CHAPTER ONE

INTRODUCTION

The chapter comprises of the background of the research in brief followed by the problem statement. After that, the objectives of the thesis, overview of the research methodology and the scope of the thesis are discussed sequentially. At the last part of the chapter, contents of the remaining chapters are briefly discussed.

1.1 Background

IoT based devices across the world is likely to be increased up to 43 billion by the end of 2023, a three times more from 2018 [1]. IoT-based application has diverse usage, for example, in automated fire control, logistics and energy management, smart health monitoring system, robotics, military surveillance, weapon system and so on [2]. IoT-based systems are equipped with wireless functionality along with sensors, communication channel between devices and back-end systems. Despite of such huge interconnectivity, security aspects of IoT world has huge scope to be explored to a great extent.

IoT based applications demonstrates mentionable security vulnerabilities. Few of the such surfaces are: IoT devices (i.e., sensors and actuators), IoT-specific applications, backend data storage and most importantly communication networks beween the devices and the back-end system etc. [3]. IoT-based platforms use several communication protocols-Advanced Message Queuing Protocol (AMQP), Constraint Application Protocol (CoAP), Message Queue Telemetry Transport (MQTT) and many others. Every such protocol has its own strength and vulnerability. MQTT is the most extensively used IoT based communication protocols. MQTT standard has no specific requirements about the security standards. Less bandwidth usage and less memory consumption make it lucrative to IoT developers uses this protocol because of its requirement and [4]. IoT device sends private data that is authorized to specific people or devices. The Protocol only provides verification for security and it does not setup encryption mechanism of data in transfer. Therefore, data security, validation, and reliability can be threatened during its implementation. As such, after finding the specific vulnerabilities of MQTT protocol, while proposing a security enhanced MQTT protocol, verification by a standard verifier, i.e., ProVerif, seems effective. Identified vulnerabilities shall be analyzed and removed while proposing such security enhanced similar protocol [5].

1.2 Problem Statements

A significant number of researches have been undertaken focusing to the security of different IoT protocols. Several researchers consider to improve the security features of the widely used IoT protocol namely MQTT. However, many of them have critics for their complexity along with the constraints of the IoT environment. Thus, further studies are required focusing on the design of security enhanced MQTT considering dynamic IoT constraints. As such, the design principles used for developing such security enhanced MQTT protocol demands to carry out a widely accepted security analysis to achieve maximum reliability, usability and compliance.

1.3 Thesis Objectives

The major objectives of the thesis are, firstly, to understand and explore the existing vulnerabilities in MQTT protocol used in IoT based applications. This will lead to identification of vulnerabilities in MQTT. Secondly, to analyze such use cases of MQTT protocol and design a security enhanced similar protocol without introducing new constraints.

1.4 Methodological Overview

The research is carried out in the number of steps sequentially. The related literature was reviewed to explore the present state of the MQTT communication protocol along with its existing variant. The study will identify the security drawbacks of MQTT protocol along with a list of use cases. Basing on the findings (security drawbacks and list of use cases in MQTT protocol), the root cause of the missing security features will be identified and analyzed. Considering the constraint and heterogeneous IoT environment the security functionality is selectively included in the existing protocol without impacting the present infrastructure. Finally, a formal verification will be conducted using the ProVerif cryptographic protocol verifier to demonstrate that the proposed communication protocol fulfills the expected security needs . This verification will cover most of testing scenarios that may have been ignored in the original protocol standard. The evaluation metrics includes confidentiality protection, integrity protection, authentication mechanism etc. [7].

1.5 Thesis Scope

The scope of this thesis has been limited to focusing the design and development of an IoT based communication protocol. In doing that, one of the most frequently used IoT communication protocols MQTT is considered. It is one of the lightest and low powered protocols in the IoT domain. This thesis has considered the vulnerability and security threat of MQTT protocol basing on number of use cases and previous studies. Basing on the findings of such analysis a security enhanced MQTT protocol is designed. validation is carried out by ProVerif verifier to check the security vulnerabilities. This verification covers most of testing scenarios that may have been ignored in the original protocol standard.

1.6 Thesis Outline

The thesis is outlined as follows:

Chapter 2: In this chapter MQTT protocol is introduced in broader perspective. The chapter also highlights on the security aspects of IoT protocols briefly. It also discusses the relevant studies and thesis works in recent past.

Chapter 3: It provides the overview of the research methodology pertinent to the thesis covering a short description of all the phases. It also briefly give an overview of ProVerif cryptographic analyser.

Chapter 4: Here the proposed security enhanced MQTT protocol is discussed in details where all the steps of authentication and steps are elaborated.

Chapter 5: It focuses on the proper analysis of the proposed protocol using the tool. The analysis takes place with the support of ProVerif analyzer software and output is projected here.

Chapter 6: Final simulation results and outcome of the thesis is proposed protocol is projected here.

Chapter 7: This chapter concludes the thesis with a brief description of various limitations. It also includes future scope of works basing on the findings of the research.

CHAPTER TWO

THEORETICAL BACKGROUND AND RELATED WORKS

This chapter firstly introduces few of the key concepts to introduce background theories followed by the focus to MQTT protocol. At the end of discussion, a critical summary is presented to highlight the research gap and motivation to this research. A number of researches have been undertaken focusing to MQTT security vulnerability and its enhancement. This section briefly introduces few of such works and identifies salient observations.

2.1 Introduction to IoT Protocol

IoT is a combination of two emerging technologies: wireless based connectivity and sensors. These connected embedded systems are independent microcontroller-based computers that use sensors to collect data from a network. It enables integration between the physical domain and IT based networks [8]. This concept emerged much before, through the development sensor technology and connected objects. With current internet infrastructure, wireless communication networks play a vital role in IoT devices allowing them to transmit and receive messages. Therefore, the vitality of these messages lies in authentication and security. Numerous key management techniques have also been introduced to provide a secured transmission over the network. It meaningfully underwrites to increase efficiency, correctness, and financial advantages [9].

With the passage of time in the context of IoT, many protocols have been devised for validation and transmission management with security. Few of them are: CoAP, ZigBee, 6LoWPAN, Bluetooth etc. Among them, MQTT is extensively used protocols in most Machine-to-Machine (M2M) communication. However, all complete IoT systems works by the integration of four distinct modules: hardware devices, connectivity, processing of data, and interface [10].

ZigBee is an IEEE 802.15.4-based protocol used for top level communication used to create personal area networks with low-powered and small digital radios. Few of the examples are house automation, medical system, and many other small scale projects. CoAP is a focused Internet Application Protocol used with constrained devices, as per defined slandered of RFC 7252. CoAP is basically an interpretation of the HTTP protocol running above UDP which limits the use of bandwidth. The 6LoWPAN protocol was created from the Internet Protocol which is applied to the tiniest devices with minor processing competences. AMQP is an open sourced protocol for commercial transmission of OASIS standard data. The protocol provides classy functionalities and is extensively used at present in many circumstances where a dependable and asynchronous transmission between terminals is needed. AMQP facilitates Simple Authentication and Security Layer (SASL) architecture for client authentication and TLS for ensuring integrity and privacy of communication. Bluetooth is a popular protocol for short-range communication. It is secure and perfect for short-range, low powered, cheap and provides manageable wireless transmission between electronic devices very easily. Bluetooth Low Energy (BLE) is an enhanced version of Bluetooth which is getting popular and familiar day by day. [11-13].

2.1.1 Security Aspects of IoT Protocol

Due to the security constraints of IoT domain and related privacy of IoT users, it exposes a versatile threat to the whole system. Therefore, design of enhanced IoT based secured protocol is a crucial issue. Although the current area in the IoT business is on the user friendliness, to advance practical possessions, and minimize costs, there is an crucial need to evaluate the security standards of IoT protocols. With current internet based connected domain, wireless communication plays a energetic role in deploying IoT devices. This allow them to communicate with messages. Therefore, the strength of these messages lies in confirmation.

2.1.2 Characteristics of IoT Protocols

There are seven basic IoT characteristics [14]:

a. **Connectivity**. Connection between various levels among devices and associated hardware, sensors and other electronics is a must for this protocol.

b. **Things**. Hardware Devices should comprise of sensors or sensing constituents can be fixed to hardware devices and items.

c. **Data**. It is the core aspect of IoT. It is the first step towards different deed and intellect.

d. **Communication**. Hardware get linked to exchange with data which can be investigated. Communication can take place over short or over a long distance. Examples: Bluetooth, Wi-Fi, LoRa, ZigBee etc.

e. **Intelligence**. It is the sensing competences in IoT devices. The intelligence collected from big data analysis.

f. **Action**. Actions can be manual or automated basing on the circumstances.

g. **Ecosystem**. A different perspective of other skills, societies, goals and the picture. Here IoT fits well from other contemporary technologies.

2.2 An Overview of MQTT Protocol

MQTT is a IoT based communication protocol which is articulated by Andy Stanford-Clark and Arlen Nipper. It uses a publish/subscribe mechanism. It is currently following OASIS (Organization for the Advancement of Structured Information Standards). Currently, the MQTT protocol also has standard defined in ISO/IEC 20922: 2016 [15]. Both Publish and subscribe operations in MQTT can be depicted like client and server models. The central server in MQTT is named as broker that acts as the recipient of the message from the client which is, otherwise the entire node involved in the transmission process. The message itself can be in the form of publish or subscribes topic. Furthermore, all the devices connected using this protocol can become publishers and subscribers. Every device that has been registered as a subscriber to a specific topic will receive a message from the broker each time the topic is updated. It is a lightweight publisher and subscriber-based protocol. Thus, MQTT is consisted of five main components, those are [16]:

a. **The Broker**: It is the worker that gets and distributes messages between customers.
b. **The Message:** It is the holder of the information that has been shipped off the agent by the distributer or has been gotten by the supporter from the intermediary.
c. **The Publisher**: It is the gadget which sends messages to the representative to refresh the information of certain topics.
d. **The Subscriber**: It is the gadget which gets messages from the representative that conveys the refreshed status of the agent's topics.
e. **The Topic**: The element on the dealer where the distributor forwards messages to it and the supporter gets messages from it.

It doesn't have obligatory necessities with respect to the security administrations like validation, privacy, information honesty, and access control [17]. Presently, tackling the security related issues as an undertaking or potentially execution explicit issue, there is no particular normalization to deal with.

2.3 Security Aspects of MQTT Protocol

In order to secure MQTT there are few commonly used approaches, like Advanced Validation Mechanisms, Authentication, Securing MQTT Systems, TLS / SSL, etc. There are few other security concepts and implementations with MQTT - X509 Client Certificate Validation, OAuth 2.0, Payload Security, Message Data Integrity etc.

The protocol itself postulates only a few security instruments. MQTT implementations commonly use other well-known security standards: for example, SSL/TLS for transport security. Since security is difficult to ensure, it is rational to makes sense to shape upon universally familiar ideals [18].

Sometimes, IoT device sent confidential data that should only be handled by authorized people or hardware devices only. The MQTT protocol only provides validation for the security mechanism which, by default, does not encrypt the data in transit. Thus, data confidentiality, validation, and data honesty become concern in MQTT execution.

2.4 Use of ProVerif for Security Testing

ProVerif is a software testing tool used for examining the security properties of different IoT based protocols. The tool has been developed by Bruno Blanchet [19].

This creative computerized tool utilized during the validation testing phase of the security resolution. It depends on Pi math and it can settle the vagueness and the anonymous belongings of the respective domain. It can process an limitless number of meetings for the convention during a test [20-21].

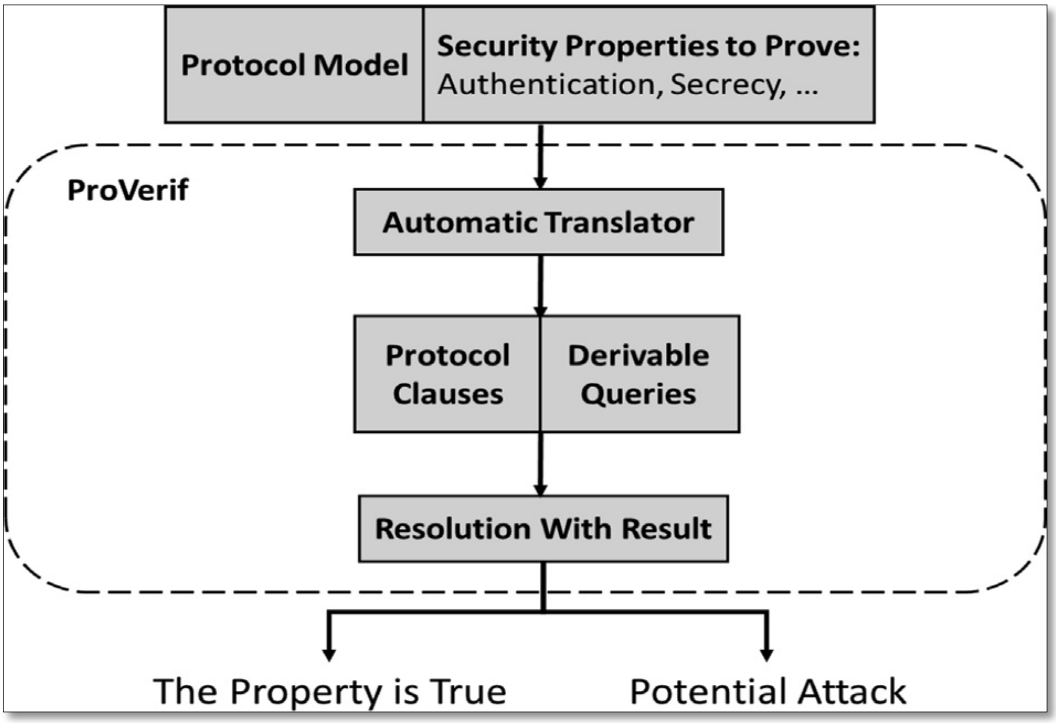


Figure 2.1: ProVerif Flow Diagram

As outlined in Fig. 3.2, ProVerif takes data sources and interprets them into convention and security inquiries needed to be confirmed utilizing the programmed interpreter [22].

