

GOLPMA: GROUP ORIENTED LIGHTWEIGHT PAYLOAD-BASED MUTUAL AUTHENTICATION FOR SECURED IoT COMMUNICATION

P.S. Akshatha*, Harshith D. and S.M. Dilip Kumar

Received: April 21, 2023; Revised: August 06, 2023; Accepted: August 11, 2023

Abstract

The rise of the Internet of Things (IoT) has increased the emphasis on protecting device communication. The Constrained Application Protocol (CoAP) extensively utilizes Datagram Transport Layer Security (DTLS) to provide end-to-end security; however, DTLS has a high connection overhead. This work presents a new Group Oriented Lightweight Payload-based Mutual Authentication (GOLPMA) mechanism that authenticates clients and servers before permitting communication using the Advanced Encryption Standard (AES). GOLPMA combines and improves essential parts of the CoAP architecture, resulting in a low-overhead, dependable, and computationally efficient alternative to DTLS. The payload-based mutual authentication technique only requires two round-trip message exchanges and restricts the payload of each message to 256 bits. GOLPMA surpasses typical DTLS/CoAP techniques based on throughput, latency, packet delivery ratio, average response time, mean jitter, authentication time and energy consumption in two separate network topologies with 15 and 30 nodes when examined using the Cooja simulator on Contiki OS. The results suggest that GOLPMA could be a safe, low-power replacement for DTLS in IoT applications

Keywords: Internet of Things (IoT), DTLS, CoAP, AES, Payload encryption, Security

Introduction

The proliferation of IoT devices has resulted in the development of numerous applications and services that enable data exchange and collection. These devices work in various situations, including industrial and consumer settings, and create massive amounts of data that must be transmitted securely and efficiently. However, the resource limits of IoT devices, such as low memory and computing capacity, make elaborate security methods difficult

to deploy (Vorakulpipat *et al.*, 2018; Alfandi *et al.*, 2021; Akshatha *et al.*, 2022; Alam, 2022; Gunnarsson *et al.*, 2022).

The CoAP protocol was created to let IoT devices communicate effectively with each other while simultaneously dealing with their resource limitations. IoT supports a request/response architecture similar to HTTP, allowing devices to communicate with one another using a simple

Department of CSE, University of Visvesvaraya College of Engineering, Bangalore University, India. E-mail: akshatha.ps@gmail.com; harshithharshi253@gmail.com; dilipkumarsm@gmail.com

* Corresponding author

DOI:

messaging language. The fact that CoAP is made to operate over UDP lessens the overhead involved in establishing and maintaining a connection, making it a better choice for IoT devices than TCP. CoAP is susceptible to security concerns, including eavesdropping, data tampering, and impersonation attacks, because it lacks built-in end-to-end security safeguards. Numerous security techniques have been proposed to remedy the lack of built-in security in CoAP. As a security solution for CoAP, the DTLS protocol has seen widespread adoption (Bhattacharjya *et al.*, 2020; Masadeh *et al.*, 2023a). End-to-end security for IoT devices is provided by DTLS, which encrypts CoAP messages using a Public Key Infrastructure (PKI) and allows mutual authentication between the client and server. However, due to its high connection overhead and processing complexity, DTLS has limits and can be difficult to deploy in resource-constrained IoT devices. These concerns have driven the creation of lightweight security methods capable of enabling efficient and secure communication between IoT devices.

To provide low-overhead and efficient security for CoAP communication, this study proposes a new approach known as Group Oriented Lightweight Payload-based Mutual Authentication (GOLPMA). GOLPMA authenticates the clients and servers before permitting connection, providing a secure alternative to DTLS for IoT devices by employing AES and a payload-based mutual authentication scheme. The suggested technique provides a low-power, dependable, and secure alternative to DTLS for IoT devices, with the potential to replace DTLS in IoT applications. This paper makes the following contributions:

- To incorporate basic DTLS elements into CoAP to provide GOLPMA for secure communication between authorized entities in a distributed network.
- To establish a handshaking protocol based on payload encryption, ensuring a speedy and secure application layer authentication process.
- To Add a NACK-based retransmission approach to GOLPMA for conveying specific missing data to the sender during flight, reducing communication burden.
- To include a buffer size to convey the exact size to the server, to eliminate DTLS buffer overflow, and to boost GOLPMA performance without risking security.
- To eliminate unnecessary protocol version and sequence number components of the nonce explicit field from the packet header in

GOLPMA, considerably reducing system complexity.

- To introduce the nonce and seed values to dynamically generate the secret and session keys to increase the security of DTLS.
- To provide a comprehensive comparative analysis between GOLPMA and the existing CoAP-DTLS method, highlighting the advantages and effectiveness of GOLPMA in terms of reduced delay, improved throughput, reduced packet drops, lower response time, reduced jitter, reduced authentication time, and lower energy consumption.

Related Work

This section compiles a list of noteworthy related works published in recent years.

Payload Encryption-based Optimisation Scheme (PEOS) is a simple authentication method for distributed sensors. In order to modify the DTLS handshaking and retransmission methods, PEOS uses payload encryption and NACK messages. PEOS integrates and optimises essential DTLS components within the CoAP architecture. Additionally, it improves CoAP by using techniques for dynamic key generation and optimised AES operations to lower the danger of key tracing and guessing attacks. On Contiki OS, PEOS effectiveness is evaluated using the Cooja simulator (Abosata *et al.*, 2022).

For monitoring and controlling standard sensing devices in IoT and IIoT applications, CoAP enables multicast communication. Both unicast and multicast CoAP communication must be protected to protect the devices confidentiality and integrity. The DTLS Handshake can generate session keys and perform mutual authentication for unicast CoAP, but multicast CoAP cannot make use of it. A group key and a set of pairwise keys are necessary for multicast CoAP security. It just takes two message exchanges between a CoAP client and a collection of CoAP servers to use a security bootstrapping technique to generate keys in a speedy and secure manner. The multicast CoAP security architecture and associated protocols are implemented by the exchange of two multicast CoAP messages (Park *et al.*, 2020).

