




## Article

# CoAP/DTLS Protocols in IoT Based on Blockchain Light Certificate

David Khoury<sup>1,†</sup>, Samir Haddad<sup>2,\*</sup>, Patrick Sondi<sup>3,†</sup>, Patrick Balian<sup>4,†</sup>, Hassan Harb<sup>5,†</sup>, Kassem Danach<sup>6,†</sup>, Joseph Merhej<sup>7,†</sup> and Jinane Sayah<sup>8,†</sup>

<sup>1</sup> Laboratoire d'Informatique Signal et Image de la Côte d'Opale (LISIC UR 4491), Université du Littoral Côte d'Opale, 59375 Calais, France; dkhoury@aust.edu.lb

<sup>2</sup> Department of Computer Science and Mathematics, Faculty of Arts and Sciences, University of Balamand, Koura P.O. Box 100, Lebanon

<sup>3</sup> Center for Digital Systems, IMT Nord Europe, Institut Mines-Télécom, 59650 Lille, France; patrick.sondi@imt-nord-europe.fr

<sup>4</sup> Department of Computer Science, Faculty of Engineering, American University of Science and Technology (AUST), Beirut P.O. Box 16-6452, Lebanon; pbalian@aust.edu.lb

<sup>5</sup> College of Engineering and Technology, American University of the Middle East, Egaila 54200, Kuwait; hassan.harb@aum.edu.kw

<sup>6</sup> Basic and Applied Sciences Research Center, Al Maaref University, Beirut P.O. Box 5078/25, Lebanon; kassem.danach@mu.edu.lb

<sup>7</sup> Computer Science Department, LaRRIS Laboratory, Lebanese University, Fanar P.O. Box 6573/14, Lebanon; joseph.merhej@ul.edu.lb

<sup>8</sup> Department of Telecom and Networks, Issam Fares Faculty of Technology, University of Balamand, Koura P.O. Box 100, Lebanon; jinane.sayah@balamand.edu.lb

\* Correspondence: samir.haddad@balamand.edu.lb

† These authors contributed equally to this work.

**Abstract:** The Internet of Things (IoT) is expanding rapidly, but the security of IoT devices remains a noteworthy concern due to resource limitations and existing security conventions. This research investigates and proposes the use of a Light certificate with the Constrained Application Protocol (CoAP) instead of the X509 certificate based on traditional PKI/CA. We start by analyzing the impediments of current CoAP security over DTLS with the certificate mode based on CA root in the constrained IoT device and suggest the implementation of LightCert4IoT for CoAP over DTLS. The paper also describes a new modified handshake protocol in DTLS applied for IoT devices and Application server certificate authentication verification by relying on a blockchain without the complication of the signed certificate and certificate chain. This approach streamlines the DTLS handshake process and reduces cryptographic overhead, making it particularly suitable for resource-constrained environments. Our proposed solution leverages blockchain to reinforce IoT gadget security through immutable device characters, secure device registration, and data integrity. The LightCert4IoT is smaller in size and requires less power consumption. Continuous research and advancement are pivotal to balancing security and effectiveness. This paper examines security challenges and demonstrates the effectiveness of giving potential solutions, guaranteeing the security of IoT networks by applying LightCert4IoT and using the CoAP over DTLS with a new security mode based on blockchain.

**Keywords:** IoT; device certificate; blockchain; PKI; authentication; CoAP; DTLS



Academic Editor: Amiya Nayak

Received: 28 October 2024

Revised: 26 December 2024

Accepted: 28 December 2024

Published: 2 January 2025

**Citation:** Khoury, D.; Haddad, S.; Sondi, P.; Balian, P.; Harb, H.; Danach, K.; Merhej, J.; Sayah, J. CoAP/DTLS Protocols in IoT Based on Blockchain Light Certificate. *IoT* **2025**, *6*, 4. <https://doi.org/10.3390/iot6010004>

**Copyright:** © 2025 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

The Internet of Things (IoT) significantly changes device communication, enabling smart technologies to integrate seamlessly into daily life. In this landscape, the Con-

strained Application Protocol (CoAP) has emerged as a lightweight and energy-efficient communication protocol tailored for IoT devices operating in resource-constrained environments. However, the decentralized nature of CoAP and the limitations of IoT devices in terms of memory size and processor power present challenges in ensuring robust security, data integrity, and privacy. The current solution, which uses the DTLS protocol with the PreSharedKey and Certificate based on X509, has been successful in traditional settings. However, it may not be suitable for IoT devices due to their limited resources and many devices within IoT ecosystems. Some research papers have suggested Lithe [1]; an integration of DTLS and CoAP for IoT was proposed in this solution, which includes a new DTLS header compression scheme aimed at significantly reducing energy consumption by leveraging the 6LoWPAN standard. In a different research paper [2], the authors argue that certificates based on PKI/CA can address several challenges that symmetric key-based solutions struggle with. These certificates allow for the authentication of objects and services across network boundaries without needing a pre-configured pair-wise state. They can also authorize communication in machine-to-machine contexts. Initial overhead estimates for certificate-based DTLS suggest that it is a feasible authentication method in numerous network scenarios. In our paper, we propose a light certificate for DTLS to reduce energy consumption based on the blockchain rather than the traditional PKI/CA interactions. We used the blockchain to store light certificates, a new approach introduced by LightCert4IoT [3]. On the server side, we have proposed a blockchain-based solution that eliminates the need for PKI/CA [4,5]. The verification of certificates is performed by allowing the server and IoT device to access the blockchain during the handshake phase. Additionally, the symmetric key or session key remains the same as in the standard and could be generated using the Diffie Hellman encryption protocol or (Pre-Shared Key) PSK.

To address these concerns, this research paper explores the implementation of LightCert4IoT [3] to CoAP over DTLS. LightCert4IoT is based on blockchain and aims to improve the security, decentralization, and data integrity of IoT communication. The paper also proposes a new modified handshake protocol in DTLS for IoT device authentication, public key verification, and establishment of the session key with the Application server. This modified DTLS for CoAP could be considered as a framework and guideline for securing IoT networks, including other IoT-constrained protocols. The main objective is to introduce a new approach for issuing certificates for IoT devices and applying strong security measures to IoT protocols such as CoAP, MQTT, and others that run over UDP (User Datagram Protocol). The inherent characteristics of blockchain, including decentralization, immutability, and transparency, offer a robust solution to the weaknesses of traditional centralized systems. We consider CoAP in this study, but the results are valid for all IoT-constrained protocols. By incorporating blockchain into CoAP, the research seeks to remove reliance on central authority there, reducing the risk of a single point of failure and increasing the IoT network's overall robustness.

The document evaluates the potential benefits of this solution, examining the challenges that may arise. To achieve these objectives, the research addresses fundamental questions such as the practical implementation of blockchain on IoT protocols; its advantages to IoT communication; and the performance implications on energy efficiency, scalability, and memory size, especially for constrained IoT devices. Our examination starts by scrutinizing the current security instruments utilized by CoAP, pinpointing their limitations. Subsequently, we propose a groundbreaking arrangement that saddles the control of blockchain to fortify IoT device security. The main contributions of this work are as follows:

- This paper presents a novel contribution in the form of a modified handshake protocol for Datagram Transport Layer Security (DTLS) that leverages the Ethereum blockchain

to improve authentication. Specifically, it verifies IoT devices' and servers' public key certificates, as outlined in references [4,5]. This approach streamlines the DTLS handshake process and reduces cryptographic overhead, making it particularly suitable for resource-constrained environments.

- Proposing a new security mode in DTLS for constrained IoT protocol CoAP by adopting LightCert4IoT certificates. The proposed DTLS method can be generally applied to other IoT-constrained protocols, although none of these protocols have been specifically addressed.
- This work is especially pertinent to the IoT landscape, tackling critical challenges such as energy consumption, processing overhead, and the security of constrained devices.

The paper is organized as follows: Section 2 lists the motivation behind the study. Section 3 discusses the related work on IoT security based on PKI/CA, addressing the issues and offering recommended solutions. Section 4 covers background and preliminaries, including an overview of blockchain technology, IoT applications, protocols, and the security of CoAP over DTLS. Section 5 provides a detailed explanation of the proposed system by comparing DTLS certificates based on PKI versus blockchain. This is followed by a security analysis in Section 6 and a summary in Section 7. Finally, the conclusions and future work are presented in Section 8.

