University Journal of Research: Computer Science and Applications

Rogue AP: Assessing The Threat of Smart Watch

Mr. Preet Shah[a], Dr. Kashyap C. Patel[b], Dr. Sachinkumar Anandpal Goswami[c], Dr. Jagruti Patel[d] and Pooja Pancholi[e]

[a]DevOps Engineer, Comcast Business, Bensalem, Pennsylvania, United States

[b]Assistant Professor, FCA, Ganpat University, Ganpat Vidyanagar, Gujarat, India

[c]Assistant Professor, FCA, Ganpat University, Ganpat Vidyanagar, Gujarat, India

[d]Assistant Professor, FCA, Ganpat University, Ganpat Vidyanagar, Gujarat, India

[e]Assistant Professor, FCA, Ganpat University, Ganpat Vidyanagar, Gujarat, India

Corresponding Author Email: drkashyapcpatel@gmail.com

Abstract

Smartwatches integrate with centralized networks for real-time health tracking, location services, and software upgrades, making them vital in healthcare, business, and personal health monitoring. Rogue access point (AP) attacks can intercept, modify, and exploit data sent between devices and servers due to their dependency on wireless communication channels (Bluetooth, Wi-Fi, and cellular networks). Rogue APs enable Man-in-the-Middle (MitM) exploits, firmware manipulation, GPS spoofing, and DDoS assaults, threatening data integrity and device operation. Wi-Fi Pineapple, Aircrack-ng, and Kali Linux are used to spoof network settings, intercept data, and disseminate malware. This research work analyses rogue AP attacks on smartwatches, their real-time effects, and possible defenses. We propose a blockchain-based security framework using immutability, transparency, decentralization, and smart contracts to address these vulnerabilities. Blockchain prevents unauthorized access and long-term attacks via secure firmware updates, decentralized authentication, GPS data integrity, and automatic threat detection. We also explore AI-driven intrusion detection, multi-factor authentication, and adaptive security measures to improve smartwatch ecosystems. This work uses blockchain and advanced security protocols to mitigate rogue AP threats and secure wristwatch communication and data in an increasingly interconnected IoT world.

Keywords: Smart Watch Threat, Rogue Access Point, Firmware Compromise, MiTM, GPS Spoofing

1. Introduction

Health care, corporations, and personal health monitoring require wristwatch integration with the network of linked systems. They transmit data with servers via Bluetooth Low Energy, Wi-Fi, and cellular networks for real-time health monitoring, location tracking, and software updates. However, rogue access points may cause security difficulties. A rogue AP spoofs a legal network access point to intercept, manipulate, and exploit data between smartwatches [1]–[3] [6]–[8] [25] and their centralized servers [25]–[27]. Rogue AP attacks against smartwatch-centered systems can inject malicious software, intercept data, change device settings, compromise the device's secure connection, or both. Unlike conventional approaches that target the smartwatch directly, rogue AP exploits the dependency on wireless communication channels by pretending to be trusted to bypass protective device layers. This could result in unwanted access to sensitive data, gadget malfunctions, and latent long-term backdoors enabling device monitoring or control. Rogue APs may enable some attacks, which we continue to study. These include Man-in-the-Middle [1]–[7] (MitM) attacks, which intercept and manipulate smartwatch-server traffic; firmware modification attacks, which can cause unsafe behavior or disable security mechanisms; GPS spoofing [1]-[7] [9]-[12, when smartwatch location data is fabricated to misreport user locations. Rogue APs also cause DoS attacks [1]-[7] by flooding the central server with traffic, preventing legitimate devices from connecting. With technical knowledge, we explain how the rogue AP works and what hardware and software to utilize in such attacks. Wi-Fi Pineapple [1]-[7], Aircrack-ng [1]-[7], and Kali Linux [1]-[7] [21] are used because a rogue AP may spoof legitimate network settings and disconnect devices from legitimate access points while intercepting data. It describes how rogue APs can drain batteries, compromise software, and even disrupt smartwatch networks in real time. This introduction of rogue AP risks highlights the necessity for more central smartwatch safety protocols. Rogue AP assaults and smartwatch exploits: understanding their mechanics Developers, security professionals, and system administrators will have a solid foundation to implement hardened prevention features like encrypted communications, authenticated firmware updates, and continuous network monitoring [2]-[8] [25] [26]. This chapter should teach readers how to identify and address connected smartwatch and wearable security issues.

Rogue AP : Setup and Configuration

To set up a rogue AP for educational or ethical hacking purposes, we need a combination of hardware and software tools. Below is a list of recommended tools and devices that we used for this purpose.