The work (Khan *et al.*, 2021) presents a safe and privacy-preserving key agreement and authentication protocol for smart grids that eliminates key escrow concerns. The protocol's security is explicitly evaluated using the random oracle model, and its validity is established. To communicate with the smart grid, a lightweight anonymous key agreement mechanism is being

created. The proposed LAKAF protocol's security is validated using the AVISPA security verification tool. LAKAF includes security attributes such as protection against a man-in-the-middle attack, replay attack, anonymity, message authentication, service provider anonymity, key freshness, impersonation attack, session key agreement, and non-traceability.

Due to their resource-intensive handshaking techniques, CoAP and MQTT are two IoT application layer protocols that present difficulties when employing TLS and DTLS for security. The authors (Nathi *et al.*, 2019) provide a brand-new security technique to deal with this issue that uses CoAP as the transport protocol for end-to-end message delivery and is based on object (payload) security with a symmetric key. By including private keys into the encryption when the 'Things' are made, the Payload encryption with a necessity for resource-intensive key distributions.

A new security mode built on a group-oriented strategy is provided to investigate end-to-end security rather than merely a handshake between a CoAP client and a collection of servers and the capability ID for granular access management of CoAP servers in a preassigned group (Park *et al.*, 2018). The authors (Jan *et al.*, 2019) suggest a simple mutual authentication technique that uses four-way handshakes for identity validation and payload-based encryption for real-world IoT objects. The proposed method protects against unauthorized access by using pre-shared secrets for object identification that are only known to authorized clients and servers.

The authors (Raza *et al.*, 2016) examine the potential problems that arise when DTLS is employed directly on IoT networks. To overcome these concerns, the authors propose dividing the DTLS protocol into two phases: handshake and encryption. Because data encryption and decryption are done at the end node, this technique enhances performance at both the device and network levels. Furthermore, based on the properties of CoAP-enabled sensors, the author suggests two scenarios (inbound and outbound). Despite the use of delegation, the proposed method assures secure end-to-end communication.

Although DTLS-based schemes provide a wide choice of cipher suites for security provisioning, DTLS built for networks with plentiful resources. The resource-intensive sophisticated cipher suites of DTLS do not prioritize message length as a fundamental design criterion for network security. As a result, deploying DTLS for

IoT systems can be expensive and may not be appropriate for network security. The suggested GOLPMA architecture could be a secure and low-power alternative to DTLS in IoT applications.

Proposed GOLPMA Architecture

The DTLS security protocol is used to implement the CoAP protocol at the IoT application layer. For secure communication, DTLS employs an encryption scheme approach and four round trips to establish an encrypted connection between clients and servers for CoAP. Because TCP's three-way handshake, retransmission, and error-detection processes need more energy and create longer communication delays, CoAP is based on UDP rather than TCP (Park *et al.*, 2014; Naik *et al.*, 2017; Kumar *et al.*, 2020; Yilmaz *et al.*, 2021; Almeghle *et al.*, 2023; Helbig *et al.*, 2023; Masadeh *et al.*, 2023b). The basic DTLS handshake process consists of six phases. The procedure begins with the client sending a *Client Hello* message to the server, which contains information on the supported protocol versions, cipher suites, and compression methods. The server will acknowledge with a *Hello Verify* message in order to confirm that the client is legitimate. Next, the client sends a second instance of the Client Hello message along with a specified cookie.

The server replies to the client's message with a *Server Hello* message that contains information about the protocol version, Cipher suite, and, if available, a digital certificate for authentication. Following the formation of a key pair, the client notifies the server through *Client-key-exchange* message and includes its public key. The server acknowledges by sending a *Change Cipher Spec* message, indicating that all further messages will be encrypted using the predefined settings. The six-flight handshake procedure in DTLS increases process complexity and may affect latency and energy usage.

In two-way communication, the GOLPMA system suggested for DTLS-CoAP establishes the identities of client nodes and servers. The authentication method uses a lightweight client and server challenges for two-way authentication using encrypted payloads with increased DTLS capabilities, which enhances the payload-based authentication technique's performance and minimizes its overhead. AES-CCM encryption is used to employ cipher block chaining and message authentication codes. Figures 1a and 1b depict the existing basic DTLS design with the suggested architecture.

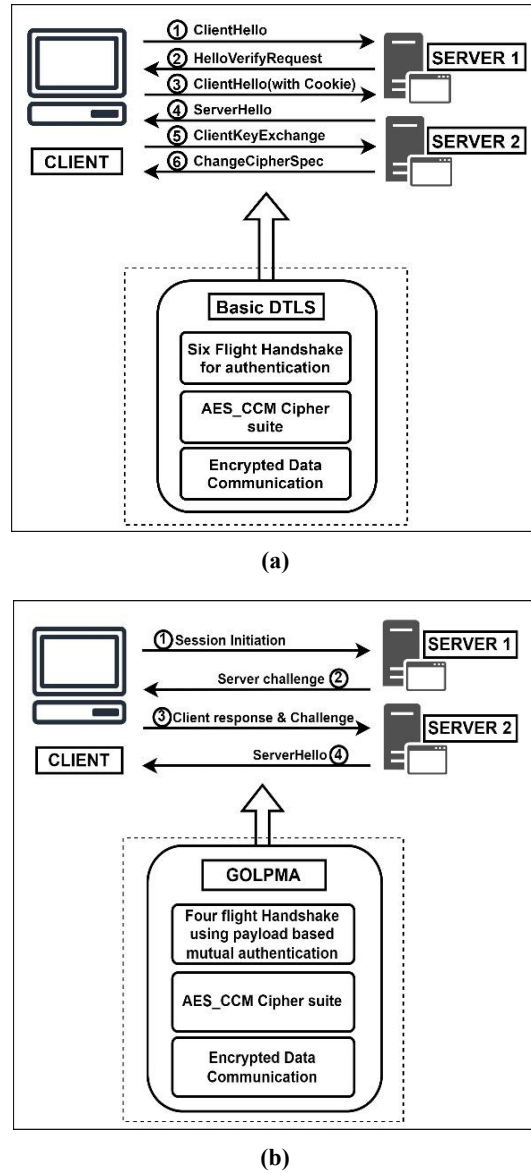


Figure 1. (a) Existing DTLS Architecture (b) Proposed Architecture

This paper examines the encrypted payload-based mutual authentication method in four-way handshaking with new packet features. Existing work, which employs a six-way handshake, aims to reduce CoAP overhead by utilizing four-way authentication mechanisms. However, the six-way handshake architecture hinders performance compared to the streamlined approach of GOLPMA. Due to the structure of DTLS flight messages, where the sender is uninformed of the exact lost data in the current flight, a single DTLS fragment loss or delay necessitates re-transmission of the entire flight. With the NACK approach in GOLPMA, when the receiver detects the absence of

certain data fragments, it sends a negative acknowledgment back to the sender, indicating the missing data's exact location within the flight. Armed with this information, the sender can selectively retransmit only the missing fragments, optimizing the retransmission process and reducing the communication burden.

