Exploring Security Vulnerabilities in Smart Health IoT Devices: A Case Study Using Wireshark and Nmap Analysis with DDoS Attack Evaluation

Yogadisha Sendhil Kumar
Department of Computer and Information Technology
Purdue University
West Lafayette, Indiana, USA
ysendhil@purdue.edu

***Abstract*—This research investigates security vulnerabilities in smart health IoT devices, employing Wireshark and Nmap analysis alongside a DDoS attack evaluation. Initially focusing on medical IoT using a pulse oximeter, the study transitioned to utilizing a humidity sensor within the realm of smart health due to compatibility issues with the pulse oximeter's connectivity to an ESP32 device for AWS integration. Humidity sensors are integral components in smart health applications, contributing to respiratory health monitoring and wound care management. In the context of respiratory health, these sensors are integrated into home monitoring systems to track indoor humidity levels, alerting individuals or caregivers to suboptimal conditions associated with respiratory conditions such as asthma or COPD. Similarly, in wound care, humidity sensors are incorporated into devices or smart bandages to monitor moisture levels in the wound environment, facilitating real-time feedback to healthcare providers or patients to maintain appropriate humidity levels conducive to healing and infection prevention.This research delves into exploring DDoS attacks and other vulnerabilities in smart health IoT devices, using humidity sensors as a primary use case in the smart health domain.**

***Index Terms*—Smart Health, IoT Devices, Security Vulnerabilities, DDoS Attacks, Wireshark Analysis, Nmap Analysis, Pulse Oximeter, Humidity Sensor, Respiratory Health Monitoring, Wound Care Management, ESP32 Device, AWS Integration**

I. Introduction

The Internet of Things (IoT) refers to a network of physical devices, vehicles, appliances, and other objects embedded with sensors, software, and network connectivity, enabling them to collect and share data [1]. IoT devices—also known as "smart objects"—span a wide spectrum, from simple "smart home" devices like thermostats to wearables like smartwatches and complex industrial machinery [1]. Technologists envision entire "smart cities" built on IoT technologies, where interconnected devices autonomously exchange data and perform tasks [1]. However, the widespread adoption of IoT technologies presents significant security challenges. IoT security is complex due to inherent vulnerabilities in devices and systems. Challenges include a lack of visibility, as users often deploy devices without IT department oversight, leading to difficulties in inventory management and monitoring. Integrating IoT devices into security systems is hindered by their diverse nature and scale. Moreover, reliance on open-source code in firmware increases the risk of bugs and vulnerabilities. The sheer volume of data generated by IoT devices complicates data oversight and protection. Inadequate testing and unpatched vulnerabilities leave systems exposed to exploitation, while vulnerable APIs and weak passwords provide entry points for cybercriminals [2].

In today's interconnected world, IoT has revolutionized various industries, including healthcare, promising improved efficiency, accessibility, and patient care. However, rapid adoption of IoT technologies also brings significant security challenges. Smart health IoT devices, from wearable monitors to hospital equipment, are increasingly targeted by malicious actors exploiting vulnerabilities. These vulnerabilities expose sensitive healthcare data, compromise patient privacy, and pose risks to healthcare services' integrity and availability. Understanding and addressing security vulnerabilities in smart health IoT devices are crucial to safeguarding patient safety and maintaining trust in healthcare systems.

This paper delves into the intricate landscape of security vulnerabilities in smart health IoT devices, presenting a comprehensive analysis of emerging threats and attack vectors. Through a case study, we explore the utilization of Wireshark and Nmap analysis techniques to assess smart health IoT ecosystems' security posture. Furthermore, we examine the implications of Distributed Denial of Service (DDoS) attacks on healthcare infrastructure, highlighting potential ramifications and mitigation strategies. By shedding light on the evolving threat landscape and providing practical insights into vulnerability assessment methodologies, this research aims to contribute to robust security frameworks for smart health IoT environments.