Setup of Centralized Server and Smart Watch connection:

In the connected world of today, a central server is a fundamental component in every IoT system. This is the central gathering place for data to be collected, processed, and analyzed for real-time monitoring and control over the devices. So let's start discussing how to set up a centralized server, other alternatives available, and what will be the benefits of each.

A hardware-based central server is a traditional setup in which a physical server is mounted on-premises; it is an investment of a big upfront hardware, software, and maintenance.

1.1 Setup Process:

Hardware Selection: Choose a server hardware that meets your requirements, considering factors like processing power, memory, and storage.

Operating System Installation: Install a suitable operating system, such as Linux or Windows, on the server hardware.

Centralized Server Software Installation: Install centralized server software, like Apache or Nginx, to manage and process data from devices.

Database Management System Installation: Install a database management system, like MySQL or PostgreSQL, to store and manage data.

Configuration and Testing: Configure the centralized server, database management system, and devices, and test the setup to ensure everything is working as expected.

1.2 VM-Based Centralized Server

A VM-based centralized server is a virtualized setup where a virtual machine (VM) is created on a physical host machine. This approach offers greater flexibility and scalability than a traditional hardware-based setup.

Process of set up:

Host Machine Selection: Choose a host machine that meets your requirements, considering factors like processing power, memory, and storage.

Hypervisor Installation: Install a hypervisor, like VMware or VirtualBox, on the host machine.

VM Creation: Create a VM on the host machine, allocating resources like CPU, memory, and storage.

Operating System Installation: Install a suitable operating system on the VM.

Centralized Server Software Installation: Install centralized server software on the VM.

Database Management System Installation: Install a database management system on the VM.

Configuration and Testing: Configure the centralized server, database management system, and devices, and test the setup to ensure everything is working as expected.

1.3 Cloud-Based Centralized Server

A cloud-based centralized server is a setup where a centralized server is hosted on a cloud platform, like Amazon Web Services (AWS), Google Cloud Platform (GCP), or Microsoft Azure.

Benefits:

- **Scalability:** Cloud-based centralized servers can scale up or down to meet changing demands, making it an ideal solution for growing IoT systems.
- **Cost-Effective:** Cloud-based centralized servers are more cost-effective than hardware-based setups, as they eliminate the need for dedicated hardware.
- **Reliability:** Cloud platforms offer high uptime and reliability, ensuring that your centralized server is always available.

Process of Setup:

- **Cloud Platform Selection:** Choose a cloud platform that meets your requirements, considering factors like pricing, scalability, and reliability.
- **Instance Creation:** Create an instance on the cloud platform, allocating resources like CPU, memory, and storage.
- **Operating System Installation:** Install a suitable operating system on the instance.
- **Centralized Server Software Installation:** Install centralized server software on the instance.
- **Database Management System Installation: Install a database management system on the instance.**
- **Configuration and Testing:** Configure the centralized server, database management system, and devices, and test the setup to ensure everything is working as expected.

A centralized server is essential for constructing an IoT system. Hardware-based centralized servers are traditional, but VM- and cloud-based ones are more flexible, scalable, and cost-effective. Choose the proper approach to provide a stable and efficient centralized server that fulfils IoT system needs.

1.4 Smartwatch Connection to Centralized System

Smartwatches can connect to the centralized system using various communication protocols and technologies. The connection process between a smartwatch and the centralized system typically involves the following steps:

- **Device Discovery:** The smartwatch discovers the centralized system using a discovery protocol, such as Bluetooth or Wi-Fi.
- **Authentication:** The smartwatch authenticates with the centralized system using a secure authentication protocol, such as SSL/TLS.
- **Data Transmission:** The smartwatch transmits data to the centralized system, which can include sensor data, user input, or other information.
- **Data Processing:** The centralized system processes the data received from the smartwatch, which can include data analysis, storage, or forwarding to other systems.
- **Feedback:** The centralized system provides feedback to the smartwatch, which can include notifications, updates, or other information.

1.5 Rogue AP Setup to Connect to Centralized System

For a rogue AP to practically connect to a centralized server, a few key techniques can be used to exploit network vulnerabilities. Here are practical approaches that attackers might use to establish such unauthorized connections:

Step 1. Rogue AP with SSID Spoofing: The attacker configures a rogue AP using the same SSID and MAC address of the legitimate access point, masquerading it closely to simulate the true AP. Incidentally, the attacker positions the rogue AP in some location in the network with low signal strength from the real AP.

Step 2. Captive Portal for Credential Collection: TheMiTM AP will be able to offer a captive login page, typically known as the login page that users run into every time they try to join a new network. These machines will pass through this false login trying to reach the central server and as such lose all their credentials. This is widely happening in open or public Wi-Fi.