By minimizing retransmissions to only the necessary data, the proposed GOLPMA system improves overall communication efficiency and reduces latency, leading to faster and more reliable data exchange between the client and the server. This NACK-based retransmission approach significantly enhances the payload-based mutual authentication procedure, making it more robust and efficient in handling potential data loss scenarios. As a result, the GOLPMA system offers improved performance and reliability, making it a promising solution for secure and efficient CoAP communication in IoT networks.

Buffer overflow is a common issue in computer systems where the data being received exceeds the capacity of the allocated buffer, leading to data loss or corruption. In the case of DTLS-CoAP communication, if the server is not aware of the exact size of the incoming data, there is a risk of buffer overflow, especially if the received payload is larger than the buffer size allocated for it. This can result in data truncation and loss of critical information.

To address this concern, we have introduced a buffer size component in the GOLPMA system. The client conveys the exact size of the payload to the server during the handshake process. By doing so, the server can allocate an appropriate buffer size to accommodate the incoming data accurately. This prevents buffer overflow situations and ensures that all data sent by the client is received and processed correctly by the server.

The authentication is carried out by the GOLPMA technique with DTLS-CoAP architecture using the payload of messages sent back and forth between the client and server. The two-way authentication method uses four handshake messages, updated DTLS features, messages with accurate buffer sizes, and a system for retransmitting lost data using the NACK approach. During the authentication process, both the client and server challenge one another. The Advantages of GOLPMA over basic CoAP-DTLS are listed below:

1. *Streamlined Handshaking*: GOLPMA adopts a 4-way handshaking process, reducing complexity and communication overhead compared to the 6-way handshaking in basic CoAP-DTLS.

2. **Intelligent Load Distribution:** GOLPMA utilizes two servers, ensuring balanced workloads and faster response times by allowing clients to connect to either server based on availability.
3. **Enhanced Authentication:** GOLPMA employs lightweight client and server challenges for two-way authentication, utilizing encrypted payloads with enhanced DTLS capabilities for improved performance and reduced overhead.
4. **Efficient Encryption:** GOLPMA utilizes AES-CCM encryption, enhancing data security and confidentiality in communication between client nodes and servers.
5. **Reduced Retransmissions:** GOLPMA optimizes payload-based mutual authentication with proper buffer size intimation and missing data retransmission mechanisms, minimizing unnecessary retransmissions and communication burden.
6. **Improved Error Handling:** GOLPMA introduces a NACK mechanism for precise error handling, conveying specific missing data during flight, and avoiding the need for resending the entire flight in case of DTLS fragment loss or delay.

The GOLPMA architecture introduces a significant improvement by incorporating multiple servers capable of collectively handling clients' requests. This innovative group-oriented server approach sets it apart from conventional methods and significantly enhances overall efficiency and performance. When clients, such as A and B, send

their requests to the servers, the system ensures swift responses by distributing the load among available servers. This workload allocation among servers improves the system's performance significantly and creates a more responsive and effective communication environment.

The four methods shown in Figure 2 are all included in the suggested strategy.

Session/Connection initiation:

In GOLPMA, the client utilizes the POST method to initiate a session with the server. This method allows the client to send data to the server's "/.well-known/authorize" resource. The message includes a unique message ID and is of type "CON" (Confirmable) to expect an acknowledgment. Before session initiation, clients establish a 128-bit AES pre-shared key with the server. Each client's query includes a token for linking to the response and the object ID in the payload to identify the resource to be accessed. The request also contains "Auth" and "Auth-Msg-Type" arguments, with "Auth=TRUE" and "Auth-Msg-Type=0" denoting the start of the server session. Upon successful client ID verification, the server grants resource authorization. The combination of elements in the CoAP message facilitates the GOLPMA scheme's secure authentication and authorization, establishing a secure communication channel between the client and server using CoAP with DTLS. Figure 3 depicts the GOLPMA's initial connection phase.