II. Use Case

Initially focused on medical IoT using a pulse oximeter, the study shifted to incorporating a humidity sensor within the smart health domain due to compatibility challenges with the pulse oximeter's integration with an ESP32 device for

AWS connectivity. Humidity sensors play a vital role in smart health applications, particularly in respiratory health monitoring and wound care management. In respiratory health, these sensors are integrated into home monitoring systems to monitor indoor humidity levels, alerting individuals or caregivers to suboptimal conditions associated with conditions like asthma or COPD. Similarly, in wound care, humidity sensors are integrated into devices or smart bandages to monitor moisture levels in the wound environment, providing real-time feedback to healthcare providers or patients to maintain optimal humidity levels for healing and infection prevention.

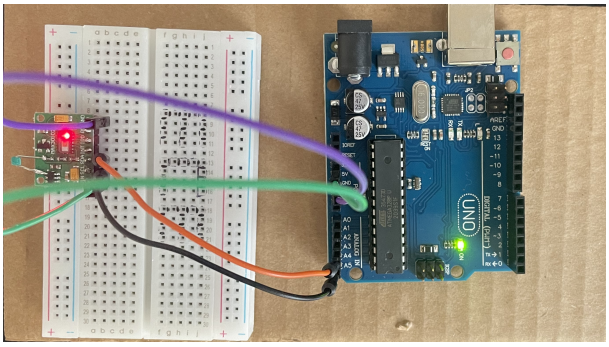


Fig. 1. Pulse Oximeter MAX30102 Setup

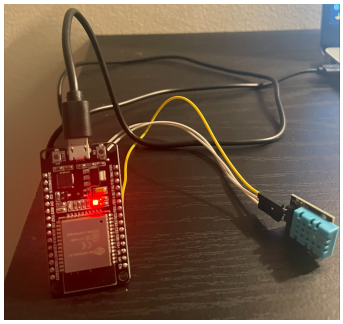


Fig. 2. Humidity Sensor DHT11 Setup

As mentioned earlier the study focused on integrating a pulse oximeter as the IoT usecase. However, upon starting, the first issue encountered was a significant design flaw in the MAX30102 pulse oximeter chip intended for use. This flaw hindered its effective functioning. To address this, the study explored methods to rectify the problem. One solution involved disconnecting the 4.7k ohm pull-up resistors from the 1.8V supply voltage and reconfiguring them to pull up to 3.3V. This adjustment aimed to mitigate the design flaw and improve the chip's performance. However, despite efforts to resolve the issue, the study encountered additional compatibility issues between the pulse oximeter and the ESP32 device, particularly regarding voltage connectivity. Consequently, the decision was made to pivot the focus within the smart health domain, integrating a humidity sensor, specifically the DHT11, into the IoT framework. This shift allowed for overcoming the challenges posed by the pulse oximeter's compatibility issues and continuing the exploration of smart health applications effectively.

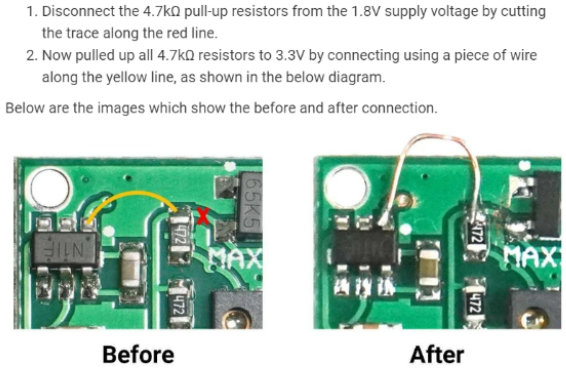


Fig. 3. Correction of design issue in MAX30102 sensor

III. Literature Survey

A. Hands-On Denial of Service Lab Exercises Using Slowloris and RUDY

The paper "Hands-On Denial of Service Lab Exercises Using Slowloris and RUDY" presents an interactive exercise aimed at teaching students about offensive denial of service techniques commonly employed by hackers. Through practical exercises conducted on a Java-based graphical interface, students learn how denial of service (DoS) attacks work and how to defend against them. The exercise focuses on two specific DoS attacks—Slowloris and RUDY—and encourages students to enhance their analytical skills while gaining insights into TCP, HTTP, and application-layer security considerations.

