**Lab 11-01: Gather Information using Online Footprinting Tools**

**Scenario**

As a cybersecurity professional working for SecureNet Solutions, an organization specializing in IoT and OT security assessments, you perform a detailed security audit of a client’s smart infrastructure. The client has reported concerns about potential vulnerabilities in their IoT and OT network, which includes smart sensors, CCTV cameras, and industrial control devices. Your first objective is to gather comprehensive information about the target devices using online footprinting techniques.

**Solution**

As a Certified Ethical Hacker, the company allows you to gather information about IoT and OT devices using online footprinting tools. You use a variety of online sources to get information about the target IoT and OT devices, including Whois domain lookup, advanced Google hacking, and the Shodan search engine. The information acquired can be used to search the devices for vulnerabilities and exploit them for future attacks.

This lab will focus on footprinting the MQTT protocol, a Machine-to-Machine (M2M)/Internet of Things connectivity protocol. It is beneficial for connecting to remote locations that require minimal code footprint or have limited network bandwidth. You may also choose your preferred protocol or device to use for footprinting.

|  |
| --- |
| 1. Turn on your **Windows 10** virtual machines. Then open any web browser and go to the following website using this link: [**https://www.whois.com/whois**](https://www.whois.com/whois).    2. Enter **www.oasis-open.org** in the search field of **Whois Domain Lookup**. Then click on the **SEARCH** button. Oasis is an organization that has released the MQTT v5.0 standard, which marks a considerable improvement in the refinement and functionality of the messaging protocol that now enables IoT.    3. The result shows the following: **Domain Information**, **Registrant Contact**, and **Raw Whois Data**. This information is about the organization that created the MQTT protocol, and it may be useful in tracking revisions and version changes to the target protocol. A Whois lookup gives information on a hostname, IP address, or domain.    4. Open a new tab in a web browser and go to the website using the following link: [**https://www.exploit-db.com/google-hacking-database**](https://www.exploit-db.com/google-hacking-database). The **Google Hacking Database** page opens; enter **SCADA** in the **Quick Search** field and press **Enter**. The result shows the Google dork associated with SCADA.    5. Now, we will use the dorks obtained in the previous step to query results in Google. Open a new tab in the web browser and go to a website using the following link: **https://www.google.com**. In the search field, type **“login” intitle: “scada login”**. The search result appears; scroll down and click on **SEAMTEC SCADA login**.    6. The **SEAMTEC SCADA** login page opens. To get access to the targeted SCADA system, you can brute-force the login credentials.    7. In the Google search field, type **intitle: “index of” scada** and press **Enter** to search sensitive SCADA directories exposed on sites.    8. Open a new tab in the web browser and go to a website using the following link: **https://account.shodan.io/login**. The **Login with Shodan** page opens; enter your **username** and **password** in the **Username** and **Password** fields, respectively. Then click on the **Login** button. If you do not already have an account, click the **Register** button to create one.    9. The **Account Overview** page opens, which shows the account-related information. Click on **Shodan** in the window’s top-left corner to go to Shodan’s main page.    10. The **Shodan** main page opens; enter **port:1883** in the address bar and press **Enter**. Port 1883 is the default MQTT port; IANA defines 1883 as MQTT over TCP. The result shows the IP address list with port **1883** enabled. Click on the first IP address, **34.8.85.30**.    11. Detailed results for the selected IP address appear, displaying information regarding Ports, Services, Hostnames, ASN, etc.    12. Similarly, you may get further information on a target device using the following Shodan filters:   * In the search field, enter **port:502** tosearch for Modbus-enabled ICS/SCADA systems:      * In the search field, enter “**Schneider Electric**” to search for SCADA systems using the PLC name:      * In the search field, enter **SCADA Country: “US”** tosearch for SCADA systems using geolocation:     14. Shodan provides information about SCADA systems used in water treatment plants, nuclear power plants, HVAC systems, electrical transmission lines, home heating systems, and other applications.  15. This concludes the lab on obtaining information on a target device using multiple approaches, such as Whois lookup, advanced Google hacking, and the Shodan search engine. |

**Lab 11-02: Capture and Analyze IoT Traffic using Wireshark**

**Scenario**

As a cybersecurity consultant at CyberSecure Solutions, a firm specializing in IoT and OT security assessments, you are tasked with evaluating the network security of a client’s smart surveillance system. The client has deployed several IoT security cameras that allow remote monitoring and configuration through web interfaces. However, the client has expressed concerns about the security of the data transmitted between these devices and their associated applications.

