IOT AND OT HACKING

Module-18

Aniket Sunil Pagare

📘 IoT and OT Hacking – Table of Contents

---

1. Description

- Overview of IoT and OT Hacking
- Importance in cybersecurity and infrastructure

🛰️ 2. Internet of Things (IoT)

🏭 3. Operational Technology (OT)

3.9 Top Challenges in OT Environments
3.10 Common OT Vulnerabilities

🧪 4. Practical IoT and OT Hacking with ICSIM

4.1 What is ICSIM?
4.2 Objectives of ICSIM
4.3 Key Features of ICSIM
4.4 Components of ICSIM
4.5 ICSIM for IoT / OT Hacking Learning
4.6 Learning Outcomes
4.7 Perform Attacks Using ICSIM

- Simulating control loss
- Packet interception
- Replay attacks

🌐 5. Exploring IoT and OT Device Exposure Using Shodan

5.1 What is Shodan?
5.2 What Does Shodan Do?
5.3 Example Use Cases of Shodan
5.4 Perform Activity Using Shodan Search Engine

- Find open webcams, SCADA panels, routers
- Analyze headers, ports, CVEs

🎯 6. EXTRA ACTIVITY: Additional Search Engines

🔎 6.1 ZoomEye Search Engine

- What is ZoomEye?
- Perform Activity Using ZoomEye
    - Search ICS devices, IP cameras, PLCs

🔍 6.2 FOFA Search Engine

- What is FOFA?
- What is FOFA Used For?
- Perform Activity Using FOFA
    - Explore IoT, SCADA, and server exposure

🛠️ 6.3 Exploit Database (Exploit-DB)

- What is Exploit-DB?
- How Exploit-DB Is Used in IoT and OT Hacking
- Perform Activity Using Exploit-DB
    - Search vulnerabilities by CVE or product

⚠️ 6.4 Criminal IP

- What is Criminal IP?
- Features for OSINT and threat intel
- Use Cases: IoT exposure mapping, reputation checking

IoT And OT Hacking

The significant development of the paradigm of the Internet of Things (IoT) is contributing to the proliferation of devices in daily life. From smart homes to automated healthcare applications, IoT is ubiquitous. However, despite the potential of IoT to make our lives easier and more comfortable, we cannot underestimate its vulnerability to cyber-attacks. IoT devices lack basic security, which makes them prone to various cyber-attacks.

The objective of a hacker in exploiting IoT devices is to gain unauthorized access to users' devices and data. A hacker can use compromised IoT devices to build an army of botnets, which, in turn, is used to launch DDoS attacks.

Owing to a lack of security policies, smart devices are easy targets for hackers who can compromise these devices to spy on users' activities, misuse sensitive information (such as patients' health records, etc.), install ransomware to block access to the devices, monitor victims' activities using CCTV cameras, commit credit-card-related fraud, gain access to users' homes, or recruit the devices in an army of botnets to carry out DDoS attacks.

As an ethical hacker and penetration tester, you must have sound knowledge of hacking IoT and OT platforms using various tools and techniques. The labs in this module will provide you with real-time experience in performing footprinting and analyzing traffic between IoT and OT devices

Internet Of Things

🔰 1. What is IoT?

IoT (Internet of Things) refers to the network of physical objects (devices, vehicles, appliances, etc.) that are embedded with sensors, software, and connectivity to collect and exchange data over the internet.

🧠 2. Key Concepts

- **Smart Devices:** Devices that can collect and share data.
- **Sensors & Actuators:** Collect real-world data (e.g., temperature, motion).
- **Connectivity:** Communication via Wi-Fi, Bluetooth, Zigbee, LTE, etc.
- **Data Processing:** Local (Edge) or Cloud-based analysis.
- **Automation & Control:** Devices can perform tasks automatically based on data.

📐 3. Architecture of IoT

1. **Perception Layer (Device Layer):** Sensors, RFID, actuators.
2. **Network Layer:** Transfers data – Wi-Fi, 5G, Zigbee, etc.
3. **Processing Layer:** Data processing – cloud or edge.
4. **Application Layer:** Services for users (e.g., Smart Home app).

🔌 4. IoT Communication Protocols

a. Network Protocols

- **IPv6, 6LoWPAN:** IP for low-power devices
- **Wi-Fi / Ethernet / Cellular / LPWAN**

b. Messaging Protocols

- **MQTT (Message Queuing Telemetry Transport)**
- **CoAP (Constrained Application Protocol)**
- **AMQP (Advanced Message Queuing Protocol)**
- **HTTP/HTTPS (For REST APIs)**

🧪 5. IoT Applications

a. Consumer:

- Smart Home (lights, thermostats, locks)
- Wearables (fitness bands, smartwatches)

b. Industrial (IIoT):

- Predictive maintenance
- Smart factories

c. Healthcare:

- Remote patient monitoring
- Smart medical devices

d. Agriculture:

- Soil monitoring
- Automated irrigation

e. Smart Cities:

- Smart traffic lights
- Waste management systems

🛡️ 6. IoT Security

Security is one of the most critical parts of IoT due to the massive number of connected devices.

Common Threats:

- Weak/default credentials
- Unpatched firmware
- Insecure communication

Security Measures:

- Device authentication
- Secure boot and firmware
- Data encryption
- Regular updates

🧰 7. Tools and Platforms

a. Hardware Platforms:

- Raspberry Pi
- Arduino
- ESP32
- Intel Edison

b. Software Platforms:

- **Google Cloud IoT Core**
- **AWS IoT**
- **Microsoft Azure IoT Hub**
- **ThingsBoard**
- **Node-RED** (Visual programming tool)

🌐 8. Cloud & Edge Computing in IoT

- **Cloud Computing:** Centralized data storage and analytics.
- **Edge Computing:** Data is processed near the source (low latency, high speed).

📡 9. IoT and AI

- Predictive analytics
- Smart automation
- Anomaly detection using ML

📊 10. Data Flow in IoT

Device → Gateway → Cloud Platform → Application

🔧 11. IoT Protocol Stack

Layer	Protocol Examples
Application	MQTT, CoAP, HTTP
Transport	TCP, UDP
Network	IPv6, 6LoWPAN
Data Link	IEEE 802.15.4, LoRa
Physical	Bluetooth, Wi-Fi, Zigbee, LTE

💼 12. IoT Career Scope

Roles:

- IoT Developer
- Embedded Systems Engineer
- IoT Security Analyst
- Firmware Developer
- IoT Data Analyst

Skills Required:

- Embedded C/C++
- Python/JavaScript for scripting

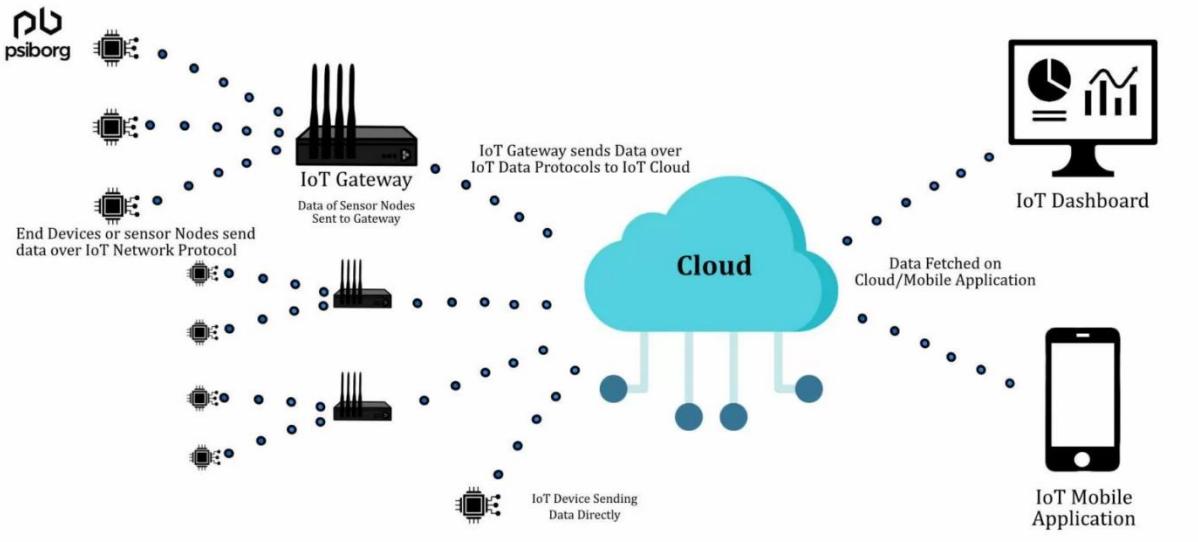
- Networking fundamentals
- Cloud platforms
- Security concepts

🧪 13. Practical Projects to Learn

- Smart Home Automation with NodeMCU
- IoT Temperature & Humidity Monitor
- IoT-based Smart Dustbin
- GPS Tracker with GSM module
- Smart Street Lighting System

🌐 IoT Communication Model – Explained with Diagram

The **IoT Communication Model** defines **how devices in an IoT system communicate** with each other, cloud platforms, and users. It involves data collection, transmission, processing, and action.



IoT Communication Model

🔟 OWASP Top 10 IoT Threats

No.	Threat Category	Description
1	🔐 **Weak, Guessable, or Hardcoded Passwords**	Use of default or easy-to-guess credentials, or hardcoded passwords in firmware.
2	🛑 **Insecure Network Services**	Services running on the device that are vulnerable, exposed to the internet, or not needed.
3	📡 **Insecure Ecosystem Interfaces**	Insecure web, backend, cloud, or mobile interfaces (e.g., weak authentication, lack of encryption).
4	🧠 **Lack of Secure Update Mechanism**	Insecure or no firmware/software updates, no signing or verification process.
5	🕳️ **Use of Insecure or Outdated Components**	Using libraries or components with known vulnerabilities or unsupported versions.
6	📜 **Insufficient Privacy Protection**	Poor handling of personal data, such as logging without consent, or insecure data storage.
7	🦹 **Insecure Data Transfer and Storage**	Data is sent or stored without encryption, or in an easily accessible format.
8	🔓 **Lack of Device Management**	No way to securely manage, update, or monitor devices once deployed.
9	👤 **Insecure Default Settings**	Devices ship with insecure settings (e.g., open ports, remote access enabled).
10	🧪 **Lack of Physical Hardening**	Devices can be physically tampered with, allowing firmware extraction, USB access, or debugging.

📊 Comprehensive IoT Security Threats Table

No.	Threat Name	Description
1	Weak/Hardcoded Passwords	Default or hardcoded credentials, often unchangeable.
2	Insecure Network Services	Services exposed unnecessarily, exploitable remotely (e.g., Telnet, FTP).
3	Insecure Web Interfaces	Web UIs vulnerable to XSS, CSRF, SQLi, and weak login protections.
4	Lack of Secure Update Mechanism	No firmware validation or OTA update capability.
5	Use of Insecure/Outdated Components	Libraries and OS versions with known CVEs.
6	Insufficient Privacy Protection	Data collected without consent or poor data protection.
7	Insecure Data Storage and Transmission	Plaintext storage or lack of TLS/HTTPS encryption.
8	Lack of Device Management	No control over deployed device settings, patches, or status.
9	Insecure Default Settings	Open ports, admin panels, or debug interfaces enabled by default.
10	Lack of Physical Hardening	Easy physical access to device internals (e.g., JTAG, UART).
11	Insecure APIs	No rate limits, exposed endpoints, missing auth tokens.
12	Poor Authorization Controls	No role-based access control or privilege separation.
13	Botnet Infections (e.g., Mirai)	Devices hijacked for DDoS or mining attacks.

No.	Threat Name	Description
14	Bluetooth/NFC/RF Vulnerabilities	BLE spoofing, jamming, or unauthorized access via proximity protocols.
15	Supply Chain Attacks	Vulnerable firmware loaded during manufacturing or shipping.
16	Denial of Service (DoS)	Devices overwhelmed via traffic floods or malformed packets.
17	Remote Code Execution (RCE)	Exploits like buffer overflow enabling attacker control.
18	Lack of Logging and Monitoring	No audit trail or alerts for suspicious activity.
19	Improper Device Decommissioning	Devices sold/disposed with intact sensitive data.
20	Cloud/Mobile App Vulnerabilities	Insecure IoT companion apps or cloud APIs (e.g., token leakage).
21	Eavesdropping & Traffic Analysis	Attackers intercept MQTT/CoAP or analyze patterns for intel.
22	IT/OT Network Integration Risks	IoT exposure allows attackers to pivot to critical infrastructure (ICS/SCADA).
23	Geo-Tracking & Location Leaks	GPS data leakage from IoT tracking systems or wearables.

Operational Technology

Operational Technology (OT) refers to the **hardware and software systems** that monitor and control **physical devices, processes, and events** in industrial environments. These devices are essential in industries like **manufacturing, energy, water, oil & gas, and transportation**.

OT vs IT – Key Difference

Category	OT (Operational Technology)	IT (Information Technology)
Focus	Physical process control	Data processing, storage, communication
Devices	PLCs, RTUs, HMIs, SCADA	Computers, servers, routers, switches
Availability	High availability is critical	Security and confidentiality are key
Protocols	MODBUS, DNP3, PROFIBUS, OPC-UA	TCP/IP, HTTP, FTP
Updates	Rarely updated, stable systems	Frequent updates and patches

---

Common OT Devices and Their Functions

Device	Full Form / Type	Description
PLC	Programmable Logic Controller	Brain of the industrial system; executes logic in real time
RTU	Remote Terminal Unit	Sends sensor data to SCADA over long distances

Device	Full Form / Type	Description
HMI	Human Machine Interface	Interface for operators to monitor and control industrial processes
DCS	Distributed Control System	Control system with local processing units; often used in refineries
SCADA	Supervisory Control and Data Acquisition	Software/hardware to control and monitor OT devices across locations
Sensors	(Analog/Digital)	Detect temperature, pressure, vibration, etc.
Actuators	(Motors/Valves)	Take physical action (open valve, rotate motor)
IED	Intelligent Electronic Devices	Used in electrical substations for protection and monitoring
Data Historian	—	Stores time-series data from OT devices for long-term analysis
Industrial Gateways	—	Converts OT protocols to IT-compatible ones (e.g., MODBUS to MQTT)

🔧 **OT Architecture Overview (Simplified)**

```
+-----------------------------+
|      Enterprise IT Layer    | <-- ERP, Cloud, Analytics
+-----------------------------+
          ↑
+-----------------------------+
|       SCADA / DCS Layer     | <-- Control and Supervision
```

```
+-----------------------------+
          ↑
+-----------------------------+
|     Control Devices Layer     | <-- PLCs, RTUs, IEDs
+-----------------------------+
          ↑
+-----------------------------+
|   Field Devices (Sensors, Actuators)  |
+-----------------------------+
```

Common OT Protocols

Protocol	Purpose	Transport
MODBUS	Communication between PLCs	TCP/RS-485
DNP3	Substation automation	Serial/TCP
OPC-UA	Interoperability and standardization	TCP/IP
PROFIBUS	Factory automation	Fieldbus
EtherCAT	Real-time communication	Ethernet
BACnet	Building automation	Ethernet/IP

Security Challenges in OT Devices

Risk	Description
Outdated Systems	Many OT systems run legacy Windows or embedded OS
No Encryption	Protocols like MODBUS are plaintext
No Authentication	Devices trust any command sent to them

Risk	Description
Physical Access Risks	Devices may be in remote, unguarded locations
Lack of Patching	Downtime is unacceptable, so updates are avoided
Air-Gapped Assumption	Many believe OT networks are isolated (not true anymore)

🔐 Securing OT Devices – Best Practices

1. **Network Segmentation (IT/OT separation)**
2. **Firewall and Deep Packet Inspection (DPI)**
3. **Regular Vulnerability Assessment**
4. **Access Control & Role-Based Access**
5. **Disable Unused Services/Ports**
6. **Patch Management Plan**
7. **Use of Security Gateways**
8. **Monitor with SIEM/SOC tools**
9. **Asset Inventory & Logging**
10. **Zero Trust Architecture for OT**

🏭 Use Cases of OT Devices by Industry

Industry	OT Devices Used	Purpose
Manufacturing	PLC, HMI, SCADA	Assembly line automation
Energy (Grid)	RTU, IED, DNP3, SCADA	Grid monitoring, load control
Oil & Gas	DCS, RTU, Sensors	Pipeline monitoring and safety

Industry	OT Devices Used	Purpose
Water Management	SCADA, Sensors, Actuators	Water treatment and flow regulation
Transportation	PLC, HMI	Train control systems, signaling

⚠️ Top Challenges of OT (Operational Technology)

Category	Challenge	Explanation
1. Legacy Systems	Outdated hardware and software	Many OT systems are 10–20+ years old and lack modern security features.
2. Security Risks	Vulnerable to cyberattacks	Weak authentication, no encryption, and exposed protocols like MODBUS.
3. Lack of Updates	Rare or no patching	Systems must run continuously — making downtime for patching difficult.
4. IT-OT Integration	Convergence creates new attack surfaces	Blending IT networks with OT exposes sensitive OT devices to internet threats.
5. Limited Visibility	Poor monitoring and logging	Many OT systems don't log events or support real-time alerts.
6. Vendor Dependency	Closed/proprietary ecosystems	Updating or customizing systems may require vendor assistance or approval.

Category	Challenge	Explanation
7. Insecure Protocols	Legacy fieldbus protocols (e.g., MODBUS, DNP3)	These are unauthenticated and unencrypted, easy targets for interception.
8. Physical Access	Devices in remote/unsafe locations	Attackers may tamper with devices physically (e.g., USB, JTAG, UART access).
9. Lack of Security Awareness	Operational teams lack cybersecurity training	OT staff focus on uptime, not cyber hygiene — risky default settings remain.
10. Compliance Pressure	Increasing regulatory requirements	Standards like NERC-CIP, IEC 62443, or ISO 27019 demand strict controls.
11. Poor Asset Inventory	Incomplete device and software listing	Hard to secure what you don't know exists in your network.
12. No Role-Based Access	Same login for all operators	No separation of duties; increases risk of misuse or insider threats.

🧨 Common OT Vulnerabilities

No.	Vulnerability	Description
1️⃣	**Default or Hardcoded Credentials**	Factory-set usernames/passwords often never changed (e.g., admin/admin).
2️⃣	**Lack of Authentication/Authorization**	Devices like PLCs/RTUs trust any command — no access control or session auth.

No.	Vulnerability	Description
3	**Insecure Protocols (MODBUS, DNP3, etc.)**	These protocols send data in plaintext and lack authentication.
4	**No Data Encryption**	Data in transit is not encrypted (no TLS, SSL, or VPN).
5	**Buffer Overflow Vulnerabilities**	Poor input validation in firmware allows RCE (Remote Code Execution).
6	**Outdated Firmware/OS**	Devices run old, unsupported systems with known CVEs (e.g., Windows XP, VxWorks).
7	**Lack of Secure Boot**	Firmware tampering possible if there's no integrity check during boot.
8	**Physical Access Ports Open**	JTAG, UART, USB ports accessible and unprotected in remote/plant devices.
9	**No Logging or Monitoring**	No visibility of changes, tampering, or malicious activity.
10	**Improper Input Validation**	Malicious commands accepted by PLCs/RTUs due to poor input sanitization.
11	**Remote Access Services Enabled**	Telnet, SSH, RDP, or VNC exposed to external networks without proper security.

No.	Vulnerability	Description
1 2	**Lack of Network Segmentation**	OT and IT networks mixed; attack on one compromises both.
1 3	**Weak or No Patch Management**	Vendors may not provide regular updates, or patching causes downtime.
1 4	**Firmware Extraction or Tampering**	Attackers dump firmware using tools like Binwalk, JTAGulator, etc.
1 5	**Misconfigured Services/Devices**	Open services, unnecessary features, and default configurations left unchanged.

Practical IoT & OT Hacking: Simulating Attacks with ICSim

ICSim (Industrial Control System Simulation) is an open-source training environment created by **Digital Bond** to simulate **ICS/SCADA environments**, specifically **CAN Bus** systems used in industrial and automotive settings.

ICSim emulates a simple industrial setup (like a water plant or traffic light control system) using CAN protocol, allowing ethical hackers to test **OT/ICS vulnerabilities** safely.

🎯 2. Objectives of Using ICSim

- Understand **Industrial Control Systems (ICS)** and **Operational Technology (OT)**.
- Learn about **CAN Bus communication**.
- Practice **passive and active attacks** on ICS systems.
- Train on **realistic but virtual environments** without causing damage.
- Explore **packet sniffing**, **replay attacks**, and **message injection**.

🛠️ 3. Key Features

1. **Simulates a Real ICS System**
    - Shows how industrial systems like traffic lights work using a virtual setup.
2. **Uses CAN Protocol**
    - Based on the same communication used in cars and factories (CAN Bus).
3. **No Physical Devices Needed**
    - Runs completely on your computer using virtual interfaces.

4. **Hands-on Hacking Practice**
    - Lets you try sniffing, replaying, and injecting fake messages.
5. **Includes GUI (Graphical Interface)**
    - Visual simulation of traffic lights and controls – easy to see what's happening.
6. **Safe Learning Environment**
    - You can perform attacks without harming any real system.
7. **Works with Real CAN Tools**
    - Uses tools like candump, cansend, canplayer to simulate real scenarios.
8. **Open Source & Free**
    - Available for everyone to download, use, and modify.

🧰 4. Components of ICSim

- **icsim** – Main GUI traffic light simulator
- **candump, cansniffer, canplayer** – CAN utilities (from **can-utils**)
- **vcan** – Virtual CAN interface (no hardware required)
- **controls.py** – Send fake CAN messages (for attacks)
- **can-utils** – Open-source set of CAN network tools

🔒 5. ICSim for IoT/OT Hacking Learning

- Realistic simulation of industrial/automotive networks.
- Practice **red teaming** on OT environments.
- Learn **protocol weaknesses** of legacy ICS.

