Mini Project Report

on

Efficient PUF-Based Authentication and Key Exchange for IoT WiFi Networks

Submitted by

Raghava Gatadi, Rahul Verma, Lohith Varma, Suryansh Raj 21BCS088, 21BCS089, 21BCS095, 21BCS124

Under the guidance of

Dr. Suvadip Hazra

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY DHARWAD

18/11/2024

Certificate

This is to certify that the project, entitled **Efficient PUF-Based Authentication and Key Exchange for IoT WiFi Networks**, is a bonafide record of the Mini Project coursework presented by the students whose names are given below during <2024-2025> in partial fulfilment of the requirements of the degree of Bachelor of Technology in Computer Science and Engineering.

Roll No	Names of Students
21BCS088	Raghava Gatadi
21BCS089	Rahul Verma
21BCS095	Lohith Varma
21BCS124	Suryansh Raj

Dr. Suvadip Hazra
(Project Supervisor)

Contents

List of Figures

List of Tables

1 Introduction

The Internet of Things (IoT) is fundamental to smart applications like smart cities, building automation, automotive engineering, and industrial IoT (IIoT). While IoT drives connectivity and automation, it also presents challenges such as interoperability, security, and privacy concerns, even for those unfamiliar with the technology. With over 12 billion devices currently connected, IoT networks are projected to grow significantly, yet these devices often operate under resource constraints and lack built-in security features, making them vulnerable to issues like botnet attacks [9].

Traditional cryptographic methods, although effective in secure environments, are often unsuitable due to the limited resources and computational constraints of IoT devices. Several encryption protocols demand significant memory, which can be impractical for low-end IoT devices with limited resources, especially those deployed in environments with little supervision. This makes them susceptible to physical attacks, including tampering and semi-invasive threats. Physically Unclonable Functions (PUFs) present a viable solution by generating unique hardware-based digital signatures, harnessing natural variations from the manufacturing process to generate identities that are both tamper-resistant and impossible to clone [10], [8]. PUF-based solutions provide low-cost, hardware-level security that complements IoT devices by minimizing computation and storage overhead.

As Wi-Fi remains the primary connectivity method for IoT, it exposes devices to multiple threats, including MAC spoofing, de-authentication, and rogue access points. Current Wi-Fi protocols, from WEP to WPA2, have known vulnerabilities, especially during connection establishment, where attacks like the evil twin are common [5], [4], [6]. These vulnerabilities stem partly from the inadequacy of Wi-Fi protocols to secure the physical layer and their reliance on shared passwords. In this context, PUF-based authentication methods can address Wi-Fi's limitations by providing secure, low-resource authentication for IoT devices with minimal human intervention, even in insecure or extreme environments.

This paper presents a PUF-based protocol designed to enhance Wi-Fi authentication and key exchange security. When integrated with current Wi-Fi standards, this protocol offers strong protection against various security threats. It operates during the Wi-Fi connection setup phase, after which regular Wi-Fi protocols take over for ongoing communication. By utilizing the challenge-response mechanism inherent in PUFs, the protocol provides effective defense against tampering and other invasive attacks, making it suitable for use in environments with limited security. Previous attempts to implement PUF-based authentication for IoT devices often overlooked the unique challenges associated with Wi-Fi authentication [10], [2], [1].

Key features of the proposed protocol are as follows:

1. A lightweight, PUF-based protocol is developed specifically for Wi-Fi authentication and key exchange in IoT devices.
2. Security is enhanced through the use of three challenge-response pairs.
3. The protocol requires minimal computation and storage resources.
4. When combined with existing Wi-Fi protocols, it effectively protects IoT devices from tampering, invasive, and semi-invasive attacks.