## 2. Motivation

- IoT devices' limitations, from memory size to processor power, present challenges in ensuring robust security, data integrity, and privacy.
- The current solution, which uses the CoAP/DTLS protocol with the PreSharedKey and Certificate based on X509, has been successful in traditional settings. However, due to their limited resources, it may not be suitable for IoT devices.
- Initial overhead estimates for certificate-based DTLS suggest that it is a feasible authentication method in numerous network scenarios.
- The existing assignment of certificates is not always suitable for the Internet of Things (IoT). PKI/CA requires a large infrastructure to manage certificates, and there is a risk of having a rogue CA or subverting a CA and a single point of failure.
- Distributing certificates for numerous IoT devices is challenging and impractical in an IoT environment.
- Other solutions, like hardcoding the credentials on IoT devices, introduce the threat of reverse engineering and make it difficult to ensure privacy due to fixed IDs.
- Configuring the IoT client to use a certificate is inconvenient, particularly for non-technical users. Furthermore, assigning and provisioning millions of certificates for IoT applications is impractical.
- Verifying certificates assigned to IoT and servers during the handshake protocol consumes energy and increases the delay in establishing a session, as it relies on public key cryptography. The cryptographic per-handshake overheads add up to 15 s for a certificate-based DTLS handshake.
- A complete certificate-based DTLS (Datagram Transport Layer Security) handshake comprises up to 15 messages. These messages may need to be fragmented at the DTLS layer or below. For example, even short certificates can exceed 220 bytes in size, which surpasses the limited frame size of link layers. This fragmentation can lead to an increased total number of transmitted packets.

## 3. Related Research on IoT Security Using PKI

The main security concern in the IoT domain is device authentication, typically achieved using Public Key Infrastructure (PKI). However, traditional PKI implementa-

tions that assign a certificate to IoT devices face significant performance limitations due to their resource-constrained nature. As a result, “lightweight” PKI solutions tailored for IoT devices have been detailed in recent academic papers, particularly in the work of Mohammed El-Hajj and Pim Beune [6]. The most important of these papers are listed in the following paragraph.

### 3.1. Centralized Technologies

In their research, Forsby et al. [7,8] proposed a compact version of the X.509 certificate for IoT using compression and encoding algorithms. This lightweight certificate is compatible with all current PKI implementations. However, a limitation is that assigning the certificates to the IoT is cumbersome using traditional PKI/CA methods.

Höglund et al. [8] outlined two solutions to overcome the challenges of using PKI for the Internet of Things. They proposed a new Concise Binary Object Representation (CBOR) encoding for X.509 certificates that reduces their size. This CBOR encoding is particularly suitable for deployment in IoT devices that come with pre-installed certificates from the factory. It allows for the design of lightweight X.509 profiles specifically tailored for devices with resource constraints. However, it still relies on a Public Key Infrastructure (PKI) and Certificate Authority (CA) framework, as indicated in reference [3]. The main advantage of LightCert4IoT is its effectiveness in environments with a high density of IoT devices. In addition, they suggested a method in which the IoT device already knows the knowledge of which enrollment CA to contact for verification. However, they also noted that denial-of-service attacks (DoS) are possible if the perpetrator floods a victim with erroneous requests. The attacker can also potentially access the device hardware and extract the private key. They recommended deploying additional secure hardware within the system to address these risks.

Marino et al. [9] proposed the PKIoT architecture to facilitate compact, certificate-based authentication for resource-constrained IoT devices. This architecture enables IoT nodes to delegate security-related tasks to an external server. However, a drawback of this solution is the requirement to establish a new server for node authentication.

Additionally, various works that propose centralized PKI solutions have a main weakness in being a single point of failure (references [10–12]). Other proposals (references [7,13]) are susceptible to denial-of-service (DoS) attacks, as well as the proposed work [14].

Adversaries can significantly limit the possible keyspace by using timestamps as the seeds for generating random keys. This makes it very efficient to guess keys through brute force. Additionally, protocols that reuse one-time pads (as referenced in [15]) create a major vulnerability. If opponents obtain two ciphers that have been XORed with the same one-time pad (key), they can retrieve the XOR result of the two plaintext messages. This allows them to perform statistical analyses to obtain both plaintexts individually.

### 3.2. Decentralized Technologies

Decentralized technologies involve the development of a Public Key Infrastructure (PKI), a system for creating, managing, and revoking digital certificates. In a decentralized PKI, trust and authentication are established through a network of interconnected nodes rather than relying on a single central authority. This distributed approach offers several advantages over traditional PKI systems. Firstly, it enhances security by reducing the risk of a single point of failure or compromise. Secondly, it promotes transparency by allowing all participants to see the network’s operations. Additionally, it prioritizes privacy by minimizing the collection and storage of sensitive user data in a central repository. Lastly, it facilitates interoperability by enabling different systems to work together seamlessly,

regardless of their underlying infrastructure. These benefits make decentralized PKI attractive for organizations seeking robust and reliable security solutions [16].

Won et al. (2012) [17] presented IoT-PKI, a decentralized Public Key Infrastructure (PKI) system operating on a blockchain network. This system uses distributed nodes to manage scalability effectively. Hoogland (2013) [18] also introduced DECKIN, a PKI solution that builds upon an existing blockchain protocol. DECKIN addresses key management issues by leveraging Physical Unclonable Functions (PUFs).

Singla and Bertino [19] proposed and analyzed three blockchain-based alternatives to the conventional Certificate Authority-Public Key Infrastructure (CA-PKI) for certificate administration. The first proposal involves utilizing the Emercoin blockchain, which provides Name Value Storage (NVS) for securely storing data. This approach allows for the decentralized and tamper-proof storage of certificate-related information. The second proposal focuses on using Ethereum smart contracts, which are self-executing contracts with the terms of the agreement directly written into code. This enables automated and trustless execution of certificate-related transactions. The third proposal employs the Ethereum Light Sync mode, which allows for the synchronization of the Ethereum blockchain without the need to host a full node. This approach reduces the resource requirements for certificate administration and enhances accessibility.

Mustafa Al-Bassam introduced SCPKI [20], a smart contract-based Public Key Infrastructure (PKI) and identity system, in his research. In a separate study, Magnusson [21] evaluated the performance of this PKI by utilizing smart contracts on the Ethereum blockchain and deploying it to a Raspberry Pi 2, an Internet of Things (IoT)-like device. The study found that implementing Public Key Infrastructure (PKI) on this IoT device required more than 20 GB of storage to support the blockchain. Additionally, the synchronization of the blockchain led to high CPU and RAM usage. Consequently, it was concluded that these technical challenges limit the practical feasibility of using SCPKI to replace current PKI solutions.

Our analysis shows that traditional PKI is deemed inadequate for the IoT landscape for two primary reasons.

- The IoT network comprises many devices that are vulnerable to attacks by malicious actors. These attacks can compromise the entire Public Key Infrastructure (PKI) system, as perpetrators can issue certificates for any device.
- The traditional method of assigning certificates is unsuitable for IoT devices because of their large numbers and resource constraints, which can negatively impact their performance.

Our proposal involves implementing a decentralized solution based on the Ethereum blockchain, utilizing lightweight, compact certificates and self-signed certificates for IoT applications. This approach goes beyond the traditional Certification Authorities (CAs). Blockchain's inherent characteristics, such as decentralization, transparency, and immutability, make it an ideal choice for securing IoT devices. In a blockchain-based PKI, digital certificates can be stored on a distributed ledger, eliminating the need for a central authority.

## 4. Background and Preliminaries

### 4.1. IoT Protocols

The Internet of Things (IoT) protocols allow the interconnection of countless devices, each susceptible to numerous security threats. Application protocols often include the Extensible Messaging and Presence Protocol (XMPP), Message Queuing Telemetry Transport (MQTT), and Constrained Application Protocol (CoAP).

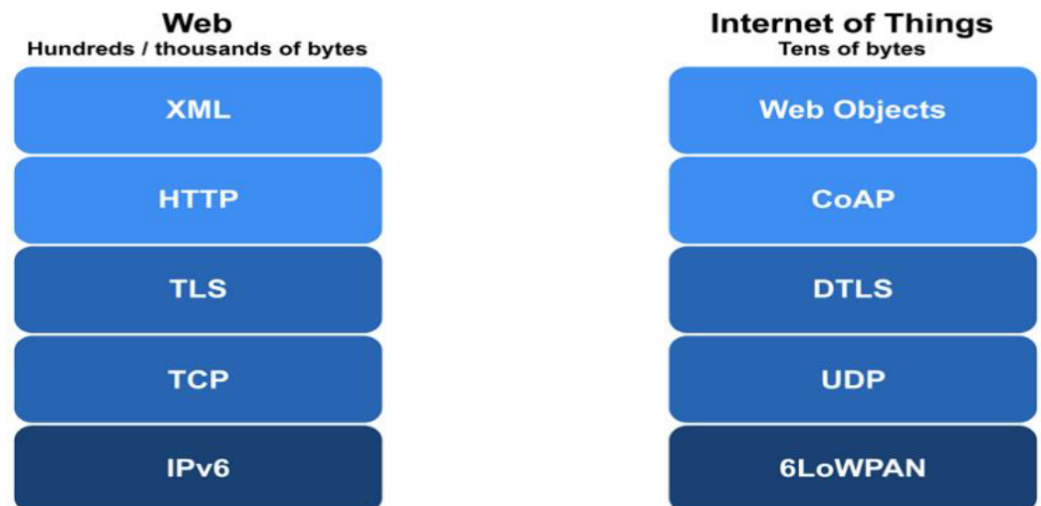
These devices are primarily resource-constrained and rely on communication protocols designed for efficiency. The Constrained Application Protocol (CoAP) is a lightweight protocol tailored for IoT devices.

