

Article

IPChain: Blockchain-Based Security Protocol for IoT Address Management Servers in Smart Homes

Bello Musa Yakubu ¹, Majid Iqbal Khan ² and Pattarasinee Bhattarakosol ^{1,*}

¹ Department of Mathematics and Computer Science, Faculty of Science, Chulalongkorn University, Bangkok 10330, Thailand

² Department of Computer Science, COMSATS University, Islamabad 45550, Pakistan

* Correspondence: pattarasinee.b@chula.ac.th

Abstract: The dynamic host configuration protocol (DHCP) servers are forms of an Internet of Things (IoT) address management server (IoTAMS) that gives network configuration settings to newly connected hosts. Administrators of a network may save time by setting DHCP servers instead of every network node. However, the absence of a more robust authentication method for DHCP servers makes hosts susceptible to attacks since neither the server nor the users are able to check the other's authenticity during DHCP connections. These concerns result in both internal and external threats to the system that have the potential to impair network services. Among these threats are malicious DHCP servers and DHCP starvation. This paper aims to provide a novel approach for tackling these issues and protect the DHCP protocol. The proposed model uses the Diffie–Hellman key exchange mechanism, the elliptic curve discrete logarithm problem (ECDLP), a one-way hash function, blockchain technology, and a smart contract. In addition, registration and validation processes provide support for the proposed model in combating DHCP risks for both internal and external system threats. Results from this study show that the proposed model has an average of 21.1% more resistance to a growing number of adversaries than the benchmark models, thus revealing that the model is better suited for the security of IoT address management servers in smart homes, thereby enhancing resilience against related threats and the success of IP address management.

Keywords: DHCP; blockchain; IoTAMS; smart homes; IoT; authentication; smart contract



Citation: Yakubu, B.M.; Khan, M.I.; Bhattarakosol, P. IPChain: Blockchain-Based Security Protocol for IoT Address Management Servers in Smart Homes. *J. Sens. Actuator Netw.* **2022**, *11*, 80. <https://doi.org/10.3390/jsan11040080>

Academic Editors: Mohamed Benbouzid, Leandros Maglaras and Mohamed Amine Ferrag

Received: 31 October 2022

Accepted: 21 November 2022

Published: 24 November 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

As the amount of data gathered and processed increases, it becomes increasingly challenging to safeguard sensitive data and information from cyberattacks [1]. Consequently, it is crucial that the network's security system be expanded and bolstered so that undesirable occurrences such as data theft and access abuse do not occur; especially on networks that include private data that can only be viewed by a restricted number of users. The design of an enterprise-level computer network allows all devices to access the same resources [2]. This will make it easy to keep track of many distinct types and amounts of services between different network users.

An IoT address management server (IoTAMS) is the mechanism through which a newly connected host receives information about network settings [3,4]. Such setups include internet protocol (IP) address, domain name server, gateway, subnet mask, and lease length. Using IoTAMS, an information technology (IT) administrators can save time by not configuring these preferences on every network machine. However, there are many different kinds of IoTAMS, but one of the most used right now is the dynamic host configuration protocol (DHCP) [4,5].

During DHCP packet exchanges occur, however, there is no verification of the packets' provenance [6]. Moreover, neither the DHCP server nor the users can verify the authenticity of one another during communications. Thus, when services are broadcast via DHCP, they

are susceptible to intrusion threats from both inside and outside the network. A rogue DHCP server or a DHCP starvation attack are both examples of a DHCP threats [7–9].

Several cryptographic protocol security methods, including that of Abou El Houda et al. [10], Yang and Mi [11], Lee et al. [3], Dinu and Togan [12], and Yao et al. [13], have been offered in order to defend the DHCP protocol in which some of them relies on digital certificates. Examples include the technique described by Yao et al. [13], which requires that both the user and server to have their respective digital certificates mounted on the host before executing the protocol. Digital certificates and digital signatures added to each communication may be used to certify the identities of users and servers. However, these protocols need significant changes to be compatible with the regular DHCP protocol [13–16].

The technique of Younes [15] and Yao et al. [13] provided a suitable way based on shared-key DHCP and the Diffie–Hellman key exchange mechanism. The method tries to circumvent DHCP's party authentication issues caused by MAC address spoofing. Instead of relying on a host's MAC address, the authors of this paper suggested using the processor identification and disc value to verify a DHCP client. However, these static parameters can be intercepted throughout the network. The protocol employs symmetric cryptography; however, the client-server shared key update method is ambiguous. Similarly, traffic analysis may be used to decrypt communications that have been encrypted with the same key [8].

In response to the security challenges highlighted above, as well as the issues raised by the cryptographic techniques utilized before to ensure security during communications in the smart home scenario, blockchain technology can be leveraged to provide a more secure and private solution [17–19]. Because of the immutability, security, and flexibility of blockchain, a blockchain-based method for protecting IoT devices during communications can be built. Blockchain technology can also be used to address DHCP protocol difficulties related to user and server authentication and verification [20,21].

This study provides a unique way to address the shortcomings of previous approaches and safeguard the DHCP protocol via the use of blockchain and smart contract technologies. Similarly, by combining the security protocol with digital certificates and shared private keys, it provides a practical answer that is backwards-compatible with the original DHCP standard. To put it another way, this facilitates two-way verification between DHCP users and DHCP servers. Furthermore, prior to the initial DHCP message broadcast, the shared private keys between the DHCP user and server can be utilized to address concerns about businesses who do not have public-key infrastructure.

Contributions

First, a secure protocol for managing IP addresses in an IoT smart home based on blockchain technology is proposed. The proposed model employs the Diffie–Hellman key exchange mechanism, the elliptic curve discrete logarithm problem (ECDLP), blockchain technology, a one-way hash function and a smart contract for server-user authentication.

Second, the proposed model is supported by the registration and validation phases. Registration requires phases of registering and authenticating network components. The first approach in the validation phase is initial validation, which focuses on entity authentication and key management. The second approach is the DHCP security smart contract (DSSC) protocol, which is based on message authentication.

Thirdly, the one-way hash function and elliptic curve discrete logarithm problem (ECDLP) are utilized together with blockchain and Diffie–Hellman key exchange method in the initial validation procedure. It is used for user and DHCP server two-way authentication, session key exchange, and digital signature generation. The DSSC protocol approach uses digital signatures established during the initial validation procedure to validate the authenticity of DHCP packets sent between the user and server.

The remainder of the paper is structured as follows. The related works are discussed in Section 2. In Section 3, the system model is presented. In Section 4, the IPChain

framework is presented. In Section 5, the security analysis is presented. Section 6 contains the performance analysis results. In Section 7, we discuss how our model can be applied in the real world and in the business world. In Section 8, the study concludes with a summary.

2. Related Works

DHCP authentication in a dispersed network, such as a smart home, has not been the focus of as many investigations as network security in traditional networks. Most of the studies use standard network security protocols, such as Kerberos. For example, a validation and key distribution center using Kerberos has been demonstrated by Hornstein and Aboba [22] to provide users and servers with encrypted tickets. In DHCP communications, either the user or the server will need to provide an authentication ticket. However, Kerberos server failure owing to a single point of failure exists, and exact timekeeping between clients and servers is required for ticket timestamps to avoid replay attacks [23].

Shete et al. [24] and R. Droms [25] verified DHCP packets using a setup key and a postponed verification technique. To configure the system, client and server secretly exchanged tokens. This method only protects unintentional DHCP servers, since DHCP connections lack authentication [3,15]. Using a common private key and the MD5 message-digest method, the approach also generates DHCP message authentication codes. However, collision attacks may detect collisions in only seconds on a computer, reducing the security of MD5 [26]. These approaches authenticated entities and messages using a secret key, but did not demonstrate how the key is handled, which is crucial when dealing with several clients. The option field is limited to 255 bytes of authentication data due to DHCP's one-byte length field [9,15].

Duangphasuk et al. [27] proposed two DHCP security measures. The first method employs digital certificates for DHCP packet and organization authentication. The second, a secure DHCP with shared secret keys, authenticates DHCP packets using a shared secret key. The prospect of the digital certificate being bigger than the DHCP message wasn't accounted for in this approach. A challenge–response style authentication procedure for DHCP discovery was described in Yaibuates and Chairsricharoen [9]. When the server receives DHCPDISCOVER, it may encrypt the challenge answer and attach it to DHCPPOFFER using this method. Clients and servers must agree on the hashing method and secret key for the challenge. The researchers did not comment on how the private key is protected throughout the transit from server to user.

To address the preceding issues above, the techniques in [7,16] were proposed to secure DHCP using spoofing techniques. The approaches utilize detection and prevention modules to mitigate the threat. The detection module catches address resolution protocol (ARP) packets using an analysis tool and an algorithm. After identifying unusual network activity, DHCP snooping and ARP inspection techniques are employed to neutralize the attack and establish a more secure network. Furthermore, the authors in [28,29] employed the use of Edwards-curve digital signature algorithm (EdDSA) to avoid a rogue DHCP server attack by confirming the source and integrity of DHCP server messages, regardless of whether they originate from a legal or rogue DHCP server. The legitimacy of DHCP client-server connections is validated using digital certificates. The public's digital certificate is issued by a trustworthy server or authority. However, the digital certificate may be too big to fit inside a single DHCP packet. Similarly complex was the implementation of digital certificate revocation and a certificate-based delayed authentication has been recorded [30].