2 Related Work

The concept of Physically Unclonable Functions (PUF) was first introduced by Pappu et al. in 2002 and has since been widely adopted in securing IoT applications such as authentication, key exchange, and IP protection [7]. PUFs are particularly effective in countering physical and invasive attacks, making them ideal for use in resource-constrained IoT devices where traditional cryptographic methods would demand too much computation and storage. Chatterjee et al. proposed a PUF-based authentication and key exchange protocol designed for IoT devices, which uses a secure server for client communication and incorporates identity-based encryption and elliptic curve pairing. However, this approach requires significant storage for the challenge-response pairs (CRPs), posing scalability challenges for large IoT networks [3].

Several researchers have focused on creating lightweight PUF-based solutions specifically for IoT devices connected to Wi-Fi networks. For example, Mahalat et al. introduced a protocol utilizing three sets of PUF-CRPs along with an XOR operation to reduce computational load during authentication [3]. While effective, this protocol faces scalability issues in large Wi-Fi networks, as routers must store multiple CRPs for each device, which limits the number of devices that can be supported [10]. Additionally, Aman et al. designed a PUF-based IoT authentication protocol using a Message Authentication Code (MAC) to secure device-server communications. Although the use of MAC improves security, the authors did not specify which MAC is used, and the protocol's reliance on multiple hash functions increases its computational complexity [1].

These existing works underscore the potential of PUF-based protocols to provide lightweight security solutions in IoT. However, most do not explicitly address the unique challenges of Wi-Fi security for IoT, motivating the need for a PUF-based Wi-Fi protocol that combines efficiency and scalability for a rapidly expanding IoT ecosystem.

3 Background Materials

3.1 Physically Unclonable Function (PUF)

3.1.1 Introduction to Physically Unclonable Functions (PUFs)

Physically Unclonable Functions (PUFs) are security primitives that leverage the inherent microscopic physical variations in a device's structure, such as those arising during the manufacturing process, to generate a unique identity for each device. These variations are impossible to replicate, making PUFs inherently secure and ideal for cryptographic applications, particularly in resource-constrained environments like the Internet of Things (IoT) [8]. A PUF generates a unique response to a challenge input, ensuring that each device is distinguishable from others, even those that are physically identical but produced in different fabrication runs.

3.1.2 Principle of Operation of a PUF

A PUF operates by mapping a challenge input, C_i, to a corresponding response output, R_i, as represented by the function $P(C_i) \to R_i$. The mapping is determined by the unique physical characteristics of the PUF instance, making each PUF response difficult to predict. The challenge-response pair (CRP) produced by a PUF is uniquely determined by its underlying physical properties, ensuring that even slight variations in the fabrication process result in unique, unclonable identifiers.

The physical variation in PUFs can be attributed to numerous factors, including inconsistencies in semiconductor material properties, device geometry, and the fabrication process.

These variations make the PUF behavior unpredictable, ensuring that identical devices, when subjected to the same challenge, will generate different responses.

3.1.3 Key Properties of PUFs

PUFs have several key characteristics that make them ideal for secure applications, especially in IoT devices:

- **Unclonability**: The variations in the physical structure of a PUF are inherently unpredictable, meaning that no two PUFs can be replicated, even by the manufacturer.
- **Tamper Resistance**: PUFs are resistant to tampering; if a device is physically altered, the response pattern of the PUF changes, rendering it unusable or unreliable.
- **Volatility**: The response generated by a PUF is volatile, meaning it is not stored in the device but is computed dynamically based on the challenge. This makes PUFs ideal for use in scenarios where security keys should not be stored permanently.
- **Uniqueness**: Each PUF instance has a distinct response pattern, ensuring that no two devices will produce the same set of challenge-response pairs.

3.1.4 Types of PUFs

PUFs are classified into two main categories based on the number of challenge-response pairs (CRPs) they support:

Weak PUFs

Weak PUFs support a limited number of challenge-response pairs, making them suitable for simpler applications with a small number of possible responses. These are typically used in low-security applications where only a few responses are needed for authentication or identification. While weak PUFs offer a level of security due to their unclonable nature, their security is not as robust as strong PUFs due to the limited number of possible responses.

- **Examples of weak PUFs**: Ring Oscillator PUFs, Arbiter PUFs.