**Solution**

Your objective is to capture and analyze the network traffic of these IoT devices to identify potential vulnerabilities, such as the use of insecure HTTP protocols or default factory credentials. Using Wireshark, a powerful network analysis tool, you will intercept and inspect data packets flowing between the temperature sensors and their controlling applications.

Wireshark is a free and open-source packet analyzer that helps with network troubleshooting, analysis, software, communications protocol development, and education. It can detect the target operating system and sniff/capture the target computer’s response to the machine from which the request originated.

MQTT is a lightweight messaging protocol that utilizes a publish and subscribe communication model. Designed specifically for low-bandwidth environments, it is well-suited for Machine-to-Machine (M2M) communication and IoT applications. Virtual IoT devices can be created on a virtual network using the Bevywise IoT Simulator on the client side, enabling communication with the server through an MQTT Broker web interface. This interface facilitates data collection and displays the status and messages of connected devices within the network.

**Note: Windows 10** and **Windows Server 2022 are** two virtual machines we are using in this lab. The Windows 10 IP address is **192.168.56.108,** and the Windows Server 2022 IP address is **192.168.56.106**. Do not use these lab IP addresses. Use your virtual machine IP address at the time when you are performing this lab.

|  |
| --- |
| 1. Turn on the **Windows 10** and **Windows Server 2022** virtual machines. Switch to the Windows 10 virtual machine and open any web browser. Go to the following website using this link: [**https://www.bevywise.com/mqtt-broker/download.html**](https://www.bevywise.com/mqtt-broker/download.html). Enter your information in the form, then click on the **Download** button.    2. Go to the Downloads folder and double-click on the **Bevywise\_CrystalMQ\_5.0.exe** file.    3. The CrystalMQ 5.0 setup window appears. Select **I accept the agreement** radio button. Then click on the **Next** button.    4. Leave the destination as the default and click on the **Next** button.    5. Click on the **Create a desktop shortcut** check box and click on the **Next** button.    6. Click on the **Install** button.    7. Click on the **Launch CrystalMQ** check box. Then click on the **Finish** button.    8. If a Windows Security Alert window appears, click the **Allow access** button.    9. The MQTTRoute will execute, and the command prompt window will open. The **TCP** port may be seen using **1883**. We have successfully installed the MQTT Broker in the Windows 10 virtual machine and left the Bevywise MQTT running.    10. To create IoT devices, we must install the IoT simulator on the Windows Server 2022 virtual machine. Switch to the **Windows Server 2022** virtual machine, open any browser, and go to the website using the following link: [**https://www.bevywise.com/iot-simulator/**](https://www.bevywise.com/iot-simulator/). Click on the **Download Now** button. Enter your information in the form and then click on the **Download** button.    11. Go to the **Downloads** folder and double-click on the **Bevywise\_IoTSimulator\_3** setup.    12. The **Bevywise IoTSimulator** setup window appears. Select **I accept the agreement** radio button. Then click on the **Next** button.    13. Click on the **Next** button and leave the destination as the default.    14. Click on the **Create a desktop shortcut** check box and click on the **Next** button.    15. Click on the **Install** button.    16. Click on the **Bevywise\_IoTSimulator\_3** check box. Then click on the **Finish** button.    17. The web interface of the IoT Simulator opens in your default browser. In the IoT Simulator, you can see the default network named **HEALTH\_CARE** and several devices. Now, we will create a virtual IoT network and devices. Click on the **menu** icon. Then select the **+New Network** option.    18. In the **Create New Network** popup, enter **CEH\_FINACE\_NETWORK** in the name field and enter your description in the **Description** box. Then click the **Create** button.    19. In the **Simulator Setting,** set the **Broker IP Address** as **192.168.56.108,** the IP address of the **Windows 10** virtual machine. Since we installed the Broker on the web server, the established network will communicate with the server via the MQTT Broker. Leave all the settings as default and click on the **Save** button.    20. Click on the **Yes** button to proceed with the network creation.    21. Click **OK** to complete the IoT virtual network creation process.    22. Click the **Add blank Device** button to add IoT devices to the created network.    23. In the **Create New Device** popup, enter **Temperature\_Sensor** in the **Device Name**. In the **Device Id,** enter **TS1**. Enter your description in the **Description** box. Click on the **Save** button.    24. The device will be included in the **CEH\_FINANCE\_NETWORK**. Click the **Start Network** red circular icon in the right corner to connect the **Network** and the added devices to the server or Broker.    