Yao et al. [13] and Xie et al. [31] suggested authenticating DHCP clients, servers, and messages using several authentication approaches such as fingerprint recognition and key agreements. This method uses a shared secret key to compute DHCP message authentication codes. The DHCP transmission's secret key is generated using a random value based on the assumption that all involved parties know it. However, the techniques failed to explain how they obtained the random number or how they derived the new secret key from the old one. Similarly, the methods for storing private key midst users and

server have single-point-of-failure issues and does not scale adequately. Table 1 presents an overview of the most relevant prior research.

Table 1. An overview of the related works.

Technique(s) Used	Objective(s)	Limitation(s)
Kerberos [22]	To provide user and servers with encrypted tickets	Single-point-of-failure, difficulties in exact timestamp management
Secretly exchange tokens and MD5 message-digest [24,25]	To authenticated entities and messages using a secret key	Prone to collision attacks, no information on how the key is managed
Digital certificates and shared secrete keys [27]	To authenticated entities and messages	The digital certificate being bigger than the DHCP message
Hashing and secret key [9]	Challenge–response style authentication procedure for DHCP discovery	No information on how the private key is protected
Spoofing techniques [7,16]	To enhance the security in DHCP processes	Prone to single-point-of-failure and man-in-the-middle attack.
Digital certificates [28,29]	To verify the legitimacy of DHCP client-server connections	Prone to single-point-of-failure, digital certificate may be too hefty for DHCP packet, high authentication latency
Fingerprint recognition and key agreements [13,31]	To authenticate DHCP clients, servers	Prone to single-point-of-failure and scalability issues, no details on how the random number or new secret key are produced.

3. System Model

This section contains a description of the proposed architecture’s overview. The section begins with a study of the network model, followed by a discussion of the DHCP server attack model and a description of the adversary’s characteristics in the attacker model.

3.1. Network Model

The system consists of a private Ethereum blockchain network, with a DHCP server (DS) linked to cluster head devices (CHD) over the blockchain home network as given in Figure 1. Each CHD is further linked to a group of smaller smart home devices making up its unit. A smart home network administrator (Admin) is the owner of a private network that controls the network and registers all devices using the proof-of-authority (PoA) consensus algorithm. Each CHD as well as DS and Admin utilizes a distinct Ethereum accounts, thus, they all have a unique Ethereum address (EA) for network communication. We assume in this study that both the DS, CHD and Admin devices are computationally capable of handling the blockchain ledger, therefore the CHD is employed to relieve the smart home devices of the computing constraint imposed by the blockchain ledger. And hence, the DS, CHD and Admin are regarded as the primary stakeholders in this study. Table 2 provides the descriptions of all the notations used in this paper.

3.2. DHCP Server Attack Model

The attack against DS may occur in two major categories: from both within and outside of the smart home network. The attack from outside the smart home network is carried out by external users or attackers who want to disrupt the network or prevent the server from performing its job. The attackers impersonate a valid user or a legitimate DS, intercepting the requests/queries of both legitimate users and DHCP servers and responding with bogus responses. This method can result in a denial-of-service attack.

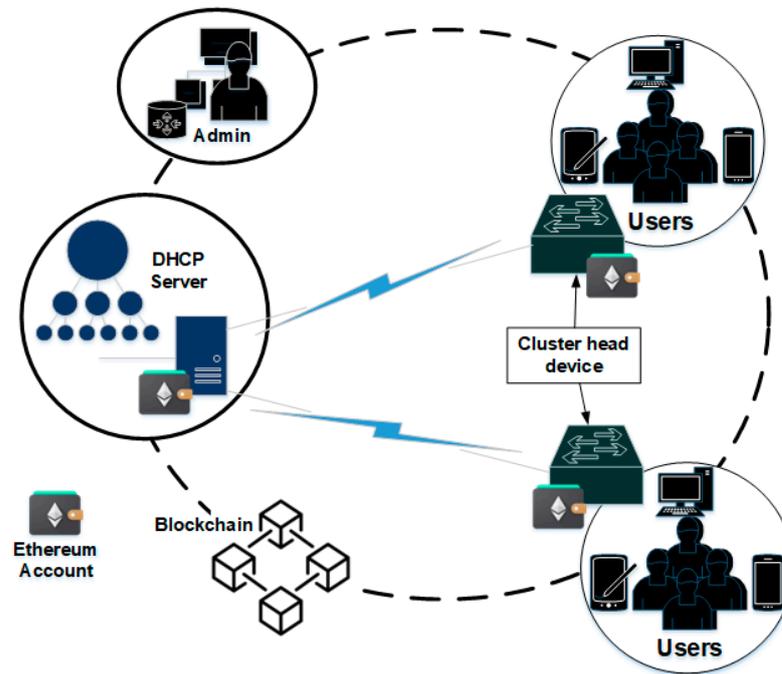


Figure 1. Proposed system model.

Table 2. Notations.

Symbols	Descriptions	Symbols	Descriptions
DS	DHCP Server	DS_{ED}	EA of DS
CHD	Cluster head devices	w, x, y, z	Other DS parameters
EA	Ethereum Address	$Pcode$	Password verification code
$User_{id}$	User identity	μ_1, μ_2	Random values
CHD_{EA}	EA of CHD	DS_{pkey}	DS public key
$User_{pwd}$	User password	$\varphi, \delta, \varphi_1, \rho, w_1, w_2, \eta$	Authentication parameters
$User_{MAC}$	User MAC address	$h(.)$ and $H(.)$	One-way hash function and elliptic curve point map function
DS_{skey}	DS private key	DS_{IP}	DS IP address
P	Point on the elliptic curve	$Offer_{IP}$	Offered IP address
σ	Replay detection value	LT	Lease term
$Sign_{User}$	User signature	$Sign_{DS}$	DS signature

Two types of attacks can be launched from inside a smart home network: DHCP exhaustion and attacks from rogue servers. As part of a DHCP exhaustion attack, the adversary utilizes all available IP addresses to overrun the service [2]. Since the DS is unable to distinguish between a legal host and a spoofed one, it distributes all client network numbers to any user who wants them. Once all accessible IP addresses have been allocated to spoofed users, any legitimate users attempting to acquire an IP address will be left without an IP connection.

During attacks from a DHCP rogue server, an adversarial user masquerades as a DS and answers the requests with a fake IP address. Both the malicious and the authentic DHCP server will receive the DHCPDISCOVER message and supply IP addresses and the

default gateway when users join a network. In response to a DHCP request, the malicious server returns incorrect parameters. It is conceivable that the default gateway, DNS server, or IP address information is incorrect. If the malicious DHCP server becomes the default gateway for the network, it will have access to all network data. Consequently, they may access, modify, and steal any data sent by the infected system.

3.3. Attacker Model

The following characteristics are associated with our attacker in this study:

- A rogue user may acquire illegal network access and then utilize network services without authorization.
- A malicious user initiates a DHCP starvation attack, which exhausts the server's available IP addresses.
- The malicious user can deploy a malicious server and execute attacks unique to such a service.
- In this study, it is assumed that the adversary cannot compromise the blockchain, the DS, the CHD, or the admin devices; hence, these devices are tamper-resistant.

Remarkably, the proposed scheme does not account for the possibility that blockchain, DS, CHD, or Admin devices could be compromised. Consequently, this scenario is the most significant limitation of the proposed method that we aim to address in our future work.

4. Proposed IPChain Model

The proposed IPChain model is built using one-way hash function, Diffie–Hellman key exchange mechanism, elliptic curve discrete logarithm problem (ECDLP), and smart contract. The ECDLP was used due to its fast computation capabilities compared to other public key cryptosystems. The proposed model involves two phases: the registration phase and the validation phase.

4.1. Registration Stage

Initially, the admin registers the EAs of both the DS and CHD in the private network and copies of the EAs are then stored in the private blockchain ledger as demonstrated in Figure 2. The admin assigns an identity $User_{id}$ and password $User_{pwd}$ to the user's device and then maps the user's device to its corresponding CHD. Then, $User_{id}$, $User_{pwd}$, and MAC address ($User_{MAC}$) of the user's device are shared with all primary stakeholders via the blockchain network and a copy is also stored in the private ledger. The DS chooses a private key DS_{skey} and computes its public key using a point (P) on the elliptic curve as: $DS_{pkey} = DS_{skey} \cdot P$, then using its EA (DS_{ED}), it broadcasts all its parameters as $DS_{ED} \left(DS_{pkey}, P, w, x, y, z \right)$ on the blockchain. Similarly, using a one-way hash function and DS_{skey} , the DS computes a password verification code as given in Equation (1), and then stores it in its memory. Figure 2 gives the detail processes applied during the registration stage of the proposed IPChain model.

$$Pcode = h \left(User_{id} \parallel DS_{skey} \right) \oplus User_{pwd} \quad (1)$$

4.2. Validation Stage

Two sub-techniques can be used to complete the validation process in this study: the initial validation procedure and the DHCP security smart contract (DSSC) protocol, which are explained in Figure 3 and Algorithm 1 respectively. When a user, whether adversary or legitimate, wants to connect to a private network with DS as its IoTAMS, the two validation procedures will always be in effect.

