood: **HIGH**
Affected Scope	10.0.0.12	

Details:
The loyalty program for TCC includes a QR code that is likely scanned at the front desk to redeem points for a form of discount. When the QR code is read, the encoded data is `username+rewardpoints`.

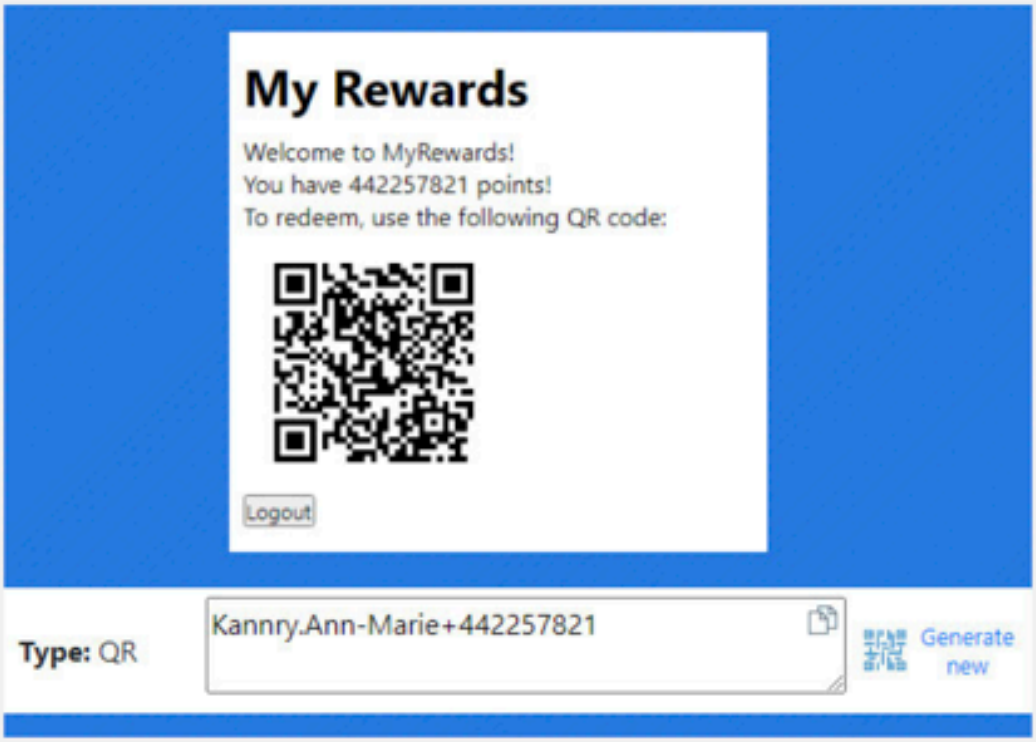
Impact:
This QR code data uses plaintext and does not use double encryption when being scanned by TCC. This means that a customer can create their own QR code with a payload that gives them a large number of reward points. They can then show this QR code when redeeming points and gain unearned discounts. Note that we did not find a system that would scan and use the loyalty points provided, however based on what we found we can reasonably assume how the system works.

Remediation:
This can be remediated by separating the rewards amount from the QR code and storing the rewards on a TCC server. Then, the QR code will simply validate the user's identity and retrieve their rewards from a server.

Relevant Screenshots



When logging into the Rewards portal, a QR code is used to redeem loyalty points.



A QR code reader showing the decoded payload.

Payment Portal SQL Injection

Severity	Vector String	CVSS Score
High	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	**7.5**
Risk Rating	**3**	Likelihood: **MODERATE**
Affected Scope	10.0.0.200	

Details:
The data passed to the endpoint is placed directly into a python f-string, without going through any sort of validation. This allows a malicious actor to pass in SQL queries into the code arbitrarily, allowing for SQL injection.

Impact:
This would allow an attacker to dump the contents of the database, place files, and potentially execute arbitrary commands on the vulnerable machine. This could lead to legal troubles if sensitive data is being stored, or to an attack as large as ransomware.

Remediation:
Instead of using python f-strings to take input from the user directly, sanitization of the input should take place first, so that SQL code cannot be interpreted as SQL code when being passed in.

Steps for Reproduction:
Navigate to http://10.0.0.200:8000/doc/#/admin/rooms/ in a browser and put some standard SQL injection payloads into the URL after the final slash. You can also go to the source code for that endpoint and see the direct use of the `room_id` variable in a SQL statement.

Relevant Screenshots

```python
@api.route('/admin/rooms/<string:room_id>', methods=['GET'])
def return_room_details(room_id):
    log_message = f"GET /admin/rooms/{room_id}; "
    room_details_query = f"SELECT * FROM wp_sr_room_types where id = {room_id}"
    log_message += f"Query: {room_details_query}; "
    try:
        res = query_db_hotel(room_details_query, 'select')
        log_message += f"Status: 200; Message: Information for {room_id} queried; Data: {jsonify(res)}"
        AppLog.info(log_message)
        return jsonify(res)
    except Exception as error:
        log_message += f"Status: 500; Message: {str(error)}"
        AppLog.error(log_message)
        return jsonify({"error":str(error)}), 500
```

The value for `room_id` is piped directly into the SQL query, allowing arbitrary SQL commands to be run without authentication.

Rewards Server Source Code Available

Severity	Vector String	CVSS Score
Medium	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:L/I:L/A:L	**7.4**
Risk Rating	**3**	Likelihood: **MODERATE**
Affected Scope	10.0.0.12	

Details:
While brute-forcing endpoints on the Rewards web server located at http://10.0.0.12, the `/query` endpoint returned with a 200 Status Code. Opening the page revealed a Python script that would be run by PHP on the backend of the server after authentication. This server would make the actual changes to the database for rewards accounts. A set of admin and guest database credentials were also hard-coded into the script.

Impact:
The attacker switches from a black-box test (where no source code/underlying system knowledge is known) to a white-box test (where source code is available). For instance, we never would have known that Python code was run under the hood if this file wasn't made available to us. In addition, it allows attackers to find and exploit mistakes in the code that would not be likely to be found through brute force.

Remediation:
Move the `query` Python script to a non-publicly available directory inside of the machine.

Steps to Reproduce:
Open https://10.0.0.12/query in an incognito tab of your browser.

Relevant Screenshots

A section of the `query` *script from the web browser showing hard-coded, privileged credentials*

Null Kiosk Administrator Credentials

Severity	Vector String	CVSS Score
High	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:L	**7.3**
Risk Rating	**4**	Likelihood: **HIGH**
Affected Scope	10.0.200.101, 10.0.200.102, 10.0.200.103, 10.0.200.104	

Details:
The password for the built-in Administrator account is null, meaning no password is required. Basic active reconnaissance with Nmap reveals that SMB is open on port 445 and RDP is open on port 3389. When using RDP, an HTML page is displayed full screen with the Windows key disabled, preventing users from accessing other applications. However, since remote access is enabled, commands may be sent through psexec remotely.

Impact:
This vulnerability provided both shell access and persistent credentials that were consistent across all four hosts. If desired, those hosts could have been reset, wiped, or otherwise modified. These four computers, as well as any critical services they were running, could have easily been taken down or used to pivot to other machines.

Remediation:
Ensure that user accounts, including and especially administrator accounts have unique, strong passwords. In addition, only allow remote access to users that require it for business purposes. Lastly, only keep ports like WinRM or RDP open if necessary. Otherwise, configure to close this port and any others that are not critical to business operations.

Steps for Reproduction:
Open a Metasploit console on a host with network access to the vulnerable host. Type `use auxiliary/scanner/winrm/winrm_login` into the Metasploit console, followed by `options` to see the options necessary to run this command. Set remote host IP address with `set RHOSTS <IP>`, and specify wordlists for username and password brute forcing with `set PASS_FILE /path/to/file` & `set USERPASS_FILE /path/to/file`. Make sure to allow for blank passwords by typing `set BLANK_PASSWORDS true`. Execute the exploit with the command `exploit`.

This demonstrates that an Administrator with no password is valid. These credentials can then be used in Windows RDP or WinRM to access the machine.

Relevant Screenshots

```
msf6 auxiliary(scanner/winrm/winrm_login) > options

Module options (auxiliary/scanner/winrm/winrm_login):

   Name              Current Setting                                     Required  Description
   ----              ---------------                                     --------  -----------
   BLANK_PASSWORDS   false                                               no        Try blank passwords for all users
   BRUTEFORCE_SPEED  5                                                   yes       How fast to bruteforce, from 0 to 5
   DB_ALL_CREDS      false                                               no        Try each user/password couple stored in the current da
   DB_ALL_PASS       false                                               no        Add all passwords in the current database to the list
   DB_ALL_USERS      false                                               no        Add all users in the current database to the list
   DB_SKIP_EXISTING  none                                                no        Skip existing credentials stored in the current databa
   DOMAIN            WORKSTATION                                         yes       The domain to use for Windows authentification
   PASSWORD                                                              no        A specific password to authenticate with
   PASS_FILE         /usr/share/wordlists/metasploit/password.lst        no        File containing passwords, one per line
   Proxies                                                               no        A proxy chain of format type:host:port[,type:host:port
   RHOSTS            10.0.200.104                                        yes       The target host(s), see https://github.com/rapid7/meta
                                                                                   t
   RPORT             5985                                                yes       The target port (TCP)
   SSL               false                                               no        Negotiate SSL/TLS for outgoing connections
   STOP_ON_SUCCESS   false                                               yes       Stop guessing when a credential works for a host
   THREADS           1                                                   yes       The number of concurrent threads (max one per host)
   URI               /usman                                              yes       The URI of the WinRM service
   USERNAME                                                              no        A specific username to authenticate as
   USERPASS_FILE     /usr/share/wordlists/metasploit/unix_users.txt      no        File containing users and passwords separated by spac
   USER_AS_PASS      false                                               no        Try the username as the password for all users
   USER_FILE                                                             no        File containing usernames, one per line
   VERBOSE           true                                                yes       Whether to print output for all attempts
   VHOST                                                                 no        HTTP server virtual host
```

Metasploit Console options for WINRM Exploit

```
RHOSTS => 10.0.200.104
msf6 auxiliary(scanner/winrm/winrm_login) > exploit

[-] 10.0.200.104: - LOGIN FAILED: WORKSTATION\: (Incorrect: )
[-] 10.0.200.104: - LOGIN FAILED: WORKSTATION\4Dgifts: (Incorrect: )
[-] 10.0.200.104: - LOGIN FAILED: WORKSTATION\abrt: (Incorrect: )
[-] 10.0.200.104: - LOGIN FAILED: WORKSTATION\adm: (Incorrect: )
[-] 10.0.200.104: - LOGIN FAILED: WORKSTATION\admin: (Incorrect: )
[+] 10.0.200.104:5985 - Login Successful: WORKSTATION\administrator:
[*] Error: 10.0.200.104: WinRM::WinRMHTTPTransportError No response ().
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Metasploit Exploit in action successfully authenticating to the WINRM service on 10.0.200.104:5985

Relevant Screenshots cont'd

```
msf6 auxiliary(scanner/winrm/winrm_login) > exploit

[-] 10.0.200.101: - LOGIN FAILED: WORKSTATION\: (Incorrect: )
[-] 10.0.200.101: - LOGIN FAILED: WORKSTATION\4Dgifts: (Incorrect: )
[-] 10.0.200.101: - LOGIN FAILED: WORKSTATION\abrt: (Incorrect: )
[-] 10.0.200.101: - LOGIN FAILED: WORKSTATION\adm: (Incorrect: )
[-] 10.0.200.101: - LOGIN FAILED: WORKSTATION\admin: (Incorrect: )
[+] 10.0.200.101:5985 - Login Successful: WORKSTATION\administrator:
[*] Error: 10.0.200.101: WinRM::WinRMHTTPTransportError No response ().
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/winrm/winrm_login) > set RHOSTS 10.0.200.102
RHOSTS => 10.0.200.102
msf6 auxiliary(scanner/winrm/winrm_login) > exploit

[-] 10.0.200.102: - LOGIN FAILED: WORKSTATION\: (Incorrect: )
[-] 10.0.200.102: - LOGIN FAILED: WORKSTATION\4Dgifts: (Incorrect: )
[-] 10.0.200.102: - LOGIN FAILED: WORKSTATION\abrt: (Incorrect: )
[-] 10.0.200.102: - LOGIN FAILED: WORKSTATION\adm: (Incorrect: )
[-] 10.0.200.102: - LOGIN FAILED: WORKSTATION\admin: (Incorrect: )
[+] 10.0.200.102:5985 - Login Successful: WORKSTATION\administrator:
[*] Error: 10.0.200.102: WinRM::WinRMHTTPTransportError No response ().
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/winrm/winrm_login) > set RHOSTS 10.0.200.103
RHOSTS => 10.0.200.103
msf6 auxiliary(scanner/winrm/winrm_login) > exploit

[-] 10.0.200.103: - LOGIN FAILED: WORKSTATION\: (Incorrect: )
[-] 10.0.200.103: - LOGIN FAILED: WORKSTATION\4Dgifts: (Incorrect: )
[-] 10.0.200.103: - LOGIN FAILED: WORKSTATION\abrt: (Incorrect: )
[-] 10.0.200.103: - LOGIN FAILED: WORKSTATION\adm: (Incorrect: )
[-] 10.0.200.103: - LOGIN FAILED: WORKSTATION\admin: (Incorrect: )
[+] 10.0.200.103:5985 - Login Successful: WORKSTATION\administrator:
[*] Error: 10.0.200.103: WinRM::WinRMHTTPTransportError No response ().
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Successful exploit of WINRM on hosts: 10.0.200.101-3

Information Disclosure in Payment Database

Severity	Vector String	CVSS Score
High	CVSS:3.0/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N	**7.1**
Risk Rating	**3**	Likelihood: **MODERATE**
Affected Scope	10.0.0.210	

Details:
All credit card information is stored in the payment database, including name, credit card number, expiration date, cvv, and zip code.