IoT applications can be categorized based on their use cases, resulting in diversity:

- Consumer IoT: Examples include household equipment, voice assistants, and light fixtures.
- Commercial IoT is applied in industries such as transportation and healthcare, involving devices like smart pacemakers and monitoring systems.
- Military Things (IoMT): This refers to IoT technologies in the military sector, encompassing items such as surveillance robots and combat-oriented, wearable biometric devices.
- Industrial Internet of Things (IIoT): This application is employed in industrial settings such as manufacturing and energy sectors. It involves technologies such as digital control systems, smart agriculture solutions, and the management of industrial big data.
- Infrastructure IoT: This type supports connectivity in smart urban environments. Examples include infrastructure sensors and comprehensive management systems.

#### 4.2. CoAP Security and DTLS

The CoAP (Constrained Application Protocol) is specifically tailored to meet the requirements of HTTP-based IoT systems. Although HTTP serves as the fundamental protocol for data communication on the Web, it is unsuitable for IoT applications due to its heavy and power-intensive nature. CoAP overcomes this drawback by adapting the HTTP model for use in resource-constrained devices and network environments as shown in Figure 1. CoAP is characterized by low overhead and support for multicast, which is required for IoT microcontrollers and other devices with limited resources. It is commonly employed in applications related to smart energy and building automation [22].

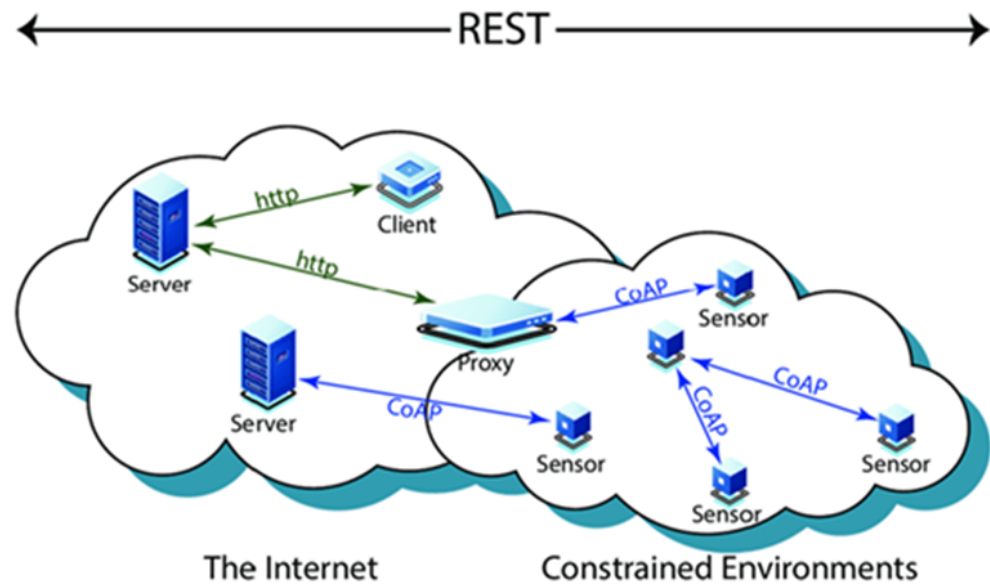


**Figure 1.** (Extracted from [23]). Performance analysis of end-to-end DTLS and IPsec-based communication in IoT environments.

Numerous IoT devices use the IEEE 802.15.4 standard [24], which defines the operation of low-rate wireless personal networks [23]. This standard functions on the physical and media access control layers of the 6LoWPAN network. TCP introduces more overhead and consumes more energy, which is not suitable in an IoT environment because IoT devices are mainly resource-constrained and embedded. Often, IoT devices are unable to run the full IP stack. As a result, the IETF developed a protocol known as the Constrained Application

Protocol (CoAP) to provide the necessary HTTP functionalities for IoT devices as shown in Figure 2.

DTLS and IPsec protocols provide security for CoAP communication real-time data running over the lightweight UDP protocol. CoAP accepts Representative State Transfer architecture (REST). This means that devices can use standard HTTP methods—GET, POST, PUT, and DELETE—to communicate, ensuring a simple and familiar route.



**Figure 2.** Enhancements and Challenges in CoAP—A Survey, Sensors 2020 [25].

CoAP (Constrained Application Protocol) security relies on the Datagram Transport Layer Security (DTLS) protocol to secure communication over UDP, making it suitable for IoT devices with limited resources. The Constrained Application Protocol (CoAP) standard does not include a specific authentication mechanism. Instead, it relies on Datagram Transport Layer Security (DTLS) to provide security at the protocol level. After provisioning, the standard defines four security modes for devices, as detailed in reference [26].

1. NoSec—No protocol-level security indicates that DTLS is disabled.
2. PreSharedKey—DTLS is enabled, and the device has a list of pre-shared keys, each specifying the nodes it can communicate with.
3. RawPublicKey—DTLS is enabled, and the device possesses an asymmetric key pair without a certificate. The key is validated using an out-of-band (OOB) mechanism.
4. Certificate—DTLS is enabled, and the device has an asymmetric key pair and an X.509 certificate signed by a common and trusted Root CA.

In this paper, we propose using the standard CoAP over DTLS protocol with LightCert4IoT to authenticate devices and establish session security. Our approach aims to enhance CoAP's security, decentralization, authentication, and information integrity mechanisms, all of which are integral to blockchain technology.

#### 4.3. Blockchain Overview

The concept of blockchain, a decentralized global ledger, was first introduced alongside the Bitcoin digital currency. Other platforms like Ethereum later emerged, introducing smart contracts. Blockchain consists of interconnected blocks containing transactions that occur at regular intervals. These transactions are arranged chronologically, leading to the current system state. Data immutability is achieved through cryptographic signatures and a decentralized, distributed consensus mechanism. This mechanism is supported

by a redundant and transparent peer-to-peer (P2P) data storage structure. Blockchain is an unalterable ledger where data and transactions remain impervious to modifications by external entities. The Ethereum blockchain uses classical cryptographic algorithms to ensure the security of transactions, including authentication, confidentiality, and integrity.

A notable innovation within the blockchain context is the concept of smart contracts. These are applications that operate over an underlying blockchain infrastructure, enabling parties to formulate contracts and subsequently execute outcomes within the peer-to-peer network. Ethereum is a decentralized platform that enables the execution of computer programs by nodes that distrust each other, eliminating the need for a central authority. We have moved with blockchain from Web 2.0 to Web 3.0; this transition marks a significant evolution in the architecture and functionality of the Internet. Web 2.0, characterized by user-generated content, social networking, and centralized platforms, revolutionized the way individuals interacted online, enabling widespread collaboration and communication. In contrast, Web 3.0 represents a paradigm shift towards decentralization, interoperability, and user sovereignty.

In the context of the Ethereum blockchain, limited Internet of Things (IoT) devices or applications connect to the broader blockchain network through node providers like Infura, QuickNode, and others. These providers give direct access to blockchain functions like checking transaction data, sending and receiving transactions, and running smart contracts. The node providers offer API endpoints that allow IoT devices to communicate with the Ethereum network without operating a full blockchain node. This reduces the resources and complexity required for limited devices. This setup enables IoT devices to use the blockchain's security, immutability, and decentralized nature while concentrating on their specific tasks and functions.

## 5. Proposed System

### 5.1. Assignment of LightCert4IoT to CoAP over DTLS

The CoAP over DTLS protocol uses certificates to authenticate devices and establish session security. The proposed solution suggests using the LightCert4IoT Certificate based on blockchain instead of the X509 certificate signed by a trusted root CA. The notations and components of the solution are summarised in Tables 1 and 2. This approach has the main advantages of being smaller, requiring less processor computation power, avoiding a single point of failure, and allowing simple certificate configuration and authentication for IoT devices. As detailed in [3], assigning LightCert4IoT to IoT devices involves three phases.

**Table 1.** Descriptions of the notations.

Configuration Component	Description
Verification Method	LRAs verify IoT devices with either a special Token given during device configuration or a Serial number (SN) assigned during production.
Serial Number (SN)	Each SN identifies hardware devices and undergoes verification through Electronic Notary (EN) or LRAs.
LRA Server	Upon successful verification, LRAs enable communication with blockchain and executing storage transactions for multiple constrained devices as a single transaction.
Direct blockchain access	Users can communicate directly with blockchain using the Light Ethereum Subprotocol (LES). However, this method may not be suitable for resource-limited machines due to higher computation and storage requirements.



**Table 2.** Components of the solution.

Notation	Description
LRA	Local Registration Authority.
HW Serial Number (SN)	Unique serial number assigned to the IoT device during production.
Edge Node/LRAs	LRA could be identified as part of the Edge node Server.
Ethereum	Blockchain network based on Smart contract.
LightCert smart contract	Certified software inside Ethereum, authenticating public keys and storing the IoTs' LightCert4IoT.
UUID	User identity, a unique identifier associated with each IoT device.