- **Limitations**: Due to the limited set of responses, weak PUFs are more susceptible to attacks like machine learning-based prediction, making them less secure for high-risk applications.

Strong PUFs

Strong PUFs, on the other hand, support a much larger number of challenge-response pairs, offering a vast number of unique responses. This increased unpredictability provides a higher level of security and resistance to attacks. Strong PUFs are considered more secure for use in high-security applications, such as secure key generation, authentication, and device identification.

- **Examples of strong PUFs**: Volatile PUFs, Coating PUFs, and Optical PUFs.
- **Advantages**: Strong PUFs provide an exponentially greater number of unique responses, making them far more difficult to predict or replicate. This makes them suitable for complex security applications where robust cryptographic protection is required.

3.1.5 Application of PUFs in IoT Security

The unique properties of PUFs make them ideal for securing IoT devices, which often face stringent constraints in terms of computation, storage, and power. Some of the key IoT security challenges that PUFs address include:

- **Device Authentication**: By utilizing PUFs, each IoT device can have a unique identifier that is unclonable and resistant to tampering. This allows for secure authentication of devices in a network without the need for storing sensitive keys.
- **Key Generation and Exchange**: PUFs can generate cryptographic keys dynamically based on challenges, removing the need to store sensitive key material on the device. This makes key exchange protocols more secure, as the keys are not exposed to potential attackers.
- **Resistance to Physical Attacks**: Since the responses of PUFs are based on the physical characteristics of the device, tampering with the device will alter the PUF's response, providing a layer of security against physical attacks.

3.1.6 Challenges and Limitations of PUFs

While PUFs offer significant security advantages, their use in practical applications, especially in IoT, comes with a set of challenges:

- **Environmental Variations**: PUFs can be affected by environmental factors such as temperature and voltage, which may cause variations in their behavior. This issue can be mitigated through careful design, but it remains a challenge for reliable and stable operation over time.
- **Aging Effects**: Over time, the physical characteristics of a PUF

3.2 WiFi Security Threats

WiFi networks, particularly in Internet of Things (IoT) setups, are exposed to various security vulnerabilities that could compromise data confidentiality, network integrity, and overall availability. These vulnerabilities can lead to unauthorized access, data manipulation, or interception of sensitive communications. Below is a breakdown of some of the key threats to WiFi networks.

3.2.1 MAC Address Spoofing

One of the most common threats to WiFi networks is MAC address spoofing. This occurs when an attacker impersonates a legitimate device by mimicking its MAC address, bypassing security filters that rely on MAC address-based whitelisting.

- **Process of MAC Spoofing**: Attackers can capture the MAC address of a legitimate device via network sniffing tools. Once the MAC address is captured, the attacker changes their own device's MAC to match the stolen one, thus gaining unauthorized access to the network.
- **Consequences of Spoofing**: This attack allows the attacker to seamlessly infiltrate the network as though they were a trusted device, posing a significant security threat, especially in IoT environments where device monitoring is minimal.

- **Impact on IoT Networks**: Many IoT devices, due to their limited processing capabilities, often lack sophisticated security mechanisms to detect or block such spoofing attempts, making them highly susceptible to this kind of attack.

3.2.2 Rogue Access Point Attack

A rogue access point attack is a situation where an attacker sets up a malicious AP that mimics a legitimate one, tricking devices into connecting to it. This can allow the attacker to intercept and manipulate the communication between devices.

- **How the Attack Works**: The attacker configures the rogue AP with the same SSID as the legitimate AP. Once clients are disconnected by a de-authentication attack, they may reconnect to the rogue AP without realizing it's a fake. This gives the attacker an opportunity to eavesdrop on or alter the traffic.
- **De-authentication Phase**: Initially, the attacker sends a de-authentication packet to the legitimate AP, causing devices to disconnect. This packet is spoofed with the device's MAC address, making the de-authentication appear legitimate.
- **Dangers of Rogue APs**: When users unknowingly connect to rogue APs, attackers can steal sensitive data, inject malicious payloads into the network, or even launch further attacks. These types of attacks are particularly hazardous in open environments where users are less cautious.