2.6 Critical Summary of Theoretical Studies

The study carried out within the scope of identifying vulnerability for the IoT based communication protocols. In sum through this theoretical study of security aspects of MQTT protocol, most of them could not recommend a trustworthy protection mechanism. Earlier works were attentive to present the development of various tools to progress the protocol. From investigative perspective, the testing was conducted mostly basing on little theatrical analysis. Finally, no smart and technology-based tools like ProVerif was used to analyze and identify the vulnerabilities. Therefore, the focus of this thesis is to understand the underlying security threats of MQTT protocol.

2.5 Related Works

2.5.1 Token Based Authentication

For last few years a new Token-Based Lightweight User Authentication for IoT devices and Hardware is being deployed. Analysis depicts that security parameter of the proposed scheme. are getting prevalent day by day. Bhawiyuga et al. in 2017 [23] recognized a token-based verification for using a JSON Web Token (JWT) worker as a validation worker. They proposed a framework design in which the client sends his/her username and secret word to the JWT authentication worker. At that point, the worker checks its information base for the legitimacy of the client certification. In the event that those actions are legitimate, the worker forwards the token to the client who preserves that token in his/her nearby storage. In the event that it is substantial, the Broker permits the client to allocate/purchase in to the required subjects [24]. The sequence of their framework is demonstrated in Figure 2.2.

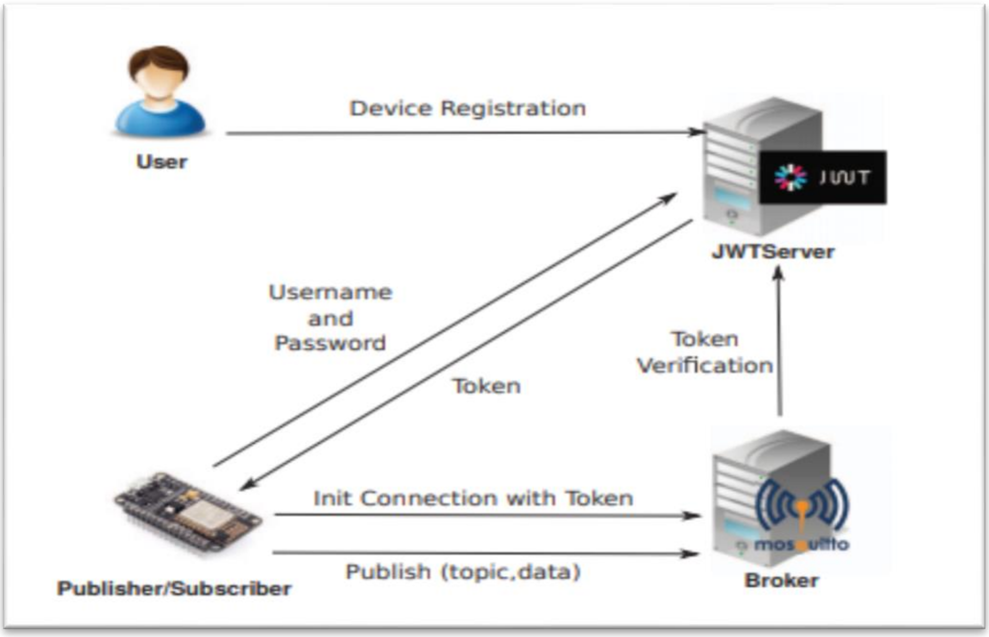


Figure 2. 2: Framework sequence - JWT

2.5.1.1 Working Procedure

It works following the sequence:

a. The client asks from Authentication server using its own security credentials for a token.

b. The broker delivers token upon authentication of identifications.

c. The client exploits the token in the linking with the Broker.

d. The Broker looks into the lawfulness by the token presented by Customer with validation worker.

e. Validity of the token of the Broker is replied by the authentication server.

f. For a lawful token the request is approved by the Broker

g. The customer begins to get to the subjects of the Broker.

2.5.1.2 Comments

The token-based algorithm creates less amount of message flow as compared to Non-Token based Procedure. It has Single Point of Failure and this may create an additional concern for development of such authentication procedure.

2.5.2 OAuth Approval Standards

OAuth is an open-sourced approval protocol that describes how different servers and services can safely be allowed to be allowed to get access to the assets without sharing the single logon details. Niruntasukrat et al. [25] introduced an approval instrument for MQTT utilizing OAuth 1.0a endorsement standard. They stated that since OAuth 2.0 [26] doesn't support any security above TLS/SSL, OAuth 1.0a is more rational for the IoT climate than OAuth 2.0. Their proposed component is introduced in Figure 4.2.

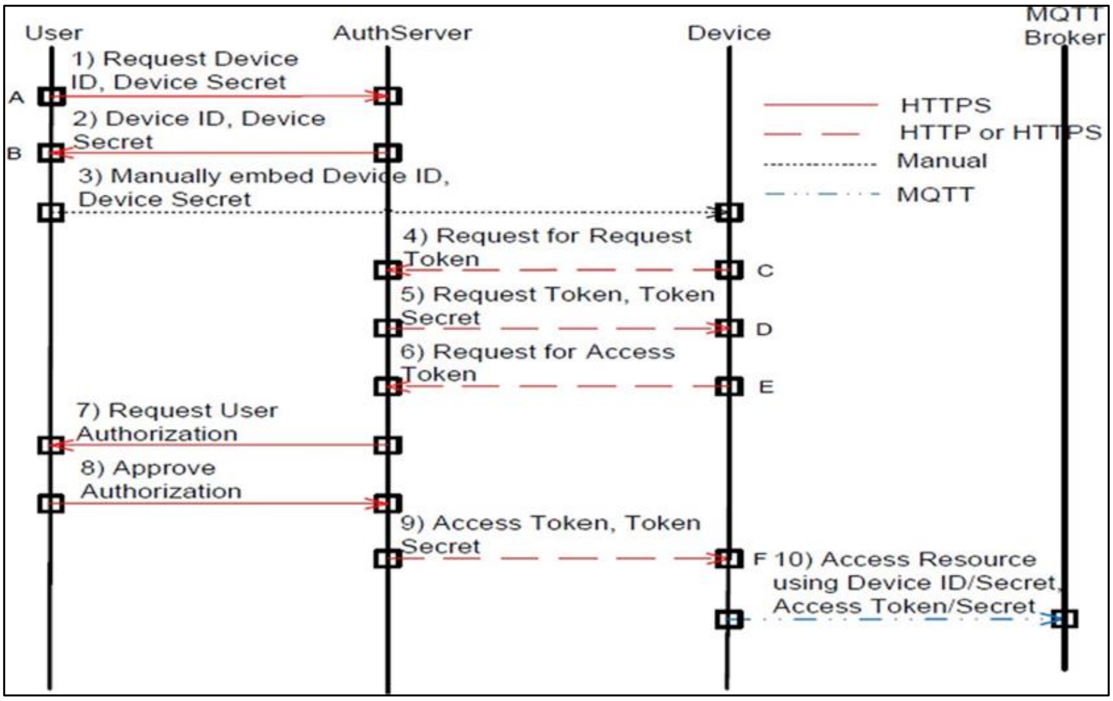

User
AuthServer
Device
MQTT Broker
1) Request Device ID, Device Secret
A
2) Device ID, Device Secret
B
3) Manually embed Device ID, Device Secret
HTTPS
HTTP or HTTPS
Manual
MQTT
4) Request for Request Token
C
5) Request Token, Token Secret
D
6) Request for Access Token
E
7) Request User Authorization
8) Approve Authorization
9) Access Token, Token Secret
F
10) Access Resource using Device ID/Secret, Access Token/Secret

Figure 2.3: Authorization Mechanism in MQTT System using OAuth Slandered

2.5.2.1 Working Procedure

It follows the following sequence [27]:

a. The HTTPS message is directed by user to the Auth Server to ask the Device ID and its security detgails.

b. Device credentials (ID and its secret) is allowed by the Auth server.

c. The Device technical details are stored in local memory is pushed by the user manually.

d. The device directs to the AuthServer to ask for a Invitation Token. This message is signed using the HMACSHA1 procedure where the HKey is the Device hidden.

e. A request Token and its security details is issued by AuthServer after checking the Device credentials.

f. The device sends to the AuthServer to request an Access Token. This communication is prudently marked using the HMAC SHA1 algorithm using HKey as the Request Token and the Device hidden details.

g. The AuthServer demands user endorsement utilizing email or SMS.

h. The user permits the Device ID and the permission possibility.

i. The AuthServer honors the Access Token and its details to the gadget.

j. The Device can contact to the MQTT Broker where the username is the Device ID with linked timestamp and the hidden detils will be shaped from the access mark. Here the Hkey will be the Access and the Device Secret Token [28].

2.5.2.2 Comments

It provides security over top of TLS/SSL. During the communication process, every session needs to be approved by user. This creates additional delay and overhead in the complete authentication process.

2.5.3 Attribute Based Encryption (KP/CP ABE)

Attribute-based encryption is a type of public-key encryption process in which the secret key and the encrypted text are relied upon their attributes (e.g. the country type of subscription etc.). For such kind of system, the decryption of a cipher text is possible only attributes of the user key matches the attributes of the cipher text. A vital security feature of attribute-based encryption is collision-proof: An opponent holding multiple keys should only be able to access data if minimum one individual key allows access permission.

Rahman et al. in 2018 [29] offered the use of Key using Policy/ Code using Policy Attribute Based Encryption (KP/CP ABE) utilizing ECC to get a changed MQTT convention fit for conveying secure correspondence between end gadgets. The arrangement chart of their proposed framework architecture is appeared in Figure. 4.3.

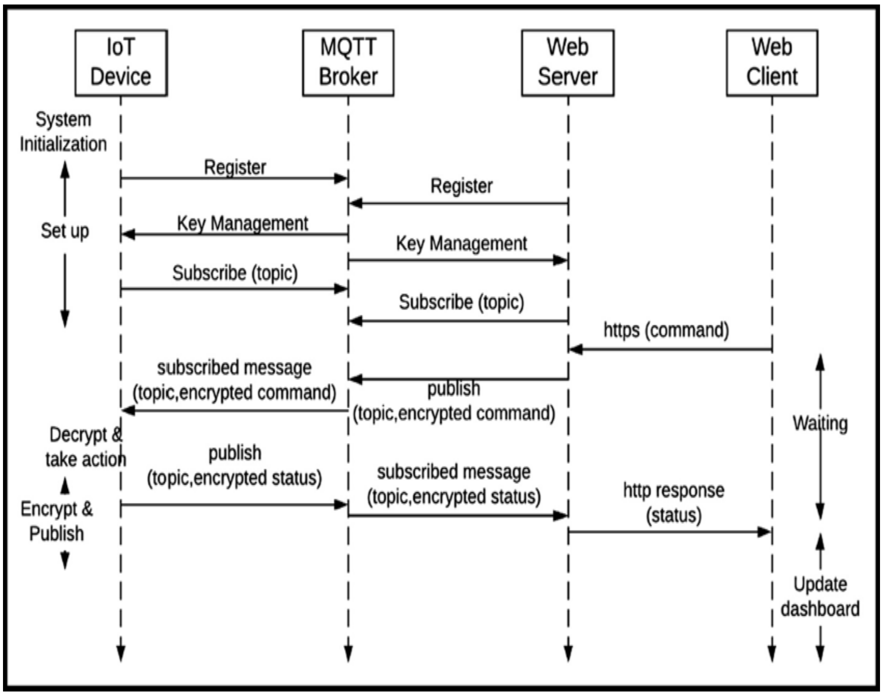


Figure 2.4: Sequence Diagram of System Architecture of ECC

2.5.3.1 Working Procedure

This design has the accompanying stages [30]:

a. After framework introduction, both the Hardware Device and the network based Workers will enlist in the Broker.
b. The key management stage is performed among the MQTT Broker, the IoT Device Hardware and the Web based Server.
c. Both the Device and the Web Server will buy in the required subjects of the Broker.
d. At the point when an approved customer sends an order to the Web Worker, it will encrypt the order and distribute it to the Broker.
e. The Broker will pass the encoded message to the Device where the decoding process will be done and necessary actions will be taken.
f. The device will encode the readied reaction and distribute the encoded message to the MQTT Broker.
g. MQTT Broker will send the encoded response to the Web Server.
h. Web Server will decipher the reaction.
i. Decoded reaction is conveyed to the customer.

2.5.3.2 Comments

This protocol provides security over top of TLS/SSL. During the process every session needs to be approved by user. The process is complex and time consuming.

2.5.4 Use of Lightweight Cryptography

Lightweight cryptography is an algorithm personalized for using in constrained surroundings that includes RFID labels, sensors, smart cards with contactless features, medical hardwares and so on. It also delivers satisfactory security. Bali et al. in 2019 [31] handled a lightweight instrument for confirmation in MQTT stages utilizing logical calculation with block cypher. A simulation model was introduced as shown in Fig. 4.4. Referenced proposed approach relies upon the huge variety of the tempestuous calculation and difficult algorithm. A safer framework is accomplished. Also, they explained that high variety between the backend keys by suitably choosing the fluctuating boundaries and they relies upon the distance entropy during selection of these MQTT based boundaries.