---

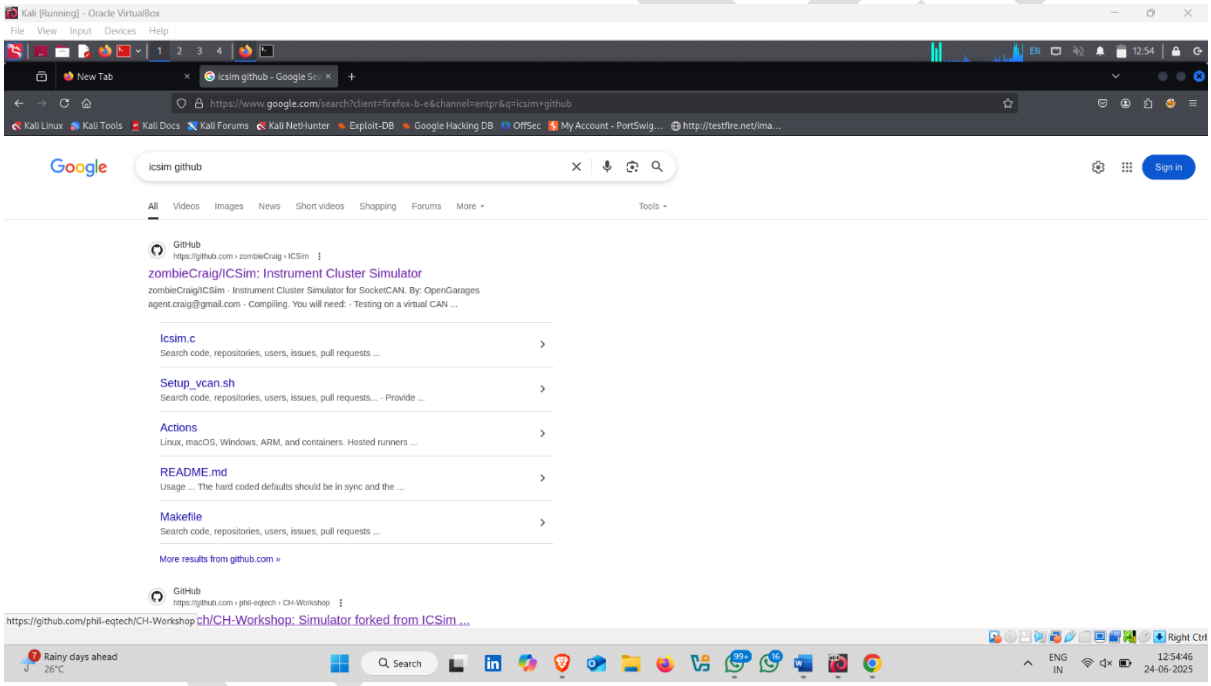
6. Learning Outcomes

By the end of the module using ICSim, learners should be able to:

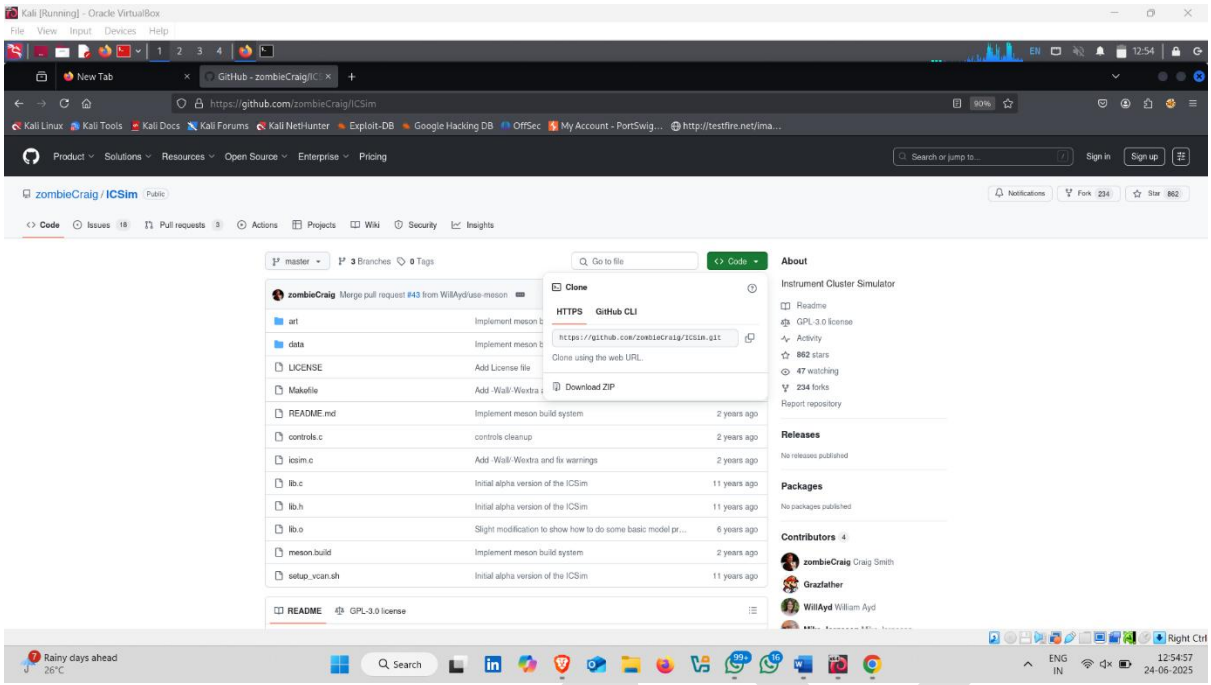
- Understand basic ICS/SCADA architecture.
- Demonstrate packet sniffing and injection on CAN bus.
- Explain and simulate OT threats in critical systems.
- Perform ethical hacking on industrial protocols.

How to use it -:

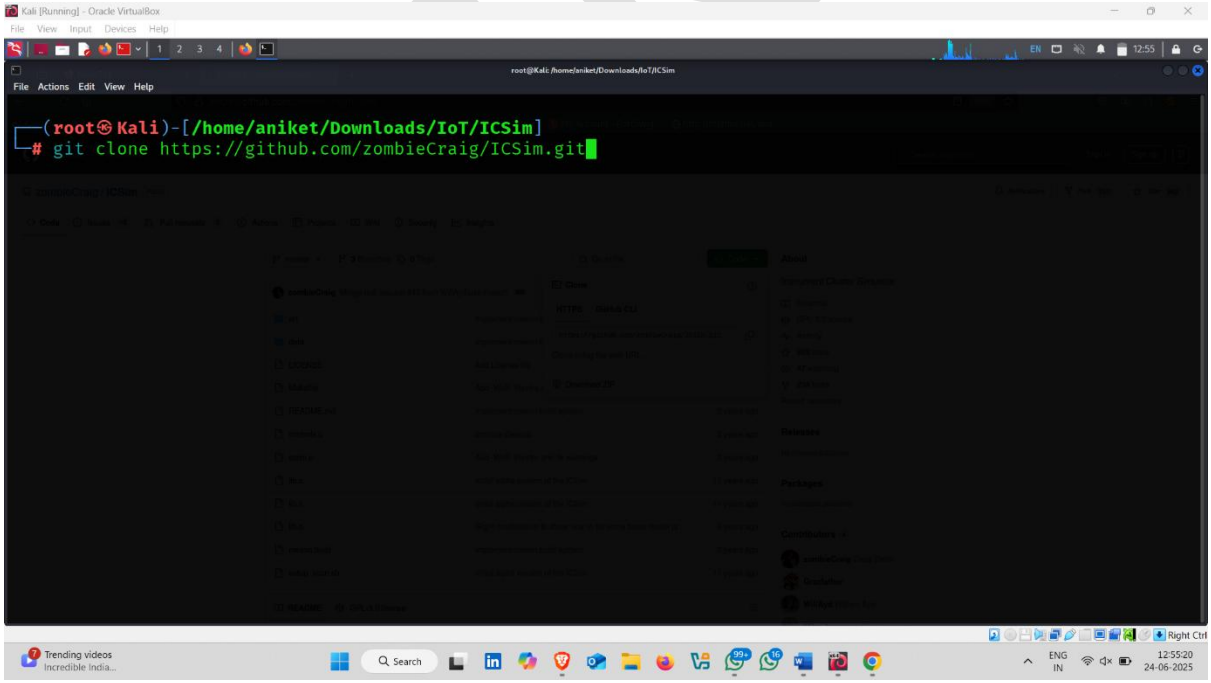
- Open kali linux/parrot Os and seach **Icsim git hub.**
- Click on first website



- Click on **Green code** button and **copy HTTPS Url**



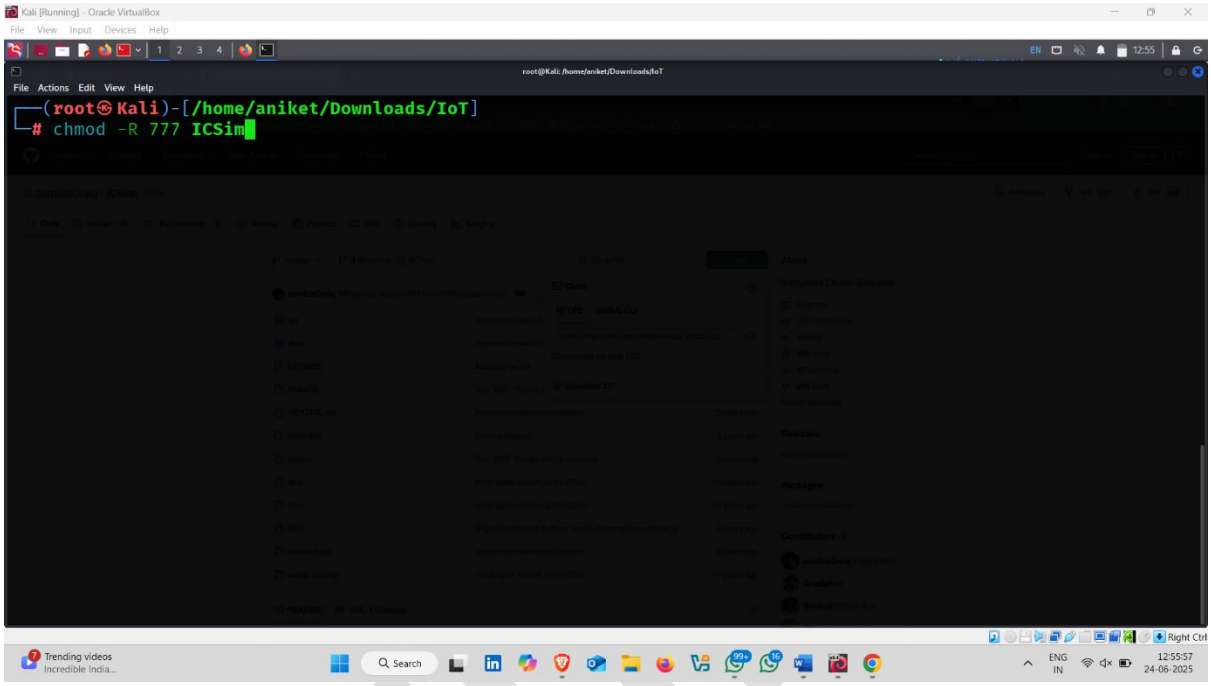
- Open kali linux terminal and **paste HTTPS url with git clone**



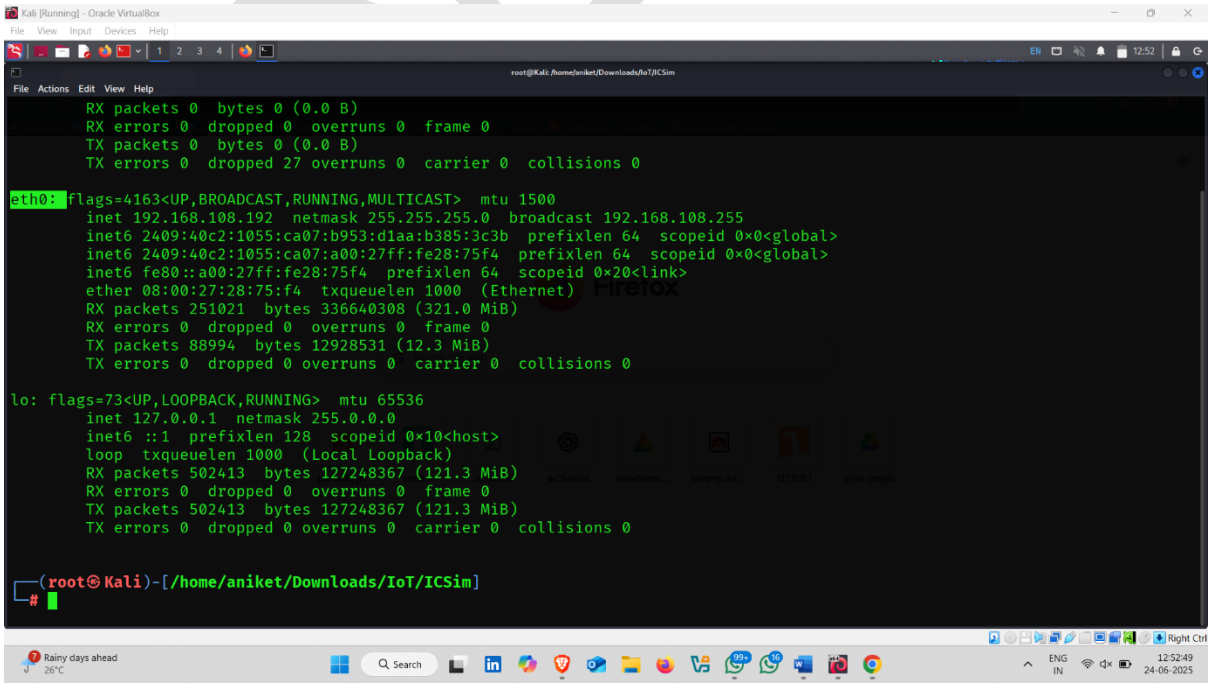
- After downloading Icsim change folder permission

Command -: chmod -R 777 ICSim

Explaination-: Gives **full permission** to **everyone** for the **ICSim folder** and **everything inside it**.



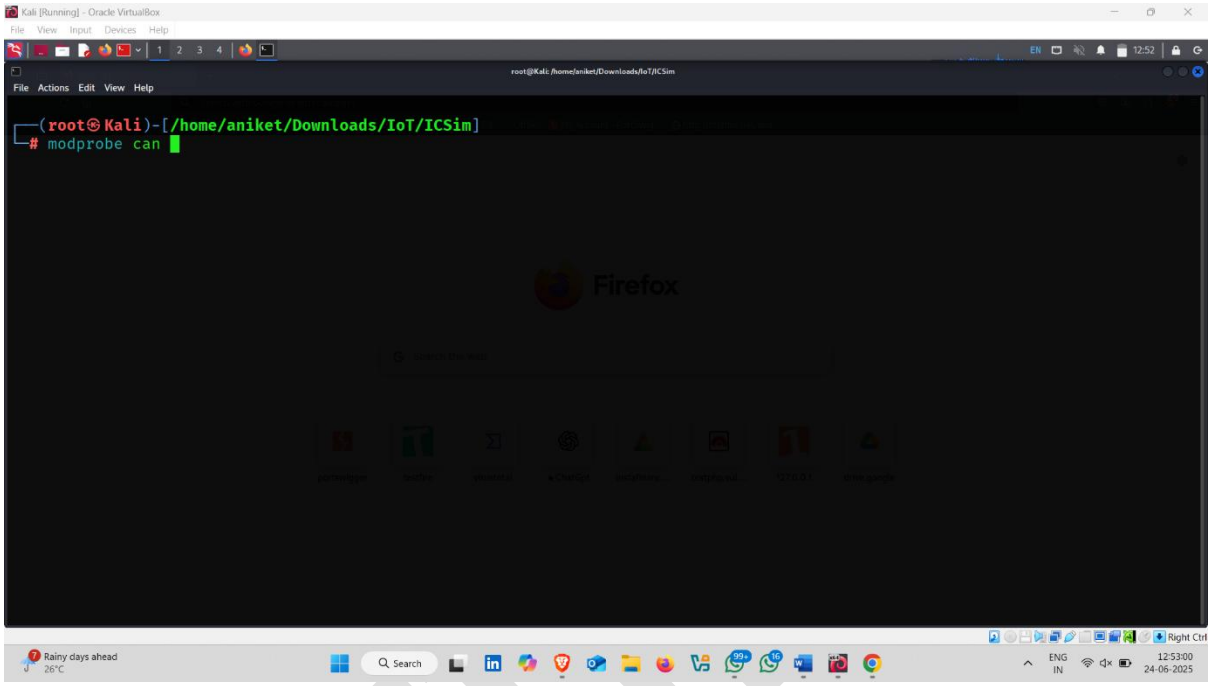
- Now use **ifconfig to check network interface**



- Now go to ICSim Directory and type following command

Command -: modprobe can

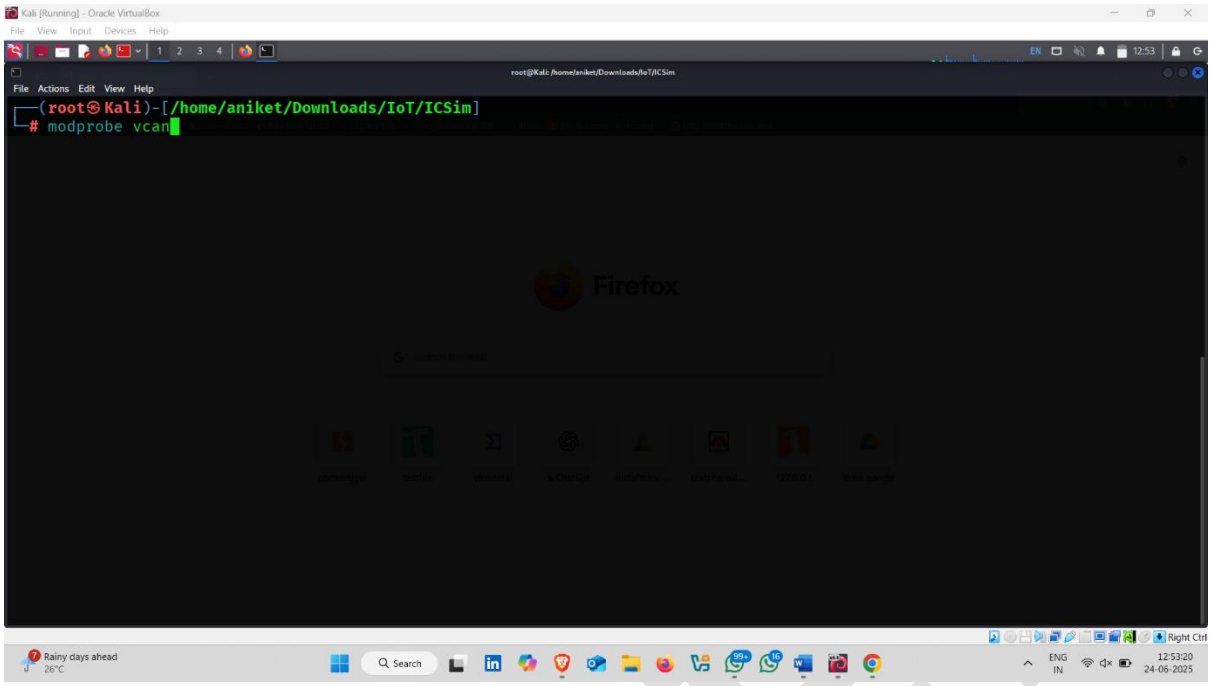
Explanation -: This command **loads the CAN (Controller Area Network) kernel module** in Linux.



- Now type following command

Command -: modprobe vcan

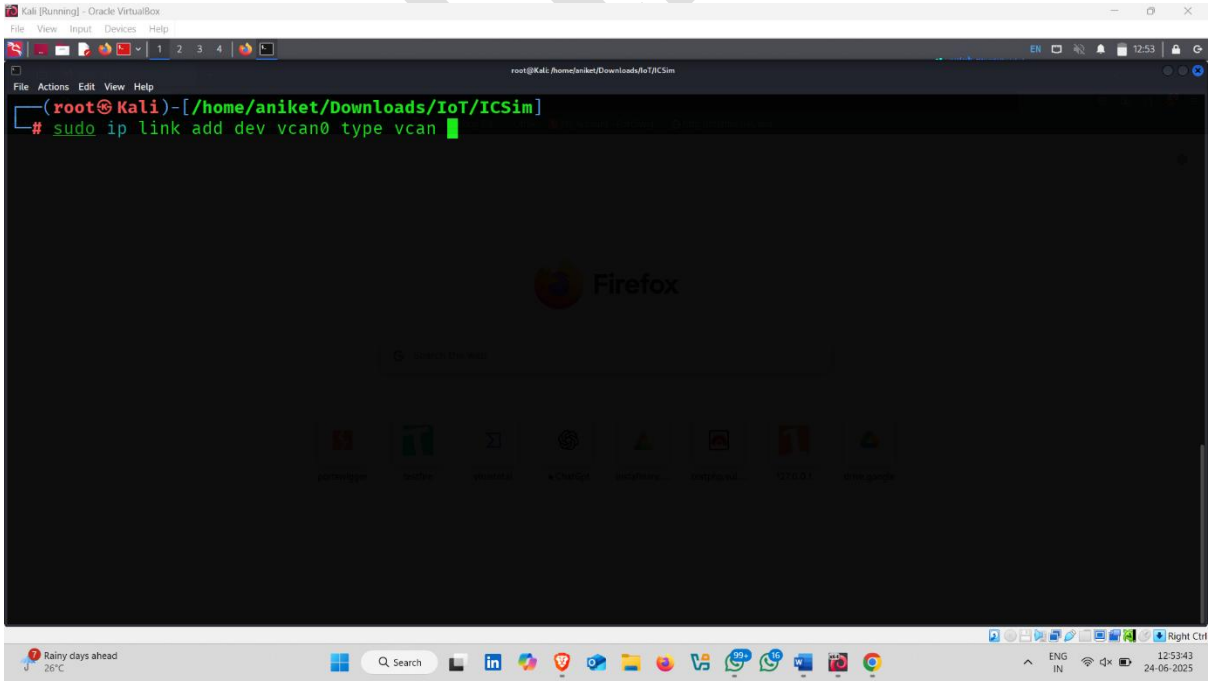
Explaination -: This command **loads the "vcan" (Virtual CAN) module** into the Linux kernel.



- Next command

Command -: sudo ip link add dev vcan0 type vcan

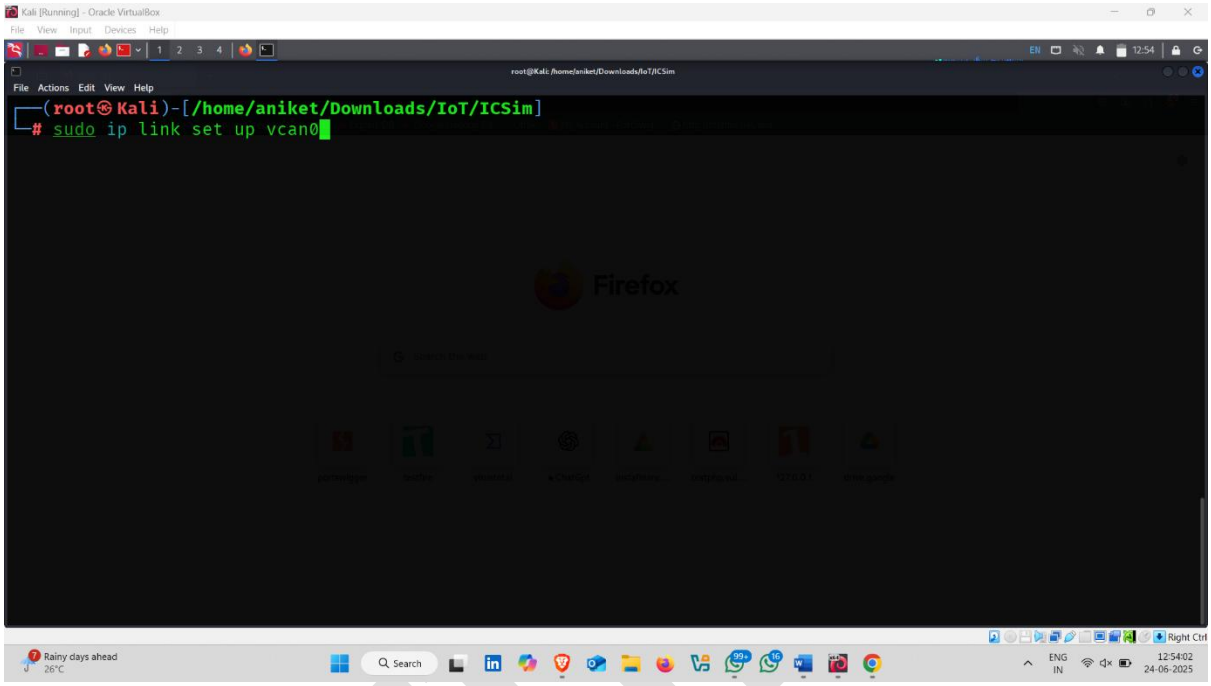
Explanation -: This command **creates a virtual CAN network interface** called vcan0.



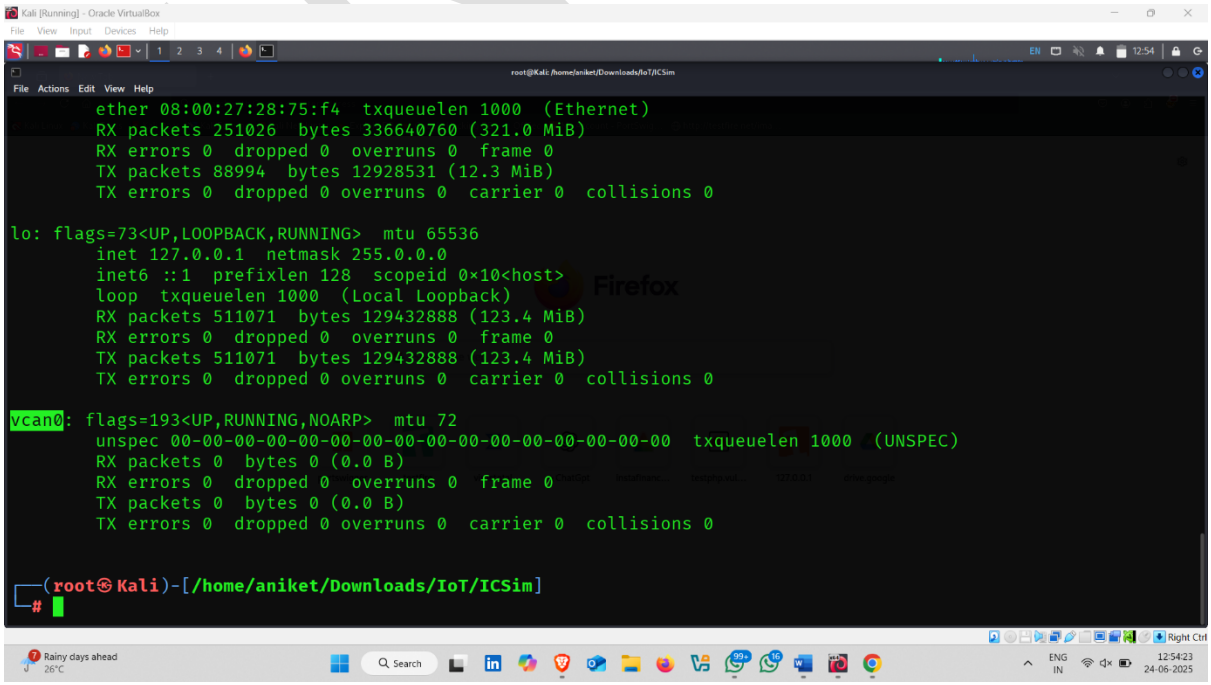
- Next command

Command -: sudo ip link set up vcan0

Explanation -: This command **activates or starts the virtual CAN interface** named vcan0.