A critical aspect highlighted in the paper is the need for a safe environment for students to engage in hands-on experience with DoS attacks. By providing a controlled laboratory setting, educators can facilitate effective learning while addressing ethical considerations associated with offensive techniques. The exercises emphasize both offensive and defensive aspects of cybersecurity, encouraging students to explore mitigation techniques alongside attack strategies.

The paper discusses various tools and techniques used in the lab exercises, drawing from existing tutorials and pedagogical resources developed by security professionals and educators. It underscores the importance of ease of deployment for instructors and the need for exercises that foster analytical skills while addressing real-world security challenges.

Furthermore, the paper outlines the technical aspects of Slowloris and RUDY attacks, highlighting vulnerabilities in the HTTP protocol stack exploited by these attacks. It describes the implementation of the lab exercises using virtual machine images, which offer flexibility and accessibility in academic environments.

Through student engagement in the exercises, educators observed improved understanding of system behavior and basic principles of DoS attacks. The exercises provided a balance between comprehensibility and complexity, offering students both a graspable introduction to the topic and opportunities for deeper exploration of unresolved issues and challenges in DoS mitigation.

Overall, "Hands-On Denial of Service Lab Exercises Using Slowloris and RUDY" contributes valuable insights into teaching offensive cybersecurity techniques in an educational setting while emphasizing ethical considerations and practical skill development[4].

B. Performance Comparison and Analysis of Slowloris, GoldenEye, and Xerxes DDoS Attack Tools

The paper titled "Performance Comparison and Analysis of Slowloris, GoldenEye, and Xerxes DDoS Attack Tools" examines Denial-of-Service (DoS) attacks, which aim to disrupt network services by overwhelming servers with unsolicited messages. It highlights the severity of Distributed Denial of Service (DDoS) attacks, citing instances such as the 2018 attack with a traffic rate of 1.35 terabits per second and the 2016 Mirai botnet attack with 1.1 terabits per second. The study evaluates three DDoS attack tools—Slowloris, GoldenEye, and Xerxes—based on their performance metrics and impact on network resources.

The paper emphasizes the urgent need for effective countermeasures against DDoS attacks, given their significant threat to large networks. It proposes criteria for characterizing DDoS attacks and discusses defense mechanisms, including entropy and frequency-based detection methods. Various studies by researchers like Kulkarni, Bush, Marciel, and Nagy are cited for their contributions to DDoS attack detection. The study aims to provide insights into the traffic patterns generated by modern DDoS attacker tools and compares the experimental results of Slowloris, GoldenEye, and Xerxes.

DDoS attack tools like GoldenEye exploit open connections to overwhelm web servers, while Slowloris utilizes partial HTTP connections to exhaust server resources. Xerxes, on the other hand, is noted for its rapid attack launch time compared to Slowloris and GoldenEye. The paper also discusses different DDoS attack types, architectures, and mitigation techniques, underscoring the importance of understanding and defending against these attacks in network security.

In summary, "Performance Comparison and Analysis of Slowloris, GoldenEye, and Xerxes DDoS Attack Tools" provides valuable insights into the capabilities and impact of various DDoS attack tools, highlighting the need for robust defense strategies to mitigate their effects on network infrastructure and services[5].

C. Statistical Approaches to DDoS Attack Detection and Response

The paper titled "Statistical Approaches to DDoS Attack Detection and Response" delves into the intricacies of combating Distributed Denial of Service (DDoS) attacks on large networks, such as the Internet. It explores various methods proposed to identify DDoS attacks, including computing entropy and applying Pearson's chi-square test to packet attributes. These approaches aim to detect anomalies in network traffic indicative of DDoS attacks accurately and efficiently.

Several studies have explored different techniques to address the challenges posed by DDoS attacks. Notably, the paper discusses computing entropy values of specific packet header fields to measure the randomness and uniformity of source IP addresses. Additionally, it explores the application of Pearson's chi-square test to compare distributions of discrete values, such as TCP SYN flag values or protocol numbers. These techniques have shown promise in effectively detecting DDoS traffic and mitigating its impact on downstream network resources.