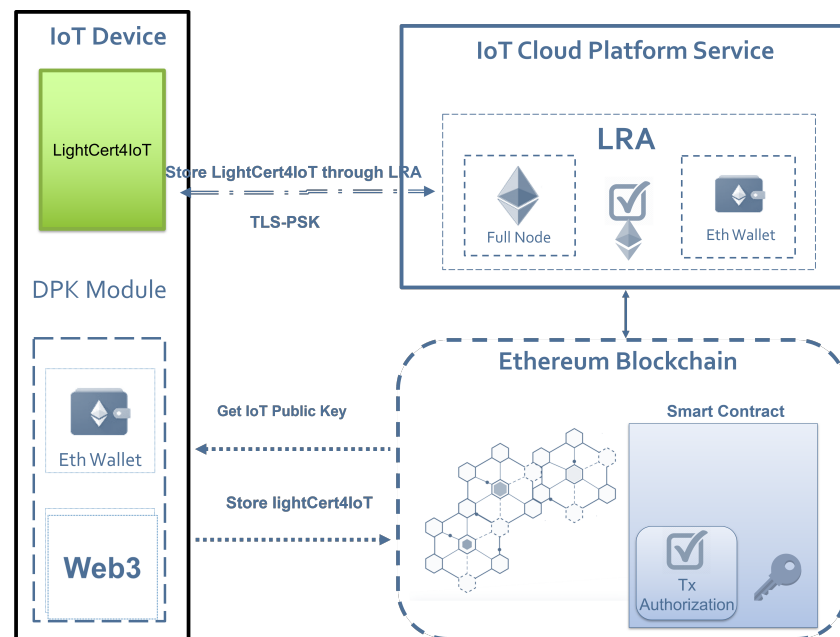
5.2. The Overall System Architecture

5.2.1. System Architecture Based on Ethereum

The solution runs on the DPK platform built on the Ethereum blockchain. It is a generic platform for applications that use Ethereum to store certificates, public keys, and other data. Paper [27] describes this platform well.

DPK is a generic platform built on Ethereum as shown in Figure 3. The main components are as follows:

1. Wallet Management Module (PKM).
2. Application Servers like LRA.
3. Client-Side Module: a plug-in software installed on the device that transparently creates an Ethereum address.
4. Smart Contract: deployed in the Ethereum Blockchain network, it contains the necessary functions to store and retrieve user data, such as the cryptographic public key.



**Figure 3.** Overall system architecture.

A universal plug-in DPK is installed on the user’s equipment. The DPK contains an Ether Wallet, Token, and the Web3 module to access the blockchain. The DPK module

approves its identifier with a Wallet Manager (PKM) (a Provider like Infura) and obtains cryptocurrency from the PKM.

PKM, which is part of the blockchain platform.

The APPs on the user equipment generate and transmit user public keys to the DPK module, which also generates its Ethereum public key. After a storage transaction requirement by the DPK module is approved by a blockchain, the DPK module sends the obtained user public keys to the blockchain so that the user's public keys are stored in the blockchain. The user's public keys are never stored outside the blockchain or in a third-party server. The stored user public keys are retrieved to the DPK module when necessary, such as when P2P communication is performed.

The application server is primarily responsible for deploying smart contracts associated with the application running on the device. One example of this application server is the LRA, which stores application certificates. Another application server function is storing a device's public key.

LRA: Network function located on the server side that authenticates the IoT user and approves the blockchain storage requests.

IoT Client generating the LightCert4IoT includes the DPK client module: Plug-in module software installed on the device that creates an Ethereum address and securely stores LightCert4IoT.

Smart contracts in the Ethereum Blockchain network containing the significant functions needed to store certificates and retrieve the keys.

### 5.2.2. Ethereum vs. Private Blockchains

We developed a solution for Let's Encrypt called B-DCV that enhances the automation and verification of the certificate and eliminates many trust and security issues of Let's Encrypt. The solution was implemented on the Ethereum Private Cloud Blockchain provided by Kaleido, which offers permissioned blockchain network variants of the core Ethereum, allowing authenticated and identifiable users to communicate and transact securely in a private setting. Public Ethereum requires a "gas price" for all transactions to prevent malicious behavior; however, in the scope of permissioned networks, the price can typically be set to zero regardless of the gas price.

Kaleido's private Ethereum blockchain features a fast consensus algorithm accelerating transaction processing.

- Gas prices are not an issue for our IoT solution. We store the certificate on the blockchain infrequently.
- Transaction speed is not an issue in our solution.

A potential solution could resemble B-DCV. This product would include a new software module on the server and IoT device through LRA that interfaces with the Ethereum blockchain through the Kaleido API, along with a new smart contract designed to store the public key of the domain's certificate and the IoT device.

Additionally, a software module in the web browser would read the domain's public key from the blockchain, enhancing and simplifying security in the DTLS protocol.

However, this approach would ultimately require the development of a new implementation paper, which we can address next.

### 5.3. Solution Details

#### 5.3.1. Phase 1—IoT Device Registration/Configuration

Before activating IoT protocols, end-users of IoT devices start by registering a unique token or Serial Number (SN) at the Local Registration Authority (LRA) server. This token/SN is used to identify and authenticate devices. Registration can be performed

manually or automatically through the HTTPS protocol between the configuration manager Node and the LRA. During configuration, the IoT device is assigned its identity with its token in the LRA. The SN is given to the IoT device as a unique global identity, and the token is randomly generated by the LRA. The SN is given to the IoT device as a unique global identity, and the LRA randomly generates the token. The Universal User Identity (UUID) refers to any device that does not have an addressable identity. The IoT device begins by registering its UUID, including its token or SN, in the LRA server and then generates a self-signed certificate.

### 5.3.2. Phase 2—Certificate Mechanism Creation and IoT Device Authentication

After the registration process, IoT devices create their self-signed certificates by generating the public and private keys. A Self-Signed Certificate (SSC) is an X.509 (or similar) certificate that is not signed by a trusted Certificate Authority but is instead signed with its private key. The verification process relies on a special token provided during IoT device configuration. In the case of a Serial Number (SN) that is assigned during production and configured into the LRA, each SN is allocated to identify each hardware device and is verified during configuration. The process of validating and matching the self-signed certificate with the UUID linked to the token involves a protocol executed between the IoT device and LRA. This protocol can be carried out through a secure CoAP, which can be achieved using the DTLS Pre-Shared Key between the IoT device and LRA. The device possesses a list of pre-shared keys with each LRA node, including a list of the nodes with which it can communicate using Lightweight DTLS Authentication. The LightCert4IoT certificate could be utilized by all IoT protocols over DTLS.

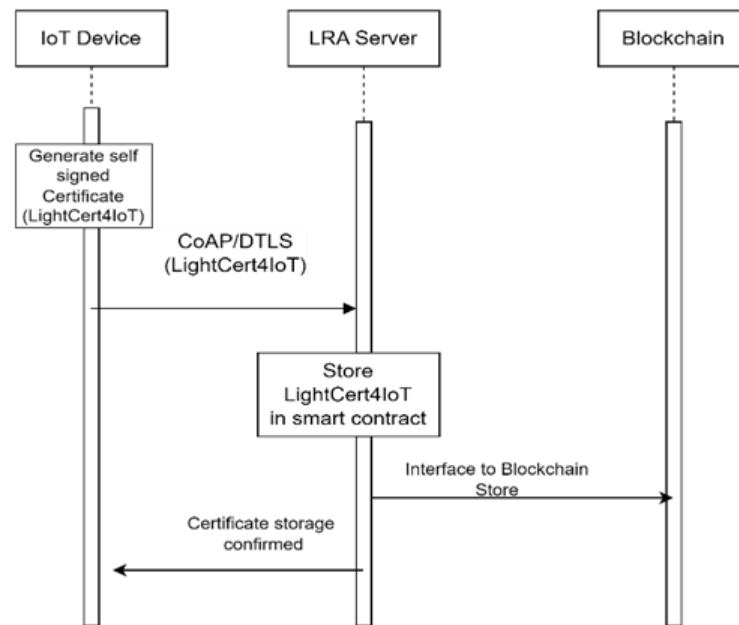
### 5.3.3. Phase 3—Storage of the Certificate in Blockchain

The LRA servers, which can function as full nodes for the blockchain, store certificates on the blockchain. After certificate authentication, the LRA server performs certificate storage transactions on the blockchain and seamlessly integrates with the blockchain network. The platform is primarily based on an Ethereum smart contract. It stores the LightCert4IoT and acts as the central repository for IoT public keys and the associated device data for each identity. The LRA server may also function as a complete Ethereum node, performing tasks such as mining, storage, and validation within the Ethereum blockchain. The LightCert smart contract is a certifier within the Ethereum blockchain. It first authenticates the users' public keys submitted by the LRA and then maps them with their corresponding user identities. The LightCert4IoT certificate format primarily comprises the following data: UUID (User Identity) as the device's public key, LRA domain name or IP public key, and certificate signature, refer to the sequence diagram in Figure 4.