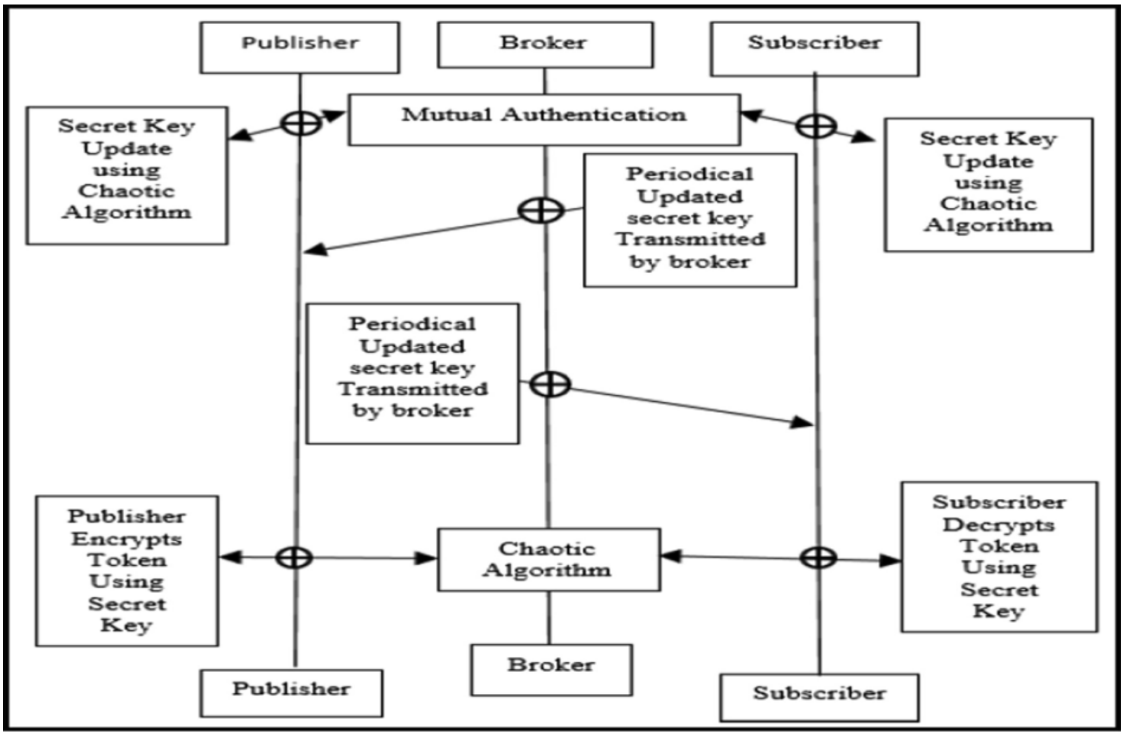

Publisher
Broker
Subscriber
Secret Key Update using Chaotic Algorithm
Mutual Authentication
Secret Key Update using Chaotic Algorithm
Periodical Updated secret key Transmitted by broker
Periodical Updated secret key Transmitted by broker
Publisher Encrypts Token Using Secret Key
Chaotic Algorithm
Subscriber Decrypts Token Using Secret Key
Broker
Publisher
Subscriber

Figure 2.5: System Architecture of Lightweight cryptography

In this process one to one publisher - subscriber relationship is considered which is viewed as under use of resources. Moreover, complex Public Key Cryptography is used during the process.

2.6 Summary of the Related Works

After conducting thorough analysis of the mentioned studies, it can be identified that each of them fulfills one or few aspects of the security requirement like authentication, confidentiality, data integrity, authorization etc.

IoT technology provides huge chances and brings many new tasks related to the verification of IoT devices. Using PINs or passwords have disadvantages that limit the use of IoT applications. Thus, validating users basing on password mechanism not only meet the purpose. Therefore, Token-Based Lightweight User Authentication (TBLUA) is commonly used in present days. It is based on token technique in order to improve the strength of verification, Tokens functions like a stamped ticket. The user holds the access as long validity of the remains. When the user logs out or quits an app, the token is overturned. It provides a second layer of security and administrators. They posses control over each deed and contract [32 -33].

Many social media platforms use the OAuth 1.0a method to act, or make API requests. But there are common errors like access token failure, matching of token, token expired, refusal of timestamp etc. There are other issues like signing every request, addressing native applications and separation of roles. This is eliminated in OAuth2.0 where multiple flows are presented deliberately [34].

Attribute-based encryption uses public-key encryption technique user's key and the cipher text are dependent upon attributes (e.g., residing country, or subscription types). The systems lacks mainly from two disadvantages: inefficiency and lack of attribute cancellation process. Other few challenges related to key are: organization, escrow, cancellation etc. [35].

Lightweight cryptography is to use of less memory, reduced computing resource and less power supply to facilitate resource-scared hardware resources. The lightweight cryptography is faster and quicker than orthodox cryptography [36].

However, since such cryptographic algorithms are designed to handle small amounts of information, they do not have high bandwidth. The very existence of constraints says that light ciphers primarily designed not to soft but to hardware implementation. The inherent disadvantage of lightweight cryptography is less secured [37-38].

CHAPTER THREE

METHODOLOGY

This chapter discusses the methodology being followed to obtain the expected outcome. In doing that, beside the common parameter (defining objective and Evaluation), there are few steps - Background study, Analyzing requirements, Identify Vulnerability and Designing of the customized protocols are the important steps. An overview of the smart cryptographic analyzer (ProVerif) is also highlighted at the last part of the chapter.

3.1 Key Phases of Research Methodology

An overview of the research methodology is presented in figure 3.1. This research is organized in three stages sequence to obtain the research objectives.

3.1.1 Identification of Research Questions

The research problem was identified and formulates the research questions through reviewing the existing literature. Exhaustive study is carried out on available research work found in open source. Major undertaken efforts of modifying this protocol were taken into consideration to analyze the requirements.

3.1.2 Designing Modified Protocol

Basing on the findings from the earlier step specific requirements was identified. Accordingly a modified protocol was defined and developed. During this process, every steps of authentication was analyzed and listed.

3.1.3 Analysis of the Modified Protocol

It is the last step of the research. Here, developed protocol is tested by a smart cryptographic analytical tool, ProVerif. Basing on the analytical result from ProVerif analysis, final outcome of the thesis is formulated. Accordingly the research outcome is evaluated basing on specific output of the compiler.

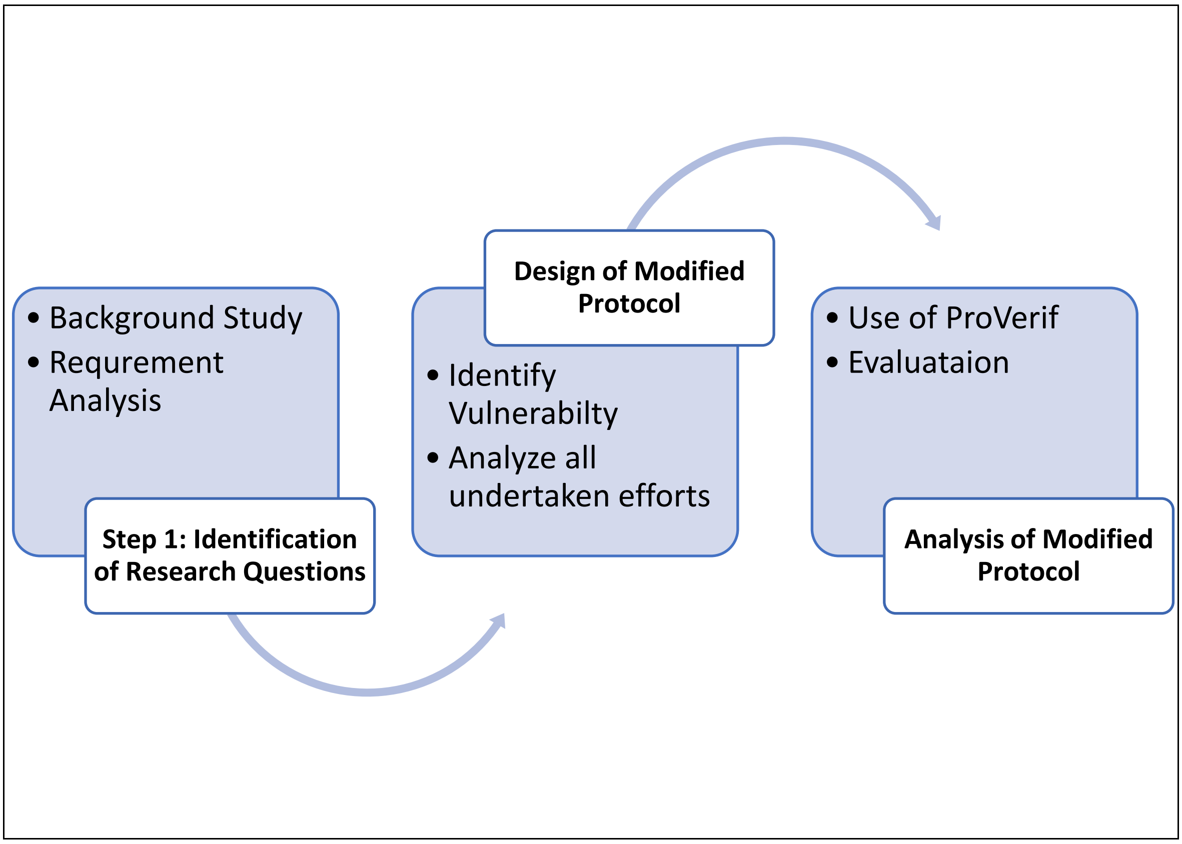

• Background Study
• Requrement Analysis
Step 1: Identification of Research Questions
Design of Modified Protocol
• Identify Vulnerabilty
• Analyze all undertaken efforts
• Use of ProVerif
• Evaluataion
Analysis of Modified Protocol

Figure 3.1: Steps followed in Research Methodology

3.2

3.2 Research Type

The gathered research articles and studies are largely experimental based. A good number of Handbooks and theoretical texts are also taken into consideration. Few practical oriented running systems are also referred during the research.

3.2 Reference Publication Year and Types

The study materials referred in this thesis followed systematic literature review within timeframe of 2014 to 2019. Majority of referenced materials comes from international journals. In analyzing IoT based protocols, it is observed that limited number of studies were undertaken to find the security vulnerability of MQTT protocols [39].

3.2 Steps Followed to use ProVerif

For our thesis, we followed few steps while using ProVerif ttols

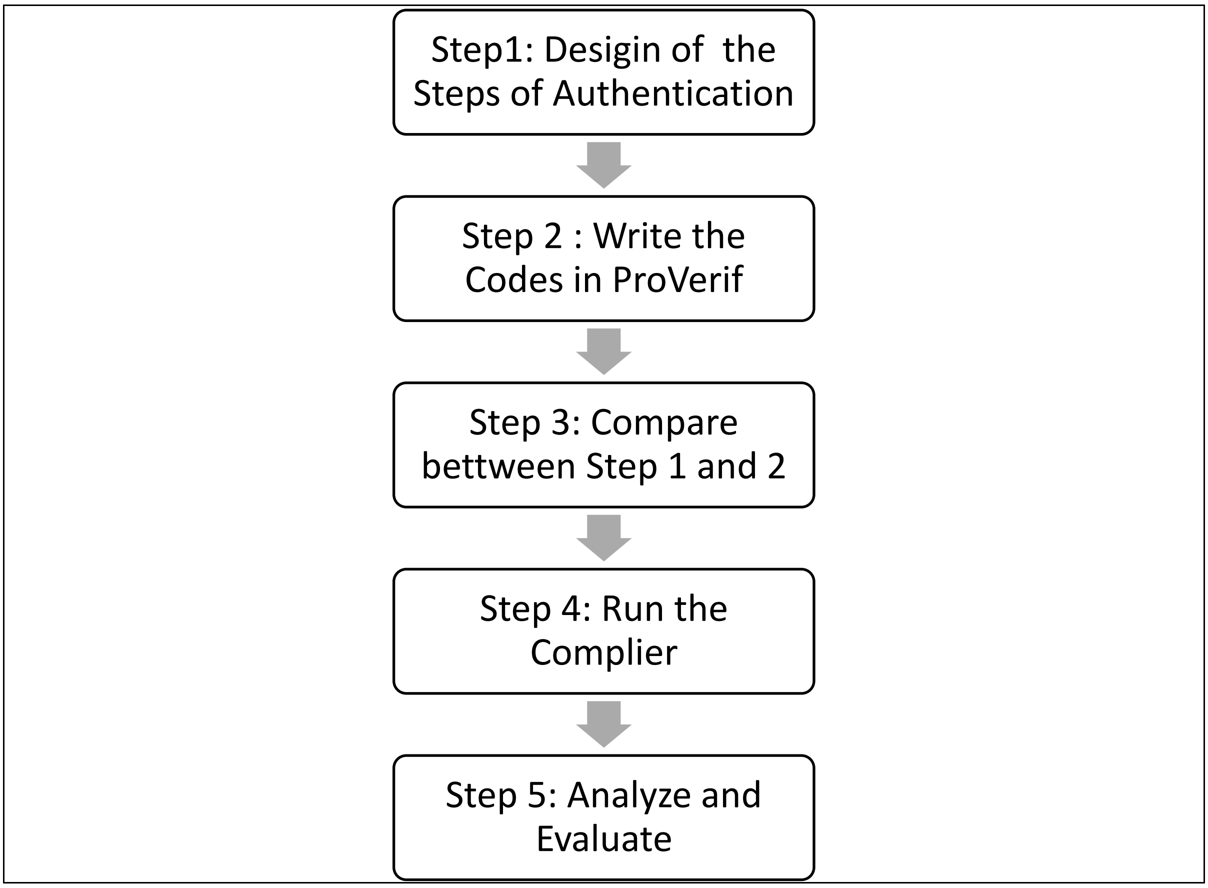

Step1: Desigin of the Steps of Authentication
Step 2 : Write the Codes in ProVerif
Step 3: Compare bettween Step 1 and 2
Step 4: Run the Complier
Step 5: Analyze and Evaluate

Figure 3.2: Sequence of Using ProVerif

The tool is capable of measuring reachability, good for sending communication and observational uniformity [40].