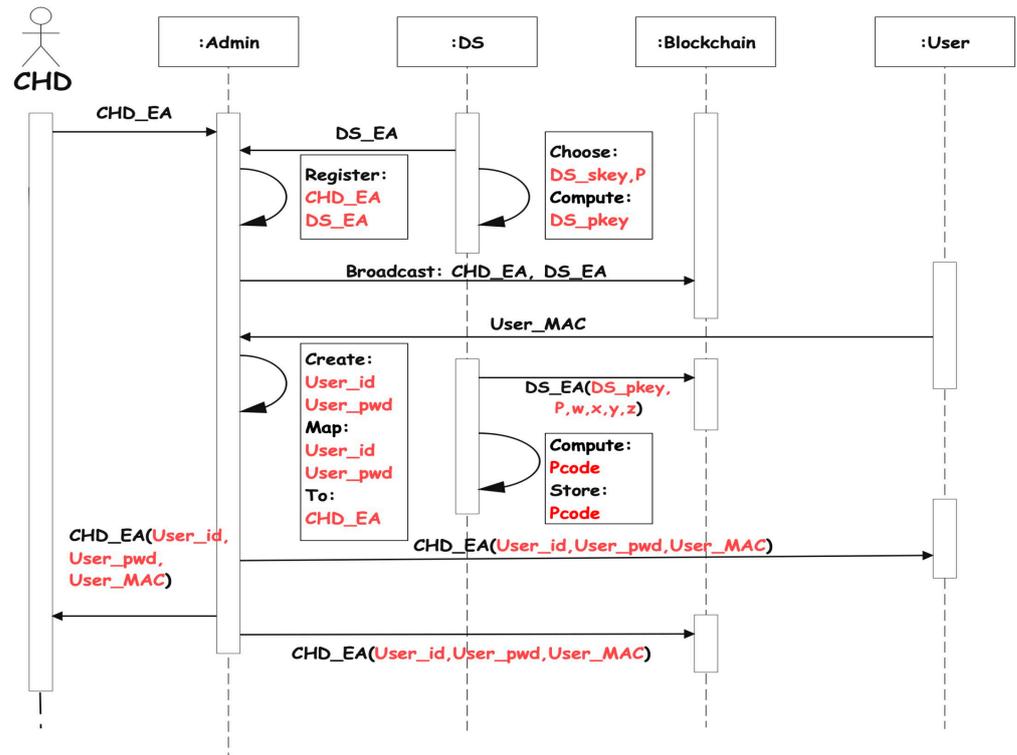


Figure 2. IPChain registration stage.

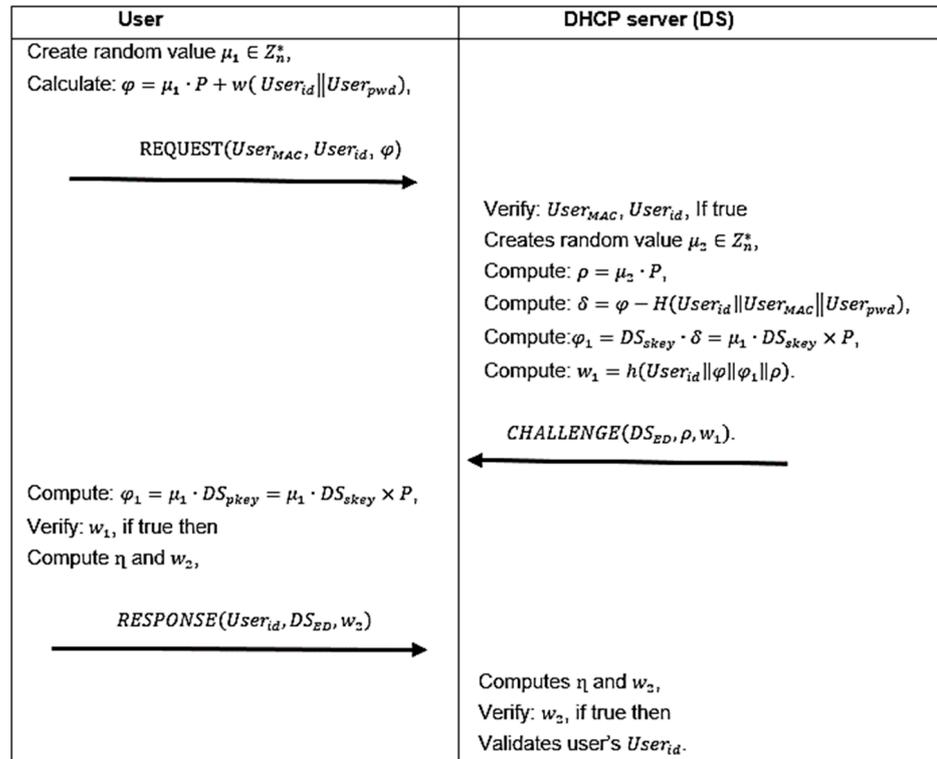


Figure 3. IPChain validation stage.

4.2.1. Initial Validation Procedure

First, a random value μ_1 is created by the user's device with id $User_{id}$, then it utilizes the DS-broadcasted values p and w to calculate $\varphi = \mu_1 \cdot P + w \cdot h\left(User_{id} \parallel User_{pwd} \right)$ as demonstrated in Figure 3. The user's $User_{MAC}$, $User_{id}$, and value φ are then submitted as constituents of a REQUEST message to the DS as $REQUEST(User_{MAC}, User_{id}, \varphi)$.

Second, assuming the DS has received the request message, the DS then verifies the user's $User_{MAC}$ and $User_{id}$. If the DS cannot locate the $User_{MAC}$ and $User_{id}$, the request is rejected. If not, the DS creates a random value μ_2 and uses the database-stored, $User_{id}$ -bent $Pcode$ together with the transmitted value φ to derive the user's password $User_{pwd}$ and the values δ , φ_1 , and ρ to derive the value w_1 as: $\rho = \mu_2 \cdot P$, $\varphi_1 = DS_{skey} \cdot \delta = \mu_1 \cdot DS_{skey} \times P$, $w_1 = h\left(User_{id} \parallel \varphi \parallel \varphi_1 \parallel \rho \right)$. Where $\delta = \varphi - H\left(User_{id} \parallel User_{MAC} \parallel User_{pwd} \right)$, while $h(\cdot)$ and $H(\cdot)$ are the one-way hash function and elliptic curve point map function, respectively. With this, using its EA, the DS sends a challenge message containing ρ , w_1 to the requesting user in the form of $CHALLENGE(DS_{ED}, \rho, w_1)$.

Third, on reception of the challenge message, the user calculates $\varphi_1 = \mu_1 \cdot DS_{pkey} = \mu_1 \cdot DS_{skey} \times P$ to verify $w_1 = h\left(User_{id} \parallel \varphi \parallel \varphi_1 \parallel \rho \right)$. If w_1 is determined to be different from what was received, the user-DS connection is terminated. If not, the user must compute η which is used for computing w_2 as follow: $\eta = \mu_1 \cdot \rho$, $w_2 = h\left(User_{id} \parallel DS_{ED} \parallel \varphi \parallel \varphi_1 \parallel \rho \parallel User_{pwd} \parallel \eta \right)$. The user then sends a RESPONSE message to the DS as $RESPONSE(User_{id}, DS_{ED}, w_2)$.

Lastly, after receiving the response message, the DS computes η and its w_2 and then verifies it with the received one. Using the shared and unique session key η , the DS validates user $User_{id}$ if the computed value for w_2 matches the one received in the response message. Figure 3 provides the detail processes applied during validation stage of the proposed IPChain model.

4.2.2. DHCP Security Smart Contract (DSSC) Protocol

Motivated by [15], DHCP security smart contract (DSSC) protocol supports the message verification mechanism in this study, which is an improvement on DHCP that provides a verification mechanism for DHCP packets, to guard against DHCP threats. Since DSSC is an improvement on DHCP, the DS system's requirements for message exchange, timeout, and caching are identical to those of the existing DHCP servers [28,31]. Therefore, DSSC messages may be handled by users without an installed DSSC module. Notwithstanding, for a secure LAN, it is suggested that all users use DSSC. The following subsections describes the process of message verification.

DSSC Protocol Layout Requirements

Certain design requirements must be met to ensure that the DSSC technique is well-matched with the existing DHCP application. These include: the proposed method must first be compatible with the existing format for verification options. Second, the finite state of the DHCP client or server cannot be modified, meaning no new states may be added. Thirdly, it is prohibited to send additional DHCP messages. Users who do not utilize the proposed protocol must still have access to the server's settings.