LightCert4IoT fields are based on the X509 certificate format with fewer fields and smaller contents. LightCert4IoT is a self-signed certificate where the end user's device UUID has been verified by an LRA instead of using CA, as other traditional certificates. The features of LightCert4IoT are listed below:

- LightCert4IoT is based on the X509 certificate format, but some of the fields are smaller.
- LightCert4IoT is approved and verified by LRA rather than using a CA.
- The IoT device creates a self-signed certificate.
- The LRA server acts as the intermediary between the IoT device's tokens and its UUID.
- The LRA server might function as a full Ethereum node, performing tasks like mining, storage, and validation in the Ethereum blockchain.
- The LRA server serves as a node provider for the Ethereum blockchain.
- The LRA server could also act as an Ether provider.
- LRAs validate the public key certificate.

- LRA stores the certificate directly in the blockchain through a single transaction.
- The transaction enables the storage of multiple IoT device certificates.



**Figure 4.** IoT device certificate generation and storage in blockchain.

#### 5.4. Modified Handshake Protocol in DTLS for IoT and Application Server Authentication

While DTLS/CA in Figure 5 has the advantages of authenticating the server and ensuring confidentiality between the communicating parties, it also has some drawbacks. Firstly, the client is often not authenticated because it is inconvenient to configure the client to use a certificate, especially for non-technical users. Additionally, it is impractical to assign and provision millions of certificates in the IoT case. Secondly, there is a risk of having a rogue CA or subverting a CA, as evidenced by the case of DigiNotar, which was a well-established and reputable certificate authority [2].

The blockchain-based certificate eliminates the centralized PKI/CA and simplifies the DTLS/TLS handshake protocol by verifying the certificate on both the client and server side through access to the blockchain. In our discussion about the server-side certificate, we refer to papers [4,28], where the authors proposed using blockchain technology to decentralize the Automated Certificate Management Environment (ACME) protocol. This would eliminate the need for a trusted CA. The authors implemented and tested this system on the Ethereum Blockchain.

The IoT device authentication with IoT Application servers is secured by verifying the IoT device public keys stored in the blockchain with the one sent by the IoT client in DTLS. We propose the following change:

- Once the public key is received in the Application server, it interacts with the blockchain and retrieves the stored IoT device public key.
- It compares the IoT public key sent to the Application server with the IoT public key retrieved from the blockchain.
- If the verification's result is positive, the public keys are exchanged between the application server and IoT device; then, the session keys are defined and the secure connection is established.

The LRA node that interacts with the blockchain and where UUID addresses IoT devices.

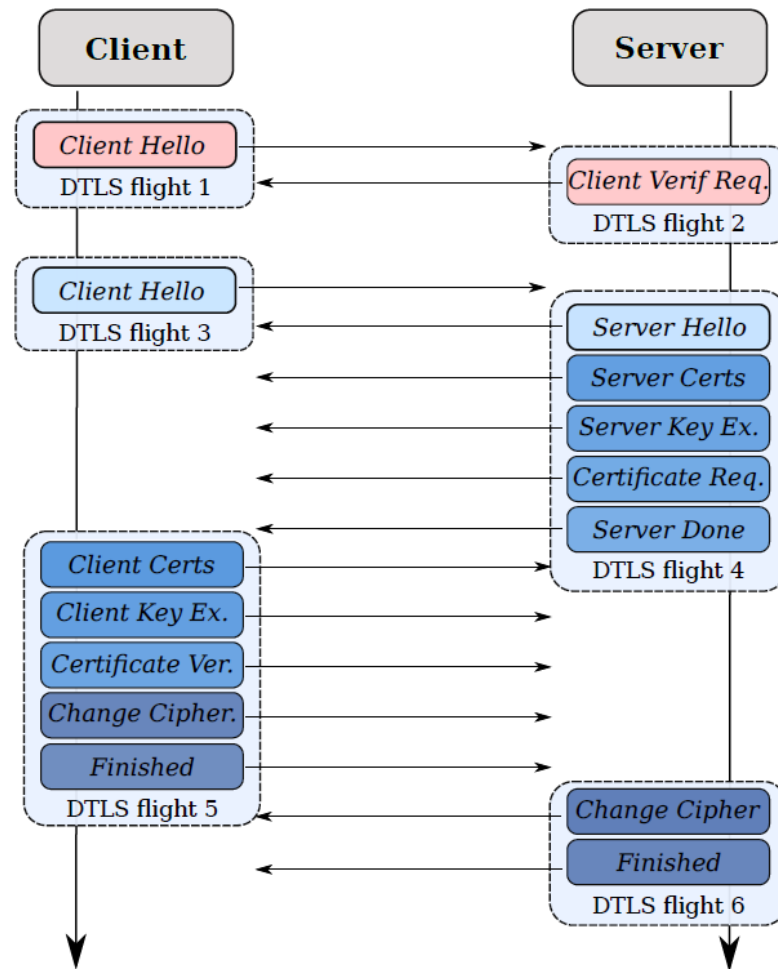
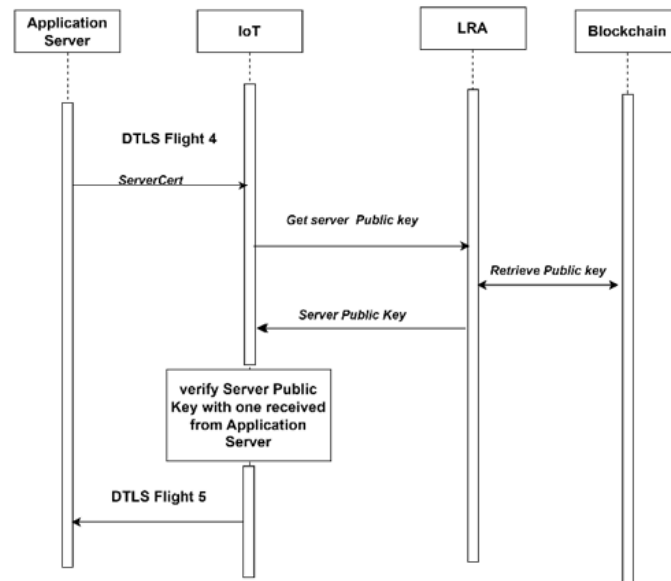


Figure 5. (D)TLS handshake protocol extracted from [29].

On the server side, we propose a similar mechanism to verify the server's certificate by the IoT device interfacing with the blockchain (Figure 6). This modification's main advantage is that it simplifies the certificate verification on both sides by relying on a blockchain without the complication of the signed certificate and certificate chain. We added a new security mode for DTLS: Certificate—LightCert4IoT. DTLS is enabled, and the device has a self-signed certificate stored in blockchain. In this mode, the DTLS handshake protocol can be simplified in phases 4 and 5, resulting in the generation of the session key being simpler, faster, and more secure.

#### 5.4.1. Handshake Flow in DTLS

In a complete DTLS handshake, there are up to six message flights, each containing multiple messages. Flights 3, 4, and 5 handle negotiating security parameters, peer authentication, and session key establishment. If certificate-based authentication is used, certificates are exchanged within flights 4 and 5. The finished messages in flights 5 and 6 complete the handshake and verify its correctness. The handshake protocol's main function is to negotiate a session. This session includes a session identifier, an optional peer certificate (X.509v3), a compression method, a cipher suite, a master secret, and a Boolean value indicating if the session is resumable. The handshake occurs in different phases, as illustrated in the message flow in Figure 5. The client and the server begin by exchanging hello messages (ClientHello and ServerHello) to agree on a compression method and cryptographic algorithms. These messages also contain randomness and indicate support for session resumption [29].



**Figure 6.** New method in DTLS for application server authentication.

#### 5.4.2. Proposed Modification

##### Verification Certificates of the IoT Application Server Using Blockchain

The session starts with CoAP over DTLS between the IoT device and the application server. During in-flight 4, the server sends a *ServerCertificate* message containing a list of X.509v3 certificates from a root Certificate Authority or a self-signed certificate, as described in [29], and then notifies the client with the *ServerDone* message.

The modification involves the server certificate verification process. Instead of sending a list of X.509v3 certificates from a root CA, the server certificate's validity is checked on the blockchain, following the details provided in the paper [4,28]. Figure 5 illustrates the sequence of messages exchanged between the client and the server for flight 4.

However, in the case of IoT devices, the device verifies the server certificate instead of following this traditional approach by checking the blockchain through the LRA. In this process, the IoT device compares the public key stored in the blockchain with the certificate's public key received from the Application server.