Step 3. MitM Attack Operation: The attackers will perform the operation by sniffing and relaying traffic between the ADAS system or device and legitimate server through ARP spoofing. The rogue AP acting as the server can capture information to relay back to the attacker for him to monitor or alter the content of data exchanges.

Step 4. Manipulating Weak Authentication or Encryption Protocols: As authentication protocols created quite some time ago, such as WEP or weakened WPA configurations, are easily compromised in a brute-force attack (as shown in Aircrack-ng or Hashcat), the rogue AP can break these authentications and, once it is able to successfully pretend to be a legitimate device, may proceed on to penetrate the centralized server without raising an alarm in real time.

Step 5. Packet Sniffing and Session Hijacking: The malicious AP intercepts the data packets using Wireshark or Ettercap. It waits for session tokens and then can hijack an active session of someone if secure authentication is not enforced, using mechanisms such as HTTPS or VPN.

Step 6. DNS Spoofing and Redirection: After intercepting the traffic stream, the rogue AP now redirects requests from the source to malicious or cloned sites using DNS spoofing. The system will believe it's talking to the legitimate server; however, this is not the case because it's actually going through the attacker's infrastructure.

Step 7. Reverse Proxy Configuration: Rogue AP acts as a reverse proxy that sends all the requests to the legitimate server, keeps logging all the data transmissions, and allows the attackers to analyze and modify requests and responses in real-time. This can be facilitated by using tools such as Bettercap and MITMf (Man-in-the-Middle Framework).

Step 8. SSH/FTP Exploits on Centralized Server Configuration: When the centralized server allows remote [25]-[27] access to its systems' maintenance, the rogue AP could attempt to exploit weak SSH or FTP credentials, especially if those services use default passwords or lack multi-factor authentication.

These approaches highlight the need for robust encryption, secure authentication, and network monitoring to detect and neutralize rogue APs before they gain access to critical systems.

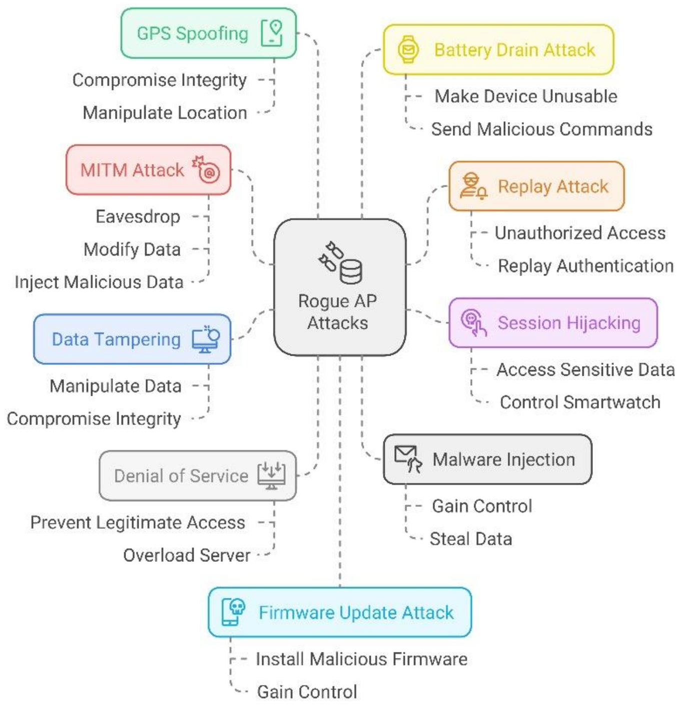

Fig. 1. Possible threats of Smart Watch using Rogue AP

Rogue AP attacks include GPS Spoofing, Battery Drain Attacks, and Replay Attacks to get unwanted access. MITM Attacks, eavesdropping, session hijacking, malware injection, data manipulation, DoS, and firmware update attacks were used to compromise linked devices.

2. Firmware Compromise of Smartwatches Integrated via Centralised Server through Rogue AP

Through a rogue access point in a centralized server system, firmware for connected smartwatches is stolen. Using malicious firmware updates, the attacker wanted to seize control of the smartwatches. Therefore, the attacker might change device behaviour, deactivate security features, or build backdoors for future assaults. The next example shows real-time steps:

2.1 Setup Scenario

Scenario: A firm has issued over 1000 smartwatches to staff linked with a central server. All about the smartwatch is controlled by the server, including firmware upgrade, battery management, GPS tracking, and network setup.

- **Hardware:** The smartwatches are ESP32-based and have Wi-Fi connection with the server. They strictly work on the centralized system to receive all OTA firmware upgrades and similar essential operations.
- **Rogue AP:** This hacker creates a rogue access point that masquerade as the legitimate access point used by the smartwatches. The rogue AP is connected to a server with malicious intentions. It intercepts and manipulates traffic between the smartwatches and the legitimate server.

