OFFPORT

Cyber Security Services

Contents

1 Introduction

1.1 Executive Summary

1.1.1 Background

[Client Legal Name] (“Client” or “[Client Short Name]”) engaged Unit 42, a Palo Alto Networks, Inc. company, (“Consultant” or “Unit 42”) to perform an [insert type of test] on Client’s [internal/external/application’s] assets to identify potential security vulnerabilities. The assessment was conducted from [Assessment Start Date], to [Assessment End Date].

1.1.2 Team

Name	Role	Contact
First Last	Analyst	email1@corp.com
First Last	Senior Analyst	email2@corp.com
First Last	Senior Consultant	email3@corp.com

1.1.3 Scope

- 10.1.0.0/22
- 10.1.4.0/22
- 10.1.8.0/24
- 10.1.10.0/23
- 10.1.12.0/24
- 10.1.14.0/24

1.1.3.1 Scope Exclusions

Following reconnaissance of [Client Short Name]'s environment, the following domains were added to the scope of the assessment:

- 64.21.XXX.XXX
- 198.233.XXX.XXX

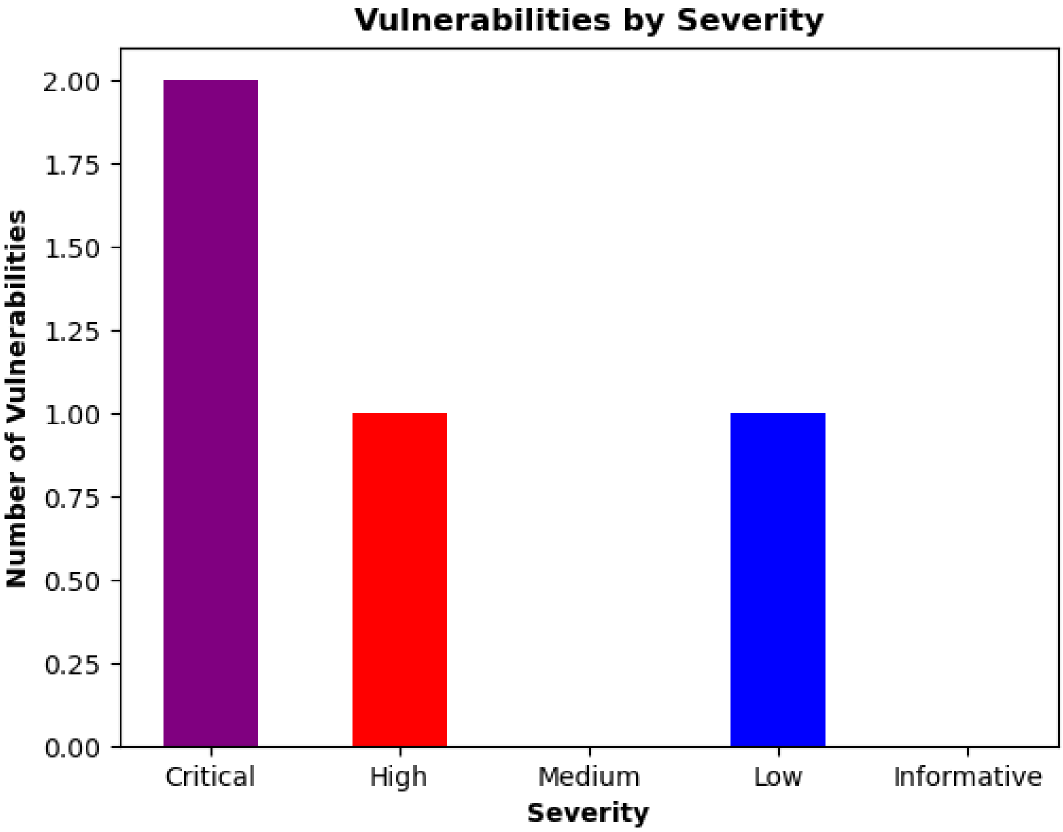
- 64.21.XXX.XXX
- 20.40.XXX.X

1.1.4 Limitations

- Time allowed for engagement [#Days]
- Disallowed attacks

1.1.5 Summary of findings

The below table shows the total number of findings categorized by overall risk to the organization.



1.1.6 Summary of Recommendations

The following recommendations provide direction on improving the overall security posture of networks and business-critical applications:

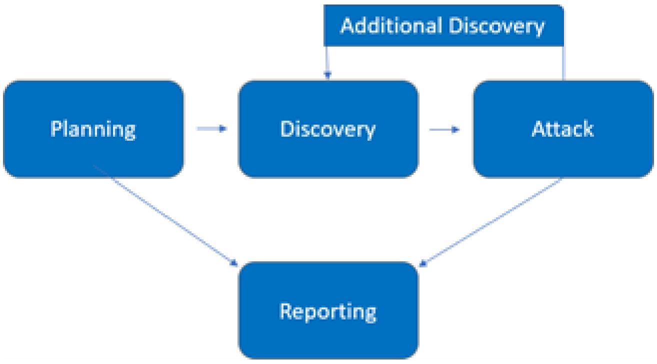
- Patch this
- Upgrade that
- Disable this

2 Technical Details

2.1 Methodology

All testing performed is based on the NIST SP 800-115 Technical Guide to Information Security Testing and Assessment, OWASP Testing Guide (v4), and customized testing frameworks. Phases of penetration testing activities include the following:

- Planning – Customer goals are gathered and rules of engagement obtained.
- Discovery – Perform scanning and enumeration to identify potential vulnerabilities, weak areas, and exploits.
- Attack – Confirm potential vulnerabilities through exploitation and perform additional discovery upon new access.
- Reporting – Document all found vulnerabilities and exploits, failed attempts, and company strengths and weaknesses.