The authors of "Statistical Approaches to DDoS Attack Detection and Response" have also implemented prototype detector modules for popular network intrusion detection systems like Snort. These modules enable real-time monitoring and analysis of network traffic by logging entropy values and chi-square statistics for each packet attribute. Such functionalities facilitate manual or automatic tuning of detection parameters, as discussed in the paper.

Evaluation of DDoS attack detectors has been a focal point of research. Researchers have exposed these detectors to various network environments and simulated attack traffic to assess their effectiveness. By overlaying existing DDoS attack tools' traffic onto network traces, they evaluate the detectors' ability to identify and mitigate attacks. While detection methods based on entropy and chi-square analysis show promise, the paper acknowledges challenges in accurately characterizing attack traffic and minimizing false positives.

The response mechanisms discussed in the paper play a crucial role in mitigating DDoS attacks' impact. These mechanisms involve classifying packets as benign or suspect based on detection results and applying defensive measures accordingly. Filtering and rate-limiting packets using techniques such as netfilter and Linux Advanced Routing and Traffic Control (LARTC) are highlighted. Future research aims to enhance the integration of detection and response algorithms, enabling more focused filtering and reducing the impact of responses on legitimate network traffic.

In summary, "Statistical Approaches to DDoS Attack Detection and Response" provides valuable insights into the ongoing efforts to detect and respond to DDoS attacks. While significant progress has been made, the paper emphasizes

the need for continued research to improve the accuracy and efficiency of detection methods and enhance the effectiveness of response mechanisms in mitigating attack effects[3].

IV. Methodology

A. Network analysis tools

Ethical hacking, also known as penetration testing, involves authorized intrusion into systems or networks to identify potential threats and data breaches. White hat hackers, or ethical hackers, possess various skill sets to hack into devices and uncover system vulnerabilities. IoT hacking/ analysis tools are software programs used to exploit computer security or networks, helping to identify security patches and mitigate system threats. As cyberattacks continue to rise due to the increasing reliance on cloud-based infrastructure and IoT technology, ethical hacking plays a crucial role in safeguarding organizations' digital assets. The IoT hacking/ analysis tools aid in detecting security issues and preventing data theft, making them essential components of a company's security strategy in today's digital landscape[6].

1) Wireshark: Wireshark is a powerful network packet sniffing tool that captures and analyzes network traffic in real-time. It allows users to examine individual data packets and understand how they traverse the network by searching and filtering for them. As an open-source program, Wireshark is crucial for network diagnostics and security, utilized by administrators, IT professionals, and penetration testers alike. Its benefits include cross-platform compatibility, detection of traffic issues, and the ability to decode data transmitted by others[6].

Even with multiple analysis tools available Wireshark stands as a cornerstone in the realm of network analysis tools. With its capability to capture and dissect network traffic in real-time, Wireshark provides immediate insights into network behavior, facilitating the swift detection and resolution of potential issues. Its extensive protocol support ensures that users can analyze a wide range of network traffic types, from standard TCP/IP protocols to more specialized ones like HTTP and DNS. Moreover, Wireshark's powerful filtering capabilities empower users to focus their analysis on specific packets or protocols of interest, streamlining the troubleshooting process. The tool's detailed packet analysis features provide granular information about each captured packet, including source and destination addresses, protocol headers, and payload data, enabling thorough diagnosis of network problems and security threats. Wireshark's cross-platform compatibility ensures accessibility across various operating systems, making it a versatile solution for network analysis needs. Furthermore, being open-source and freely available, Wireshark democratizes network analysis, allowing users to harness its capabilities without financial constraints. Supported by a vibrant user community and comprehensive documentation, Wireshark continues to serve as an indispensable tool for network administrators, security professionals, and enthusiasts alike, empowering them to maintain robust and secure network infrastructures.