3.2.3 Physical Tampering of IoT Devices

Physical tampering is another significant risk, where attackers gain access to IoT devices to extract critical data, such as stored WiFi credentials or cryptographic keys.

- **Exploiting Device Vulnerabilities**: Many IoT devices store WiFi passwords or cryptographic keys in non-volatile memory, which can be extracted through physical manipulation. Attackers might open up the device to access this sensitive information directly.
- **Risk of Data Exposure**: Once attackers retrieve such credentials, they can either impersonate the device or decrypt communication channels to gain unauthorized access to the network.

- **Lack of Protection in IoT Devices**: Due to the emphasis on low cost and power efficiency, many IoT devices lack adequate tamper-resistant features, making them susceptible to physical attacks.

3.2.4 Replay Attacks

Replay attacks occur when a malicious actor intercepts valid messages or communication exchanges between devices and replays them to deceive the system into executing unauthorized actions.

- **Mechanism of Replay Attacks**: The attacker captures valid network traffic (e.g., authentication packets) and retransmits it to impersonate a legitimate device. The receiving device may mistakenly accept the replayed message as valid.
- **Impact on Security**: This type of attack is particularly damaging in the context of IoT, where weak or no authentication mechanisms leave the network open to replaying intercepted data. It can allow attackers to bypass security measures or disrupt communication.
- **Preventive Measures**: To protect against replay attacks, techniques such as nonces (one-time random numbers) or timestamps are essential to ensure that past communications cannot be reused or replayed.

3.2.5 Denial of Service (DoS) and De-authentication Attacks

Denial of Service (DoS) and de-authentication attacks can disrupt the availability of WiFi networks, preventing legitimate users from accessing the network.

- **De-authentication Attack**: In this attack, the attacker sends de-authentication packets to devices connected to the legitimate access point, causing them to disconnect. This can result in users being repeatedly disconnected, effectively denying them access to the network.
- **Effectiveness of DoS Attacks**: DoS attacks are particularly harmful in IoT networks where continuous device connectivity is crucial. These attacks can render IoT devices unresponsive and disrupt their intended operations.

- **Mitigation Strategies**: To mitigate the impact of DoS attacks, methods such as stronger encryption, secure authentication, and advanced traffic analysis can help detect and prevent these disruptions.

3.2.6 General Vulnerabilities in IoT Devices

Beyond specific attacks, IoT devices themselves often exhibit vulnerabilities that make them prime targets for cybercriminals. These vulnerabilities include the use of weak default passwords, inadequate software updates, and unprotected communication channels.

- **Weak or Default Passwords**: Many IoT devices come with pre-configured passwords that are easy to guess or widely known. Attackers can easily gain access by exploiting these weak credentials.

- **Firmware Vulnerabilities**: Some IoT devices lack the ability to receive firmware updates or patches, leaving them open to known security flaws.

- **Insecure Communication**: Devices that use unencrypted communication protocols (like HTTP instead of HTTPS) are vulnerable to eavesdropping and man-in-the-middle attacks, allowing attackers to capture sensitive data.

4 FRAMEWORK OVERVIEW

In this work, we propose a lightweight protocol based on Physically Unclonable Functions (PUF) for securely connecting IoT devices to WiFi routers. The protocol is structured into three key phases: the one-time enrollment phase, the initialization phase, and the authentication phase. Each of these phases is described in detail below.

4.1 One-time Enrollment Phase

Before connecting to an access point (AP) or server, the IoT device $\wedge$ must exchange a set of challenge-response pairs with the corresponding router or server π. This process is performed only once for each router or server, meaning the IoT device needs to undergo this phase just a single time. It is assumed that the PUF circuit (P) is already embedded in the IoT device.