Impact:
Storing cvv values of users is a very serious violation of Payment Card Industry Data Security Standards (PCI-DSS). PCI compliance fines can vary from $5,000 to $100,000 USD per month depending on the size of the company and scope of the non-compliance.

Remediation:
All customer cvv values should be purged from the database in order for TCC to be in compliance with PCI-DSS. In addition, we recommend reviewing all PCI-DSS requirements to ensure that TCC can remain compliant and avoid any possible fines.

Relevant Screenshots

```
payment=# select * from billing.credit_cards where name like '%Carol%';
 id |        name        |       number        | expiration | ccv |  zip
----+--------------------+---------------------+------------+-----+-------
  2 | Carol              | 3                   | 0          |     | 7
  1 | Carol              | 2                   | 1          |     | 0
  9 | Carol              | 5                   | 0          |     | 8
  0 | Carol              | 3                   | 0          |     | 4
  3 | Carol              | 6                   | 0          |     | 1
  7 | Carol              | 4                   | 0          |     | 6
  7 | Carol              | 4                   | 0          |     | 0
  8 | Carol              | 4                   | 1          |     | 3
  2 | Carol              | 3                   | 1          |     | 8
  3 | Carol              | 4                   | 0          |     | 4
  2 | Carol              | 4                   | 0          |     | 1
  7 | Carol              | 4                   | 0          |     | 7
```

Payment information stored in the `credit_cards` table includes full card numbers, expiration dates, CCVs, and zip codes.

Weak Credentials for OpenLDAP Admin

Severity	Vector String	CVSS Score
Medium	CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H	**6.8**
Risk Rating	**2**	Likelihood: **MODERATE**
Affected Scope	10.0.0.100	

Details:
The 10.0.0.100 host runs the OpenLDAP service and has weak admin credentials that are easily guessed. These weak credentials allow attackers to easily gain administrative access to the OpenLDAP service, which stores sensitive information pertaining to users and groups on the domain.

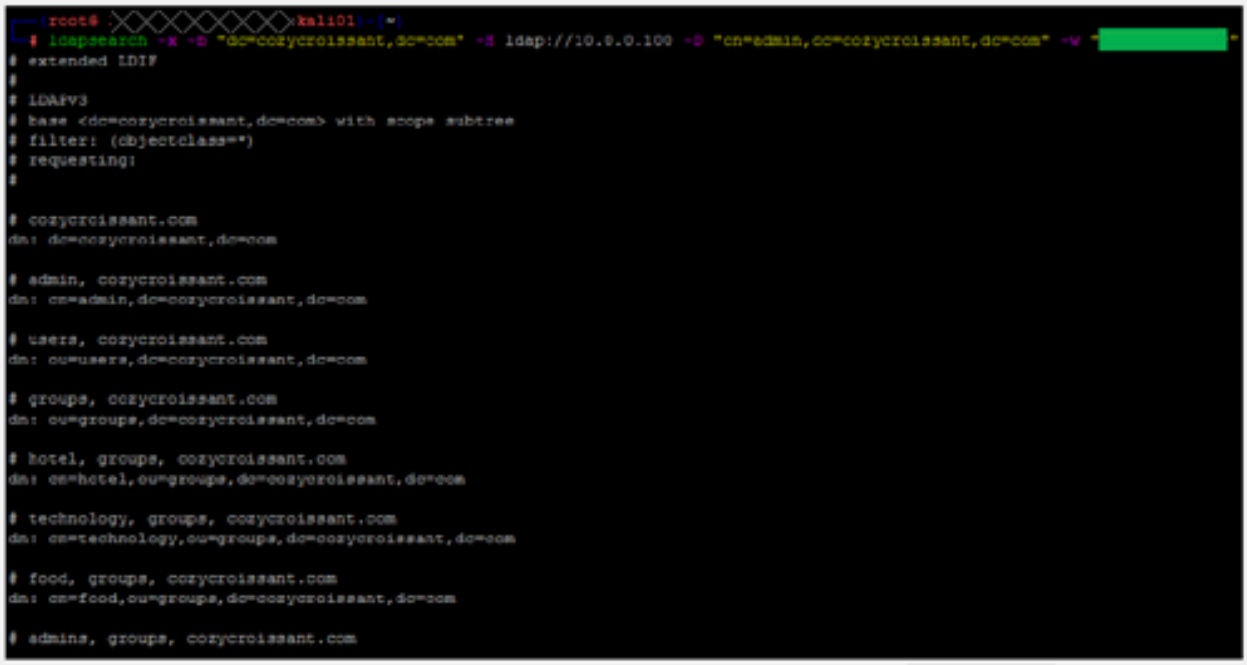
Impact:
This could potentially compromise the confidentiality and integrity of TCC's customer information.

Remediation:
This can be remediated by changing the username of the account and changing the password to be longer and more complex, making it significantly difficult to guess the password.

Steps for Reproduction:
Go step by step on how to confirm this vulnerability

- Run the ldapsearch command on ldap://10.0.0.100
- Use the username admin and the guessed password as credentials in the ldapsearch command
- View the information provided from the search

Relevant Screenshots



Using weak OpenLDAP admin credentials to obtain information with `ldapsearch` command

Passwords Stored in Plaintext

Severity	Vector String	CVSS Score
Moderate	CVSS:3.1/AV:N/AC:L/PR:H/UI:N/S:C/C:H/I:H/A:N	**8.7**
Risk Rating	**4**	Likelihood: **HIGH**
Affected Scope	10.0.0.12	

Details:
Users credentials stored in the MySQL database are stored in plaintext. If an attacker gained access to the MySQL database, they would then be able to view all users passwords in the database, thus having the ability to access their accounts.

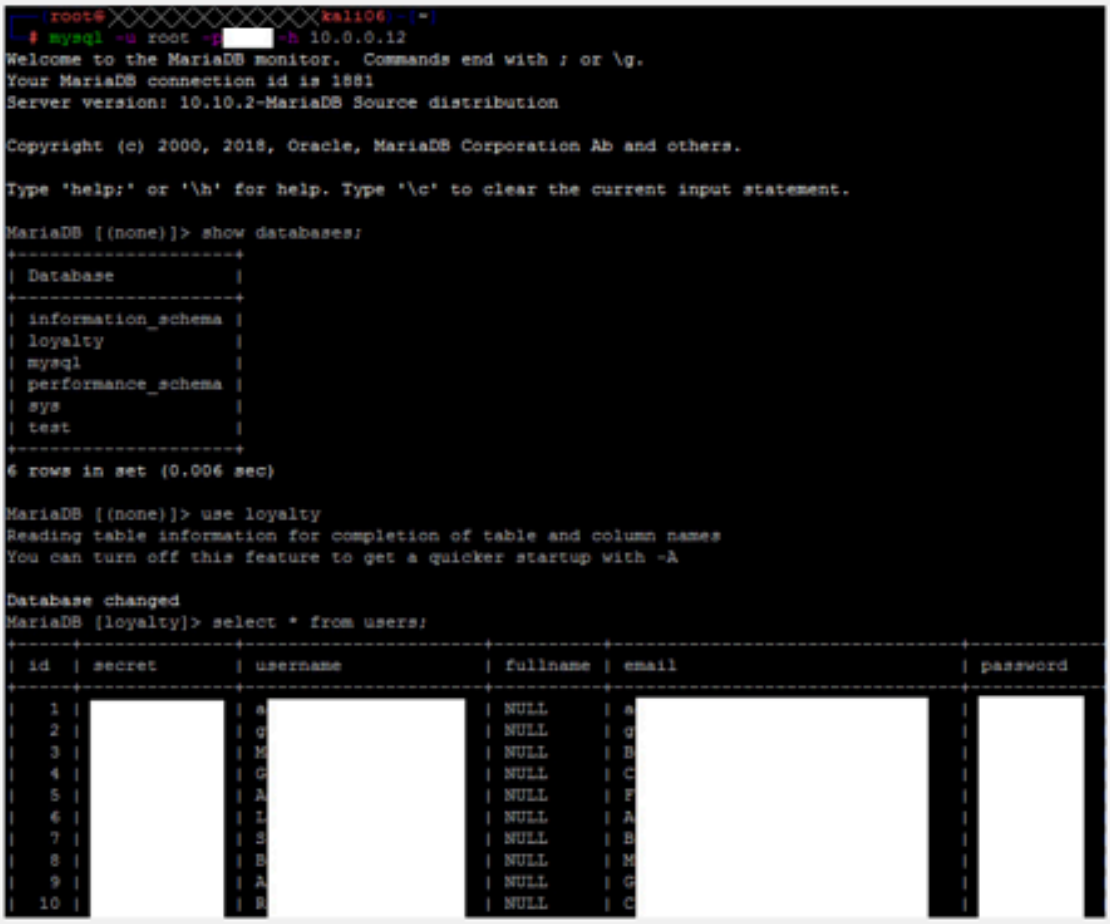
Impact:
Storing user credentials in plaintext greatly increases the chances of their accounts being compromised. In the event of a breach on this database, the attackers would have the user's passwords in plaintext, rather than only having the password hashes.

Remediation:
This can be mitigated by configuring your MySQL database to store all user passwords with a secure hashing algorithm, such as SHA-512. This will ensure that even if information in the database is leaked or an attacker has root credentials to MySQL, the user's passwords are not shown in plaintext.

Steps for Reproduction:

1. Root credentials to the MySQL database are needed to reproduce this attack
2. Access the MySQL database by running `mysql -u root -p <password> -h 10.0.0.12`
3. After logging into the database, run `show databases` to view all databases
4. Access the loyalty database by running `use loyalty`
5. To view the user table and the plaintext passwords stored in it, run the SQL query `select * from users;`

Relevant Screenshots



```
┌──(root㉿[redacted])-[~]
└─# mysql -u root -p[redacted] -h 10.0.0.12
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 1881
Server version: 10.10.2-MariaDB Source distribution

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)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| loyalty            |
| mysql              |
| performance_schema |
| sys                |
| test               |
+--------------------+
6 rows in set (0.006 sec)

MariaDB [(none)]> use loyalty
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [loyalty]> select * from users;
+-----+--------------+--------------------+----------+--------------------------------+--------------+
| id  | secret       | username           | fullname | email                          | password     |
+-----+--------------+--------------------+----------+--------------------------------+--------------+
|   1 | [redacted]   | a[redacted]        | NULL     | a[redacted]                    | [redacted]   |
|   2 | [redacted]   | g[redacted]        | NULL     | g[redacted]                    | [redacted]   |
|   3 | [redacted]   | M[redacted]        | NULL     | B[redacted]                    | [redacted]   |
|   4 | [redacted]   | G[redacted]        | NULL     | C[redacted]                    | [redacted]   |
|   5 | [redacted]   | A[redacted]        | NULL     | F[redacted]                    | [redacted]   |
|   6 | [redacted]   | I[redacted]        | NULL     | A[redacted]                    | [redacted]   |
|   7 | [redacted]   | S[redacted]        | NULL     | B[redacted]                    | [redacted]   |
|   8 | [redacted]   | B[redacted]        | NULL     | M[redacted]                    | [redacted]   |
|   9 | [redacted]   | A[redacted]        | NULL     | G[redacted]                    | [redacted]   |
|  10 | [redacted]   | R[redacted]        | NULL     | C[redacted]                    | [redacted]   |
```

The password field in the users table does not contain hashes, but rather plaintext passwords.