25. The red button changes to **green** when a connection is established between the network, the additional devices, and the web server or the MQTT Broker.    26. Switch to the **Windows 10** virtual machine. Open any web browser and go to **http://localhost:8080.**    27. Click on the **Dashboard;** you will observe that the connection request from virtual machine **192.168.56.106** for the device **TS1** is under the **Client Log** section since the **Broker** was left running.    28. Switch back to the **Windows Server 2022** virtual machine. Click on the **Plus** icon in the top right corner and select the **Subscribe to Command** option to create the **Subscribe** command for the device **Temperature\_Sensor.**    29. In the **Subscribe for command – TS1** popup, select **On start** in the **Subscribe on**. In the **Topic,** select **High\_Temp**. In **QoS,** select **1 Atleast once**. Click on the **Save** button.    30. Scroll down; the **Topic** is under the **Subscribe to Commands** section.    31. Next, we will monitor the traffic between the **virtual IoT network and the MQTT Broker** to ensure a safe connection. Open the **Wireshark** software and select the **Ethernet** network adapter. A network adapter should have a different name when doing this lab in your virtual environment. Click the **Start Wireshark** icon to begin gathering packets; leave Wireshark running. Leave the IoT simulator running.    32. Switch to the **Windows Server 10** virtual machine. Navigate to the web browser, go to the **Crystal MQ** dashboard, and then click on the **Clients.** Then click on the **TS1** device.    33. Click on the **Send command** tab. Now, we will send the message to **TS1** via the **High\_Tempe** topic. In the **Send Command** section, enter **High\_Tempe** in **Topic.** Then, enter **Alert for High Temperature** in the **Message** field. Click on the **Send** button.    34. Below the **Message** box, the **Message sent to TS1** appears, which shows that the message was successfully delivered to TS1.    35. Switch to the **Windows Server 2022** virtual machine. We have kept the IoT simulator running in the web browser. To see the alert message, expand the arrow of the **Device Log** under the connected **Temperature\_Sensor**. You can explore the alert message "**Alert for High Temperature**".    36. Switch to the **Wireshark** traffic capturing window to verify the communication. Enter **mqtt** under the filter field and press **Enter**. To display only the **MQTT** protocol packets.    37. Choose any **Publish Message** packet from the **Packet List** pane. Expand the **Transmission Control Protocol**, **MQ Telemetry Transport Protocol**, and **Header Flags** nodes in the **Packet Details** pane at the center of the window.  38. Under the **MQ Telemetry Transport Protocol** nodes, you can analyze details such as **Msg Len**, **Topic Length**, **Topic**, and **Message**.  39. Publish Message may be used to get the message submitted by the MQTT client to the broker.    40. Choose any **Publish Release** packet from the **Packet List** pane. Expand the **Transmission Control Protocol**, **MQ Telemetry Transport Protocol**, and **Header Flags** nodes in the **Packet Details** pane at the center of the window.  41. In the **MQ Telemetry Transport Protocol** nodes, you can analyze details such as **Msg Len**, **Message Type**, and **Message Identifier**. A Publish Release (PUBREL) packet is in action to a Publish Received (PUBREC) packet.    42. Now, scroll down, look for the **Publish Complete** packet from the **Packet List** pane, and click on it. Expand the **Transmission Control Protocol**, **MQ Telemetry Transport Protocol**, and **Header Flags** nodes in the **Packet Details** pane at the center of the window.  43. You can analyze details such as **Msg Len** and **Message Identifier** in the **MQ Telemetry Transport Protocol** nodes. The Publish Complete (PUBCOMP) packet replies to the Publish Release (PUBREL) packet.    44. Now, scroll down, look for the **Publish Received** packet from the **Packet List** pane, and click on it. Expand the **Transmission Control Protocol**, **MQ Telemetry Transport Protocol**, and **Header Flags** nodes in the **Packet Details** pane in the middle of the window.  45. You may examine data like **Message Type**, **Msg Len**, and **Message Identifier** in **MQ Telemetry Transport Protocol** nodes.    46. Similarly, you can choose **Ping Request**, **Ping Response**, and **Publish Ack** packets and observe the details.  47. This completes the lab of capturing and analyzing MQTT protocol packets. We used Wireshark to evaluate the various procedures involved in the communication between an MQTT client and a broker. Understanding these KPIs, as well as the workflow, can assist you in swiftly detecting MQTT-related issues. |

**Lab 11-03: Perform Replay Attack on CAN Protocol**

**Scenario**

As a cybersecurity consultant at TechSecure Analytics, an organization specializing in IoT and OT penetration testing, you have been tasked with assessing the security vulnerabilities of a client’s connected vehicle systems. The client’s automotive IoT devices, which utilize the Controller Area Network (CAN) protocol for internal communication, have raised concerns about potential exploitation.