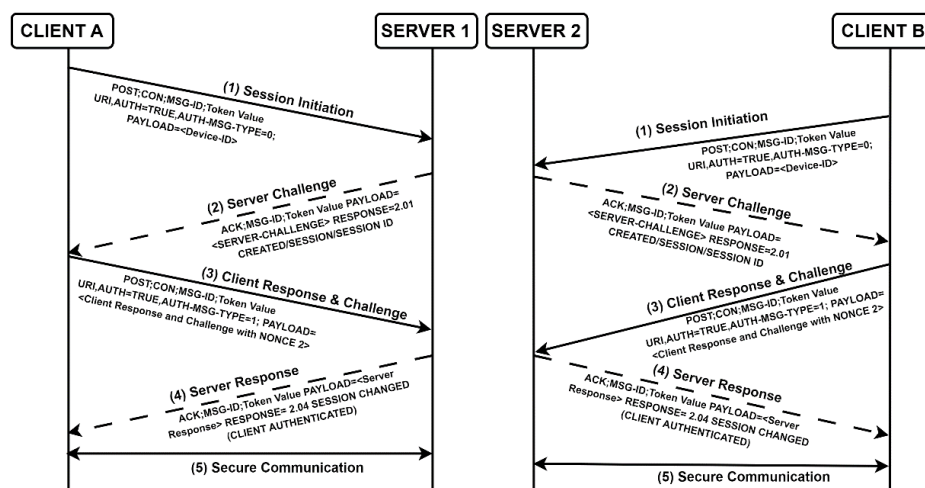


Figure 2. (a) A payload-encrypted two-way authentication system

```

00:20:290 ID:13 DEBUG: Sending to peer: [fe80:0000:0000:0000:0200:0000:0000:0003]:20220
00:20:292 ID:7 Auth true Auth_Msg_Type 0
00:20:295 ID:13 /* Session Initiation Phase */
00:20:297 ID:14 DEBUG: Client set connection to: [fe80:0000:0000:0000:0200:0000:0000:0003]:20220
00:20:298 ID:13 Auth true Auth_Msg_Type 0
00:20:298 ID:7 Request Message Sent to server from NodeId : 7
00:20:300 ID:14 dtls_new_peer: :20220
00:20:304 ID:7 Id : 7 --Payload Start Message Send :
00:20:305 ID:14 DTLSV12: Initialize HASH_SHA256
00:20:305 ID:13 Request Message Sent to server from NodeId : 13
00:20:311 ID:13 Id : 13 --Payload Start Message Send :
00:20:312 ID:14 send handshake packet of type: client_hello (1)
00:20:312 ID:7 Payload Start Message : 0|7|true|.well-known|core|1|707|
00:20:319 ID:13 Payload Start Message : 0|7|true|.well-known|core|1|901|
00:20:322 ID:7 Id : 7 Payload DeviceId Request : POST/boot|CON (Payload(DeviceId)cf5c)
00:20:322 ID:14 DEBUG: Sending to peer: [fe80:0000:0000:0000:0200:0000:0000:0003]:20220
00:20:326 ID:14 /* Session Initiation Phase */
00:20:329 ID:13 Id : 13 Payload DeviceId Request : POST/boot|CON (Payload(DeviceId)cf74)
00:20:330 ID:14 Auth true Auth_Msg_Type 0

```

Figure 3. Session Connection Initiation Phase

[illegible]

Figure 4. Server Challenge (Server 1)

```
00:20.952 ID:3 send handshake packet of type: client_hello (1)
00:20.955 ID:12 Got Response Data 20 NodeId 12
00:20.961 ID:12 Energy Consumption Is 17373 NodeId 12
00:20.962 ID:3 DEBUG: Sending to peer: [0000:0000:0000:0000:0000:0000:0000:0000]:20220
00:20.965 ID:3 /* server challenge */
00:20.966 ID:12 /* Client Response and challenge */
00:20.970 ID:12 Auth true Auth_Msg_Type 1
00:20.971 ID:12 server received request from clientid 3
00:20.981 ID:12 payload client side appkey : POST/boot[CON (Payload(appkey_client)1141ea4136)]
00:20.982 ID:11 payload client Encrypted REQUEST :
34323063000000000000000001000000015a0000c00000485a00030ffa0000526573706f6e73652066726fd2073657276657200000000000000000000004300ffdf7f
00:20.985 ID:12 request from nodeid : 12420c
00:20.986 ID:11 Data Send Time 20 NodeId 11
00:20.990 ID:3 Verification-Process : deviceID : c96d Array : c96d|6| is matched wltch recorded deviceId and psharedKey
00:20.990 ID:11 Post Requested Data 20 NodeId 11
00:20.991 ID:16 payload client Encrypted REQUEST :
34323063000000000000000001000000015a0000c00000485a00030ffa0000526573706f6e73652066726fd2073657276657200000000000000000000004300ffdf7f
00:20.992 ID:3 Notification is fresh for clientNode 3
00:20.995 ID:16 Data Send Time 20 NodeId 16
00:21.000 ID:16 Post Requested Data 20 NodeId 16
00:21.000 ID:16
00:21.000 ID:3 2.0! session created deviceId and array is matched.....
```

Figure 5. Server Challenge (Server 2)

Server challenge:

During the server-to-client communication in GOLPMA, the server sends an ACK message to respond to the client's request. This ACK message contains a unique message ID and a token value for linking it to the client's original request. The payload of the ACK message includes the server's challenge, denoted as “SERVER-CHALLENGE”, along with additional information. In the challenge phase, the server generates two 128-bit strings: a potential session key and a pseudo-random nonce.

With these values, the server creates an encrypted payload. To identify the client, the server extracts the object ID from the message payload and looks it up in a table to find a matching pre-shared

key. Once a match is found, the server sends back an AES-encrypted payload. The successful establishment of the client's new session is indicated by the response code 2.01 (created). Additionally, during this phase, the server generates the application session key using the nonce and pre-shared key. Figures 4 and 5 illustrate the server challenge phases for GOLPMA servers 1 and 2, respectively.

Client response and challenge:

In this phase, the client sends a POST message to the server, indicating a CON request with a specific message ID and a token value for linking the message to the relevant response. The URI in the

is encrypted and sent to the client. Upon receiving this response, the client deciphers the encrypted payload during the response phase, revealing the details of the client's challenge. Successful decryption serves as validation of the client's self-authentication, confirming its identity.

During the same response phase, the server also decrypts the encrypted payload received from the client's challenge. The server's successful decryption indicates that the client has effectively completed its authentication process. This successful mutual authentication and decryption demonstrate that both the client and the server have established trust and authenticity. As a result, the client and server now share a session key that will be used for secured packet exchange. This shared session key ensures that subsequent communication between the client and server remains secure and confidential, protecting sensitive information and data integrity throughout their interactions. Figures 7 and 8 depict the server response phases of server 1 and server 2.

In this phase, the server responds to the client's request with an ACK message, containing the message ID and token value for proper correlation. The payload includes the "Server Response", which

Figure 6. Client Response and Challenge Phase

Figure 7. Server Response (Server 1)

Figure 8. Server Response (Server 2)

A detailed process for achieving secure communication and authentication in a system is depicted in Algorithm 1. Each client receives a distinct Object_ID and pre-shared key during the initialization step, which are then saved in the array designated by the server. The algorithm then moves through distinct steps to set up a secure session between the clients and the server. In the initial stage, Session Initiation, the client transmits a CON message to the server with its Object_ID. After locating the relevant pre-shared key related to the Object_ID it had just received, the server then evaluates the transmitted CON data. After then, the algorithm follows a predetermined path to create a secure session between the clients and the server. In the first phase, the client sends a CON message to the server with its Object_ID. The server then assesses the provided CON data and obtains the matching pre-shared key linked to the received Object_ID.

The server responds with information signaling the beginning of the session if the evaluation is successful. The client is denied access if the evaluation fails and an unauthorized response

is generated. This approach then uses a challenge-response system to verify the identities of the client and server after the initial stage. By sending an encrypted payload using the shared pre-shared key, the client expressly responds to the server's challenge. In order to identify discrepancies or errors, the server compares the client's answer to the requests. In the event that the responses are consistent, access is allowed and the client's authenticity is confirmed. Access is only permitted, though, provided the answers line up, and the client doesn't provide any more authentication.