Jellyfin Unauthorized Logon

Severity	Vector String	CVSS Score
Medium	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:L	**6.5**
Risk Rating	**3**	Likelihood: **HIGH**
Affected Scope	10.0.0.20	

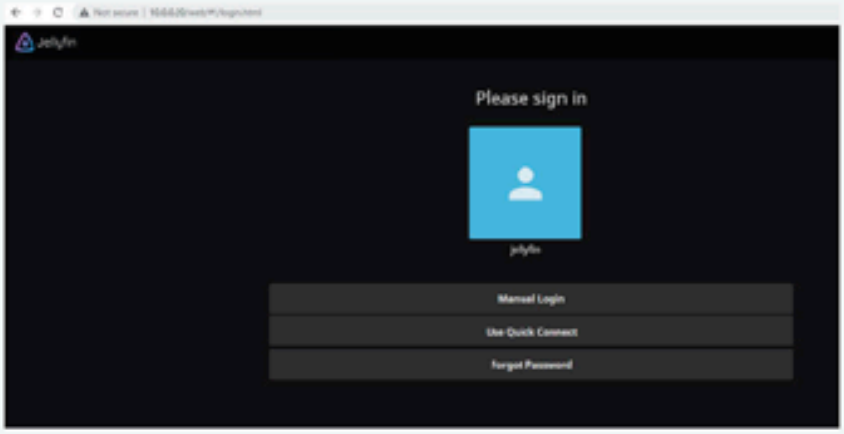
Details:
When accessing the Jellyfin media sharing service, our team found that customers can access the setup dashboard without authentication. This dashboard contains many configurations and settings including where media is mounted, access logs, and user information.

Impact:
If a malicious or careless user were to access this dashboard, they would be able to delete all media that is currently mounted to the system. This means that customers wanting to stream entertainment through Jellyfish would have no content to enjoy. In addition, this vulnerability can be used by an attacker to enumerate part of the media server. For example, by mounting the `/home` directory as a media share, one can see which users are on the system by looking at the folders that show up. This gives the attacker more knowledge to later brute force a login.

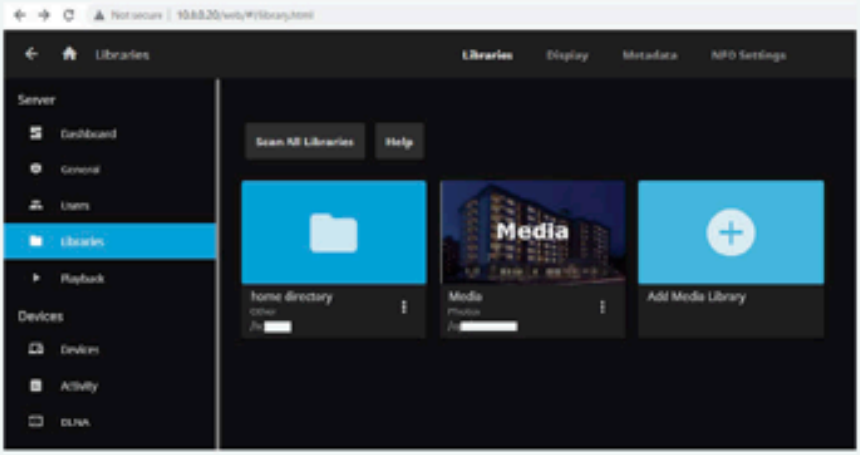
Remediation:
This can be mitigated by creating one Jellyfin administrator account with a strong password and giving all other Jellyfin users only streaming privileges.

Steps for Reproduction:
When accessing Jellyfin, click on the Jellyfin user icon to bring you into the entertainment console without authentication. From here, you can navigate to the dashboard and then the libraries page. In libraries, you are able to add a new folder such as the `/home` directory which will allow you to see the names of users on the machine hosting Jellyfin.

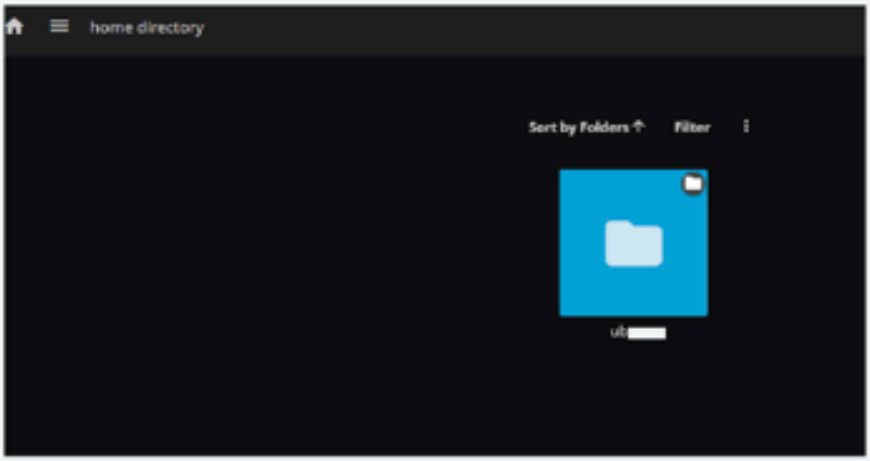
Relevant Screenshots



Login page for Jellyfin



The dashboard of Jellyfin showing libraries



An example of user enumeration by mounting the home directory

Lack of Access Control in Rewards Portal

Severity	Vector String	CVSS Score
Medium	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:L	**6.3**
Risk Rating	**2**	Likelihood: **MODERATE**
Affected Scope	10.0.0.12	

Details:
Any user in the customer database is being elevated to admin for that database, except for the Guest user, thus potentially giving them access to more in the database than they should have.

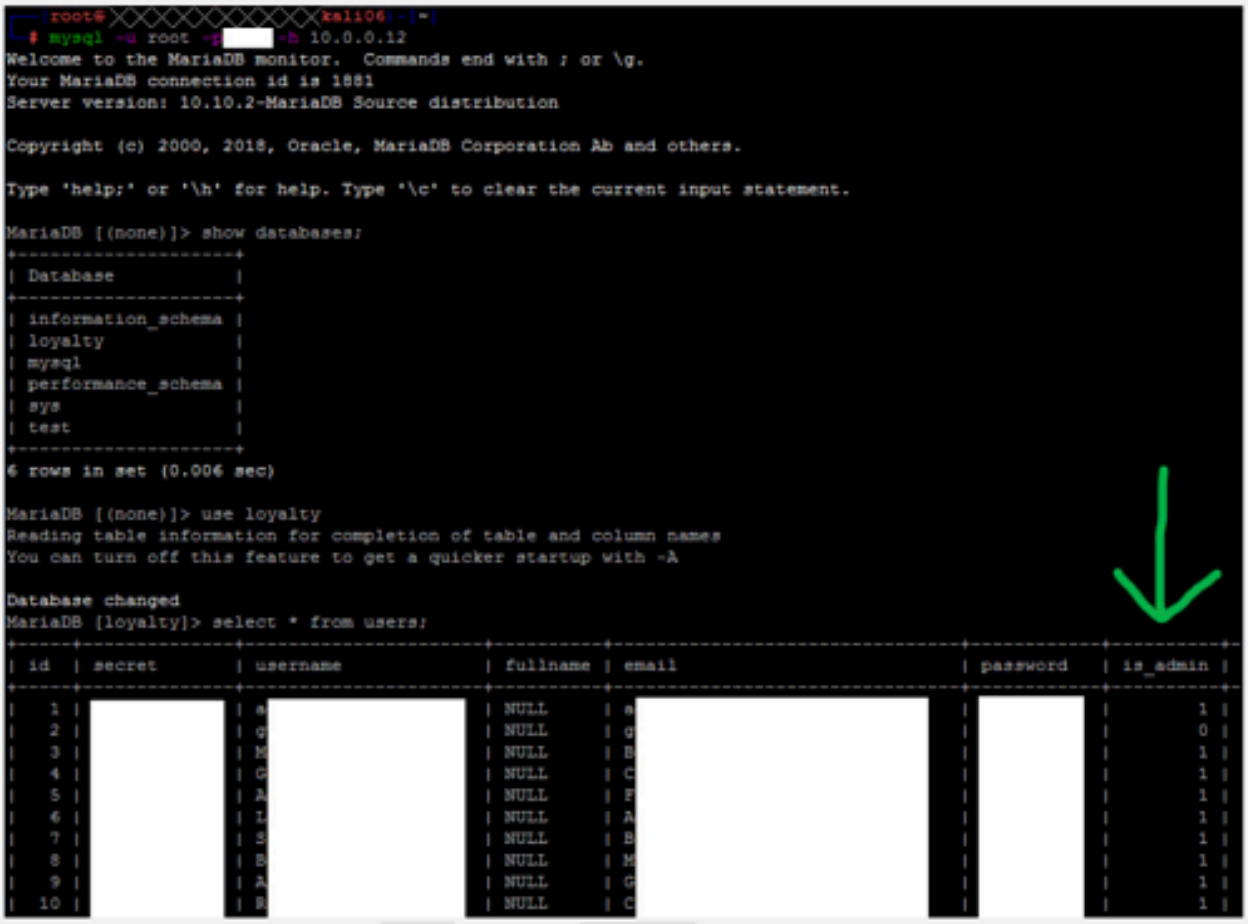
Impact:
Giving admin credentials to users who do not need those privileges is inviting misuse of that access. For example, this may allow the user to have increased access to information they shouldn't have, or access to functions that they shouldn't be performing.

Remediation:
Change the default value of a new customer to not be an admin, following the principle of least privilege.

Steps for Reproduction:

1. Root credentials to the MySQL database are needed to reproduce this attack
2. Access the MySQL database by running `mysql -u root -p <password> -h 10.0.0.12`
3. After logging into the database, run `show databases` to view all databases
4. Access the loyalty database by running `use loyalty`
5. To view the user table and `is_admin` field, run the SQL query `select * from users;`

Relevant Screenshots



```
┌──(root㉿[redacted]kali106)-[~]
└─# mysql -u root -p[redacted] -h 10.0.0.12
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 1881
Server version: 10.10.2-MariaDB Source distribution

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)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| loyalty            |
| mysql              |
| performance_schema |
| sys                |
| test               |
+--------------------+
6 rows in set (0.006 sec)

MariaDB [(none)]> use loyalty
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [loyalty]> select * from users;
+-----+--------------+----------------------+----------+-----------------------------------+--------------+----------+
| id  | secret       | username             | fullname | email                             | password     | is_admin |
+-----+--------------+----------------------+----------+-----------------------------------+--------------+----------+
|   1 | [redacted]   | a[redacted]          | NULL     | a[redacted]                       | [redacted]   |        1 |
|   2 | [redacted]   | g[redacted]          | NULL     | g[redacted]                       | [redacted]   |        0 |
|   3 | [redacted]   | M[redacted]          | NULL     | B[redacted]                       | [redacted]   |        1 |
|   4 | [redacted]   | G[redacted]          | NULL     | C[redacted]                       | [redacted]   |        1 |
|   5 | [redacted]   | A[redacted]          | NULL     | F[redacted]                       | [redacted]   |        1 |
|   6 | [redacted]   | L[redacted]          | NULL     | A[redacted]                       | [redacted]   |        1 |
|   7 | [redacted]   | S[redacted]          | NULL     | B[redacted]                       | [redacted]   |        1 |
|   8 | [redacted]   | B[redacted]          | NULL     | M[redacted]                       | [redacted]   |        1 |
|   9 | [redacted]   | A[redacted]          | NULL     | G[redacted]                       | [redacted]   |        1 |
|  10 | [redacted]   | B[redacted]          | NULL     | C[redacted]                       | [redacted]   |        1 |
```

Every single user in the `users` table has `is_admin` set to 1, except for the guest account.

World-Readable Sensitive File

Severity	Vector String	CVSS Score
Medium	CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H	6.3
Risk Rating	**3**	Likelihood: **LOW**
Affected Scope	10.0.0.7, 10.0.0.210	

Details:
On both of these servers, environmental variables for database servers are being stored in plain text with the .env files readable by any unprivileged users.

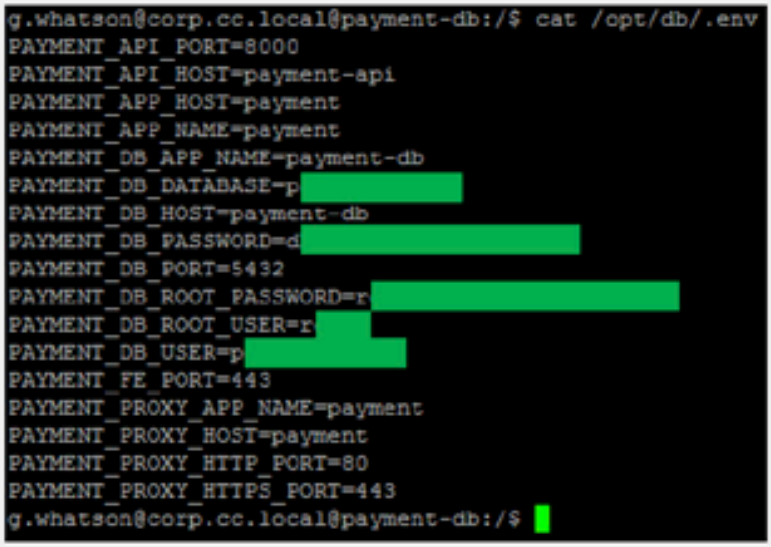
Impact:
An attacker could gain access to these files and learn the credentials to the corresponding databases. With those credentials, the attacker could log onto the databases, view any data stored in them, and edit information. This would affect the confidentiality and integrity of the data in those databases. In addition, secrets such as JSON Web Token secrets are exposed, allowing an unprivileged user to forge session cookies and impersonate other users.