CHAPTER FOUR

PROPOSED SECURITY ENHANCED MQTT PROTOCOL

In this chapter, authentication procedure and security aspects of proposed protocols are discussed step by step. Beside all common elements of MQTT protocol, our proposed protocol has one additional hardware - Authentication severs (As in figure 4.1).

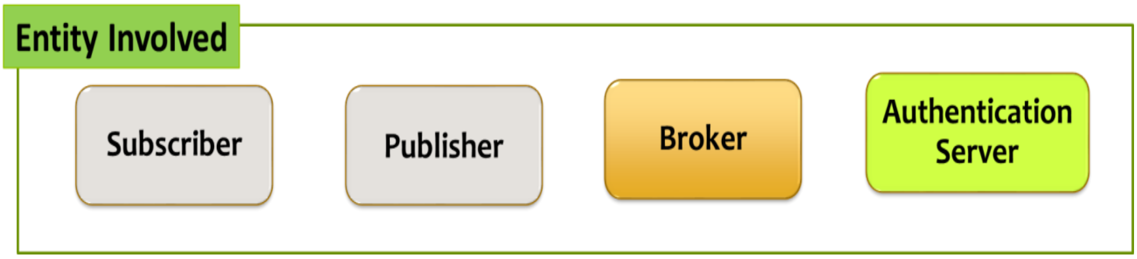


Figure 4.1: Various Entity of the Proposed Protocol

4.1 Authentication Server

An authentication server provides network oriented service to validate the identifications, usually account names and security details of their users. When a client submits a valid set of identifications, it obtains a cryptographic token which can subsequently be used to access different facilities. Authentication is used as the foundation for sanction, which says whether permission can be granted to a particular user or process. Inability to deny having some service that was authorized before to be done based on the this authentication [41].

4.2 Steps and Services

In proposed protocol, there are four measures of security and protections as shown in figure 4.2. Authentication pre-processing is the initial stage which creates a database for the potential users, facilitates in achieving mutual authentication, confidentiality protection and integrity protection. This steps are built in within subsequent steps.

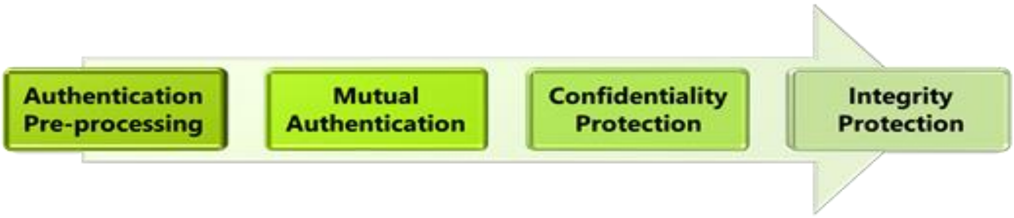


Figure 4.2: Security Features of Proposed Model

4.3 Authentication Pre-Processing Stage

The server shall permit and define roles of users (i.e. subscriber, broker or publisher). It generates unique ID and security credentials to respective users. That information is stored and shall be used for subsequent sessions by the respective users. These activities are termed as Pre-processing stage in our proposed scheme.

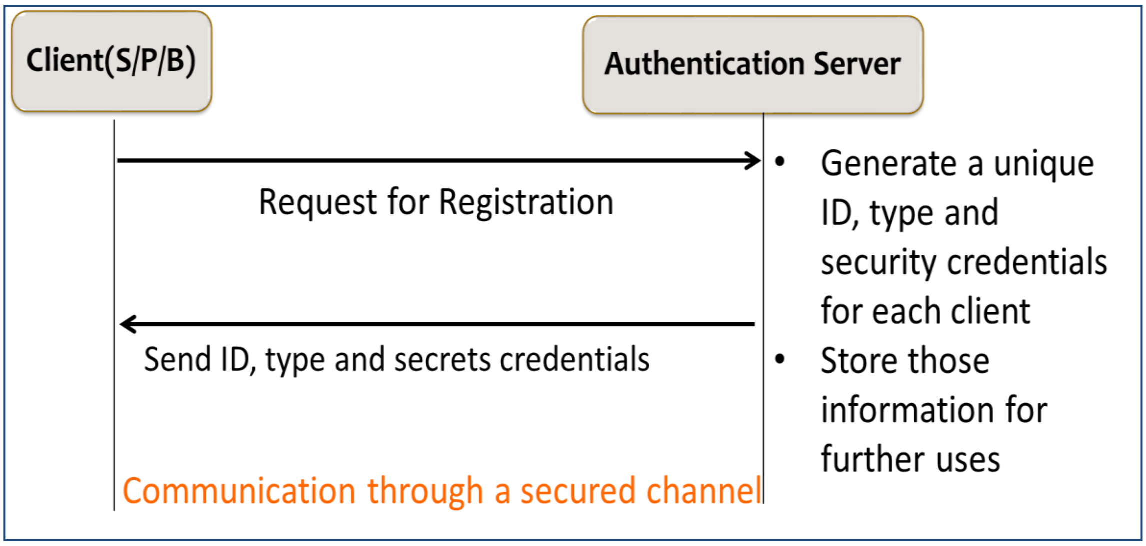

Client(S/P/B)
Authentication Server
Request for Registration
Send ID, type and secrets credentials
Communication through a secured channel
• Generate a unique ID, type and security credentials for each client
• Store those information for further uses

Figure 4.3: Pre-processing stage

4.4 Different Steps of Authentication

Once some client has achieved its required authenticity from pre-processing stage, it needs to establish separate session before starting data transfer. Generating a Random Number (R_u), using individual Client and Broker identity (U_{id}, B_{id}), separate encryption and decryption function ($E_{u.sec,c}$, $D_{u.sec.c}$), generating session key, and using Hash functions for integrity checking are the main activities of authentication. Here, is the sequence of events:

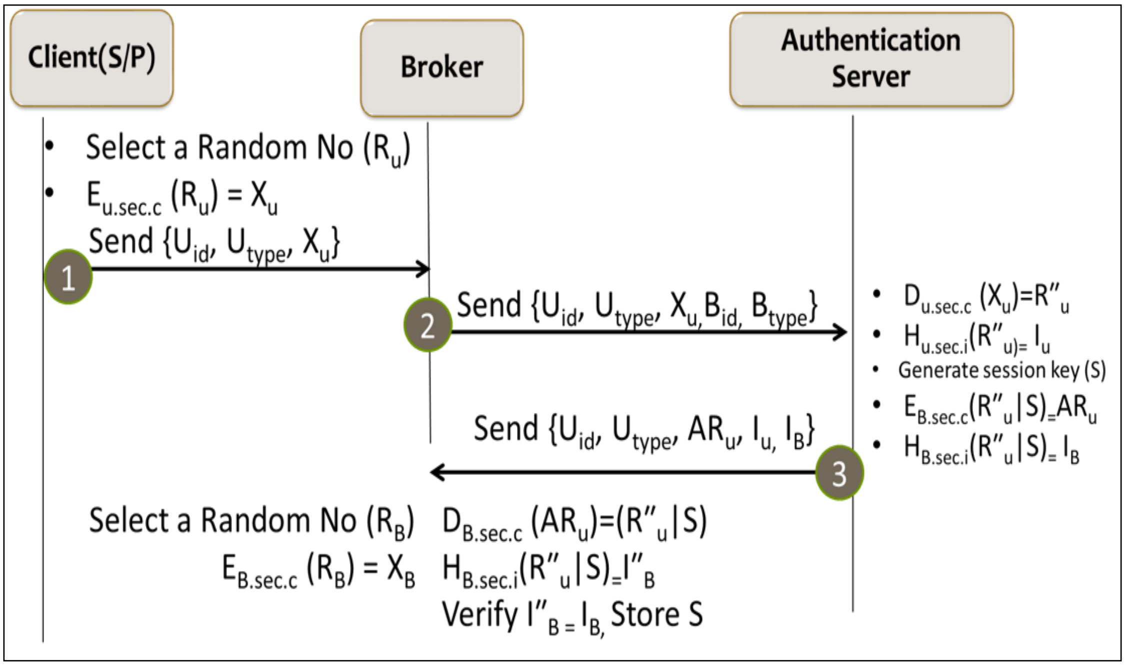

Client(S/P)
Broker
Authentication Server
• Select a Random No (R_u)
• $E_{u.sec.c}$ (R_u) = X_u
Send {U_{id}, U_{type}, X_u}
1
2
Send {U_{id}, U_{type}, $X_{u,}B_{id,}$ B_{type}}
• $D_{u.sec.c}$ (X_u)=R''_u
• $H_{u.sec.i}$($R''_{u)=}$ I_u
• Generate session key (S)
• $E_{B.sec.c}$(R''_u|S)$_=$$AR_u$
• $H_{B.sec.i}$(R''_u|S)$_=$ I_B
Send {U_{id}, U_{type}, AR_u, $I_{u,}$ I_B}
3
Select a Random No (R_B)
$E_{B.sec.c}$ (R_B) = X_B
$D_{B.sec.c}$ (AR_u)=(R''_u|S)
$H_{B.sec.i}$(R''_u|S)$_=$$I''_B$
Verify $I''_{B\,=}$ $I_{B,}$ Store S

Figure 4.4: Communication to Authentication Server through Broker

At the beginning, a client (i.e. a subscriber or a broker) generates a random number and send the encrypted value of the number along with its own appearance to the broker. The broker forwards the encrypted data, the client’s identity and its own credentials to the authentication server. The authentication server validates the identity and if all is well, t generates a corresponding session key and return to the broker (see Figure 4.4).

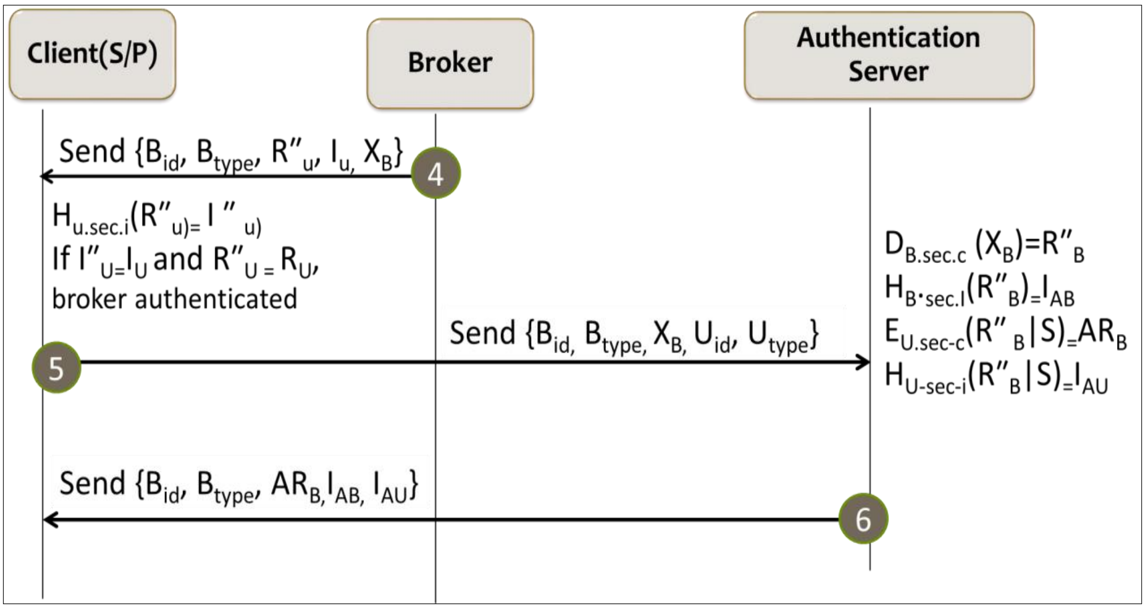

Client(S/P)
Broker
Authentication Server
Send $\{B_{id}, B_{type}, R''_{u}, I_{u,} X_{B}\}$
4
$H_{u.sec.i}(R''_{u)=} I''_{u)}$
If $I''_{U=}I_{U}$ and $R''_{U=}R_{U}$,
broker authenticated
Send $\{B_{id,} B_{type,} X_{B,} U_{id}, U_{type}\}$
5
$D_{B.sec.c}(X_{B})=R''_{B}$
$H_{B\cdot sec.I}(R''_{B})_{=}I_{AB}$
$E_{U.sec\text{-}c}(R''_{B}|S)_{=}AR_{B}$
$H_{U\text{-}sec\text{-}i}(R''_{B}|S)_{=}I_{AU}$
Send $\{B_{id}, B_{type}, AR_{B,}I_{AB,} I_{AU}\}$
6

Figure 4.5: Confirmation of Broker

Upon receiving the response to the challenge from the authentication server, the broker response to the client with the received response and initiates a similar challenge to the client (See Figure 4.5).