Symbol	**Description**
$\wedge$	IoT device
π	Router/Server
$M_\wedge$	MAC address of $\wedge$
PUF	Instance which produces the response to given challenge
C_i	i^th Challenge
R_i	i^th Response
nonce	Random number
——	Concatenation
H	Hash function

Table 1
Symbols and corresponding descriptions

For added security, the length of the challenge bit stream must be equal to the length of the response bit stream. The steps of this phase are outlined as follows.

1. Generate three random numbers C_1, C_2, C_3 in the range 0 to $2^n - 1$, where n is the bit-stream length of the PUF challenge.

2. Acquire the corresponding responses from the PUF: $R_1 = P(C_1)$, $R_2 = P(C_2)$, and $R_3 = P(C_3)$.

3. Store the challenge-response pairs (C_1, R_1), (C_2, R_2), and (C_3, R_3) in a secure database associated with the router or server π, indexed by the MAC address $M_\wedge$ of the IoT device $\wedge$.

It is assumed that this process occurs in a secure environment, monitored by a trusted party. For the proposed protocol, only three challenge-response pairs are necessary to ensure the required level of security.

4.2 Initialization Phase

After completing the enrollment of π, the IoT device $\wedge$ can initiate a secure connection to π. The process is as follows:

1. $\wedge$ sends a connection request along with its MAC address, $M_\wedge$, to π.

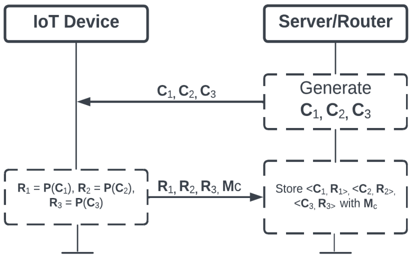
IoT Device
Server/Router
Generate
C_1, C_2, C_3
C_1, C_2, C_3
$R_1 = P(C_1), R_2 = P(C_2), R_3 = P(C_3)$
R_1, R_2, R_3, Mc
Store <C_1, R_1>, <C_2, R_2>, <C_3, R_3> with M_c

Figure 1. Device Enrollment and PUF Challenge Exchange

2. π checks its database to find a matching entry for $M_\wedge$. If a match is found, π continues with the next phase; otherwise, it ignores the request and terminates the connection attempt.

4.3 Authentication Phase

In this phase, the IoT device $\wedge$ authenticates the router or server π by performing the following steps:

1. Generate a random number *nonce*, where the length of *nonce*'s bit stream matches the response bit-stream length of the PUF P.

2. Calculate the new challenge and response pairs as follows:

$$C'_1 = nonce \oplus C_1, \quad C'_2 = nonce \oplus C_2, \quad R'_1 = nonce \oplus R_1, \quad R'_3 = nonce \oplus R_3$$

3. Compute the hash value $H_\wedge$ by concatenating C_1, C_2, R_1, R_3, and *nonce*, then calculate its hash:

$$H_\wedge \leftarrow Hash(C_1||C_2||R_1||R_3||nonce)$$

4. π sends $message_2$, which includes C'_1, C'_2, C_3, R'_1, R'_3, and $H_\wedge$ to $\wedge$.

Upon receiving this message, the IoT device $\wedge$ proceeds with the following calculations.

1. $R_3 = PUF(C_3)$

2. nonce $= R_3 \oplus R'_3$

3. $C_1 = nonce \oplus C'_1$

4. $C_2 = nonce \oplus C'_2$

5. $R_1 = nonce \oplus R'_1$

Upon receiving the message, the IoT device $\wedge$ calculates the hash value of the concatenation of $(C_1||C_2||R_1||R_3||nonce)$ and checks if it matches $H_\wedge$. If the values are the same, π is confirmed to be authentic, and the process continues. If the hash values do not match, it indicates either that π is not authentic or the values were altered during transmission (possibly by an attacker).

Next, the IoT device $\wedge$ generates a new nonce $nonce'$ as follows:

$$nonce' = R_2 \oplus nonce$$

This new nonce $nonce'$ will be used for the next phase of authentication, where $\wedge$ will authenticate itself with the router or server π. The IoT device $\wedge$ then proceeds to calculate the appropriate values for this phase.