2.2 Breakdown of the attack

Step1. Reconnaissance Phase

The hacker performs network reconnaissance to identify information about smartwatches, access points, and the centralized server system.

Objective: To know the information in respect of Wi-Fi SSID, MAC addresses of smartwatches and what is the firmware upgrade.

Step 2. Creating Rogue Access Point

For a rogue AP, a device like Raspberry Pi or ESP32 is used to build. The rogue AP is configured with the same SSID as the legitimate network and utilizes tools de-authenticate the legitimate AP.

Objectives: It implants the smartwatches to unsuspectingly connect to the rogue AP, which now is a man-in-the-middle (MITM) between the smartwatches and the centralized server.

Step 3. Man-in-the-Middle (MITM) Attack

The rogue AP captures all the communications that took place between the smartwatches and the centralized server. As smartwatches believe the server for firmware updates, it can now intercept, modify, or inject traffic with the help of rogue AP to exploit that belief.

Goal: Obtain control over the firmware update process by intercepting the communication between the smartwatch and the server.

Step 4. Compromising the Firmware Update Process

The attacker obtains access to the firmware update service of the centralized server. When the smartwatches place a request for firmware upgrade, the rogue AP passes on malicious firmware instead of forwarding the request to the true server.

Process:

The attacker has created malicious firmware containing:

Backdoors for the remote access of the smartwatch

Disabling safety features such as encryption or password protection

Spyware or keyloggers to track the users.

The hacker uses a tool such as esptool.py to transmit malicious firmware into the smartwatch after the smartwatch has connected the centralized server via rogue AP.

Step 5. Malicious firmware created by the hacker

Objective: It loads malicious firmware to all the connected smartwatches with their security open for any kind of attack in the future.

Step 6. Firmware Compromise Execution

Once the hack firmware is installed in smartwatches, the hacker totally takes over these devices. Now the hack smart watches do the following:

It sends data back to the hacker's server rather than the legitimate server.

The vulnerability to other attacks increases since security features are disabled.

The device runs in the background of the legitimate server or the administrators of those devices do not have a clue as these devices have been compromised.

Purpose: To take long term control of the smartwatches, that would be able to do activities such as:

GPS based location tracking

Listening of accessible communication

Capturing sensitive information

2.3 Time-Real Impact of the Attack

- **Massive Scale Takeover of Gadgets**

  All 1000+ smartwatches become vulnerable at one go since the forged AP can sneak in and grab all communication to the centralized server.

  The hacker can now take control, observe, and manipulate the devices, leading to a security violation across the firm.

- **Disabling Security Features**

  The malicious firmware may result in deactivating a number of important security features like SSL encryption, whose removal will expose the communication to another attack.

  Password protections and firewalls may be disabled, hence making penetration into the system easier for attackers later on.

- **Device Instability**

  The manipulated firmware makes the smartwatches behave erratically through:

  It is draining the battery through keeping the device in high-power mode

  Disabling network connectivity to deny communication with a legit server.

  Crashing or turns nonfunctional upon installation of the malicious firmware.

Due to over-the-air update vulnerabilities, rogue access point firmware compromise attacks on smartwatches are effective in real time. Since communication between the centralized server and smartwatches can be intercepted, malicious firmware implants and unauthorized access to several devices can occur. An assault like this could generate long-term security, data, and device performance issues. Secure boot, digitally signed software, WIDS [1]-[3] [6] monitoring, and multi-factor authentication can protect IoT networks from rogue AP attacks.