In addition to its fundamental role in network diagnostics and security, Wireshark proved invaluable in visualizing and analyzing Distributed Denial of Service (DDoS) attacks within our study. Leveraging Wireshark's robust capabilities helped gain valuable insights into the nature and impact of DDoS attacks on our network infrastructure which is discussed later on in this paper. Its ability to capture and dissect network traffic in real-time provided a comprehensive view of the attack vectors and their effects on network performance. Wireshark's intuitive interface and detailed packet analysis features facilitated the identification of malicious traffic patterns. Overall, Wireshark emerged as an indispensable tool in our arsenal, empowering us to effectively visualize the DDoS attack.

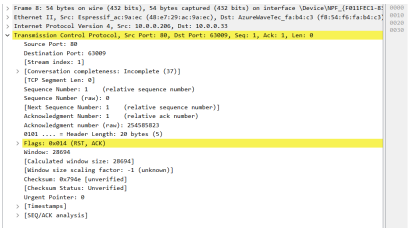


Fig. 4. Packet capture

2) NMap: Nmap, or Network Mapper, is a widely-used open-source scanning tool designed for network discovery. It assists network administrators in identifying available hosts, open ports, and potential security vulnerabilities. Operating across various platforms, Nmap is favored for its user-friendly interface and ability to scan a single host for up to 1,000 ports. Ethical hackers leverage Nmap's scripting engine to interact with target hosts and uncover system vulnerabilities during scanning. Key benefits include host identification, scripting capabilities, threat detection, and network auditing for server discovery[6].

Running a scan in Zenmap, the graphical user interface for Nmap, allows users to conduct network scans with ease and efficiency. By specifying target IP addresses, scan types, port ranges, and timing options, users can initiate scans tailored to their specific needs. Zenmap then sends packets to the specified targets, analyzes the responses received, and generates detailed reports summarizing the scan results. These reports include information about open ports, services running on those ports, detected operating systems, and potential vulnerabilities. Overall, Zenmap simplifies the process of network reconnaissance, enabling users to identify active hosts, discover open ports, and assess potential security risks through a user-friendly interface. As we can see below in the 5 when we give an IP Address in the Target section Zenmap gives the report. The Zenmap scan results indicate that all 1000 scanned ports on the target machine with IP address 10.0.0.206 are closed. This means that there are no open ports that can be

accessed externally. The scan was conducted using an “Intense scan” mode, and it completed in approximately 8 seconds. The host at 10.0.0.206 responded with zero latency, indicating that it is up and reachable. The closed ports suggest that the machine is well-protected and not exposing any services to the network.

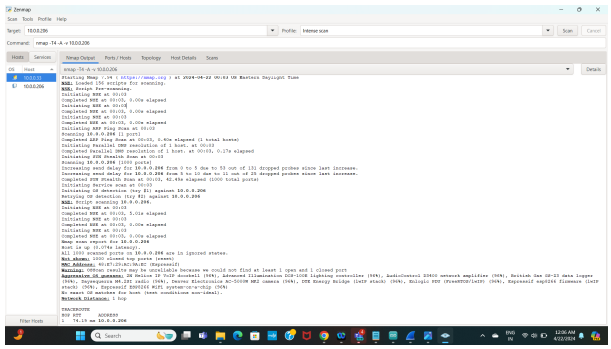


Fig. 5. Zenmap analysis

Now to scan a hostname, simply run the command nmap IP Address, as you see below in 6 [7]:

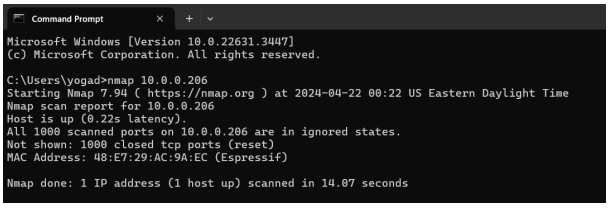


Fig. 6. Basic Nmap Scan against IP

The most famous type of scan is the Nmap ping scan (so-called because it’s often used to perform Nmap ping sweeps), and it’s the easiest way to detect hosts on any network. To run Nmap ping use the command nmap -sn IP Address as shown below in 7 [7].