1. $C_1^{new} = C_1 \oplus nonce'$

2. $C_2^{new} = C_2 \oplus nonce'$

3. $R_1^{new} = P(C_1^{new})$, $R_2^{new} = P(C_2^{new})$

4. $R'_1 = R_1^{new} \oplus nonce'$

5. $R'_2 = R_2^{new} \oplus nonce'$

6. $R'_3 = R_3^{new} \oplus nonce'$ where $R_3^{new} = P(C_1^{new} \oplus C_2^{new}))$

7. $H_\pi = \mathrm{H}(R_1^{new}$ —— R_2^{new} —— R_3^{new} —— $nonce')$

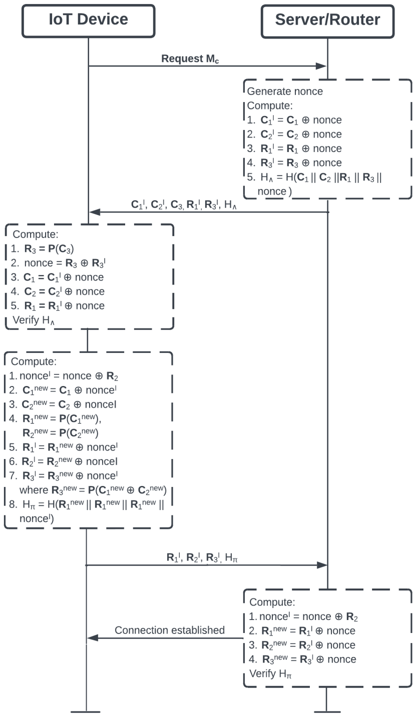
IoT Device
Server/Router
Request M_c
Generate nonce
Compute:
1. $C_1^l = C_1 \oplus nonce$
2. $C_2^l = C_2 \oplus nonce$
3. $R_1^l = R_1 \oplus nonce$
4. $R_3^l = R_3 \oplus nonce$
5. $H_\wedge = H(C_1 \| C_2 \| R_1 \| R_3 \| nonce)$
$C_1^l, C_2^l, C_3, R_1^l, R_3^l, H_\wedge$
Compute:
1. $R_3 = P(C_3)$
2. $nonce = R_3 \oplus R_3^l$
3. $C_1 = C_1^l \oplus nonce$
4. $C_2 = C_2^l \oplus nonce$
5. $R_1 = R_1^l \oplus nonce$
Verify $H_\wedge$
Compute:
1. $nonce^l = nonce \oplus R_2$
2. $C_1^{new} = C_1 \oplus nonce^l$
3. $C_2^{new} = C_2 \oplus noncel$
4. $R_1^{new} = P(C_1^{new})$, $R_2^{new} = P(C_2^{new})$
5. $R_1^l = R_1^{new} \oplus nonce^l$
6. $R_2^l = R_2^{new} \oplus noncel$
7. $R_3^l = R_3^{new} \oplus nonce^l$ where $R_3^{new} = P(C_1^{new} \oplus C_2^{new})$
8. $H_\pi = H(R_1^{new} \| R_1^{new} \| R_1^{new} \| nonce^l)$
$R_1^l, R_2^l, R_3^l, H_\pi$
Compute:
1. $nonce^l = nonce \oplus R_2$
2. $R_1^{new} = R_1^l \oplus nonce$
3. $R_2^{new} = R_2^l \oplus nonce$
4. $R_3^{new} = R_3^l \oplus nonce$
Verify H_π
Connection established

Figure 2. Proposed Authentication Protocol

The Iot device $\wedge$ sends the $message_3$ which contains R_1', R_2', R_3' and H_π to π. π upon recieving the message calculates the following values

1. $nonce' = R_2 \oplus nonce$
2. $R_1^{new} = R_1' \oplus nonce'$
3. $R_2^{new} = R_2' \oplus nonce'$
4. $R_3^{new} = R_3' \oplus nonce'$
5. $H'' = \mathrm{H}(R_1^{new} \text{ —— } R_2^{new} \text{ —— } R_3^{new} \text{ —— } none')$

Now verify whether H_π matches with H'', if both values matches then the IoT device $\wedge$ is authentic because only $\wedge$ can generate these values. If $H'' \neq H_\pi$ then the authentication failed and π terminates the communication.