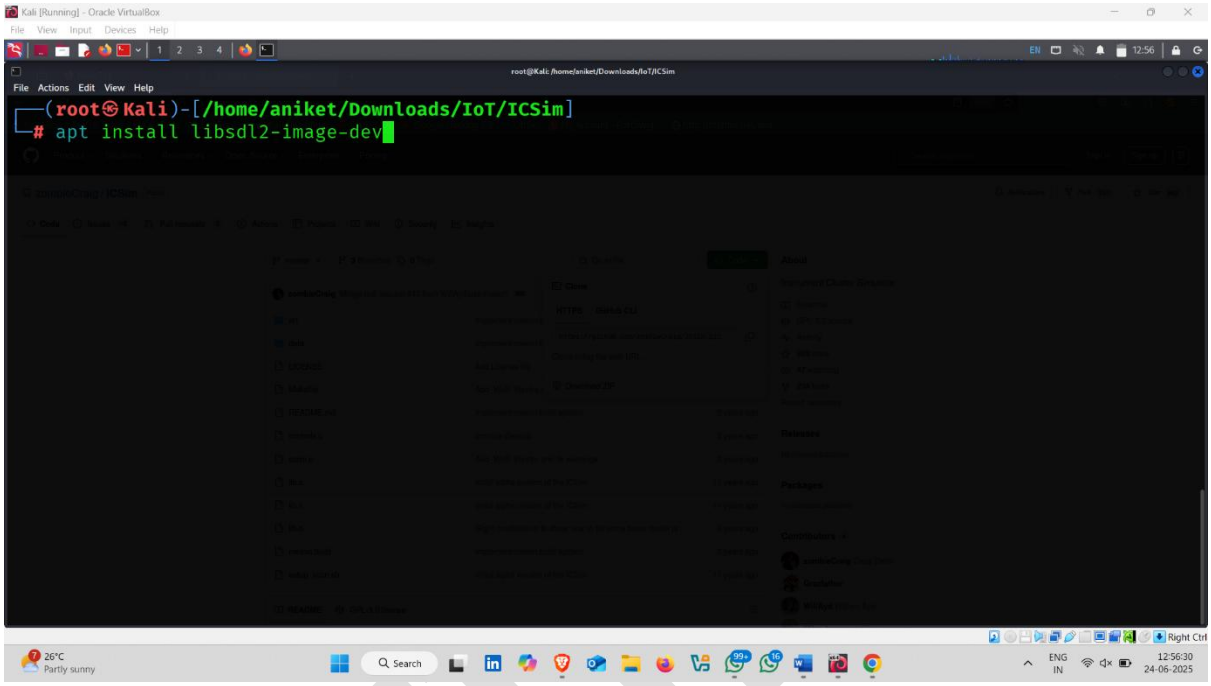
- Now type ifconfig to check our virtual network create or not
- Virtual interface created



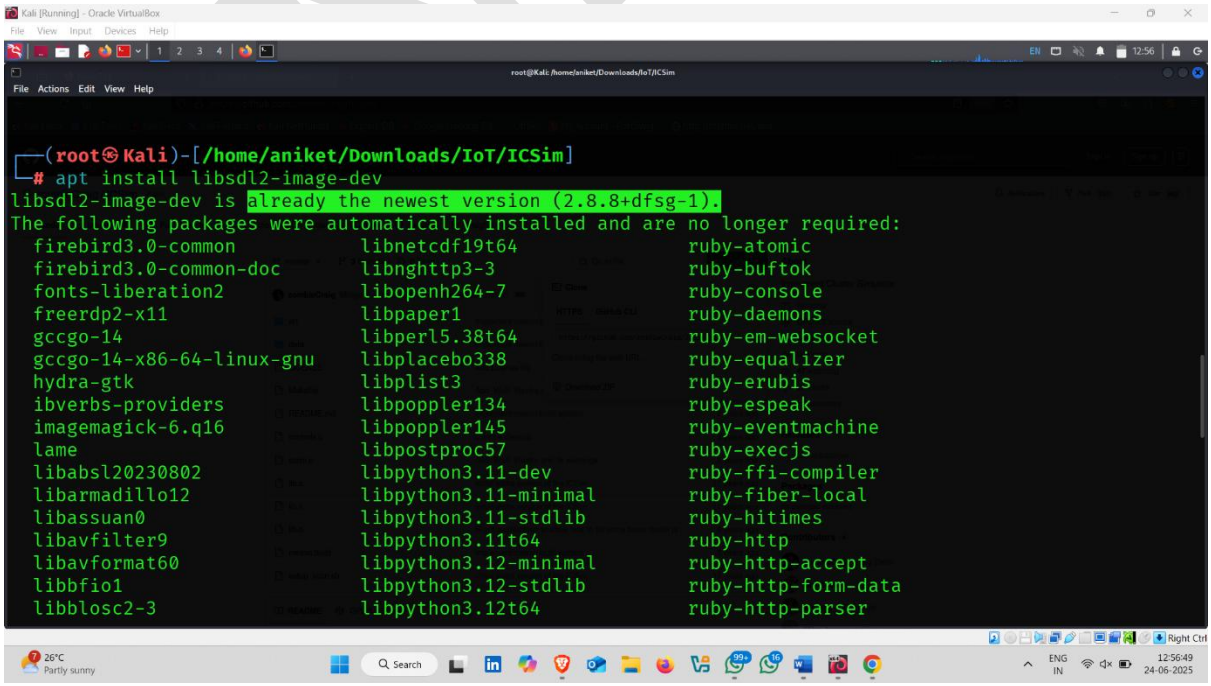
- Now type next command

Command-: apt install libsdl2-image-dev

Explanation -: his command **installs the SDL2 image development library** on your Linux system.



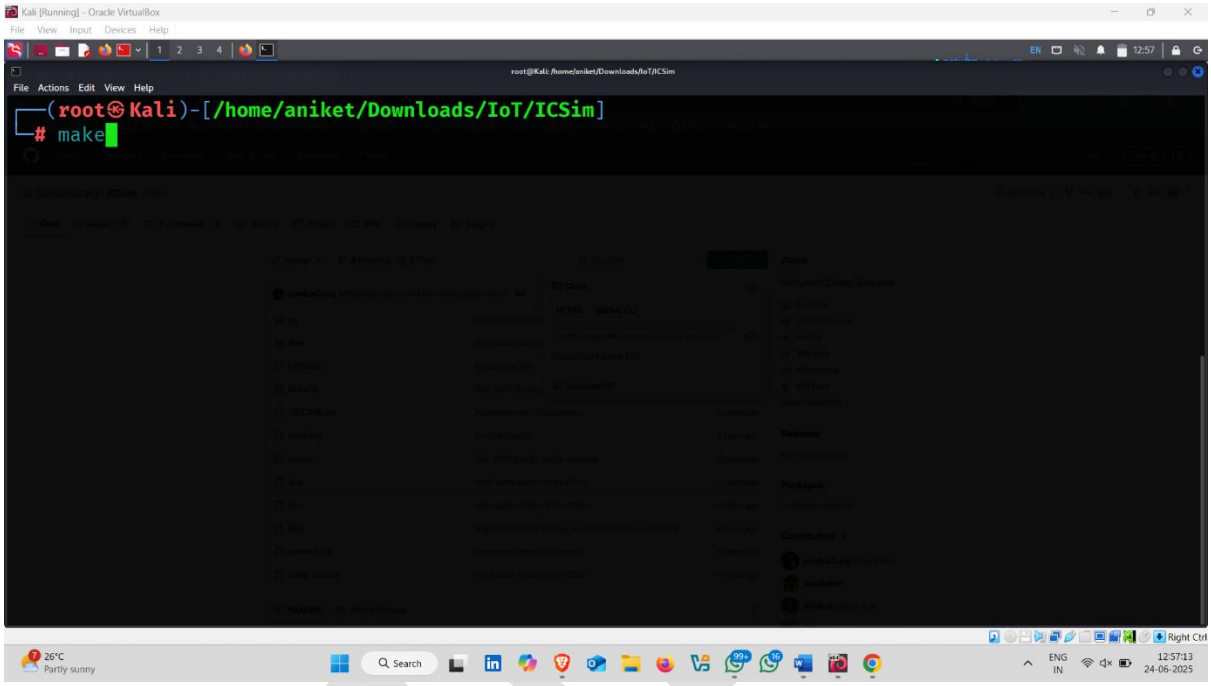
- I have already newest version of SDL2 image



- Next command

Command -: make

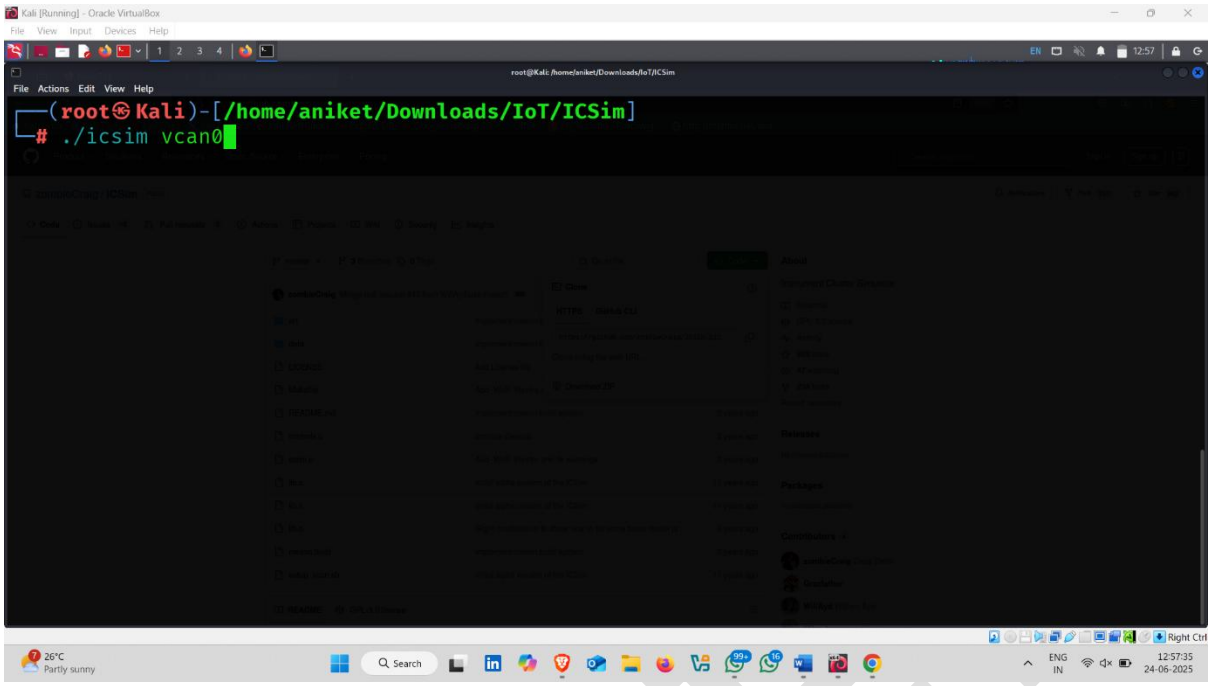
Explanation -: The make command **compiles source code** into an executable program by reading instructions from a file called Makefile.



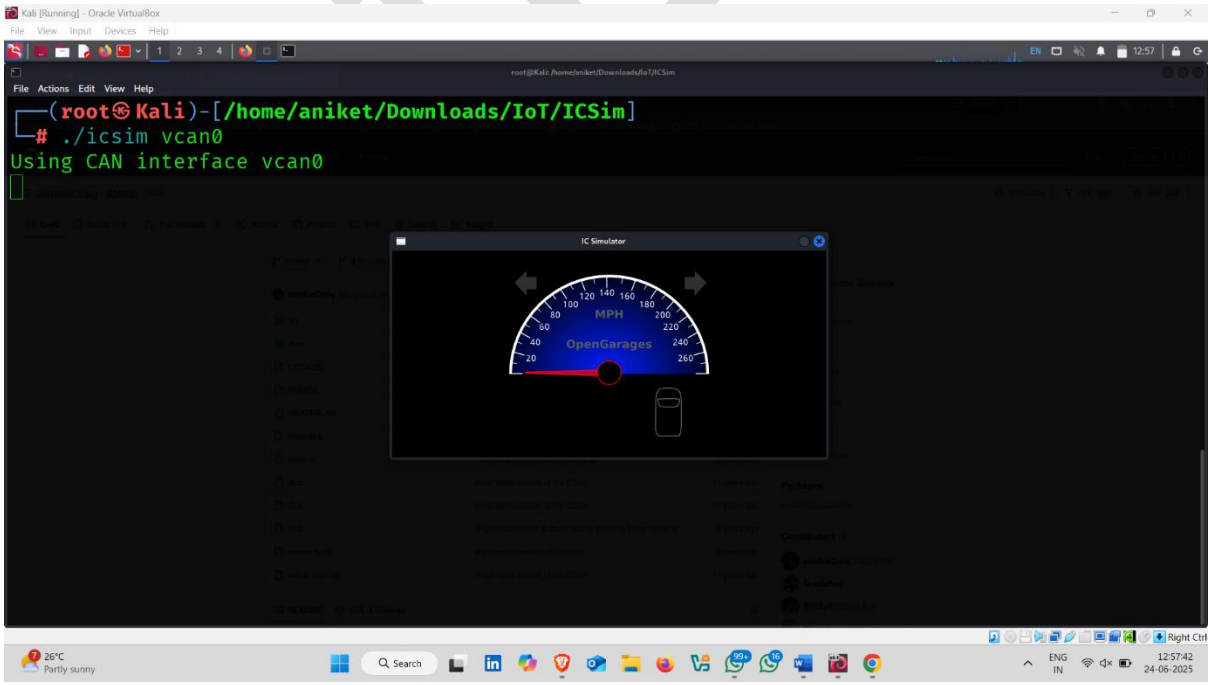
- Next command

Command -: ./ICSim vcan 0

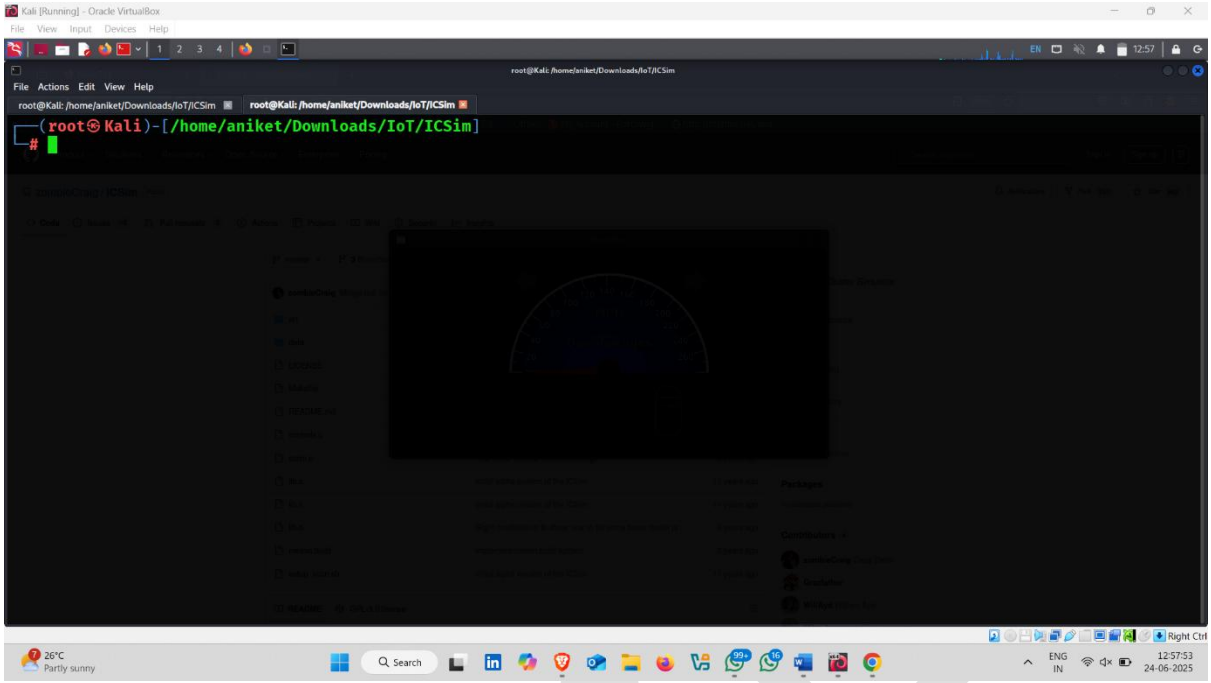
Explanation-: is used to **start the ICSim (Instrument Cluster Simulator)** and connect it to a virtual CAN (Controller Area Network) interface named vcan0.



- This shows you successfully ran the **ICSim simulator** using the virtual CAN interface vcan0.
- The message Using CAN interface vcan0 confirms that ICSim is connected and listening on that virtual network.



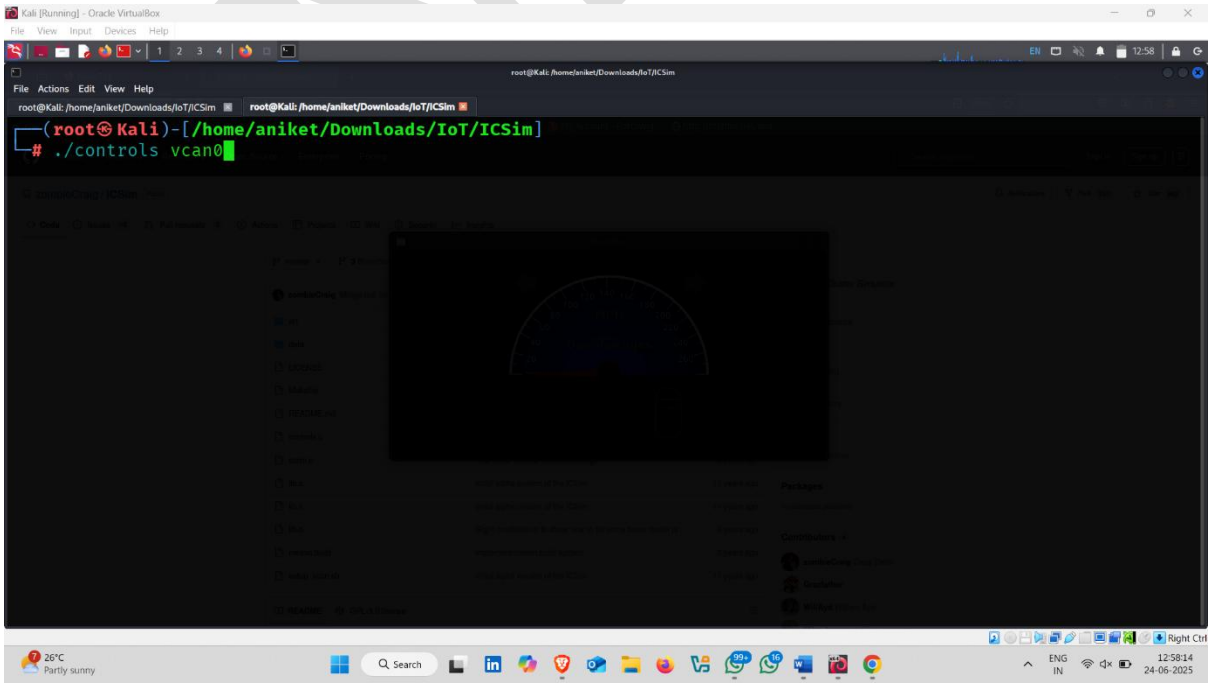
- Now open new terminal



- Type following command

Command -: ./ controls vcan0

Explanation-: This command **launches the ICSim controller interface** and connects it to the **virtual CAN interface** vcan0.



- You ran the controller interface for ICSim using the vcan0 virtual CAN network.
- The **"No joysticks connected"** message is just a warning — it's fine! It just means you're using a keyboard, not a game controller.

🔁 **How it Works:**

- When you press a key (e.g., **Up Arrow**), it sends a **CAN message over vcan0**.
- The **ICSim dashboard** (previous window) receives this and **updates the speedometer or lights** accordingly.

🎮 **Controller Window (CANBus Control Panel) – Simple Explanation**

This window is a **virtual controller** that lets you control the ICSim simulator using your **keyboard**. It sends commands as **CAN messages** through the virtual network (vcan0).

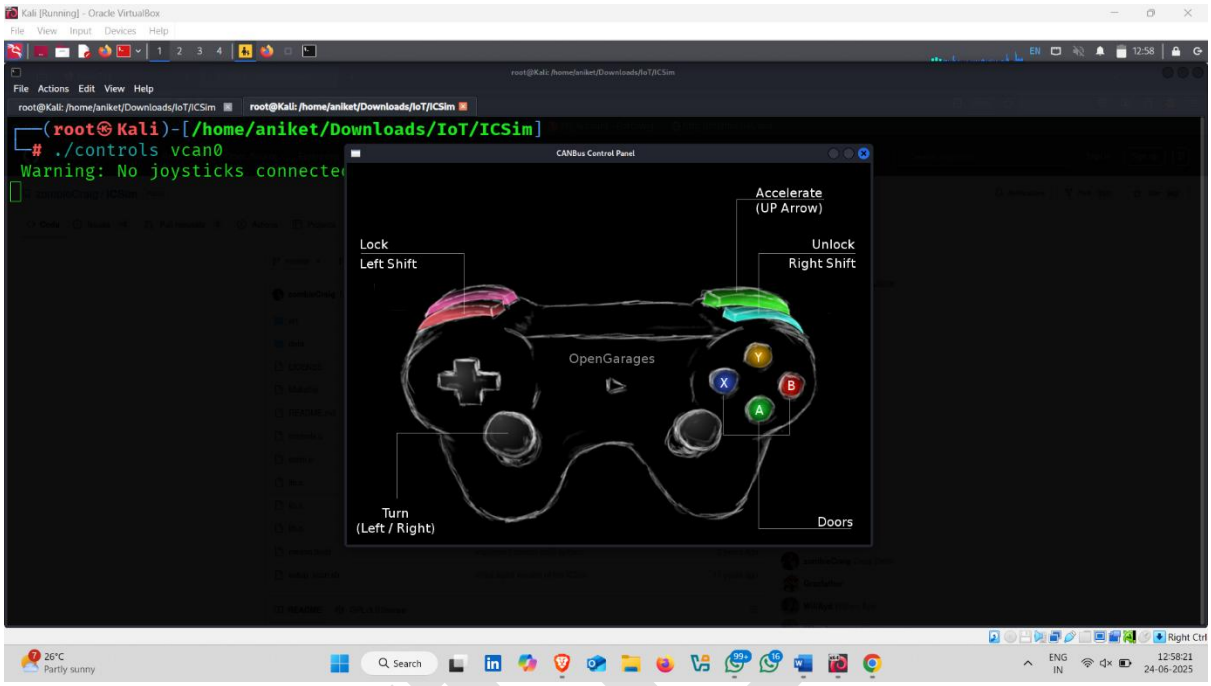
You can press different keys on your keyboard to control the vehicle:

- Press the **Up Arrow** key to **accelerate** (increase the speed).
- Use the **Left Arrow** or **Right Arrow** keys to **turn left or right**, which simulates the **blinkers**.
- Press **Left Shift** to **lock the doors**.
- Press **Right Shift** to **unlock the doors**.
- Press keys like **A**, **B**, **X**, or **Y** to **control the doors** (open or close them).