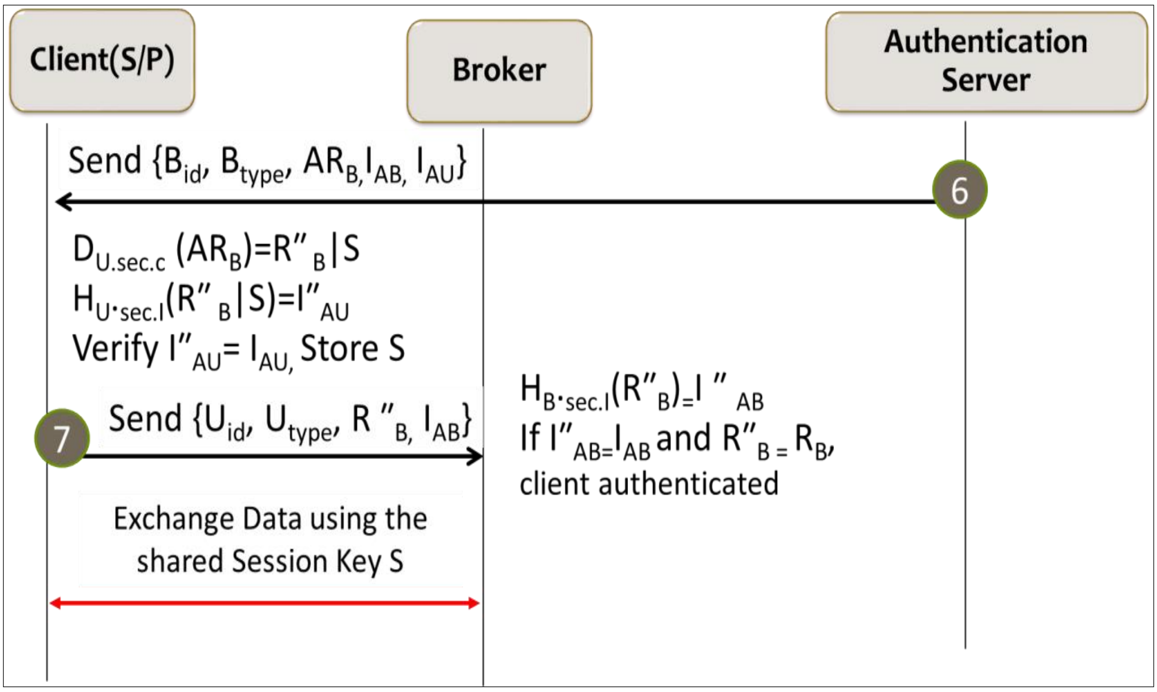

Client(S/P)
Broker
Authentication Server
Send $\{B_{id}, B_{type}, AR_{B,}I_{AB,} I_{AU}\}$
6
$D_{U.sec.c}(AR_{B})=R''_{B}|S$
$H_{U\cdot sec.I}(R''_{B}|S)=I''_{AU}$
Verify $I''_{AU}= I_{AU,}$ Store S
Send $\{U_{id}, U_{type}, R''_{B,} I_{AB}\}$
7
$H_{B\cdot sec.I}(R''_{B})_{=}I''_{AB}$
If $I''_{AB=}I_{AB}$ and $R''_{B=}R_{B}$,
client authenticated
Exchange Data using the shared Session Key S

Figure 4.6: Validating Broker

At the end of the challenge – response messages in both directions (client to broker and broker to client), both client and broker are mutually authenticated and establish security credential for ensuring security in sub sequent communications.

4.5 Key Features of the Proposed Protocol

- Challenge-response authentication scheme is applied for mutual authentication between subscriber / publisher and broker.
- Only symmetric key encryption is used.
- Each party receives its session key for data security during the successful Authentication.
- Authentication of the broker to the subscriber / publisher is introduced.
- All security management services are dedicated to the authentication server.
- Making broker to deal with only its original services.

CHAPTER FIVE

FORMAL MODELING FOR PROPOSED PROTOCOL

In this chapter, we first introduce the ProVerif software tools [42]. To verify the security attributes of the proposed protocol, we followed a formal process of verification. We defined all kind of declarations and variables, functions and process macros used in the complier. We describe the results of the analysis in the next chapter.

5.1 Procedure being followed to Use ProVerif

The tool is intended to be used to verify the secrecy and validation properties of a cryptographic protocol that continuously uses public communication channels. Since these channels are assumed to be controlled by a very authoritative setting, capturing an attacker with Dolev-Yao design model [43]. The simulation of the Dolev-Yao attacker model takes place in our analysis. Here, an attacker applies complete control over the communication channels; that is, the attacker may read, modify, delete, and insert messages. However, cryptography is assumed to be flawless; that is, the attacker is only able to perform cryptographic operations when in possession of the appropriate keys. In other words, it is limited to applying only the cryptographic orders specified by the user. The environment also captures the behavior of dishonest participants; hence, only honest participants need to be modeled in this tool. The tool can analyze protocols with boundless sessions using automated procedures.

To analyze the protocol, a ProVerif model and the security properties of interest are provided as input to the tool. The tool translates the input into a set of statements, known as Horn items, which are spontaneously provided to the tool's resolve algorithm. The resolve algorithm verifies the security properties using these Horn sections, and outputs an signal of whether the security properties hold for the input protocol model. If the tool can derive a fact in contradiction to a desired security property, it finds an attack and outputs the actions an attacker may take to break the security property; such an attack trace can be used to reconstruct the attack. However, when no fact contradicting the desired security property can be derived, the tool outputs that the security property holds.

This tool has been used to authenticate a versatile types of cryptographic protocols. For example, Smyth et al. [44] and Kremer, Ryan and Smyth [45] used it to analyze the security properties of a amount of e-voting protocols, and Delaune, Kremer and Ryan [46] used it to analyze the privacy properties of a similar set of protocols. Chen and Ryan used ProVerif to evaluate the authentication protocols employed by the trusted platform module (TPM), and discovered vulnerabilities. Abadi and Blanchet used ProVerif to verify the certified email protocol [47-48]. A team of Germany used ProVerif to verify the security properties in their `fixed' version of the AKA protocol [49]. Some of the scientists used ProVerif to verify the security stuffs of their privacy boosted verification and key agreement protocol for mobile networks.

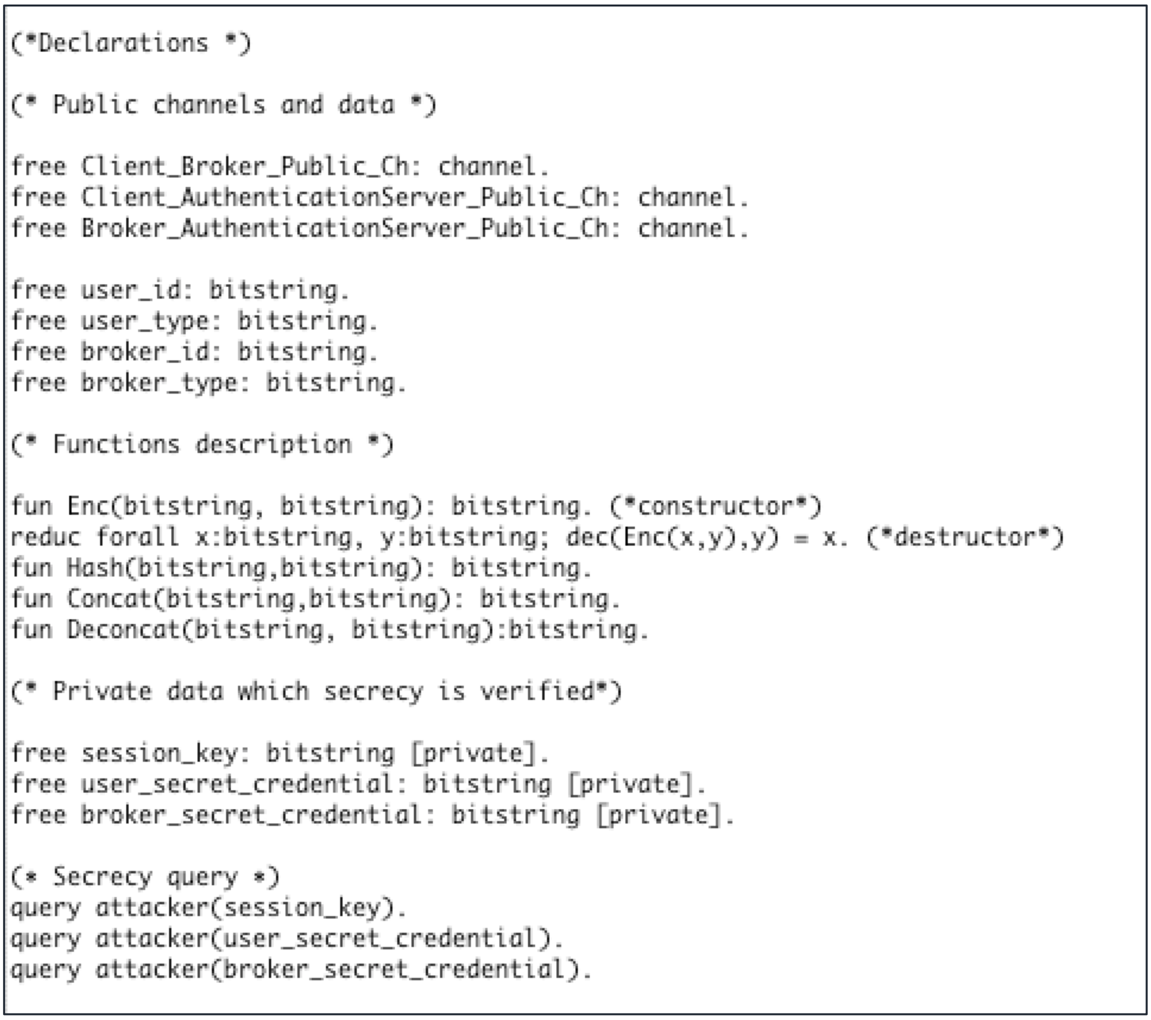
5.2 Model of the Proposed MQTT Scheme

Basic grammar of this language is described in the last part of this chapter. Further details of the grammar and syntax of the ProVerif language and its syntax can also be found in the ProVerif manual [50].The complier is divided into three parts: declarations, process macros, and main process. We next provide a summary of these three parts of the model. Full details are provided in Appendix A.

5.2.1 The Declarations

The declarations part includes a finite set of types and free variables, and formalizes the behavior of cryptographic primitives using a set of functions known as constructors, and corresponding rewrite rules known as destructors. Constructors are used to build terms used by a protocol, and take the form *fun f (t1 , ..., tn) : t* , where *f* is a constructor, *t* is its return type, and t_1,.....,t_n are the types of its arguments. In the syntax discussed here, n is always a positive integer. The term returned by a constructor can be a single variable. Destructors are used to manipulate terms formed by constructors, and take the form reduc forall x_1 : t_1,.....,x_n : t_n; g(M_1,.....,M_n) = M, where g is a destructor, and the terms M_1,.....,M_n and M are built from the application of constructors to variables x_1, ..., x_n of types t_1, ..., t_n, respectively. When an instance of the term g(M_1,.....,M_n) is encountered during execution, it is replaced by M.

Figure 6.1 contains the salient parts of the declarations for the proposed MQTT protocol. The full listing is given in Appendix. In figure 6.1, the Enc, dec, Hash, Concat, Deconcat constructors model the authentication-specific cryptographic functions, bitstring and channel are the ProVerif's built-in type. The constructor Enc models any standard 128-bit symmetric key encryption algorithm that could be used in the proposed scheme. To retrieve the encrypted data, the destructor dec is used. The constructor Hash generates a user-specific message authentication code using the user's secret credential. The constructor Concat and Deconcat simulates the behavior of string concatenation and retrieval of the intended string from the concatenated string.



```
(*Declarations *)

(* Public channels and data *)

free Client_Broker_Public_Ch: channel.
free Client_AuthenticationServer_Public_Ch: channel.
free Broker_AuthenticationServer_Public_Ch: channel.

free user_id: bitstring.
free user_type: bitstring.
free broker_id: bitstring.
free broker_type: bitstring.

(* Functions description *)

fun Enc(bitstring, bitstring): bitstring. (*constructor*)
reduc forall x:bitstring, y:bitstring; dec(Enc(x,y),y) = x. (*destructor*)
fun Hash(bitstring,bitstring): bitstring.
fun Concat(bitstring,bitstring): bitstring.
fun Deconcat(bitstring, bitstring):bitstring.

(* Private data which secrecy is verified*)

free session_key: bitstring [private].
free user_secret_credential: bitstring [private].
free broker_secret_credential: bitstring [private].

(* Secrecy query *)
query attacker(session_key).
query attacker(user_secret_credential).
query attacker(broker_secret_credential).
```

Figure 5.1: Summary of declaration - Enhanced MQTT Model

The declarations part also formalizes the security properties to be verified. We are interested in verifying the secrecy property of the session key used in the message transaction and the secret credentials of the client and the broker received earlier from the authentication server. The ProVerif tool verifies the secrecy of any term by proving the reachability property. The following query syntax is used to achieve our goal.