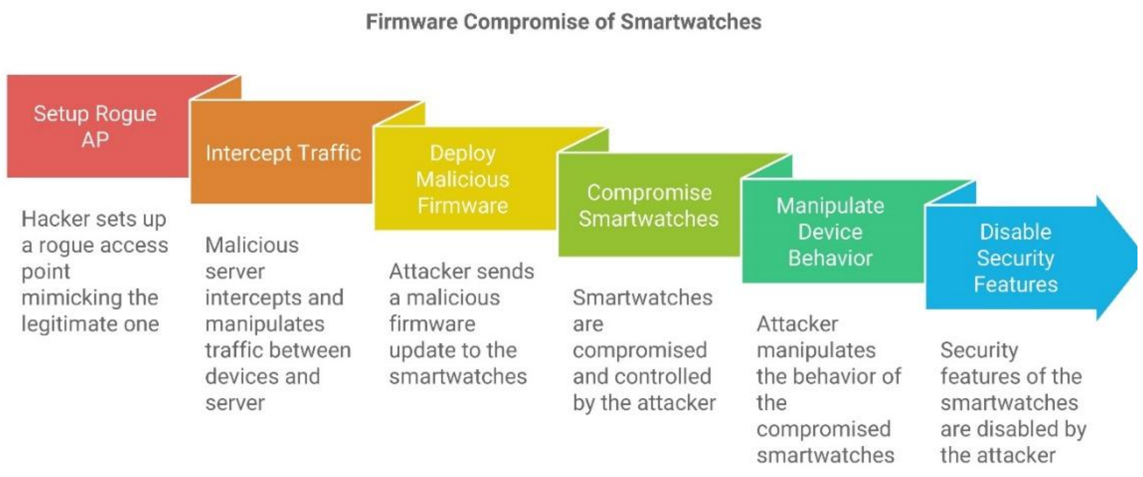
Firmware Compromise of Smartwatches
Setup Rogue AP
Hacker sets up a rogue access point mimicking the legitimate one
Intercept Traffic
Malicious server intercepts and manipulates traffic between devices and server
Deploy Malicious Firmware
Attacker sends a malicious firmware update to the smartwatches
Compromise Smartwatches
Smartwatches are compromised and controlled by the attacker
Manipulate Device Behavior
Attacker manipulates the behavior of the compromised smartwatches
Disable Security Features
Security features of the smartwatches are disabled by the attacker

Fig. 2. Firmware compromise through Rogue AP

Figure 2 shows a rogue access point compromising smartwatch firmware. After setting up the rogue AP, traffic is intercepted and modified. Malware then installs to control the smartwatch and disable all security. This guide shows how to hack smartwatch firmware.

3. Compromise of GPS of Smartwatches Connected via Centralized Server using Rogue AP

A rogue AP utilizes a centralized server [25]-[27] to infect connected smartwatches by extracting firmware and deploying malicious updates. This provides attackers with power over the devices, allowing them to manipulate behaviour, disable security features, or implant backdoors for subsequent attacks. The following is a concise overview of the various attack types and their consequences:

GPS Spoofing, Blocking, or Permanent Disconnect

The rogue AP now exploits the communication between the server and watches in a manner that it dictates the functionalities of the GPS for the smartwatches. The attacker may spoof GPS data, block GPS signals, or permanently disconnect GPS services across all devices.[1]-[6][9]-[12]

3.1 GPS Spoofing Attack

This sends fake GPS data to the smartwatches, thereby creating a situation as if it were elsewhere. It really creates disruption because the application is about employee tracking, logistics, or geofencing. [1]-[6][9]-[12]

Process:

- Acquiring real GPS data and using tools such as GPS-SDR-SIM [22] to generate a spoofed GPS signal.
- Broadcasting the spoofed GPS data to all connected smartwatches through the rogue AP.
- Updating the central server with incorrect GPS information.

Objectives: The fabricated GPS coordinates in reported location data by every smartwatch.

3.2 GPS Blocking Attack

Malicious block GPS, and as a result of GPS blocking, GPS data cannot be received by smartwatches, thus the devices losing their location services and stopping the major applications.

Breakdown:

Malformed GPS data or disable GPS by breaking the GPS functionality on the smartwatches.
The smartwatches cannot receive legitimate GPS signals from the satellite
The central server cannot trace the location of smartwatches.

Goal: Block all the GPS signals in the smartwatch. The location services are lost.

3.3 Permanent GPS Disconnect Attack

Permanent disability of the GPS functionality can be achieved through corruption of the firmware or by sending commands that cause all smartwatches to disconnect the GPS receiver. This would imply that the devices will never establish a GPS connection again.

Process:

Send a malicious firmware update via rogue AP that disables receivers or corrupts the GPS system.
Alternatively, send commands to permanently disable the GPS module at the hardware level on every smartwatch.
Smartwatches will not auto reconnect to GPS services even after reboot

Objective: Permanent GPS failure on all smartwatches and rendering them incapable of providing location services.

Consequent Attack Effects

Location Misinformation (GPS Spoofing):

The employees using the smartwatches will have incorrect location data, that might even lead to operational failure for logistics, emergency services, or field operations.

The alarms or tracking system with the help of geofencing will misreport the location of the employees or the devices, and some will even trigger a false security alarm or an operational error.

Temporary or Permanent Loss of GPS Services (Blocking or Permanent Disconnect):

Location-based applications, employee tracking, health monitoring, security systems will not work. The organization would lose real-time access to the smartwatches, which could pose a security or safety risk.

If the GPS is permanently disabled, then the watches will not be able to reconnect to location services and will need to be physically recalled or reprogrammed.