**Solution**

As a Certified Ethical Hacker, your objective is to simulate a replay attack on the CAN protocol to demonstrate the risks associated with inadequate security measures. This involves capturing legitimate CAN messages transmitted between the vehicle’s Electronic Control Units (ECUs) and replaying these messages to manipulate the vehicle’s behavior. By leveraging tools and techniques tailored for CAN traffic analysis, you will identify the impact of weak authentication mechanisms and a lack of proper encryption in the communication protocol.

The Controller Area Network (CAN) protocol is a highly reliable communication system that facilitates interaction between microcontrollers and devices without requiring a central computer. It employs a message-based communication approach to ensure dependable data exchange, even in high-noise environments. Widely utilized in the automotive industry for its simplicity and efficiency, the CAN protocol performs an important function. Modern vehicles enable seamless communication between critical systems such as engine controls, braking systems, and infotainment units. However, this extensive interconnectivity can be exploited by malicious actors to manipulate vehicle functions, potentially compromising safety.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. Turn on the **Ubuntu** virtual machine. Open the **Terminal** and execute the **sudo su** command to run programs with root privileges. Then, execute the following command: **sudo apt-get install can-utils** to install CAN protocol. While installing, if prompted **Do you want to continue?**, type **Y** and press **Enter**.    2. Execute the following command: **git clone <https://github.com/zombieCraig/ICSim.git>** to clone the GitHub repository of ICSim.    3. To setup a virtual CAN interface, execute the below-provided commands:   * **sudo modprobe can** * **sudo modprobe vcan** * **sudo ip link add dev vcan0 type vcan** * **sudo ip link set up vcan0**     4. Execute the **ifconfig** command to verify whether the virtual CAN interface is correctly established. The **vcan0** interface is present here, indicating that our Virtual CAN interface was successfully configured.    5. Execute the following command: **chmod -R 777 ICSim** to provide permissions to the ICSim folder.  6. Execute the **cd ICSim** command to go to the ICSim directory. Then execute the **make** command to create two executable files for IC Simulator and CANBus Control Panel.    7. Execute the **./icsim vcan0** command to start the ICSim simulator. The IC Simulator window appears.    8. Open a new terminal window and execute **sudo su** command. Then, execute the **cd ICSim** command to go to the ICSim folder.  9. Execute the **./controls vcan0** command to start the CANBus Control Panel. CANBus Control Panel window appears.    10. Now, we will start a sniffer to capture the traffic sent to the ICSim Simulator by the CANBus control panel simulator. To do so, open a new terminal window and execute **sudo su** command. Execute the **cd ICSim** command to go to the ICSim folder.  11. Execute the following command: **cansniffer -c vcan0** to start sniffing on the vcan0 interface. Leave this sniffer on.    12. Open a new terminal window and execute **sudo su** command. Then execute **cd ICSim** to go to the ICSim folder. Execute the following command: **candump -l vcan0** to capture the logs.  13. After collecting the logs, open the ICSim and Controller simulator and perform actions like acceleration, turning left/right, and opening and closing doors to produce logs. When you are done, terminate the continuing activity by hitting **Ctrl + C**. Use the below-provided keys in a table to perform various functions.   |  |  | | --- | --- | | **ICSim Functions** | **Keys** | | Accelerate | Up arrow | | Left/Right Turn | Left arrow/ Right arrow | | Unlock Rear Left/Right doors | Right Shift + X / Right Shift + Y | | Unlock Front Left/Right doors | Right Shift +A / Right Shift + B | | Lock all doors | Hold Right Shift key + Tap Left Shift | | Unlock all doors | Hold Left Shift key + Tap Right Shift |   14. Execute the **ls** command to verify that the log file is created. The **.log** file has been successfully generated.      15. Open a new terminal window and execute **sudo su** command. Execute the **cd ICSim** command to go to the ICSim folder. Execute **ls** command to list the files in the directory. Execute the following command: **canplayer -I candump-2024-12-11\_083827.log** to perform a replay attack. Once the log file has been run, you may view the movements made while producing it in real-time in the IC Simulator and CANBus control panel simulator. The log file name may change when doing the lab. |