As you press these keys, the ICSim dashboard will react — the speed will go up, lights will blink, and door status may change.

✅ In short:

The controller window helps you **send commands to the ICSim dashboard** just like you're **driving or controlling a smart car** — all from your keyboard.



- Now open another terminal
- Type following command on new terminal

Command -: cansniffer vcan0

Explanation -: This command **sniffs (monitors) live CAN messages** on the vcan0 interface and shows them in **real time**.

🧱 What happens when you run it:

- You'll see **CAN IDs** and **data bytes** changing on the screen.
- It **updates automatically** when new CAN messages are sent (e.g., from controls).

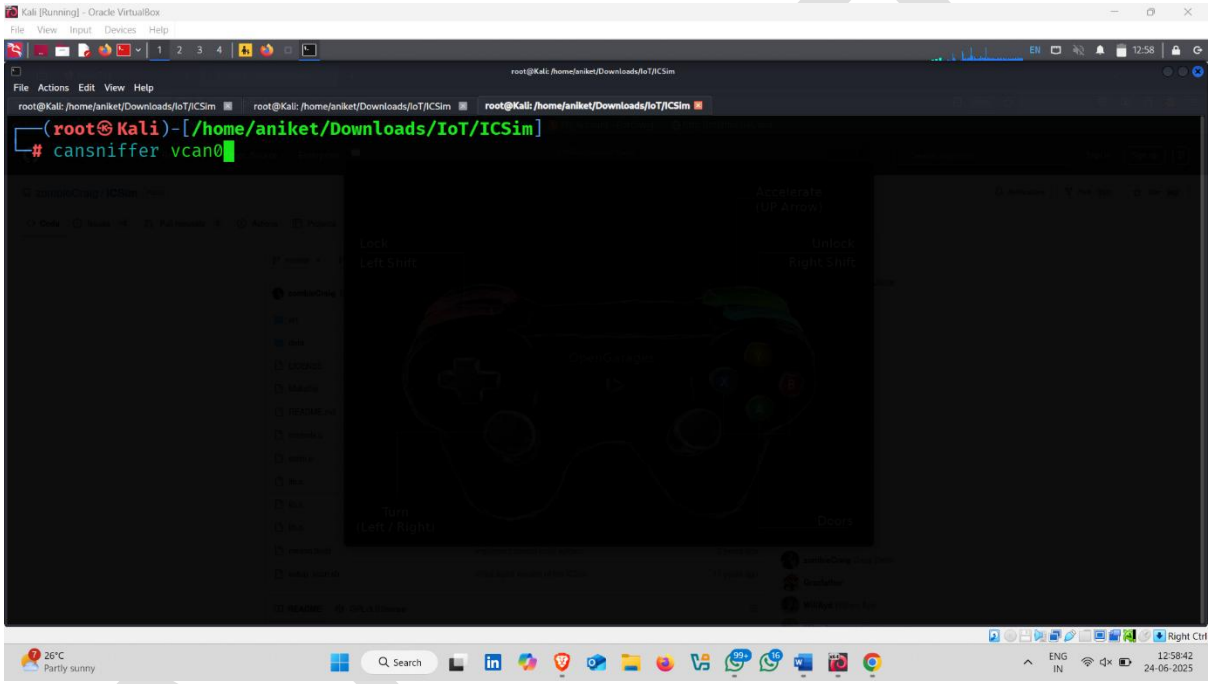
🧪 Why it's useful:

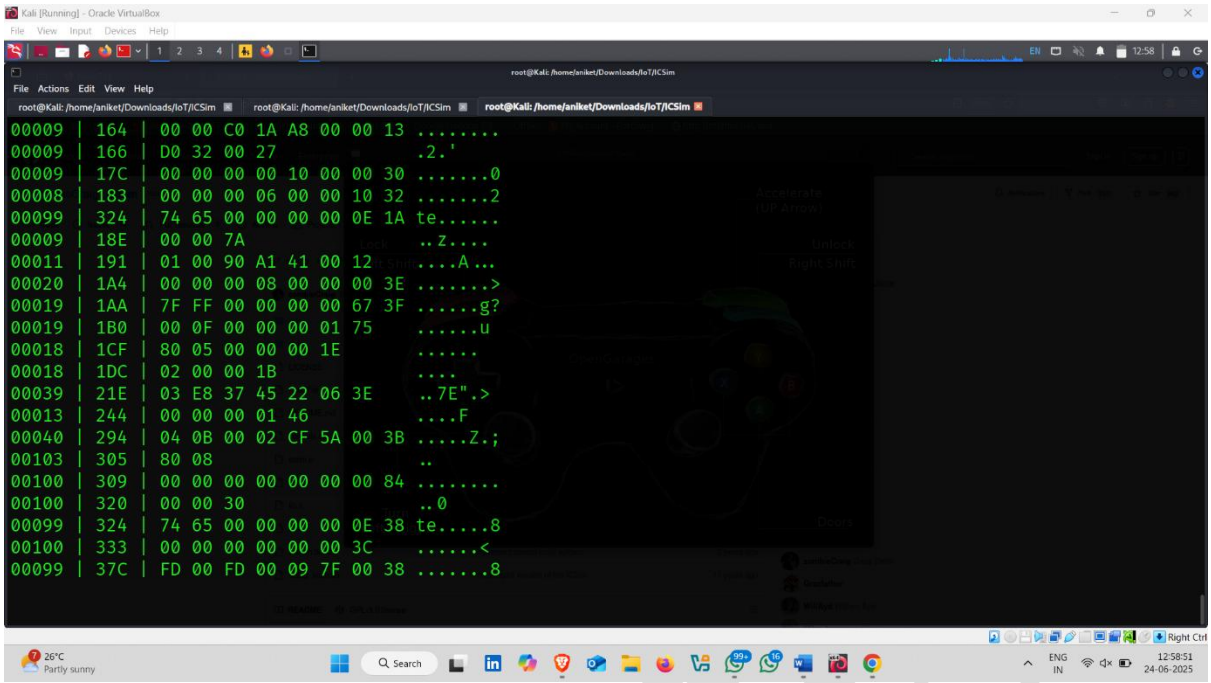
- Helps you **analyze which CAN IDs are active**

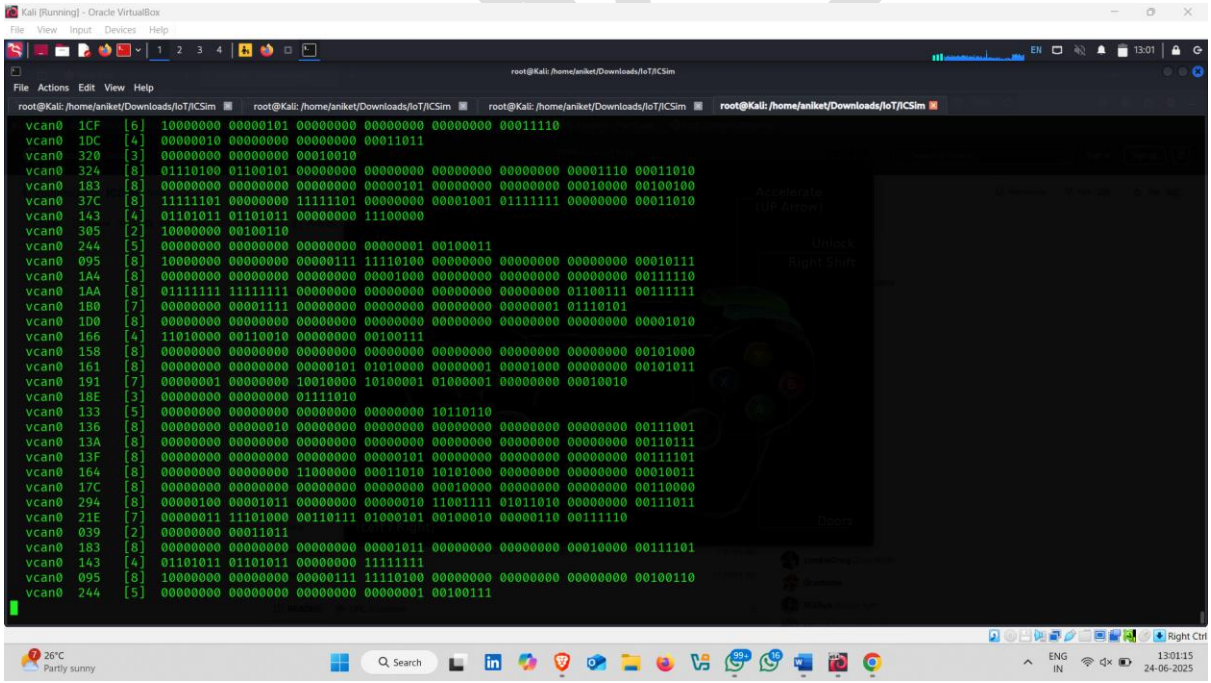
- Useful for **reverse engineering** and identifying which message controls what
- Can help you spot messages for **speed**, **blinkers**, **locks**, etc.

✅ In simple words:

cansniffer vcan0 lets you **watch live CAN messages** being sent over your virtual CAN bus — like a **live monitor for the car's brain**.



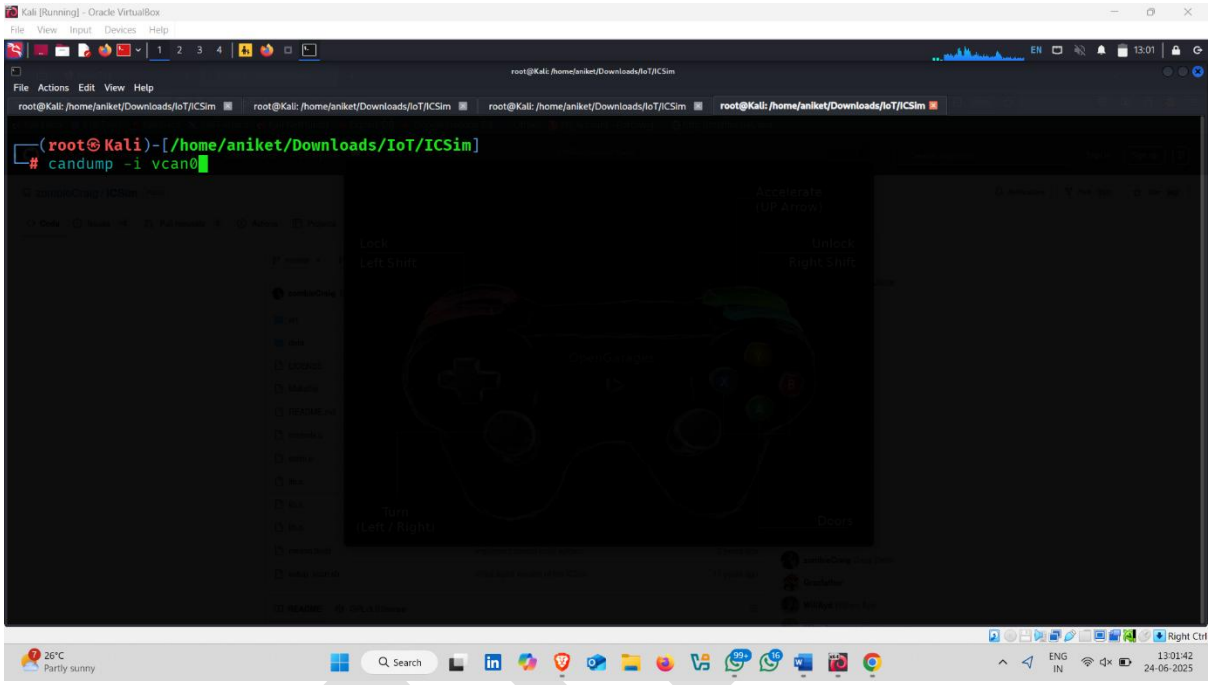

Kali [Running] - Oracle VirtualBox
File Actions Edit View Help
root@Kali: /home/aniket/Downloads/IoT/ICSim
00009 | 164 | 00 00 C0 1A A8 00 00 13 ....
00009 | 166 | D0 32 00 27 .2.'
00009 | 17C | 00 00 00 00 10 00 00 30 0
00008 | 183 | 00 00 00 06 00 00 10 32 2
00099 | 324 | 74 65 00 00 00 00 0E 1A te......
00009 | 18E | 00 00 7A ..z....
00011 | 191 | 01 00 90 A1 41 00 12 ....A ...
00020 | 1A4 | 00 00 00 08 00 00 00 3E ....>
00019 | 1AA | 7F FF 00 00 00 00 67 3F ......g?
00019 | 1B0 | 00 0F 00 00 00 01 75 ......u
00018 | 1CF | 80 05 00 00 00 1E ......
00018 | 1DC | 02 00 00 1B ....
00039 | 21E | 03 E8 37 45 22 06 3E ..7E".>
00013 | 244 | 00 00 00 01 46 ....F
00040 | 294 | 04 0B 00 02 CF 5A 00 3B .....Z.;
00103 | 305 | 80 08 ..
00100 | 309 | 00 00 00 00 00 00 00 84 ....
00100 | 320 | 00 00 30 ..0
00099 | 324 | 74 65 00 00 00 00 0E 38 te.....8
00100 | 333 | 00 00 00 00 00 00 3C ......<
00099 | 37C | FD 00 FD 00 09 7F 00 38 8
26°C
Partly sunny
Search
ENG
IN
12:58:51
24-06-2025


Kali [Running] - Oracle VirtualBox
File Actions Edit View Help
root@Kali: /home/aniket/Downloads/IoT/ICSim
vcan0 1CF [6] 10000000 00000101 00000000 00000000 00011110
vcan0 1DC [4] 00000010 00000000 00000000 00011011
vcan0 320 [3] 00000000 00000000 00010010
vcan0 324 [8] 01110100 01100101 00000000 00000000 00000000 00000000 00001110 00011010
vcan0 183 [8] 00000000 00000000 00000000 00000101 00000000 00000000 00010000 00100100
vcan0 37C [8] 11111101 00000000 11111101 00000000 00001001 01111111 00000000 00011010
vcan0 143 [4] 01101011 01101011 00000000 11100000
vcan0 305 [2] 10000000 00100110
vcan0 244 [5] 00000000 00000000 00000000 00000001 00100011
vcan0 095 [8] 10000000 00000000 00000111 11110100 00000000 00000000 00000000 00010111
vcan0 1A4 [8] 00000000 00000000 00000000 00001000 00000000 00000000 00000000 00111110
vcan0 1AA [8] 01111111 11111111 00000000 00000000 00000000 00000000 01100111 00111111
vcan0 1B0 [7] 00000000 00001111 00000000 00000000 00000000 00000001 01110101
vcan0 1D0 [8] 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00001010
vcan0 166 [4] 11010000 00110010 00000000 00100111
vcan0 158 [8] 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00101000
vcan0 161 [8] 00000000 00000000 00000101 01010000 00000001 00001000 00000000 00101011
vcan0 191 [7] 00000001 00000000 10010000 10100001 01000001 00000000 00010010
vcan0 18E [3] 00000000 00000000 01111010
vcan0 133 [5] 00000000 00000000 00000000 00000000 10110110
vcan0 136 [8] 00000000 00000010 00000000 00000000 00000000 00000000 00000000 00111001
vcan0 13A [8] 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00110111
vcan0 13F [8] 00000000 00000000 00000000 00000101 00000000 00000000 00000000 00111101
vcan0 164 [8] 00000000 00000000 11000000 00011010 10101000 00000000 00000000 00010011
vcan0 17C [8] 00000000 00000000 00000000 00000000 00010000 00000000 00000000 00110000
vcan0 294 [8] 00000100 00001011 00000000 00000010 11001111 01011010 00000000 00111011
vcan0 21E [7] 00000011 11101000 00110111 01000101 00100010 00000110 00111110
vcan0 039 [2] 00000000 00011011
vcan0 183 [8] 00000000 00000000 00000000 00001011 00000000 00000000 00010000 00111101
vcan0 143 [4] 01101011 01101011 00000000 11111111
vcan0 095 [8] 10000000 00000000 00000111 11110100 00000000 00000000 00000000 00100110
vcan0 244 [5] 00000000 00000000 00000000 00000001 00100111
26°C
Partly sunny
Search
ENG
IN
13:01:15
24-06-2025

- Open another new terminal
- Type following command

Command -: candump -I vcan0

Explanation-: This command **monitors and displays all CAN messages** on the vcan0 interface.



- You have successfully set up a **simulated car dashboard (ICSim)** and are **controlling it using a virtual controller**. Both the dashboard and the controller communicate over a **virtual CAN interface (vcan0)**.

By pressing keys like the **Up Arrow**, **Left/Right Arrow**, **Shift**, or **A/B/X/Y**, you're sending **CAN messages** through vcan0. These messages control the virtual vehicle's **speed**, **blinkers**, **locks**, and **doors**, just like in a real automotive CAN network.

This setup is commonly used for:

- **Learning CAN protocols**
- **Practicing car hacking**
- **Reverse engineering CAN traffic**

It’s a **safe and offline lab environment** to simulate **real-world vehicle systems** and test various security concepts.

Exploring IoT and OT Device Exposure Using Shodan

Shodan is a **search engine for internet-connected devices**.

Unlike Google, which searches websites, **Shodan searches devices** like:

- **Webcams**
- **Routers**
- **Smart TVs**
- **Industrial Control Systems (ICS/OT)**
- **SCADA systems**
- **Traffic lights, printers, servers, etc.**

🔍 What does it do?

Shodan scans the internet and collects information such as:

- **Open ports**
- **Running services**
- **Software versions**
- **Location**
- **Security vulnerabilities**

✅ Why it's used:

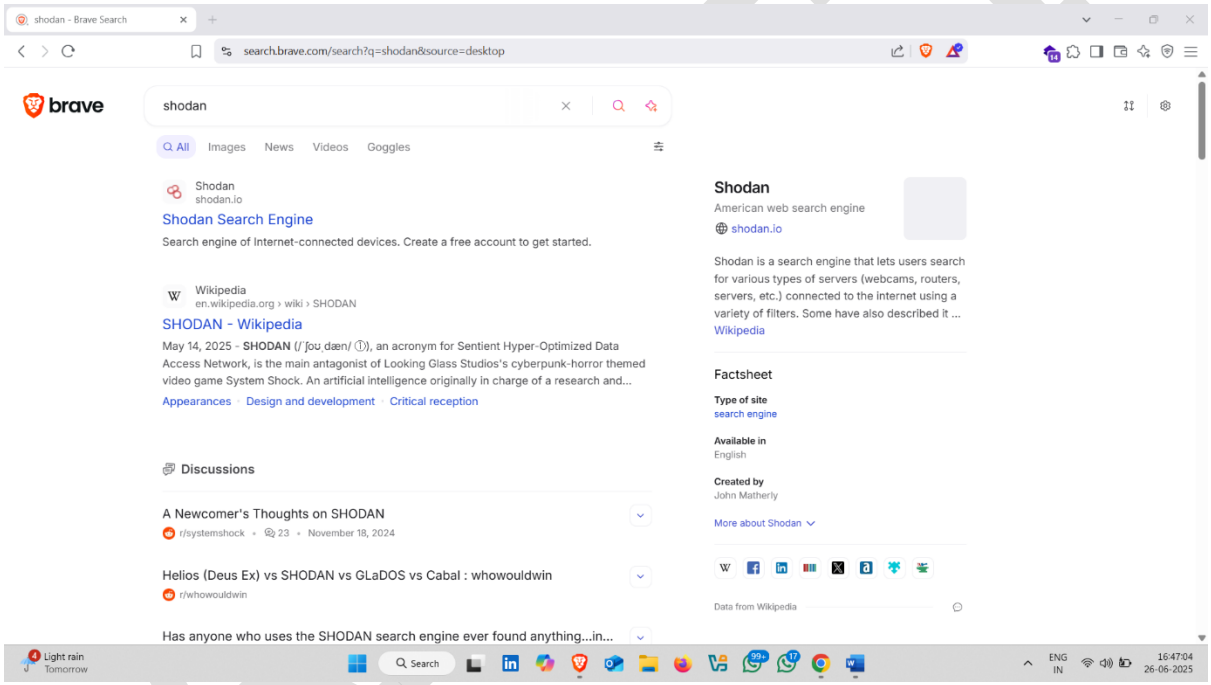
- To **discover exposed IoT or OT devices**
- For **security research and ethical hacking**
- To perform **footprinting and reconnaissance**
- To check **what your own devices are exposing**

🧪 Example Use Cases:

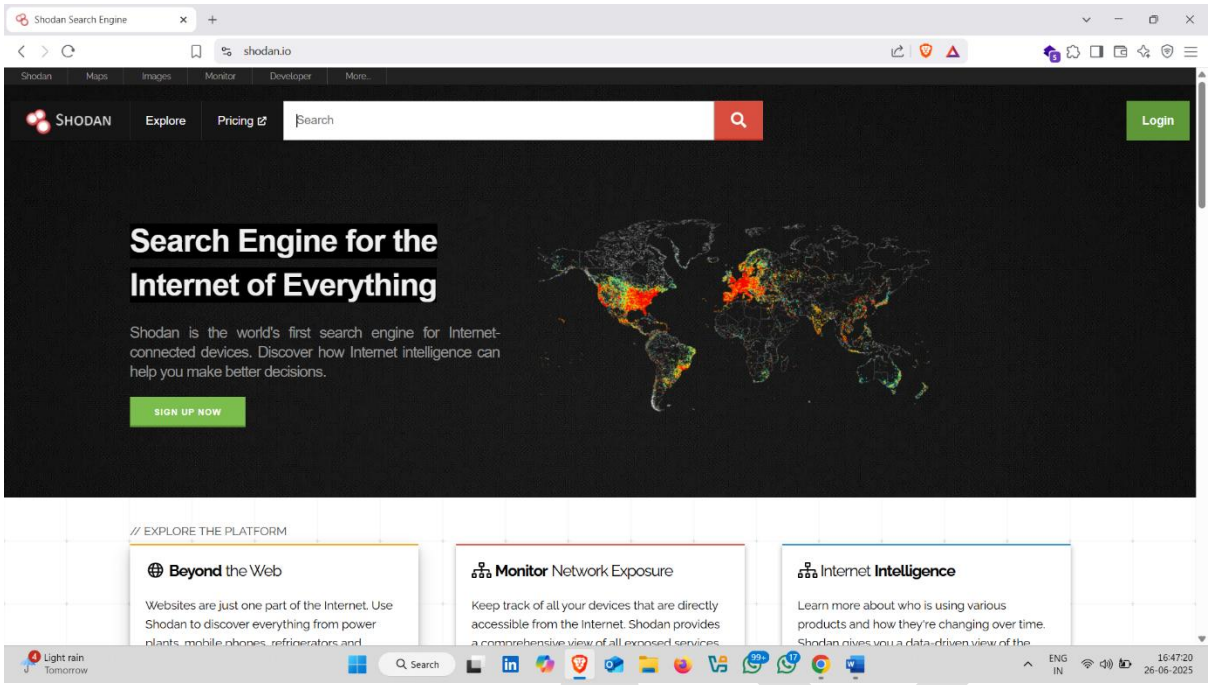
- Find **cameras or traffic systems** with no passwords
- Discover **ICS devices running Modbus or Siemens S7**
- Search for **routers with outdated firmware**
- Help companies fix **exposed or vulnerable devices**

How to use it -:

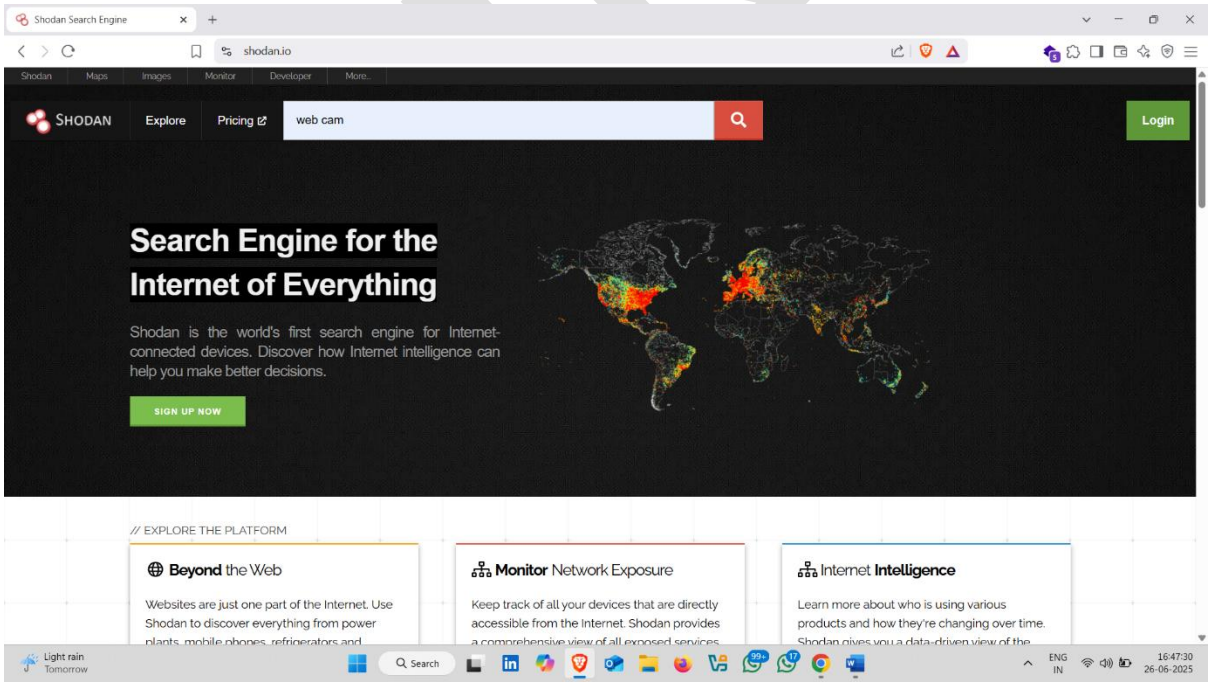
- Open browser and search **shodan**
- Click on first website **shodan.io**



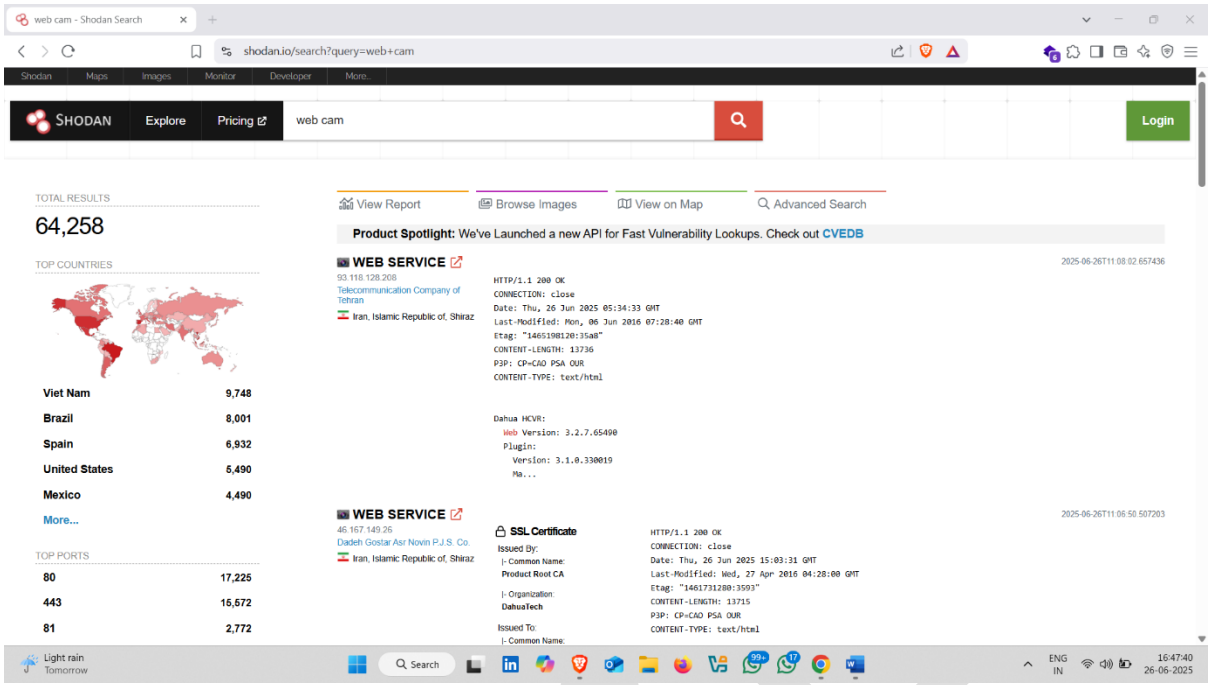
- Now search **web cam**



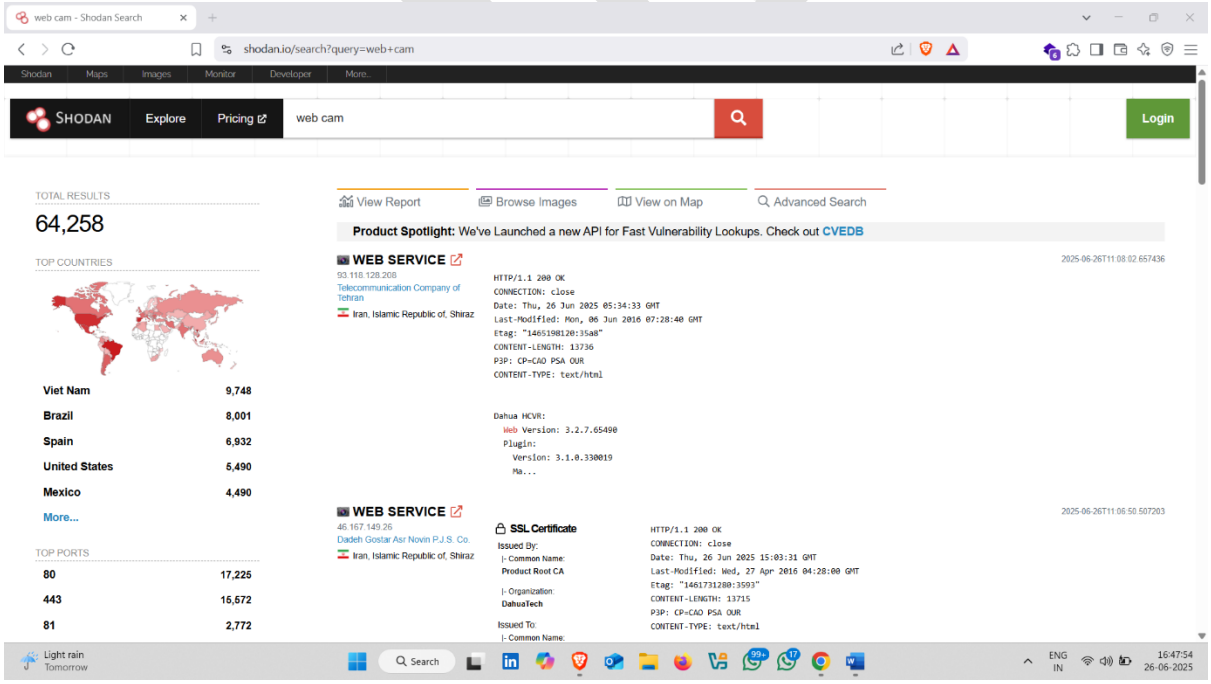
- Click on search icon



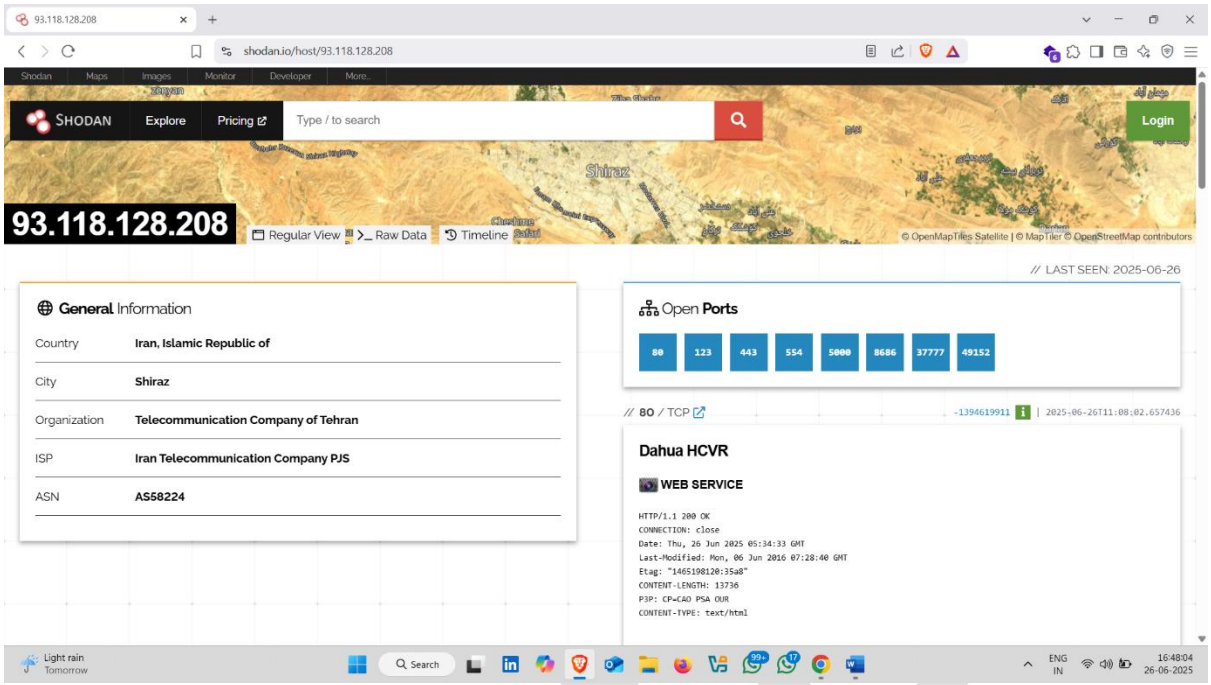
- Here , it finds many webcam devices



- Now click on any web services to get detailed information



- Detailed information



EXTRA ACTIVITY

Additional Search Engines

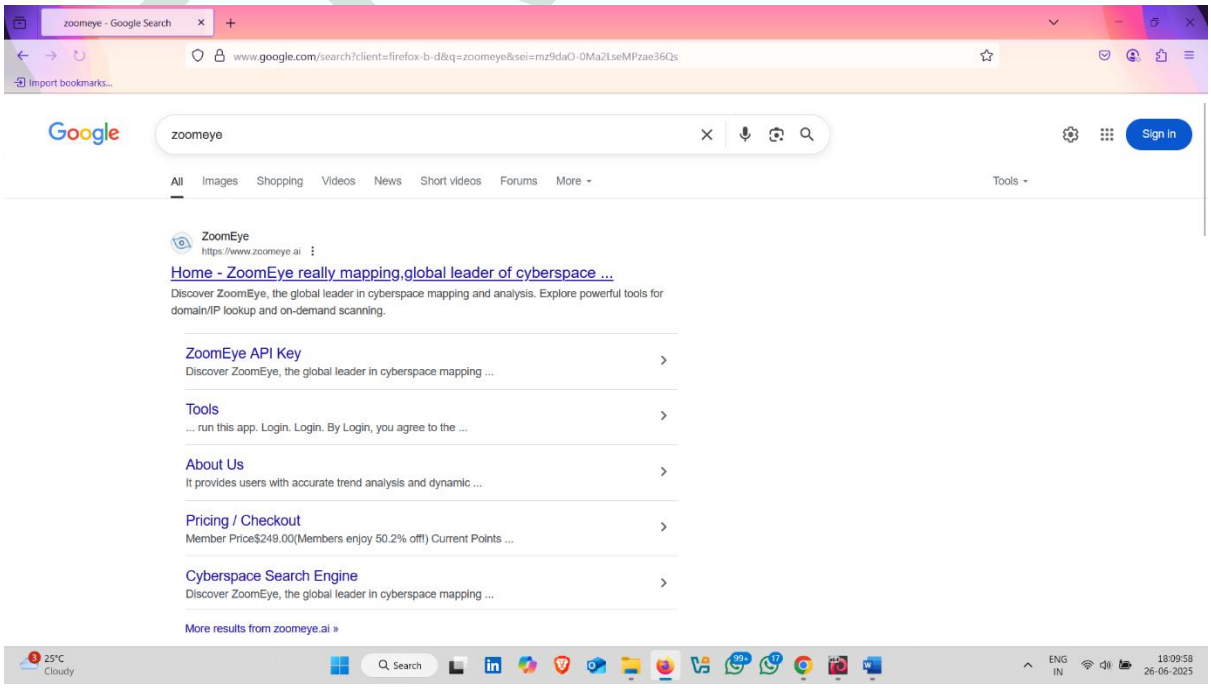
1.ZoomEye search Engines

ZoomEye is a **cybersecurity search engine** developed by **360 Netlab** (a Chinese cybersecurity firm). It scans the entire internet for **connected devices and services**, helping security professionals find:

- **IoT devices** (like webcams, DVRs, routers)
- **Web servers**
- **Databases**
- **Industrial control systems (ICS/SCADA)**
- **Vulnerable systems**

How to use it -:

- Open Browser and search **ZoomEye**
- **Click on first official website** 👇 ✅



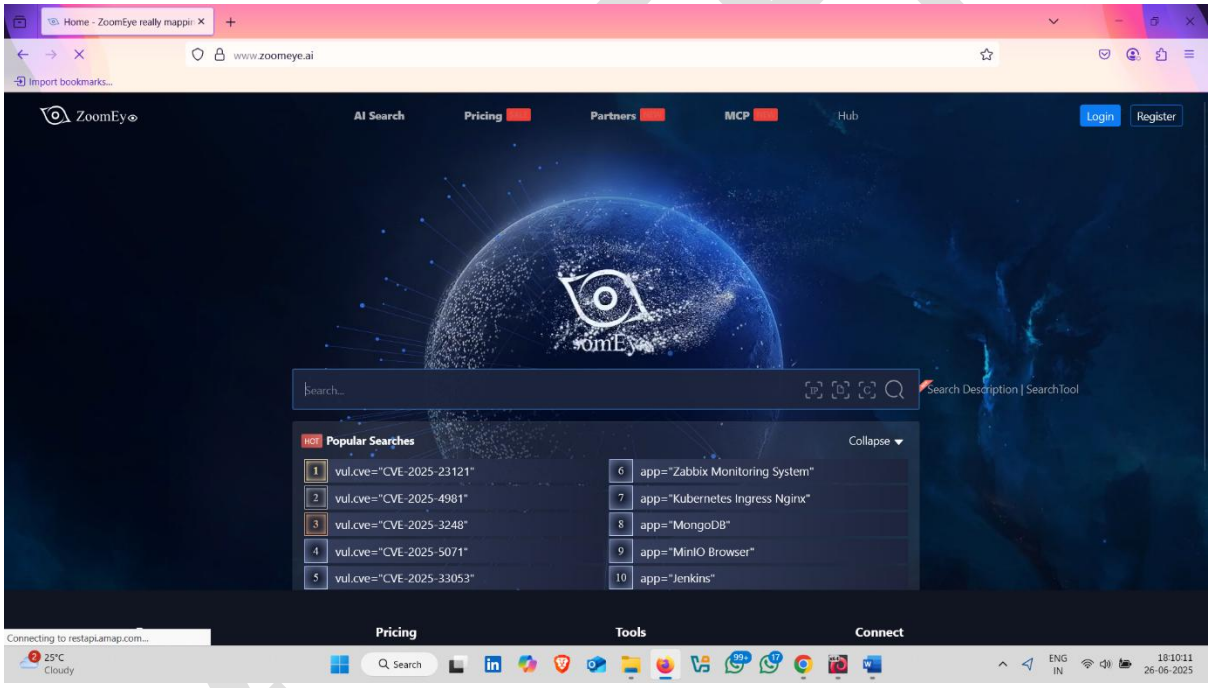
- **Here it also shows popular vulnerabilities searches**
- Search scada

SCADA (Supervisory Control and Data Acquisition) is a system used to monitor, control, and collect data from industrial processes in real time, often from remote locations.

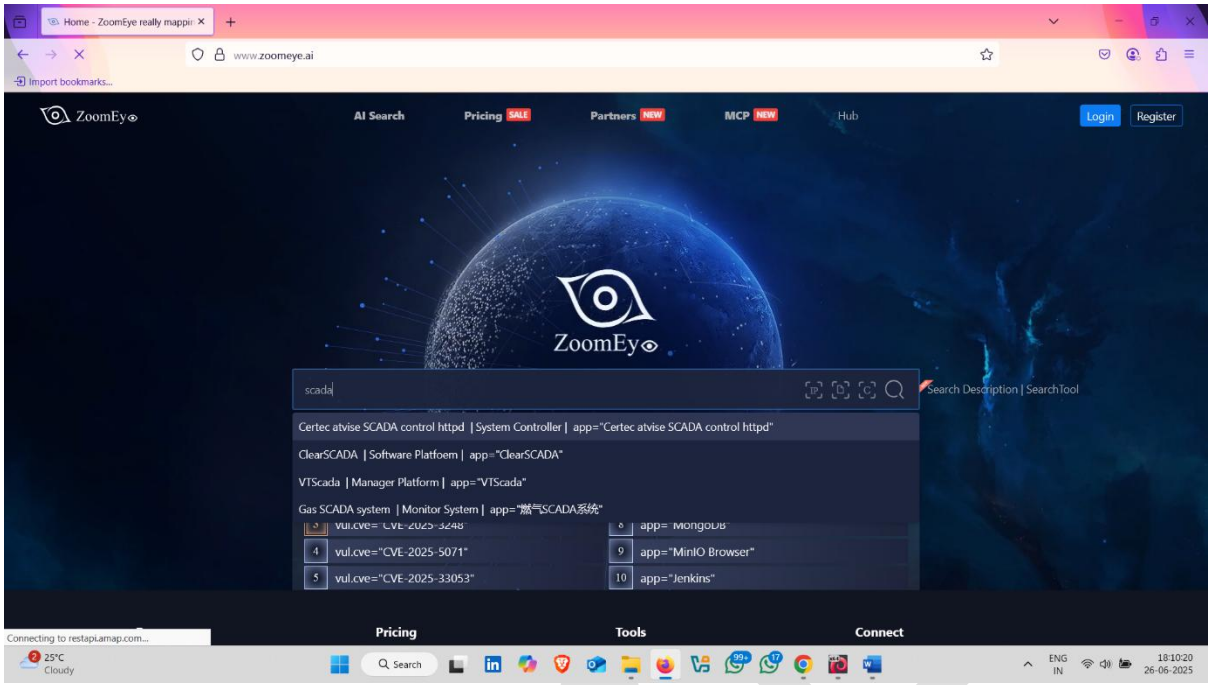
Example:

SCADA helps manage systems like:

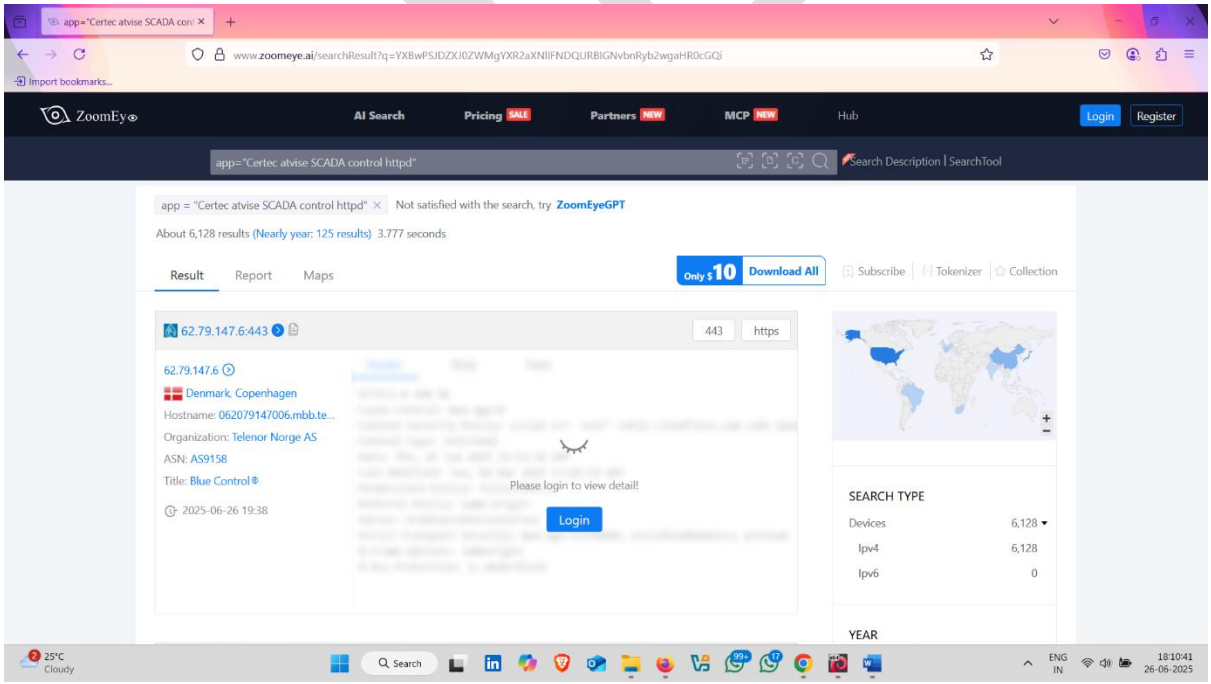
- Power grids ⚡
- Water treatment 💧
- Factory machinery 🏭



- Click on search



- **Here it shows all publically available results ✅ 👇**



2.FOFA Search Engines

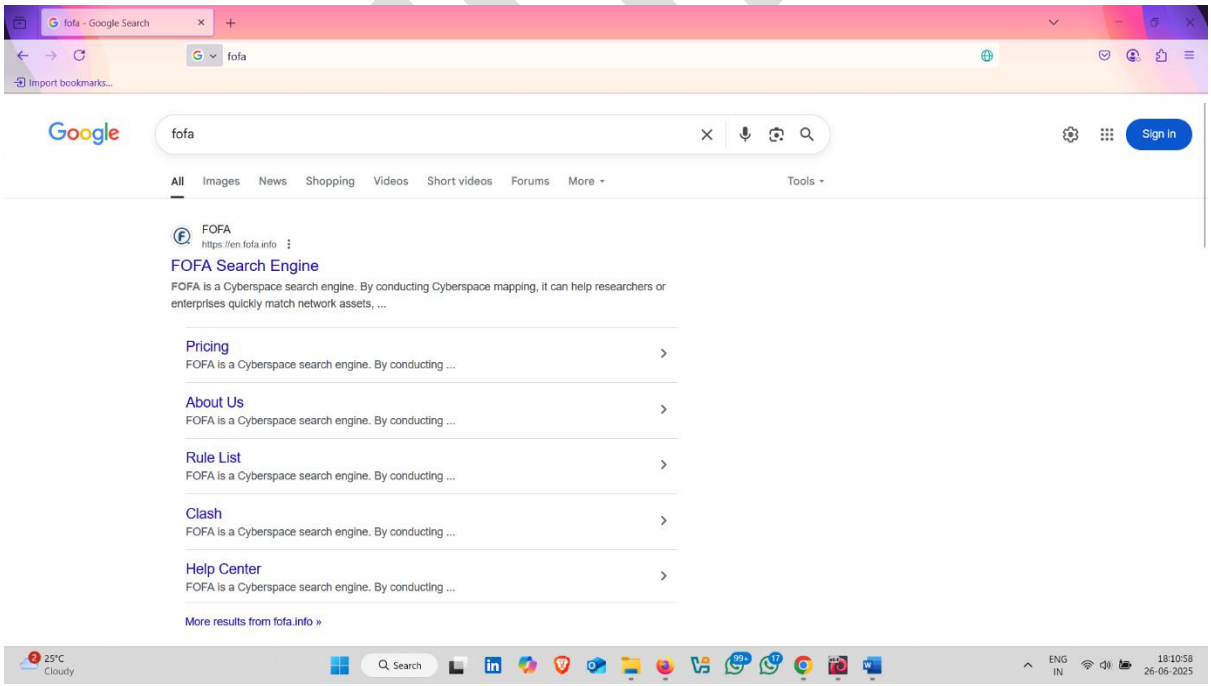
FOFA is a cyber intelligence platform that helps find IoT/OT devices, servers, webcams, industrial systems, and more by scanning the internet for open ports, services, protocols, and digital fingerprints.