Centralized Control Hijacking

Because the rogue AP is interfering with communication between the central server and smartwatches, the hacker would have their way on altering the GPS settings in all the devices in one hit. The server administrators might never know that their GPS has been compromised until enough damage has been caused. Like loss suffered as a result of wrong tracking.

A malicious rogue AP can control smartwatch GPS services connected to a central server to spoof, block, or disable GPS on all devices. This disturbs geofencing, logistics, and tracking. Attackers can trigger catastrophic system failures by GPS-SDR-SIM spoofing, jammers, or firmware manipulation. Multi-factor authentication, secure OTA updates, rogue AP monitoring, and robust encryption are needed for protection. FIG 2 shows how a rogue AP intercepts communication, distributes malicious software, disables security, and gives attackers full device control.

4. Mitigation Strategies - Blockchain-Based Solutions

Traditional security measures play a crucial role in protecting smartwatches from cyber threats. End-to-end encryption should be implemented for all communications between smartwatches and servers to prevent data interception. Multi-factor authentication (MFA) enhances security by requiring multiple verification steps for firmware updates and user access, reducing the risk of unauthorized modifications. Additionally, regular updates and timely patches are essential to address known vulnerabilities, ensuring that devices remain protected against emerging threats.[28]-[30]

4.1 Key Blockchain Features for Mitigation

- **Immutability:** Blockchain technology offers several inherent security features that help prevent and mitigate cyberattacks. Immutability ensures that once data is recorded on the blockchain, it cannot be altered or tampered with without consensus from the network. This property is particularly useful in preventing firmware tampering, GPS spoofing, and unauthorized modifications to smartwatch configurations. For example, firmware updates can be logged on the blockchain, ensuring that only verified and untampered updates are applied to devices.[28]-[30]
- **Transparency:** Another key feature is transparency, which allows all transactions and data stored on the blockchain to be visible to authorized participants. This makes it easier to audit and verify activities, detect rogue APs, unauthorized data harvesting, and malicious commands sent to devices. A practical example of this is maintaining a transparent log of all GPS data changes, which can help identify anomalies caused by spoofing or blocking attacks. [28]-[30]
- **Decentralization:** Decentralization is another crucial aspect of blockchain security, as it eliminates single points of failure and reduces the risk of centralized server breaches. This helps protect against Distributed Denial-of-Service (DDoS) attacks and rogue AP exploitation. Instead of relying on a single centralized server for GPS data, a decentralized network of nodes can validate and distribute location information securely, making it much harder for attackers to manipulate data. [28]-[30]
- **Smart Contracts:** Finally, smart contracts provide an automated and secure mechanism for enforcing predefined security policies. These self-executing contracts ensure that actions, such as firmware updates or access control enforcement, occur only when specific conditions are met. For instance, a smart contract can verify the authenticity of a firmware update before allowing it to be installed on a smartwatch, preventing attackers from pushing malicious updates. By integrating these blockchain-based security mechanisms, smartwatches can achieve enhanced resilience against cyber threats and maintain data integrity, authentication, and availability in a highly connected ecosystem. [28]-[30]
- **Firmware Security:** Firmware protection keeps hackers from delivering malicious firmware updates to smartwatches. A solution based on blockchain supplements this by storing records of cryptographic hashes of approved firmware versions. The smartwatch verifies the firmware hash against the blockchain prior to installation, guaranteeing

authenticity. Only updates signed by approved users with digital signatures are accepted, minimizing the likelihood of unauthorized modification. [28]-[30]

4.2 GPS Spoofing Prevention:

GPS spoofing attacks manipulate the location information of a smartwatch, leading to geofencing and navigation failures. To counter this, GPS coordinates with timestamp can be safely stored on the blockchain. A consensus protocol can authenticate GPS information from satellites and IoT sensors with less dependence on a single source. Anomaly detection protocols can detect inconsistent reported and blockchain-validated locations, allowing real-time detection of spoofing attempts.

4.3 Rogue AP Detection:

Rogue APs pose a threat to smartwatch traffic by intercepting it. Blockchain authentication guarantees only authentic APs connect, reducing MITM attack threats. Logging AP connections on a blockchain enables transparent audits. Smart contracts can shut down unauthorized APs, preventing smartwatches from connecting to malicious networks.

4.4 DDoS Attack Mitigation:

DDoS attacks target centralized smartwatch servers by overwhelming them with traffic and taking services offline. A decentralized blockchain can distribute traffic across nodes, with no points of failure. Blockchain rate-limiting identifies and blocks excessive rates of requests from a single point. This approach provides scalability and resistance to massive DDoS attacks while remaining accessible to services.