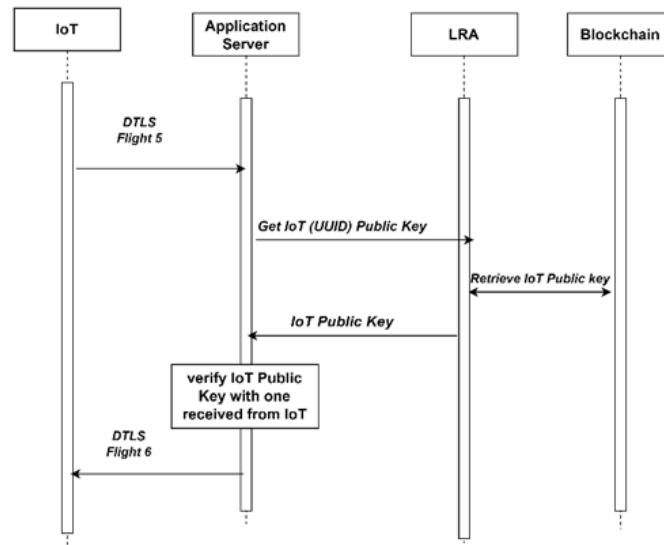
We must note that we use the LRA containing a Web3 module to interface with the blockchain, decreasing the load on IoT devices.

As an alternative, the application server in Figure 6 can directly interface with the blockchain to obtain the IoT public key.

##### Verification Certificate of the IoT Device

When the server needs to perform mutual authentication, it asks for a client certificate using a *CertificateRequest* message in flight 4. The client then sends its certificate chain in response. Instead, we propose that the IoT device sends the *LightCert4IoT*, through *ClientCerts*, to the application server, which extracts the LRA IP address, UUID, and IoT public key from the certificate. The modification proposed in flight 5 involves the server verifying the authenticity of the IoT device (client) by consulting the LRA. The LRA retrieves the IoT device public key from the blockchain, which the server then compares with the one sent by the client upon receiving the *CertificateVerify* message. If the IoT device public key is verified, the application server terminates the handshake protocol with DTLS flight 6. If the verification's result is positive, the public keys are exchanged between the application server and the IoT device, the session keys are defined, and the

secure connection is established. If not, the handshake protocol is halted. This new method in Figure 7 offers the advantage of reducing processing time during the handshake.



**Figure 7.** New method in DTLS for IoT authentication with client-side application servers.

After completing the verification process, the IoT device proceeds with the ClientKeyExchange and CertificateVerify messages to indicate the installation of the newly derived session key. It must then send the Finished message before the server’s final response. This Finished message from the client includes the shared master secret, a string, and a hash of the entire handshake, all encrypted using the negotiated encryption algorithm and the new session key.

The server then responds with its ChangeCipherSpec message and its Finished message. These two messages provide the server with the client’s key share and a signature calculated with the client’s private key over the message transcript. As a result, the server can verify the client’s authenticity and derive the shared master secret.

### Session Key

The process begins with exchanging cryptographic parameters between peers to establish a shared master secret, which is essential for secure communication. The peers may utilize ECDHE-ECDSA as a key exchange method depending on the authentication method and cipher suite. It is important to note that TLS and DTLS also support pre-shared key (PSK) and raw public key (RPK) authentication methods.

Once the server certificate is verified and the client and server have agreed on an ephemeral key exchange mechanism, the server sends a ServerKeyExchange message. This message is crucial for initiating the key establishment process.

The most significant processing times are observed during the negotiation of session keys. In the case of DTLS, the cipher suite ECDHE-ECDSA-WITH-AES-128-CCM-8 is commonly utilized. Notably, the smallest curve for this suite is NIST P-256.

### Interface to Blockchain

We propose a robust and secure DTLS handshake protocol using the blockchain to verify the public key of LightCert4IoT. A new interface is introduced between the application server and LRA; GET IoT Public key from LRA by addressing the LRA by its IP and the IoT device by its UUID. LRA reads the respective IoT public key through the blockchain interface. When the IoT device public keys are verified, the application

server terminates the handshake protocol by the DTLS flight 6. For the certificate server verification, refer to [22,23].

### 5.5. Testing and Evaluation

#### 5.5.1. Ethereum Blockchain

We have run the evaluation over the Ethereum Sepolia test network in Figure 8, which utilizes the Proof of Stake consensus algorithm. The smart contract stores the mapping between the UUID of the IoT device and the device structure [30]. The device structure contains the following fields:

```

struct device {
    address certOwner; // host of the certifiact (its owner)
    string publicKey; // public key of the device
    string LRADomain;
    string LRAIpAddress; // ip address of the LRA
    uint expiryDate; // unix format epoch time
    bool isValid;
    bool registered;
}

```

Figure 8. Smart contract coding.

The deployment of the smart contract on the Sepolia test costs 0.00180161 ETH; Figure 9 shows the details:

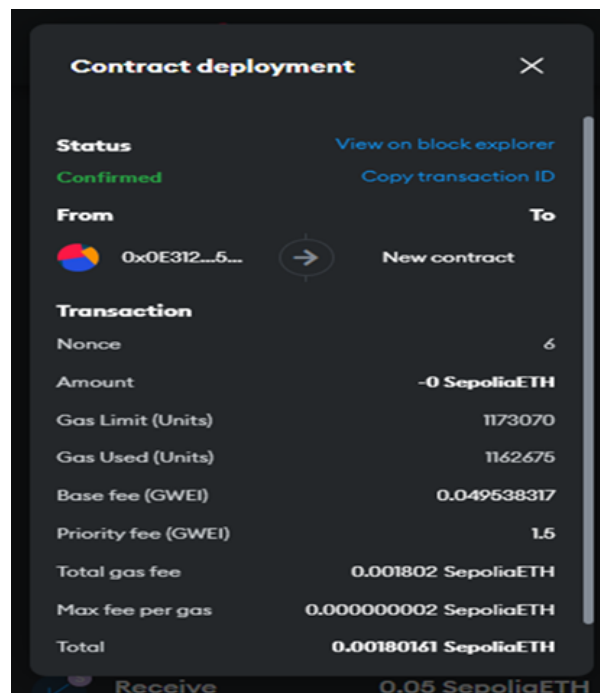


Figure 9. Smart contract deployment.

The smart contract was compiled using REMIX IDE and the wallet used was MetaMask. The Deployment time average is 30 s. Upon deployment on the test network, a mapping is created with test data for testing purposes, which affects the cost of the deployment without the default data deployment cost: 0.00153345 Sepolia ETH [30].

The function used to register the test device inserts into the map a device structure holding the following test values for a test key of value “123”:

- Address: (current MetaMask wallet address of the deployment);
- publicKey: pki;



- LRADomain: lradomain.com;
- LRAIpAddress: ip;
- expiryDate: block.timestamp+block.timestamp;
- isValid: true;
- registered: true.

Retrieving these data from the Ethereum network is free (not payable transaction) as the mapping is public. Getting this information is a near-instant operation (5 ms maximum) and returns the following when debugging using REMIX.

The Ethereum blockchain test results [5] confirm that the modified DTLS handshake protocol is feasible. This is achieved by obtaining the public key from the Ethereum network through LRA. The blockchain side has minimal impact on the DTLS handshake regarding time and cost. Still, establishing a secure DTLS handshake without the signed certificate and certificate chain complication is an overall time benefit.

We examine the overhead related to the PKI certificate-based DTLS handshake, which encompasses the mutual authentication of the communicating peers. This evaluation considers both transmission overhead and processing time. Furthermore, we emphasize the enhancements achieved through our proposed modification to DTLS, which incorporates blockchain technology for certificate verification. This assessment is grounded in the testing results detailed in ref. [2].

#### 5.5.2. Gain in Processing Times

CoAP specifies the cipher suite TLS-ECDHE-ECDSA-WITH-AES-128-CCM-8 as the mandatory option for certificate-based authentication. The minimum curve required for this suite is NIST P-256 [31]. The signature verification and the Diffie–Hellman key agreement are the primary cryptographic operations affecting the processing time of the DTLS handshake. We extracted the results from [2], where they analyzed the processing time for ECDSA and ECDH operations using the P-256 curve by conducting 100 independent ECDSA signature generations and verifications and ECDH key–pair generations and agreements. The results are presented below:

- Single-signature certificate verification requires about 6 s. This overhead grows linearly with the length of the certificate chain.
- Authenticated ephemeral Diffie–Hellman key exchange. Here, generating the Diffie–Hellman key pair requires 2.1 s, whereas 2.5 s is needed to sign the public value. The derivation of a shared secret requires 4.3 s.

The cryptographic per-handshake overheads add up to 15 s for a certificate-based DTLS handshake, even with minimal certificate chains of length 2.

A certificate chain is an ordered list of certificates containing Certificate Authority (CA) certificates that enable the receiver to verify that the sender and all CAs are trustworthy. The chain begins with the SSL/TLS certificate, and each certificate in the chain is signed by the entity identified by the following certificate in the chain.

While these overheads may be acceptable during sporadic configuration phases of a smart object, such delays are likely to render certificate-based security solutions inapplicable if performed on a per-connection basis. With the blockchain-based certificate, we eliminate at least 6s of the signature certificate.