As per the standard, DHCP messages cannot be fragmented, which is one of the most significant restrictions. In other words, the Ethernet maximum transmission unit (MTU) imposes a maximum packet size of 1500 bytes on DHCP packets [15]. After deducting the size of the IP, server, and user datagram protocol headers, the overall length of a DHCP packet is inadequate. A digital certificate or extensive encryption keys are too large to transmit over this connection.

DSSC Protocol Process

The DSSC protocol verifies each DHCP transmission prior to delivery. A DHCP user first sends a discovery message to locate the DS. The user specifies DSSC in the

packet's option field. The RFC 3118 structure must be used for DHCP protocol entity and message verification to utilize the option field in server packets. The replay detection field, which consists of a timer storing the present date and time that increases monotonically, should also be utilized to avoid replay threats. Algorithm 1 depicts user and DS messages for DSSC.

If the option field of the user indicates the DSSC protocol is disabled, the DS will send the DHCPDISCOVER message to its system. When this is not the case, the protocol examines the message prior to forwarding it to the DS system. The replay detection field is tested for correctness; if the test fails, messages are not transmitted, else the DS will generate a DHCPOFFER message (Line 3–16, Algorithm 1). The server allocates the user an IP address upon receipt of the DHCP OFFER packet. This message contains the user's MAC address, DS's IP address (DS_{IP}), offered IP address ($Offer_{IP}$), lease term (LT), and cluster head device EA.

Option field of the DHCPOFFER message contains the DS signature for message validation and authorization. The DS signature ($Sign_{DS}$) comprises the user's MAC address, the DS's EA, the user's EA, the cluster head device EA, the replay detection value (σ) and other parameters as depicted in Equation (2). Modifications are also made to the replay detection field to avoid replay threats.

$$Sign_{DS} = h(DS_{EA} || \rho || \eta || User_{pwd} || User_{MAC} || Offer_{IP} || DS_{IP} || CHD_{ED} || User_{id} || \sigma) \quad (2)$$

On receipt of the message at the user end, if the verification option field of the DHCPOFFER message is said to be true, the protocol forwards it to the DHCP system; otherwise, it is handled by the user. When this option is enabled, the replay detection field is examined by the protocol. If the value of the replay detection field is greater than previously, processing will proceed. However, the message is deleted if the event does not occur.

DSSC protocol verifies the DS's signature and the authenticity of the message after examining the replay detection field. The DS Signature is determined after extracting the message's $User_{MAC}$, $User_{id}$, DS_{IP} , CHD_{ED} , and σ for signature verification. If the user's $User_{MAC}$ and $User_{id}$ match those in the DHCPOFFER message and the computed DS Signature matches the one in the DHCPOFFER message, then the DHCPREQUEST is produced. Messages that do not match these requirements are eliminated (Line 17–37, Algorithm 1).

In response to DHCPOFFER, the DHCPREQUEST message is delivered to accept the IP address. DHCPREQUEST and DHCPOFFER have fields that are substantially identical. The recipient modifies the field for replay detection. Line 45–47 of Algorithm 1 demonstrates that the user's signature (Equation (3)) is provided in the DHCPREQUEST packet to validate and verify it. The modified replay detection value is then applied to the creation of a new user signature.

$$Sign_{user} = h(User_{id} || \rho || \eta || User_{pwd} || User_{MAC} || User_{IP} || DS_{IP} || CHD_{ED} || User_{id} || \sigma) \quad (3)$$

Upon receiving a DHCPREQUEST packet, the DS checks the user's replay detection field and signature. The relevant validation fields are obtained (σ , $User_{IP}$, DS_{IP} , CHD_{ED} , and $User_{MAC}$). Next, the user's signature is generated and compared to the one obtained in the DHCPREQUEST. If the DS validates the replay detection field and signature of the user, it will prepare the DHCPACK message (Line 38–56, Algorithm 1). DHCPACK contains all configuration data, thus, the validation of DHCPACK is comparable to that of DHCPOFFER. The DHCP user configures the network interface after validating DHCPACK using approved settings.

Algorithm 1: DSSC protocol

```

Input:  $User_{MAC}, User_{id}, CHD_{ED}, DS_{ED}, DHCPDISCOVER, \sigma$ 
Output:  $DHCPOFFER_{DS}, DHCPREQUEST_{user}, DHCPACK_{DS}$ 
1. Function: Submit ( $DHCPDISCOVER$ )
2.    $CHD_{ED}(User_{id}, DHCPDISCOVER) \rightarrow DS_{ED}$ 
3. Function: Compute: ( $DHCPOFFER$ )
4.   For all  $DHCPDISCOVER$  received,
5.     If DSSC protocol field = = enabled,
6.       Check for replay detection field correctness,
7.       If replay detection field is effective  $\sigma$ ,
8.         Compute  $DHCPOFFER_{DS}$ ,
9.    $DHCPOFFER_{DS} = DS_{ED}(User_{MAC}^*, User_{id}^*, DS_{IP}, Offer_{IP}, LT, CHD_{ED}) + Sign_{DS}$ ,
10.    Modify replay detection field value to  $\sigma_{t+1}$ ,
11.    Transmit  $DHCPOFFER_{DS}$ ,
12.    Else  $DHCPDISCOVER$  messages are not transmitted.
13.  End If
14.  Else send the  $DHCPDISCOVER$  message to system.
15. End If
16. End For
17. Function: Compute: ( $DHCPREQUEST$ )
18.  For all  $DHCPOFFER_{DS}$  received,
19.    If DSSC protocol field = = enabled
20.      Compute replay detection field value  $\sigma_{t+2}$ ,
21.      Check for replay detection field correctness (i.e., if  $\sigma_{t+2} > \sigma_{t+1}$ ),
22.      If replay detection field is effective,
23.        Extract  $DHCPOFFER_{DS}$ ,
24.        Compute  $Sign_{DS}^*$ ,
25.        Verify  $Sign_{DS}$ ,
26.        If ( $Sign_{DS} == Sign_{DS}^* \& User_{MAC} == User_{MAC}^* \& User_{id} == User_{id}^*$ )
27.          Compute  $DHCPREQUEST_{user}$ ,
28.     $DHCPREQUEST_{user} = CHD_{ED}(User_{MAC}, User_{id}, DS_{IP}, User_{IP}, LT, CHD_{ED}) + Sign_{user}$ ,
29.      Modify replay detection field value to  $\sigma_{t+3}$ ,
30.      Transmit  $DHCPREQUEST_{user}$ ,
31.      Else  $DHCPREQUEST$  messages are not transmitted.
32.    End If
33.    Else  $DHCPREQUEST$  messages are not transmitted.
34.  End If
35.  Else send the  $DHCPREQUEST$  message to system.
36. End If
37. End For
38. Function: Compute ( $DHCPACK$ )
39.  For all  $DHCPREQUEST_{user}$  received,
40.    If DSSC protocol field = = enabled
41.      Compute replay detection field value  $\sigma_{t+4}$ ,
42.      Check for replay detection field correctness (i.e., if  $\sigma_{t+4} > \sigma_{t+3}$ ),
43.      If replay detection field is effective,
44.        Extract  $DHCPREQUEST_{user}$ ,
45.        Compute  $Sign_{user}^*$ ,
46.        Verify  $Sign_{user}$ ,
47.        If ( $Sign_{user} == Sign_{user}^* \& User_{MAC} == User_{MAC}^* \& User_{IP} == Offer_{IP}$ )
48.          Compute  $DHCPACK_{DS}$ ,
49.          Transmit  $DHCPACK_{DS}$ ,
50.        Else  $DHCPACK$  messages are not transmitted.
51.      End If
52.      Else  $DHCPACK$  messages are not transmitted.
53.    End If
54.    Else send the  $DHCPACK$  message to system
55.  End If
56. End For

```

5. Security Analysis

In this section, we will discuss how the proposed IPChain model can mitigate the most significant security threats that a DS or user may run into in a smart home network. The security vulnerabilities considered here include rouge DS attack, DS starvation attack, attack by impersonating a user, stolen-verification codes, and brute-force attack on passwords. Table 3 provides several parameters used in curtailing these security vulnerabilities in this work.

Table 3. Security analysis table.

Security Threats	Blocking Parameters
Rogue DS Attack	EA, event logs, $User_{pwd}$, DS_{skew} , μ_1 and $Pcode$
DS Starvation Attack	$Sign_{user}$, ρ , η , and $User_{pwd}$,
Attack by impersonating a user	φ and μ_1
Stolen-Verification Codes	DS_{skew}
Brute-Force Attack on Passwords	φ , ρ , φ_1

5.1. Rouge DS Attack

The rogue DS of an attacker is a non-administrated DHCP server. An adversary may launch a man-in-the-middle attack by configuring a rogue DS as the default gateway and domain name server. To address this problem, the Ethereum blockchain employs 20-byte EAs that are assigned to each network node. This guarantees that all major network stakeholders are notified of any changes made to the blockchain network through the event logs. Before the update can be implemented, all linked stakeholders must approve it. Since all created events are tamper-resistant and signed by the smart contract, it is safe against such attacks and will be found even if an attacker attempts to change or modify an EA or the event logs.