The client can establish a secure communication channel with the server using a shared session key when the client has successfully authenticated. The algorithm successfully supports mutual authentication and secure communication between the clients and the server. The algorithm builds a strong foundation of trust and ensures the security of internal communication by utilizing pre-shared keys, a challenge-response mechanism, and encrypted payloads.

Algorithm 1 : Secured Communication and Authentication

Initialization:

- a. Every Client C_i obtains a distinct Object_ID i along with pre-shared key λ_i
- b. The server S is provided with every Object_ID i and pre-shared key λ_i stored in array A [][]

Step 1 [Session Initiation] : [Input : (Object_ID i , pre-shared key λ_i)]

For i : =1 to N

For j : = 1 to 2

Input ($A[i][j]$) (Object_ID i and pre-shared key λ_i of Client C_i are stored in an array)

Client C_i transmits a CON memo containing Object_ID i within the payload to server S

Step 2 [Server Challenge] : The available Server S retrieves Object_ID i and finds matching λ_i to evaluate the CON information that has been communicated.

If Object_ID i == $A[i][j]$

Session generate 2.01

Server S retransmits an encapsulated payload, AES (λ , ($\lambda_i \oplus \mu$ key | server)) where μ is the key

Else

Unauthorized 4.01

Step 3 [Client Response and Challenge] : Client C_i decodes the request and delivers an encrypted payload, AES (μ key, (server $\oplus \lambda_i$ | Client))

Step 4 [Server Response] : The server compares the client request to the server request to check for errors

If they are both equivalent

[Access is permitted] - Client C_i Verified

The Server responds as AES

Else

[Access Denied] - Client C_i Unauthenticated

The client compares Client and Server response

If both match

Server S authenticated

Step 5 [Secure Communication] : Server S and Client C exchange data with each other

Else

Server S is not proven

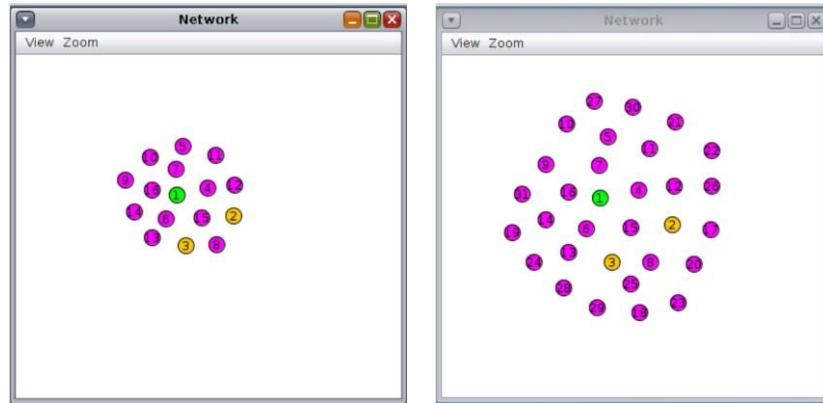


Figure 9. Network Design for 15 Nodes and 30 Nodes Respectively

Table 1. Experimental Parameters

Application Layer	CoAP
Transport Layer	UDP
Network Layer	6LoWPAN
MAC layer protocol	802.15.4
Total Nodes	15 & 30 Nodes
Transmission Range	50m
Simulation Area	100m x 100m
No. of Servers	2
Payload Size	256 bytes
Simulation Time	300 seconds

Experimental Results

The suggested GOLPMA approach combined with DTLS and CoAP structure is shown using the Contiki Cooja network simulator (Ferreira *et al.*, 2023; Behal *et al.*, 2023). The goal of this experimental setup is to establish a secure connection utilizing DTLS-CoAP and the GOLPMA implementation for mutual authentication. The experimental results and conclusions from analyzing DTLS/CoAP and GOLPMA are shown in the section that follows. The experiment is carried out over a period of 300 seconds with 15 and 30 client nodes, one border router, and two servers. Figure 9 depicts the network design for 15 nodes and 30 nodes respectively. Wismote and z1 mote types are utilized in the simulation for server and client nodes and border routers, respectively. The simulation is run with various client node counts to explore the outcome of network scalability on the effectiveness of lightweight authentication methods. The distributed IoT network's payload size is 256 bytes, and the simulation is run in a 100×100m² region. Every node has a 50-meter maximum communication range. The experimental parameters are simulated

using two distinct topologies, as indicated in Table 1: the 15-Node Structure with 1 border router, 2 servers, and 13 clients, and the 30-Node Structure with 1 border router, 2 servers, and 28 clients

Performance Parameters

This subsection discusses the parameters for lightweight payload-based mutual authentication using GOLPMA and CoAP/DTLS.

- **Packet Delivery Ratio:** It is the proportion of data packets sent from the sender that are successfully received at the gateways, indicating the effectiveness of data transmission. equation (1). defines the formula used to evaluate the PDR.

$$\text{PDR (\%)} = (\text{Total received data packets} / \text{Total sent data packets}) \times 100 \quad (1)$$

The PDR percentage for 15 nodes in the traditional CoAP-DTLS system is 35.6%, however, in the GOLPMA scheme, it is 36.08%. The PDR percentage for 30 nodes in the DTLS-CoAP scheme is 53.5% while it is 54.2% in GOLPMA. Figure 10 shows that the suggested method outperforms the standard DTLS-CoAP strategy.

- **Energy Consumption:** It is the average energy consumed across the network's sensor nodes divided by the number of nodes. equation (2). defines the formula used to evaluate the energy consumption.

$$\text{Energy consumption} = \text{Total Energy} / \text{No of nodes} \quad (2)$$

The traditional DTLS-CoAP uses 0.000699 joules of energy for 15 nodes, than the proposed system's 0.000599 joules. Similarly, the traditional

DTLS-CoAP consumes 0.0006888 joules for 30 nodes, the Golpma uses 0.000607 joules. As depicted in Figure 11, the Golpma approach consumes less energy while functioning comparable to standard CoAP/DTLS.

