بسم الله،
اهلا بك، صديقي العزيز، في رحلة تعلم مجال ألمن السيبراني Cybersecurity المستوى الاول، المهارة الثانية. هذه التمارين تم إعدادها في مارس ،2025.

خطة التعلم الخطة تعتمد على اكتساب مهارة جديدة كل أسبوع، بإذن الله . هذا هو رابط الفيديو المرتبط بهذه التمارين:

https://www.youtube.com/watch?v=6dOUY55XJ-0

من المفترض أن تأخذ وقتك وراحتك أثناء حل التمارين، لأنها تتطلب منك بعض البحث عن المواضيع التي تم شرحها وذكرها في الحلقة، كيف يكون شكل ال format الخاصة بال ipv6 والفرق بينه وبين الـ ipv4 ، وتصنيفاته.

ولأنه ايضا في هذة التمارين سيتطلب عليك معرفة بالـ python، فإذا لم تقم بدراستها بعد، فعد الى الفديو السابق…

مطالب منك حل اليوم الاول والثاني، اما الباقي اختياري في الوقت الحالي، لانه سيتم شرح مواضيعهم باستفاضة لاحقا.

تنبيه مهم: إذا قمت بنسخ السؤال ولصقه في ChatGPT مباشرة للحصول على الحل، فرجاً ء قم بإلغاء اشتراكك واالنسحاب مبكًرا من الرحلة. صدقني، أنا أساعدك على توفير وقتك ومجهودك غير المرئي""

ملاحظة: اذا انهيت جميع التحديات المطلوبة هنا، سيتم نقلك مباشرة الى C class hacker.

🔥 Day 1: IP Addresses & Subnetting

Topics: IP Basics, IPv4, IPv6, Subnetting, CIDR Notation

- Exercises:

1. Write a Python script that validates whether a given IP address is IPv4 or IPv6.
2. Convert an IPv4 address from dotted-decimal to binary format.
3. Extract all IP addresses from a text file.
4. Write a function that checks if an IP belongs to a private network (`192.168.x.x`, `10.x.x.x`, etc.).
5. Given an IP and subnet mask, calculate the network address and broadcast address.
6. Generate a list of all possible IPs in a given subnet (`192.168.1.0/24`).
7. Create a script that checks if an IP is reachable by pinging it.
8. Convert an IPv6 address to its compressed and expanded forms.
9. Write a function that determines the class of an IPv4 address (`A, B, C, D, E`).
10. Given two IPs, determine if they belong to the same subnet.

🔥 Day 2: Ports & Protocols

Topics: Port Numbers, Well-Known Ports, Protocols (TCP/UDP)

- 🔹 Exercises:

1. Print a list of common ports and their corresponding services (e.g., `80 → HTTP`).
2. Write a function that checks if a given port number is valid (0-65535).
3. Write a script that scans the first 1000 ports of a given IP to see if they are open.
4. Detects whether a given port uses TCP or UDP.
5. Randomly generate 5 open ports between `1024-65535`.
6. Create a function that tells whether a given port is in the privileged range (`0-1023`).

7. Write a script that finds all listening ports on your local machine (`127.0.0.1`).
8. Implement a basic TCP port scanner using Python's `socket` module.
9. Implement a basic UDP port scanner.
10. List all reserved ports (1-1023) that are commonly used in hacking.

🔥 Day 3: TCP & UDP

Topics: Connection-oriented vs. Connectionless, Sockets, Packet Handling

- 🔹 Exercises:

1. Write a simple TCP client-server program that sends a message from client to server.
2. Modify the above program to work with UDP instead of TCP.
3. Create a TCP server that listens on port 9999 and responds with `"Hello, Client!"`.
4. Write a function that measures the round-trip time (RTT) of a TCP connection.
5. Simulate a basic SYN flood attack (for controlled environments).
6. Write a script that monitors and logs all incoming TCP connections.
7. Implement a TCP handshake simulation in Python.
8. Create a script that tests whether a remote server allows telnet access (port 23).
9. Write a UDP client that sends a DNS request manually to `8.8.8.8`.
10. Implement a simple packet sniffer that captures TCP and UDP packets.

🔥 Day 4: DNS & Domain Name Resolution

Topics: DNS Resolution, Reverse DNS Lookup, Subdomain Enumeration

- 🔹 Exercises:
  1. Resolve a given domain name to an IP address.
  2. Perform a reverse DNS lookup for a given IP.
  3. Extract all domain names from a given text file.
  4. Find the NS (Name Server) records of a domain.
  5. Retrieve the MX (Mail Exchange) records of a domain.
  6. Enumerate all subdomains of `example.com` using a wordlist.
  7. Check if a domain has an SPF record for email security.
  8. Perform a WHOIS lookup for a given domain.
  9. Create a script that monitors if a domain's IP has changed over time.
  10. Implement a brute-force subdomain scanner using Python's `socket` module.

🔥 Day 5: HTTP, Requests & Web Scraping

Topics: HTTP Headers, Requests, Scraping, Status Codes

- 🔹 Exercises:
  1. Send an HTTP request to `http://example.com` and print the response headers.
  2. Extract all links from a webpage using `BeautifulSoup`.
  3. Write a script that checks if a website is up or down.
  4. Extract the title of a webpage.
  5. Write a program that fetches the robots.txt file of a website.

6. Write a script that sends a custom user-agent header with an HTTP request.
7. Simulate a web login request with Python's `requests` module.
8. Write a script that downloads all images from a given URL.
9. Implement a basic web scraper that collects headlines from a news website.
10. Perform a basic web enumeration using a wordlist (e.g., find admin pages).

🔥 Day 6: Network Analysis & Sniffing

Topics: Capturing Packets, Analyzing Network Traffic

- 🔹 Exercises:

1. Capture all outgoing packets on your network interface.
2. Write a script that monitors for DNS queries on your local machine.
3. Extract all HTTP requests from captured packets.
4. Write a tool that sniffs for passwords sent over HTTP.
5. Detect if there is an ARP spoofing attack happening on your network.
6. Write a script that lists all MAC addresses on your network.
7. Capture and analyze TCP handshakes.
8. Create a script that listens for broadcast messages on the network.
9. Extract all visited URLs from a PCAP file.
10. Implement a simple Wi-Fi deauthentication attack simulation.

🔥 Day 7: Hacking Simulations & Mini Projects

Topics: Offensive Security & Automation

- Exercises:

1. Build a port scanner that scans a range of IPs and reports open ports.
2. Create a brute-force script for a web login page (ethical use only).
3. Write a network monitoring tool that logs all connected devices.
4. Simulate a Man-in-the-Middle (MITM) attack (controlled environment).
5. Implement a basic packet injection tool.
6. Automate a ping sweep to find live hosts on a subnet.
7. Write a script that continuously logs all new devices joining the network.
8. Create a basic phishing detection tool that checks URLs for suspicious patterns.
9. Build a script that extracts metadata from image files (EXIF data).
10. Write a Python script that detects DNS tunneling attempts.