Furthermore, by requiring DS to identify itself to the user during validation and DHCP message transmission, DSSC protocol avoids DHCP rouge server attacks. The server validation procedure depends on $Pcode$ and φ_1 for w_1 . The attacker needs to know $User_{pwd}$, DS_{skew} , and μ_1 to create φ_1 . Even if an attacker knows the user’s password, they cannot produce $Pcode$ without the high-entropy secrets DS_{skew} and μ_1 . The password verifier and the secret DS_{skew} are required for the attacker to obtain the user’s password.

To impersonate a DS, an authorized user must supply the genuine DS’s signature after authentication and during DHCP message exchange. This requires both secret DS information and a password. Assuming the attacker is aware of ρ , they must predict μ_1 and μ_2 , which is impractical. To calculate η , the adversary must additionally solve the ECDLP. Given ρ , and P , μ_1 and μ_2 cannot be calculated in polynomial time. Online password guessing is impractical because DS_{skew} is an unguessable, high-entropy value and $Pcode$ is unavailable to the attacker.

5.2. DS Starvation Attack

The objective of a DS starvation attack is to exhaust the DS’s available IP addresses by delivering DS DHCPREQUEST packets with faked MAC addresses. Consequently, the DS does not provide victim network users with the desired IP address. The adversary may then either install a malicious DS on the network or designate their workstation as the default gateway and begin sniffing network data.

The proposed IPChain model eliminates DS starvation attacks by having the DS verify the message’s signature whenever it receives a DHCPREQUEST packet from a user. DSSC protocol DS will not process DHCPREQUESTs without user signatures. If not, the DS parses the message header for $User_{id}$, $User_{MAC}$, $User_{IP}$, DS_{IP} , and σ . The user’s signature is generated by utilizing $User_{id}$ and $User_{MAC}$ to search up $User_{id}$, ρ , η , and $User_{pwd}$ in the private ledger. Comparing the created signature to the user’s signature. Consequently, discrepancies will block the transmission of request messages.

To perform a DS starvation attack, an attacker must modify the MAC address in the DHCPREQUEST packet’s header and affix a user signature; the DS will recognize the difference between the two. The starvation attack against DS needs access to several user signatures. If the attacker knows $User_{id}$ and $User_{MAC}$, they must estimate a user’s ρ , η , and $User_{pwd}$, which will be challenging. This becomes much more difficult as the number of users increase.

5.3. Attack by Impersonating a User

This sort of attack is also known as a masquerade, spoofing, or malicious user threat. Without permission, the adversary assumes the identity of a genuine user. Given the requirement of the client parameters $User_{pwd}$, $User_{id}$, $User_{MAC}$, and φ for an adversary to assume the identity of user $User_{id}$, the proposed model is able to withstand such an attack from an adversarial user. The value of φ cannot be rightly determined by an adversary, regardless of how well they know $User_{pwd}$ since the random value μ_1 is not predictable.

5.4. Stolen-Verification Codes

Using a stolen-verification-codes attack, an adversary may breach the DS and gain access to sensitive client data, such as the verification code for an individual's login credentials. After obtaining this information, the attacker attempts to impersonate the real user. However, the model protects users against this kind of attack by preventing an attacker from deducing or guessing the user's password from the user ID and password verification code. Since the secret DS_{skew} of the DS is not saved in memory, the adversary is unable to retrieve them.

5.5. Brute-Force Attack on Passwords

An adversary makes many guesses at the private credentials (such as the user's or DS's password) and compares each one to the matching cryptographic hash in a brute-force password-cracking attack. The proposed model guards against this kind of attack by assuring that, even if an attacker passively intercepts all messages exchanged and properly deduce the password, other private values such as φ , ρ , φ_1 etc., cannot be easily deduced, due to the fact that the values are function-mapped pointers to an elliptic curve, an extremely challenging task.

6. Performance Analysis

Here, we describe the simulation environment and performance parameters utilized to develop the proposed model (IPChain). In addition, the assessment and outcomes were compared to those of current models in Yao et al. [13] and Xie et al. [31]. These models were selected because, in their respective contexts, they are both contemporary and analogous to our proposed model.

6.1. Simulation Set Up

The IPChain model was build using the solidity programming language. As a wallet for Ethereum, Metamask [32] was used. Using the Rinkeby testnet, a proof-of-authority (PoA) consensus process was employed during simulations [33]. Python was utilized throughout development to construct the model's logic via developing smart contract interactions using the Web3.py Python library and other relevant modules. Web3.py provides interaction between Python and Ethereum VM (virtual machine). This results in its use in decentralized applications (dApps), among other use cases, for connecting with smart contracts, transferring transactions, accessing block data, and executing IP network operations. The tests were conducted on a machine with a 3.41 GHz Intel(R) Core (TM) i7-6700 M processor and 16.0 GB of RAM. To remove bias and maximize observation and findings, all models were deployed in the exact same setting. Furthermore, the experiments were carried out with the following parameter settings:

1. Except for the DS and Admin devices, the smart home network population was restricted to 50 IoT devices by increasing the population for each scenario.
2. The simulation was conducted by altering the number of IoT devices in the network between 10, 20, 30, 40, and 50 devices in each scenario.
3. Each communication interaction involved the DS and a randomly selected user device. In 500 min, 500 interactions were carried out, allowing each of the 50 devices and the DS to complete a sufficient number of mutual authentication rounds [34].

4. In several testing scenarios, the percentage of adversaries in the network was set between 0%, 20%, 40%, 60%, and 80%.

Observations of the performance of each process were made via the execution of events containing sufficient data. Several situations were examined to validate the smart contract code's logic. The proposed DSSC smart contract was assessed using the Oyente tool for analyzing the security of smart contracts [35]. With 57% EVM code coverage, the results demonstrate that the smart contract is devoid of known security issues such as reentrancy, timestamp reliance, transaction dependency, and parity multisig flaws [36].

6.2. Performance Metrics and Evaluation

To analyze the performance of the IPChain, many measures have been used, including robustness during authentication, which is determined based on user authentication latency (UAL), DS message authentication latency (DAL), and finalization latency (FL), all with variable percentages of adversaries in the network. UAL is the amount of time required to authenticate a user's device, which includes the processing time for the request, response messages and challenges. The DAL latency is the amount of time required for the user and DS to process and exchange packets during validation processes. While the time required to get the network configuration parameter from DS is referred to as the FL, which equals the total of the UAL and DAL. Other metrics include the overall likelihood of the model's resilience against varied percentages of network adversaries, and the computational cost, which is calculated based on the model's average execution latency.

6.2.1. Robustness and Resiliency

As seen in Figures 4–6, the proposed IPChain model loses part of its robustness as the number of system components increases owing to an increase in processing latency. This is because as the number of user devices rises, so does the possibility that there will be a greater number of adversaries inside the system. Moreover, this increases the time necessary to process messages inside the DS for user or packet authentication. In addition, it is self-evident that the system's robustness will decrease if the number of adversaries increases while the number of users whose identities are validated remains the same. This is because an increase in the number of adversaries will lead to an increase in the number of attacks, which in turn will result in a decrease in the number of legitimate users. Notably, the DS will reduce the rate at which it processes authentication packets as the proportion of adversaries in the system increases. In turn, this stops adversaries from obtaining legitimate authentication packets. Thus, even with an increase in the number of adversaries in the system, the findings reveal that the proposed approach has a high level of resilience, since it can prevent the adversary from completing the attack effectively.

To simulate the massive interaction behaviors and DHCPREQUEST and DHCPPOFFER packets, we altered the percentage adversaries influencing the system from 20%, 40%, 60%, and 80%, as previously mentioned. During the first phase, DS and user devices utilize their random values and other identifiable values, such their identities and other shared parameters, to establish connections with one another. Moreover, when a DS or user device first connects to a network, interaction with these shared parameters becomes more likely. Consequently, various authentication and interactions that lead to these shared values across network members were pre-programmed so that the DS or user device may have sufficient shared parameters for further interactions in its early stages.

This research defines the resilience of our model to the relevant threats as the ratio of the probability of resilience against adversaries in the system to the percentage of adversaries at a given moment. In this study, the robust performance of IPChain is compared to Yao et al. [13] and Xie et al. [31] benchmark models. Figure 7 depicts the outcomes of the study, which were conducted to determine the resilience of each model against an increasing number of adversaries in the system.