5 SECURITY PROOF

5.1 Cryptanalysis

The XOR operation is frequently used in cryptographic protocols due to its simplicity and efficiency. However, to ensure security, it is important to meet several key criteria:

1. The two inputs being XORed must have the same length. 2. The key used should be a randomly generated number. 3. The key should never be reused.

To adhere to these conditions, our protocol ensures that the lengths of the challenge, response, and nonce bit-streams are equal. If the bit-stream length is denoted by n, the nonce has 2^n possible values. For large values of n (e.g., 64 or 128 bits), predicting the nonce becomes extremely difficult. Each new connection uses fresh challenge-response pairs and a new nonce, making the probability of repeating a nonce-challenge pair combination $1/(2^{2n})$, which is very low. Consequently, brute-force attacks are not feasible.

Moreover, when two large numbers are XORed together, it becomes practically impossible to deduce the original values from the result, adding another layer of security. Let's consider the values that π sends to the IoT device $\wedge$ during the authentication phase: C_1', C_2', C_3, R_1', and R_3'. The following XOR combinations are possible:

1. $C_1' \oplus C_2' = C_1 \oplus C_2$
2. $C_1' \oplus C_3 = C_1 \oplus nonce \oplus C_3$
3. $C_1' \oplus R_1' = C_1 \oplus R_1$
4. $C_1' \oplus R_3' = C_1 \oplus R_3$
5. $C_2' \oplus C_3 = C_2 \oplus nonce \oplus C_3$
6. $C_2' \oplus R_1' = C_2 \oplus R_1$
7. $C_2' \oplus R_3' = C_2 \oplus R_3$
8. $C_3 \oplus R_1' = C_3 \oplus R_1 \oplus nonce$
9. $C_3 \oplus R_3' = C_3 \oplus R_3 \oplus nonce$
10. $R_1' \oplus R_3' = R_1 \oplus R_3$

These XOR combinations do not reveal any information about the challenge-response pairs or the nonce. Further XOR combinations, such as:

1. $(1) \oplus (8) = C_1 \oplus C_2 \oplus C_3 \oplus R_1 \oplus nonce$

2. (1) $\oplus$ (9) $= C_1 \oplus C_2 \oplus C_3 \oplus R_3 \oplus nonce$
3. (1) $\oplus$ (10) $= C_1 \oplus C_2 \oplus R_1 \oplus R_3$

do not leak any information about $(C_1 \oplus C_3)$, $(C_2 \oplus C_3)$, $(C_3 \oplus nonce)$, or R_2. Thus, an attacker cannot deduce the values of the new challenges.

During the authentication phase, the IoT device $\wedge$ sends the following values:

1. $R'_1 = P(C_1^{new}) \oplus nonce'$
2. $R'_2 = P(C_2^{new}) \oplus nonce'$
3. $R'_3 = P(C_1^{new} \oplus C_2^{new}) \oplus nonce'$
4. $H_\pi = H(R_1^{new}||R_2^{new}||R_3^{new}||nonce')$

Since hashing is a one-way function, H_π cannot reveal any information about the original message. XORing other values only produces the XOR result of the PUF output, which does not provide any useful information to an attacker. Therefore, even if packets are sniffed during transmission, the adversary will not be able to extract valuable data.

5.2 WiFi Security

The proposed protocol enhances WiFi security by leveraging PUFs and a robust challenge-response mechanism. Here's how it addresses key security concerns:

1. **MAC Spoofing Protection**: The protocol prevents MAC spoofing by relying on unique challenge-response pairs from the PUF, which can't be duplicated. This means an attacker can't impersonate the IoT device even if they spoof the MAC address.