Remediation:
Since Docker commands cannot be run by unprivileged users and the container code is private, the files placed into the container should also be kept private. The entire /opt/app or /opt/db folders should be changed to only be read by sudo users (`sudo chmod -R 770 /opt/app /opt/db`).

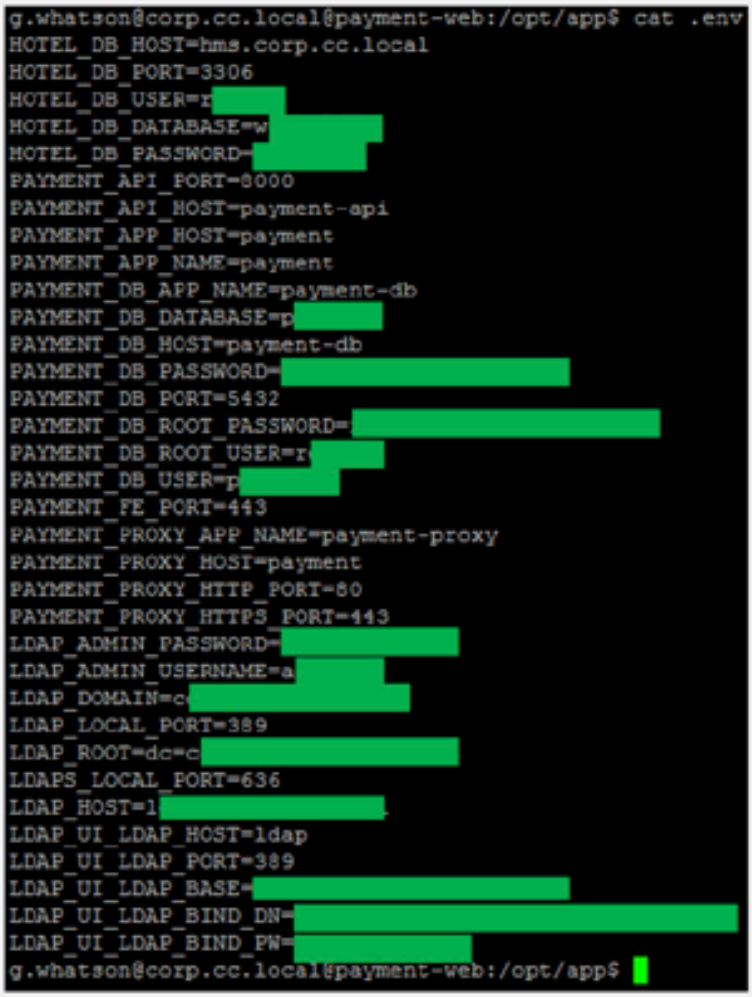
Steps for Reproduction:
1. Log onto the server with an unprivileged user, or switch your account to an unprivileged user with a command like sudo su account.
2. Running the command `cat /opt/app/.env` or `cat /opt/db/.env` will show the environmental variables used for database connection strings and other secrets.

Relevant Screenshots

Environmental variables for DOAPI



Environmental variables for PAYMENT-DB



Environmental variables for PAYMENT-WEB

Passwords Present in Active Directory Description

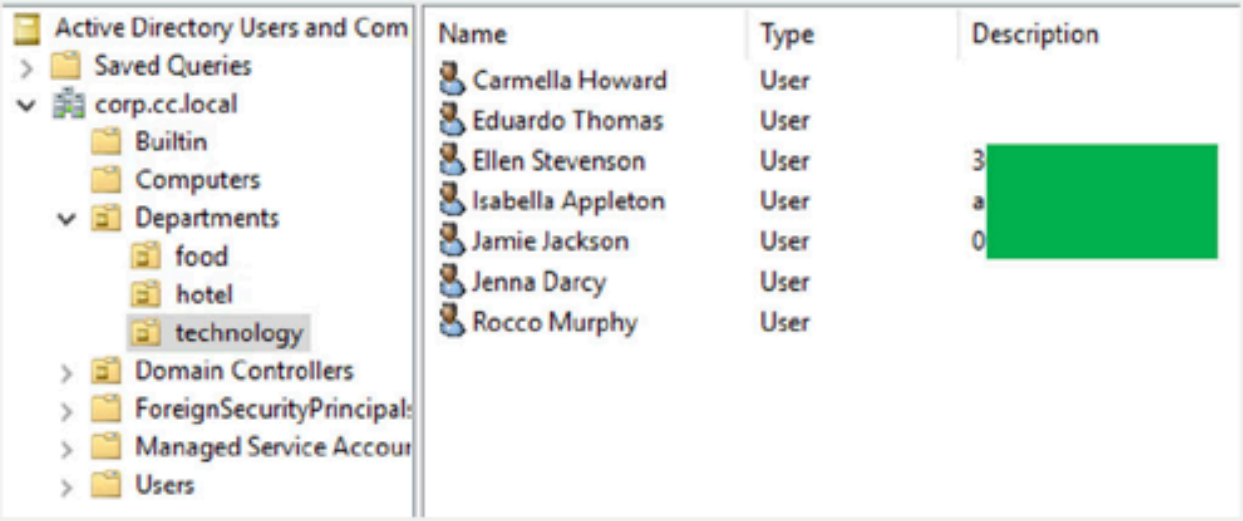
Severity	Vector String	CVSS Score
Medium	CVSS:3.0/AV:L/AC:L/PR:H/UI:N/S:U/C:H/I:H/A:N	**6.0**
Risk Rating	**2**	Likelihood: **MODERATE**
Affected Scope	10.0.0.5, 10.0.0.6, 10.0.0.7, 10.0.0.11, 10.0.0.12, 10.0.0.20, 10.0.0.200, 10.0.0.210	

Details:
When logged into the domain controller as domain admin, a user can view active directory which shows a list of domain users. For some domain users, a description field next to their name included their password in plaintext.

Impact:
If an attacker is able to authenticate as one domain admin, they can then learn the credentials of other users and infiltrate more accounts. In addition, authorized users should not know the credentials of other users.

Remediation:
This can be fixed by first clearing out all of the description sections for users in active directory. Then, domain admins should be trained to not include descriptions when creating new users to avoid this issue in the future.

Relevant Screenshots



Plaintext passwords being stored in the description section of domain users

Remotely-Accessible Wordpress Site

Severity	Vector String	CVSS Score
Critical	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:L/I:N/A:N	**5.8**
Risk Rating	**2**	Likelihood: **MODERATE**
Affected Scope	10.0.0.11	

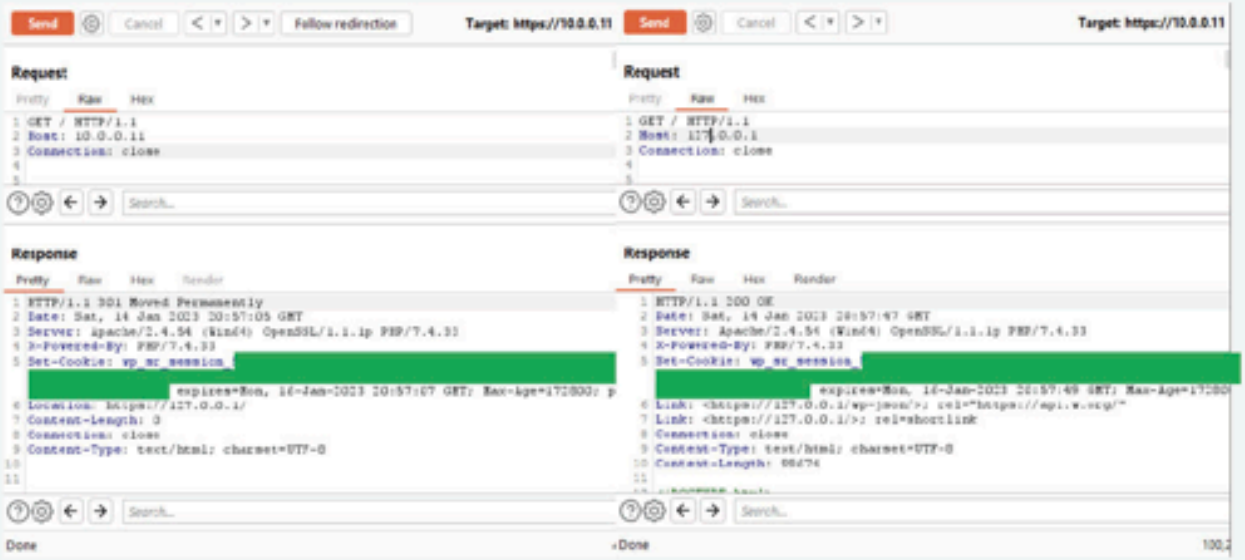
Details:
Using the BurpSuite tool, we were able to access the Wordpress Site remotely. The unmodified GET request to the Wordpress site returned a 301 status code, which means that the site was moved permanently. Modifying the GET request to set its Host header as 127.0.0.1 returned the status code 200 and the content to the page.

Impact:
This allows any users to remotely access this site and the content on it when access to the site should be more restricted.

Remediation:
The Wordpress site should check for what local interface the traffic is coming from instead of a user-controlled HTTP header (like Host or X-Forwarded-For). For instance, only packets from localhost should arrive on the loopback interface.

Steps for Reproduction:
1. Open up Burp Suite and intercept a request heading to http://10.0.0.11/
2. Send this request to Repeater and send - you'll notice no content is returned, and a 301 status code is returned.
3. Modify the host header to 127.0.0.1 (see request on right in screenshot below) and send the request - you'll notice content is returned, allowing access to the Hotel Management System.

Relevant Screenshots



The left request with the Host header set to 10.0.0.11 receives a response with a return code of 301 and no content. The right request with a Host header set to 127.0.0.1 (but still sent to https://10.0.0.11/) receives a response with a 200 OK status code and content.

Lack of Payment Portal Permissions

Severity	Vector String	CVSS Score
Medium	CVSS:3.1/AV:A/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:L	**5.5**
Risk Rating	**2**	Likelihood: **HIGH**
Affected Scope	10.0.0.200	

Details:
The Payment Portal does not check user permissions before allowing users access to information or make changes. This allows any user, guest, admin, or other to do any action.

Impact:
This vulnerability could allow any user to change information in the Payment Portal, for themselves or possibly for other users, due to the lack of permission controls in the source code.

Remediation:
Update the source code for the Payment Portal to check the permission of a user before any requests to change or view information.

Information Disclosure in Rewards Web Server

Severity	Vector String	CVSS Score
Medium	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N	**5.3**
Risk Rating	**3**	Likelihood: **MODERATE**
Affected Scope	10.0.0.12	

Details:
When visiting the Rewards webserver, and inputting credentials to try and gain access, if the credentials are wrong, nothing but an error appears on the page. However, when looking at the response body when using a short string as the username value, it can be seen that the response actually contains information about the first database entry whose email address contains the provided username in some substring within. The database entry contains the email address, name, username, database id number, password, and number of rewards points.

As an example, if the username of 'a' is submitted to the web server, and there was a database entry such as aaaaaaaaa.test@gmail.com, then the response will contain all information for the account associated with that email, even on an error.

Impact:
This has a fairly high impact on the company, as it is fully unauthenticated access to sensitive information. This can lead to legal troubles for the company, and to potential further attacks using the credentials gained.

Remediation:
Ensure that the user input for the username and password is not searching for a substring and returning data even when the full string does not match.

Steps for Reproduction:
Navigate to the web page containing the login page for the Rewards server. Send a string in the username field such as 'as' or 'ge'. Submit the credentials then check the Network tab in the developer's tools for your browser. You should be able to find the Response body for the web page.