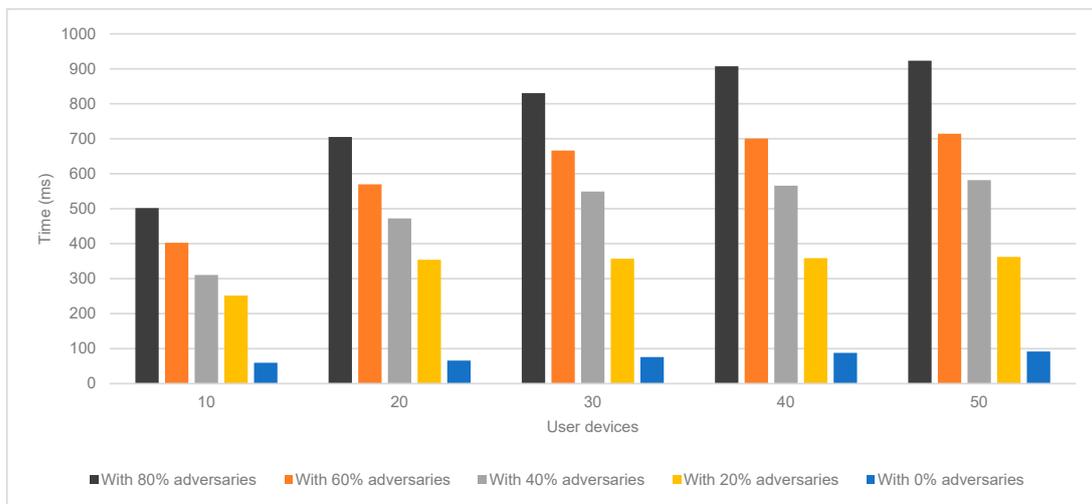


Figure 4. User authentication latency evaluation.

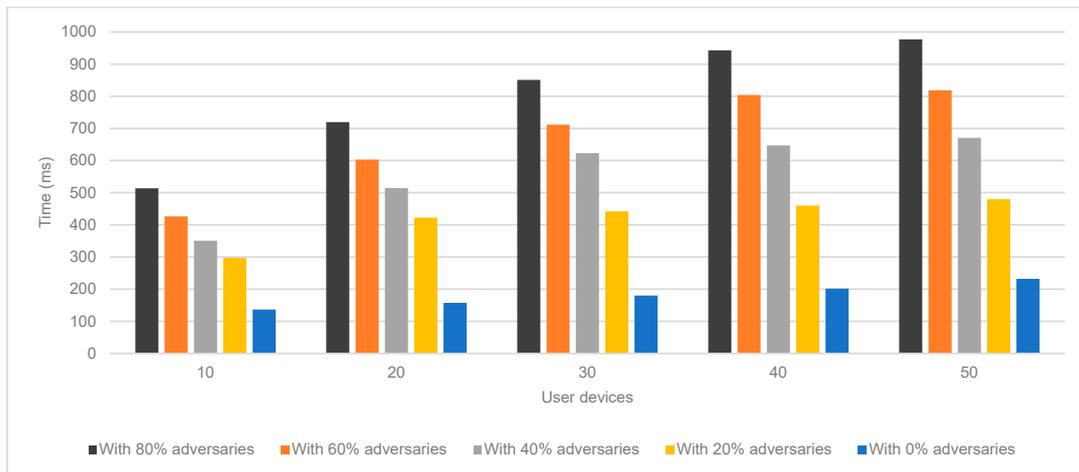


Figure 5. DS message authentication latency evaluation.

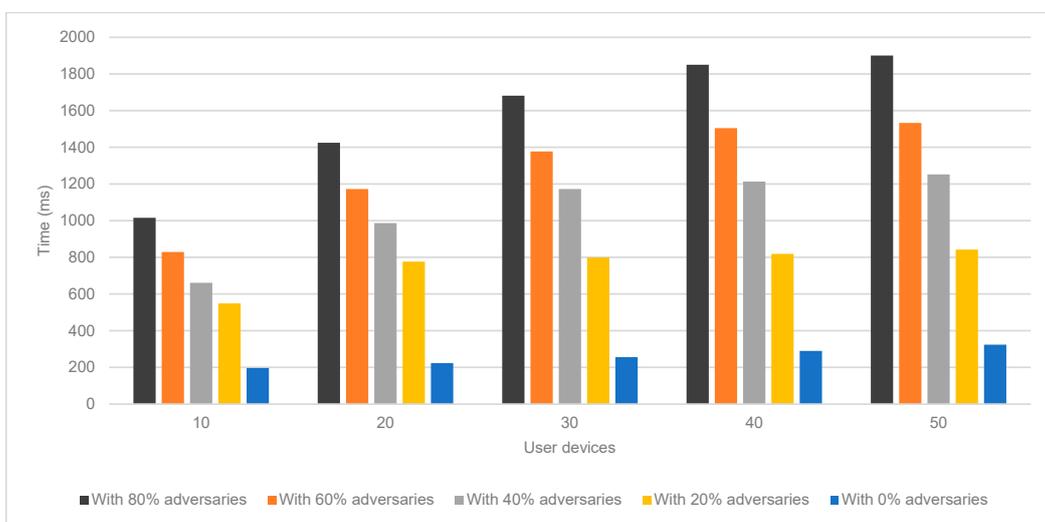


Figure 6. Finalization latency evaluation.

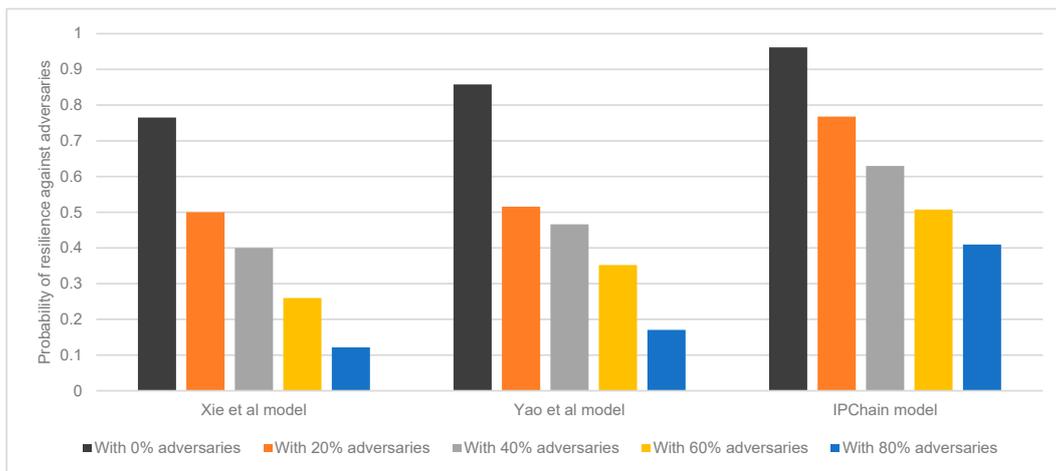


Figure 7. Evaluation of the models' resiliency against varied percentages of adversaries.

According to Figure 7, if the fraction of adversary operations that has an influence on the system rises, all models will be impacted to some degree. Comparing the results reveals that even with 80% of adversaries in the system, our model outperforms every benchmark model by a substantial margin. The proposed IPChain model outperforms Yao et al. [13] and Xie et al. [31] benchmark models by a margin of 17.8% and 24.4%, respectively. Accordingly, the IPChain model has an average of 21.1% more resistance to a growing number of adversaries than the benchmark models. Consequently, this demonstrates that the proposed IPChain model has a larger tolerance than the reference models.

The outcomes indicates that, the proposed IPChain model, which integrates the Diffie–Hellman key exchange mechanism, elliptic curve discrete logarithm problem (ECDLP), one-way hash function, blockchain technology, and smart contract features, would be more suitable for IoT address management servers' security in smart homes, boosting resistance to associated threats and IP address management success.

6.2.2. Computational Cost

In our proposed model, the authentication latency for a rising number of user devices in the network, up to a maximum of 50 devices, was recorded, measured, and compared with those of Yao et al. [13] and Xie et al. [31]. The latency was assessed based on the system's processor timer that operates during the authentication operations and network interactions. Figure 8 demonstrates that the proposed IPChain model has a higher execution latency than Yao et al. [13]. This is due to the security techniques used, which include a one-way hash function, the Diffie–Hellman key exchange mechanism, the elliptic curve discrete logarithm problem (ECDLP), blockchain technology, and smart contract features.

Furthermore, the methods presented in Xie et al. [31] and Yao et al. [13] authenticate DS–user interactions using techniques such as digital certificates and ECC, respectively. The authentication time is assessed for various numbers of nodes, where the digital certificate encryption key size ranges from 2048 to 3072 bits, while 224 to 521 bits are utilized in the case of ECC. As seen in Figure 8, the authentication execution latency when using the Xie et al. [31] method is much longer than when using the proposed scheme. However, in the proposed approach execution latency is somewhat higher in terms of Yao et al. [13], particularly when a large number of clients are involved.

However, the IPChain model ensures a safe and effective method for DS–user authentication procedures while posing a lower security risk than the benchmark models. In our system, the increase in latency is proportional to the number of transactions per unit of time. Similarly, there is an inverse relationship between latency and throughput; more transactions result in greater latency and lower throughput since more transactions need more time, hence the latency will grow as the number of transactions increases. As

described in Section 5, while the proposed method causes a delay in its operations due to this cost, it makes the network more secure by limiting several threats. In addition, it is evident that overhead costs are minimal.