2. **Changing WiFi Passwords**: For each new connection, a different nonce is generated, leading to a new WiFi password every time. This makes predicting or reusing passwords impossible, reducing the risk of attacks like de-authentication.

3. **Protection from Replay Attacks**: Even if an attacker captures the transmitted data, they can't use it to replay a connection attempt. Each session uses a fresh challenge and nonce, making previous data useless for future authentication.

4. **Brute Force Deterrence**: The XOR operations combined with unpredictable nonces make brute-force attacks highly impractical. The size of the bit streams adds to the difficulty of guessing the correct values.

6 PERFORMANCE EVALUATION

The proposed protocol is designed to be both efficient and secure, with a clear focus on minimizing computational and storage demands. It involves the use of the PUF function eight times and the hash function four times, making the computational complexity 8P + 4H. This lightweight approach ensures that the protocol can operate swiftly, even in resource-constrained IoT environments. Storage requirements are also optimized, with the protocol needing only 6n bits to store data. Additionally, the size of any transmitted message does not exceed 6n bits, reducing communication overhead. These efficiency improvements make the protocol scalable, enabling secure, rapid authentication without burdening IoT devices with heavy computation or storage needs.

7 CONCLUSION

In conclusion, the proposed PUF-based lightweight protocol presents a secure and efficient solution for WiFi authentication and key exchange in IoT environments. By utilizing the unique characteristics of PUFs, the protocol effectively tackles critical security issues such as MAC spoofing, de-authentication attacks, and tampering threats without depending on traditional cryptographic techniques. Its design minimizes computational and storage overhead, making it ideal for resource-constrained IoT devices. This protocol not only strengthens the security of IoT networks but also ensures scalability and performance, positioning it as a promising candidate for future wireless communication standards.

References

[1] Muhammad N Aman, Kee Chaing Chua, and Biplab Sikdar. Position paper: Physical unclonable functions for iot security. In *Proceedings of the 2nd ACM international workshop*

on IoT privacy, trust, and security, pages 10–13, 2016.

[2] Urbi Chatterjee, Rajat Subhra Chakraborty, and Debdeep Mukhopadhyay. A puf-based secure communication protocol for iot. *ACM Transactions on Embedded Computing Systems (TECS)*, 16(3):1–25, 2017.

[3] Urbi Chatterjee, Vidya Govindan, Rajat Sadhukhan, Debdeep Mukhopadhyay, Rajat Subhra Chakraborty, Debashis Mahata, and Mukesh M Prabhu. Building puf based authentication and key exchange protocol for iot without explicit crps in verifier database. *IEEE transactions on dependable and secure computing*, 16(3):424–437, 2018.

[4] Heqing Huang, Yanjun Hu, Yan Ja, and Shiliang Ao. A whole-process wifi security perception software system. In *2017 International conference on circuits, system and simulation (ICCSS)*, pages 151–156. IEEE, 2017.

[5] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Stefanos Gritzalis. Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset. *IEEE Communications Surveys & Tutorials*, 18(1):184–208, 2015.

[6] Omar Nakhila and Cliff Zou. User-side wi-fi evil twin attack detection using random wireless channel monitoring. In *MILCOM 2016-2016 IEEE Military Communications Conference*, pages 1243–1248. IEEE, 2016.

[7] Ravikanth Pappu, Ben Recht, Jason Taylor, and Neil Gershenfeld. Physical one-way functions. *Science*, 297(5589):2026–2030, 2002.

[8] G Edward Suh and Srinivas Devadas. Physical unclonable functions for device authentication and secret key generation. In *Proceedings of the 44th annual design automation conference*, pages 9–14, 2007.

[9] Laura Vegh. A survey of privacy and security issues for the internet of things in the gdpr era. In *2018 International Conference on Communications (COMM)*, pages 453–458. IEEE, 2018.

[10] John Ross Wallrabenstein. Practical and secure iot device authentication using physical unclonable functions. In *2016 IEEE 4th international conference on future internet of things and cloud (FiCloud)*, pages 99–106. IEEE, 2016.