Relevant Screenshots

Request

```
GET /userapi.php?login&;type=
user;user=Tin pass=test HTTP/2
Host: 10.0.0.12
Sec-Ch-Ua: "Chromium";v="109", "Not_A
Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0;
 Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/109.0.5414.75
Safari/537.36
Sec-Ch-Ua-Platform: "Windows"
Accept: */*
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://10.0.0.12/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

Response

Pretty Raw Hex Render

```
HTTP/2 200 OK
Server: nginx
Date: Sat, 14 Jan 2023 21:02:32 GMT
Content-Type: application/json;
charset=utf-8
Host: app
X-Powered-By: PHP/7.4.33
Access-Control-Allow-Methods: GET,
POST, OPTIONS
Access-Control-Allow-Headers:
DNT,User-Agent,X-Requested-With,If-Mod
ified-Since,Cache-Control,Content-Type
,Range
Access-Control-Expose-Headers:
Content-Length,Content-Range

{
  "active":true,
  "admin":true,
  "data":[
    {
      "active":true,
      "admin":true,
      "email":
      "Tin[redacted]",
      "id":[redacted],
      "name":null,
      "password":"[redacted]",
      "points":[redacted]
      "secret":"[redacted]",
      "type":"admin",
      "user":"[redacted]",
      "username":"[redacted]"
    }
  ],
  "email":"[redacted]",
  "error":1,
  "error_msg":"invalid password",
  "id":[redacted],
  "name":null,
  "password":"[redacted]",
  "points":[redacted],
  "secret":"[redacted]",
  "type":"admin",
  "user":"[redacted]",
  "username":"[redacted]"
}
```

The first result in the database containing the substring found in the `user` parameter is returned with lots of sensitive information.

Outdated TLS versions 1 and 1.1

Severity	Vector String	CVSS Score
Medium	CVSS:3.1/AV:A/AC:H/PR:H/UI:R/S:U/C:N/I:N/A:N	**5.3**
Risk Rating	**2**	Likelihood: **LOW**
Affected Scope	10.0.0.11, 10.0.0.12, 10.0.0.102, 10.0.0.200	

Details:
TLS is the successor of SSL, a method of initiating secure communication over network connections and is mainly used between a server and a web browser. Researchers have found that versions 1 and 1.1 have weaknesses that allow attackers to listen in on the network traffic.

All of the listed machines support up to TLS version 1.3, but also have TLS 1.0 and 1.1 enabled. This opens the possibility for an attacker to perform a down-grade attack to force the server to communicate over TLSv1.

Impact:
This vulnerability could allow attackers to listen in on conversations between the server and client browser, possibly allowing the attacker to retrieve usernames, passwords, and other user PII.

Remediation:
To remediate this vulnerability, TLSv1 and v1.1 need to be disabled on the web servers in the environment.

Apache Web Servers:
Edit the ssl.conf (httpd-ssl.conf on some installs) file of the Apache installation and change the line starting with SSLProtocol to be the following:

```
SSLProtocol all -SSLv3 -TLSv1 -TLSv1.1
```

The following guide gives more in-depth information about how to disable weak versions of TLS:
https://www.ssl.com/guide/disable-tls-1-0-and-1-1-apache-nginx/

Steps for Reproduction:
Use the following command on a Linux host with nmap installed to check what versions of TLS a server support:

```
nmap -p 443 -script=ssl-enum-ciphers
```

Relevant Screenshots

```
# Nmap 7.93 scan initiated Sat Jan 14 06:21:17 2023 as: nmap -p 443 --script=ssl-enum-ciphers -oN tls12.txt
10.0.0.12
Nmap scan report for 10.0.0.12
Host is up (0.0052s latency).

PORT    STATE SERVICE
443/tcp open  https
| ssl-enum-ciphers:
|   TLSv1.0:
|     ciphers:
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (ecdh_x25519) - A
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (ecdh_x25519) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 4096) - A
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 4096) - A
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 4096) - A
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 4096) - A
|     compressors:
|       NULL
|     cipher preference: server
|     warnings:
|       Key exchange (ecdh_x25519) of lower strength than certificate key
|   TLSv1.1:
|     ciphers:
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (ecdh_x25519) - A
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (ecdh_x25519) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 4096) - A
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 4096) - A
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 4096) - A
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 4096) - A
|     compressors:
```

Snippet of an nmap scan on host 10.0.0.12 enumerating the supported TLS versions. Not pictured is the entry for TLSv3 that is supported on the server

Weak TLS Ciphers

Severity	Vector String	CVSS Score
Medium	CVSS:3.1/AV:A/AC:H/PR:N/UI:N/S:U/C:H/I:N/A:N	**5.3**
Risk Rating	**2**	Likelihood: **LOW**
Affected Scope	10.0.0.11	

Details:
This vulnerability exists because the encryption ciphers being used on this server are weak and susceptible to being broken by an attacker due to their short block length. Generally, the shorter the block length of an encryption stream like this, the easier it is to crack and gain access to the information it was protecting. This server is vulnerable to the SWEET32 cipher attack which would allow an attacker to listen in on network connections between a server and client web browser.

Impact:
An attacker could potentially launch a man-in-the-middle attack between a server and a client web browser. This would allow the attacker to view any information being passed between the two and even inject information into the stream.

Remediation:
Configuring web servers to use secure ciphers is a complicated process. An in-depth guide can be found here: https://crashtest-security.com/secure-tls-configuration/

Steps for Reproduction:
Run the following command on a Linux machine with nmap installed to check what ciphers are being used:

```
nmap -p 443 -script=ssl-enum-ciphers
```

For an explanation of how to exploit SWEET32, visit this link:
https://crashtest-security.com/prevent-ssl-sweet32/

Relevant Screenshots

```
# Nmap 7.93 scan initiated Sat Jan 14 06:24:07 2023 as: nmap -p 443 --script=ssl-enum-ciphers -oN tlscipher11.txt 10.0.0.11
Nmap scan report for 10.0.0.11
Host is up (0.0057s latency).

PORT    STATE SERVICE
443/tcp open  https
| ssl-enum-ciphers:
|   TLSv1.0:
|     ciphers:
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (ecdh_x25519) - F
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 1024) - F
|       TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (dh 1024) - F
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (ecdh_x25519) - F
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 1024) - F
|       TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (dh 1024) - F
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 1024) - F
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 1024) - F
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 1024) - F
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 1024) - F
|       TLS_DHE_RSA_WITH_SEED_CBC_SHA (dh 1024) - F
|       TLS_RSA_WITH_SEED_CBC_SHA (rsa 1024) - F
|       TLS_RSA_WITH_IDEA_CBC_SHA (rsa 1024) - F
|     compressors:
|       NULL
|     cipher preference: server
|     warnings:
|       64-bit block cipher IDEA vulnerable to SWEET32 attack
|       Insecure certificate signature (SHA1), score capped at F
|   TLSv1.1:
|     ciphers:
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (ecdh_x25519) - F
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 1024) - F
|       TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (dh 1024) - F
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (ecdh_x25519) - F
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 1024) - F
|       TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (dh 1024) - F
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 1024) - F
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 1024) - F
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 1024) - F
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 1024) - F
|       TLS_DHE_RSA_WITH_SEED_CBC_SHA (dh 1024) - F
|       TLS_RSA_WITH_SEED_CBC_SHA (rsa 1024) - F
|       TLS_RSA_WITH_IDEA_CBC_SHA (rsa 1024) - F
|     compressors:
|       NULL
|     cipher preference: server
|     warnings:
|       64-bit block cipher IDEA vulnerable to SWEET32 attack
|       Insecure certificate signature (SHA1), score capped at F
```

Snippet of an nmap scan showing the weak TLS ciphers being used on 10.0.0.11

Unrestricted Logon Access

Severity	Vector String	CVSS Score
Medium	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:N/A:N	**4.3**
Risk Rating	**1**	Likelihood: **LOW**
Affected Scope	10.0.0.12, 10.0.0.20, 10.0.0.200, 10.0.0.210	

Details:
Normally, machines are restricted in who can actually log into them. However, anyone with domain credentials can access the Linux systems in the domain. These are production machines, not workstations, and so this level of access is more than most employees need.

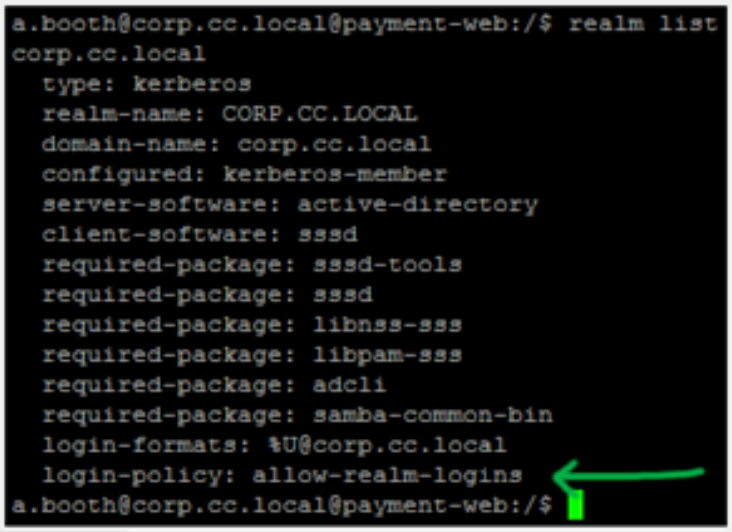
Impact:
This vulnerability allows an attacker with domain credentials to log in to any of the machines and to do what they want with those machines. This can lead to a loss of confidentiality, and potentially in availability as well, possibly leading to a monetary loss as well due to low-starred reviews.

Remediation:
Restrict access to the Linux machines to only those individuals or groups who need to access them.

Steps for Reproduction:

- Log into one of the Linux machines via ssh
- Run the command `realm list corp.cc.local` to see the policies applied to the Linux machines in the domain
- Look for the `login-policy` and see that it's set to `allow-realm-logins`

Relevant Screenshots



The `login-policy` field allows anyone in the Active Directory realm to login.

Administrator Accounts Enabled

Severity	Vector String	CVSS Score
Low	CVSS:3.0/AV:N/AC:L/PR:H/UI:N/S:U/C:L/I:L/A:N	**3.8**
Risk Rating	**1**	Likelihood: **LOW**
Affected Scope	10.0.0.5, 10.0.0.6, 10.0.0.11, 10.0.0.12, 10.0.0.20, 10.0.0.51, 10.0.0.52, 10.0.0.200, 10.0.0.210	

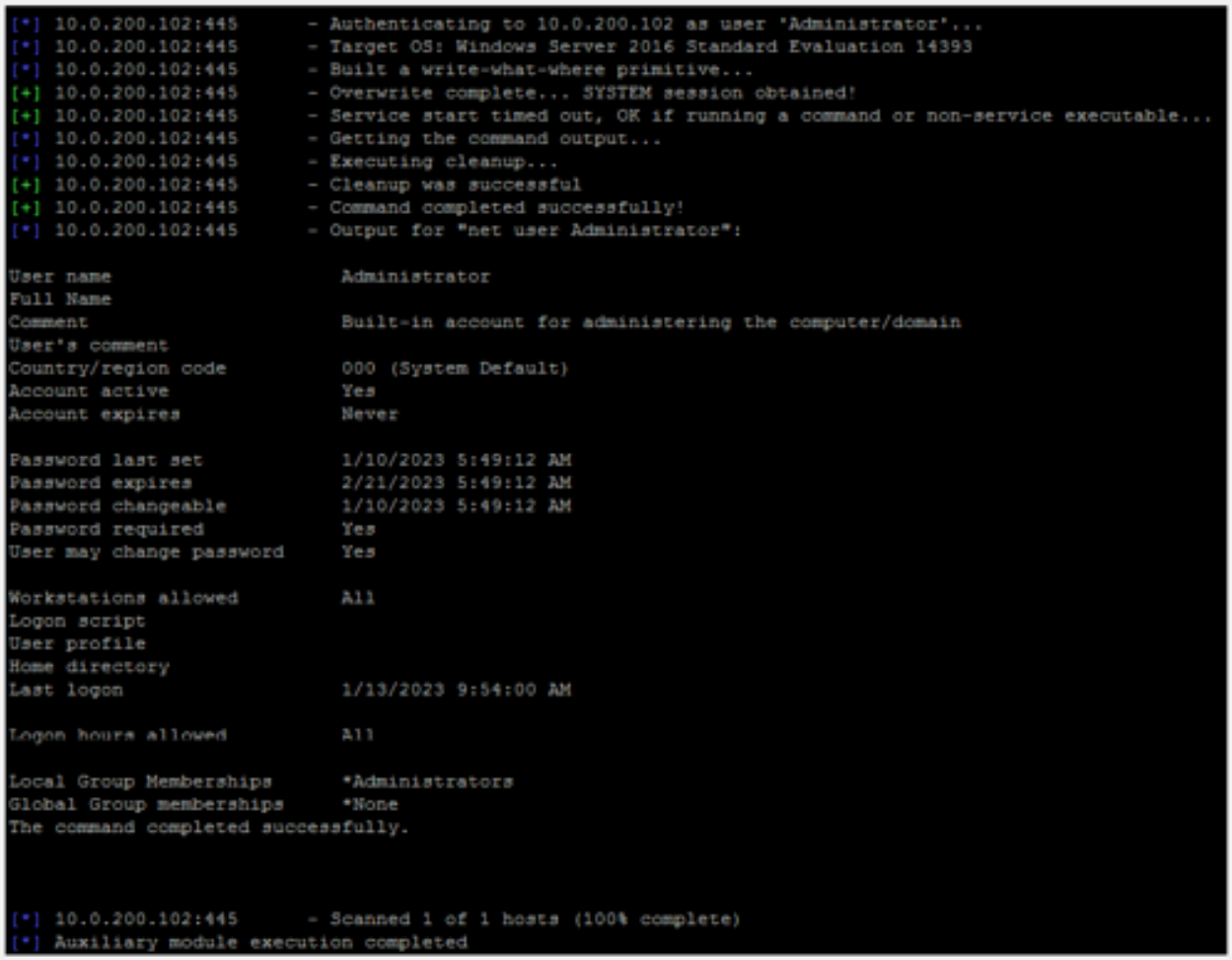
Details:
On each of these systems, the administrator account is enabled by default.

Impact:
As a general practice, it is not advised to enable the default administrator account. The reason for this is that if an attacker is trying to gain access to a computer, they can know that administrator is a valid user and they can begin to brute force crack the password.

Remediation:
After setting up the machine, remove the administrator account. This shores up the defenses of the machine and makes it harder for attackers to have guaranteed knowledge about the machines.

Steps to Reproduce:
Press `WIN+R` and type `lusrmgr.msc` to open Local Users and Computers. When selecting the Users folder, notice that the Administrator account is still enabled.

Relevant Screenshots



An active directory dump showing a policy to enable Administrator accounts

PermitRootLogin Allowed on Linux Hosts

Severity	Vector String	CVSS Score
Low	CVSS:3.0/AV:N/AC:L/PR:H/UI:N/S:U/C:L/I:L/A:N	**3.8**
Risk Rating	**2**	Likelihood: **HIGH**
Affected Scope	10.0.0.7, 10.0.0.12, 10.0.0.20, 10.0.0.200, 10.0.0.210	

Details:
The policy PermitRootLogin is set to yes on all Linux Hosts, meaning that the root user is able to ssh into the machine directly.

Impact:
Since the root username never changes and the access rights are unlimited, this account is the highest target for attackers. This means that intruders are most likely to attack the root account when using brute force and if compromised, they will gain complete control over the system.

Remediation:
To disable PermitRootLogin on a Linux host, use the following command:

```
/etc/ssh/sshd_config:
      PermitRootLogin no #disabled
```

Then, restart the ssh service with this command:

```
/etc/init.d/sshd restart
```

Relevant Screenshots

```
a.booth@corp.cc.local@payment-web:/$ cat /etc/ssh/sshd_config | grep 'PermitRoot'
PermitRootLogin yes
# the setting of "PermitRootLogin without-password".
a.booth@corp.cc.local@payment-web:/$
```

`PermitRootLogin` is set to yes, allowing the root user to SSH into the machine directly.

SMB Version 1 Enabled

Severity	Vector String	CVSS Score
Informational	CVSS:3.1/AV:A/AC:H/PR:H/UI:R/S:U/C:N/I:N/A:N	**0.0**
Risk Rating	**1**	Likelihood: **LOW**
Affected Scope	10.0.0.5, 10.0.0.6, 10.0.0.11, 10.0.0.51, 10.0.0.52	

Details:
SMB version 1 is an old version of the Server Message Block (SMB) service that has been deprecated for some time now. One common exploit that uses SMBv1 is EternalBlue, which enables remote code execution on a machine. No machines in your environment were found to be vulnerable to EternalBlue. With that in mind, it is recommended to disable SMBv1 and SMBv1.1 in this environment as an extra level of protection.

Impact:
No immediate exploitation of SMBv1 was found in this environment during this round of testing; however, that does not guarantee that other exploits do not exist or be found in the future. While the impact of this misconfiguration is minimal right now, that could easily change in the future.

Remediation:
Mitigation of this risk is as simple as turning off this service on all Windows machines in the environment.

Microsoft provides excellent documentation on these steps, which can be found here: (https://learn.microsoft.com/en-us/windows-server/storage/file-server/troubleshoot/detect-enable-and-disable-smbv1-v2-v3?tabs=server).

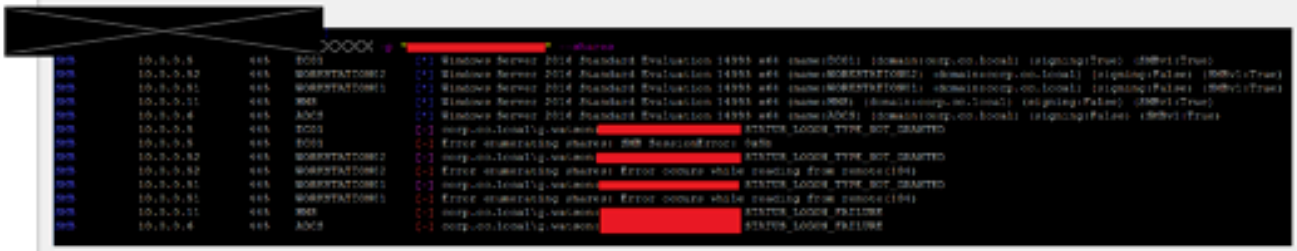
Powershell command for disabling SMBv1:

```
Disable-WindowsOptionalFeature -Online -FeatureName SMB1Protocol
```

Steps for Reproduction:
On a Linux host with the tool crackmapexec installed, run this command:

```
$ crackmapexec smb <ip or subnet to check> -u '<username>' -p 'password'
```

Relevant Screenshots



Using crackmapexec to show that certain machines have SMBv1 enabled

Single Constant Output in PasswordGenerator

Severity	Vector String	CVSS Score
Informational	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:C/C:N/I:N/A:N	**0.0**
Risk Rating	**1**	Likelihood: **MODERATE**
Affected Scope	10.0.200.103	

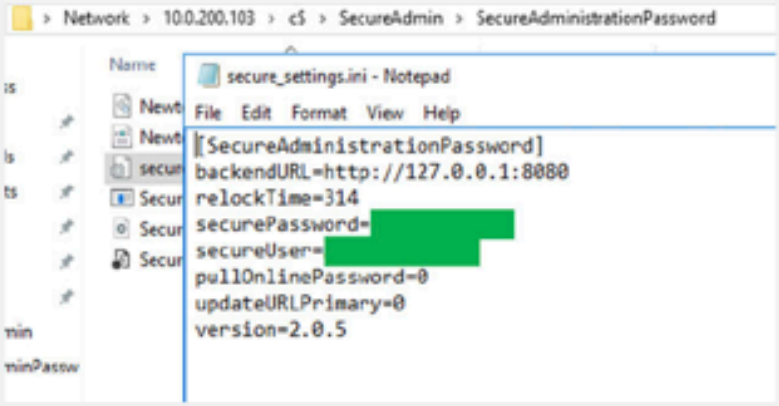
Details:
We found a folder in the 10.0.200.102 host titled `SecureAdministrationPassword`. In there, we found a file `secure_settings.ini`. Viewing the contents of the file, we found a plaintext password and associated username. We found that there was an executable file that showed to generate a secure password, but always outputted the password found in the ini file rather than generating a new password for each execution.

Impact:
Storing passwords in plaintext is bad practice, as well as storing them in files on user accounts. If admin passwords are stored in plaintext on user accounts, this could allow attackers to find them very easily, thus gaining administrative access to systems in TCC. Although we did not notice this particular password being used anywhere in the network, it may have been used in places we didn't try, and users may (without understanding how it works) use this password thinking it's secure.

Remediation:
This can be remediated by implementing a policy in TCC that passwords will not be stored in plaintext, as was found. Training TCC employees on this and ensuring that they know of the security impacts that this could have will help to mitigate this vulnerability.

Steps for Reproduction:
Navigate to `C:\SecureAdmin\SecureAdminPassword` in File Explorer and run `SecurePassword.exe`. You'll notice that it outputs the `securePassword` value in `secure_settings.ini` each time.

Relevant Screenshots



Inside of the SecureAdministrationPassword folder, an executable is run to generate an admin password. The output for this executable is always the value of `securePassword` *in* `secure_settings.ini`.

SMB Signing Disabled

Severity	Vector String	CVSS Score
Informational	CVSS:3.1/AV:A/AC:H/PR:H/UI:R/S:U/C:N/I:N/A:N	**0.0**
Risk Rating	**2**	Likelihood: **MODERATE**
Affected Scope	10.0.0.6, 10.0.0.11, 10.0.0.51, 10.0.0.52	

Details:
This vulnerability allows attackers to launch man-in-the-middle attacks against any machines using the SMB protocol and have SMB signing disabled.

Impact:
Attackers could use this method to gain access to sensitive information related to SMB shares on a network.

Remediation:
SMB signing needs to be enabled on all machines using the SMB protocol. This can be done on Windows through the command line interface or by using the Windows Local Group Policy Editor program.

Steps:
Launch `gpedit.msc`
Navigate to Computer > Windows Settings > Local Policies > Security Options
Set the following policy values:

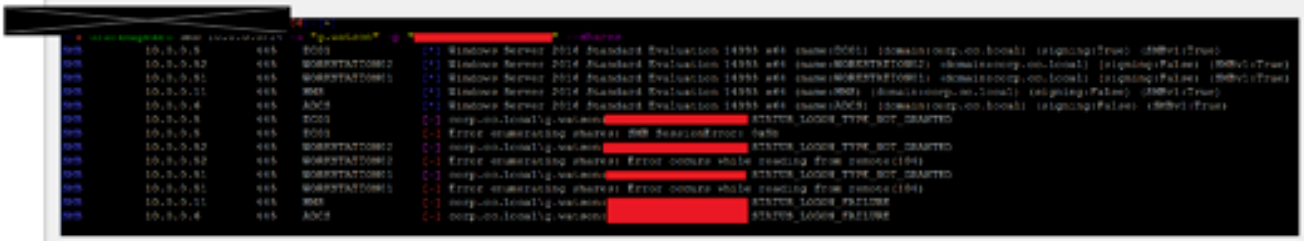
- Microsoft network client: Digitally sign communication (always)
- Microsoft network client: Digitally sign communication (if server agrees)
- Microsoft network server: Digitally sign communication (always)
- Microsoft network server: Digitally sign communication (if client agrees)

Once completed, restart the computer.

Steps for Reproduction:
On a Linux machine with the tool crackmapexec, run the following command:

```
$ crackmapexec smb <ip or subnet to scan> -u '<username>' -p '<password>'
```

Relevant Screenshots



Use of crackmapexec to show what hosts have SMB signing disabled

Unnecessarily Strict Password Policy

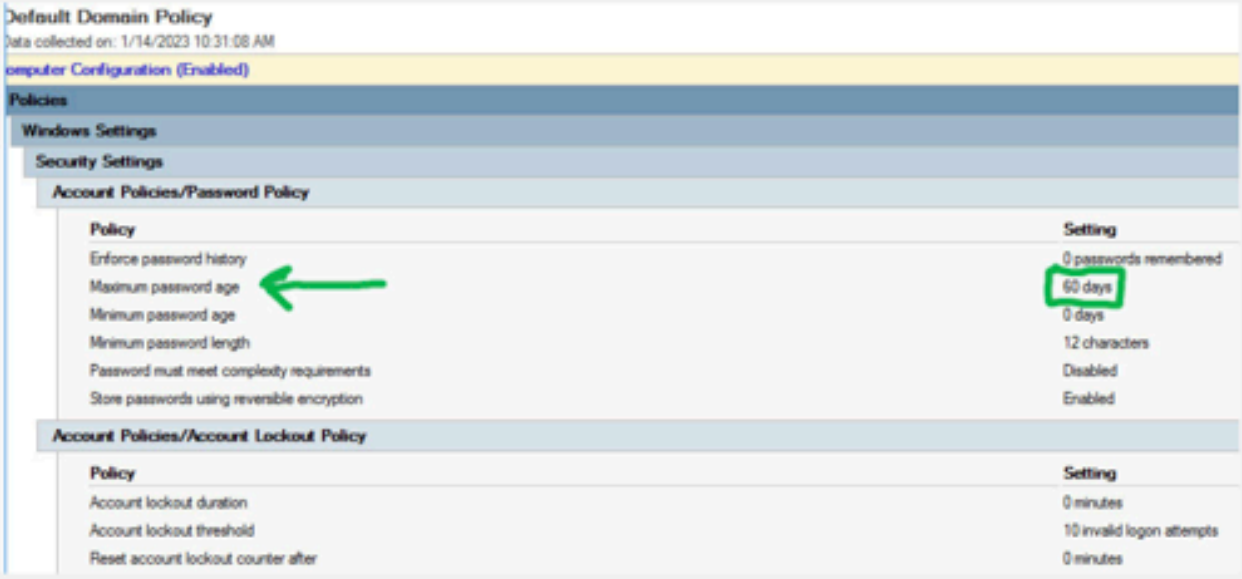
Severity	Vector String	CVSS Score
Informational	CVSS:3.0/AV:L/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:N	**0.0**
Risk Rating	**0**	Likelihood: **LOW**
Affected Scope	10.0.0.5, 10.0.0.6, 10.0.0.11, 10.0.0.12, 10.0.0.20, 10.0.0.51, 10.0.0.52, 10.0.0.200, 10.0.0.210	

Details:
The current domain password policy is for users to switch the password every 60 days.

Impact:
While it may seem like a good policy for users to change their passwords every two months, this actually encourages negligent password creation. When asked to change passwords too often, many users begin to use repetitious passwords that are more guessable.

Remediation:
Instead of focusing on switching passwords often, it is more beneficial to focus on password complexity or additional precautions like two-factor authentication. We recommend lengthening the time in between password changes and implementing a two-factor authentication service.

Relevant Screenshots



Default Domain Policy
Data collected on: 1/14/2023 10:31:08 AM
omputer Configuration (Enabled)
Policies
Windows Settings
Security Settings
Account Policies/Password Policy

Policy	Setting
Enforce password history	0 passwords remembered
Maximum password age	60 days
Minimum password age	0 days
Minimum password length	12 characters
Password must meet complexity requirements	Disabled
Store passwords using reversible encryption	Enabled

Account Policies/Account Lockout Policy

Policy	Setting
Account lockout duration	0 minutes
Account lockout threshold	10 invalid logon attempts
Reset account lockout counter after	0 minutes

The maximum password age is set to 60 days.

No Restriction on Ability to Add Workstations to Domain

Severity	Vector String	CVSS Score
Informational	CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H	**0.0**
Risk Rating	**0**	Likelihood: **LOW**
Affected Scope	10.0.0.5	

Details:
The Group Policies in the domain allow any authenticated user in the domain to add a computer into the domain. This may allow a malicious actor with access to stolen credentials to perform a number of attacks against the domain, such as better enumeration.

Impact:
An attacker could create workstations on the domain without restriction, allowing further enumeration of the domain, as well as being considered trusted within the domain. While not a major risk in and of itself, this can lead to further attacks of greater criticality.

Remediation:
Restrict the permission to add workstations to roles that would feasibly need to add workstations to the domain, such as technology specialists.

Steps for Reproduction:
Check the default domain policy in the GPO.

Relevant Screenshots

Any user on the Active Directory domain can add a new workstation to the domain.

Inactive Accounts Present in Domain

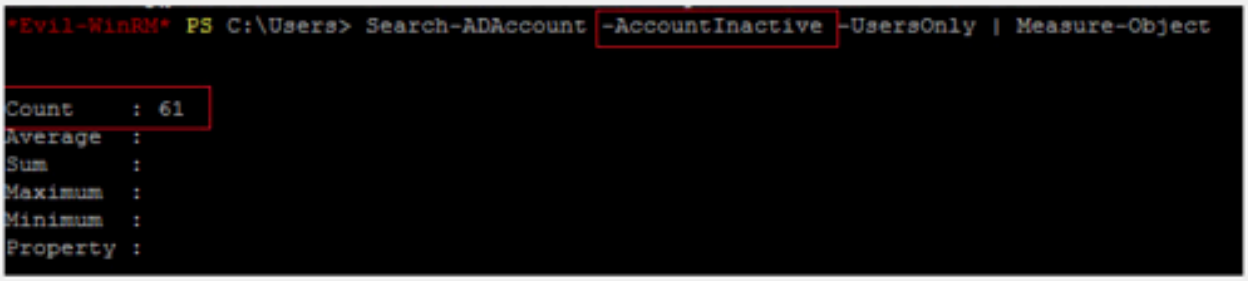
Severity	Vector String	CVSS Score
Informational	CVSS:3.0/AV:L/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:N	**0.0**
Risk Rating	**0**	Likelihood: **LOW**
Affected Scope	10.0.0.5, 10.0.0.6, 10.0.0.11, 10.0.0.12, 10.0.0.20, 10.0.0.51, 10.0.0.52, 10.0.0.200, 10.0.0.210	

Details:
In our active directory evaluation, we found that 61 domain accounts are inactive. This means that there has not been a password change or logon attempt for a significant period. These accounts likely belong to former employees.

Impact:
If a former employee develops a malicious motive against TCC, they could easily login to their old account and wreak havoc on existing systems. In addition, if an attacker breaks into an inactive account and begins to tamper, it is less likely to be noticed because no active user will notice changes to their account.

Remediation:
This issue can be resolved by having domain admins remove employee accounts upon termination. Or, a group policy object can be implemented to auto-disable inactive accounts after a certain period of time.

Relevant Screenshots



A count for inactive domain accounts is run through Evil-WinRM, totalling to 61 accounts

Engagement Interactions

Throughout both engagements, the Engagement team responded to various requests and in-person interactions. Below are descriptions of each request and results and deliverables of the Engagement Team.

In-Person Engagement

1. Physical Assessment of a Safe purchased by TCC.
    a. Objective: Analyze the lock, keypad and locking mechanism using non-destructive entry methods to the safe.
    b. Engagement Team Results:
        i. First Compromise: Weak Code Credential
            1. The Safe provided, contained extensive wear on two keys specifically on the keypad. By brute forcing the possibilities given two known digits, we were able to quickly open the safe.
            2. Impact: Because the Safe allows the code to only have a minimum of 1 digit, customers may potentially arm the safe with an easily guessed code. This can lead to easy access by an adversary. The safe does contain a 30 second lockout after 3 incorrect guesses, but that alone is not enough to prevent brute forcing a code with at minimum, 9 possible combinations.
        ii. Second Compromise: Disengaging Spring Mechanism
            1. Internal to the safe is a spring, which keeps the door locked in place. By striking the top of the safe while turning the key, the spring may be compressed and the lock disengaged at the right time, allowing the door to be opened.
            2. Impact: Malicious actors may be able to access the contents inside the safe without having to know any combinations. Note that this attack vector only works if the safe does not have anything bolted on top of it.
        iii. Third Compromise: Externally Engaging Code Reset
            1. Two holes are present in the back of the safe, and a reset button is placed on the inside of the safe for resetting the code. A long, thin, hard object such as a metal straw can be used to reach through one of the holes to the reset mechanism
            2. Impact: An attacker may be able to reset the code for the safe without knowing the old code, allowing them access and denying the actual user access. Note that this attack vector only works if the holes are not covered up by walls.
    c. Assessment of Provided Safe
2. Scope Violation Plan
    a. Task- Define a Scope violation plan that covers the following two scenarios:

i. A member of the engagement team performs an action on a resource that is out of scope of the engagement.
ii. A member of the engagement team performs an action that causes a service outage.

b. Solution
i. Our plan addressed both scenarios with similar steps, those steps being the following:
1. Initiate contact with the Client's IT team prior to performing any action with the capability to severely impact service availability.
2. Once receiving the go-ahead, perform the action or exploit of concern, and observe for any significant impact to the environment.

3. Presentation of OSINT Findings
a. It was requested that the Engagement team report in person on the results of the pre-engagement OSINT work. The discovered information can be reviewed in Appendix D.