Data Privacy and Integrity: Cybercriminals target smartwatches because they save health metrics, GPS locations, and biometric data. Encrypt this data before storing it on the blockchain. Zero-knowledge proofs (ZKPs) allow data verification without revealing its contents, improving privacy. Permissioned blockchains also allow user-controlled access, complying with GDPR and protecting data.

4.5 Real Time Monitoring and Auditing:

Without insight into device and network behaviour, cyber risks are hard to spot. Blockchain technology lets smartwatches record firmware updates, GPS changes, and authentication attempts in an immutable log. Blockchain explorers or security dashboards allow managers to monitor threats in real time. Periodic audits can also detect questionable activities, improving security and providing continual protection.

4.6 Example: Implementation of the Smartwatch Security Framework

A blockchain-based smartwatch security framework improves safety via secure firmware upgrades, GPS data verification, decentralized authentication, and automated threat response mechanisms.

In Step 1, manufacturers create a cryptographic hash for each firmware version and record it on the blockchain alongside information. Smartwatches authenticate the hash prior to installation, guaranteeing that only legitimate updates are implemented.

Step 2 ensures the security of GPS data by requiring smartwatches to periodically transmit coordinates to the blockchain, where consensus algorithms authenticate accuracy through cross-referencing with reliable sources, indicating inconsistencies for further examination.

Step 3 enhances authentication by allocating a distinct digital identity to each smartwatch, which is recorded on the blockchain. Devices authenticate via private keys, whereas unauthorized access points are identified and obstructed based on unsuccessful authentication attempts. Step 4 implements automated threat response via smart contracts that monitor device behaviour and initiate actions—such as unplugging compromised devices or notifying administrators—upon detection of abnormalities like illegal firmware updates or GPS manipulation. This system guarantees real-time security, data integrity, and resistance against cyber threats.

5. Future Enhancement

To detect rogue access points in wristwatch networks in real time, AI-driven detection systems may be added. Such systems would utilise AI to detect data traffic irregularities to foresee and neutralise threats. Blockchain technology might also enable decentralised, tamper-proof firmware updates for smartwatches, maintaining software integrity. Enhancing authentication protocols with multi-factor authentication and biometric verification could further secure communication between smartwatches and central servers. Security evaluations and adaptive modifications to defences against increasingly complex rogue AP techniques would require regular system audits. These innovations could make smartwatch ecosystems more secure and resilient to attackers.

6. Conclusion

The research into rogue access points indicates that they are significant threats to the smartwatch, especially within a centralized IoT ecosystem. In detailed terms, rogue APs can take advantage of vulnerabilities in smartwatches by intercepting communications for man-in-the-middle attacks or injecting malicious firmware for in-field overwriting of firmware, including manipulating GPS data. Such attacks may cause problems in smartwatch operations, compromise user privacy by draining batteries or even cause mass device failures. This reflects the importance of the implementation of strong security policies, such as encryption and secure firmware update procedures with associated real-time network monitoring mechanisms, to strengthen the smartwatch ecosystem. Blockchain enhances smartwatch security by preventing GPS spoofing, rogue APs, firmware tampering, and DDoS attacks through immutability, transparency, decentralization, and smart contracts. Addressing scalability, energy use, and compliance is key to its effective implementation in IoT security.

7. References

[1] Patel, K. C., & Patel, A. (2022, November). Rogue access point: The WLAN threat. In 2022 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS) (pp. 943-950). IEEE.

[2] Patel, K. C., & Patel, A. (2022, March). Taxonomy and future threat of rogue access point for wireless network. In 2022 9th International Conference on Computing for Sustainable Global Development (INDIACom) (pp. 679-688). IEEE.

[3] Patel, K. C., & Goswami, S. A. (2024). Rogue Access Points: A Critical Threat to Electric Vehicle Charging Station Security. COMPUTER, 24(7).

[4] Patel, Dr. K. C. (2024). Crippling Connectivity: Unveiling the Network Disruption Potential of Vehicular Rogue Access Points. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 10(3), 632–643. https://doi.org/10.32628/ijsrcseit

[5] Patel, Dr. K. C. (2022). Recognition of Rogue Access Points Using a Machine Learning Approach. International Journal of Creative Research Thoughts - IJCRT (IJCRT.ORG), 10(12), C663–C672. https://ijcrt.org/papers/IJCRT2212283.pdf

[6] Chaitanyakumar, P. K. An Experimental Study and Novel Approach for Detection and Suppression of Rogue Access Point in Wlan.

[7] Siboni, S., Shabtai, A., & Elovici, Y. (2018, April). Leaking data from enterprise networks using a compromised smartwatch device. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing* (pp. 741-750).

[8] Siboni, S., Shabtai, A., & Elovici, Y. (2018). An attack scenario and mitigation mechanism for enterprise BYOD environments. ACM SIGAPP Applied Computing Review, 18(2), 5-21.