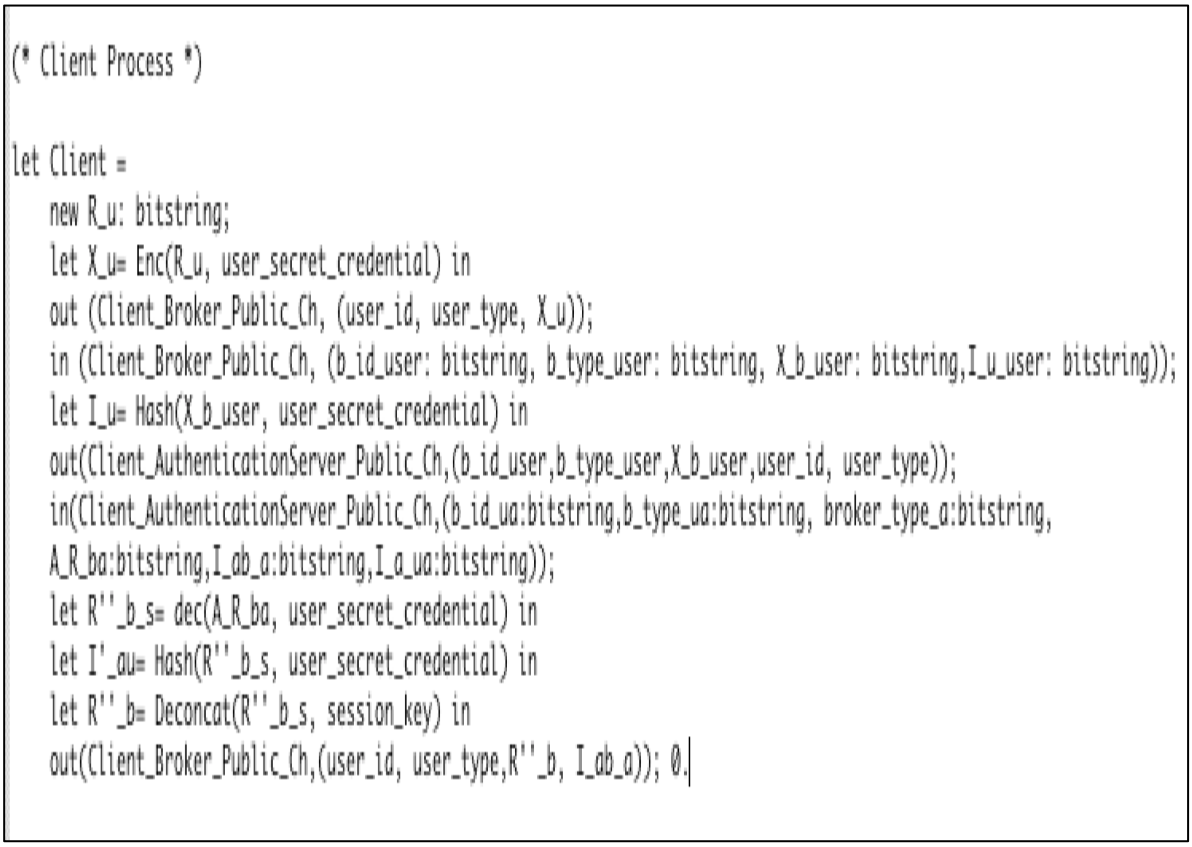
- Query attacker (M), queries the secrecy of the term M. If an attacker finds a way to learn M, the query fails. In other words, ProVerif attempts to verify that any state in which the term M is known to the adversary is unreachable.

- To demonstrate the privacy property of the proposed MQTT protocol, we use the query statement as query attacker (session_key), query attacker (user_secret_credential), and query attacker (broker_secret_credential).

We declare few free variables (session_key, user_secret_credential, and broker_secret_credential) of type bit string. Free variables are available to the attacker unless they are declared private by appending [private]. Since we are interested in verifying the secrecy of this variable, we declare the variable as private. Later, we query the security properties of these variables for the proposed scheme.

5.2.2 The Process Macros

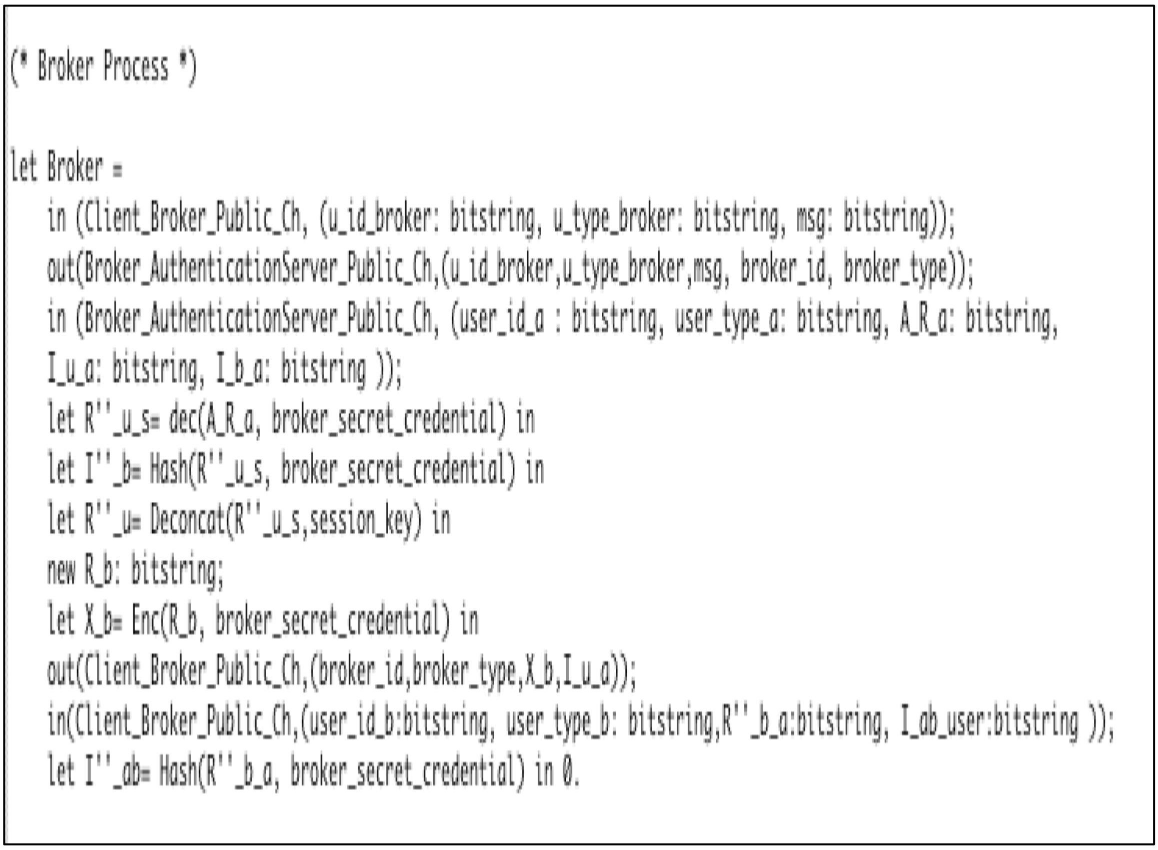
Process macros are sub-processes defined in order to ease development. Macros take the form let $R(x_1 : t_1, ..., x_n : t_n) = P$, where R is the macro name, P is the sub-process being defined, and $x_1,.....,x_n$ of types $t_1,.....,t_n$ respectively are the free variables of P. We define three process macros to model the proposed MQTT protocol, namely Client, Broker, and Auth.



```
(* Client Process *)

let Client =
    new R_u: bitstring;
    let X_u= Enc(R_u, user_secret_credential) in
    out (Client_Broker_Public_Ch, (user_id, user_type, X_u));
    in (Client_Broker_Public_Ch, (b_id_user: bitstring, b_type_user: bitstring, X_b_user: bitstring,I_u_user: bitstring));
    let I_u= Hash(X_b_user, user_secret_credential) in
    out(Client_AuthenticationServer_Public_Ch,(b_id_user,b_type_user,X_b_user,user_id, user_type));
    in(Client_AuthenticationServer_Public_Ch,(b_id_ua:bitstring,b_type_ua:bitstring, broker_type_a:bitstring,
    A_R_ba:bitstring,I_ab_a:bitstring,I_a_ua:bitstring));
    let R''_b_s= dec(A_R_ba, user_secret_credential) in
    let I'_au= Hash(R''_b_s, user_secret_credential) in
    let R''_b= Deconcat(R''_b_s, session_key) in
    out(Client_Broker_Public_Ch,(user_id, user_type,R''_b, I_ab_a)); 0.
```

Figure 5.2: Client Process - Enhanced MQTT Model

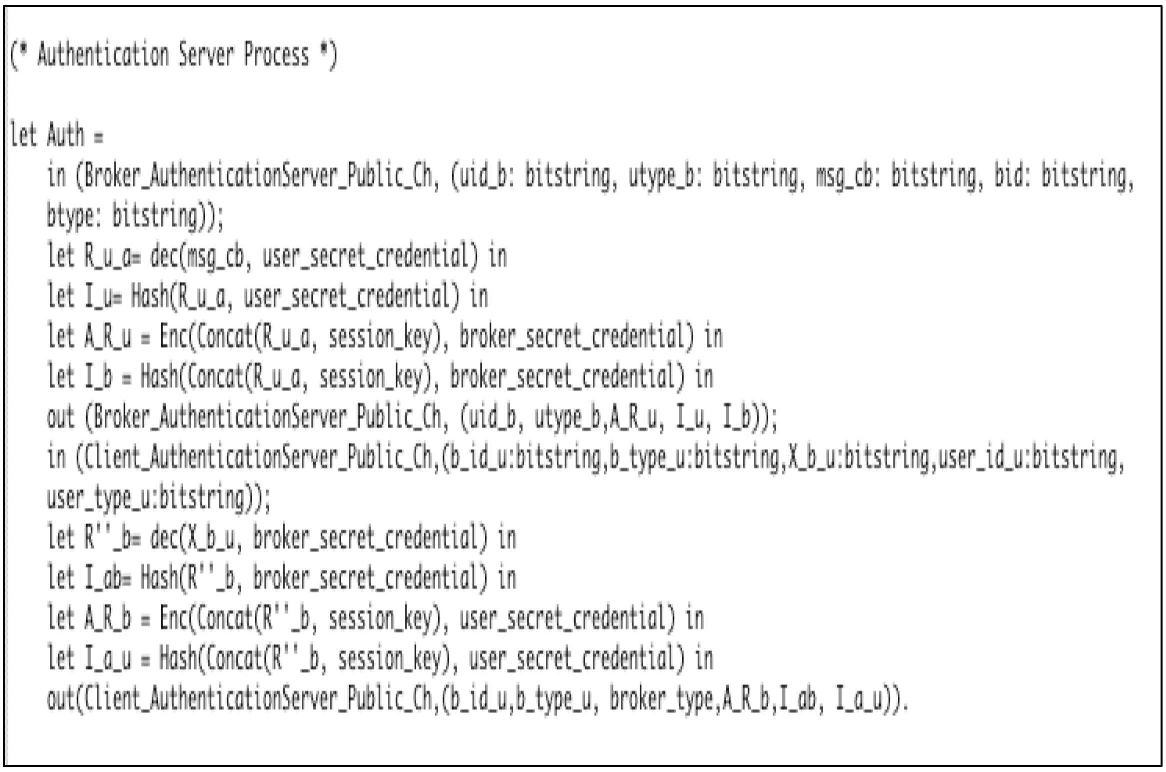
We assume that the Client and Broker communicate with each other through a public channel (the Client_Broker_Public_Ch variable in the model). The communication channel between the Client and Authentication Server (the Client_AuthenticationServer_Public_Ch variable in the model) and the communication channel between the Broker and Authentication Server (the Broker_AuthenticationServer_Public_Ch variable in the model) are also public. The receipt of a message in a process is represented by in(c, x), where c is the communication channel and x is the received message. Similarly, sending a message is represented by out(c,y), where c is the communication channel and y is the sent message. A destructor application of the form let M = D in P else Q tries to rewrite D and matches the result with M; if this succeeds, then the variables in M are instantiated accordingly and P is executed; otherwise, Q is executed. The conditional construct, if M = N then P else Q, checks the equality of two terms M and N, and then behaves as P or Q accordingly. We omit the else branch of a let or a conditional, when the process Q is 0, meaning a null process.



```
(* Broker Process *)

let Broker =
    in (Client_Broker_Public_Ch, (u_id_broker: bitstring, u_type_broker: bitstring, msg: bitstring));
    out(Broker_AuthenticationServer_Public_Ch,(u_id_broker,u_type_broker,msg, broker_id, broker_type));
    in (Broker_AuthenticationServer_Public_Ch, (user_id_a : bitstring, user_type_a: bitstring, A_R_a: bitstring,
    I_u_a: bitstring, I_b_a: bitstring ));
    let R''_u_s= dec(A_R_a, broker_secret_credential) in
    let I''_b= Hash(R''_u_s, broker_secret_credential) in
    let R''_u= Deconcat(R''_u_s,session_key) in
    new R_b: bitstring;
    let X_b= Enc(R_b, broker_secret_credential) in
    out(Client_Broker_Public_Ch,(broker_id,broker_type,X_b,I_u_a));
    in(Client_Broker_Public_Ch,(user_id_b:bitstring, user_type_b: bitstring,R''_b_a:bitstring, I_ab_user:bitstring ));
    let I''_ab= Hash(R''_b_a, broker_secret_credential) in 0.
```

Figure 5.3: Broker Process - Enhanced MQTT Model

The details of the respective process macros, i.e. the Client, Broker, and the Auth, simulating the activities of the proposed MQTT protocol entities the Client, Broker, and the Authentication Server are shown in figure 6.2, 6.3, and 6.4 respectively.