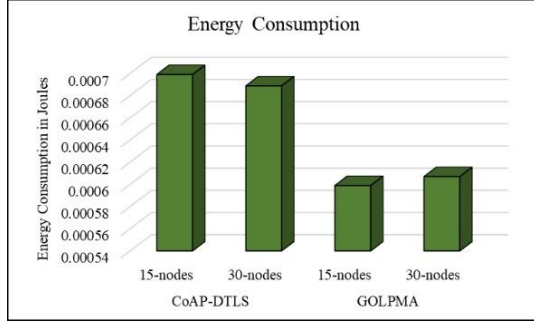


Figure 11. Energy Consumption in CoAP/DTLS vs GOLPMA

- **Delay:** The average amount of time required for a packet to be transmitted from the originating source to the gateway across the network is called as a delay. It can be calculated using equation (3).

$$\text{Delay} = \text{Total Delay} / \text{Total \#Clients} \quad (3)$$

The typical DTLS-CoAP technique has a delay of 6.57 seconds for 15 nodes, but GOLPMA has a delay of 2.56 seconds. Similarly, for 30 nodes, the traditional DTLS-CoAP technique has a delay of 8.22 seconds, whereas GOLPMA has a delay of only 2.66 seconds. As indicated in Figure 12, the GOLPMA technique surpasses the typical DTLS-CoAP method based on minimizing delay in the network, as evidenced by the much lower delay values seen in the GOLPMA approach than the DTLS-CoAP method.

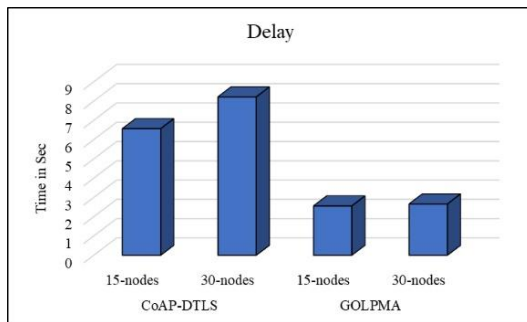


Figure 12. Delay in CoAP/DTLS vs GOLPMA

- **Throughput:** It is the amount of data sent to the nodes per second. The throughput is calculated using equation (4).

$$\text{Throughput} = (\text{Total Packets} \times \text{Data Length} \times 8) / 300 \quad (4)$$

The standard DTLS-CoAP strategy achieves a throughput of roughly 46.8 bps in a simulation where 15 nodes are added to the IoT topology to simulate CoAP communication, whereas the recommended GOLPMA scheme boosts the throughput to 56.64 bps. The DTLS-CoAP scheme provides a throughput of approximately 97.2 bps in a simulation of the IoT topology with 30 nodes added, but the GOLPMA scheme increases the throughput to 117.12 bps. Figure 13 shows that the suggested GOLPMA technique outperforms the default DTLS-CoAP strategy based on throughput.

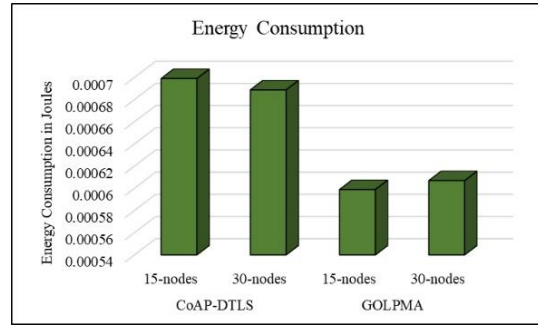


Figure 13. Throughput in CoAP/DTLS vs GOLPMA

- **Average Response Time:** It represents the mean time between the initiation of a client's request and the reception of the corresponding response from the server. It is calculated using equation (5).

$$\text{Average Response Time} = (\Sigma \text{Response Time for all Requests}) / \text{Number of Requests} \quad (5)$$

In the CoAP-DTLS system with 15 nodes, the average response time is measured at 477 ms, while in the GOLPMA scheme, it is significantly reduced to 392 ms. Similarly, for 30 nodes, the CoAP-DTLS system exhibits an average response time of 608 ms, whereas GOLPMA achieves a lower response time of 501 ms.

These results indicate that the GOLPMA architecture outperforms the traditional CoAP-DTLS approach in terms of average response time for both 15-node and 30-node scenarios. Figure 14

represents the average response time for the proposed GOLPMA over the standard DTLS-CoAP strategy.

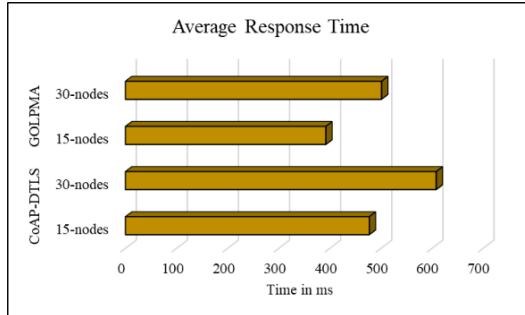


Figure 14. Average response time in CoAP/DTLS vs GOLPMA.

- *Mean Jitter*: It measures the typical departure from anticipated packet arrival timings. In other words, it displays the discrepancy between the actual and predicted packet arrival timings. It is calculated using equation (6).

$$\text{Mean Jitter} = \Sigma | (\text{Packet Arrival Time} - \text{Expected Arrival Time}) | / \text{Number of Packets.} \quad (6)$$

In the CoAP-DTLS system with 15 nodes, the mean jitter is 1.201 seconds, while in the GOLPMA scheme, it is reduced to 0.682 seconds. For 30 nodes, the CoAP-DTLS system shows a mean jitter of 1.614 seconds, whereas GOLPMA achieves a lower mean jitter of 0.791 seconds. These results highlight the significant improvement in jitter achieved by GOLPMA compared to the traditional CoAP-DTLS approach for both 15-node and 30-node scenarios. Figure 15 shows that the suggested GOLPMA technique outperforms the default DTLS-CoAP strategy based on mean jitter.

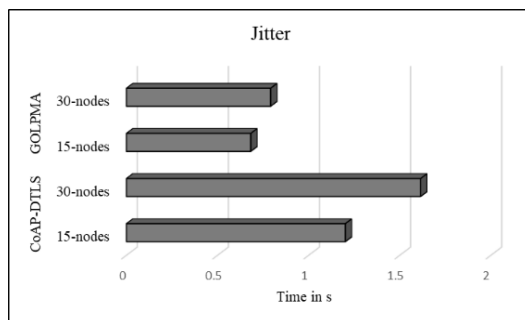


Figure 15. Mean jitter in CoAP/DTLS vs GOLPMA

- *Authentication Time*: It represents the duration from the initiation of the session by the client to the successful establishment of the shared session key between the client and the server. It is calculated using equation (7).

$$\text{Authentication Time} = T_{\text{session}} + T_{\text{challenge}} + T_{\text{response}} + \delta_{\text{proc}} \quad (7)$$

Where T_{session} is the round-trip time taken by the session initiation request, $T_{\text{challenge}}$ is the round-trip time taken by the client response & challenge, T_{response} is the round-trip time taken by the server response and δ_{proc} is the processing time taken by the client during the authentication process.