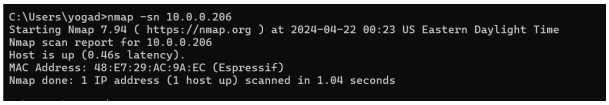


Fig. 7. Nmap ping scan

The command ”nmap -p 1-65535 localhost” performs a comprehensive port scan on the local machine (localhost) as shown below in 8. It scans all 65,535 TCP ports (1 to 65,535) to determine which ports are open, closed, or filtered on the localhost. This scan provides a thorough examination of all TCP ports on the local system, helping to identify potential services or applications running on different ports.

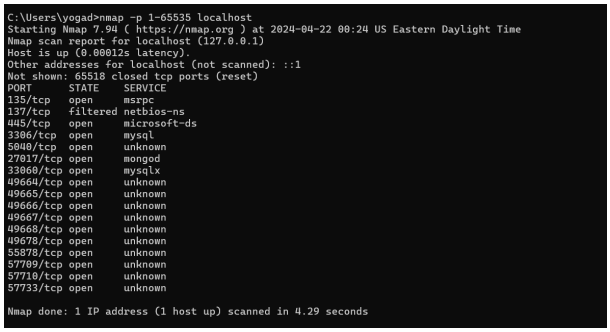


Fig. 8. Scan entire port ranges

Using “–top-ports” parameter along with a specific number lets scan the top X most common ports for that host as shown below in 9. This command is useful for quickly identifying potential services or applications running on the target system without scanning all possible ports, thereby saving time and network resources.

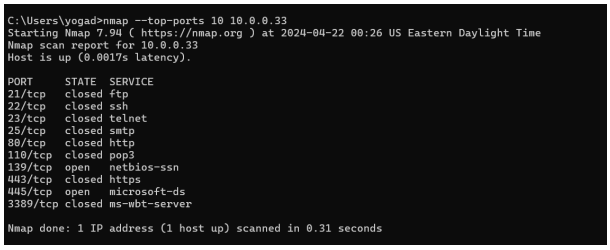


Fig. 9. Scan the most popular ports

The command ”nmap -sV localhost” as shown below in 10 instructs Nmap to conduct a service version detection scan (-sV) on the local system (localhost). This command aims to identify the versions of services running on various ports on the local machine, providing information about the software and its associated versions.

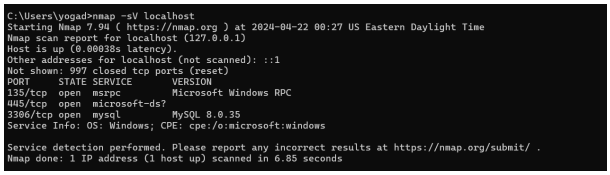


Fig. 10. Detect service/daemon versions

The command ”nmap –script ftp-brute -p 21 10.0.0.206” as shown below in 11 directs Nmap to conduct an FTP brute-force attack against the target host, specifically targeting port 21, which is commonly associated with FTP (File Transfer Protocol) services. This command utilizes the ”ftp-brute” script in Nmap’s scripting engine to attempt to guess valid usernames and passwords for the FTP service running on the specified port, potentially identifying vulnerabilities or weak authentication credentials.

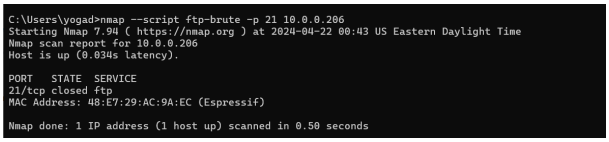


Fig. 11. Testing brute force attacks

The command ”nmap -sV –script=http-malware-host 10.0.0.206” as shown below in 12 instructs Nmap to perform a service version (-sV) scan on the target host, while also executing the ”http-malware-host” script. This script is designed to detect if the target is hosting known malware through HTTP (Hypertext Transfer Protocol). By combining service version detection with the malware-host script, the command aims to identify potential malware infections on web servers or other HTTP services running on the specified host.