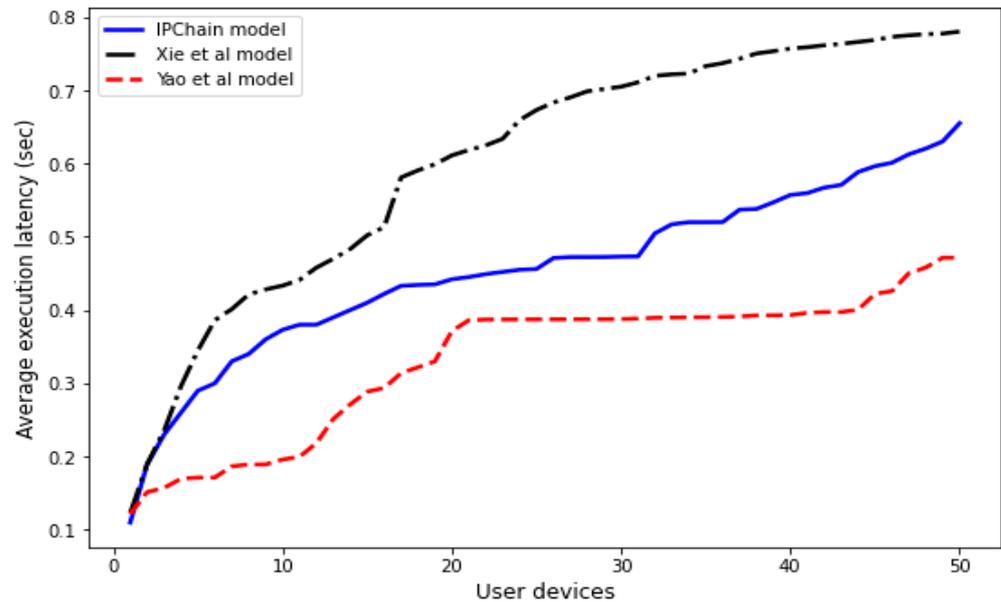


Figure 8. Evaluation of execution latencies of the models [13,31].

6.3. Discussions and Summary

In the proposed IPChain architecture, as the fraction of adversaries in the system rises, the DS will process authentication packets at a slower pace. Similarly, DS and user devices will establish connections using their random values and other distinguishable values. Such characteristics provide a high level of resilience to the model since they prevent the adversary from completing the attack successfully.

In addition, the proposed IPChain model guarantees a secure and efficient technique for DS–user authentication processes while offering a reduced security risk than the benchmark models. Increases in latency in our system are proportional to the number of transactions per unit of time. Although the proposed method creates a little delay in its operations owing to an increase in transaction costs, it increases the network’s security by mitigating several threats. Moreover, the model has an average resistance to an increasing number of attackers that is 21.1% more than the benchmark models. This illustrates that the proposed IPChain model is more tolerant than the reference models.

Table 4 compares the proposed IPChain model to the two existing models by Yao et al. [13] and Xie et al. [31]. We use the terms “Yes” and “No” to signify whether a scheme meets the assessed security standards. According to the table, neither the approach by Yao et al. [13] nor Xie et al. [31] assures the security of packets during the entire packet transmission in terms of nonrepudiation and end-to-end packet transmission verification.

Furthermore, the table demonstrates that both Yao et al. [13] and Xie et al. [31] fail to safeguard the confidentiality of packets in a smart home network. As previously shown in this study, neither Yao et al. [13] nor Xie et al. [31] can ensure complete entity or packet security during authentication procedures; moreover, new research [30,37] has uncovered weaknesses in the kind of methodologies used by both systems. Nevertheless, the proposed IPChain model meets all the security requirements evaluated in this study.

Table 4. Security performance metrics between Yao et al., Xie et al., and the IPChain models.

Security Features	Yao et al. [13] Model	Xie et al. [31] Model	IPChain Model
Packets confidentiality	No	No	Yes
Packets integrity	Yes	Yes	Yes
Packets authentication	No	No	Yes
Packets nonrepudiation	No	No	Yes
Entity authentication	Yes	Yes	Yes

7. Contextual Relevance and Applicability

This section highlights the relevance of our model to the academic and business spheres. In addition, it discusses how network managers in the public and commercial sectors can utilize the research findings to improve and better safeguard their own networks. Moreover, the section describes how the findings contribute to theory development and how academics can employ conceptual models in theory development.

7.1. Business Environment

To evaluate the viability of incorporating blockchain technology into the IoTAMS, we present a working prototype of a blockchain-based system, along with security and performance analyses. To meet the safety requirements of smart home users, a private Ethereum blockchain was used to develop and test the proposed framework. Users of smart homes are more likely to continue utilizing private blockchain systems due to the confidence that encryption of data and transactions instills. The provided solution is modifiable to accommodate the evolving requirements of individual businesses. Smart contracts are easily modifiable and deployable on private blockchain systems, ensuring quick transaction and execution times, privacy, transparency, and safety.

The proposed method can also provide two-way verification between DHCP clients and DHCP servers to safeguard sensitive data during smart home connectivity. Due to the immutability, security, and adaptability of the blockchain, the proposed solution can protect IoT devices as they communicate as an added benefit. Throughout the authentication and engagement processes, all nefarious stakeholders responsible for illegal behavior are identified, thereby reducing the potential for IoTAMS attack risks.

7.2. Useability

Multiple types of locally based networks exhibit IoTAMS-typical characteristics, making them ideal for use in smart homes (e.g., hospitals, banks, military, schools, etc.). Due to the efficacy of the proposed approach for carrying out DS–user authentication operations, this research is applicable to a variety of local-area network types. The findings of this study could be utilized by public or private sector managers to implement stricter security measures, such as DS–user authentication.

Consequently, not only smart homes but any local-area network may benefit from the DS–user authentication procedures described in this study. This effort will benefit businesses in the IT industry, IT professionals, network engineers, ISPs, online retailers, IT authorities, computer specialists, and nerds. Our method is sufficiently general that it can be easily implemented on other blockchain platforms. Given the success of our proposed model in minimizing security hostilities in relation to IoTAMS authentications, IT authorities may wish to improve upon this achievement by enacting new laws aimed at further reducing network security hostilities.

8. Conclusions

Among the most common data link layer network protocols for IP address management, DHCP is utilized by numerous IoT address management services. A DHCP server is therefore susceptible to a variety of threats, including DHCP starvation threats, DHCP rouge server threats, and malevolent DHCP user threats. In response, this paper presents IPChain, a blockchain-based security protocol for smart home IoT address management servers. The proposed IPChain model employs the Diffie–Hellman key exchange mechanism, the ECDLP, a one-way hash function, blockchain technology, and a smart contract. The ECDLP was chosen due to its superior computational speed compared to other public key cryptosystems. The proposed model has two major components: the registration and validation phases. The proposed system was analyzed for security flaws, and the results showed that the most prevalent threats to a DHCP server or user devices in a smart home network are neutralized. In addition, research into the performance of the proposed scheme revealed that it offers an average of 21.1% more resistance to a growing number of adversaries than the benchmark models.

This study has two key limitations on its application. One is that the proposed method was only tested on a limited selection of IoT devices (i.e., 50 devices). Second, there is the question of the blockchain's actual structure. Malicious activity in the context of IP address management services is complicated and impacted by several factors; thus, it will be the focus of a more in-depth analysis of stakeholder behavior associated with IP address management services in the future. Moreover, the influence of the proposed model on individual behavior in massive network settings can only be shown if the model's essential properties and building blocks are technologically feasible. Given that blockchain technology's fundamental issue is its inability to scale, it is logical to assume that the general solutions being developed to improve blockchain technology's scalability will also be relevant to IP address management services. Thus, scalability should be the primary focus of future research.

Author Contributions: Conceptualization, B.M.Y., M.I.K. and P.B.; methodology, B.M.Y.; software, B.M.Y.; validation, B.M.Y. and P.B.; formal analysis, B.M.Y.; investigation, B.M.Y.; resources, P.B.; data curation, B.M.Y. and P.B.; writing—original draft preparation, B.M.Y.; writing—review and editing, P.B. and M.I.K.; visualization, M.I.K.; supervision, P.B.; project administration, B.M.Y. and P.B.; funding acquisition, P.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Ratchadapisek Somphot Fund for Postdoctoral Fellowship through the Graduate School, Chulalongkorn University, Bangkok, Thailand. The APC was funded by Chulalongkorn University, Bangkok, Thailand. Authors are thankful for the support.

Data Availability Statement: The code of this work is available in the github public repository. Reader may avail it at the following link for academic research purposes only according to creative commons rules: <https://github.com/sysbel07/IPChain.git>. [accessed on 22 September 2022].

Conflicts of Interest: The author(s) declare that there is no conflict of interest regarding the publication of this article.