The authentication time in the CoAP-DTLS system with 15 nodes is recorded at 3.172 seconds, while GOLPMA achieves a remarkable reduction to 0.93 seconds. Similarly, in the 30-node scenario, the CoAP-DTLS system shows an authentication time of 3.531 seconds, whereas GOLPMA achieves a significantly lower authentication time of 1.02 seconds. These findings underscore the considerable improvement in authentication time offered by GOLPMA over the conventional CoAP-DTLS approach for both 15-node and 30-node scenarios. As indicated in Figure 16, the GOLPMA technique surpasses the typical DTLS-CoAP method based on minimizing authentication time in the network.

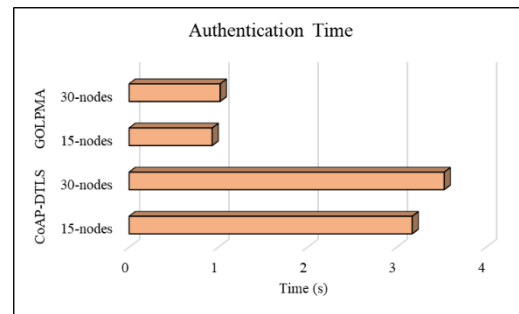


Figure 16. Authentication time in CoAP/DTLS vs GOLPMA

In terms of PDR, energy usage, throughput, average response time, mean jitter, authentication time and delay, the suggested GOLPMA technique has demonstrated impressive performance gains over the existing CoAP/DTLS approach. GOLPMA also provides several other advantages in addition to these. firstly, it uses a quick and effective client-server authentication handshake method that requires only two message exchanges, resulting in

a quick and secure authentication procedure. Secondly, the GOLPMA payload-based encryption technique uses a simplified four-way handshake procedure to increase data security without adding needless complexity. The suggested system also offers a strong defence against several attacks while being computationally efficient, resulting in minimal connection overhead. Overall, GOLPMA provides a comprehensive solution that excels in a variety of areas, making it a potential option for secure and efficient CoAP communication in IoT networks.

Application Domains

GOLPMA, as a proposed authentication approach, holds the potential to be effectively incorporated into various IoT application domains, offering secure communication to address their distinct security and real-time communication needs. Some of the domains where GOLPMA can find effective integration include:

1. *Industrial Automation and Control Systems (IACS)*: In industries where various devices and sensors need to communicate securely in real-time to monitor and control processes, GOLPMA ensures that data exchanged between these devices remains confidential and tamper-proof.
2. *Smart Grids*: In modern smart grids, multiple devices such as smart meters, energy monitoring systems, and grid control systems communicate with each other. GOLPMA ensures the integrity and confidentiality of data transmitted over the network, preventing unauthorized access and potential cyber-attacks.
3. *Healthcare and Medical Devices*: Real-time health monitoring devices, wearable health trackers, and medical equipment in hospitals can utilize GOLPMA to securely transfer sensitive patient data between devices and healthcare systems.
4. *Connected Vehicles and Intelligent Transportation Systems (ITS)*: In connected vehicle environments, where vehicles communicate with each other and infrastructure for real-time traffic information and safety notifications, GOLPMA helps ensure secure and trustworthy communication.
5. *Home Automation and Smart Homes*: In smart home applications, where various IoT devices are interconnected to create an automated environment, GOLPMA can be used to safeguard user privacy and prevent unauthorized access to home devices.

6. *Agriculture and Precision Farming*: In precision farming, where IoT devices and sensors are used to optimize agricultural processes, GOLPMA provides secure communication for real-time data sharing and control.
7. *Asset Tracking and Supply Chain Management*: GOLPMA can be employed to ensure the security of real-time data transmission in supply chain management and asset tracking systems, preventing data manipulation or theft during transportation and logistics processes.

Conclusions

The proposed study introduces Group Oriented Lightweight Payload-based Mutual Authentication for Constrained Application Protocol in IoT networks, a unique mutual authentication protocol. The basic DTLS features are optimized and integrated into the CoAP architecture by GOLPMA, eliminating the necessity for a separate DTLS implementation on a different channel. With payload encryption and NACK messages, the DTLS handshaking and re-transmission protocols in GOLPMA are improved. With GOLPMA, the adoption of the Advanced Encryption Standard is advised. GOLPMA's efficacy is assessed using the Cooja simulator on Contiki OS and contrasted with the traditional DTLS-CoAP method. For 15-node and 30-node topologies, simulation outcomes show that GOLPMA achieves higher Packet Delivery Ratio percentages of 36.08% and 54.2%, lower energy consumption of 0.000599 joules, and 0.000607 joules, reduced delay of 2.56 seconds and 2.66 seconds, and increased throughput of 56.64 bps and 117.12 bps, respectively. Additionally, GOLPMA demonstrates notable improvements in performance metrics such as average response time, mean jitter, and authentication time. Specifically, GOLPMA achieves an average response time of 392 ms and 501 ms for 15-node and 30-node scenarios, respectively, indicating faster and more efficient communication. The mean jitter in GOLPMA is significantly reduced to 0.682 seconds and 0.791 seconds for 15-node and 30-node topologies, respectively, compared to CoAP-DTLS values of 1.201 seconds and 1.614 seconds. Moreover, the authentication time in GOLPMA is substantially decreased to 0.93 seconds and 1.02 seconds for 15-node and 30-node scenarios, respectively, compared to CoAP-DTLS values of 3.172 seconds and 3.531 seconds. These comprehensive results demonstrate that GOLPMA outperforms the typical DTLS-CoAP strategy across multiple performance metrics,

making it a more efficient and reliable approach for secure communication in IoT networks.