#### 5.5.3. Transmission Overheads

A complete certificate-based DTLS (Datagram Transport Layer Security) handshake comprises up to 15 messages. These messages may need to be fragmented at the DTLS layer or below. For example, even short certificates can exceed 220 bytes in size, which often surpasses the limited frame size of link layers like IEEE 802.15.4, which is 127 bytes.

This fragmentation can lead to an increased total number of transmitted packets. If a single packet is lost, it necessitates retransmitting the entire set of messages, further increasing the DTLS network overhead [32,33].

Additionally, the receiving device's packet buffer must be large enough to store a complete DTLS message. This requirement can present challenges for devices with limited RAM, potentially hindering the successful reception of large certificates or lengthy certificate chains.

Moreover, extra network overhead arises from the need to use time synchronization protocols like NTP (Network Time Protocol) and certificate status protocols such as the Online Certificate Status Protocol (OCSP). These protocols are essential for verifying certificate validity periods and revocation statuses. With the LightCert4IoT certificate, it will never exceed 220 bytes, eliminating the issue.

IoT devices can update or revoke certificates using a smart contract on the Ethereum Blockchain. However, before any changes can occur, they must first gain approval from the LRA. This process guarantees a controlled and secure system for managing certificates. In general, revocation is not applicable in the case of self-certificate creation.

The performance evaluation of LightCert4IoT was carried out using the Cooja–Contiki network simulation [34]. The results indicate that the energy consumption for computing the LightCert4IoT certificate is lower compared to the conventional X509 certificate. The memory size required is approximately 200 bytes, significantly less than the X509 standard and IoT profile. LightCert4IoT has contributed to improving the memory size and power consumption of CoAP over DTLS.

## 6. Performance Analysis

Table 3 summarizes the major performance in the text document and from previous papers.

**Table 3.** Performance analysis.

Metric	Based on X509 PKI/CA	LightCert4IoT Blockchain
Energy consumption	1.15 mW	0.86 mW
Memory size	1500 bytes	<200 bytes
Configuration and distribution of Certificates	Extensive time	Simple, self-signed
Revocation	Extensive time	Not applicable
CoAP/DTLS Handshake		
Gain in processing times	12 s	< 6 s
Transmission Overheads	Packet fragmentation can lead to an increased total number of transmitted packets due to certificates larger than 200 bytes	No packet fragmentation
Interface to Ethereum Retrieving Data: Public key	Not applicable	<5 ms

### *DTLS Certificate Based on PKI Versus Blockchain*

1. LightCert4IoT is tailored for the DTLS/CoAP protocol in IoT devices that consume less power (see reference [34]). The current solution, which uses the DTLS protocol with the PreSharedKey and Certificate based on X509, has been successful in traditional settings. However, it may not be suitable for IoT devices due to their limited resources.

2. The existing assignment of certificates is not always suitable for the Internet of Things (IoT). PKI/CA requires an extensive infrastructure to manage certificates, and there is a risk of having a rogue CA or subverting a CA and a single point of failure. Our solution is based on blockchain and operates on the Ethereum Solidity smart contract platform and LRA, eliminating the dependency on a trusted PKI/CA, given the following features and advantages:

- LightCert4IoT end users can generate self-signed certificates using their signatures.
- IoT devices are authenticated by local registration authorities (LRAs) or Edge nodes and securely stored on the Ethereum blockchain.
- Simple configuration and distribution certificates for numerous IoT devices. Other solutions, like hard-coding the credentials on IoT devices, introduce the threat of reverse engineering and make it difficult to ensure privacy due to fixed IDs.

3. A new modified handshake protocol in DTLS based on blockchain for verifying IoT and server certificates and establishing the session key with application servers. The existing Verifying certificates assigned to IoT and servers during the handshake protocol consume energy and increase the delay in establishing a session, as it relies on public key cryptography. The cryptographic per-handshake overheads add up to 15 s for a certificate-based DTLS handshake.

4. LightCert4IoT solves the issue of fragmented messages at the DTLS layer. A complete certificate-based DTLS (Datagram Transport Layer Security) handshake comprises up to 15 messages. This fragmentation can lead to an increased total number of transmitted packets.

5. Blockchain's impressive ability to timestamp and protect information significantly improves data integrity and auditing in IoT applications, especially in crucial areas such as healthcare and industrial IoT.

Scalability of the system is established by combining blockchain and distributed Edge nodes.

By incorporating blockchain into CoAP/DTLS, we eliminate reliance on a central authority, mitigating the risk of a single failure and enhancing the overall security robustness of IoT networks.

## 7. Summary of the Security Analysis

Here, we present a summary of the security analysis and guidelines for the proposed framework. It covers potential vulnerabilities and ways to access data from the blockchain, particularly in terms of the light receivers for IoT devices. The increasing use of billions of IoT devices introduces new types of redirected attacks, including human-generated DDoS attacks, where IoT nodes can be used as attackers.

The following are the issues present in the current solution and configuration for IoT networks and the advantages of our approach from a security perspective:

1. IoT networks consist of many IoT devices; so, we have implemented automated configuration and self-signed certificate issuance.
2. The Public Key Infrastructure (PKI) needs to be distributed and scaled to avoid a high load on the Certificate Authorities (CAs) and to prevent a single point of failure. We addressed this by implementing the PKI in blockchain and distributing the Local Registration Authority (LRA) to solve scalability and single point of failure issues.
3. The DTLS/CoAP is simplified, and handshake time processing is reduced as certificate verification on both sides (IoT device and server) relies on a blockchain without the signed certificate and certificate chain complication.

4. We have reduced the size of certificates and implemented lightweight cryptographic algorithms, such as elliptic curve cryptography (ECC), optimized for low-power and resource-constrained devices.

5. Each device in the network can manage its own certificate instead of relying on a central CA. This eliminates the system's central point of failure, making it more scalable and secure.

6. LightCert4IoT can be considered as Device-Embedded Digital certificates (DEDs) and can be used to authenticate IoT devices to establish secure connections. The certificate can be run on hardware security modules (HSMs).

This solution is applicable for sensor networks where energy efficiency is crucial due to the finite source of energy for sensor nodes. Some solutions have been proposed to optimize energy usage. After analyzing the current issues of IoT security, the LightCert4IoT over CoAP or other constrained IoT protocols fulfills the recommendations outlined above. The certificate is smaller in size, self-signed, stored, and verified with the distributed PKI built on the blockchain.

For further discussion on scalability and performance in large-scale networks and how the system would operate in such an environment, you can refer to the previously cited papers referenced in [3,34].

## 8. Conclusions and Future Work

In our paper, we have made two key improvements in the IoT domain:

Firstly, we have recently implemented a groundbreaking approach known as LightCert4IoT to issue certificates specifically tailored for the DTLS/CoAP protocol used in IoT devices. This innovative method operates on the Ethereum solidity smart contract platform, eliminating dependency on a trusted PKI/CA. With LightCert4IoT, end users can generate certificates using their own signatures, which are subsequently authenticated by local registration authorities (LRAs) or Edge nodes and securely stored in the Ethereum blockchain. By adopting this approach, we have successfully overcome the challenges commonly associated with client-side certificate solutions within the broader IoT infrastructure. LightCert4IoT not only meets the unique requirements of the IoT environment but also delivers the advantages of authentication and signatures facilitated through LRAs on blockchain networks, all while maintaining a compact size.

Secondly, we have proposed a modified handshake protocol in DTLS based on blockchain for client IoT device authentication and establishing the session key with Application servers.

Blockchain's impressive ability to timestamp and protect information greatly improves data integrity and auditing in IoT applications, especially in crucial areas such as healthcare and industrial IoT. In conclusion, our proposed system takes advantage of blockchain technology to enhance the security limitations of CoAP. This results in cutting-edge authentication, a secure method for device registration, and highly robust data integrity, ensuring efficient and scalable security for IoT. By incorporating blockchain into CoAP/DTLS, we aim to eliminate reliance on a central authority, mitigating the risk of a single failure and enhancing the overall security robustness of IoT networks. LightCert4IoT benefits IoT environments with a huge number of constrained devices by reducing energy consumption and memory size, and by simplifying the DTLS protocol with secure authentication based on blockchain. An overall testing lab or simulation should follow the solution. This document aims to present the theoretical solution with data and results from previous studies and estimate its benefits.

The future work entails presenting the solution to the standardization body, particularly focusing on the modified DTLS protocol for the IoT device domain. Additionally, an

investigation into the generalization of this approach for all IoT-constrained protocols will be conducted.