🔍 What FOFA is used for:

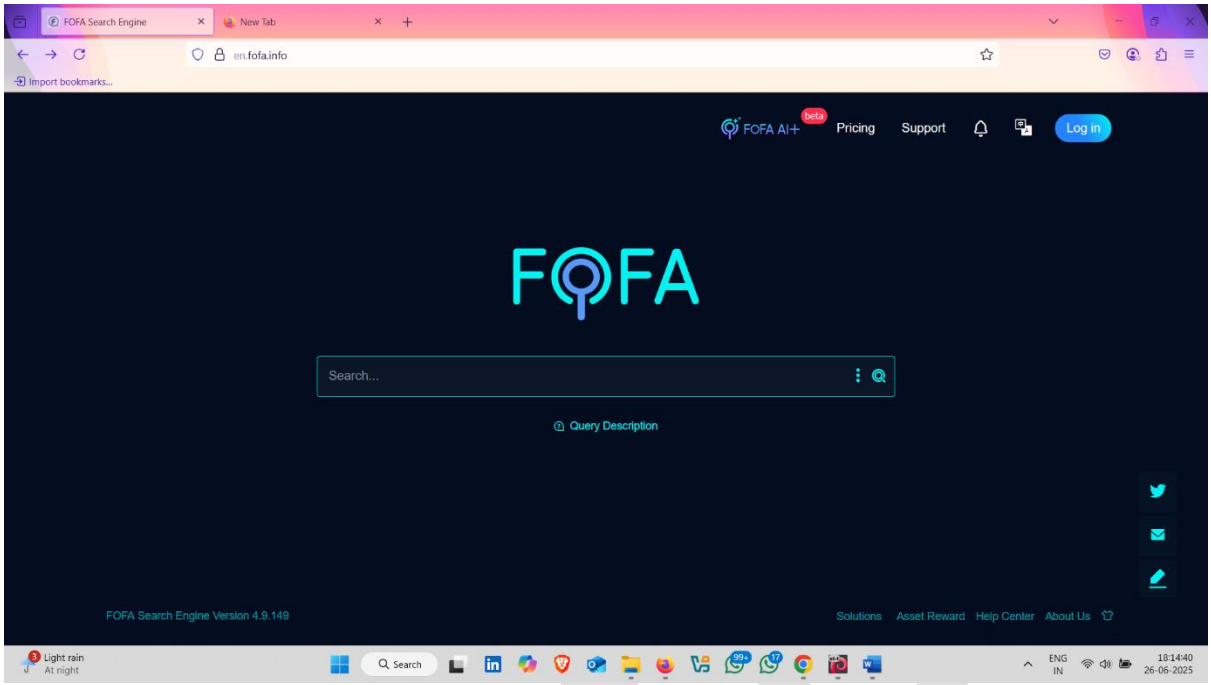
- Finding **vulnerable devices** online
- Collecting **cyber threat intelligence**
- Performing **IoT/OT asset discovery**
- Assisting in **bug bounty** and **red teaming**
- Tracking **exposed APIs, ports, or services**

How to use it -:

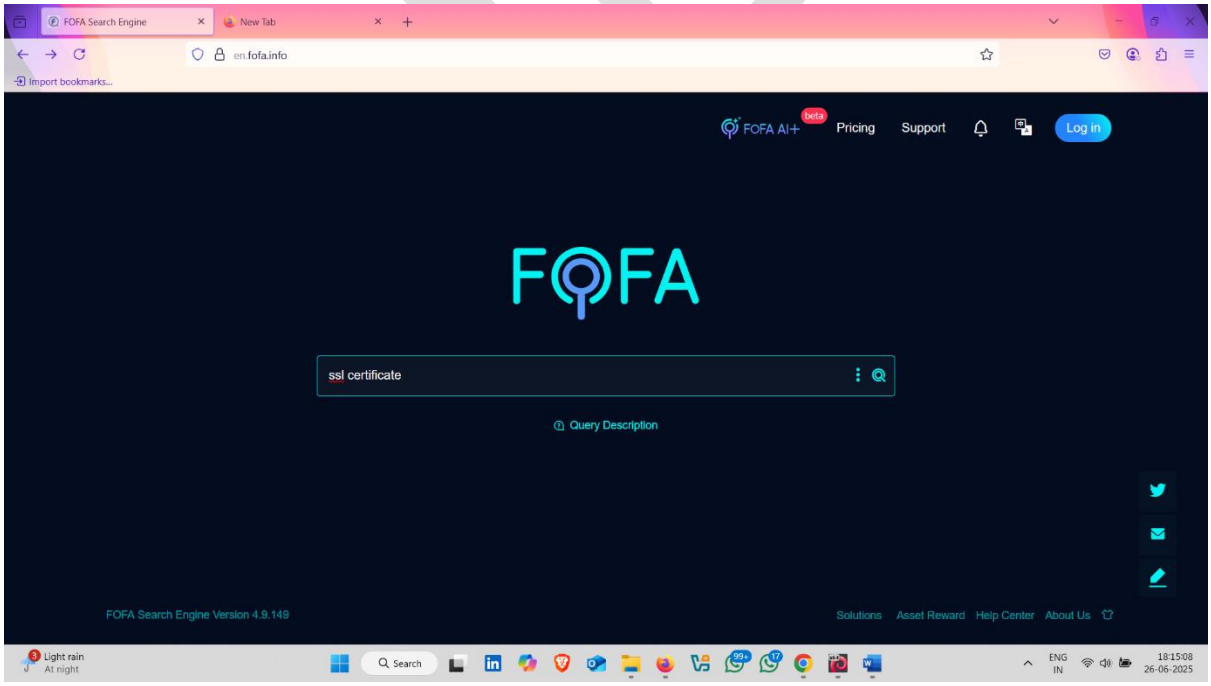
- Open browser and search **FOFA**



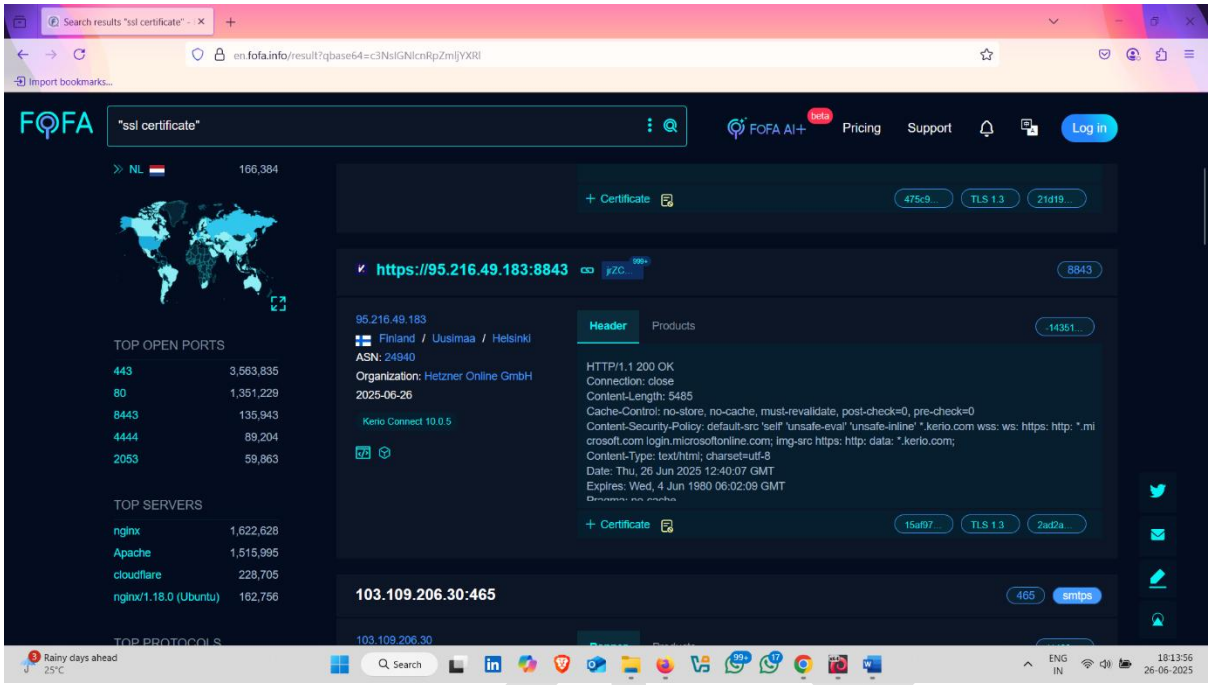
- Now search SSL Certificate



- Click on search button ✅ 👇



- **Here it shows all publically available SLL Certificate details** 👇✅

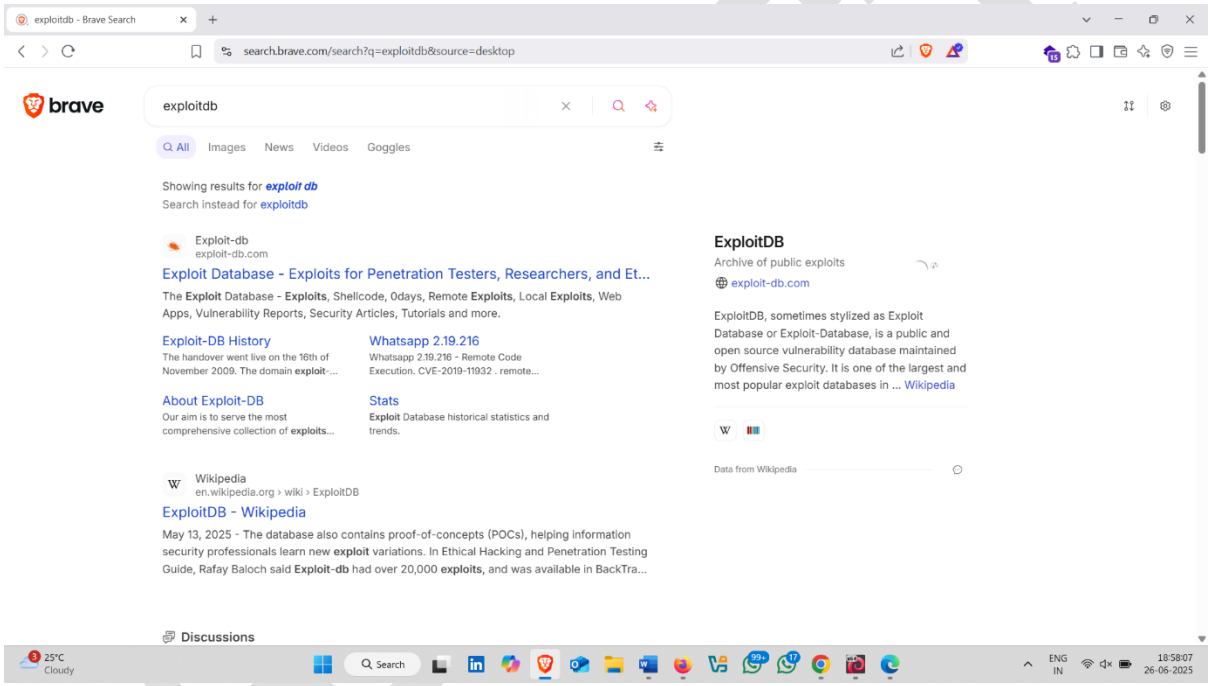


3.Exploit DB

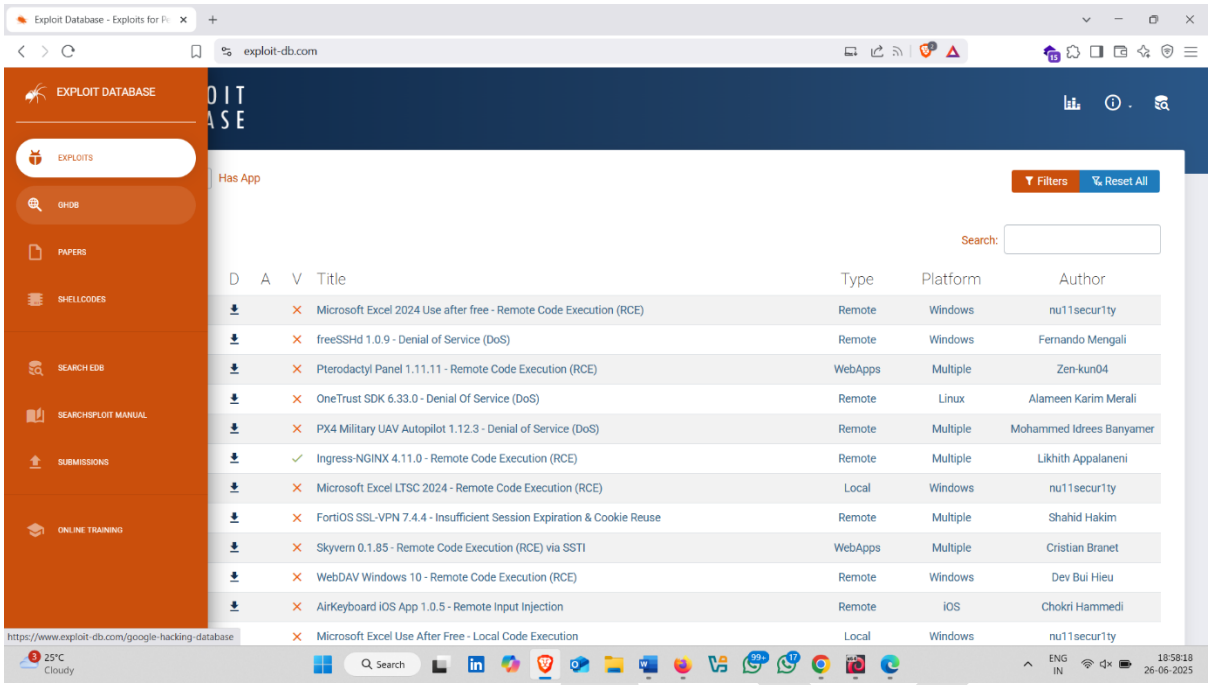
Exploit-DB is a public repository of known exploits and vulnerabilities. In the context of **IoT (Internet of Things)** and **OT (Operational Technology)**, security researchers and penetration testers use Exploit-DB to find **existing vulnerabilities** in smart devices, industrial systems, and their software/firmware.

How to use it -:

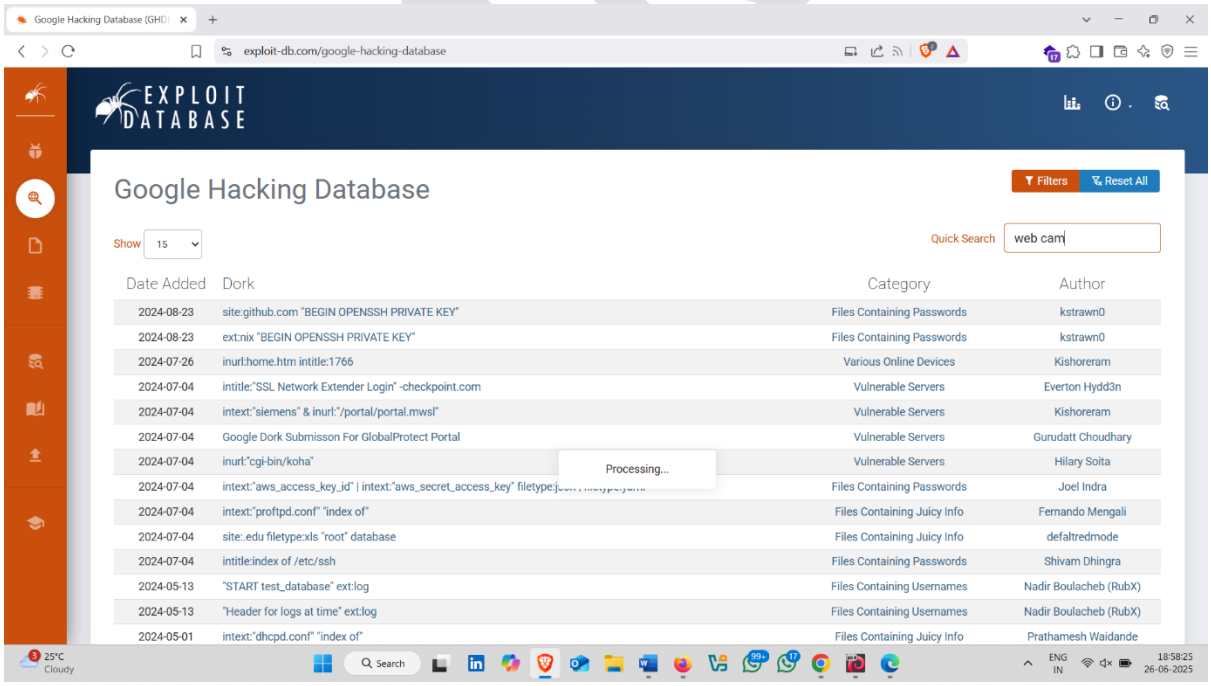
- Open browser and search **exploit-DB and click on first website**



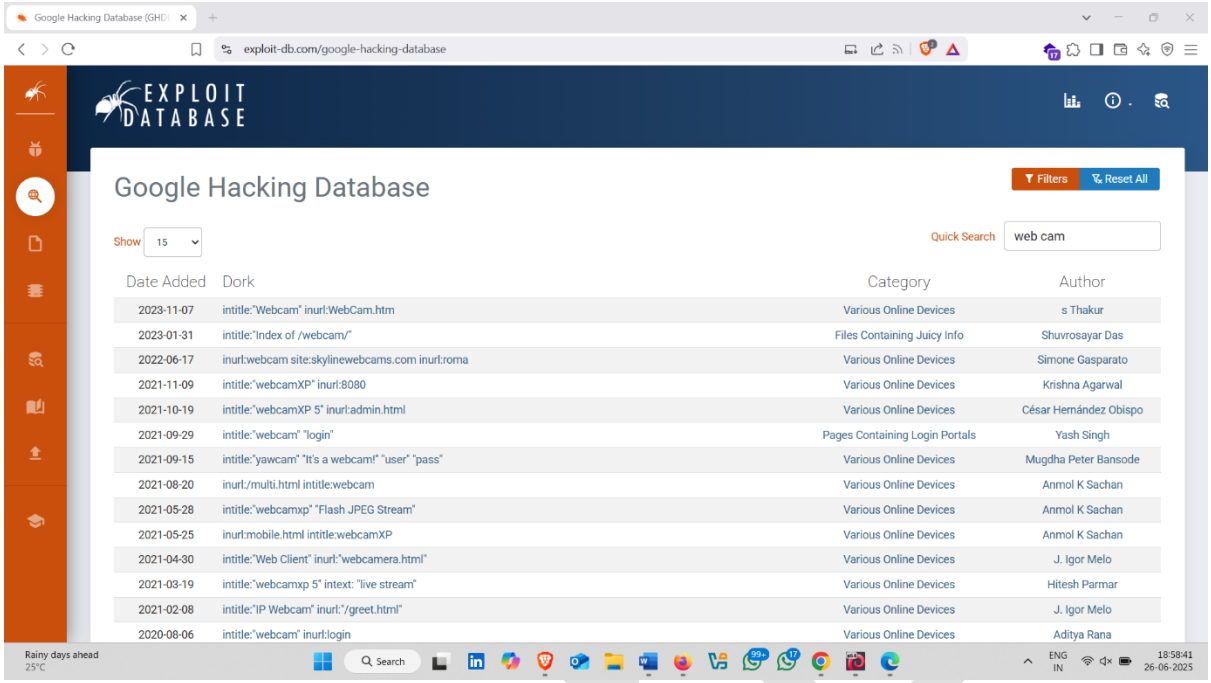
- Click on **GHDB**



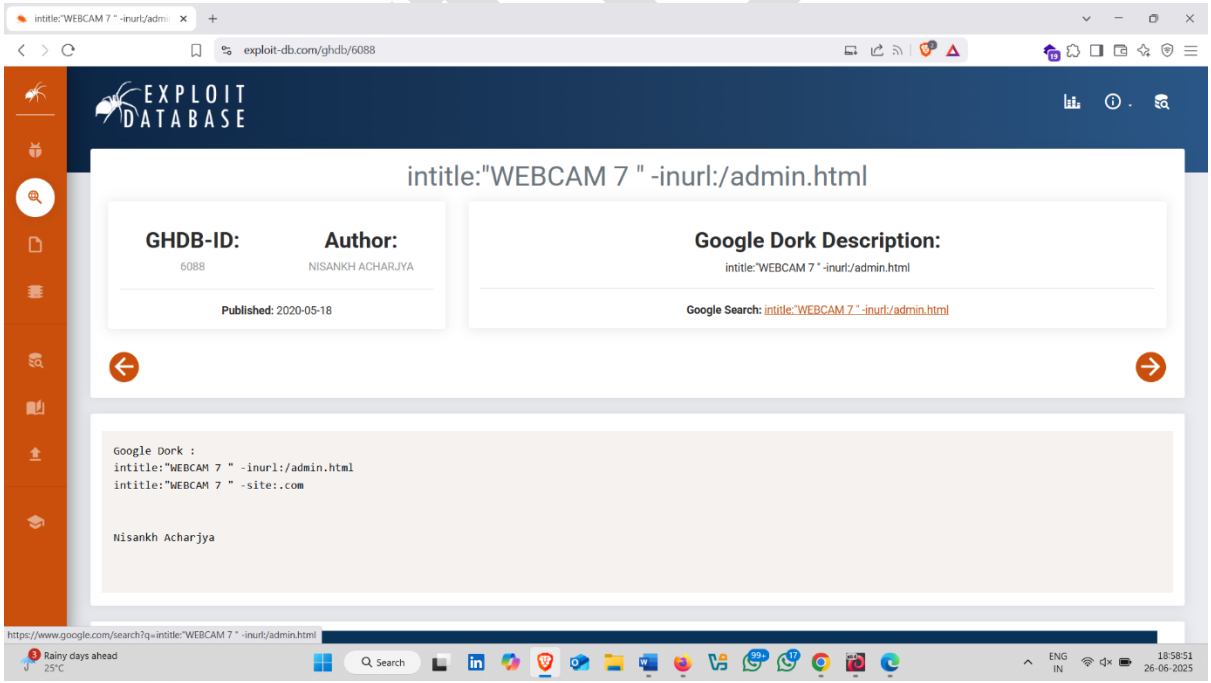
- Seach for **webcam**



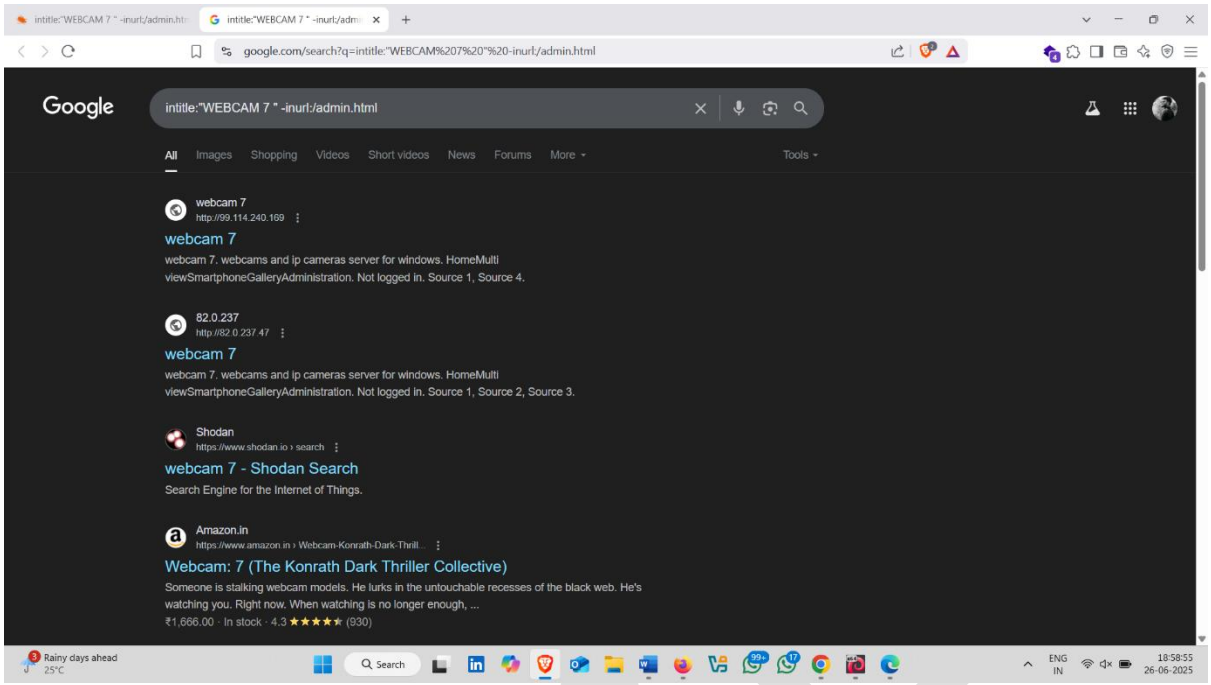
- **All webcam related results 👇 , click on any link**