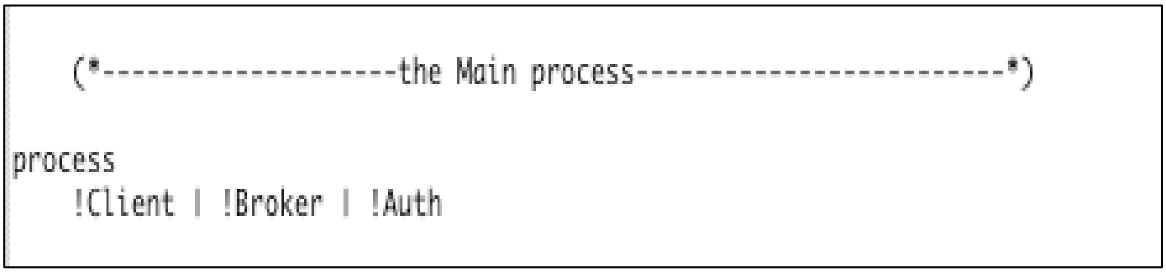


```
(* Authentication Server Process *)

let Auth =
    in (Broker_AuthenticationServer_Public_Ch, (uid_b: bitstring, utype_b: bitstring, msg_cb: bitstring, bid: bitstring,
    btype: bitstring));
    let R_u_a= dec(msg_cb, user_secret_credential) in
    let I_u= Hash(R_u_a, user_secret_credential) in
    let A_R_u = Enc(Concat(R_u_a, session_key), broker_secret_credential) in
    let I_b = Hash(Concat(R_u_a, session_key), broker_secret_credential) in
    out (Broker_AuthenticationServer_Public_Ch, (uid_b, utype_b,A_R_u, I_u, I_b));
    in (Client_AuthenticationServer_Public_Ch,(b_id_u:bitstring,b_type_u:bitstring,X_b_u:bitstring,user_id_u:bitstring,
    user_type_u:bitstring));
    let R''_b= dec(X_b_u, broker_secret_credential) in
    let I_ab= Hash(R''_b, broker_secret_credential) in
    let A_R_b = Enc(Concat(R''_b, session_key), user_secret_credential) in
    let I_a_u = Hash(Concat(R''_b, session_key), user_secret_credential) in
    out(Client_AuthenticationServer_Public_Ch,(b_id_u,b_type_u, broker_type,A_R_b,I_ab, I_a_u)).
```

Figure 5.4: Authentication Server Process - Enhanced MQTT Model

5.2.3 The Main Process

The main process of the compiler model encodes the complete protocol. We encode the proposed MQTT protocol using the macros defined in the previous section. We model the execution of unbounded number of client processes as (!Client), where the symbol '!' represents replication and instantiates the parallel execution of an unbounded number of copies of Client, in parallel to the Broker and Authentication Server processes. The parallel execution of any processes P and Q is represented as P|Q. Figure 6.5 shows the main process that embodies the proposed MQTT protocol. A full listing of the ProVerif code can be found in Appendix.



```
    (*--------------------the Main process-------------------------*)

process
    !Client | !Broker | !Auth
```

Figure 5.5: The Main process (Call up functions) - Enhanced MQTT Model

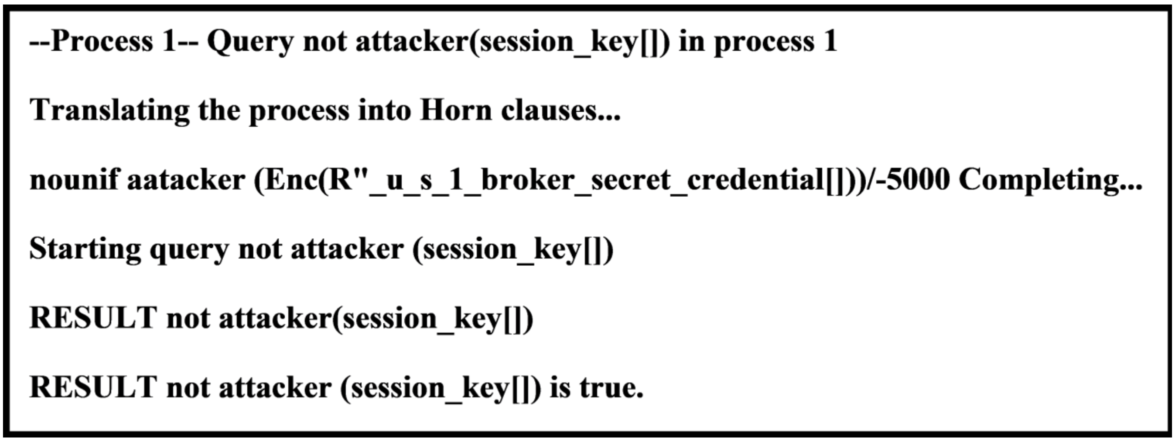
CHAPTER SIX

RESULT AND PERFORMANCE EVALUATION

In this chapter we present the simulation result of the model described in chapter Five and also discuss on the performance and security issues of the proposed modified MQTT protocol.

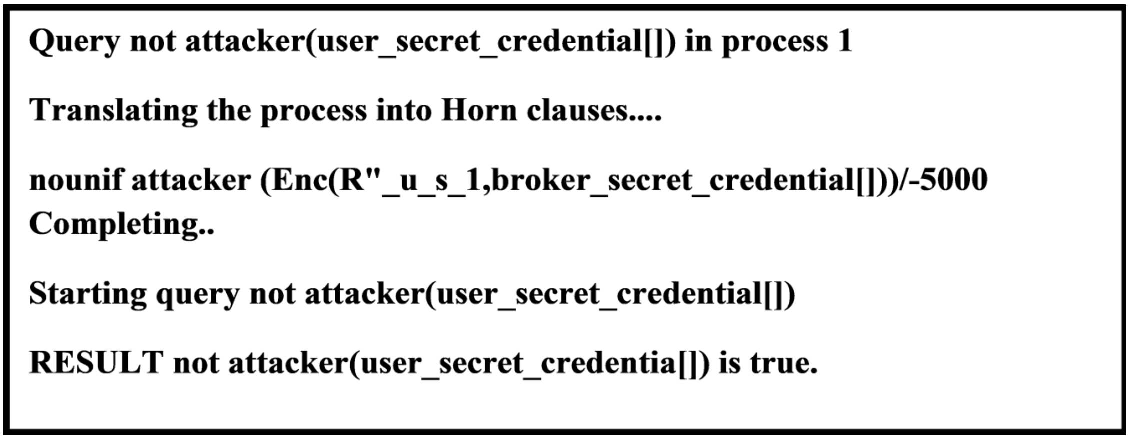
6.1 ProVerif Verification Result

We ran the encoded protocol described in Chapter 5 to verify the secrecy and validity properties. The tool also output RESULT not attacker(user_secret_credential) and RESULT not attacker(broker_secret_credential) are true, which means that both the user and the broker's credentials used in the proposed MQTT protocol are not compromised to any adversary. Hence the ProVerif verification proves that the secrecy of session key, user and broker credentials are maintained in the proposed MQTT protocol. The details of the verification result are shown in figure 6.1, 6.2, and 6.3.



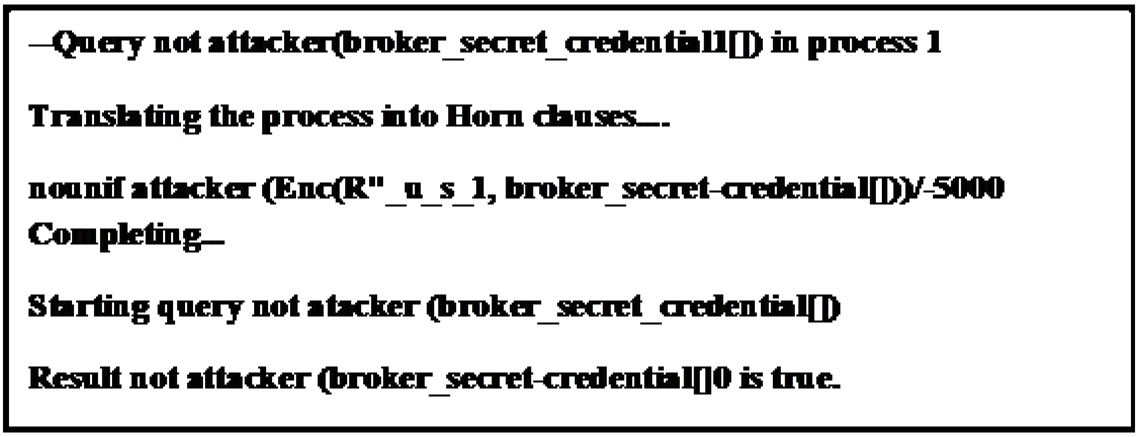
```
--Process 1-- Query not attacker(session_key[]) in process 1

Translating the process into Horn clauses...

nounif aatacker (Enc(R"_u_s_1_broker_secret_credential[]))/-5000 Completing...

Starting query not attacker (session_key[])

RESULT not attacker(session_key[])

RESULT not attacker (session_key[]) is true.
```

Figure 6.1: The proposed MQTT protocol verification result - Session Key



```
Query not attacker(user_secret_credential[]) in process 1

Translating the process into Horn clauses....

nounif attacker (Enc(R"_u_s_1,broker_secret_credential[]))/-5000
Completing..

Starting query not attacker(user_secret_credential[])

RESULT not attacker(user_secret_credentia[]) is true.
```

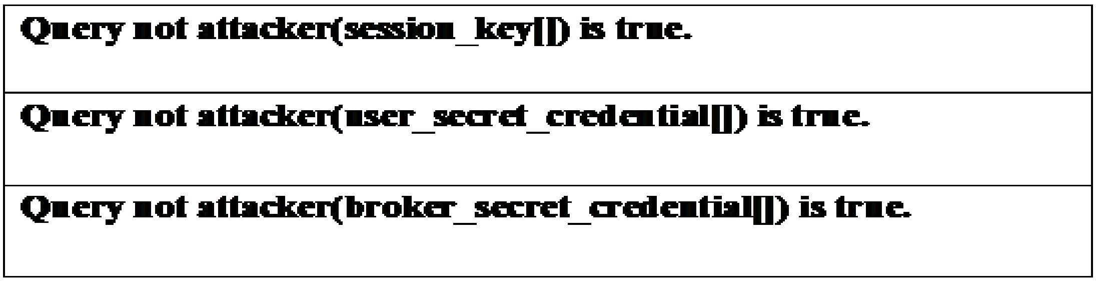
Figure 6.2: The proposed MQTT protocol verification result -User credential secrecy



```
—Query not attacker(broker_secret_credential1[]) in process 1

Translating the process into Horn clauses....

nounif attacker (Enc(R"_u_s_1, broker_secret-credential[]))/-5000
Completing...

Starting query not atacker (broker_secret_credential[])

Result not attacker (broker_secret-credential[]0 is true.
```

Figure 6.3: The proposed MQTT protocol verification result - Broker credential secrecy

The protocol verification summary is as follow.



```
Query not attacker(session_key[]) is true.

Query not attacker(user_secret_credential[]) is true.

Query not attacker(broker_secret_credential[]) is true.
```

Figure 6.4: The proposed MQTT protocol – Final Outcome

This means that an opponent is not being able to contact the cryptographic keys that will be used in the sessions tailed by the fruitful verification. It also proves that the security credentials that are used by the publisher, subscriber, or broker are also not accessible by any adversary in the proposed security enhanced MQTT protocol. It validates the intended security of the proposed MQTT protocol.

6.2 Analysis of the Proposed MQTT Protocol

6.2.1 Use of Symmetric Key Cryptography

The proposed modified MQTT protocol makes use of the symmetric key cryptography only. As we know that the symmetric key cryptography is light weight, it would be suitable for the IoT environment. Moreover, symmetric key cryptography is faster in compare to the other form of cryptography and so it would be an added advantage for the concern deployment environment.

6.2.2 Inclusion of Broker Authentication

The fundamental assumption in the original MQTT protocol is that the broker is considered as a trustworthy entity. However, the general assumption that the broker to which the subscribers and the publishers are connected for exchange of information are trustworthy may not true in several context.

The proposed modified MQTT protocol does not assume broker as a trustworthy entity in MQTT and hence, introduce the authentication of MQTT broker to the subscriber and to the publisher.

6.2.3 Mutual Authentication between Broker and Publisher

The proposed protocol initiates a parameter by sending a challenge among the publisher and the broker. The broker with the support of authentication center responses to the challenge and initiate a new challenge to the publisher. The publisher in the same way response to the challenge. If both of the challenges are addressed positively, then the authentication is dimmed complete. Thus, mutual authentication between broker and publisher (in the original MQTT context) is ensured in the proposed scheme.

6.2.4 Mutual Authentication between Broker and Subscriber

In the similar way, as discussed in Section 7.2.3, the mutual authentication between subscriber and broker is introduced in the proposed scheme.

6.2.5 Authentication with Key Distribution

In the proposed scheme, when two entities, for example, a broker and a publisher/subscriber initiate an authentication procedure, both of them communicates with the authentication server to validates each other. Since, authentication server has contact with both the parties involved in the authentication, it also shares the symmetric keys (confidentiality and integrity key) for the concern session if the authentication is successful.