References

1. Anand, P.; Singh, Y.; Selwal, A.; Alazab, M.; Tanwar, S.; Kumar, N. IoT Vulnerability Assessment for Sustainable Computing: Threats, Current Solutions, and Open Challenges. *IEEE Access* **2020**, *8*, 168825–168853. [CrossRef]
2. Cai, X.Q.; Deng, Y.; Zhang, L.; Shi, J.C.; Chen, Q.; Zheng, W.L.; Liu, Z.Q.; Long, Y.; Wang, K.; Li, C.; et al. The Principle and Core Technology of Blockchain. *Jisuanji Xuebao/Chin. J. Comput.* **2021**, *42*, 1–15. [CrossRef]
3. Lee, K.; Kim, S.; Jeong, J.P.; Lee, S.; Kim, H.; Park, J.S. A framework for DNS naming services for Internet-of-Things devices. *Futur. Gener. Comput. Syst.* **2019**, *92*, 617–627. [CrossRef]
4. Trombeta, L.; Torrisi, N.M. DHCP Hierarchical Failover (DHCP-HF) Servers over a VPN Interconnected Campus. *Big Data Cogn. Comput.* **2019**, *3*, 18. [CrossRef]
5. Sutherland, K. DHCP (Dynamic Host Configuration Protocol). In *Understanding the Internet: A Clear Guide to Internet Technologies*; Routledge: Oxfordshire, UK, 2020.
6. Syafei, W.A.; Soetrisno, Y.A.A.; Prasetijo, A.B. Smart Agent and Modified Master-Backup Algorithm for Auto Switching Dynamic Host Configuration Protocol Relay through Wireless Router. *Int. J. Commun. Netw. Inf. Secur.* **2020**, *12*, 248–255. [CrossRef]

7. Nuhu, A.A.; Echobu, F.O.; Olanrewaju, O.M. Mitigating DHCP Starvation Attack Using Snooping Technique. *FUDMA J. Sci.* **2020**, *4*, 560–566.
8. Samuel, R.A.; Punithavathani, D.S. Designing a New Scalable Autoconfiguration Protocol with Optimal Header Selection for Large Scale MANETs. *J. Circuits Syst. Comput.* **2020**, *29*, 2050068. [[CrossRef](#)]
9. Yaibuates, M.; Chaisricharoen, R. Starvation delayed dhcp service for enabling pool recovery. *Malays. J. Comput. Sci.* **2019**, 15–34. [[CrossRef](#)]
10. Abou El Houda, Z.; Hafid, A.S.; Khoukhi, L. Cochain-SC: An Intra- and Inter-Domain Ddos Mitigation Scheme Based on Blockchain Using SDN and Smart Contract. *IEEE Access* **2019**, *7*, 98893–98907. [[CrossRef](#)]
11. Yang, Y.; Mi, J. Design of DHCP Protocol Based on Access Control and SAKA Encryption Algorithm. In Proceedings of the ICCET 2010—2010 International Conference on Computer Engineering and Technology, Chengdu, China, 16–18 April 2010.
12. Dinu, D.D.; Togan, M. DHCP Server Authentication Using Digital Certificates. In Proceedings of the IEEE International Conference on Communications, Bucharest, Romania, 29–31 May 2014.
13. Yao, Z.; Zhu, Z.; Ye, G. Achieving Resist against DHCP Man-in-the-Middle Attack Scheme Based on Key Agreement. *Tongxin Xuebao/J. Commun.* **2021**, *42*, 103–110. [[CrossRef](#)]
14. Tok, M.S.; Demirci, M. Security analysis of SDN controller-based DHCP services and attack mitigation with DHCPguard. *Comput. Secur.* **2021**, *109*, 102394. [[CrossRef](#)]
15. Younes, O.S. A Secure DHCP Protocol to Mitigate LAN Attacks. *J. Comput. Commun.* **2016**, *4*, 39–50. [[CrossRef](#)]
16. Adjei, H.A.S.; Shunhua, M.T.; Agordzo, G.K.; Li, Y.; Peprah, G.; Gyarteng, E.S.A. SSL Stripping Technique (DHCP Snooping and ARP Spoofing Inspection). In Proceedings of the International Conference on Advanced Communication Technology, ICACT, PyeongChang, Republic of Korea, 7–10 February 2021.
17. Tahir, M.; Sardaraz, M.; Muhammad, S.; Khan, M.S. A Lightweight Authentication and Authorization Framework for Blockchain-Enabled IoT Network in Health-Informatics. *Sustainability* **2020**, *12*, 6960. [[CrossRef](#)]
18. Fan, Q.; Chen, J.; Deborah, L.J.; Luo, M. A secure and efficient authentication and data sharing scheme for Internet of Things based on blockchain. *J. Syst. Archit.* **2021**, *117*, 102112. [[CrossRef](#)]
19. Aggarwal, S.; Chaudhary, R.; Aujla, G.S.; Kumar, N.; Choo, K.K.R.; Zomaya, A.Y. Blockchain for smart communities: Applications, challenges and opportunities. *J. Netw. Comput. Appl.* **2019**, *144*, 13–48. [[CrossRef](#)]
20. Khan, S.N.; Loukil, F.; Ghedira-Guegan, C.; Benkhelifa, E.; Bani-Hani, A. Blockchain smart contracts: Applications, challenges, and future trends. *Peer-to-Peer Netw. Appl.* **2021**, *14*, 2901–2925. [[CrossRef](#)] [[PubMed](#)]
21. Altaf, A.; Iqbal, F.; Latif, R.; Yakubu, B.M. A Survey of Blockchain Technology: Architecture, Applied Domains, Platforms, and Security Threats. *Soc. Sci. Comput. Rev.* **2022**, 1–22. [[CrossRef](#)]
22. Hornstein, K.; Ted, L.; Aboba, B.; Jonathan, T. DHCP Authentication Via Kerberos V. IETF DHC Working Group. 2001. Available online: <https://datatracker.ietf.org/doc/draft-hornstein-dhc-kerbauth/06/> (accessed on 20 November 2022).
23. Uddin, M.A.; Stranieri, A.; Gondal, I.; Balasubramanian, V. A survey on the adoption of blockchain in IoT: Challenges and solutions. *Blockchain Res. Appl.* **2021**, *2*, 100006. [[CrossRef](#)]
24. Shete, A.; Lahade, A.; Patil, T.; Pawar, R. DHCP Protocol Using OTP Based Two-Factor Authentication. In Proceedings of the 2nd International Conference on Trends in Electronics and Informatics, ICOEI 2018, Tirunelveli, India, 11–12 May 2018.
25. Droms, R.; Arbaugh, W. Authentication for DHCP Messages. The Internet Society, Network Working Group, RFC 3118 2001. Available online: <https://www.rfc-editor.org/rfc/rfc3118> (accessed on 20 November 2022).
26. Mohammed Ali, A.; Kadhim Farhan, A. A Novel Improvement with an Effective Expansion to Enhance the MD5 Hash Function for Verification of a Secure E-Document. *IEEE Access* **2020**, *8*, 80290–80304. [[CrossRef](#)]
27. Duangphasuk, S.; Kungpisdan, S.; Hankla, S. Design and Implementation of Improved Security Protocols for DHCP Using Digital Certificates. In Proceedings of the ICON 2011—17th IEEE International Conference on Networks, Singapore, 14–16 December 2011.
28. Al-Ani, A.; Anbar, M.; Al-Ani, A.K.; Hasbullah, I.H. DHCPv6Auth: A mechanism to improve DHCPv6 authentication and privacy. *Sadhana-Acad. Proc. Eng. Sci.* **2020**, *45*, 33. [[CrossRef](#)]
29. Al-Ani, A.; Anbar, M.; Abdullah, R.; Al-Ani, A.K. Proposing a New Approach for Securing DHCPv6 Server against Rogue DHCPv6 Attack in IPv6 Network. In *Advances in Intelligent Systems and Computing*; Springer: Cham, Switzerland, 2019.
30. Farrah, D.; Dacier, M. Zero Conf Protocols and Their Numerous Man in the Middle (MITM) Attacks. In Proceedings of the Proceedings—2021 IEEE Symposium on Security and Privacy Workshops, SPW 2021, San Francisco, CA, USA, 27 May 2021.
31. Xie, W.; Yu, J.; Deng, G. A Secure DHCPv6 System Based on MAC Address Whitelist Authentication and DHCP Fingerprint Recognition. In Proceedings of the 2021 7th Annual International Conference on Network and Information Systems for Computers, ICNISC 2021, Guiyang, China, 23–25 July 2021.
32. Metamask Brings Ethereum to Your Browser. Available online: <https://metamask.io/> (accessed on 19 September 2022).
33. Rinkeby Transaction Details. Available online: <https://rinkeby.etherscan.io/tx/0xe685f0ea29afce5d5a86265e87416be613dd36878570ddd71e49cd9d6444f263> (accessed on 15 August 2022).
34. Latif, R. ConTrust: A Novel Context-Dependent Trust Management Model in Social Internet of Things. *IEEE Access* **2022**, *10*, 46526–46537. [[CrossRef](#)]
35. Luu, L.; Chu, D.H.; Olickel, H.; Saxena, P.; Hobor, A. Making Smart Contracts Smarter. In Proceedings of the ACM Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; pp. 254–269.

36. Praitheeshan, P.; Pan, L.; Yu, J.; Liu, J.; Doss, R. Security Analysis Methods on Ethereum Smart Contract Vulnerabilities: A Survey. *arXiv* **2019**, arXiv:1908.08605.
37. Pradana, D.A.; Budiman, A.S. The DHCP Snooping and DHCP Alert Method in Securing DHCP Server from DHCP Rogue Attack. *IJID(Int. J. Inform. Dev.* **2021**, *10*, 38–46. [[CrossRef](#)]