**Author Contributions:** Conceptualization, J.S.; Methodology, P.B. and K.D.; Software, P.B. and J.M.; Validation, D.K. and H.H.; Resources, S.H.; Writing—original draft, D.K.; Writing—review & editing, S.H., P.S. and J.S.; Supervision, S.H., P.S. and J.S. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Raza, S.; Shafagh, H.; Hewage, K.; Hummen, R.; Voigt, T. Lithe: Lightweight Secure CoAP for the Internet of Things. *IEEE Sens. J.* **2013**, *13*, 10. [CrossRef]
2. Hummen, R.; Ziegeldorf, J.H.; Shafagh, H.; Raza, S.; Wehrle, K. Towards Viable Certificate-based Authentication for the Internet of Things. In Proceedings of the Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks, Budapest, Hungary, 19 April 2013. [CrossRef]
3. Garba, A.; Khoury, D.; Balian, P.; Haddad, S.; Sayah, J.; Chen, Z.; Guan, Z.; Hamdan, H.; Charafeddine, J.; Al-Mutib, K. LightCert4IoTs: Blockchain-Based Lightweight Certificates Authentication for IoT Applications. *IEEE Access* **2023**, *11*, 28370–28383. [CrossRef]
4. Khoury, D.; Balian, P.; Kfoury, E. Implementation of blockchain domain control verification (B-DCV). In Proceedings of the 2022 45th International Conference on Telecommunications and Signal Processing (TSP), Prague, Czech Republic, 13–15 July 2022; pp. 17–22. [CrossRef]
5. Wood, G. Ethereum Yellow Paper: A Secure Decentralized Generalized Transaction Ledger. Berlin Version Beacfbf. October 2022. Available online: <https://membres-ljk.imag.fr/Jean-Guillaume.Dumas/Enseignements/ProjetsCrypto/Ethereum/ethereum-yellowpaper.pdf> (accessed on 27 December 2024).
6. El-Hajj, M.; Beune, P. Lightweight public key infrastructure for the Internet of Things: A systematic literature review. *J. Ind. Inf. Integr.* **2024**, *41*, 100670. [CrossRef]
7. Höglund, J.; Lindemer, S.; Furuheid, M.; Raza, S. PKI4IoT: Towards a public key infrastructure for the Internet of Things. *Comput. Secur.* **2020**, *89*, 101658. [CrossRef]
8. Forsby, F.; Furuheid, M.; Papadimitratos, P.; Raza, S. Lightweight X.509 digital certificates for the Internet of Things. In *Interoperability, Safety, and Security in IoT*; Springer: Berlin, Germany, 2017; pp. 123–133.
9. Marino, F.; Moiso, C.; Petracca, M. PKIoT: A public key infrastructure for the Internet of Things. *Trans. Emerg. Telecommun. Technol.* **2019**, *30*, e3681. [CrossRef]
10. Chanda, S.; Luhach, A.K.; Alnumay, W.; Sengupta, I.; Roy, D.S. A lightweight device-level Public Key Infrastructure with DRAM-based Physical Unclonable Function (PUF) for secure cyber-physical systems. *Comput. Commun.* **2022**, *190*, 87–98. [CrossRef]
11. Kadri, B.; Feham, M.; Mohamed, A. Lightweight PKI for WSN  $\mu$ PKI. *J. Secur. Commun. Netw.* **2010**, *10*, 135–141.
12. Toorani, M.; Beheshti, A. LPKI-a lightweight public key infrastructure for mobile environments. In Proceedings of the 2008 11th IEEE Singapore International Conference on Communication Systems, Guangzhou, China, 19–21 November 2008; IEEE: Piscataway, NJ, USA, 2008; pp. 162–166.
13. Champagne, L. Replacing Public Key Infrastructures (PKI) by Blockchain IoT Devices Security Management. 2021. Available online: <http://hdl.handle.net/2268.2/11608> (accessed on 25 June 2021).
14. Henriques, M.S.; Vernekar, N.K. Using symmetric and asymmetric cryptography to secure communication between devices in IoT. In Proceedings of the 2017 International Conference on IoT and Application (ICIOT), Nagapattinam, India, 19–20 May 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–4.
15. Shah, D.P.; Shah, P.G. Revisiting of elliptical curve cryptography for securing the Internet of Things (IoT). In Proceedings of the 2018 Advances in Science and Engineering Technology International Conferences (ASET), Abu Dhabi, United Arab Emirates, 6 February–5 April 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–3.
16. Zhang, Y.; Sun, Z. An IoT security framework based on the blockchain and elliptic curve cryptography. *Future Gener. Comput. Syst.* **2019**, *96*, 518–527.

17. Won, J.; Singla, A.; Bertino, E.; Bollella, G. Decentralized public key infrastructure for the Internet. In Proceedings of the MILCOM 2018-2018 IEEE Military Communications Conference (MILCOM), Los Angeles, CA, USA, 29–31 October 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 907–913.
18. Hoogland, M. *A Distributed Public Key Infrastructure for the IoT*; Delft University of Technology: Delft, The Netherlands, 2018.
19. Singla, A.; Bertino, E. Blockchain-based PKI solutions for IoT. In Proceedings of the 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC), Philadelphia, PA, USA, 18–20 October 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 9–15.
20. Al-Bassam, M. SCPKI: A smart contract-based PKI and identity system. In Proceedings of the ACM Workshop on Blockchain Cryptocurrencies and Contracts, Abu Dhabi, United Arab Emirates, 2 April 2017; pp. 35–40.
21. Magnusson, S. *Evaluation of Decentralized Alternatives to PKI for IoT Devices: A Literature Study and Proof of Concept Implementation to Explore the Viability of Replacing PKI with Decentralized Alternatives*; Linköping University: Linköping, Sweden, 2018.
22. Shelby, Z.; Hartke, K.; Bormann, C. The Constrained Application Protocol (CoAP). RFC 7252. Internet Engineering Task Force (IETF). 2014. Available online: <https://datatracker.ietf.org/doc/html/rfc7252> (accessed on 27 December 2024).
23. Vignesh, K. Performance Analysis of End-to-End DTLS and IPsec-Based Communication in IoT Security and Privacy Distributed Systems Security. Master's Thesis, Faculty of Computing Blekinge Institute of Technology, Karlskrona, Sweden, 2017.
24. Howitt, I.; Gutierrez, J.A. IEEE 802.15.4 low rate—Wireless personal area network coexistence issues. In Proceedings of the 2003 IEEE Wireless Communications and Networking, WCNC 2003, New Orleans, LA, USA, 16–20 March 2003. [CrossRef]
25. Tariq, M.A.; Khan, M.; Khan, M.T.R.; Kim, D. Enhancements and Challenges in CoAP—A Survey. *Sensors* **2020**, *20*, 6391. [CrossRef] [PubMed]
26. Tschofenig, H.; Fossati, T. Transport Layer Security (TLS)/Datagram Transport Layer Security (DTLS) Profiles for the Internet of Things. Technical Report. RFC 7925. Internet Engineering Task Force (IETF). 2016. Available online: <https://datatracker.ietf.org/doc/html/rfc7925> (accessed on 27 December 2024).
27. Kfoury, E.; Khoury, D. Distributed public key infrastructure and PSK exchange based on blockchain technology. In Proceedings of the 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Halifax, NS, Canada, 30 July–3 August 2018. [CrossRef]
28. Barnes, R.; Hoffman-Andrews, J.; McCarney, D.; Kasten, J.; Automatic Certificate Management Environment (ACME). RFC 8555, RFC Editor. March 2019. Available online: <https://datatracker.ietf.org/doc/html/rfc8555> (accessed on 27 December 2024).
29. Claeys, T.; Vucinic, M.; Watteyne, T.; Rousseau, F.; Tourancheau, B. Performance of the Transport Layer Security Handshake Over 6TiSCH. *Sensors* **2021**, *21*, 2192. [CrossRef] [PubMed]
30. Buterin, V. Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform. 2017. Available online: <https://github.com/ethereum/wiki/wiki/White-Paper> (accessed on 27 December 2024).
31. McGrew, D.; Bailey, D.; Campagna, M.; Dugal, R. AES-CCM Elliptic Curve Cryptography (ECC) Cipher Suites for TLS. RFC 7251. Internet Engineering Task Force (IETF). 2014. Available online: <https://www.rfc-editor.org/rfc/rfc7251.html> (accessed on 27 December 2024).
32. Gupta, V.; Wurm, M.; Zhu, Y.; Millard, M.; Fung, S.; Gura, N.; Eberle, H.; Chang Shantz, S. Sizzle: A standards-based end-to-end security architecture for the embedded Internet. *Pervasive Mob. Comput.* **2005**, *1*, 425–445. [CrossRef]
33. Hummen, R.; Hiller, J.; Wirtz, H.; Henze, M.; Shafagh, H.; Wehrle, K. 6LoWPAN Fragmentation Attacks and Mitigation Mechanisms. In Proceedings of the ACM WiSec, Budapest, Hungary, 17–19 April 2013.
34. Khoury, D.; Haddad, S.; Sondi, P.; Abou Haidar, G.; Semaan, D.; Sayah, J. Performance Evaluation and Analysis of LightCert4IoT Using Cooja-Contiki Simulator. *IEEE Access* **2024**, *12*, 122350–122362. [CrossRef]

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.