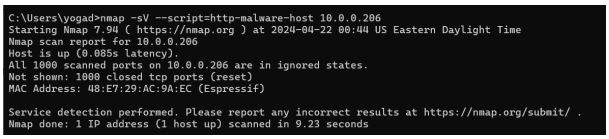


Fig. 12. Detecting malware infections

V. DDoS Attack

Slowloris is a type of DDoS (Distributed Denial of Service) attack tool designed to overwhelm web servers by exhausting their resources, such as available connections. Unlike traditional flooding attacks that use a high volume of traffic, Slowloris works by sending partial HTTP requests to web servers, which are then kept open by sending occasional header packets. By holding these connections open for as long as possible, Slowloris consumes the server’s resources, preventing it from serving legitimate requests. This attack is effective against web servers that allocate resources per connection, as it can be executed with relatively low bandwidth and from a single machine. Below shown is the python code used for the attack[8].

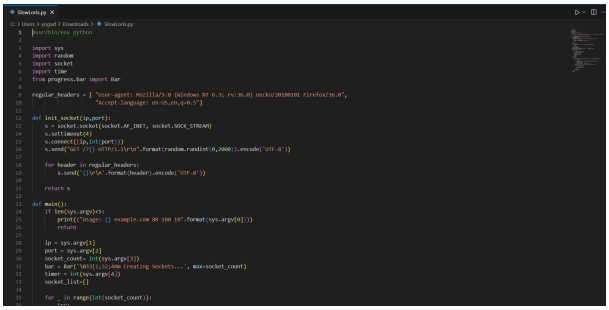


Fig. 13. SlowLoris python code

The command ”python slowloris.py 10.0.0.206 80 100 10” was run on command prompt to initiate the script to run the attack as shown below in 14.

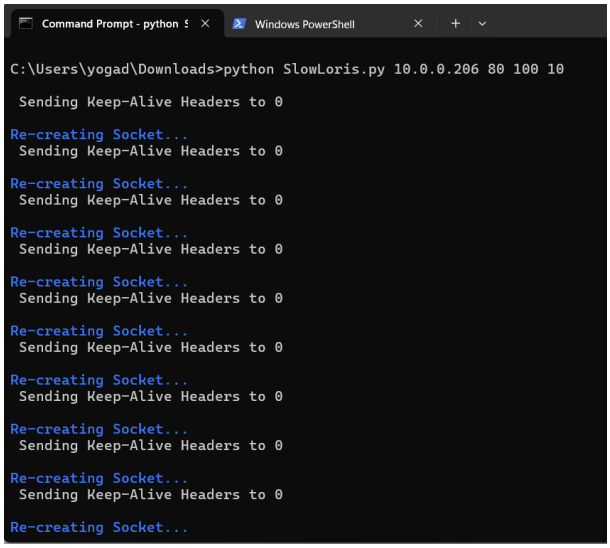
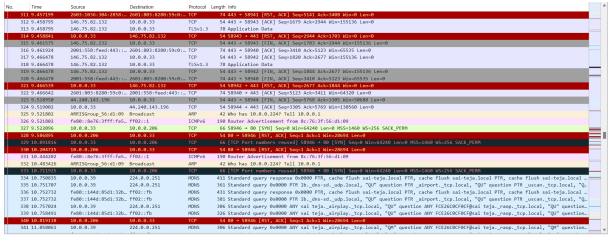


Fig. 14. Command line prompt

Wireshark was used to analyse the attack. After running Slowloris, most packets were observed as pink,red or black in Wireshark as shown in V. This colour marking indicates that Wireshark’s packet analysis algorithms have identified the packets as potentially malicious or suspicious. Slowloris, a tool that sends partial HTTP requests to keep connections open, triggers Wireshark’s analysis to flag these packets due to their abnormal behavior. While the observed coloration doesn’t inherently denote harm, it signifies that the packets exhibit characteristics Wireshark recognizes as potentially concerning.