CHAPTER SEVEN

CONCLUSION AND FUTURE WORK

7.1 Conclusion

IoT is a distinct dimension applied to several areas ranging from medication to industrialized systems. IoT connected devices worldwide is likely to be increased in geometric rate in near future. However, one of the major challenges in this area consists of proper mechanisms and protocols that meet adequate security. Particularly, the MQTT protocol has been prospected in order to prove its efficiency and performance for the communication at the application layer. IoT developers prefer this protocol because of its overriding advantages. The MQTT protocol only provides verification for security appliance and it does not encode the data in transfer period. Therefore, data confidentiality, validation, and reliability turn out to be important in MQTT implementation.

In this thesis, we look into the security analysis of this protocol and tried to propose a security improved MQTT protocol. The proposed protocol is based with added cryptographic features to offer security facilities for IoT system. The fundamental deviation from the original MQTT protocol is that the broker is considered as a trustworthy entity. However, the general assumption that the brokers to whom the subscribers and the publishers are connected for exchange of information are trustworthy may not true in several context. Mutual authentication between subscriber and broker, mutual authentication between publisher and broker, authentication with key distribution, use of only symmetric key cryptography are the few salient features of the proposed enhanced MQTT protocol. This work also conducts a formal verification for the proposed MQTT protocol to prove that the proposed protocol fulfills the projected security qualities. The ProVerif result validates that the proposed protocol ensures the secrecy property of the cryptographic credentials and hence, operates securely.

7.2 Limitations and Future Work

One of the major limitations of the thesis is that the testing and analysis is completely carried out in a laboratory environment. The result of the findings could have been much more trustworthy, if it could be tested or launched in a real life or practical area of works. Total works could also be verified any other second tools to justify the result more realistic way.

Considering present trend and diversity, the simulation can be conducted in a cloud based environment also. This would be more practically viable to meet future requirements.

REFERENCES

[1] S. R. J. Ramson, S. Vishnu, and M. Shanmugam, “Applications of Internet of Things (IoT) – An Overview,” in 2020 5th International Conference on Devices, Circuits and Systems (ICDCS), Coimbatore, India, Mar. 2020, pp. 92–95, doi: 10.1109/ICDCS48716.2020.243556.

[2] P. Fraga-Lamas, T. Fernández-Caramés, M. Suárez-Albela, L. Castedo, and M. González-López, “A Review on Internet of Things for Defense and Public Safety,” Sensors, vol. 16, no. 10, p. 1644, Oct. 2016, doi: 10.3390/s16101644.

[3] V. Hassija, V. Chamola, V. Saxena, D. Jain, P. Goyal, and B. Sikdar, “A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures,” IEEE Access, vol. 7, pp. 82721–82743, 2019, doi: 10.1109/ACCESS.2019.2924045.

[4] M. Singh, M. A. Rajan, V. L. Shivraj and P. Balamuralidhar, "Secure MQTT for Internet of Things (IoT)," 2015 Fifth International Conference on Communication Systems and Network Technologies, Gwalior, 2015, pp. 746-751.

[5] R.S Bali, F. Jaafar, P. Zavarasky. “Lightweight authentication for MQTT to improve the security of IoT communication.” Proceedings of the 3rd International Conference on Cryptography, Security and Privacy - ICCSP ’19, Kuala Lumpur, Malaysia; 2019. p. 6–12.

[6] E. Elemam, A. M. Bahaa-Eldin, N. H. Shaker, and M. Sobh, “Formal verification for a PMQTT protocol,” Egyptian Informatics Journal, vol. 21, no. 3, pp. 169–182, Sep. 2020, doi: 10.1016/j.eij.2020.01.001.

[7] A.,Niruntasukrat Issariyapat, P. Pongpaibool , K. Meesublak, P. Aiumsupucgul, A. Panya. “Authorization mechanism for MQTT-based Internet of Things”. In: 2016 IEEE International Conference on Communications Workshops (ICC), Kuala Lumpur, Malaysia; 2016, pp. 290–295.

[8] MQTT Version 5. [Online]. Available: https://docs.oasis-open.org/mqtt/mqtt/ v5.0/mqtt-v5.0.html. [Accessed: 26-Aug-2019].

[9] M. K,. Chaudhry S.A, Naqvi H, S. Kumari, , A.K Sangaiah “An elliptic curve cryptography based lightweight authentication scheme for smart grid communication” Future Generation Computer System 2018; 81:557–65.

[10] B. Blanchet, B. Smyth, V. Cheval, M.Sylvestre “ProVerif 2.00: Automatic Cryptographic Protocol Verifier, User Manual and Tutorial. Fig. 20. The Results for the Formal Verification of the Session Key Secrecy and the Encrypted Message Secrecy” E. Elemam et al. / Egyptian Informatics Journal 21 (2020) 169–182 181

[11] A. Bhawiyuga, M. Data, A. Warda “Architectural design of token-based authentication of MQTT protocol in constrained IoT device” In: 2017 11th International Conference on Telecommunication Systems Services and Applications (TSSA), Lombok, 2017, pp. 1–4.

[12] JSON Web Token (JWT). [Online]. Available: https://tools.ietf.org/html/ rfc7519. [Accessed: 26-Aug-2019].

[13] A. Rahman, S. Roy , MS. Kaiser, MDS. slam “ A Lightweight Multi-Tier S-MQTT Framework to Secure Communication between low-end IoT Nodes” In: 2018 5th International Conference on Networking, Systems and Security (NSysS), Dhaka, Bangladesh; 2018. p. 1–6.

[14] K. T. Nguyen, M. Laurent and N. Oualha, "Survey on secure communication protocols for the Internet of Things", Ad Hoc Netw., vol. 32, pp. 17-31, Sep. 2015.

[15] M. A. Ferrag, L. A. Maglaras, H. Janicke, J. Jiang and L. Shu, "Authentication Protocols for Internet of Things: A Comprehensive Survey", Security and Communication Networks, vol. 2017.

[16] L. Xiong, D. Peng, T. Peng, H. Liang and Z. Liu, "A lightweight anonymous authentication protocol with perfect forward secrecy for wireless sensor networks", Sensors, vol. 17, no. 11, pp. 2681:1-28, 2017.

[17] C. C. Chang and H. D. Le, "A Provably Secure Efficient and Flexible Authentication Scheme for Ad hoc Wireless Sensor Networks", IEEE Transactions on Wireless Communications, vol. 15, no. 1, pp. 357-366, 2016.

[18] P. Gope and T. Hwang, "A Realistic Lightweight Anonymous Authentication Protocol for Securing Real-Time Application Data Access in Wireless Sensor Networks", IEEE Transactions on Industrial Electronics, vol. 63, no. 11, pp. 7124-7132, Nov. 2016.

[19] D. Evans, “The Internet of things: how the next evolution of the Internet is changing everything,” Cisco Internet Business Solution Group White Paper, April 2011.

[20] Gartner. (2017, February 7). Gartner Says 8.4 Billion Connected "Things" Will Be in Use in 2017, Up 31 Percent From 2016. Available: http://www.gartner.com/newsroom/

[21] A. Niruntasukrat, C. Issariyapat, P. Pongpaibool, K. Meesublak, P. Aiumsupucgul and A. Panya, "Authorization mechanism for MQTT_based Internet of Things," 2016

IEEE International Conference on Communications Workshops (ICC), pp. 290-295, 2016.

[22] D. H. Mun, M. L. Dinh and Y. W. Kwon, "An Assessment of Internet of Things Protocols for Resource-Constrained Applications," 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), pp. 555-560, 2016.

[23] S. Alasmari and M. Anwar, "Security & Privacy Challenges in IoT_based Health Cloud," International Conference on Computational Science and Computational Intelligence, pp. 198-201, 2016.

[24] E. Caltum, and O. Segal, "Exploitation of IoT devices for Launching Mass-Scale Attack Campaigns," Akamai Threat Research, October 2016. [14] C. Bormann, M. Ersue, and A. Keranen, "RFC 7228 Terminology for Constrained-Node Networks," IETF, May 2014

[25] L. Markowsky and G. Markowsky, "Scanning for Vulnerable Devices in the Internet of Things," The 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing System: Technology and Applications, September 2015

[26] J. S. Kumar and D. R. Patel, "A survey on internet of things: Security and privacy issues," International Journal of Computer Applications, vol. 90, no. 11, 2014

[27] M. Abomhara and G. M. Køien, "Security and privacy in the internet of things: Current status and open issues," in Privacy and Security in Mobile Systems (PRISMS), International Conference on. IEEE, 2014, pp. 1–8.

[28] S. Chen, H. Xu, D. Liu, B. Hu, and H. Wang, “A vision of iot: Applications, challenges, and opportunities with china perspective,” IEEE Internet of Things journal, vol. 1, no. 4, pp. 349–359, 2014

[29] L. Atzori, A. Iera, and G. Morabito, “The internet of things: A survey,” Computer. Network., vol. 54, no. 15, pp. 2787–2805, Oct 2010

[30] M. M. Hossain, M. Fotouhi, and R. Hasan, “Towards an analysis of security issues, challenges, and open problems in the internet of things,” in Services (SERVICES), 2015 IEEE World Congress on. IEEE, 2015, pp. 21–28.

[31] L. Da Xu, W. He, and S. Li, “Internet of things in industries: A survey,” IEEE Transactions on industrial informatics, vol. 10, no. 4, pp. 2233 - 2243, 2014.

[32] L. M. R. Tarouco, L. M. Bertholdo, L. Z. Granville, L. M. R. Arbiza, F. Carbone, M. Marotta, and J. J. C. de Santanna, “Internet of things in healthcare: Interoperability and security issues,” in Communications (ICC), IEEE International Conference on. IEEE, 2012, pp. 6121–6125.

[33] A. Mohan, “Cyber security for personal medical devices internet of things,” in Distributed Computing in Sensor Systems (DCOSS), 2014 IEEE International Conference on. IEEE, 2014, pp. 372–374.

[34] S. Yoon, H. Park, and H. S. Yoo, “Security issues on smart home in iot environment,” in Computer Science and its Applications. Springer, 2015, pp. 691–696.

[35] R. H. Weber, “Internet of things–new security and privacy challenges,” Computer law & security review, vol. 26, no. 1, pp. 23–30, 2010.

[36] S. Babar, P. Mahalle, A. Stango, N. Prasad, and R. Prasad, “Proposed security model and threat taxonomy for the internet of things (IOT),” in International Conference on Network Security and Applications. Springer, 2010, pp. 420–429.

[36] Y. H. Hwang, “Iot security & privacy: threats and challenges,” in Proceedings of the 1st ACM Workshop on IoT Privacy, Trust, and Security. ACM, 2015, pp. 1–1.

[37] M. A. Qureshi, A. Aziz, B. Ahmed, A. Khalid, and H. Munir, “Comparative analysis and implementation of efficient digital image watermarking schemes,” International Journal of Computer and Electrical Engineering, vol. 4, no. 4, p. 558, 2012.

[38] M. Abdur Razzaq, R. A. Sheikh, A. Baig, and A. Ahmad, “Digital image security: Fusion of encryption, steganography and watermarking,” International Journal of Advanced Computer Science and Applications (IJACSA), vol. 8, no. 5, 2017.

[39] S. Singh and N. Singh, “Internet of things (IOT): Security challenges, business opportunities & reference architecture for e-commerce,” in Green Computing and Internet of Things (ICGCIoT), 2015 International Conference on. IEEE, 2015, pp. 1577–1581

[40] B. Blanchet, "An efficient cryptographic protocol verifier based on prolog rules," Proceedings. 14th IEEE Computer Security Foundations Workshop, 2001. pp. 82-96, doi: 10.1109/CSFW.2001.930138.

[41] B. Blanchet, B. Smyth, and V. Cheval. ProVerif 1.88: Automatic Cryptographic Protocol Verifier, User Manual and Tutorial. INRIA, Paris, France, August 2013

[42] D. Dolev and A. Yao, "On the security of public key protocols," in IEEE Transactions on Information Theory, vol. 29, no. 2, pp. 198-208, March 1983, doi: 10.1109/TIT.1983.1056650

[43] S. Kremer, M. Ryan, and B. Smyth,, 2010, September. “Election verifiability in electronic voting protocols” In European Symposium on Research in Computer Security (pp. 389-404). Springer, Berlin, Heidelberg

[44] S. Delaine, S.Kremer, and M Ryan, 2009. “Verifying privacy-type properties of electronic voting protocols”. Journal of Computer Security, 17(4), pp.435-487

[45] M. Abadi. and B. Blanchet. “Computer-assisted verification of a protocol for certified email. Science of Computer Programming” 58(1-2), pp.3-27, in 2005

[46] M..Abadi, and N. Glew,, 2002, May. “Certified email with a light on-line trusted third party: Design and implementation” In Proceedings of the 11th international conference on World Wide Web (pp. 387-395

[47] C. Tang,., D.A. Naumann,. and S. Wetzel, 2013, November. “Analysis of authentication and key establishment in inter-generational mobile telephony”. In 2013 IEEE 10th International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing (pp. 1605-1614)

[48] M. Arapinis,., L. Mancini,., E. .Ritter, , M. Ryan, N. Golde, K. Redon and R. Borgaonkar,, 2012, October. “New privacy issues in mobile telephony: fix and verification” In Proceedings of the 2012 ACM conference on Computer and communications security (pp. 205-216)

[49] F. Van Den Broek,, R. Verdult. and J. de Ruiter,, 2015, October. “ Defeating IMSI catchers” In Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security (pp. 340-351)

[50] M.S.A Khan,. and C.J Mitchell,., 2017, July. “Trashing IMSI catchers in mobile networks” In Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks (pp. 207-218)