4. Vishing Call
a. Task
i. Under any pretext, call the main desk of TCC and attempt to get any information and PII belonging to clients as possible.

b. Our Plan
i. Our approach was to pose as the husband of a TCC rewards member wanting to set up a weekend stay for his wife. Using this guise, we would attempt to use information we had pulled from the rewards system database to "prove" to the receptionist that we were the husband of this lady and get the address and credit card information for her that they had on file.

c. Results
i. The real-world vishing test did not go as we had planned. When we asked the receptionist about "our wife's" account, the receptionist told us that she could not find an account with that information in the system. This probably occurred because she was looking in the wrong database, perhaps the employee or other customer database. While we were unable to get any credit card or address information from the receptionist we were able to convince her to create an account under the name and email of this customer that we had pretended was our wife. We had not expected this setback and the receptionist ended the call before we could attempt to get much more information from her.

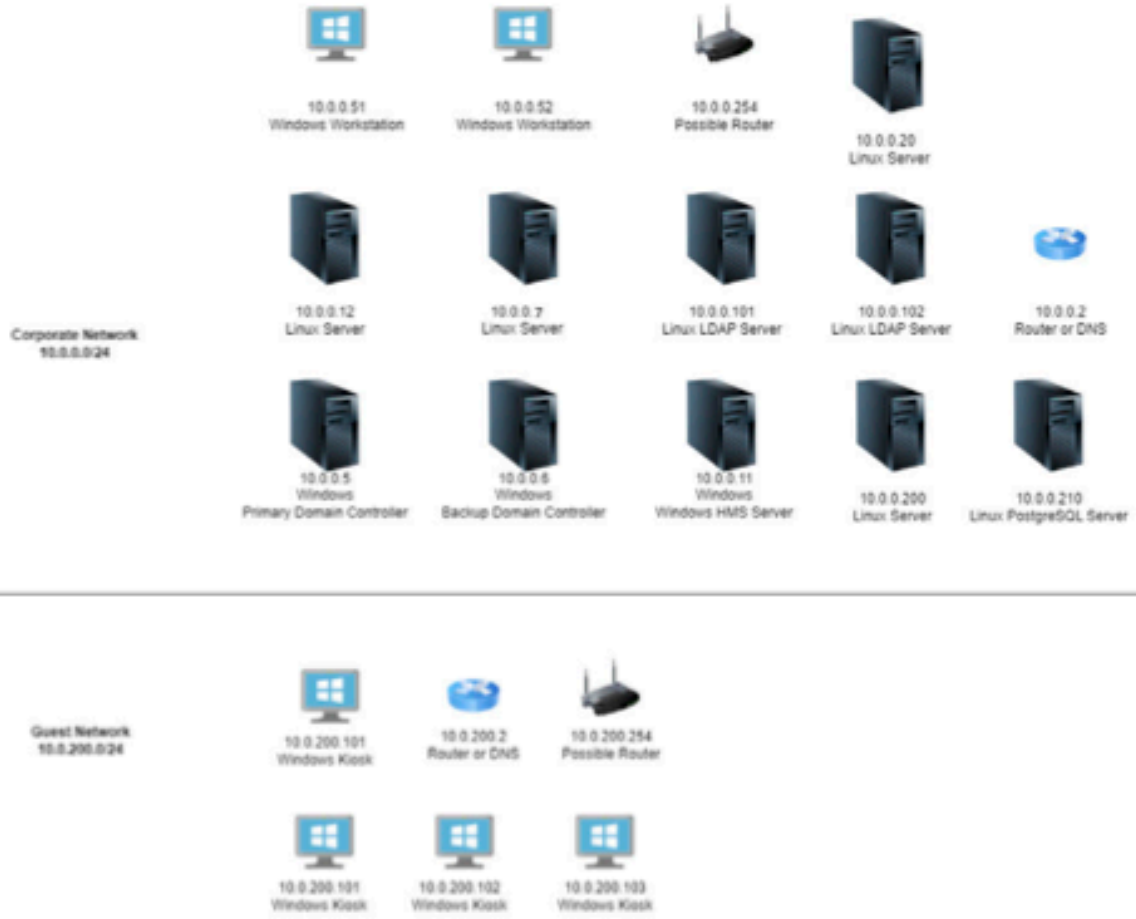
Conclusion

After our assessment and rigorous testing of various systems, we have concluded that security has been greatly improved since our last engagement. The implementation of network ACLs successfully removed much unnecessary user access to services on the TCC network and greatly increased the difficulty of finding previously discovered vulnerabilities. We also observed other security improvements, such as the updated password policy and the increased employee awareness of social engineering attempts. However, from our security assessment, we were still able to find many vulnerabilities in the TCC environment and have identified various ways that security can be improved. From our assessment, we have identified four recommendations that we would prioritize above all others. These recommendations are using stronger passwords for service and support accounts, disabling default guest and admin accounts, frequent and consistent updates and patches on all hosts, and implementing redundancy for critical services and systems.

Our team was very impressed with the security improvements made on this system by the Client. We know that the Client is passionate about keeping their customer's data secure and are confident that by implementing our recommended changes, the systems and customers will be better protected.

Appendix

Appendix A - Network Layout



Appendix B - Engagement Artifacts

- WP directories
    - While exploiting Wordpress through metasploit, the shell spawned on that host created residual files within the wordpress file structure.
    - These files are contained in folders that follow the naming convention `/wp-content/plugins/x` where x is a 10-character randomly-generated string.
- Added the computer `WIN-DGY21FACCGA$` to the domain

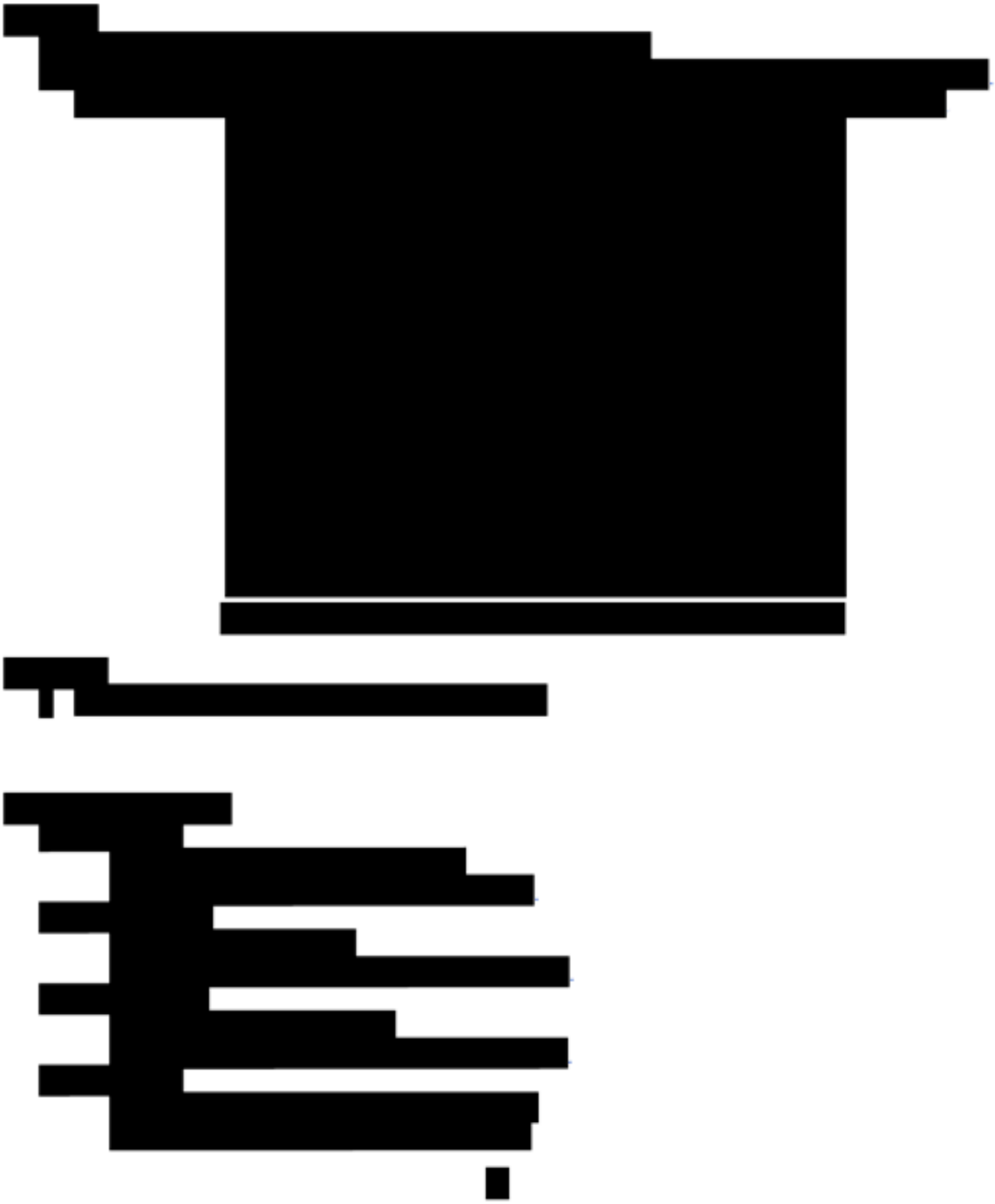
Appendix C - Tools Utilized

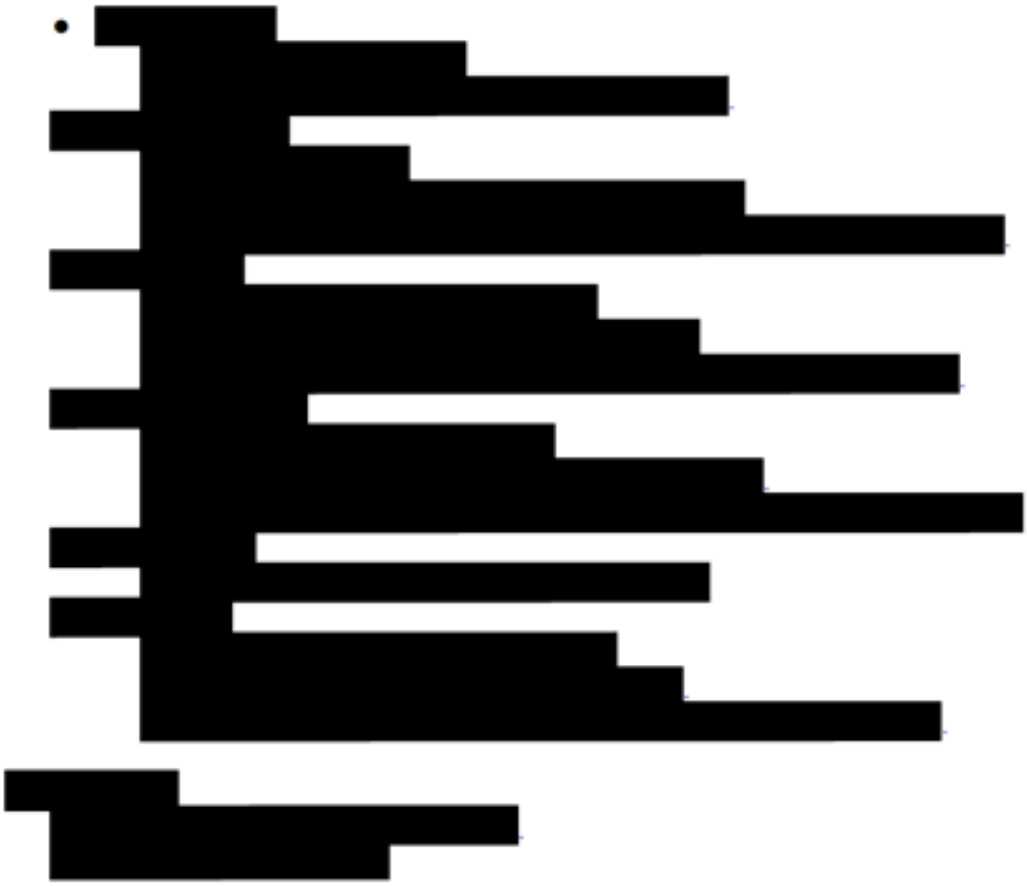
- BloodHound/SharpHound
  - BloodHound is a data discovery tool used to unwrap the layers of Active Directory and identify unique attack vectors and relationships within the Active Directory environment.
  - SharpHound is the data collection tool that works within BloodHound, written in C# to collect the data from AD. (https://bloodhound.readthedocs.io/en/latest/index.html)
- BurpSuite Community Edition
  - BurpSuite is a platform for vulnerability scanning, penetration testing and web application testing (https://portswigger.net/burp/communitydownload)
- CrackMapExec
  - Considered a "Swiss Army Knife for pentesting Windows/AD", CrackMapExec is a pure python script included in the Kali Suite. (https://github.com/Porchetta-Industries/CrackMapExec)
- CVSS Calculator
  - We used this site to calculate the final CVSS scores for this report (https://www.first.org/cvss/calculator/3.0)
- Dradis Framework Community Edition
  - Dradis is an open-source tool for tracking findings in penetration tests. We used this to keep track of our findings during the penetration test (https://dradisframework.com/ce)
- EmailRep
  - A simple email validator to verify if an email account is legitimate or not. (https://emailrep.io/)
- Enum4linux
  - Enum4linux is a tool for enumerating information from Windows systems and Samba shares (https://github.com/CiscoCXSecurity/enum4linux)
- Evil-WinRM
  - Evil-WinRM is a Windows tool that extends the features of the Windows Remote Management protocol by providing a slew of extra tools for taking advantage of Windows-based systems (https://github.com/Hackplayers/evil-winrm)
- Hydra
  - Hydra is a powerful tool for launching brute-force attacks with custom user and password lists against a number of services (https://github.com/vanhauser-thc/thc-hydra)
- Impacket
  - A collection of python scripts for exploiting network protocols (https://github.com/SecureAuthCorp/impacket)
- LDAP DomainDump
  - A tool used to retrieve Active directory information available through the LDAP protocol. (https://github.com/dirkjanm/ldapdomaindump)
- Metasploit Framework
  - Metasploit is a modular penetration testing platform for running exploits, enumerating systems, evading detection, etc. (https://github.com/rapid7/metasploit-framework)
- Neo4j
  - Neo4j is an open-source, NoSQL, native graph database. We utilized this to visualize and decipher the output of SharpHound from within BloodHound. (https://neo4j.com/)
- Nmap
  - Nmap is a tool commonly used to scan networks used for host discovery, port and service enumeration, and has a collection of scripts for testing various services (https://github.com/nmap/nmap)

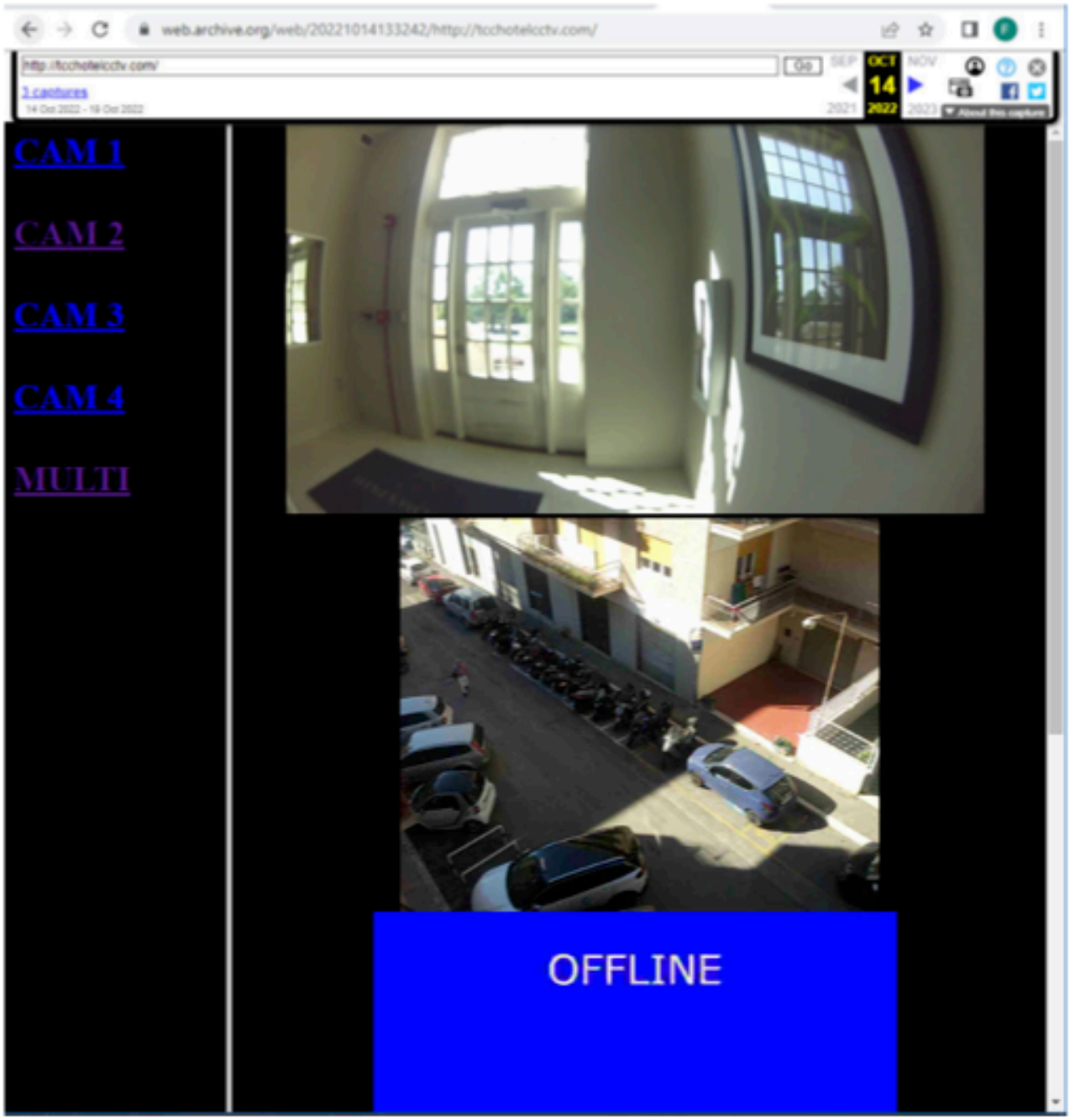
- NoPac
  - NoPac is a tool to exploit CVE-2021-42278 and CVE-2021-42287 to impersonate DA from a standard domain user (https://github.com/Ridter/noPac)
- Psql
  - PostgreSQL interactive terminal (https://www.postgresql.org/docs/current/app-psql.html)
- SMBClient
  - A tool within the Samba suite, allowing the user to interact more easily with a SMB server. (https://www.samba.org/samba/docs/current/man-html/smbclient.1.html)
- WinSCP
  - Graphical user interface for transferring files between computers via SFTP (https://www.postgresql.org/docs/current/app-psql.html)
- WPScan (enumeration capabilities only)
  - An open source word press security scanner tool for black box testing. (https://wpscan.com/wordpress-security-scanner)
- Wayback Machine
  - Internet Archive site used in our passive data collection in the OSINT portion of the engagement. (https://web.archive.org/)
- Zerologon (CVE-2020-1472) exploit
  - Zerologon allows an attacker to take over a domain controller with little effort (https://www.sprocketsecurity.com/resources/how-to-exploit-zerologon)

Appendix D - OSINT Information

For the Open Source Intelligence (OSINT) portion of this penetration test we scoured the internet for all traces of information on The Cozy Croissant and its employees. The following is a summary of the information we found available using only free, publicly available sources.







CCTV Archived Image on Web-Archive tool