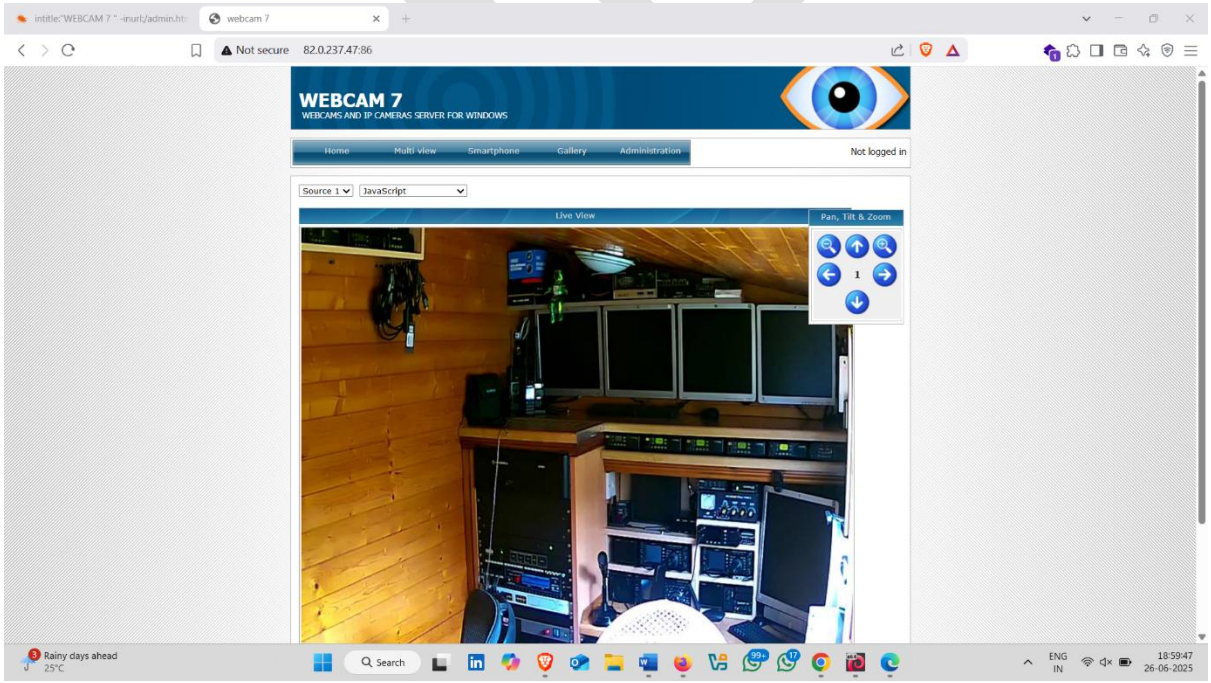
- **Open this URL 👇**



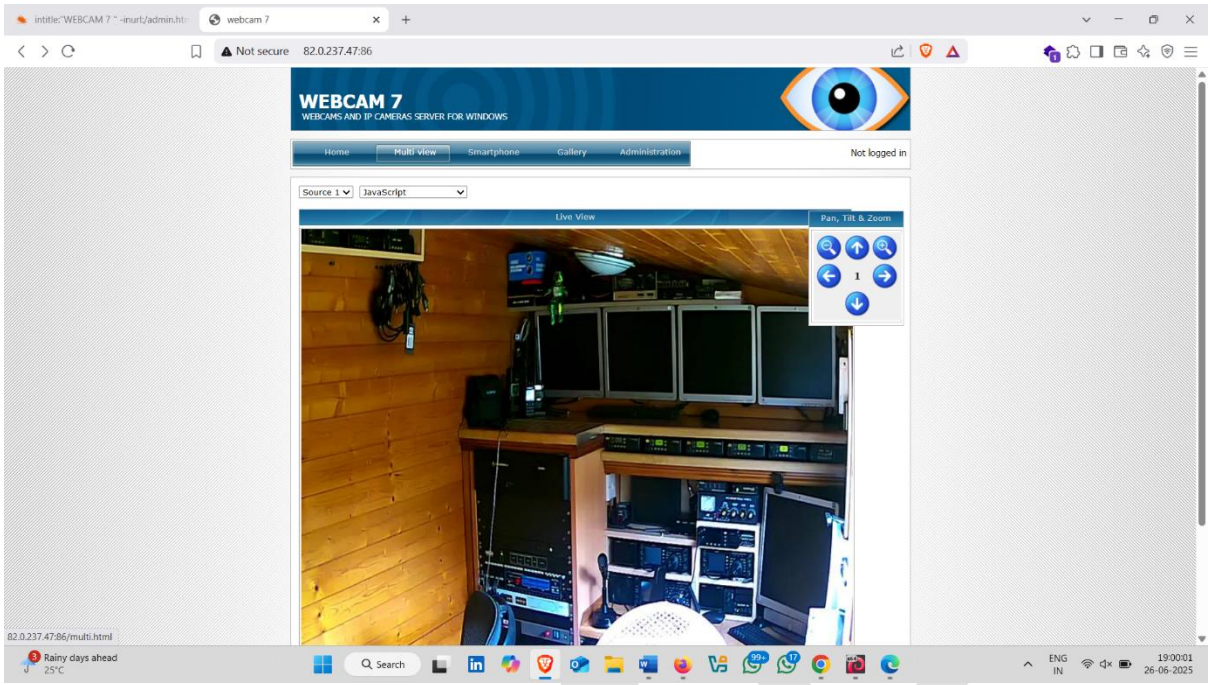
- Now open any website



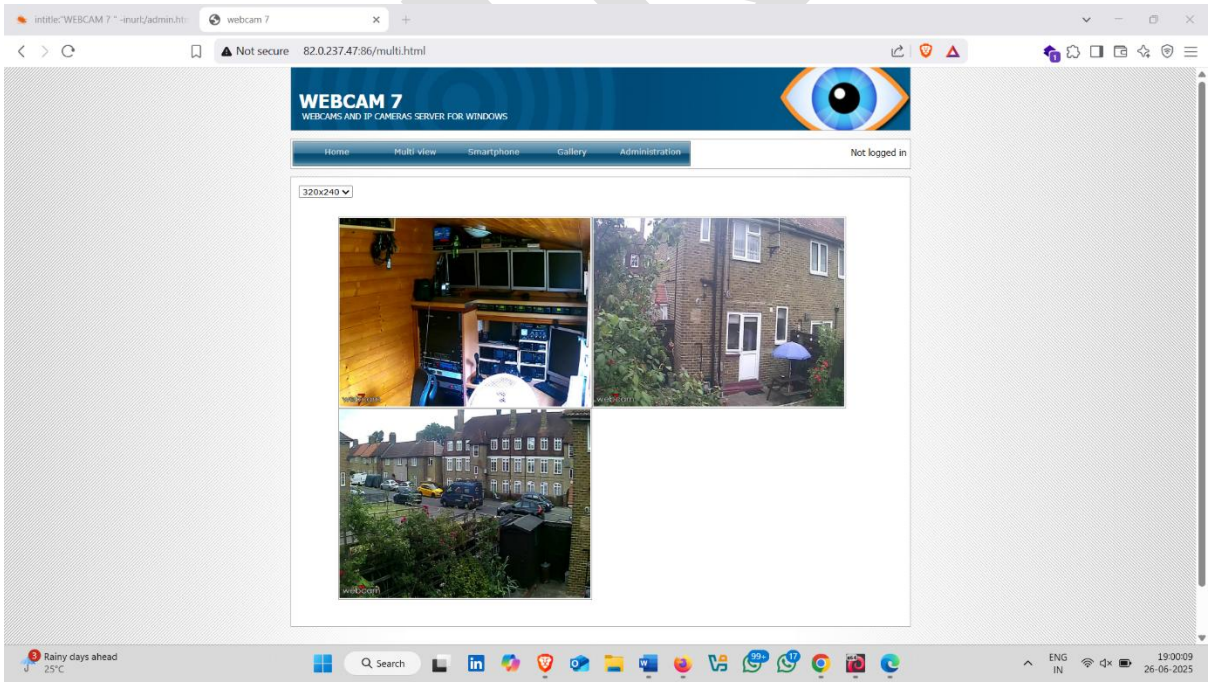
- Live camera monitoring 👇✅



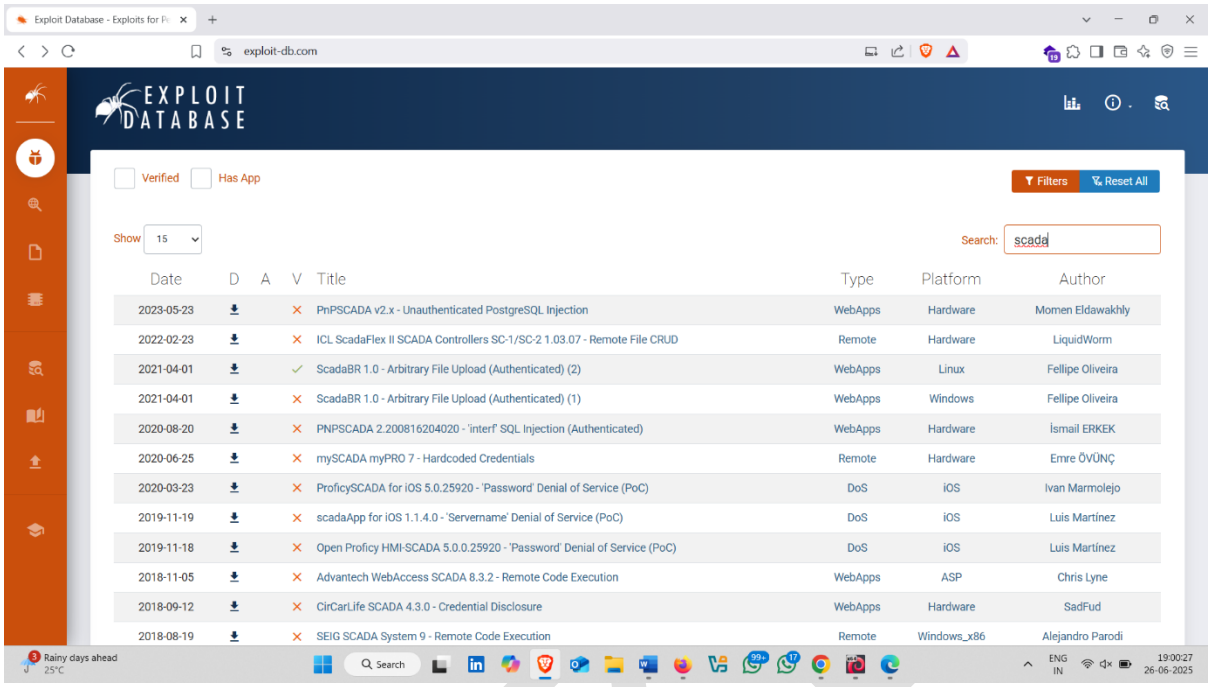
- Click on **multi view** – can access multiple camera



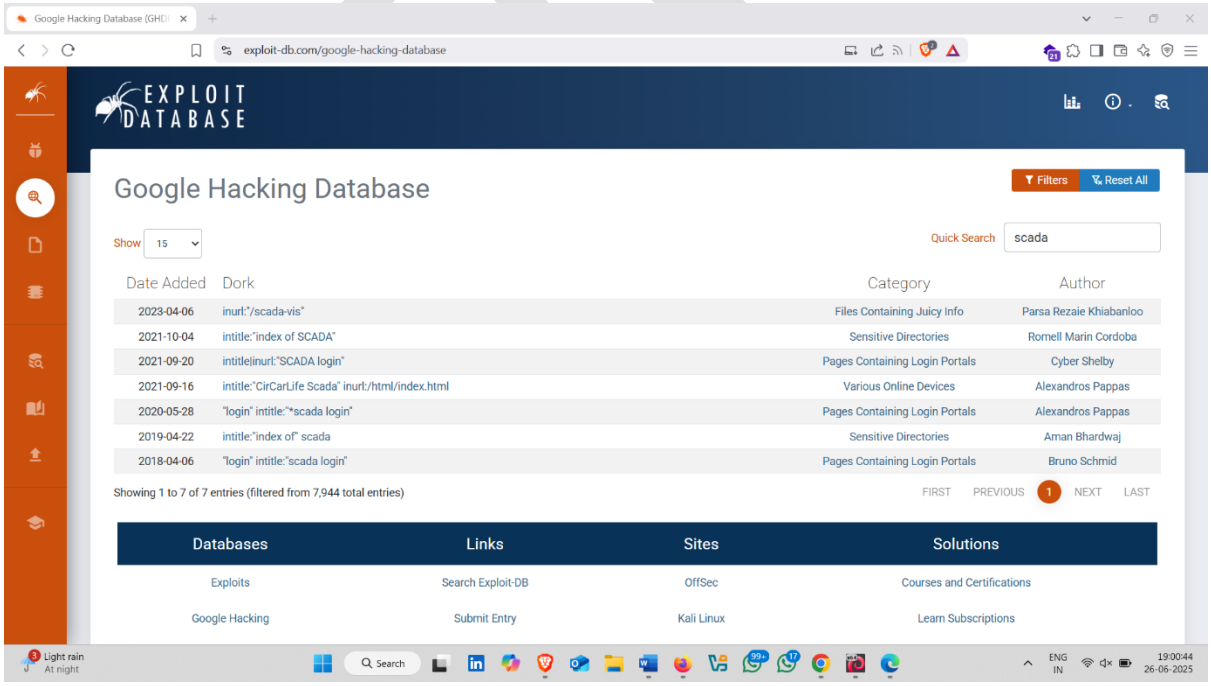
- **Multiple camera access**



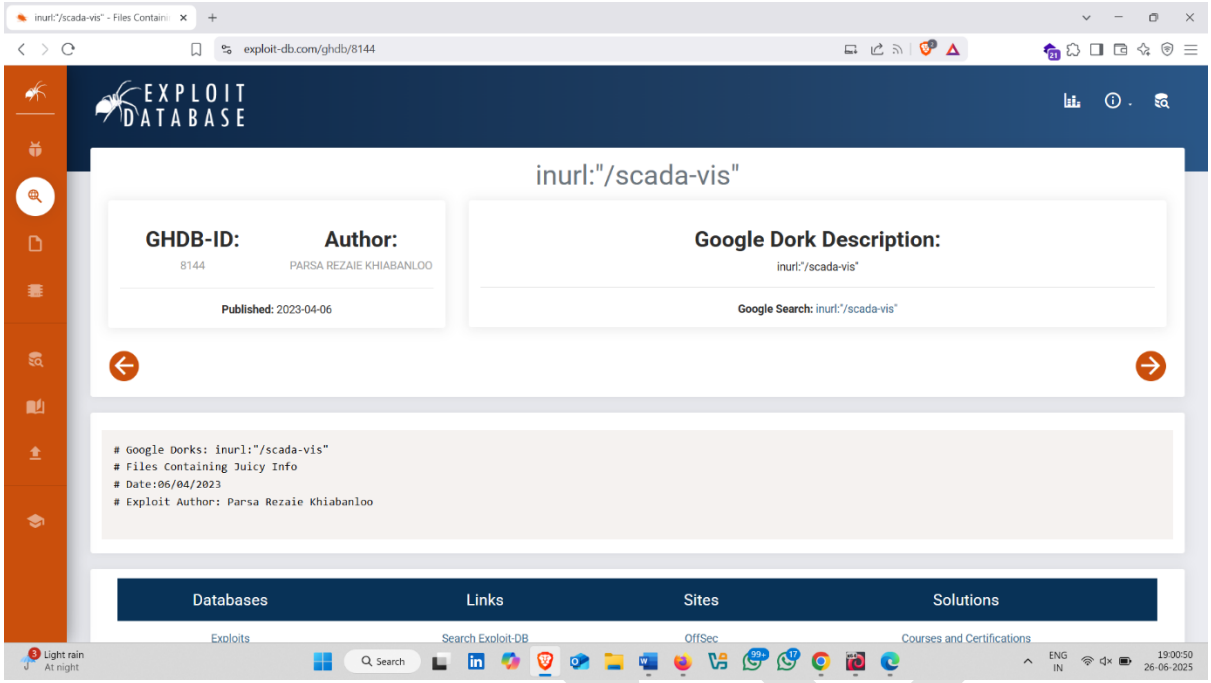
- **Now back to Exploit-DB database**
- Search Scada ✅ 👇



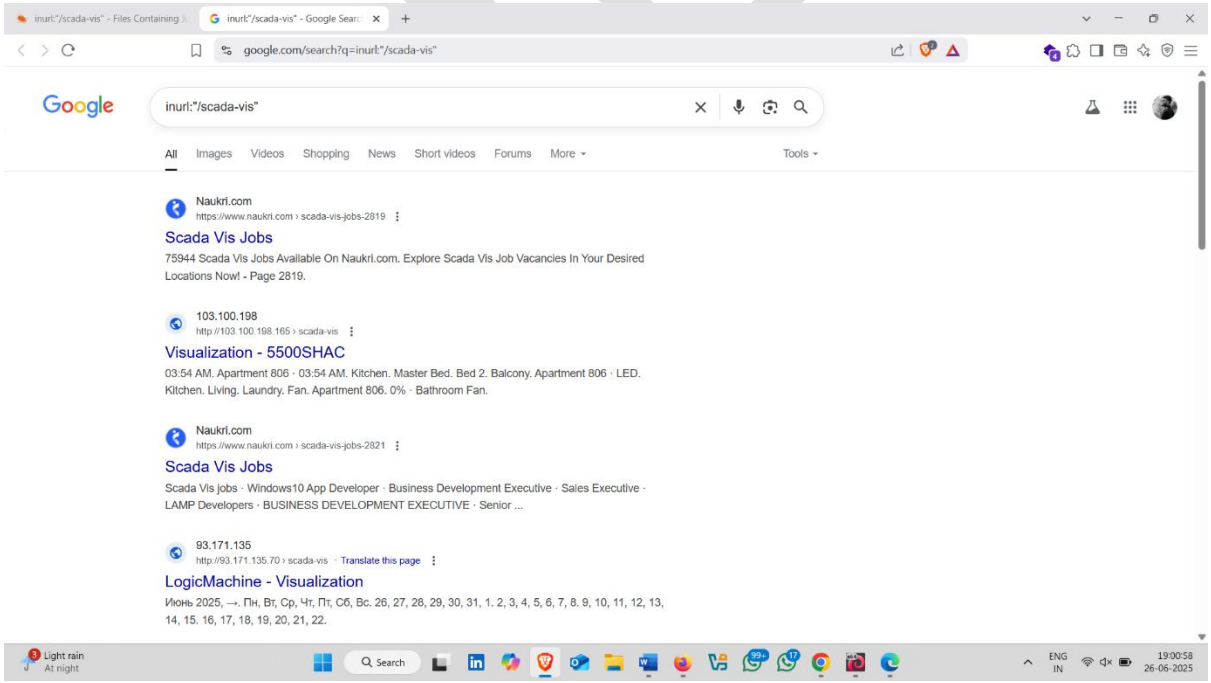
- Scada related result 👇 ✅



- **Open this url** ✅ 👇



- Click on **second website**

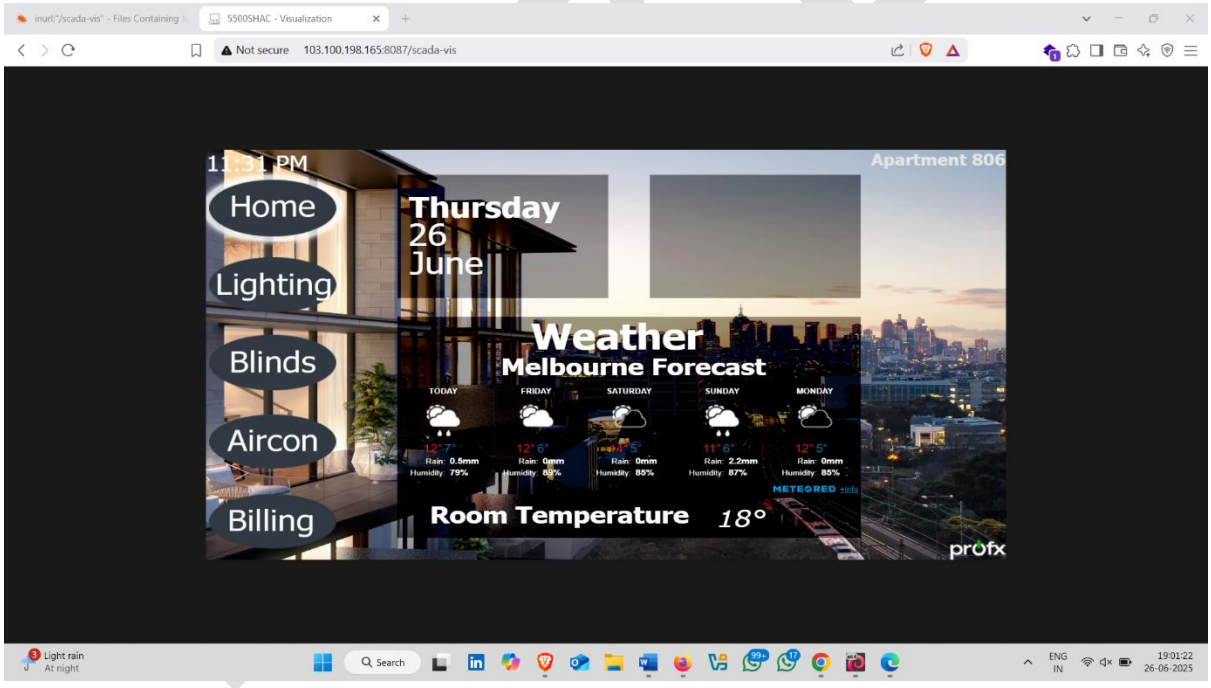


- **Result ✅ 👇**

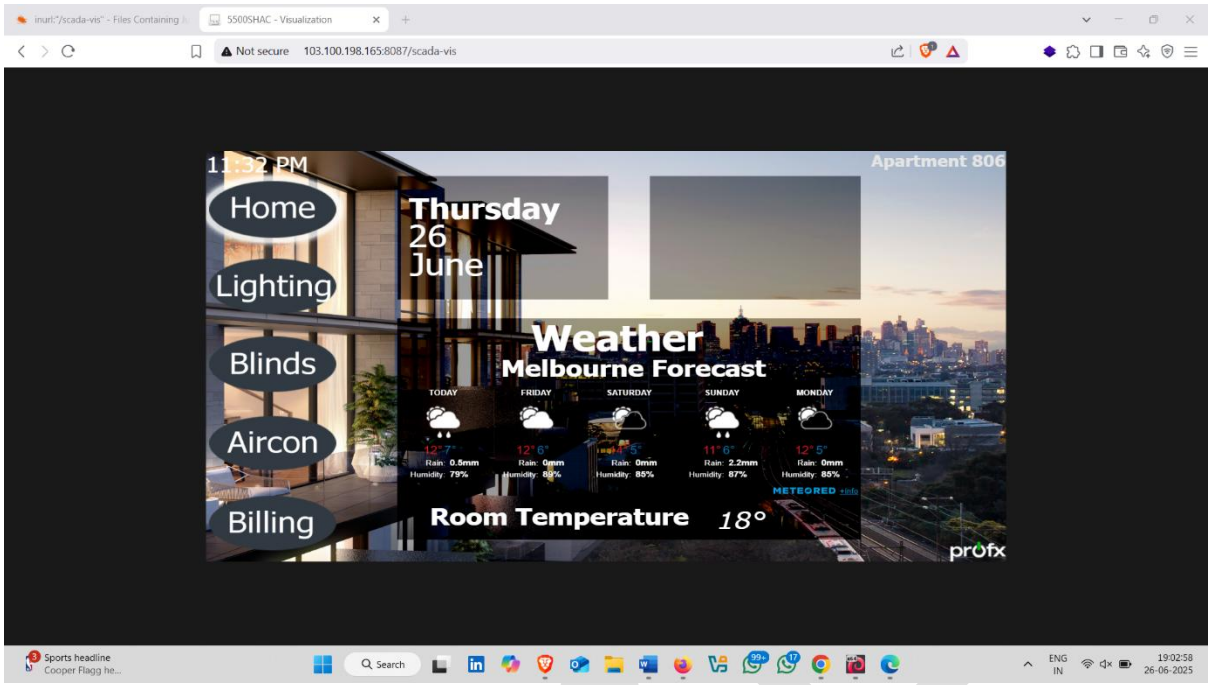
🏢 Control Panel Interface

This is a **home automation or smart building SCADA dashboard**, used for **monitoring and controlling apartment utilities remotely**. Here's what's visible:

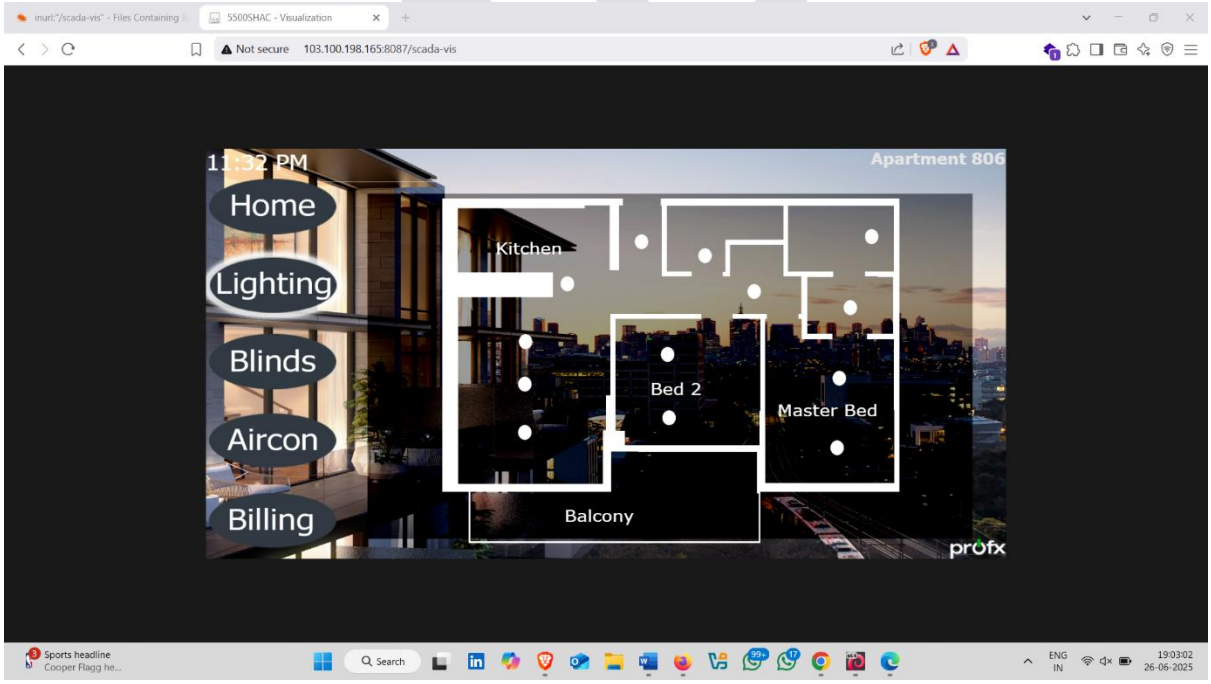
- **Sidebar Menu Options:**
  - Home: Dashboard overview
  - Lighting: Control room/apartment lights
  - Blinds: Control window blinds
  - Aircon: Control air conditioning (AC)
  - Billing: Possibly shows electricity or HVAC billing info



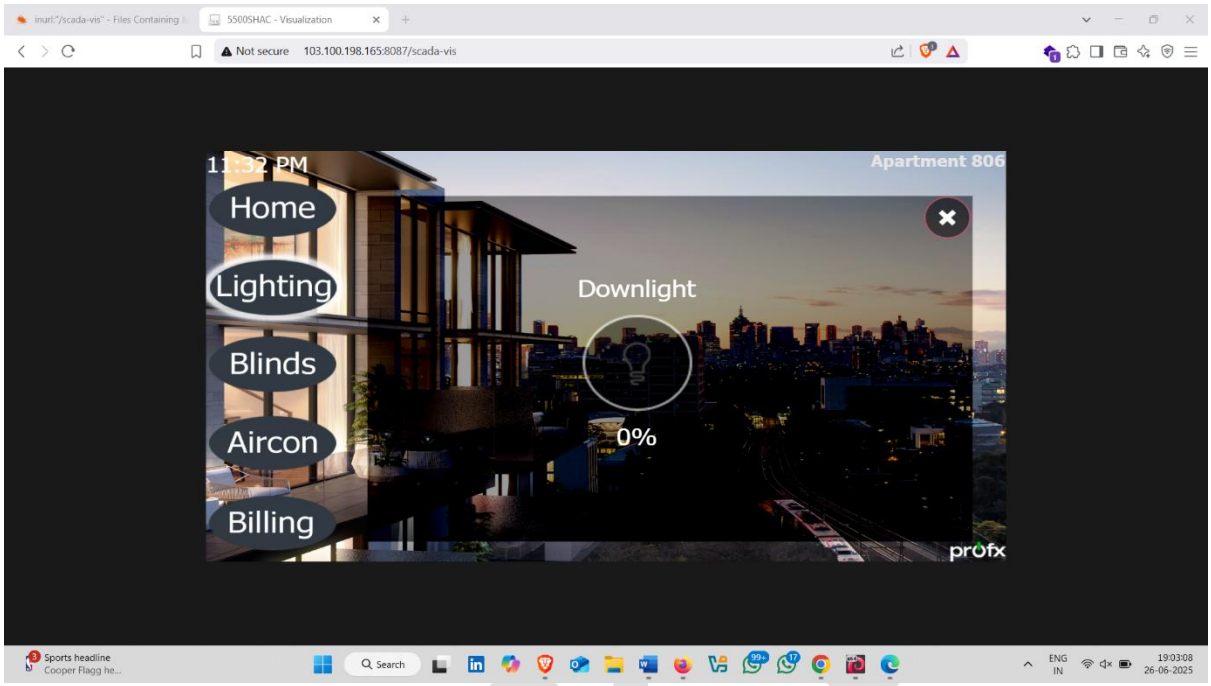
- Click on Lighting



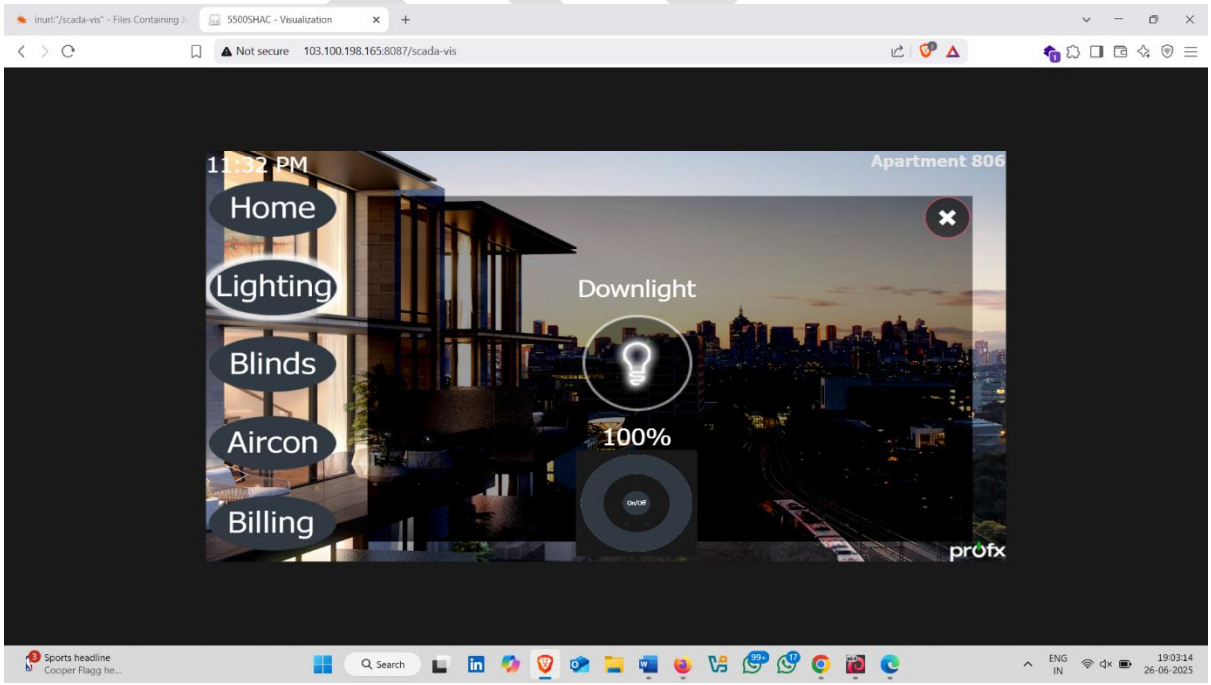
- Click on any section like -kitchen , bedroom etc



- Click on downlight options



- **Result – lights turn on 👇✅**



4.Criminal Ip

Criminal IP is a **cyber threat intelligence (CTI) search engine and platform** designed to **identify malicious or vulnerable IP addresses**, domains, and assets across the internet. It is commonly used for **OSINT (Open Source Intelligence)** in cybersecurity operations, red teaming, vulnerability research, and risk assessment.

It allows researchers and security professionals to analyze:

- IP reputations
- Vulnerable systems (IoT/OT devices)
- Exposed ports and services
- Domains linked to malware or phishing campaigns

📌 Developed by **AI SPERA**, it is often used alongside platforms like **Shodan**, **ZoomEye**, and **Censys**.

⚙️ **Features for OSINT and Threat Intelligence**

Feature	Description
IP Reputation Search	Shows malicious history, malware associations, CVEs, and geographic information about an IP address.
Device Fingerprinting	Identifies web servers, IoT/SCADA systems, open ports, services, and SSL certificates.
Vulnerability Info	Shows CVEs tied to discovered devices (e.g., outdated camera firmware, ICS/PLC vulnerabilities).
Subdomain Enumeration	Reveals subdomains associated with a domain for surface area analysis.
Domain Intelligence	Gives details about phishing domains, C2 servers, or malware-distributing hosts.

Feature	Description
Malicious URL & Domain Feed	Provides real-time indicators of compromise (IOCs) for threat hunting.
Geolocation	Maps IPs and devices to countries, cities, and ASN numbers.
Export & API Integration	Integrate with SIEM/SOAR tools for automation and advanced analysis.

🛠️ Use Cases in IoT and OT Hacking / Security

Use Case	How Criminal IP Helps
IoT Device Exposure Mapping	Detects publicly exposed IoT cameras, routers, sensors, and gateways with open ports or default credentials.
Reputation Checking	Identifies if an IP is blacklisted or known for malicious activity (e.g., used in botnets or DDoS attacks).
Reconnaissance for Red Team	Collects OSINT about ICS/SCADA environments before penetration testing or simulations.
Zero-Day and CVE Research	Finds devices affected by recent vulnerabilities (e.g., Hikvision CVEs, MQTT protocol flaws).
Network Hygiene Audit	Organizations use it to monitor their own IP ranges for unintended exposure.
Mapping OT Infrastructure	SCADA or HMI devices accidentally exposed online can be flagged and reported.
MITRE ATT&CK Alignment	Helps match findings with threat tactics, techniques, and procedures (TTPs).

🧪 Example Searches on Criminal IP:

1. port:502 AND modbus → Find exposed industrial Modbus devices
2. title:"SCADA" OR title:"HMI" → List of exposed OT interfaces
3. ssl.cert.issuer:"Dahua" → Find Dahua IP cameras with known SSL certs
4. cve:CVE-2021-36260 → Hikvision RCE vulnerability exposure
5. country:IN service:http → HTTP-based devices located in India

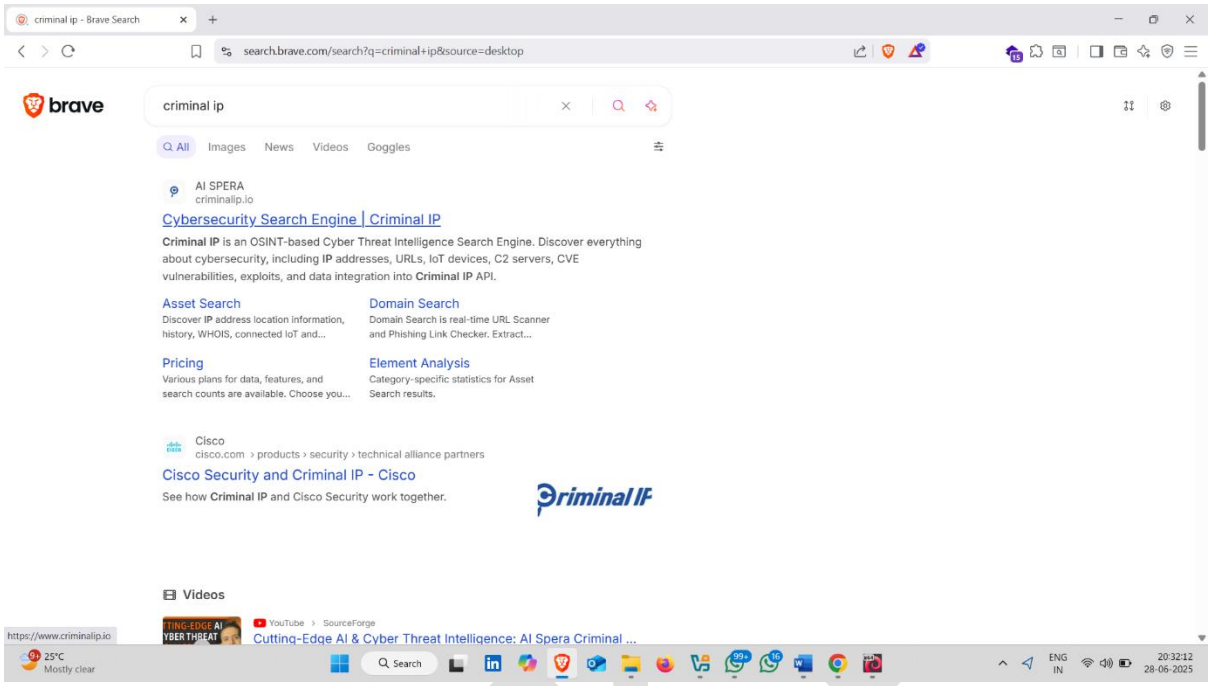
---

🧠 Why Use Criminal IP Alongside Shodan and ZoomEye?

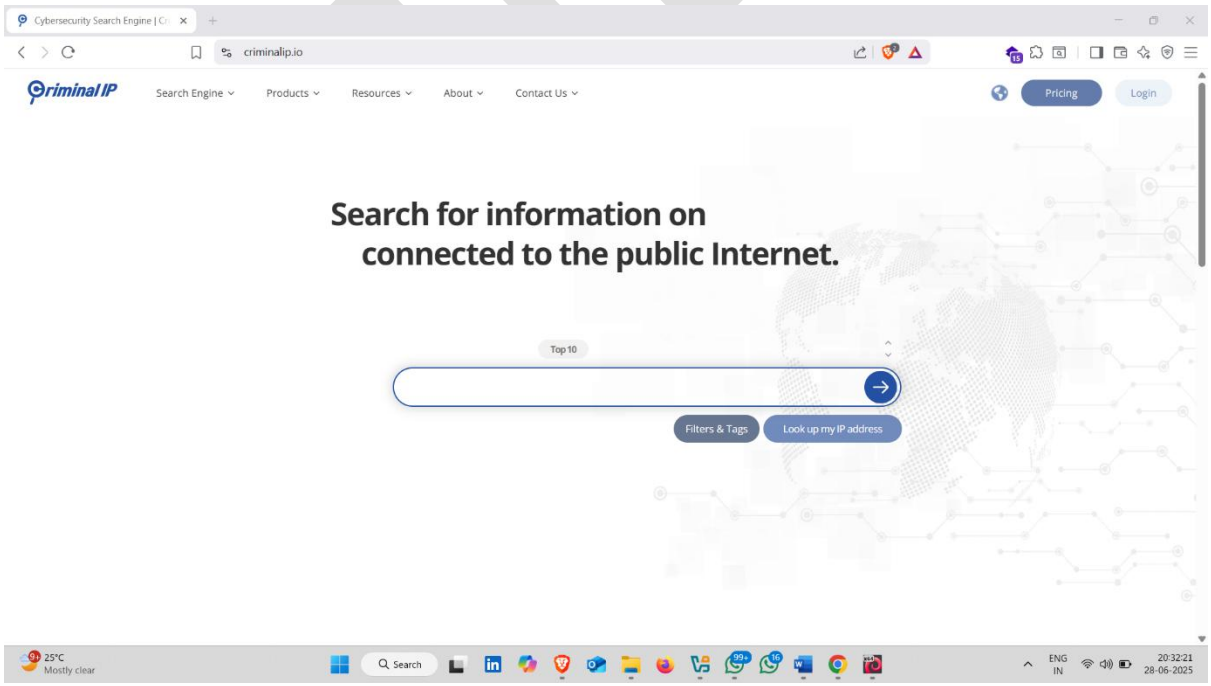
Tool	Strength
Shodan	Fastest for exposed device detection and basic filtering
ZoomEye	Detailed fingerprinting and Chinese network coverage
FOFA	Advanced logic-based asset enumeration
Criminal IP	Best for threat correlation, IP reputation scoring, and dark web linkage

How to use it –:

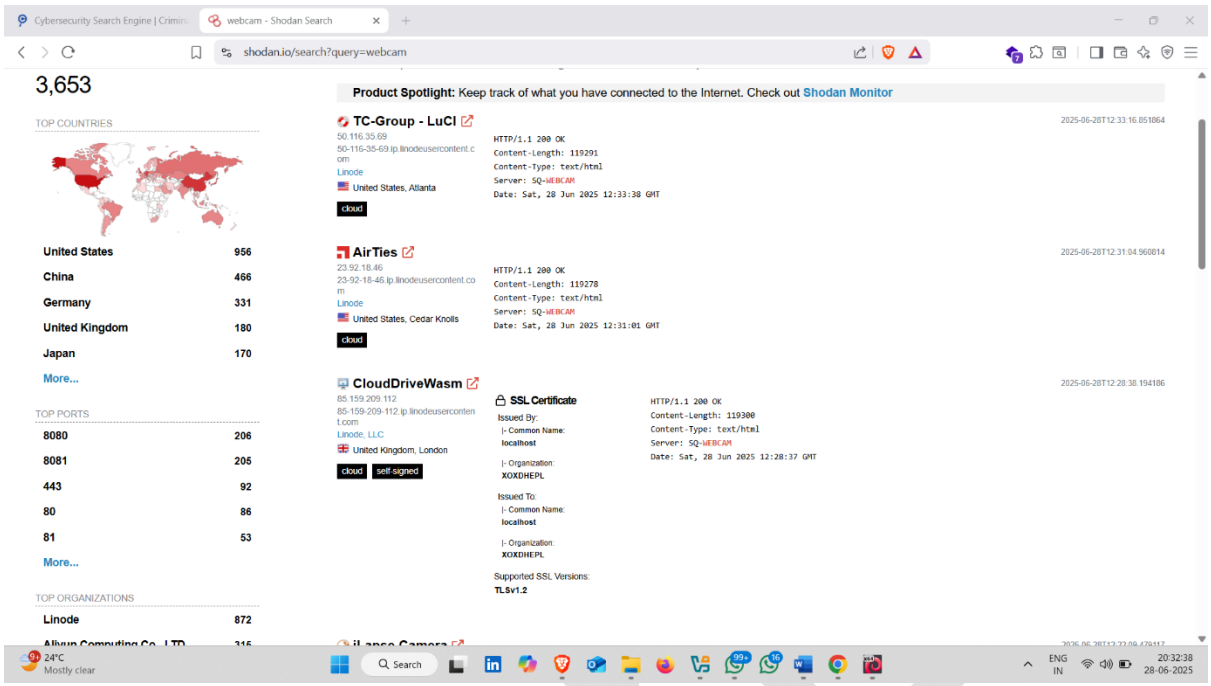
- Open browser and search criminal ip and click on first website



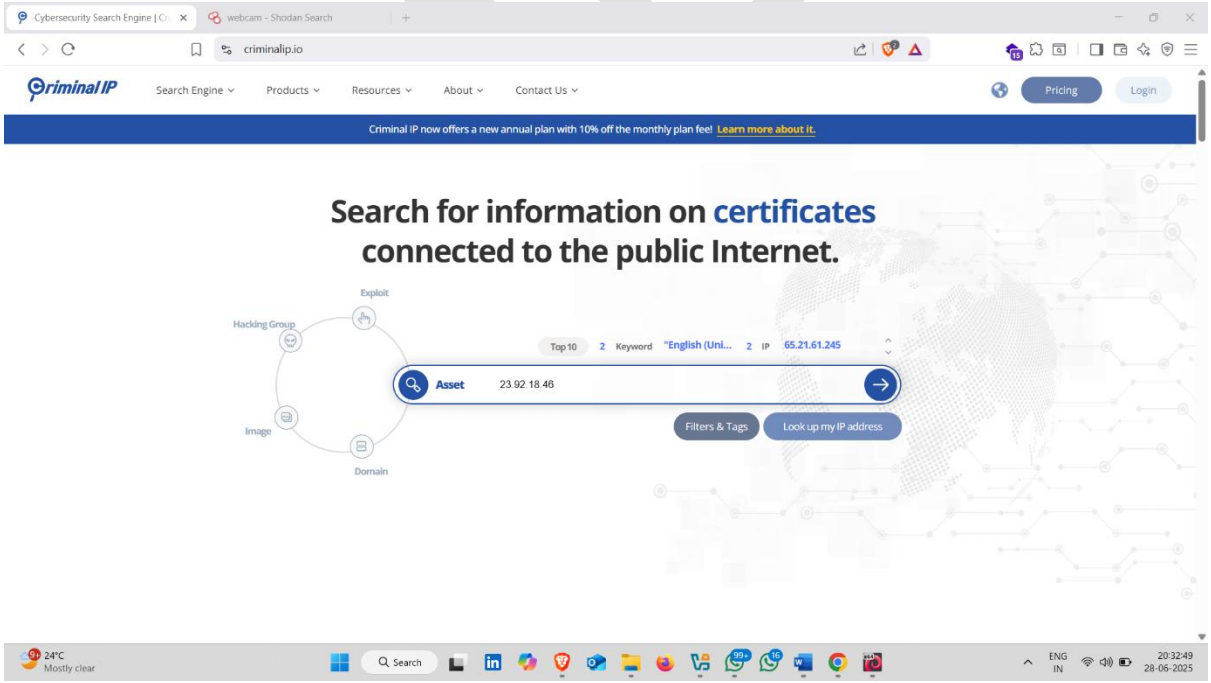
- Enter ip address



- Lets try on one of this ip address 👇 ✅



- **Now search**



- **Result 👇✅**