2.2 Level of Access

For this assessment, Unit 42 was provided local administrator credentials to a Microsoft Windows workstation on [Client Short Name]’s network

Account	Level of access
Admin 1	Administrator user with full access to the system
Test account 1	Standard user with access to XYZ
Test account 2	Standard user with access to ABC

2.3 Reconnaissance

Scope validation

External testing was conducted remotely from Unit 42 facilities and by consultants using VPNs. [Client Short Name] provided Unit 42 with a range of IP addresses to be used as potential targets for testing. Unit 42 validated these ranges using whois data provided by the American Registry for Internet Numbers ("ARIN"). Reviewing this information, Unit 42 determined that several of the IP addresses were leased from different organizations, as seen below. Clent, however, confirmed that all of these IP addresses are in scope for the assessment.

```
for x in $(cat ept-cidrs.txt); do echo "[+]" $x; whois $x | tee -a whois-
    full.txt | grep -Ei 'Organization|OrgName|netname|descr|address|bal';
    done | tee whois.txt
```

DNS enumeration

Unit 42 began testing by aggregating open-source intelligence ("OSINT") on [Client Short Name]'s systems. OSINT data consists of information scraped from public resources that can be leveraged for enumerating and attacking in-scope systems. This type of intelligence gathering is often passively collected through nontarget resources, such as search engines. Unit 42 leveraged several automated tools to perform data collection. Using the sublist3r tool, Unit 42 discovered several subdomain targets that fell within the provided scope.

```
sublist3r -d client.com -o sublisterOutput
...

# Confirming in-range targets
for x in $(cat sublist3r.txt); do echo $x; host $x; done
...
```

User harvesting

Continuing OSINT collection, Unit 42 used BridgeKeeper to scrape common search engines, LinkedIn, and other websites for employee names to generate potentially valid usernames. Using this technique, Unit 42 identified several unique email addresses that could be used in further testing.

```
python3 bridgekeeper.py --company “client”--domain client.com --depth 10
```

Network scanning

Unit 42 conducted reconnaissance throughout the assessment to help identify and enumerate external services and ensure that they were tested as part of the assessment. Unit 42 performed active scanning against in-scope assets with Nmap covering all Transmission Control Protocol (“TCP”) and User Datagram Protocol (“UDP”) ports. Service enumeration and operating system detection was also conducted with the same tools during this phase. This information was used to drive exploitation attempts and help Unit 42 target potentially vulnerable assets.

```
nmap -sSU -p 1-65535 -iL scope.txt -oA client
```

3 Findings

The following section contains the detailed findings for the [internal/external/ web application] [pen test / assessment]. See the appendix for a description of how each vulnerability is rated.

RF#	Severity	Finding
1	Critical	SQL Injection
2	Critical	Command Injection
3	High	Reflected XSS
4	Low	Information Disclosure

3.1 Critical - SQL Injection

Vulnerability Name	SQL Injection
Severity	Critical
Exploitability	Hacker
User Interaction	Not required
Attack Vector	Internet
Affected Scope	http://192.168.46.10:3000/#/login

Description

SQL injection attacks are a type of injection attack, in which SQL commands are injected into data-plane input in order to affect the execution of predefined SQL commands.

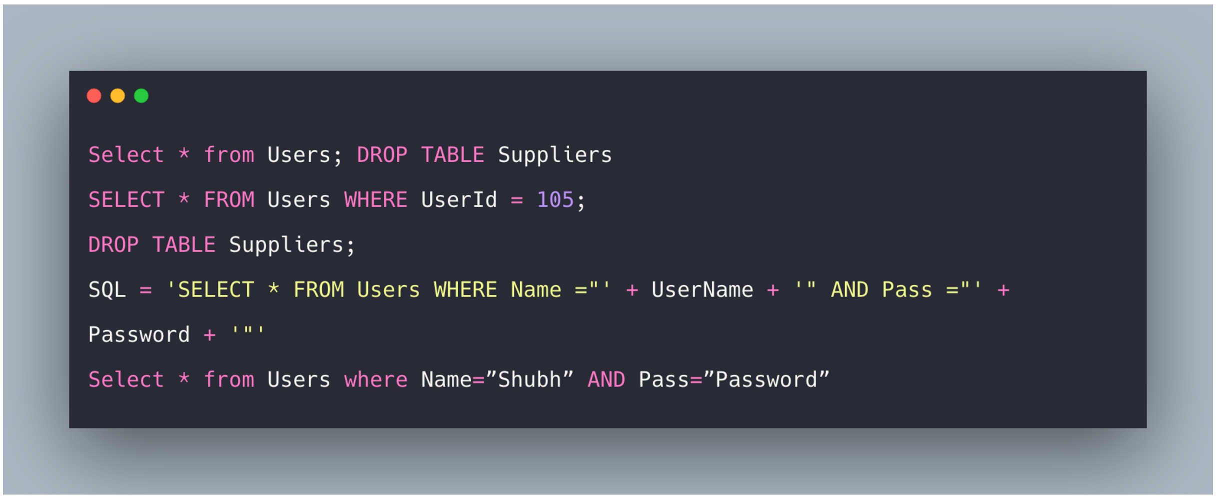
Username parameter in the login page is vulnerable to SQL injection.