[9] Liu, P., Yang, P., Song, W. Z., Yan, Y., & Li, X. Y. (2019, April). Real-time identification of rogue WiFi connections using environment-independent physical features. In IEEE INFOCOM 2019-IEEE Conference on Computer Communications (pp. 190-198). IEEE.

[10] Park, S., Shaik, A., Borgaonkar, R., & Seifert, J. P. (2016, October). White rabbit in mobile: Effect of unsecured clock source in smartphones. In Proceedings of the 6th Workshop on Security and Privacy in Smartphones and Mobile Devices (pp. 13-21).

[11] Denney, K. W. (2019). A Hardware-Assisted Insider Threat Detection and Prevention Framework.

[12] Toh, J., Hatib, M., Porzecanski, O., & Elovici, Y. (2017, April). Cyber security patrol: Detecting fake and vulnerable wifi-enabled printers. In Proceedings of the Symposium on Applied Computing (pp. 535-542).

[13] Liu, J., & Sun, W. (2016). Smart attacks against intelligent wearables in people-centric internet of things. IEEE Communications Magazine, 54(12), 44-49.

[14] Liu, W., & Papadimitratos, P. (2024). Position-based Rogue Access Point Detection. *arXiv preprint arXiv:2406.01927*.

[15] DA Arisandi, D., Nazrul Muhaimin Ahmad, N. M., Subarmaniam, A., & L Kannan, S. K. (2021). The rogue access point identification: a model and classification review. The Indonesian Journal of Electrical Engineering and Computer Science, 23(3), 1527-1537.

[16] Chakraborty, N., Chao, Y., Luo, C., Li, J., Ziying, P., Chen, J., & Pan, Y. (2020). On understanding the impact of RTT in the mobile network for detecting the rogue UAVs. IEEE Transactions on Cognitive Communications and Networking, 6(4), 1218-1229.

[17] Yadav, K. K., & Tapaswi, S. (2015). Vehicular Rogue Access Point Detection Using Speed of Vehicle. Wireless Personal Communications, 82, 849-860.

[18] Aircrack-ng. (n.d.). *Aircrack-ng: The next generation of WiFi security auditing tools*. Retrieved October 25, 2024, from https://www.aircrack-ng.org/

[19] Raspberry Pi. (n.d.). Raspberry Pi. https://www.raspberrypi.org/

[20] Espressif Systems. (n.d.). ESP32: Overview. Espressif. Retrieved October 25, 2024, from https://www.espressif.com/en/products/socs/esp32

[21] Kali Linux. (n.d.). Kali Linux. https://www.kali.org/

[22] osqzss. (n.d.). GPS-SDR-SIM. GitHub. Retrieved October 25, 2024, from https://github.com/osqzss/gps-sdr-sim

[23] Espressif Systems. (n.d.). ESPtool documentation. Espressif Systems. Retrieved October 25, 2024, from https://docs.espressif.com/projects/esptool

[24] Aircrack-ng Organization. (2023, November 21). Aireplay-ng - Aircrack-ng [Aircrack-ng]. Retrieved from https://www.aircrack-ng.org/doku.php?id=aireplay-ng

[25] Kheirkhahan, M., Nair, S., Davoudi, A., Rashidi, P., Wanigatunga, A. A., Corbett, D. B., Mendoza, T., Manini, T. M., & Ranka, S. (2019). A smartwatch-based framework for real-time and online assessment and mobility monitoring. Journal of Biomedical Informatics, 89, 29–40. https://doi.org/10.1016/j.jbi.2018.11.003

[26] Mender. (n.d.). Internet gateway device. Mender Blog. https://mender.io/blog/internet-gateway-device

[27] ActivityWatch. (n.d.). Remote server. ActivityWatch Documentation. https://docs.activitywatch.net/en/latest/remote-server.html

[28] Zheng, Z., Xie, S., Dai, H., Chen, X., & Wang, H. (2017). An overview of blockchain technology: Architecture, consensus, and future trends. *IEEE International Congress on Big Data* , 557–564. https://doi.org/10.1109/BigDataCongress.2017.85

[29] Goswami, S. A., & Patel, K. D. (2023, July). Network Security for IoT Device using Blockchain Technology. In *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)* (pp. 1-8). IEEE.

[30] Alphand, O., Amoretti, M., Claeys, T., Dall'Asta, S., Duda, A., Ferrari, G., ... & Zanichelli, F. (2018, April). IoTChain: A blockchain security architecture for the Internet of Things. In *2018 IEEE wireless communications and networking conference (WCNC)* (pp. 1-6). IEEE.