As a potential avenue for future research, it would be valuable to conduct comprehensive investigations into the effectiveness of GOLPMA in diverse IoT network scenarios. This could involve studying its performance and scalability in different network topologies, evaluating its resilience to node mobility, and analyzing its robustness against various security attacks. Such research endeavors would provide deeper insights into the real-world applicability and performance of GOLPMA, further validating its potential as a secure authentication protocol for CoAP communication in IoT networks.

References

- Abosata, Nasr, Saba Al-Rubaye, and Gokhan Inalhan. (2022). Lightweight Payload Encryption-Based Authentication Scheme for Advanced Metering Infrastructure Sensor Networks. *Sensors*, 22(2):26.
- Alam, T. (2022). Opportunistic Manet And Its Role In Next-Generation Android Iot Networking. *Suranaree Journal of Science and Technology*, 29(4):16.
- Alfandi, Omar, Salam Khanji, Liza Ahmad, and Asad Khattak. (2021). Survey on boosting IoT security and privacy through blockchain: Exploration, requirements, and open issues. *Cluster Computing*, 24(1):18
- Almeghle, S.M., AL-Ghamdi, A.A.M., Ramzan, M.S., and Ragab, M. (2023). Application Layer-Based Denial-of-Service Attacks Detection against IoT-CoAP. *Electronics*, 12(12):29
- Behal, A., Sandhu, J.K., and Gupta, G. (2023). Using The Cooja Simulator, Analysing The Routing Protocol (RPL) For Low Power And Lossy Networks In IoT. *Proceedings of IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCECS)*; Feb 18-19, 2023. Bhopal, India. p. 1-4
- Bhattacharjya, Aniruddha, Xiaofeng Zhong, Jing-Wang, and Xing Li. (2019). CoAP-application layer connection-less lightweight protocol for the Internet of Things (IoT) and CoAP-IPSEC Security with DTLS Supporting CoAP. *Digital twin technologies and smart cities*. Maryam, Farsi (Eds). Springer, Switzerland, p.151-175.
- Ferreira, A.M.A., Azevedo, L.J.D. M.D., Estrella, J.C., and Delbem, A.C.B. (2023). Case Studies with the Contiki-NG Simulator to Design Strategies for Sensors' Communication Optimization in an IoT-Fog Ecosystem. *Sensors*, 23(4):18.
- Gunnarsson, Martin, Krzysztof Mateusz Malarski, Rikard Höglund, and Marco Tiloca. (2022). Performance evaluation of group OSCORE for secure group communication in the Internet of Things. *ACM Transactions on Internet of Things*, 3(3):31.
- Helbig, C., Otoum, S., and Jararweh, Y. (2023). Modeling and Evaluation of the Internet of Things Communication Protocols in Security Constrained Systems. *Proceedings of IEEE 20th Consumer Communications & Networking Conference (CCNC)*; Jan 8-11, 2023. Las Vegas, NV, USA. p. 1-6.
- Jan, M. A., Khan, F., Alam, M., and Usman, M. (2019). A payload-based mutual authentication scheme for Internet of Things. *Future Generation Computer Systems*, 92(1):12.
- Khan, A.A., Kumar, V., Ahmad, M., and Rana, S. (2021). LAKAF: Lightweight authentication and key agreement framework for smart grid network. *Journal of Systems Architecture*, 116(c):11.
- Kumar, Priyan Malarvizhi, and Usha Devi Gandhi. (2020). Enhanced DTLS with CoAP-based authentication scheme for the internet of things in healthcare application. *The Journal of Supercomputing*, 76(6):21.
- Masadeh, R., Almomani, O., Masadeh, E., and Masa'deh, R.E. (2023). Secure CoAP Application Layer Protocol for the Internet of Things Using Hermitian Curves. In *The Effect of Information Technology on Business and Marketing Intelligence Systems*. M. Alshurideh (Eds). Springer, Switzerland, p.1,869-1,884.
- Masadeh, Raja, Omar Almomani, Esra Masadeh, and Ra'ed Masa'deh. (2023). Secure CoAP Application Layer Protocol for the Internet of Things Using Hermitian Curves. In *The Effect of Information Technology on Business and Marketing Intelligence Systems*. M. Alshurideh (Eds). Springer, Switzerland, 1,869-1,884.
- Naik, Kanchana P., and U. Rakesh Joshi. (2017). Performance analysis of constrained application protocol using Cooja simulator in Contiki OS. *Proceedings of 2017 IEEE International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*; Jul 6-7, 2017. Kerala, India. p. 547-550.
- Nathi, R.A., and Sutar, D.S. (2019). Embedded payload security scheme using CoAP for IoT device. *Proceedings of IEEE International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN)*; Mar 30-31st; Vellore, India. p. 1-6.
- Park, Chang-Seop, and Wang-Seok Park. (2018). A group-oriented DTLS handshake for secure IoT applications. *IEEE Transactions on Automation Science and Engineering*, 15(4):10.
- Park, Chang-Seop. (2020). Security architecture for secure multicast coap applications. *IEEE Internet of Things Journal*, 7(4):12.
- Park, Jiye and Namhi Kang. (2014). Lightweight secure communication for CoAP-enabled internet of things using delegated DTLS handshake. *Proceedings of IEEE International conference on information and communication technology convergence (ICTC)*; Oct 22-24, 2014; Busan. Korea (South). p. 28-33.
- PS, Akshatha, Dilip Kumar, S.M., KR, Venugopal. (2022) MQTT Implementations, Open Issues, and Challenges: A Detailed Comparison and Survey. *International Journal of Sensors Wireless Communications and Control*, 12(8):23.
- Raza, S., Seitz, L., Sitenkov, D., and Selander, G. (2016). S3K: Scalable security with symmetric keys-DTLS key establishment for the Internet of Things. *IEEE Transactions on Automation Science and Engineering*, 13(3):11.
- Vorakulpipat, Chalee, Ekkachan Rattanalerdnusunorn, Phithak Thaenkaew, and Hoang-Dang -Hai. (2018). Recent challenges, trends, and concerns related to IoT security: An evolutionary study. *Proceedings of 20th IEEE International Conference on Advanced Communication Technology (ICACT)*; Feb 11-14, 2018; Elysian gangchon, Korea (South), 405-410.
- Yilmaz, Yildiran, Leonardo Aniello, and Basel Halak. (2021). ASSURE: A hardware-based security protocol for resource-constrained IoT systems.. *Journal of Hardware and Systems Security*, 5(1):19.