Impact

The vulnerability allows to bypass the login page to login as any user including the admin without the password.

Test Details

Screenshots



```
Select * from Users; DROP TABLE Suppliers
SELECT * FROM Users WHERE UserId = 105;
DROP TABLE Suppliers;
SQL = 'SELECT * FROM Users WHERE Name ="' + UserName + '" AND Pass ="' +
Password + '"'
Select * from Users where Name=”Shubh” AND Pass=”Password”
```

Payload

```
admin' or '1'='1'--
```

Request

```
POST /rest/user/login HTTP/1.1
Host: 192.168.46.10:3000
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:92.0)
    Gecko/20100101 Firefox/92.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
Content-Length: 49
Origin: http://192.168.46.10:3000
Connection: close
Referer: http://192.168.46.10:3000/
Cookie: language=en; welcomebanner_status=dismiss; continueCode=
    E3OzQenePWoj4zk293aRX8KbBNYEAo9GL5qO1ZDwp6JyVxgQMmrlv7npKLVy

{"email":"admin' or '1'='1'--","password":"1234"}
```

Response

```
    HTTP/1.1 200 OK
    Access-Control-Allow-Origin: *
    X-Content-Type-Options: nosniff
    X-Frame-Options: SAMEORIGIN
    Feature-Policy: payment 'self'
    Content-Type: application/json; charset=utf-8
    Content-Length: 831
    ETag: W/"33f-ncvYKqi8+0GW88YIH7i3xTrVgeY"
    Vary: Accept-Encoding
    Date: Fri, 01 Oct 2021 07:07:42 GMT
    Connection: close
```

Remediation

- Most instances of SQL injection can be prevented by using parameterized queries (also known as prepared statements) instead of string concatenation within the query.
- Data sanitization. For more info, check references.

References

- https://portswigger.net/web-security/sql-injection
- https://blog.sqreen.com/preventing-sql-injection-in-node-js-and-other-vulnerabilities/
- https://javascript.plainenglish.io/sql-injection-attack-in-nodejs-a840893f228b

---

3.2 Critical - Command Injection

Vulnerability Name	Command Injection
Severity	Critical
Exploitability	Hacker
User Interaction	Not required
Attack Vector	Internet
Affected Scope	http://192.168.46.10:3000/#/login

Description

Text

Impact

Text

Test Details

Screenshots

Payload

```
code
```

Remediation

- How to fix/prevent.

References

- https://url
- https://url

3.3 High - Reflected XSS

Vulnerability Name	Command Injection
Severity	High
Exploitability	Hacker
User Interaction	Not required
Attack Vector	Internet
Affected Scope	http://192.168.46.10:3000/#/login

Description

Text

Impact

Text

Test Details

Screenshots

Payload

```
code
```

Remediation

- How to fix/prevent.

References

- https://url
- https://url

3.4 Low - Information Disclosure

Vulnerability Name	Command Injection
Severity	Low
Exploitability	Technical
User Interaction	Not required
Attack Vector	Internet
Affected Scope	http://192.168.46.10:3000/#/info

Description

Text

Impact

Text

Test Details

Screenshots

Payload

```
code
```

Remediation

- How to fix/prevent.

References

- https://url
- https://url

4 Appendix

4.1 Definitions

The team rates vulnerabilities on a scale from a low to critical risk, depending on the level of impact and exploitability of the vulnerability. Impact

The table below details the risk classification matrix used to rate vulnerabilities based on their Impact vs. Exploitability.

Level	Definition of criticality level
Critical	Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately.
High	Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible.
Medium	Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved.
Low	Vulnerabilities are non-exploitable but would reduce an organization's attack surface. It is advised to form a plan of action and patch during the next maintenance window.
Info	No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation.

Level	Definition of exploitability level
User	**The exploitation of the vulnerability requires only basic levels of skills and means.** This implies that the exploitation is easy.
Techincal	**The exploitation of the vulnerability requires the skills and means of a technical person.** This implies capabilities of developing a utility to exploit the vulnerability without necessarily having significant means.
Hacker	**The exploitation of the vulnerability requires the skills of a hacker.** This implies a considerable knowledge in information systems security.