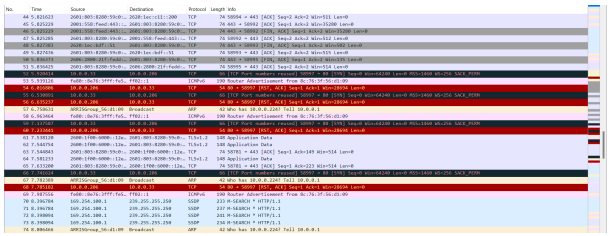


Fig. 15. SlowLoris Analysis on Wireshark

VI. Conclusion

In summary, this research paper delves into the critical realm of smart health IoT security, focusing on the vulnerabilities inherent in devices like humidity sensors. Through a meticulous

exploration of Wireshark and Nmap analysis methods coupled with an evaluation of DDoS attacks, the study aims to scrutinize the security landscape of smart health IoT ecosystems. Commencing with an overview of IoT security challenges, it underscores the imperative of addressing vulnerabilities to uphold patient safety and bolster trust in healthcare systems.

By surveying relevant literature, the paper contextualizes its methodology within the broader cybersecurity discourse. Notable discussions encompass hands-on denial of service lab exercises, comparisons of DDoS attack tools, and statistical approaches to detecting and responding to DDoS attacks. These insights set the stage for a robust methodology employing Wireshark and Nmap tools to scrutinize the security of smart health IoT devices.

The methodology section elucidates the use of Wireshark for packet analysis and Nmap for network scanning, offering a detailed exposition of various Nmap commands. These demonstrations underscore the versatility and efficacy of Nmap in identifying potential security threats in IoT devices. Additionally, a practical case study employing a Slowloris DDoS attack sheds light on real-world security vulnerabilities in smart health IoT ecosystems.

In conclusion, this research paper offers significant contributions to the understanding of IoT security, through the utilization of advanced analysis techniques and practical evaluations.

REFERENCES

[1] IBM. (n.d.). Internet of Things. Retrieved from https://www.ibm.com/topics/internet-of-things.

[2] Balbix. (n.d.). Addressing IoT Security Challenges. Retrieved from https://www.balbix.com/insights/addressing-iot-security-challenges/.

[3] Feinstein, L., Schnackenberg, D., Balupari, R., & Kindred, D. (2003). Statistical approaches to DDoS attack detection and response. *IEEE*. https://doi.org/10.1109/discex.2003.1194894.

[4] Damon, E., Dale, J., Laron, E., Mache, J., Land, N., & Weiss, R. (2012). Hands-on denial of service lab exercises using SlowLoris and RUDY. *Association for Computing Machinery*, 21–29. https://doi.org/10.1145/2390317.2390321.

[5] Shorey, T., Subbaiah, D., Goyal, A., Sakxena, A., & Mishra, A. K. (2018). Performance Comparison and Analysis of Slowloris, GoldenEye and Xerxes DDoS Attack Tools. *IEEE*. https://doi.org/10.1109/icacci.2018.8554590.

[6] CISOMAG. (n.d.). Top 5 Internet of Things (IoT) Hacking Tools Explained. Retrieved from https://cisomag.com/top-5-internet-of-things-iot-hacking-tools-explained/.

[7] Recorded Future. (n.d.). Nmap Commands: A Cheat Sheet for Cybersecurity Professionals. Retrieved from https://www.recordedfuture.com/threat-intelligence-101/tools-and-techniques/nmap-commands.

[8] 0xc0d. (n.d.). Slow Loris. GitHub. Retrieved from https://github.com/0xc0d/Slow-Loris.

[9] How2Electronics. (n.d.). Connecting ESP32 to Amazon AWS IoT Core using MQTT. Retrieved from https://how2electronics.com/connecting-esp32-to-amazon-aws-iot-core-using-mqtt/.

[10] Bromley Satellite. (n.d.). Installing Drivers for the ESP32. Retrieved from https://bromleysat.com/installing-drivers-for-the-esp32.

[11] Last Minute Engineers. (n.d.). MAX30102 Pulse Oximeter & Heart-Rate Sensor Arduino Tutorial. Retrieved from https://lastminuteengineers.com/max30102-pulse-oximeter-heart-rate-sensor-arduino-tutorial/.

[12] ElectronicWings. (n.d.). MAX30100 Pulse Oximeter Interfacing with Arduino. Retrieved from https://www.electronicwings.com/arduino/max30100-pulse-oximeter-interfacing-with-arduino.